



Universidade do Minho

Mestrado Integrado em Engenharia Informática
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Lectivo de 2016/2017

Reservas de viagens em comboios nacionais e internacionais

**André Diogo – A75505, António Silva –
A73827, Gonçalo Pereira - A74413, Helder
Gonçalves – A64286**

Novembro, 2016

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

Reservas de viagens em comboios nacionais e internacionais

**André Diogo – A75505, António Silva –
A73827, Gonçalo Pereira - A74413, Helder
Gonçalves – A64286**

Novembro, 2016

Resumo

Este trabalho constitui todo o processo de modelação de um sistema de base de dados relacional com base na aprendizagem realizada na disciplina de Base de Dados.

Começámos por idealizar uma empresa fictícia, a Ferroviária Santos, que trabalhasse no âmbito do tema do projeto e definir o caso de estudo desde as motivações e objetivos por trás da implementação dum tal sistema até ao levantamento de requisitos para esse sistema para o caso de estudo em específico.

Tratámos então de modelar este sistema para o caso de estudo em três fases e mantendo uma forte orientação aos dados.

Em primeiro lugar, uma modelação conceptual do modelo, identificando entidades, relacionamentos e atributos relevantes.

Em segundo lugar, uma modelação lógica do modelo, identificando as relações a derivar do modelo conceptual, descrevendo-as plenamente e validando-as através da normalização e verificando as suas restrições de integridade.

Finalmente, a modelação física do modelo, que consistiu em identificar as relações base, dados derivados e como os calcular, restrições a aplicar e perfis de utilização a definir em função dos requisitos.

Durante cada etapa, validámos também cada modelo em função das principais transações desejadas pela Ferroviária Santos e em mais que uma ocasião, tivemos de recomeçar o processo iterativo para refinar melhor o nosso modelo ao ser confrontados, durante todo o processo de validação, com situações que falhámos em modelar corretamente.

Após a modelação física do modelo, gerámos a base de dados, já no sistema de gestão de base de dados escolhido (*MySQL*), definimos alguns *scripts*, primeiro para a povoar e em seguida para as transações necessárias relevantes e definimos os perfis de utilização identificados anteriormente.

Por último procedemos à análise da base de dados em termos da sua ocupação em disco e adequação aos requisitos especificados para o caso da Ferroviária Santos de modo a tirarmos as nossas conclusões finais.

Chegámos à conclusão que a nossa base de dados dada a sua simplicidade e face aos requisitos bastante simples se prova adequada, rápida e proporciona as funcionalidades pedidas.

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

Palavras-Chave: Bases de Dados Relacionais, *MySQL*, Sistemas de Bases de Dados, Gestão de Bases de Dados, Modelo relacional, Entidade, Relacionamento, Diagramas ER, Notação Chen.

Índice

1. Introdução	1
1.1. Contextualização	1
1.2. Apresentação do Caso de Estudo	1
1.3. Motivação e Objetivos	2
1.4. Estrutura do Relatório	2
2. Análise geral do projeto	3
2.1. Análise e justificação da viabilidade do projeto	3
2.2. Vantagens e ganhos operacionais	3
2.3. Levantamento de requisitos	3
2.4. Caracterização dos Perfis de Utilização	4
2.4.1 Administrador	4
2.4.2 Gerente	4
2.4.3 Utilizador	4
3. Modelo Conceptual	5
3.1. Esquema do modelo de dados conceptual	5
3.2. Identificação de Entidades	6
3.3. Identificação de Relacionamentos	7
3.4. Identificação de atributos associados às entidades e aos relacionamentos	7
3.5. Identificação de chaves candidatas, primárias e alternativas	10
3.6. Verificar possíveis redundâncias no modelo	10
3.7. Validar transações desejadas no modelo	11
4. Modelo Lógico	12
4.1. Esquema do modelo de dados lógico	13
4.2. Derivação das relações do modelo	13
4.2.1 Entidades Fortes	13
4.2.2 Relacionamentos binários de 1:N ou N:1	14
4.2.3 Relacionamentos de N:M	14
4.2.4 Atributos multivalorados	14
4.2.5 Relacionamentos binários de 1:1	14
4.2.6 Descrição das relações do modelo	15
4.3. Validar relações usando a normalização	16
4.3.1 1ª Forma Normal (1NF)	16

4.3.2 2ª Forma Normal (2NF)	16
4.3.3 3ª Forma Normal (3NF)	16
4.4. Validação das relações através das transações	17
4.5. Verificação das restrições de integridade	19
4.6. Verificar crescimento futuro	21
5. Modelo Físico	22
5.1. Implementação e conversão do modelo lógico para um SGBD	22
5.1.1 Relações base	22
5.1.2 Dados derivados	25
5.1.3 Restrições gerais	25
5.2. Transações	26
5.2.1 Mapa dos caminhos transacionais	27
5.3. Política de Segurança	28
5.3.1 Regras de acessos a dados	28
5.4. Povoamento da base de dados	29
5.5. Uso de disco e estimativa de crescimento	30
5.5.1 Uso de disco	30
5.5.2 Estimativa de crescimento da base de dados	30
6. Conclusões e Trabalho Futuro	31
7. Referências Bibliográficas	32
8. Lista de Siglas e Acrónimos	33

Índice de Figuras

Figura 1 - Modelo de dados conceptual da ferroviária Santos.	5
Figura 2 – Modelo de dados lógico da ferroviária Santos.	13
Figura 3 – Inserção de Reserva de Viagem.	17
Figura 4 – Consulta de lugares disponíveis.	18
Figura 5 – Criação de uma viagem.	18
Figura 6 - Tabela de relação cliente.	22
Figura 7 - Tabela de relação comboio.	23
Figura 8 - Tabela de relação estação.	23
Figura 9 - Tabela de relação lugares.	23
Figura 10 - Tabela de relação reserva.	24
Figura 11 - Tabela de relação tipo_comboio.	24
Figura 12 - Tabela de relação viagem.	25
Figura 13 – Gatilho verifica_comboio.	26
Figura 14 – Gatilho verifica_lugar.	26
Figura 15 - Transação para inserir uma viagem.	27
Figura 16 - Transação para fazer uma reserva.	27
Figura 17 – Transação para consultar lugares disponíveis.	27
Figura 19 - Perfil de Administrador da Base de Dados.	28
Figura 20 - Perfil de Gerente da Base de Dados.	28
Figura 21 - Perfil de Utilizador da Base de Dados.	29
Figura 22 - Povoamento da BD.	29
Figura 23 - Script de Calculo de Tamanho.	30
Figura 24 - Tamanho da BD em bytes.	30

Índice de Tabelas

Tabela 1 – Entidades identificadas no modelo de dados conceptual.	6
Tabela 2 – Relacionamentos identificados no modelo de dados conceptual.	7
Tabela 3 – Atributos associados às entidades do modelo de dados conceptual.	10
Tabela 4 – Chaves candidatas, primárias e alternativas para as entidades no modelo de dados conceptual.	10
Tabela 5 - Caminhos transacionais nas relações.	28

1. Introdução

1.1. Contextualização

Este projeto está inserido na disciplina de Bases de Dados do Mestrado Integrado em Engenharia Informática e tem como tema a reserva de viagens em comboios nacionais e internacionais. Há uns anos atrás, todo o armazenamento de dados na área ferroviária era feito em papel, assim como todas as compras de bilhetes. No entanto, com a crescente necessidade de gerir, organizar, sistematizar e ordenar a enorme quantidade de informação resultante de um sistema deste género, foi necessário informatizar estes serviços considerados rigorosamente manuais. Com este pensamento, surgiram Sistemas de Bases de Dados capazes de suportar o armazenamento deste tipo de dados. Assim, deixou de haver a necessidade de armazenamento físico e de horas de trabalho para manter a uniformidade dos dados pois, com a introdução de uma plataforma eletrónica, toda a informação é organizada e armazenada de uma forma mais simples, com custos menores. No entanto, a maior vantagem que este sistema traz é a análise dos dados. Consequentemente, torna-se mais fácil fazer uma extração de conhecimento relevante a partir desta quantidade imensa de informação.

Este Projeto enquadra-se nesta área, na medida em que é necessário desenvolver um subsistema do referido acima, focando-se este na reserva de viagens em comboios. Para tal, é necessário que se possua informação acerca de todos os troços presentes na rede ferroviária em questão, assim como de todos os veículos que fazem os percursos. Para este exemplo, um pequeno sistema que permita armazenar toda a informação necessária para gerir um sistema de reservas de comboios permite ao administrador melhor otimizar toda a sua rede de comboios.

1.2. Apresentação do Caso de Estudo

A ferroviária Santos atualmente resolve pedidos de reserva recorrendo a tickets e telefonemas das estações onde é pedida a viagem para a estação destino onde um funcionário tem de estar permanentemente disponível para decidir qual o lugar da reserva ou se o comboio

em questão já se encontra lotado. Quando as reservas são efetuadas, o lugar fica reservado até à data de pagamento.

