



Universidade do Minho
Licenciatura em Engenharia Informática

Unidade Curricular de Bases de Dados

Ano Letivo de 2025/2026

Implementação de um SBD para a Novos Horizontes

Afonso Barros, Diogo Pedrosa, Luís
Pedrosa, Matheus Azevedo

Setembro, 2025

B **D**

Data de Recepção	
Responsável	
Avaliação	
Observações	

Resumo

Este relatório documenta o desenvolvimento de um *Sistema de Base de Dados* (SBD) relacional para a Escola Básica e Secundária Novos Horizontes, focado na gestão e catalogação de viagens virtuais pedagógicas. O projeto surge da necessidade de transitar de um modelo de gestão manual e local para uma solução centralizada que garanta a integridade dos dados, o controlo de acessos e que facilite o acesso ao serviço face ao aumento da procura por conteúdos multimédia educativos.

O trabalho seguiu o ciclo de vida clássico de desenvolvimento de bases de dados, iniciando-se pela definição do sistema, seguida do levantamento de requisitos de descrição, manipulação e controlo junto dos intervenientes. Na modelação conceptual, desenhamos um diagrama Entidade-Relacionamento que reflete a ligação entre alunos, professores, disciplinas e conteúdos. A fase de modelação lógica refinou esta estrutura através da normalização, consideramos normalizado a partir da 3 *Forma Normal* (FN).

A implementação física foi realizada em MySQL, incorporando mecanismos avançados como o *Controlo de Acessos Baseado em Cargos* (RBAC), indexação estratégica para otimização de consultas e lógica server-side através de gatilhos de validação cronológica e procedimentos armazenados para análise de preferências. O estado final do projeto é um sistema funcional, validado por álgebra relacional e testes de povoamento, que oferece uma base sólida para a futura integração com uma interface aplicacional.

Índice

Resumo	1
Índice	2
Índice de Figuras	4
Índice de Tabelas	5
1. Definição do Sistema	6
1.1 Contexto de aplicação	6
1.2 Motivação e Objetivos do Trabalho	6
1.3 Análise da Viabilidade do processo	7
1.4 Recursos e Equipa de Trabalho	8
1.5 Plano de Execução do Projeto	9
2. Levantamento e Análise de Requisitos	11
2.1 Método de Levantamento e de Análise de Requisitos Adotado	11
2.2 Organização dos Requisitos Levantados	12
2.3 Análise e Validação Geral dos Requisitos	19
3. Modelação Conceptual	21
3.1 Apresentação da Abordagem de Modelação Realizada	21
3.2 Identificação e Caracterização das Entidades	21
3.3 Identificação e Caracterização dos Relacionamentos	23
3.4 Identificação e Caracterização dos Atributos das Entidades e dos Relacionamentos.	25
3.5 Apresentação e Explicação do Diagrama ER Produzido	28
4. Modelação Lógica	30
4.1 Construção e Validação do Modelo de Dados Lógico	30
4.2 Apresentação e Explicação do Modelo Lógico Produzido	31
4.3 Normalização de Dados	36
4.4 Validação do Modelo com Interrogações do Utilizador	39
5. Implementação Física	44
5.1 Apresentação e explicação da base de dados implementada	44
5.2 Criação de utilizadores da base de dados	52
5.3 Povoamento da base de dados	59
5.4 Cálculo do espaço da base de dados (inicial e taxa de crescimento anual)	65
5.5 Definição e caracterização de vistas de utilização em SQL	73
5.6 Tradução das interrogações do utilizador para SQL	75
5.7 Indexação do Sistema de Dados	76
5.8 Implementação de procedimentos, funções e gatilhos	78
6. Conclusões e Trabalho Futuro	85
7. Bibliografia	87
7.1 Referências	87
7.2 Lista de Siglas e Acrónimos	87
Anexos	1
I. Diagrama de Gantt	2

II. Povoamento RelaX	3
III. Expressões Álgebra Relacional	9
IV. Script criação SQL	10
V. Script povoamento SQL	15
VI. Script Queries, Índices, Vistas, Funções, Gatilhos, Procedimentos	22
VII. Script Roles e Users	30
VIII. Modelo Conceptual	33
IX. Modelo Lógico	34

Índice de Figuras

Figura 1 - Cabeçalho da ata da reunião de estabelecimento de datas	9
Figura 2 - Diagrama de Gantt	10
Figura 3 - Cabeçalho da ata da reunião de início de levantamento de requisitos	
12	
Figura 4 - Cabeçalho da ata da reunião de fim de levantamento de requisitos	12
Figura 5 - Cabeçalho da ata da reunião de verificação dos requisitos	
levantados	20
Figura 6 - Modelo conceptual	29
Figura 7 - Modelo Lógico	36
Figura 8 - Tabela Aluno	37
Figura 9 - Tabela Aluno_Viagem	38
Figura 10 - Tabela Viagem_Disciplina	38
Figura 11 - Árvore da interrogação do RM7	40
Figura 12 - Árvore da interrogação do RM8	41
Figura 13 - Árvore da interrogação do RM10	42
Figura 14 - Árvore da interrogação do RM11	43
Figura 15- Cabeçalho da ata da reunião para obter dados para fazer estimativas sobre o tamanho inicial e taxa de crescimento da BD	
	71

Índice de Tabelas

Tabela 1 - Requisitos de Descrição	13
Tabela 2 - Requisitos de Manipulação	16
Tabela 3 - Requisitos de Controlo	18
Tabela 4 - Caracterização das Entidades	23
Tabela 5 - Caracterização dos Relacionamentos	25
Tabela 6 - Caracterização dos Atributos da entidade Aluno	25
Tabela 7 - Caracterização dos Atributos da entidade Professor	26
Tabela 8 - Caracterização dos Atributos da entidade Disciplina	26
Tabela 9 - Caracterização dos Atributos da entidade Viagem	27
Tabela 10 - Caracterização dos Atributos da entidade Conteúdo	27
Tabela 11 - Caracterização dos Atributos do relacionamento Aluno-Realiza-Viagem	28
Tabela 12 - Dicionário de dados da tabela Professor	31
Tabela 13 - Dicionário de dados da tabela Aluno	32
Tabela 14 - Dicionário de dados da tabela Disciplina	32
Tabela 15 - Dicionário de dados da tabela Viagem	32
Tabela 16 - Dicionário de dados da tabela Localizacao	34
Tabela 17 - Dicionário de dados da tabela Cidade	34
Tabela 18 - Dicionário de dados da tabela Pais	34
Tabela 19 - Dicionário de dados da tabela Conteudo	35
Tabela 20 - Dicionário de dados da tabela Aluno_Viagem	35
Tabela 21 - Dicionário de dados da tabela Viagem_Disciplina	36
Tabela 22 - Permissões do cargo aluno	54
Tabela 23 - Permissões do cargo professor	56
Tabela 24 - Permissões do cargo bibliotecario	57
Tabela 25 - Domínios utilizados e respetivo tamanho em bytes	66
Tabela 26 - Espaço ocupado por um registo da relação Aluno	67
Tabela 27 - Espaço ocupado por um registo da relação Professor	68
Tabela 28 - Espaço ocupado por um registo da relação Disciplina	68
Tabela 29 - Espaço ocupado por um registo da relação Viagem	68
Tabela 30 - Espaço ocupado por um registo da relação Conteudo	69
Tabela 31 - Espaço ocupado por um registo da relação Aluno_Viagem	69
Tabela 32 - Espaço ocupado por um registo da relação Viagem_Disciplina	69
Tabela 33 - Espaço ocupado por um registo da relação Localizacao	69
Tabela 34 - Espaço ocupado por um registo da relação Cidade	70
Tabela 35 - Espaço ocupado por um registo da relação Pais	70

1. Definição do Sistema

1.1 Contexto de aplicação

A Escola Básica e Secundária Novos Horizonte, desde o ano de 2017, criou condições para permitir que os seus alunos pudessem aceder a viagens virtuais nos computadores da sua biblioteca. Estes conteúdos eram criados pelos professores para instigar a curiosidade dos alunos e disponibilizar um meio recreativo para os alunos aprofundarem o seu conhecimento sobre os diferentes conteúdos lecionados. Estas viagens eram geralmente em locais relacionados com os conteúdos lecionados, como monumentos históricos e museus.