A ferroviária Santos quer, no entanto, modernizar e livrar-se deste modelo antiquado de resolver as coisas. Para tal, querem um sistema que lhes permita comodamente tratar das reservas automaticamente, sendo agora o cliente que trata de interrogar o sistema sobre que viagens e em que datas existem e que efetua uma reserva no sistema.

1.3. Motivação e Objetivos

As motivações principais para a implementação dum sistema de bases de dados relacional são: cortar os custos com o pessoal destacado para a gestão das reservas e recursos utilizados (chamadas telefónicas e papel) e facilitar o processo de reserva de modo a que se possa escalar futuramente para uma solução web, mais acessível ao consumidor.

Os objetivos principais desta implementação são: automatizar o processo de gestão das reservas de modo a envolver o mínimo de esforço por parte dos funcionários e oferecer um sistema de gestão de reservas rápido e robusto.

1.4. Estrutura do Relatório

Seguir-se-á, por ordem, um capítulo dedicado à análise geral do projeto de implementação de uma base de dados relacional: uma análise e justificação da viabilidade do projeto, apresentação de vantagens e ganhos operacionais atingíveis com a implementação desta mesma base dados, um plano para o levantamento de requisitos e um plano para a caracterização dos diversos perfis de utilização seguido de um três capítulos dedicado à implementação deste projeto: a explicitação do funcionamento da base de dados, de forma pormenorizada, desde a conceção à manutenção para cada um dos seus respetivos esquemas (conceptual, lógico e físico) e finalmente um capítulo dedicado a considerações e análise crítica do projeto, de acordo com a metodologia especificada em Connolly, T., Begg, C., 2004 *Database Systems, A Practical Approach to Design, Implementation, and Management*. 4th ed. Boston: Addison-Wesley.

2. Análise geral do projeto

2.1. Análise e justificação da viabilidade do projeto

Neste projeto a ferroviária disponibiliza um montante considerável de fundos para treino e contratação de funcionários capazes de utilizar e administrar a base de dados pelo que se justifica a implementação duma base de dados relacional como medida eficaz de atingir os objetivos pretendidos.

2.2. Vantagens e ganhos operacionais

Contando, após diálogo com os responsáveis nesta ferroviária, com um volume de dados modesto (na ordem dos milhares de entradas de reservas anuais) e um modelo de negócio relativamente simples, torna-se claro que o modelo relacional neste caso é capaz de resolver os requisitos, a seguir apresentados, deste sistema de reservas, de encontro aos objetivos pretendidos, pelo que veremos, em princípio grandes ganhos operacionais.

2.3. Levantamento de requisitos

Após contacto com os responsáveis nesta ferroviária, compilamos a seguinte lista de requisitos para este sistema:

- Um cliente deve possuir um nome de utilizador único, uma password e pelo menos um email associado.
- Um cliente pode efetuar múltiplas reservas sobre múltiplas viagens.
- Uma reserva deve indicar a data da reserva, o preço da reserva, a data de partida e chegada, a estação de partida e chegada e o lugar no comboio, composto por carruagem e lugar.
- O preço de uma reserva é calculado em função da viagem em questão e do tipo de comboio reservado, este último que incorre uma taxa extra sobre o valor base.

- Uma viagem tem de ter uma data de partida e chegada, estação de partida e chegada e um preço base e são efetuadas por um só comboio.
- Múltiplas viagens podem ser efetuadas por um comboio e ter a si associadas múltiplas reservas.
- Cada comboio tem um conjunto de carruagens e lugares a si associados e um tipo, com uma taxa associada a aplicar ao preço base da viagem.

2.4. Caracterização dos Perfis de Utilização

Identificámos dois perfis de utilização para este projeto, o de utilizador regular e do gerente havendo, é claro, ainda o caso do administrador da base de dados que se encontra sempre subjacente.

2.4.1 Administrador

O administrador da base de dados possuirá acesso total a todas as facetas da base de dados.

2.4.2 Gerente

O gerente utilizará a base de dados para inserir viagens e comboios, consultar viagens e reservas de viagens.

2.4.3 Utilizador

O utilizador utilizará a base de dados para consultar viagens para reservar e que já reservou, incluindo em que comboio, em que datas, para que origens e destinos, consultar preços de viagens e efetuar reservas.

3. Modelo Conceptual

Após o levantamento dos requisitos, passámos a traduzi-los para um modelo de dados conceptual, que visa traduzir estes requisitos em entidades envolvidas, os seus atributos e os seus relacionamentos dos quais queremos tirar partido e que queremos modelar. Além disso, queremos estabelecer um domínio para esses atributos, possíveis chaves candidatas e algumas restrições de integridade. Utilizámos para esquematização deste modelo a ferramenta TerraER, gratuita e algo limitada, mas útil para visualizar o modelo.

3.1. Esquema do modelo de dados conceptual

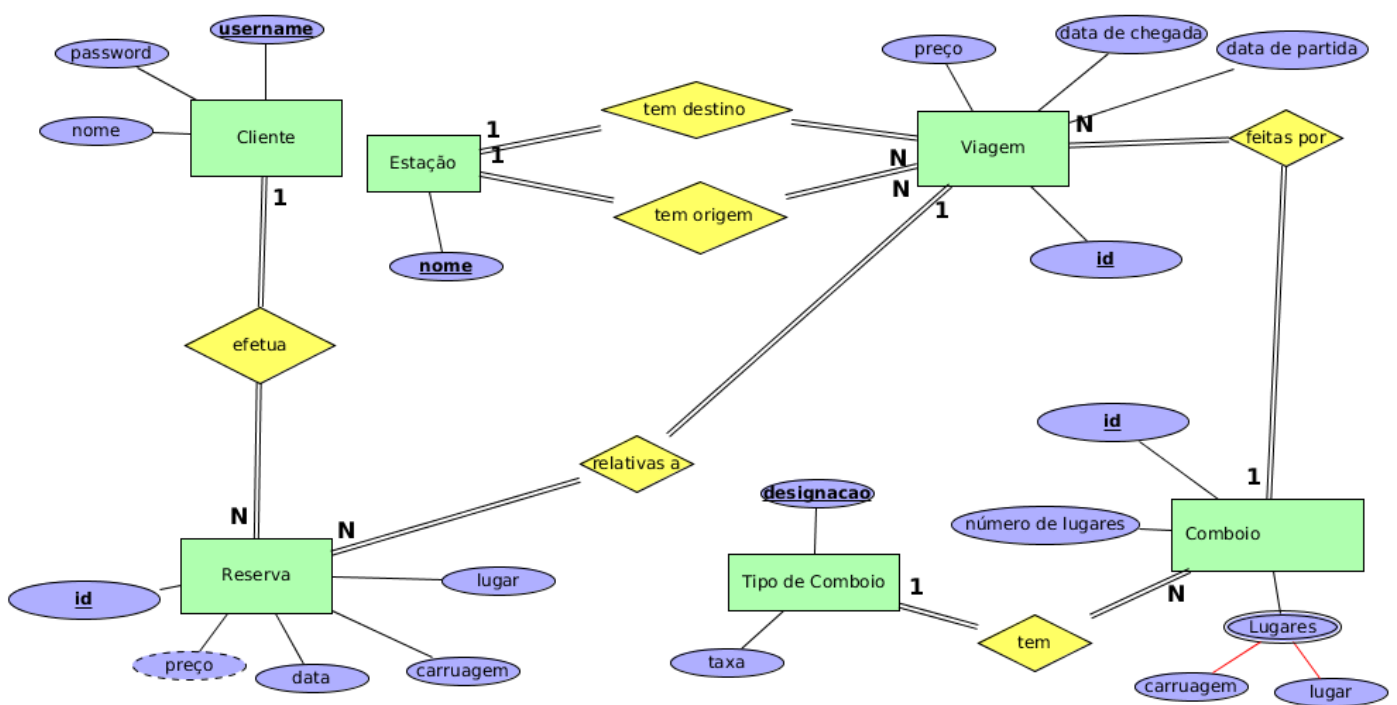


Figura 1 - Modelo de dados conceptual da ferroviária Santos.

3.2. Identificação de Entidades

Com base nos requisitos, identificámos alguns possíveis candidatos a entidades de entre os substantivos referidos com mais significância, quer por terem uma existência real independente uns dos outros e/ou por possuírem dados relevantes que queremos modelar a si associados, como é o caso do **comboio**, que possui tipo e lugares e existência real, **viagem**, que necessita de estações e datas de partida e chegada e que nos interessa modelar, **reserva**, que tem preços e o principal objetivo de modelação, **estação**, que tem também uma existência real física e nome, **cliente**, possui também as suas informações e existência real física, e **tipo de comboio**, em que cada comboio há de ter um por virtude de usar certa tecnologia ou certo tipo de motor e visto que muitos comboios podem ter o mesmo tipo tem uma existência independente e uma taxa associada.

Apresentamos então, em forma tabelar, as entidades que identificámos neste modelo:

Nome	Descrição	Sinónimos	Ocorrência
Cliente	Termo que descreve as pessoas que efetuam reservas e que se registam no sistema para tal.	-	Cada pessoa se regista no sistema e tem a si associadas um conjunto de reservas em viagens.
Reserva	Termo que descreve um bilhete de comboio.	Bilhete	Cada reserva tem um só cliente que a efetua e está associada a uma só viagem.
Viagem	Termo que descreve um itinerário percorrido por um comboio.	Itinerário, Percurso	Cada viagem representa um percurso de uma estação origem a uma estação destino efetuado por um comboio.
Comboio	Termo que descreve um comboio.	-	Cada comboio tem um tipo e efetua viagens.
Estação	Termo que descreve um local onde param comboios.	Origem, Destino, Local	Cada estação está envolvida em viagens como origem ou destino, mas não ambas.
Tipo de Comboio	Termo que descreve o tipo de um comboio.	-	Cada comboio tem de ter a si associado um tipo de comboio.

Tabela 1 – Entidades identificadas no modelo de dados conceptual.

3.3. Identificação de Relacionamentos

Com base nos requisitos, identificámos alguns relacionamentos entre estas entidades escolhidas: visto nos ser dito que o cliente efetua múltiplas reservas, identificámos um relacionamento claro entre eles de multiplicidade 1:N tal que **Cliente efetua N Reserva**.

É-nos dito também que múltiplas reservas se podem referir a uma viagem, pelo que identificámos um relacionamento claro entre estas duas entidades de multiplicidade N:1, tal que **N Reserva são relativas a 1 Viagem**.

Estão implícitos também três relacionamentos da entidade viagem para as entidades comboio e estação porque nos dizem que muitas viagens podem ser efetuadas pelo mesmo comboio, pelo que identificámos aqui um relacionamento de N:1, tal que **N Viagem são feitas por 1 Comboio**, além de que também figura nos requisitos que cada viagem tem uma estação de partida e uma de chegada, pelo que identificámos aqui os dois outros relacionamentos de N:1, tal que **N Viagem tem origem em 1 Estação** e **N Viagem tem como destino 1 Estação**.

Finalmente, está estipulado nos requisitos que cada comboio tem o seu tipo com características próprias, pelo que identificámos aqui um relacionamento de N:1, tal que **N Comboio têm 1 Tipo**.

Apresentamos então, em forma tabelar, os relacionamentos identificados neste modelo:

Nome da entidade	Multiplicidade	Relacionamento	Multiplicidade	Nome da entidade
Cliente	1..1	Efetua	1..N	Reserva
Reserva	1..N	Efetua	1..1	Cliente
	1..N	Relativas a	1	Viagem
Viagem	1	Relativas a	1..N	Reserva
	1..N	Tem origem	1	Estação
	1..N	Tem destino	1	Estação
	1..N	Feitas por	1	Comboio
Comboio	1..1	Feitas por	1..N	Viagem
	1..N	Tem	1..1	Tipo de Comboio
Estação	1..1	Tem origem	1..N	Viagem
	1..1	Tem destino	1..N	Viagem
Tipo de Comboio	1..1	Tem	1..N	Comboio

Tabela 2 – Relacionamentos identificados no modelo de dados conceptual.

3.4. Identificação de atributos associados às entidades e aos relacionamentos

Para cada entidade, pelos requisitos, documentámos os seus atributos.

Para o **Cliente** estava estipulado ter um *username*, password e um nome.

Para a **Reserva** estava estipulado ela ter um preço, uma data, uma viagem e um cliente associados. No entanto, decidimos que era necessário ela possuir um identificador único, e visto haver relacionamentos com a viagem e o cliente, era desnecessário estes serem atributos.

Para a **Viagem**, nos requisitos disseram-nos que teria datas e estações de partida e chegada, preço e um comboio. No entanto, decidimos que era necessário ela possuir também um identificador único, e visto já haver relacionamentos com o comboio e a estação, era desnecessário estes serem atributos.

Para o **Comboio**, nos requisitos referenciava um conjunto de carruagens e lugares e um tipo associado. Foi, porém, necessário aqui também dar um identificador único a esta entidade além de que, visto estarmos perante um conjunto de carruagens e lugares, de criar um atributo multivalorado composto **Lugares**, tal que este contém muitas carruagens e lugares, visto que existe uma clara composição aqui, de modo que lugares pertencem a carruagens que pertencem ao comboio. Descartámos mais uma vez devido à existência do relacionamento com o **Tipo de Comboio** o possível atributo equivalente.

Para a entidade **Tipo de Comboio**, reparámos que necessitaria, pelo que figura nos requisitos, de uma taxa a si associada. Decidimos definir o nome como uma abreviatura (devido à necessidade de apresentar ao utilizador um nome que reconheça, apesar de não estar explícito nos requisitos que requer um nome) apenas.

Finalmente, para a entidade **Estação** decidimos ter apenas também um nome abreviado, de acordo com a necessidade de apresentar ao utilizador um nome que reconheça, apesar de não estar explícito nos requisitos que requer um nome.

Apresentamos em forma tabelar os atributos associados às entidades seguidos dos atributos associados aos relacionamentos com as suas características relevantes:

Nome da Entidade	Atributos	Descrição	Domínio dos Atributos	Nulo	Multivalorado	Derivado
Cliente	<i>username</i>	Identifica o cliente	40 carateres variáveis	Não	Não	Não
	<i>password</i>	Verifica a identidade do cliente	40 carateres variáveis	Não	Não	Não
	<i>nome</i>	O nome do cliente	100 carateres variáveis	Não	Não	Não
Reserva	<i>id</i>	Número da reserva	Inteiro	Não	Não	Não

	preço		Preço da reserva	Número com 2 casas decimais.	Não	Não	Calculado através do preço da viagem multiplicado pela taxa do comboio
	data		Data da reserva	Data	Não	Não	Não
	carruagem		Número da carruagem reservada na viagem	Inteiro	Não	Não	Não
	lugar		Número do lugar reservada na viagem	Inteiro	Não	Não	Não
Viagem	id		Número da viagem	Inteiro	Não	Não	Não
	preço		Preço da viagem	Número com 2 casas decimais	Não	Não	Não
	Data de partida		-	Data	Não	Não	Não
	Data de chegada		-	Data	Não	Não	Não
Comboio	id		Número do comboio	Inteiro	Não	Não	Não
	Lugares	lugar	Número do lugar	Inteiro	Não	Sim	Não
		carruagem	Número da carruagem	Inteiro			
Estação	nome		Nome da estação	10 carateres variáveis	Não	Não	Não
Tipo de comboio	designação		Designação do tipo do comboio	4 carateres variáveis	Não	Não	Não
	taxa		Taxa associada	Número com 2	Não	Não	Não

		ao tipo de comboio que multiplica o preço da viagem	casas decimais.			
--	--	---	-----------------	--	--	--

Tabela 3 – Atributos associados às entidades do modelo de dados conceptual.

3.5. Identificação de chaves candidatas, primárias e alternativas

Nesta fase do desenho do modelo de dados conceptual, como optámos por incluir atributos inteiros com um identificador na maior parte destas entidades por não possuírem nenhuma chave candidata, quase todas as chaves primárias são os atributos **id** das diversas entidades como se pode verificar na tabela que se segue:

Nome da entidade	Chaves Candidatas	Chave primária	Chaves alternativas	Entidade Fraca
Cliente	<i>username</i>	<i>username</i>	-	Não
Reserva	id	id	data	Não
	data			
Viagem	id	id	-	Não
Estação	nome	nome	-	Não
Comboio	id	id	-	Não
Tipo de Comboio	designação	designação	-	Não

Tabela 4 – Chaves candidatas, primárias e alternativas para as entidades no modelo de dados conceptual.

3.6. Verificar possíveis redundâncias no modelo

No nosso modelo, como não existem relações de 1:1 nem caminhos redundantes entre entidades podemos afirmar que não existem redundâncias pois além destes fatores, o fator tempo não altera fundamentalmente nenhum relacionamento ou entidade.

Existe apenas um caso capaz de suscitar dúvidas quanto a possíveis redundância, que são os dois relacionamentos entre viagem e estação que são claramente distintos, visto que uma estação de partida é diferente da de chegada.