Numa fase inicial, o sistema operava num modelo local. Os conteúdos estavam disponíveis apenas nos discos rígidos dos três computadores que a biblioteca tinha disponível. Era responsabilidade do Sr. Bruno, funcionário da biblioteca, tratar manualmente da administração, cópia e publicação dos conteúdos criados em cada máquina. Após a pandemia, verificou-se um aumento do interesse por parte dos alunos por conteúdos virtuais, devido à popularização desse tipo de conteúdo na fase de confinamento, o que levou a um aumento da procura e acesso às viagens virtuais disponíveis.

1.2 Motivação e Objetivos do Trabalho

Com o aumento da popularidade do serviço de conteúdos virtuais disponibilizado pela escola, o Sr. Bruno tem tido dificuldade em gerir o processo sozinho. Primeiramente, o aumento do número de alunos que queriam utilizar os computadores começou a gerar conflitos entre alunos que, durante os intervalos, disputavam o tempo de utilização dos computadores, impedindo o uso dos equipamentos para outras finalidades. Além disso, o Sr. Bruno tem tido dificuldade em organizar e manter os três computadores sincronizados e atualizados, visto que as viagens eram adicionadas manualmente em cada um dos computadores e mantidas numa pasta única na qual os alunos muitas vezes eram incapazes de distinguir quais eram os conteúdos destinados ao seu ano escolar.

Por fim, surgiram ainda problemas de integridade e segurança, uma vez que, alguns alunos alteraram, eliminaram e duplicaram parte das viagens, uma vez que não existia um sistema de permissões ou autenticação associado à utilização destes serviços.

Por esses motivos, a direção da escola procurou ajuda e decidiu implementar um SBD para gerir as viagens virtuais, com a ajuda de um grupo de alunos da Universidade do Minho que decidiram ajudar neste projeto a título voluntário, no sentido de ganharem experiência e de aplicarem o conhecimento que aprenderam ao longo da unidade curricular de *Bases de Dados* (BD).

Dado o contexto apresentado anteriormente, a direção da escola juntamente com os seus docentes e funcionários definiu os seguintes objetivos a alcançar com a implementação da base de dados:

- Permitir que os alunos accedam à aplicação através de utilizadores individuais garantindo que eles possam visualizar e avaliar as viagens permitindo o acesso remoto através dos seus próprios computadores, reduzindo a afluência física aos computadores da escola;
- Permitir que os professores coloquem as suas viagens virtuais na base de dados diretamente sem dependerem do auxílio do Sr. Bruno;
- Manter um registo de quais são os conteúdos que os alunos estão a aceder e possibilitar que estes deixem os seus comentários e avaliações;
- Implementar ferramentas de gestão dos conteúdos da base de dados que permitam ao Sr. Bruno desempenhar o seu papel administrativo com facilidade;
- Otimizar a pesquisa, permitindo que os alunos filtrem as viagens desejadas por diversos parâmetros.

1.3 Análise da Viabilidade do processo

A gestão atual das viagens virtuais, assente num modelo manual de cópia de ficheiros entre discos rígidos e registos em papel pelo funcionário da biblioteca, apresenta falhas graves de sincronização e segurança. Com base no levantamento inicial, a implementação do *Sistema de Base de Dados* (SBD) é considerada viável e rentável, prevendo-se os seguintes ganhos tangíveis:

- **Redução do tempo de administração operacional:** O Sr. Bruno deixará de ter de realizar a cópia manual e a publicação individual de conteúdos em cada

máquina, tarefa que atualmente consome a totalidade do seu tempo dedicado ao serviço.

- **Eliminação total de redundâncias e conflitos de dados:** A centralização da informação num SBD garante que deixem de existir viagens duplicadas ou versões desatualizadas espalhadas pelos computadores da biblioteca.
- **Otimização de custos de infraestrutura:** Ao permitir o acesso remoto através de computadores próprios, a escola evita um investimento imediato na compra de novos equipamentos, maximizando a utilidade da rede interna já existente.
- **Melhoria na Integridade e Segurança:** A implementação de um sistema de permissões eliminará as eliminações acidentais de conteúdos por parte dos alunos, protegendo o património digital da escola.

A direção concluiu que o projeto é viável devido ao seu custo de desenvolvimento nulo, uma vez que a mão de obra é assegurada por estudantes de Engenharia Informática da Universidade do Minho em regime de voluntariado. A utilização de ferramentas *open-source*, o *MySQL*, e o aproveitamento do *hardware* pessoal dos alunos e da escola garantem que o benefício operacional supera largamente o investimento material necessário.

1.4 Recursos e Equipa de Trabalho

Para a criação do sistema de base de dados foram necessários recursos humanos e materiais.

No que diz respeito aos recursos humanos: temos a equipa constituída por quatro alunos de Engenharia Informática da Universidade do Minho: Afonso Barros, Diogo Pedrosa, Luís Pedrosa, Matheus Azevedo. O projeto conta ainda com a colaboração do Sr. Bruno e da direção da escola para o levantamento de requisitos. O custo associado à mão de obra por parte da equipa de desenvolvimento foi nula, uma vez que eles se voluntariaram para ajudar.

Ao nível dos recursos materiais: recorremos ao computador pessoal de cada um dos alunos e a um conjunto de ferramentas de *software* adequadas às diferentes fases do projeto: *Google Sheets* e *Google Forms* para a gestão e análise preliminar; *brModelo* para a modelação conceptual; e *MySQL Workbench* para a modelação lógica e a implementação física da base de dados. Foram ainda utilizados recursos como: *Google Meet* e *Discord* tanto para reuniões entre os alunos da universidade como com

a direção da escola; e *Notion* para tomar notas das reuniões e ideias relativas ao desenvolvimento do projeto.

1.5 Plano de Execução do Projeto

Tendo em conta a implementação do sistema de base de dados para a escola Novos Horizontes, foi realizada uma reunião inicial com os quatro estudantes e a direção da escola, onde foram debatidas datas e durações plausíveis para a realização das diferentes etapas do projeto. Durante a reunião foi utilizado o *Google Sheets* para a elaboração do Diagrama de Gantt, onde ficaram registadas as datas estabelecidas na reunião



Figura 1- Cabeçalho da ata da reunião de estabelecimento de datas

O Diagrama de Gantt pode ser consultado a seguir:

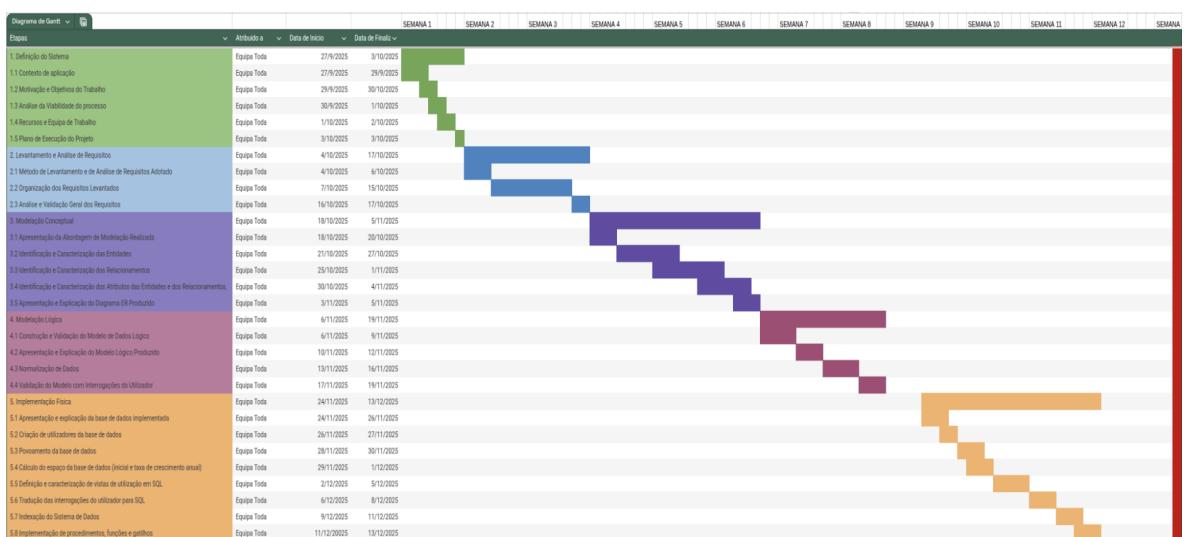


Figura 2 - Diagrama de Gantt

O diagrama do projeto foi desenhado tendo em conta aquilo que julgamos ser o grau de dificuldade das diferentes etapas, além de considerarmos as limitações de tempo devido a testes e entrega de projetos e, ainda, a urgência da implementação do sistema para a escola.

A fase de Definição do Sistema foi estabelecida para ocorrer entre 27/9 e 03/10, sendo mantida curta para garantir uma assimilação rápida do problema da escola e validar os objetivos centrais, permitindo avançar rapidamente. Para o Levantamento e Análise de Requisitos, de 4 de outubro a 17 de outubro, esperamos que demorasse mais tempo devido à necessidade de realização de reuniões e validação com a Direção da escola e o Sr. Bruno para evitar erros de interpretação que poderiam comprometer as fases seguintes.

A Modelação Conceptual foi considerada uma das mais longas etapas pois é nesta que se realiza o primeiro esboço da base de dados com a respetiva descrição de todos os seus elementos. Esta fase deve garantir que reflete de forma exata os requisitos levantados.

A quarta etapa, a Modelação Lógica foi considerada um pouco mais curta que a anterior pois exige aplicação de regras de conversão do modelo lógico para o conceptual que, sendo corretamente usadas, garantem a normalização dos dados.

Estabelecemos depois um período de pausa no desenvolvimento devido a um período de maior sobrecarga no período de avaliações.

Por fim, a Implementação Física, de dia 24 de novembro a dia 13 de dezembro, consideramos ser a parte mais longa no projeto devido ao maior número de etapas e por se tratar de uma nova linguagem com a qual não estávamos familiarizados.

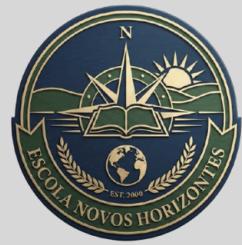
2. Levantamento e Análise de Requisitos

2.1 Método de Levantamento e de Análise de Requisitos Adotado

Para a construção do sistema de gestão da base de dados da Escola Básica e Secundária Novos Horizontes, foi necessário realizar um levantamento dos requisitos do sistema a ser implementado, de forma a garantir que este respondesse de forma eficaz às necessidades reais da escola.

O processo de levantamento de requisitos foi conduzido pela equipa de desenvolvimento constituída pelos quatro alunos da Universidade do Minho, em estreita colaboração com a direção da escola, diretores de turma e com o funcionário responsável pela biblioteca, Sr. Bruno. Foram agendadas reuniões com todos os intervenientes com o objetivo de perceber os processos atuais da gestão das viagens virtuais nos computadores da biblioteca, identificar as principais dificuldades e os requisitos que a BD necessitará.

Esta etapa foi essencial para desenhar um sistema intuitivo. Foi ainda analisado o livro de anotações do Sr. Bruno onde ele ia tomando notas relativamente ao funcionamento da plataforma de viagens e das tarefas de administração que ia executando na mesma.



Lavantamento de Requisitos

Data : 18/10/2025

Local: Google Meet- Online

Hora : 14:00

Objetivos:

- Recolher requisitos de descrição que iriam sustentar o desenvolvimento do sistema a construir
- Começar a recolher os requisitos de descrição
- Analisar os métodos atuais de gestão das viagens virtuais

Participantes:

- Grupo de 4 alunos da Universidade do Minho
- Direção da escola Novos Horizontes
- Sr. Bruno

Figura 3- Cabeçalho da ata da reunião de início de levantamento de requisitos

Lavantamento de Requisitos

Data : 23/10/2025

Local: Google Meet- Online

Hora : 15:50

Objetivos:

- Terminar de recolher os requisitos de manipulação
- Recolher os requisitos de controlo

Participantes:

- Grupo de 4 alunos da Universidade do Minho
- Direção da escola Novos Horizontes
- Sr. Bruno

Figura 4- Cabeçalho da ata da reunião de fim de levantamento de requisitos

2.2 Organização dos Requisitos Levantados

Para assegurar a consistência e utilidade dos requisitos recolhidos para o sistema de gestão das viagens virtuais da Escola Básica e Secundária Novos Horizontes, procedeu-se à sua organização e documentação segundo tabelas de requisitos. Eles foram levantados pelo grupo de estudantes que, nesta fase, atuaram como analistas. Dividimos os requisitos em três categorias:

Os *Requisitos de Descrição* (RD) definem a estrutura e os elementos de dados do sistema, incluindo entidades, relacionamentos entre elas e atributos.

Nº	Data/Hora	Descrição	Fonte
RD1	18-10-2025 14:03	Um aluno é caracterizado pelo seu número de estudante (único e atribuído pela escola). Além disso, deve ser incluído o nome, o gênero (Masculino/Feminino) e a turma: ano e letra do aluno.	Direção
RD2	18-10-2025 14:04	Um professor é identificado pelo seu número de professor (único e atribuído pela escola) e deve ser guardado o seu nome.	Direção
RD3	18-10-2025 14:11	Uma disciplina é identificada pelo código único de disciplina (número único), pelo seu nome e pelo ano em que é lecionada.	Direção
RD4	18-10-2025 14:22	Uma viagem é caracterizada por um identificador único, pelo seu título, resumo e data de publicação.	Professores/ Sr.Bruno
RD5	18-10-2025 14:25	Um conteúdo é caracterizado por um identificador único, pelo tipo de conteúdo que está anexado (Foto ou Vídeo), pelo url para aceder a esse conteúdo, pela data associada à captura do conteúdo, por um texto de descrição, um título e pela localização(nome, cidade e país).	Professores/ Sr.Bruno
RD6	18-10-2025 14:31	Um professor pode ou não criar viagens e uma viagem é sempre criada por um único professor.	Sr.Bruno
RD7	18-10-2025 14:36	Um aluno pode realizar várias viagens e no final de as realizar deve deixar um comentário e avaliação (entre 1 e 10) tendo a sua data de realização associada. Cada viagem pode ser realizada por vários alunos.	Professores
RD8	18-10-2025 14:33	Uma viagem enquadra-se em uma ou várias disciplinas, e nem todas as disciplinas têm viagens associadas. Uma disciplina pode ter várias viagens que se enquadrem nela.	Sr.Bruno
RD9	18-10-2025 14:32	Uma viagem tem pelo menos um conteúdo e um conteúdo está sempre associado a uma única viagem.	Sr.Bruno

Tabela 1 - Requisitos de Descrição

A tabela acima representa todos os requisitos de descrição que foram retirados de uma das reuniões com a direção, professores e o Sr. Bruno e que resultaram de um processo de discussão entre as partes envolvidas.

A título de exemplo, consideremos os seguintes exemplos:

Ao longo da reunião, os professores pediram que as viagens que eles adicionavam deviam ter um título e uma pequena descrição do que ela iria abordar. Daí surgiu a necessidade de criar uma entidade "Viagem" com os atributos título e resumo. Por sua vez, o Sr. Bruno mencionou que os alunos sentiam muita dificuldade em encontrar as viagens que tinham sido adicionadas recentemente, por esse motivo, decidiu-se colocar um atributo de data de publicação associado à viagem. Face aos acontecimentos mencionados anteriormente a equipa de desenvolvimento criou o RD4 que incluía também um identificador artificial único que permitisse distinguir as viagens univocamente.

Por outro lado, os professores pediram que fosse possível adicionar conteúdos multimédia às viagens, este poderiam ser uma foto ou um vídeo e deviam ser ter uma descrição para contextualização e identificar a localização - nome do local, cidade, país - e a data onde foram criados. Além disso, o Sr. Bruno mencionou que verificou que o crescimento do número de viagens estava a aumentar demasiado o consumo de memória ocupado pelas viagens. Face ao pedido dos professores, às preocupações do Sr. Bruno e com o objetivo de reduzir o tamanho da base de dados, a equipa de desenvolvimento recomendou que as imagens e vídeos não fossem carregados diretamente para a base de dados. Em vez disso, sugeriu-se o armazenamento de um *Uniform Resource Locator* (URL), acompanhado de uma indicação sobre se o conteúdo é uma imagem ou um vídeo. Esse URL apontaria para plataformas externas gratuitas onde os ficheiros estariam publicados ou guardados - como *YouTube*, *Instagram* ou *Google Drive* -, permitindo assim diminuir os recursos e consequentemente os custos associados ao armazenamento interno. Deste modo, chegou-se a um acordo que um conteúdo deveria ter um identificador artificial único, o URL para o conteúdo multimédia com a devida identificação de tipo, foto ('F') ou vídeo ('V'), a descrição, a data e a localização composta pelo nome do local, cidade e país. Ou seja, este foi o processo que gerou a criação do RD5.