3.7. Validar transações desejadas no modelo

As principais transações desejadas para a eventual base de dados são:

- a) **Reservar uma viagem:** Para um cliente reservar uma viagem, é necessário partir da 'Reserva' e percorrer a relação 'efetua' até ao cliente, de modo a obter o nome do cliente. É necessário também percorrer a relação 'relativa a' até 'Viagem' para obter o preço. A entidade reserva contém todas as outras informações relevantes.
- b) **Consultar lugares num comboio:** Para consultar lugares num comboio, vai-se de 'Viagem' e percorre-se a relação 'relativa a' até 'Reserva', de modo a obter-se os lugares reservados e respetivas carruagens, e de 'Viagem' até 'Comboio' pela relação 'feitas por' de modo a retirar apenas os lugares relativos ao comboio pedido.
- c) **Criação de uma viagem:** Para criar uma viagem, parte-se de 'Viagem', percorrendo até 'Estação' pelas relações 'tem destino' e 'tem origem', de modo a especificar a origem e o destino. Vai-se também de 'Viagem' até 'Comboio' pela relação 'feitas por', adicionando um comboio à viagem, a entidade viagem contém todas as outras informações relevantes.

Todas estas transações são, portanto, possíveis e válidas.

4. Modelo Lógico

Nesta etapa do processo de desenho de uma base de dados convertemos o esquema conceptual acima para um modelo lógico de dados, e para isso de acordo com a metodologia que seguimos, transformámos as entidades e relacionamentos acima descritos em relações/tabelas em que os atributos já aparecem com tipos concretos. Construámos o modelo lógico na ferramenta *MySQL Workbench*, e dadas as suas pequenas dimensões não há separação do modelo em vistas e como tal será apresentado na íntegra.

4.1. Esquema do modelo de dados lógico

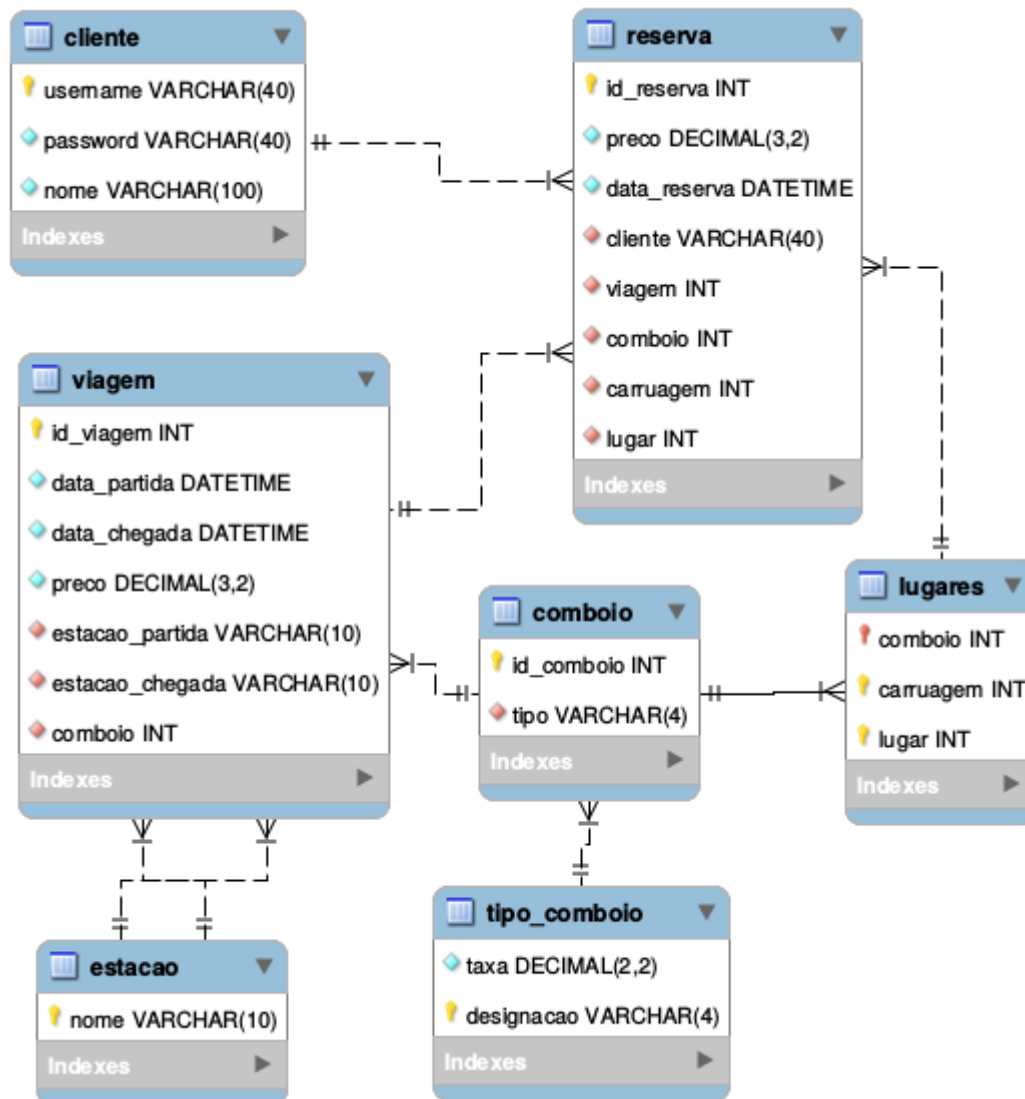


Figura 2 – Modelo de dados lógico da ferroviária Santos.

4.2. Derivação das relações do modelo

Nesta seção vamos proceder a explicitar o processo de conversão do modelo conceitual para o modelo lógico e finalmente descrever todas as relações a que chegamos através do uso da *Database Definition Language (DBDL)*

4.2.1 Entidades Fortes

Identificámos as entidades Cliente, Reserva, Viagem, Comboio, Tipo de Comboio e Estação durante a elaboração do modelo conceptual. Cada uma destas terá a sua relação

correspondente (**cliente**, **reserva**, **viagem**, **comboio**, **tipo_comboio** e **estacao**, respetivamente) com todos os atributos simples já especificados.

4.2.2 Relacionamentos binários de 1:N ou N:1

Identificámos relacionamentos binários de 1:N entre as entidades Cliente e Reserva, Viagem e Comboio, Viagem e Estação e Comboio e Tipo de Comboio e o caso do relacionamento entre Viagem e Reserva. Como tal, e observando as multiplicidades de cada respetiva entidade no relacionamento, para cada um destes relacionamentos, identificámos Cliente como entidade pai, Comboio como entidade pai, Estação como entidade pai, Tipo de Comboio como entidade pai e por último Reserva como entidade pai, respetivamente.

Assim, a relação **reserva** guarda uma cópia da chave primária da relação **cliente** como chave estrangeira e uma cópia da chave primária da relação **viagem** como chave estrangeira. A relação **viagem** guarda duas cópias de duas chaves primárias da relação **estacao** (dá-se o caso que existem dois relacionamentos de Viagem para Estação de N:1) e guarda uma cópia da chave primária da relação **comboio** também como chave estrangeira. Finalmente, a relação **comboio** guarda uma cópia da chave primária da relação **tipo_comboio** como chave estrangeira.

4.2.3 Relacionamentos de N:M

Não existem relacionamentos de N para M no nosso modelo.

4.2.4 Atributos multivalorados

Existe também apenas um caso de atributo multivalorado no modelo conceptual, que é o caso do atributo Lugares, composto por carruagem e lugar. Deste atributo derivamos uma nova relação **lugares** que recebe a chave primária como chave estrangeira da relação **comboio** derivada da entidade a que pertencia originalmente (Comboio). Torna-se, por conseguinte necessário para identificar de forma única o conjunto Lugares, visto que é composto, que ambas os atributos simples que o compõem passem a constituir também a chave primária juntamente com esta chave estrangeira recebida da relação **comboio**.

Tal é a derivação dado que só podemos identificar com unicidade um lugar se identificarmos os três unicamente: lugar, a sua carruagem e, por conseguinte, o seu comboio.

4.2.5 Relacionamentos binários de 1:1

Neste ponto da modelação reparámos numa dependência não representada no modelo conceptual, devido a limitações deste. O lugar e carruagem que figuram na relação **reserva** na verdade são aqueles estipulados pela relação **lugares** para o dado comboio na viagem. Para garantirmos que existe integridade referencial, e que não podemos ter reservas com lugares

inexistentes, há que importar a chave primária (que contém os lugares e carruagens necessários para a reserva) desta relação para a relação **reserva** como chave estrangeira para nos assegurarmos que um lugar só pode ser reservado uma vez numa viagem, como queríamos modelar originalmente, pelo que tínhamos implícito um relacionamento de 1:1 entre **reserva** e **lugares**.