Por fim, os professores disseram durante a reunião que queriam disponibilizar as viagens para que todos os alunos da escola pudessem vê-las, contudo eles gostavam que fosse possível ter um *feedback* dos alunos para que eles pudessem rever o que era mais apreciado pelos alunos. Por esse motivo, sugeriu-se que de cada vez que um aluno realizasse uma viagem deveria existir um pequeno texto de opinião sobre o que o aluno achou da viagem e uma nota de classificação. Além disso, os professores

gostavam de ter conhecimento sobre quando os alunos realizaram as viagens. Desse modo, desenvolveu-se o RD7 que explica o relacionamento entre os alunos e as viagens.

Os *Requisitos de Manipulação* (RM) especificam as operações que podem ser realizadas sobre os dados, como inserção, atualização, remoção e consultas, permitindo que os utilizadores interajam eficazmente com o sistema.

Nº	Data/Hora	Descrição	Fonte
RM1	18-10-2025 14:00	Registrar novos alunos e professores e alterar os dados dos existentes.	Direção
RM2	18-10-2025 14:00	Registrar novas disciplinas na base de dados.	Professores
RM3	18-10-2025 14:00	Registrar novas localizações geográficas, incluindo a gestão da hierarquia de cidades e países.	Sr. Bruno
RM4	18-10-2025 14:00	Registrar novas viagens associando-as às respetivas disciplinas.	Bruno
RM5	18-10-2025 14:00	Inserir conteúdos para uma viagem, associando-os a uma localização específica.	Sr. Bruno
RM6	23-10-2025 17:00	Registrar a realização de uma viagem por um aluno, incluindo a avaliação, comentário e a data de realização.	Direção
RM7	23-10-2025 17:00	Listar todas as viagens disponíveis para uma determinada disciplina dado o seu identificador, ordenadas da mais recente para a mais antiga (apresentando o id, o título, o resumo e a data de publicação)	Direção
RM8	23-10-2025 17:00	Listar os alunos (id e nome) que realizaram uma determinada viagem, ordenados alfabeticamente, apresentando a sua respetiva avaliação	Direção
RM9	23-10-2025 17:00	Gerar um "Top 10 Alunos" com mais viagens realizadas, listando o número, nome, turma (ano e letra) e o número de viagens realizadas por ele (em caso de empate, dá-se prioridade ao aluno que realizou a última viagem há mais tempo).	Sr. Bruno
RM10	23-10-2025 17:00	Consultar a classificação média atual de todas as viagens, apresentando o respetivo identificador, título e o valor	Sr. Bruno

		da média calculada, ordenando os resultados pela média de forma decrescente.	
RM11	23-10-2025 17:00	Listar o desempenho dos professores, apresentando o id do professor, nome do professor e o número total de viagens virtuais criadas por cada um, ordenado por ordem decrescente da quantidade de viagens criadas. Além disso, devem ser apresentadas as classificações médias mínimas e máximas das suas viagens.	Direção
RM12	23-10-2025 17:00	Consultar o "Perfil de Preferências de Disciplinas" de um aluno. Este mostra o número do aluno, o nome e uma lista das disciplinas com a nota média das avaliações dadas por ele em viagens que se enquadram nesta disciplina e o total de viagens (deve ordenar pela nota média e em caso de empate pela quantidade de viagens).	Professores
RM13	23-10-2025 17:00	Pesquisar viagens através de uma expressão de texto presente no título ou resumo das mesmas. Devem ser listadas todas as viagens que cumprem o requisito por ordem alfabética dos títulos.	Sr. Bruno

Tabela 2 - Requisitos de Manipulação

Na tabela acima temos os requisitos de manipulação que foram estipulados ao longo da reunião com os professores, a direção da escola e o Sr. Bruno. Vamos analisar como foi o processo de formação dos requisitos de manipulação

A título de exemplo, foi mencionado pelos professores que fosse possível obter uma lista de todas as viagens que tivessem associadas às suas disciplinas, para que dessa forma eles pudessem analisar e criar novas viagens que fossem distintas das já existentes. Daí, surgiu o requisito de manipulação 7.

Por outro lado, ao longo da reunião foi discutido como poderia se aumentar o engajamento dos alunos na plataforma. Em meio a discussão foi mencionado pelo Sr. Bruno que a motivação para utilizar a base de dados poderia ser gerada a partir de um sistema de classificação de modo a despertar o espírito competitivo dos jovens estudantes. Além disso, o Sr. Bruno acrescentou que ouvia muitas vezes os alunos a

comentar sobre as suas estatísticas nas redes sociais. Utilizando essas ideias como base foram concebidos dois requisitos de manipulação. Em primeiro lugar, o requisito de manipulação 9 (RM9) que surgiu com o objetivo de incentivar os alunos a realizar mais viagens, dado que este permitia ver quem eram os dez alunos em toda a escola que haviam realizado mais viagens, contadas a partir do número de avaliações que eles haviam feito. No “Top10” devem ser apresentados, o nome, número e turma, assim como o número de viagens realizadas pelos alunos. De modo a motivar os alunos a estarem sempre atentos à plataforma. Foi estabelecido que o critério de desempate do “Top 10” fosse colocar em primeiro o aluno que completou o seu total de viagens primeiro. Por sua vez, o requisito de manipulação 12 surgiu para garantir que cada aluno pudesse facilmente consultar as suas estatísticas relativamente às disciplinas das viagens que eles visualizaram. Devia, por isso, ser possível que um aluno apenas utilizando o seu número de estudante pudesse consultar uma lista de disciplinas favoritas ordenadas pela classificação média atribuída às viagens nas quais elas se enquadram e em caso de empate considerava-se o número de viagens como critério de ordenação.

Por fim, a pedido da direção devia ser possível consultar a lista dos professores da escola de modo a avaliar a atividade dos professores na plataforma. Para isso foi pedido que fosse possível gerar uma tabela com todos os nomes e identificadores dos professores acompanhados do número de viagens criadas pelos mesmos e a classificação média mínima e máxima das suas viagens. Face ao pedido a equipa de desenvolvimento da base de dados criou o requisito de manipulação 11.

Por último, os *Requisitos de Controlo* (RC) atuam como uma ferramenta de segurança ao auxiliarem na definição dos perfis de utilização.

Nº	Data/Hora	Descrição	Fonte
RC1	23-10-2025 14:05	Os alunos devem ser capazes de consultar todos os conteúdos da base de dados, exceto as informações pessoais sobre os alunos e professores.	Direção
RC2	23-10-2025 14:18	Os alunos devem ser capazes de adicionar a sua opinião relativa às viagens.	Sr. Bruno
RC3	23-10-2025 14:32	Os professores são capazes de ver todas as informações presentes na base de dados.	Professores
RC4	23-10-2025 14:47	Os professores apenas devem ser capazes de inserir novas viagens,	Professores

		conteúdos, localizações, cidades e países na base de dados.	
RC5	23-10-2025 15:05	Os professores apenas devem ser capazes de alterar dados relativos às viagens e aos conteúdos da base de dados.	Professores
RC6	23-10-2025 15:05	Os professores não devem ter permissões de remoção de informações da base de dados.	Professores
RC7	23-10-2025 15:20	O Sr. Bruno deve ter a capacidade de adicionar novos alunos, professores e disciplinas e atualizar as informações deles.	Direção
RC8	23-10-2025 15:38	O Sr. Bruno deve ser capaz de ver todos os dados na base de dados.	Direção
RC9	23-10-2025 16:03	O Sr. Bruno deve ser o único a ser capaz de remover dados da base de dados.	Professores
RC10	23-10-2025 16:17	Não devem existir utilizadores com a capacidade de partilhar os seus privilégios.	Sr. Bruno
RC11	23-10-2025 16:25	Nenhum dos utilizadores criados deve ter permissão de alterar a estrutura física da base de dados.	Direção
RC12	23-10-2025 16:39	Todos os utilizadores da base de dados devem ter permissão para apenas ver a vista com os detalhes das viagens.	Sr. Bruno
RC13	23-10-2025 17:01	Todos os utilizadores da base de dados devem ser capazes de apenas ver o top10 de alunos.	Sr. Bruno
RC14	23-10-2025 16:52	Todos os utilizadores devem ter a capacidade de utilizar a função de consulta do número de viagens realizadas por um aluno.	Sr. Bruno
RC15	23-10-2025 16:57	Apenas os professores e o Sr. Bruno devem ter acesso à função de análise de sucesso das viagens.	Direção
RC16	23-10-2025 17:05	Todos os utilizadores devem ter acesso ao procedimento que mostra o perfil de preferências do aluno.	Sr. Bruno
RC17	23-10-2025 17:18	Todos os utilizadores da base de dados devem ter acesso ao procedimento desenvolvido para pesquisa.	Sr. Bruno

Tabela 3 - Requisitos de Controlo

Ao longo da reunião foram discutidos quais as permissões que cada utilizador deveria ter na Base de Dados. Deste modo, foram criados os dezassete requisitos de controlo apresentados na tabela acima. Tal como os requisitos anteriores estes foram definidos

ao longo das reuniões com a intervenção de todos os presentes. A título de exemplo, foi definido que os alunos deviam ter a capacidade de ver todo o conteúdo que estava relacionado com as viagens, assim foi criado o requisito de controlo 1 (RC1). Por outro lado, os professores pediram para ter a capacidade de adicionar as suas viagens de forma independente, ou seja, os professores deviam ter permissões de adicionar todos os dados essenciais à criação de novas viagens, o que deu origem ao requisito de controlo 4 (RC4). Por outro lado, a direção definiu que o Sr. Bruno teria a responsabilidade de adicionar todos os dados estáticos da base de dados, ou seja, todas as informações relativas a alunos, professores e disciplinas ficavam da sua responsabilidade, o que deu origem ao requisito de controlo 7 (RC7). Por fim, os professores manifestaram a sua preocupação de se removerem as suas viagens por engano. Assim, foi chegado a um acordo de que as permissões de remoção de dados ficariam apenas disponíveis para o utilizador do Sr. Bruno, o que levou à criação do requisito de controlo 9 (RC9). E por questões de segurança, por exemplo, de modo a evitar que os alunos por engano tivessem a capacidade de alterar dados da base de dados deveria ser removida a permissão de todos os utilizadores de transmitir os seus privilégios, o que gerou o requisito de controlo 10 (RC10).

2.3 Análise e Validação Geral dos Requisitos

Com o objetivo de fazer uma validação final dos requisitos que foram recolhidos pela equipa de estudantes, foi realizada uma reunião com a direção da escola e o Sr. Bruno. Depois da alteração de alguns requisitos, os membros da direção da escola ficaram satisfeitos com os mesmos, permitindo a continuação do desenvolvimento do sistema de base de dados.



Verificação dos requisitos levantados

Data : 20/10/2025

Hora : 11:45

**Local: Escola Novos
Horizontes - Presencial**

Objetivos:

- Reavaliar os diferentes requisitos levantados
- Verificar se todos os aspectos necessários para a implementação do sistema estão abordados nos requisitos presentes

Participantes:

- Grupo de 4 alunos da Universidade do Minho
- Direção da escola Novos Horizontes
- Sr. Bruno

Figura 5- Cabeçalho da ata da reunião de verificação dos requisitos levantados

3. Modelação Conceptual

3.1 Apresentação da Abordagem de Modelação Realizada

Após a conclusão da fase de recolha e análise dos requisitos, a equipa de desenvolvimento iniciou o processo de planeamento da estrutura e organização da Base de Dados. Para obtermos uma visão clara sobre os elementos centrais do sistema e as interações entre eles, recorremos à construção de um modelo conceptual, representado através de um diagrama *Entidade-Relacionamento* (ER). Como ferramenta de apoio, utilizou-se o *brModelo*, que permite criar diagramas numa variante da notação de Chen com indicação explícita da cardinalidade e da participação dos relacionamentos.

3.2 Identificação e Caracterização das Entidades

A identificação das entidades constitui um passo fundamental no processo de modelação de uma base de dados, pois permite definir os objetos principais que o sistema deve representar. Cada entidade corresponde a um conjunto de elementos com características semelhantes e existência autónoma dentro do domínio do problema.

Com base nos requisitos de descrição recolhidos junto da Escola Básica e Secundária Novos Horizontes, foram identificadas as seguintes entidades principais:

Aluno

Entidade que representa os estudantes da escola que vão usar a plataforma para aceder às viagens virtuais e deixar o seu comentário. Cada aluno é identificado pelo seu número de estudante, o identificador único. É também guardado o nome, o género e a turma do aluno - ano e letra. Deste modo, o aluno justifica a sua existência como entidade independente, tal como se percebe pelo requisito **RD1**.

Professor

Corresponde aos docentes responsáveis pela criação das viagens virtuais. Uma vez que vários professores podem criar várias viagens e conteúdos, e estes possuem atributos próprios, foram considerados uma entidade necessária, tal como se evidencia no requisito **RD2**. Cada professor é caracterizado pelo seu número de professor, o identificador único, e pelo seu nome.

Disciplina

Permite associar as viagens aos conteúdos curriculares, facilitando a filtragem por área do conhecimento. Tal como descrito no **RD3**, as disciplinas são identificadas univocamente por um número artificial único, além disso, têm associadas os atributos que identificam o seu nome e o ano de escolaridade no qual são lecionadas. Por estes motivos, a disciplina foi criada como entidade autónoma.

Viagem

É a entidade central do sistema e representa um conjunto organizado de conteúdos virtuais disponibilizados aos alunos. Cada viagem possui um identificador único, um título, um resumo da viagem e a sua data de publicação. Através do requisito **RD4** é identificada como uma entidade.

Conteúdo

Segundo o requisito **RD5**, o conteúdo representa os elementos multimédia disponibilizados aos alunos no âmbito de cada viagem virtual. Cada conteúdo tem um título e uma breve descrição do conteúdo, além de incluir informação sobre a localização (nome, cidade e país) e data de captura. É também incluída uma identificação do tipo para distinguir se o conteúdo armazenado no URL é uma foto ou um vídeo. Deste modo, um conteúdo foi considerado como entidade autónoma por possuir propriedades próprias, como descrição e URL, e por variar mesmo quando associado à mesma localização.

Foi construída a seguinte tabela de caracterização das entidades identificadas:

Designação	Descrição	Sinônimo	Ocorrência	Requisito
Aluno	Pessoa que estuda na escola Novos Horizontes.	Estudante	Cada aluno tem um número de aluno único atribuído no ato da inscrição na escola.	RD1
Professor	Pessoa que é responsável por lecionar a matéria de uma determinada disciplina.	Docente	Cada professor tem um número de identificação único atribuído quando inicia funções na escola.	RD2
Disciplina	Unidade curricular associada às viagens, permitindo organizar e filtrar os conteúdos segundo as áreas lecionadas.	Unidade Curricular	Cada disciplina tem um nome e código próprios e pode estar associada a várias viagens.	RD3
Viagem	Entidade que surge para representar um conjunto de conteúdos que se relacionam e mapeiam os locais que um professor pretende mostrar.	Visita	Cada viagem é identificada por um código único e está associada a uma ou mais disciplinas.	RD4
Conteúdo	Elementos multimédia que fazem parte de uma viagem. Que podem ou não estar associados à mesma localização.	Multimédia	Cada conteúdo surge associado a uma localização, contendo o seu próprio identificador único.	RD5

Tabela 4 - Caracterização das Entidades

3.3 Identificação e Caracterização dos Relacionamentos

Após a definição das entidades principais, procedeu-se à identificação dos relacionamentos necessários para representar corretamente as interações entre os elementos do sistema. Cada relacionamento foi acrescentado ao modelo conceptual com base nos requisitos de descrição e nas regras de negócio recolhidas junto dos docentes, direção e funcionários da Escola Novos Horizontes.