4.2.6 Descrição das relações do modelo

Com base nas derivações que fizemos gerámos então estas descrições das relações derivadas:

- **cliente** (username, password, nome)
 - **Chave primária** (PK): username

- **reserva** (id_reserva, preço, data_reserva, cliente, viagem, comboio, carruagem, lugar)
 - **Chave primária** (PK): id_reserva
 - **Chave estrangeira** (FK): cliente **referencia** cliente(username)
viagem **referencia** viagem(id_viagem)
comboio **referencia** lugares(comboio)
carruagem **referencia** lugares(carruagem)
lugar **referencia** lugares(lugar)
 - **Chave alternativa** (AK): data_reserva

- **viagem** (id_viagem, data_partida, data_chegada, preco, estacao_partida, estacao_chegada, comboio)
 - **Chave primária** (PK): id_viagem
 - **Chave estrangeira** (FK): estacao_partida **referencia** estacao(id_estacao)
estacao_chegada **referencia** estacao(id_estacao)
comboio **referencia** comboio(id_comboio)

- **comboio** (id_comboio, id_tipo)
 - **Chave primária** (PK): id_comboio
 - **Chave estrangeira** (FK): id_tipo **referencia** tipo_comboio(id_tipo)

- **estacao** (nome)
 - **Chave primária** (PK): nome

- **tipo_comboio** (taxa, designacao)

- **Chave primária (PK):** designacao
- **lugares** (comboio, carruagem, lugar)
 - **Chave primária (PK):** comboio, carruagem, lugar
 - **Chave estrangeira (FK):** comboio

4.3. Validar relações usando a normalização

Nesta secção iremos mostrar que o nosso modelo lógico está normalizado até à 3ª forma normal (3NF). Para o efeito analisámos previamente o nosso modelo quanto à necessidade de existência de cada atributo para nos certificarmos de que tínhamos o mínimo de redundância possível respeitando os requisitos do sistema.

4.3.1 1ª Forma Normal (1NF)

A primeira forma normal requer verificar que a interseção de qualquer linha e coluna de uma tabela contem apenas um valor. Por virtude de decompormos toda os dados originais nos requisitos em atributos simples, nunca aparece mais que um valor em qualquer entrada de qualquer tabela neste modelo.

Assim podemos verificar que o modelo se encontra normalizado até à 1NF.

4.3.2 2ª Forma Normal (2NF)

A segunda forma normal requer verificar que todas as tabelas se encontram na 1NF e que todas os atributos que nela figuram são completamente funcionalmente dependentes da chave primária. Para tal só é necessário verificar as tabelas que possuem mais que um atributo para compor a chave primária. Existem no modelo apenas 1 tal caso, a relação **lugares**.

No caso da relação **lugares** ela é composta unicamente pela chave primária, pelo que verifica a 2NF.

Assim podemos verificar que o modelo se encontra normalizado até à 2NF.

4.3.3 3ª Forma Normal (3NF)

A terceira forma normal requer verificar que todas as tabelas se encontram na 2NF e que não existem dependências transitivas entre a chave primária e atributos que não lhe pertencem. Todo modelo se encontra desprovido de dependências transitivas, as dependências são todas dependências funcionais completas, pelo que podemos verificar que o modelo se encontra normalizado até à 3NF.

4.4. Validação das relações através das transações

As relações apresentadas no modelo lógico requerem uma validação, sendo que a sua comprovação será efetuada pela verificação das transações no mesmo modelo. Como tal, tais transações são enunciadas pelo seguinte:

a) Reservar uma viagem:

De modo a que seja efetuada a reserva de uma viagem, é assumido o pré-conhecimento do cliente e da viagem, assim como a existência de um lugar válido, disponível numa carruagem, num comboio de uma viagem em específico. Deste modo, partimos com tais informações da reserva e inserimos a reserva diretamente. A transação está enunciada pela seguinte imagem da Figura 3.

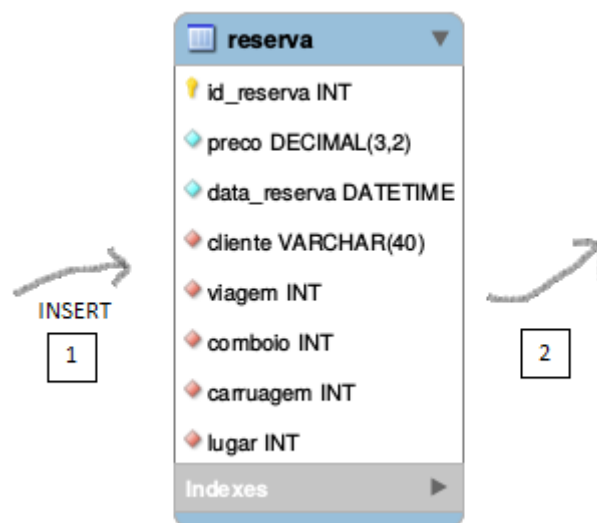


Figura 3 – Inserção de Reserva de Viagem.

b) Consultar lugares num comboio:

Para realizarmos uma consulta de lugares disponíveis num comboio, é preciso saber à partida qual a viagem em questão. Começamos por procurar o comboio responsável por tal viagem, para encontrarmos os lugares desse mesmo comboio. Por fim, são filtrados apenas os lugares disponíveis, resultado da disjunção do conjunto de lugares do comboio e os presentes na relação reserva associados à viagem em questão. Tal transação é dada pela Figura 4.

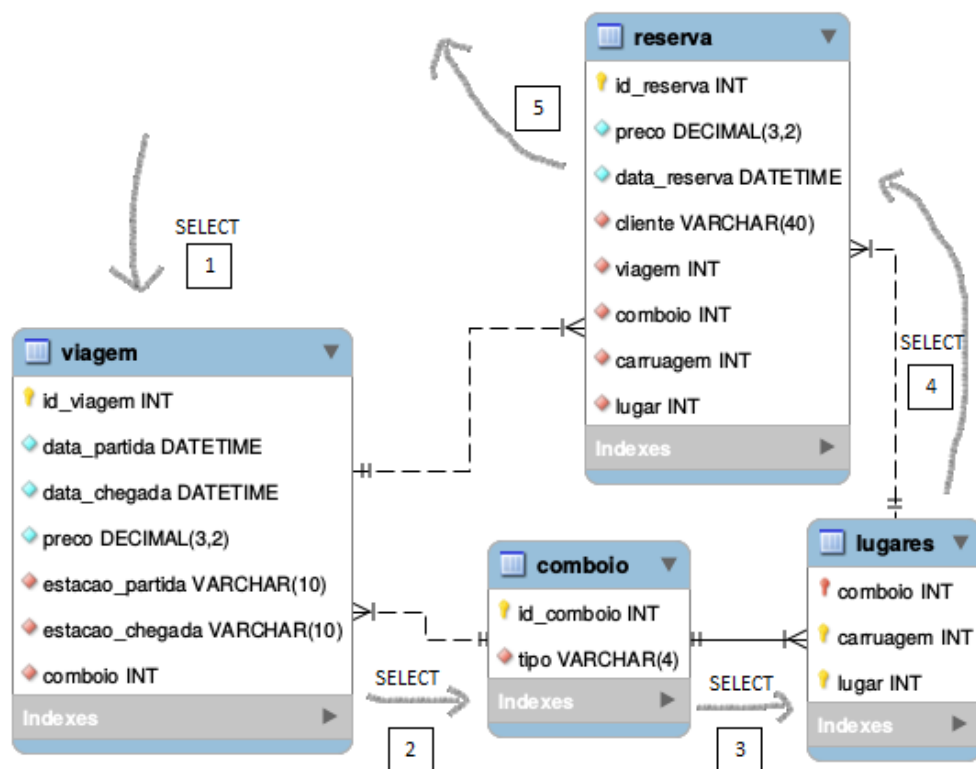


Figura 4 – Consulta de lugares disponíveis.

c) Criação de uma viagem:

Aquando da criação de uma viagem, é apenas requerida a inserção direta da tal viagem, sendo que todos os outros parâmetros são já assumidos, à partida. Deste modo, é possível retratar esta transação pela Figura 5.

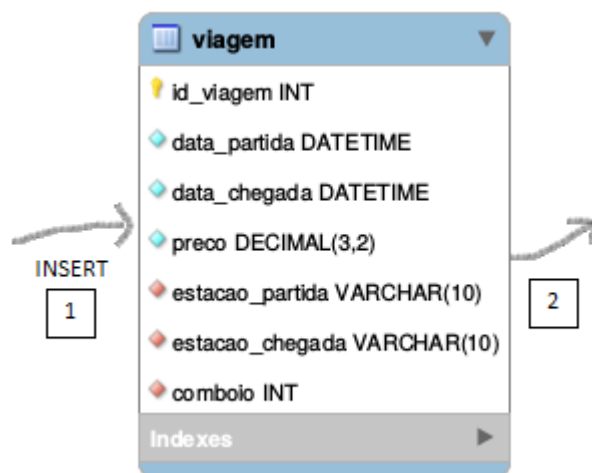


Figura 5 – Criação de uma viagem.