Os relacionamentos identificados foram os seguintes:

Professor - Cria - Viagem

Os professores são responsáveis pela criação das viagens virtuais. Todos os professores da escola são armazenados na base de dados, pelo que, este pode ou não criar viagens. Por outro lado, uma viagem é criada por um único professor, tal como se verifica no requisito **RD6**. Deste modo, concluímos que se trata de um relacionamento de **1:N** com participação **total** do lado da **Viagem** e **parcial** do lado do **Professor**.

Aluno - Realiza - Viagem

Os alunos podem aceder às viagens e nesse caso têm de deixar uma avaliação. Como um aluno pode visualizar e avaliar várias viagens e cada viagem pode receber comentários de vários alunos, tal como descrito no requisito **RD7**, este é um relacionamento **N:M**, com atributos adicionais - avaliação, comentário e data de realização - associados ao relacionamento. A participação é **parcial em ambas as entidades**, pois um aluno pode não ter realizado viagens e uma viagem pode ainda não ter visualizações.

Viagem - Enquadra-se - Disciplina

Pelo requisito **RD8** verifica-se que cada viagem enquadra-se em uma ou mais disciplinas, permitindo fornecer uma primeira ideia dos temas abordados. Do mesmo modo, uma disciplina pode ter várias viagens. Assim, este relacionamento é **N:M**, uma vez que a mesma viagem pode ser relevante para diferentes disciplinas e uma disciplina pode conter múltiplas viagens. A participação da **Viagem** é **total** (tem de abordar pelo menos uma disciplina), enquanto a da **Disciplina** é **parcial** (pode não ter viagens associadas) dado que todas as disciplinas da escola estão armazenadas na base de dados.

Viagem - Contém - Conteúdo

Analizando o requisito **RD9** percebe-se que viagens são constituídas por um ou vários conteúdos e que cada conteúdo está associado apenas a uma viagem. Por isso, este relacionamento é **1:N**, dada a possibilidade de uma viagem ter vários conteúdos. Além disso, existe participação **total** de **ambas as partes**, pois uma viagem deve ter conteúdos para existir e um conteúdo não existe sem estar ligado a uma viagem.

Estes relacionamentos, representados na tabela 5, permitem caracterizar por completo as ligações funcionais do sistema e servem de base para a estrutura do modelo ER final.

Entidade	Relacionamento	Cardinalidade	Participação	Entidade	Requisito
Professor	cria	1:N	P:T	Viagem	RD6
Aluno	realiza	N:N	P:P	Viagem	RD7
Viagem	enquadra-se	N:N	T:P	Disciplina	RD8
Viagem	contém	1:N	T:T	Conteúdo	RD9

Tabela 5 - Caracterização dos Relacionamentos

3.4 Identificação e Caracterização dos Atributos das Entidades e dos Relacionamentos.

A identificação dos atributos das entidades e dos relacionamentos baseou-se na análise detalhada dos requisitos recolhidos junto da Escola Básica e Secundária Novos Horizontes. Este processo permitiu definir as informações essenciais que cada entidade deve conter, bem como os dados adicionais necessários para registar corretamente as interações entre os elementos do sistema.

Cada entidade possui atributos que garantem a sua identificação única e suportam as funcionalidades previstas na plataforma.

Segundo o **RD1** a entidade **Aluno** tem como seus atributos o **id_aluno** - o número único de aluno -, o nome, o género (foi utilizado um *ENUM* onde 'M' corresponde ao sexo masculino e 'F' ao feminino) e a turma, que por se tratar de um atributo composto, foi dividida em dois atributos simples o ano e a letra.

Entidade Aluno					
Atributo	Descrição	Domínio	Nulo	Tipo	Exemplo
id_aluno	Identificador único do aluno	INT	Não	Determinante	2001
nome	Nome do aluno	VARCHAR(100)	Não	Simples	'Jorge Ferreira Martins'
genero	Género do aluno	ENUM('M','F')	Não	Simples	'M'
turma	Turma do aluno	-	Não	Composto	-
ano	Ano de escolaridade da turma do aluno	TINYINT	Não	Simples	9
letra	Letra da turma do aluno	CHAR(1)	Não	Simples	'B'

Tabela 6 - Caracterização dos Atributos da entidade Aluno

A entidade **Professor** possui os atributos nome e id_professor, que representa o número único de professor, essencial para identificar cada docente que cria viagens virtuais, tal como é evidenciado pelo requisito **RD2**.

Entidade Professor					
Atributo	Descrição	Domínio	Nulo	Tipo	Exemplo
id_professor	Identificador único do professor	INT	Não	Determinante	5001
nome	Nome do professor	VARCHAR(100)	Não	Simples	'Orlando Belo'

Tabela 7 - Caracterização dos Atributos da entidade Professor

Através da análise do **RD3** concluímos que a entidade **Disciplina** apresenta um nome e um id_disciplina que, neste caso, é um número inteiro usado para identificar univocamente a disciplina. Além disso, também inclui o ano letivo no qual é lecionada.

Entidade Disciplina					
Atributo	Descrição	Domínio	Nulo	Tipo	Exemplo
id_viagem	Identificador único da disciplina	INT	Não	Determinante	715
nome	Nome da disciplina	VARCHAR(45)	Não	Simples	'Matemática A'
ano	Ano de escolaridade relativo a disciplina	TINYINT	Não	Simples	11

Tabela 8 - Caracterização dos Atributos da entidade Disciplina

A entidade **Viagem** contém atributos como o id_viagem, que é um número que funciona como o seu identificador único, e tem um título, um resumo e a sua data de publicação, tal como descrito no **RD4**.

Entidade Viagem					
Atributo	Descrição	Domínio	Nulo	Tipo	Exemplo
id_viagem	Identificador único da viagem	INT	Não	Determinante	1
título	Título da viagem	VARCHAR(100)	Não	Simples	'Os vulcões dos Açores'

Entidade Viagem

data_publicacao	Data em que a viagem foi publicada por um professor	DATE	Não	Simples	2025-11-18
resumo	Resumo da viagem	VARCHAR(1000)	Não	Simples	'Expedição virtual pelo arquipélago dos Açores'

Tabela 9 - Caracterização dos Atributos da entidade Viagem

Os atributos da entidade Conteúdo foram definidos com base no requisito **RD5**, que indica a necessidade de registar todas as informações associadas aos materiais multimédia incluídos nas viagens virtuais. Deste modo, tem-se que os atributos da entidade conteúdo são um identificador único, um URL, uma descrição, um título, um tipo (foi utilizado um ENUM, onde 'F' indica que é uma foto e 'V' indica que é um vídeo), uma data e uma localização de captura. Como a localização é um atributo composto deve ser dividida em três atributos simples, o país, a cidade e o nome.

Entidade Conteúdo

Atributo	Descrição	Domínio	Nulo	Tipo	Exemplo
id_conteudo	Identificador único do conteúdo	INT	Não	Determinante	18
url	URL do conteúdo	VARCHAR(150)	Não	Simples	'https://img1.jpg'
descricao	Descrição do conteúdo	VARCHAR(500)	Não	Simples	'Fotografia detalhada da vista da montanha do pico, evidenciando a beleza natural dos Açores'
titulo	Título do conteúdo	VARCHAR(50)	Não	Simples	'Vista da montanha do Pico'
tipo	Tipo do conteúdo	ENUM('F', 'V')	Não	Simples	'F'
data_captura	Data em que o conteúdo foi capturado	DATE	Não	Simples	2025-12-12
localizacao	Localização do conteúdo	-	Não	Composto	-
pais	País da localização do conteúdo	VARCHAR(35)	Não	Simples	'Portugal'
cidade	Cidade da localização do conteúdo	VARCHAR(35)	Não	Simples	'Açores'
nome	Nome da localização do conteúdo	VARCHAR(70)	Não	Simples	'Montanha do Pico'

Tabela 10 - Caracterização dos Atributos da entidade Conteúdo

Existe também um relacionamento que inclui atributos relevantes. No caso de **Aluno — Realiza — Viagem**, baseado no requisito **RD7**, foram definidos atributos para registar a avaliação, o comentário do aluno e a data em que a viagem foi realizada, permitindo acompanhar a interação de cada utilizador com os conteúdos.