4.5. Verificação das restrições de integridade

a) Integridade de Entidades:

No momento de identificação dos atributos para cada entidade no modelo conceptual, definimos todos como não podendo ser nulos pelo que é impossível acontecer uma chave primária ter um valor nulo. A derivação para o modelo lógico resultou de seguir essas indicações e como tal é impossível de acontecer uma chave primária ter valor nulo.

b) Integridade Referencial:

1. **Chaves estrangeiras nulas:** Não são permitidas chaves estrangeiras nulas, pois a participação da entidade filha no relacionamento é obrigatória

2. Restrições existenciais:

2.1. **Remoção do tuplo da chave primária:** Não é possível a chave da entidade pai ser eliminada sem antes se eliminar todos os tuplos filho que façam referência à mesma, correspondendo à opção "NO ACTION". Deste modo, não será possível remover informação sobre viagens ou reservas sem remover toda a informação dependente.

2.2. **Modificação da chave primária:** No caso da existência de uma modificação no valor da chave da relação pai temos apenas um caso (*username* do cliente) em que o comportamento é "CASCADE", ou seja, a modificação propaga-se a todas as chaves estrangeiras que referenciem a mesma. Desta forma, o cliente poderá modificar o seu *username* caso deseje. Para todas as outras chaves, o comportamento é "NO ACTION" pois, caso contrário, implicaria que seria possível a modificação de informações de viagens e reservas.

c) Restrições Gerais:

Existem restrições que são específicas à nossa base de dados:

- O número de reservas para uma viagem não pode ser superior ao número de lugares no comboio que a faz.
- O comboio presente na reserva tem de ser o mesmo do da respetiva viagem.
- A data de reserva tem de corresponder a um momento antes da data de partida da viagem em questão.
- Para uma viagem, a data de partida tem de corresponder a um momento antes da data de chegada.

- Um lugar reservado numa viagem deve ser sempre único para a dada viagem.

d) Restrições de Domínio:

Ao criar o modelo lógico, seguimos o dicionário de dados do modelo conceptual, logo as relações respeitam as nossas restrições de domínio. Convém identificar o caso específico da relação **lugares** derivada, em que temos sempre subjacente um determinado comboio, dado ter sido derivada dum atributo multivalorado, e como tal possui uma chave estrangeira para comboio, para nos assegurarmos que não só o comboio é válido e consta da relação **comboio**, mas que seja possível identificar também a que comboio pertencem os lugares.

e) Restrições de Multiplicidade:

- Uma viagem precisa de duas estações (chegada e partida), logo possui duas chaves estrangeiras para estação, de modo a garantir que as estações são obrigatoriamente válidas e duas das que já existem na base de dados quando se insere uma nova viagem, respeitando os relacionamentos originais N:1.
- Uma viagem é feita por um comboio, logo possui uma chave estrangeira para comboio, deste modo qualquer viagem possui obrigatoriamente um dos comboios que consta na relação **comboio**, respeitando o relacionamento original N:1.
- Um comboio possui apenas um tipo, logo tem uma chave estrangeira para tipo de comboio, de modo a verificar o requisito que cada comboio possui um tipo válido dos já registados na relação **tipo_comboio**, respeitando o relacionamento original N:1.
- Uma reserva necessita de um identificador do cliente, logo possui uma chave estrangeira para cliente, de modo que qualquer reserva contém um cliente do domínio de clientes conhecidos na relação **cliente**, o que respeita o relacionamento original N:1.
- Uma reserva necessita de um id de viagem, logo possui uma chave estrangeira para viagem, assim, asseguramos que uma reserva é relativa a uma viagem do domínio de viagens conhecido pela base de dados presentes na relação **viagem**, respeitando o relacionamento original N:1.
- Uma reserva necessita de um lugar, uma carruagem e um id de comboio, logo possui três chaves estrangeiras para lugares, de tal modo que, uma reserva possui um lugar válido presente na base de dados e pertencente a um comboio válido, como era requisito original do sistema.

Todas estas restrições de multiplicidade são respeitadas.

4.6. Verificar crescimento futuro

Embora avistemos neste modelo bastantes possibilidades de expansão, passando possivelmente pela adição de classes às carruagens, funcionários que gerissem reservas também ou passar a acomodar venda de bilhetes na hora juntamente com o atual processo de reserva, cremos que o modelo relacional falha na representação fácil de um sistema mais complexo ferroviário, visto que se tentássemos modelar algo mais próximo da realidade no que toca a percursos de comboio, rapidamente nos depararíamos com problemas complexos de grafos para os quais o número e diversidade de relacionamentos entre nodos seria astronómico.

Vemos então sérias limitações ao crescimento futuro desta base de dados na vertente da modelação dos percursos mas bastante margem e adaptabilidade na vertente de negócio.

5. Modelo Físico

No modelo físico, todas as preocupações com índices e organização de ficheiros forma deixadas segundo os valores de defeito gerados pelo *MySQL Workbench*, não foram gerados índices adicionais e o motor usado para cada tabela foi uniformemente o *InnoDB*.

5.1. Implementação e conversão do modelo lógico para um SGBD

5.1.1 Relações base

Em seguida demonstraremos as relações base direto na sua implementação através das instruções de criação de tabelas em *MySQL* que constam no script de criação da base de dados, pois sendo o modelo extremamente simples, apenas traduzimos de forma completamente direta as relações lógicas, elas próprias modeladas em *MySQL Workbench*, diretamente vocacionado para os tipos base do motor escolhido:

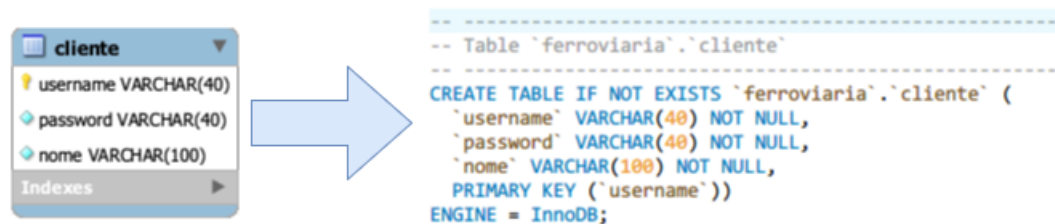


Figura 6 - Tabela de relação cliente.

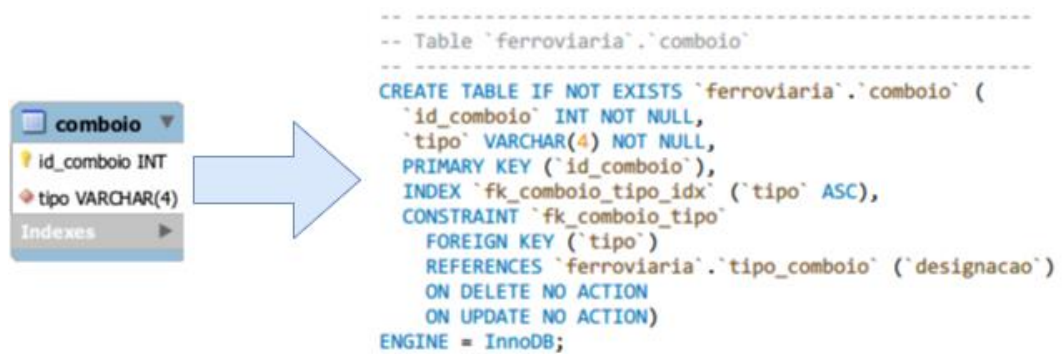


Figura 7 - Tabela de relação comboio.

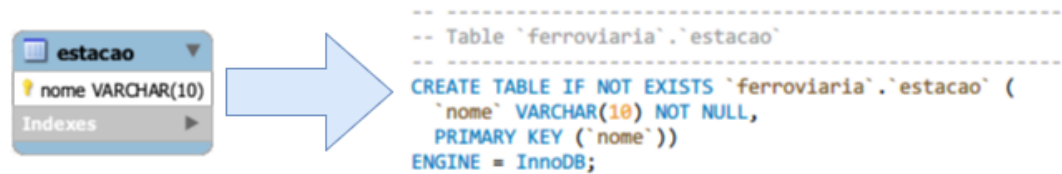


Figura 8 - Tabela de relação estação.

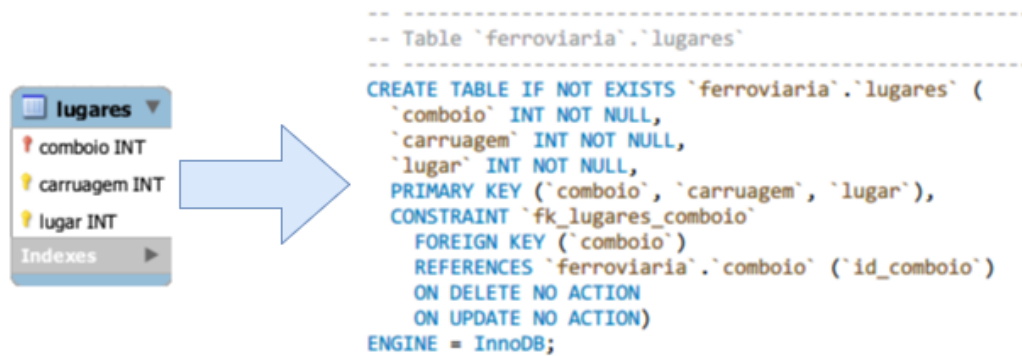


Figura 9 - Tabela de relação lugares.