Relacionamento: Aluno - Realiza - Viagem					
Atributo	Descrição	Domínio	Nulo	Tipo	Exemplo
avaliacao	Pontuação que o aluno escolhe para atribuir à viagem (de 0 até 10)	INT	Não	Simples	9
comentario	Comentário que o aluno fornece relativo à viagem	VARCHAR(200)	Não	Simples	'Viagem bastante educacional sobre as disciplinas abordadas'
data_realizacao	Data em que o aluno realizou a viagem	DATE	Não	Simples	2024-10-10

Tabela 11 - Caracterização dos Atributos do relacionamento Aluno-Realiza-Viagem

A definição clara dos atributos assegura que cada entidade e relacionamento esteja corretamente representado no modelo conceptual e que os dados essenciais possam ser registados de forma consistente e organizada.

3.5 Apresentação e Explicação do Diagrama ER Produzido

Com a definição e caracterização das entidades, dos seus atributos e dos relacionamentos, a equipa de desenvolvimento procedeu à construção do diagrama Entidade-Relacionamento. Para isso, foi utilizado o software **brModelo**, que permite criar diagramas conceptuais digitais de forma prática e clara.

O processo de elaboração do diagrama seguiu duas etapas principais. Inicialmente, foram inseridas todas as **entidades** juntamente com os seus atributos no **software**.

Posteriormente, foram adicionados **todos os relacionamentos**, incluindo a definição das **cardinalidades mínimas e máximas** e a **participação** de cada entidade. É importante destacar que a notação utilizada pelo **brModelo** apresenta algumas

diferenças em relação à notação clássica de Chen, especialmente na representação das cardinalidades.

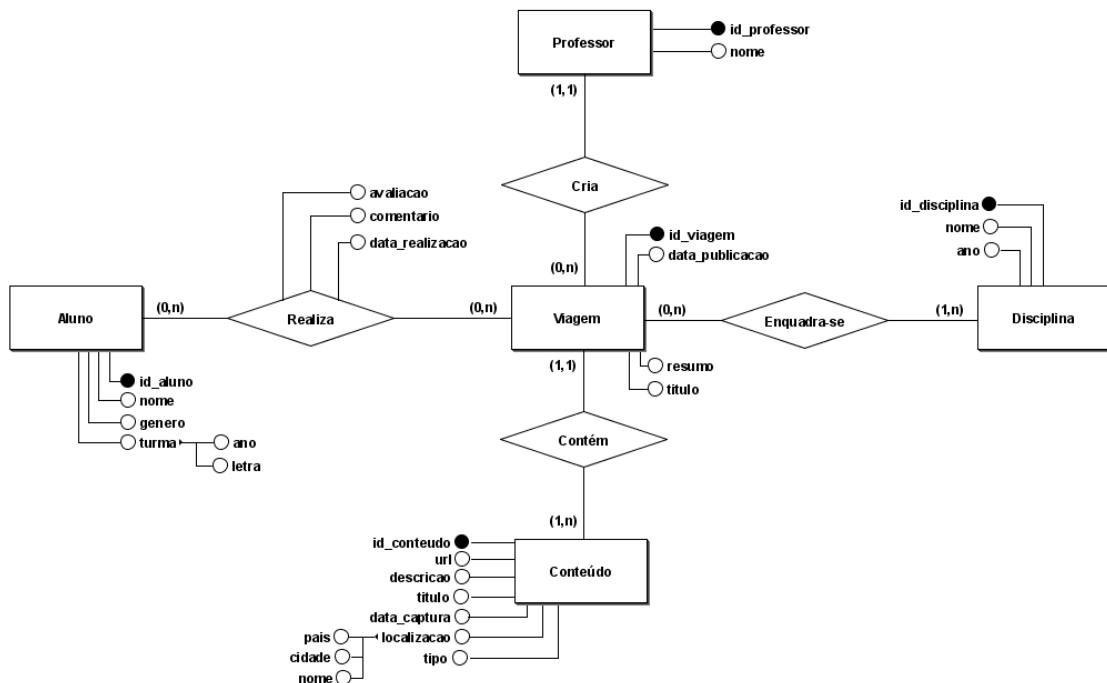


Figura 6 - Modelo conceptual

4. Modelação Lógica

4.1 Construção e Validação do Modelo de Dados Lógico

A equipa manteve a abordagem monovista utilizada para a construção do modelo conceptual dada a simplicidade do projeto em desenvolvimento. No contexto da modelação lógica, o modelo foi construído na sua íntegra de uma só vez, com base no diagrama ER previamente concebido.

A construção do modelo lógico foi realizada através da ferramenta **MySQL Workbench**, onde o modelo conceptual foi traduzido para um esquema lógico compatível com um *Sistema de Gestão de Bases de Dados* (SGBD) relacional.

Durante esta fase, foram tomadas diversas decisões estruturais para garantir a integridade e eficiência dos dados.

- **Definição de Chaves Primárias (PK):** Todas as tabelas possuem um atributo identificador que distingue diferentes registo univocamente.
- **Definição de Chaves Estrangeiras (FK):** Foram estabelecidas as relações entre as tabelas através de chaves estrangeiras, assegurando a integridade referencial (por exemplo, um **Conteúdo** não pode existir sem estar associado a uma **Viagem**).
- **Normalização e Tipagem:** O esquema foi desenhado para cumprir as formas normais, escolhendo-se domínios de dados que otimizam o armazenamento, como o uso de **ENUM** para domínios fechados, cujos detalhes específicos serão apresentados na secção seguinte.

4.2 Apresentação e Explicação do Modelo Lógico Produzido

Para a construção do esquema lógico, a equipa começou por criar uma relação para cada entidade identificada no modelo conceptual, inserindo-se os respectivos atributos simples. Os atributos compostos não entram para a relação apenas os seus sub-atributos. De seguida, procedeu-se ao mapeamento dos relacionamentos. A conversão do modelo conceptual para o lógico está dependente da cardinalidade da mesma.

No modelo construído verifica-se apenas a existência de relacionamentos binários (apenas envolvem duas entidades), sendo que entre eles observam-se dois tipos de cardinalidade diferentes: “muitos” para “muitos” e “muitos” para “um”. A conversão dos relacionamentos “muitos” para “um” ocorre através da inserção de uma chave estrangeira no lado da entidade cuja cardinalidade é N, sendo que esta referência a chave primária presente na entidade com cardinalidade 1. Os relacionamentos “muitos” para “muitos”, traduzem-se no modelo lógica na criação de uma relação a mais. A nova relação irá conter duas chaves estrangeiras cada uma a referenciar a chave primária de uma das entidades que fazem parte do relacionamento, sendo que essas em conjunto formam a chave primária da relação, que é composta (Connolly and Begg, 2015, p. 535).

A construção do modelo lógico teve em atenção uma prévia explicação das relações implementadas:

A tabela **Professor**, derivada da entidade com o mesmo nome, armazena os dados dos docentes, contendo os atributos “id_professor” (chave primária) e “nome”. Esta tabela serve de referência para a organização de viagens.

Atributo	Domínio	Nulo	Chave Estrangeira
id_professor	INT	Não	Não
nome	VARCHAR(100)	Não	Não

Tabela 12 - Dicionário de dados da tabela Professor

A tabela **Aluno** deriva da entidade homónima. Contém os atributos: “id_aluno”, “nome” e “genero”. O atributo composto “turma”, devido à conversão para o modelo lógico desaparece ficando apenas os seus sub-atributos “ano” e “letra”. O atributo “id_aluno”

foi definido como chave primária permitindo a sua identificação única, dado que este representa o número de aluno único atribuído pela escola.

Atributo	Domínio	Nulo	Chave Estrangeira
id_aluno	INT	Não	Não
nome	VARCHAR(100)	Não	Não
genero	ENUM('M', 'F')	Não	Não
ano	TINYINT	Não	Não
letra	CHAR(1)	Não	Não

Tabela 13 - Dicionário de dados da tabela Aluno

A tabela **Disciplina** provém da entidade com o mesmo nome identificada no modelo conceptual e serve para categorizar os conteúdos escolares. É constituída pelos atributos "id_disciplina" - a sua PK - , "nome" e "ano".