Figura 10 - Tabela de relação reserva.

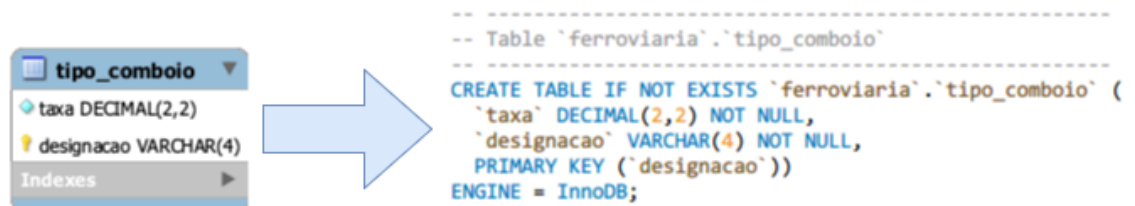
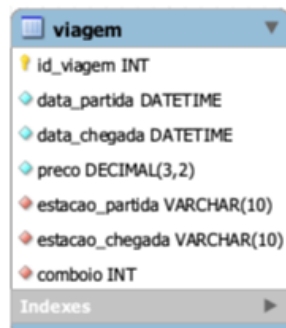


Figura 11 - Tabela de relação tipo_comboio.



viagem	
id_viagem	INT
data_partida	DATETIME
data_chegada	DATETIME
preco	DECIMAL(3,2)
estacao_partida	VARCHAR(10)
estacao_chegada	VARCHAR(10)
comboio	INT
Indexes	

```
-- Table `ferroviaria`.`viagem`
CREATE TABLE IF NOT EXISTS `ferroviaria`.`viagem` (
  `id_viagem` INT NOT NULL,
  `data_partida` DATETIME NOT NULL,
  `data_chegada` DATETIME NOT NULL,
  `preco` DECIMAL(3,2) NOT NULL,
  `estacao_partida` VARCHAR(10) NOT NULL,
  `estacao_chegada` VARCHAR(10) NOT NULL,
  `comboio` INT NOT NULL,
  PRIMARY KEY (`id_viagem`),
  INDEX `fk_comboio_idx` (`comboio` ASC),
  INDEX `fk_estacao_partida_idx` (`estacao_partida` ASC),
  INDEX `fk_estacao_chegada_idx` (`estacao_chegada` ASC),
  CONSTRAINT `fk_viagem_comboio`
    FOREIGN KEY (`comboio`)
    REFERENCES `ferroviaria`.`comboio` (`id_comboio`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_estacao_partida`
    FOREIGN KEY (`estacao_partida`)
    REFERENCES `ferroviaria`.`estacao` (`nome`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_estacao_chegada`
    FOREIGN KEY (`estacao_chegada`)
    REFERENCES `ferroviaria`.`estacao` (`nome`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Figura 12 - Tabela de relação viagem.

5.1.2 Dados derivados

Temos apenas um dado derivado, o preço de uma reserva que é calculado em função do tipo do comboio e do preço da viagem em causa. Este é calculado e inserido na reserva quando executamos o procedimento para inserção numa reserva, associado à mesma transação, pelo que aparecerá no capítulo referente às transações. Como agora no modelo físico temos de tratar estes dados derivados, esta mesma transação vai necessitar um acréscimo de consultas, em relação aquilo que foi validado para o modelo lógico.

5.1.3 Restrições gerais

Neste projeto foi necessário criar duas restrições gerais. A primeira restrição indica que um comboio numa reserva tem de ser o mesmo que faz a viagem nessa mesma reserva enquanto a segunda garante que um lugar é apenas reservado uma vez para uma viagem. Optámos por deixar ao cargo do gerente da base de dados de respeitar as outras duas restrições definidas acima durante a modelação lógica. Assim, foram desenvolvidos os seguintes gatilhos para resolver estas questões de forma transparente ao utilizador em código SQL:

```

-----
-- Restrição que garante que um comboio numa reserva é o mesmo
-- que o comboio na viagem dessa reserva
DELIMITER |
CREATE TRIGGER verifica_comboio
BEFORE INSERT ON reserva
FOR EACH ROW
BEGIN
    IF (NEW.comboio != (SELECT comboio
                        FROM viagem
                        WHERE NEW.viagem = viagem.id_viagem))

    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Comboio inválido';
    END IF;
END
|
DELIMITER ;
-----

```

Figura 13 – Gatilho verifica_comboio.

```

-----
-- Restrição que garante que um lugar é apenas reservado
-- uma vez para uma viagem
DELIMITER |
CREATE TRIGGER verifica_lugar
BEFORE INSERT ON reserva
FOR EACH ROW
BEGIN
    IF (1 = (SELECT COUNT(lugar) FROM reserva WHERE NEW.viagem = reserva.viagem AND
                                                    NEW.comboio = reserva.comboio AND
                                                    NEW.carruagem = reserva.carruagem AND
                                                    NEW.lugar = reserva.lugar))

    THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Lugar Ocupado';
    END IF;
END
|
DELIMITER ;
-----

```

Figura 14 – Gatilho verifica_lugar.

5.2. Transações

Nesta secção apresenta-se a implementação em SQL das transações especificadas na secção 3.7:

```

-----
-- Transação para inserir uma viagem
DELIMITER |
CREATE PROCEDURE insere_viagem (IN Id_viagem INT, IN Data_partida DATETIME, IN Data_chegada DATETIME, IN PRECO DECIMAL(3, 2),
                               IN Estacao_partida VARCHAR(10), IN Estacao_chegada VARCHAR(10), IN Comboio INT)
BEGIN
    DECLARE Erro BOOL DEFAULT 0;
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET Erro = 1;
    START TRANSACTION;
        INSERT INTO viagem (id_viagem, data_partida, data_chegada, preco, estacao_partida, estacao_chegada, comboio)
            VALUES (Id_viagem, Data_partida, Data_chegada, Preco, Estacao_partida, Estacao_chegada, Comboio);

    IF Erro THEN
        ROLLBACK;
    ELSE
        COMMIT;
    END IF;
END
|
DELIMITER ;
-----

```

Figura 15 - Transação para inserir uma viagem.

```

-----
-- Transação para fazer reserva
DELIMITER |
CREATE PROCEDURE faz_reserva (IN Id_reserva INT, IN Data_reserva DATETIME, IN Cliente VARCHAR (40), IN Viagem INT,
                              IN Comboio INT, IN Carruagem INT, IN Lugar INT)
BEGIN
    DECLARE Preco DECIMAL(3, 2);
    DECLARE Erro BOOL DEFAULT 0;
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET Erro = 1;
    START TRANSACTION;
        SET Preco = (SELECT preco FROM viagem WHERE viagem.id_viagem = Viagem)*
                    (SELECT taxa from tipo_comboio, comboio WHERE comboio.id_comboio = Comboio
                     AND comboio.tipo = tipo_comboio.designacao);
        INSERT INTO reserva (id_reserva, preco, data_reserva, cliente, viagem, comboio, carruagem, lugar)
            VALUES (Id_reserva, Preco, Data_reserva, Cliente, Viagem, Comboio, Carruagem, Lugar);
    IF Erro THEN
        ROLLBACK;
    ELSE
        COMMIT;
    END IF;
END
|
DELIMITER ;
-----

```

Figura 16 - Transação para fazer uma reserva.

```

-----
-- Transação para consultar lugares num comboio
DELIMITER |
CREATE PROCEDURE consultar_lugares (IN Viagem INT)
BEGIN
    START TRANSACTION;
        SELECT comboio, carruagem, lugar FROM lugares WHERE (carruagem, lugar)
            NOT IN (SELECT carruagem, lugar FROM reserva WHERE reserva.viagem = Viagem AND
                    reserva.comboio = (SELECT comboio FROM viagem WHERE id_viagem = Viagem)) AND
                    lugares.comboio = (SELECT comboio FROM viagem WHERE id_viagem = Viagem);
    END
|
DELIMITER ;
-----

```

Figura 17 – Transação para consultar lugares disponíveis.

5.2.1 Mapa dos caminhos transacionais

Nesta secção apresenta-se o mapa de utilização de tabelas por parte das transações definidas:

	consulta_lugares				faz_reserva				insere_viagem			
	I	R	U	D	I	R	U	D	I	R	U	D
cliente												
reserva		X			X							
viagem		X				X			X			
estacao												
comboio		X				X						
tipo_comboio						X						
lugares		X										

I – Insert R – Read U – Update D - Delete

Tabela 5 - Caminhos transacionais nas relações.

5.3. Política de Segurança

A política de segurança da base de dados adotada assim como os perfis de utilização seguem o modelo referido aquando a elaboração da caracterização dos mesmos.

5.3.1 Regras de acessos a dados

Seguindo os perfis de utilização especificados na Secção 2, foram criados três tipos de utilizadores, *admin*, *gerente* e *user*.

Deste modo, definiram-se diferentes permissões aos tipos de utilizadores referidos. O utilizador *admin* tem acesso a tudo e pode fazer qualquer tipo de modificação necessária na BD. O utilizador *gerente* pode seleccionar, inserir e atualizar registos nas tabelas **viagem** e **comboio**. Por fim, o utilizador *user* tem permissões para modificar apenas a tabela **cliente** e a tabela **reserva** de modo a poder atualizar os seus dados. É de notar que todos os utilizadores podem fazer seleções em todas as tabelas da BD. O código SQL necessário para criar tais permissões está presente na figura abaixo.

```
CREATE USER 'admin'@'localhost' IDENTIFIED BY 'passwordadmin';
GRANT ALL PRIVILEGES ON ferroviaria.* TO 'admin'@'localhost';
```

Figura 18 - Perfil de Administrador da Base de Dados.

```
CREATE USER 'gerente'@'localhost' IDENTIFIED BY 'passwordgerente';
GRANT SELECT ON ferroviaria.* TO 'gerente'@'localhost';
GRANT INSERT, UPDATE ON ferroviaria.comboio TO 'gerente'@'localhost';
GRANT INSERT, UPDATE ON ferroviaria.viagem TO 'gerente'@'localhost';
```

Figura 19 - Perfil de Gerente da Base de Dados.

```
CREATE USER 'user'@'localhost' IDENTIFIED BY 'passworduser';
GRANT SELECT ON ferroviaria.* TO 'user'@'localhost';
GRANT INSERT, UPDATE ON ferroviaria.cliente TO 'user'@'localhost';
GRANT INSERT, UPDATE ON ferroviaria.reserva TO 'user'@'localhost';
```

Figura 20 - Perfil de Utilizador da Base de Dados.

No que diz respeito a backups, é uma parte bastante importante pois são feitas inúmeras viagens e reservas. Assim, seria aconselhável fazer um backup de dados.

5.4. Povoamento da base de dados

Para o povoamento da Base de Dados, foram utilizadas as transações criadas assim como inserções simples de modo a poder preencher alguma informação na nossa plataforma. O código SQL criado segue abaixo:

```
-- Povoar BD
INSERT INTO cliente (username, password, nome)
VALUES ('username1', '123', 'Helder'), ('username2', '1234', 'Andre'),
('username3', '12345', 'Antonio'), ('username4', '123456', 'Goncalo');

INSERT INTO tipoComboio (taxa, designacao)
VALUES ('0.9', 'Alfa'),
('0.6', 'Int');

INSERT INTO comboio (id_comboio, tipo)
VALUES (1, 'Alfa'),
(2, 'Alfa'),
(3, 'Int'),
(4, 'Int');

INSERT INTO lugares (comboio, carruagem, lugar)
VALUES (1, 1, 1), (1, 2, 2),
(2, 1, 1), (2, 2, 2),
(3, 1, 1), (3, 2, 2),
(4, 1, 1), (4, 2, 2);

INSERT INTO estacao (nome)
VALUES ('Braga'), ('Porto'), ('Aveiro'),
('Guarda'), ('Lisboa'), ('Madrid');

CALL insere_viagem (1, '2017-1-1 09:00:00', '2017-1-1 10:00:00', '5.5', 'Braga', 'Porto', '1');
CALL insere_viagem (2, '2017-1-2 16:00:00', '2017-1-2 20:00:00', '7.5', 'Lisboa', 'Madrid', '2');
CALL insere_viagem (3, '2017-1-3 10:00:00', '2017-1-3 14:00:00', '3.5', 'Porto', 'Lisboa', '3');

CALL faz_reserva (1, '2017-1-1 09:00:00', 'username1', '1', '1', '1', '1');
CALL faz_reserva (2, '2017-1-1 09:00:00', 'username1', '1', '1', '2', '2');
CALL faz_reserva (3, '2017-1-2 16:00:00', 'username2', '2', '2', '1', '1');
CALL faz_reserva (4, '2017-1-3 10:00:00', 'username3', '3', '3', '2', '2');
```

Figura 21 - Povoamento da BD.

5.5. Uso de disco e estimativa de crescimento

5.5.1 Uso de disco

De modo a obter uma estimativa dos requisitos em termos de espaço na Base de Dados, foi executado o seguinte código SQL:

```
-- Tamanho da Base de Dados em bytes
DELIMITER |
SELECT table_name AS 'Tabela', (data_length + index_length) AS 'Tamanho (bytes)'
  FROM information_schema.TABLES
  WHERE table_schema = 'ferroviaria';
DELIMITER ;
```

Figura 22 - Script de Calculo de Tamanho.

Assim, foi obtida o tamanho da BD no momento em bytes. Apresentam-se agora os resultados:

	Tabela ▲	Tamanho (bytes)
	cliente	16384
	comboio	32768
	estacao	16384
	lugares	16384
	reserva	65536
	tipo_comboio	16384
▶	viagem	65536

Figura 23 - Tamanho da BD em bytes.

5.5.2 Estimativa de crescimento da base de dados

A ferroviária Santos espera obter um crescimento de 100% anual, o que se reflete em aproximadamente 230 Kb no primeiro ano e aproximadamente 220 Mb ao fim de 10 anos, algo que face ao hardware moderno é muito facilmente comportável em qualquer máquina.

6. Conclusões e Trabalho Futuro

Nesta etapa, e para concluir, temos de destacar em primeiro lugar algumas fraquezas e pontos fortes neste sistema que desenvolvemos e em segundo lugar explicar um pouco a nossa apreciação acrescida após o termino deste trabalho sobre a dificuldade de elaborar bons esquemas até na presença de um caso simplicíssimo como o nosso.

Como já tínhamos referido anteriormente no modelo lógico, quanto ao possível crescimento desta base de dados não vemos grande futuro na vertente de modelação dos percursos devido à sua maior adequação a um modelo orientado a grafos, na nossa opinião, mas cremos que mantém alguma robustez na capacidade de gerir que comboios participam nestes percursos e eventualmente toda uma nova lógica de negócio por trás da gestão de funcionários caso esses requisitos se verificassem. No entanto, na nossa opinião, o sistema que elaborámos pelo menos cumpre estes requisitos mais simples perfeitamente.

Quanto à elaboração e conceptualização destes esquemas finais, queremos apontar que ao ser mencionada na nossa metodologia que este processo é iterativo, tal se verificou no nosso caso. Foi-nos necessário mais que uma vez refinar e reavaliar os nossos modelos em todas as etapas da modelação, pelo que ficou bem clara a dificuldade por trás de avaliar e documentar um bom sistema de gestão de base dados até face ao nosso caso bastante elementar.

Embora tenhamos feito uma documentação bem exaustiva de todo o processo de conceptualização e modelação do nosso sistema não tratámos de um tópico também ele com muito que se lhe diga, parcialmente por ser uma área extensa e que não abordámos em grande detalhe durante o leccionamento da disciplina e parcialmente por a nossa base de dados conter tão poucos dados que passa pela área de análise, manutenção e optimização da base de dados, tarefa que podia eventualmente ficar para tópico futuro.

Gostaríamos também, caso tivéssemos tido mais tempo, de elaborar um modelo um pouco mais complexo, possivelmente tendo classes associadas às carruagens e comodidades associadas a lugares (como presença de tomadas elétricas) para melhor justificar o cálculo de um preço derivado em função do lugar reservado, ou até talvez incluir, além de reservas, funcionários e bilheteiras, passando os bilhetes a ser reservas concretizadas, e explorar a lógica por trás de tais requisitos.

7. Referências Bibliográficas

T. Connolly and C. Begg, *Database Systems, A Practical Approach to Design, Implementation, and Management*, 4th ed. Boston, Addison-Wesley, 2004.

8. Lista de Siglas e Acrónimos

<<Apresentar uma lista com todas as siglas e acrónimos utilizados durante a realização do trabalho. O formato base para esta lista deverá ser da forma como abaixo se apresenta.>>

BD	Base de Dados
SGBD	Sistema de Gestão de Base de Dados
1NF	Primeira Forma Normal
2NF	Segunda Forma Normal
3NF	Terceira Forma Normal
ER	Entidade-Relacionamento
DBDL	<i>Database Definition Language</i>
PK	<i>Primary Key</i>
FK	<i>Foreign Key</i>
AK	<i>Alternate Key</i>
SQL	<i>Structured Query Language</i>