Atributo	Domínio	Nulo	Chave Estrangeira
id_disciplina	INT	Não	Não
nome	VARCHAR(45)	Não	Não
ano	TINYINT	Não	Não

Tabela 14 - Dicionário de dados da tabela Disciplina

A tabela **Viagem** tem origem na entidade do mesmo nome, centralizando a atividade principal do sistema. Os seus atributos simples foram mapeados para a relação: "id_viagem", "titulo", "resumo", "data_publicacao". Devido ao relacionamento N:1 com a entidade "Professor", foi adicionado o atributo "id_professor" como chave estrangeira, garantindo o relacionamento entre a instância da viagem e a instância do seu respectivo criador. A chave primária escolhida foi o "id_viagem", o identificador artificial criado com esse propósito.

Atributo	Domínio	Nulo	Chave Estrangeira
id_viagem	INT	Não	Não
titulo	VARCHAR(100)	Não	Não
data_publicacao	DATE	Não	Não
resumo	VARCHAR(1000)	Não	Não
id_professor	INT	Não	Sim

Tabela 15 - Dicionário de dados da tabela Viagem

No modelo conceptual, embora **Localizacao**, **Pais** e **Cidade** não tenham inicialmente sido identificados como entidades autónomas, mas apenas como atributos da entidade **Conteudo**, considerámos necessário, nesta fase, criar uma relação específica para

cada um deles. Esta decisão surgiu da necessidade de aumentar a consistência e a integridade dos dados armazenados.

No conceptual, a localização foi identificada como um atributo composto pelos atributos simples nomeadamente o nome, o país e a cidade. Contudo, existia uma grande facilidade de registos divergentes mencionarem a mesma entidade real: por exemplo, registar “Vila Nova de Famalicão” como cidade num conteúdo e, noutra, “Famalicão” ou “V. N. de Famalicão”. Esta flexibilidade excessiva levava a redundâncias, erros de escrita e perda de fiabilidade nas pesquisas.

Desse modo, a criação de relações independentes para a **Localizacao**, o **País** e a **Cidade**, cada uma com a sua chave primária, permite centralizar a informação e reduzir, ou idealmente eliminar, a divergência nos registos da localização dos conteúdos para os mesmos locais. Assim, cada cidade, país ou localização passa a existir uma única vez na base de dados, sendo referenciada de forma consistente por chaves estrangeiras.

Além disso, a separação evita anomalias, como por exemplo, de modificação de dados caso um nome de uma localização precise de ser alterado.

Assim, a relação **Localizacao** é composta por três atributos principais. O atributo “id_localizacao”, o identificador único de cada registo de localização, que funciona como chave primária da tabela. O atributo “nome” que corresponde ao nome específico da localização, e que se refere, por exemplo, a um local em concreto dentro de uma cidade, como uma rua, edifício ou monumento. Por outro lado, a **Localizacao** está relacionada com uma **Cidade**: uma localização pertence a uma cidade e uma cidade pode conter várias localizações associadas, ou seja, N:1. Desta forma, esta relação contém ainda um atributo “id_cidade” que atua como chave estrangeira e estabelece a ligação entre cada localização e a cidade à qual esta pertence. Por fim, a relação está também relacionada com a relação **Conteudo** pois constitui o local onde a foto ou o vídeo foi capturado. Assim estabelece-se um relacionamento (Localizacao)1:N(Conteudo), ficando a chave estrangeira na tabela **Conteudo** que referencia a chave primária da tabela **Localizacao**.

Atributo	Domínio	Nulo	Chave Estrangeira
id_localizacao	INT	Não	Não
nome	VARCHAR(70)	Não	Não
id_cidade	INT	Não	Sim

Tabela 16 - Dicionário de dados da tabela Localizacao

A relação **Cidade** inclui igualmente três atributos. O atributo “id_cidade” identifica de forma única cada cidade presente no sistema e que é a chave primária da tabela. O atributo “cidade” que contém o nome da cidade, como “Braga” ou “Lisboa”. O terceiro atributo, “id_pais”, é uma chave estrangeira que estabelece a ligação entre cada cidade e a chave primária do seu país correspondente. Este atributo foi colocado devido ao relacionamento 1:N entre “cidade” e “pais” (um “pais” tem várias cidades e uma “cidade” esta apenas num país)

Atributo	Domínio	Nulo	Chave Estrangeira
id_cidade	INT	Não	Não
cidade	VARCHAR(35)	Não	Não
id_pais	INT	Não	Sim

Tabela 17 - Dicionário de dados da tabela Cidade

Por fim, a relação **Pais** é constituída por dois atributos. O atributo “id_pais”, o identificador artificial único de cada país, que funciona como sua chave primária. E o atributo “pais” que guarda o nome do país, assegurando que esta informação é registada de forma padronizada e consistente.

Atributo	Domínio	Nulo	Chave Estrangeira
id_pais	INT	Não	Não
pais	VARCHAR(35)	Não	Não

Tabela 18 - Dicionário de dados da tabela Pais

A tabela **Conteudo** resulta do mapeamento direto das respetiva entidade do modelo conceptual. Esta possui os atributos “id_conteudo” (PK), “url”, “descricao”, “titulo”, “tipo” e “data”. Para garantir os relacionamentos 1:N que tem com **Viagem** e **Localizacao** possui duas chaves estrangeiras: “id_viagem”, que associa o conteúdo à viagem, e “id_localizacao”, que associa o conteúdo ao local onde foi captado.

Atributo	Domínio	Nulo	Chave Estrangeira
id_conteudo	INT	Não	Não
url	VARCHAR(150)	Não	Não
descricao	VARCHAR(500)	Não	Não
titulo	VARCHAR(50)	Não	Não
data_captura	DATE	Não	Não
tipo	ENUM('F','V')	Não	Não

id_viagem	INT	Não	Sim
id_localizacao	INT	Não	Sim

Tabela 19 - Dicionário de dados da tabela Conteudo

A criação da tabela **Aluno_Viagem** resulta da aplicação da regra de conversão para relacionamentos de cardinalidade N:M. Conforme identificado no Modelo Conceptual, no relacionamento 'Realiza', um aluno pode participar em múltiplas viagens e uma mesma viagem pode ser realizada por diversos alunos. Neste caso, a situação não pode ser resolvida com chaves estrangeiras nas entidades originais, exigindo a criação de uma nova tabela. Assim, criou-se esta relação contendo o "id_aluno" e o "id_viagem", chaves estrangeiras que referenciam as chaves primárias das tabelas **Aluno** e **Viagem**, respectivamente. Neste caso, a chave primária é composta pelo par de chaves estrangeiras (id_aluno, id_viagem), garantindo a unicidade da participação. Importa notar que esta relação foi enriquecida com os atributos "avaliacao", "comentario" e "data_realizacao", uma decisão que possibilita o registo do comentário no momento da realização da viagem, informações estas que dependem exclusivamente da interação entre as duas entidades.

Atributo	Domínio	Nulo	Chave Estrangeira
id_aluno	INT	Não	Sim
id_viagem	INT	Não	Sim
avaliacao	INT	Não	Não
comentario	VARCHAR(200)	Não	Não
data_realizacao	DATE	Não	Não

Tabela 20 - Dicionário de dados da tabela Aluno_Viagem

A criação da tabela **Viagem_Disciplina** deriva, mais uma vez, da aplicação da regra de conversão de relacionamentos de cardinalidade N:M. Conforme identificado no Modelo Conceptual, no relacionamento 'Enquadra-se', uma viagem pode enquadrar-se em múltiplas disciplinas e uma disciplina pode ser abordada em várias viagens. Assim, criou-se esta tabela associativa contendo o "id_viagem", a chave estrangeira que referencia a chave primária da tabela viagem, e o "id_disciplina", a chave estrangeira que referencia a chave primária da tabela disciplina. A chave primária desta nova relação é composta pela junção destas duas chaves estrangeiras (id_viagem, id_disciplina), garantindo que a mesma disciplina não é associada à mesma viagem mais do que uma vez.

Atributo	Domínio	Nulo	Chave Estrangeira
id_viagem	INT	Não	Sim
id_disciplina	INT	Não	Sim

Tabela 21 - Dicionário de dados da tabela Viagem_Disciplina

Após a definição individual de todas as relações, atributos e restrições de integridade, procedeu-se à unificação do esquema. Com o auxílio da ferramenta **MySQL Workbench**, consolidaram-se todas as tabelas e relacionamentos num único esquema visual, resultando no modelo lógico final apresentado na figura seguinte:

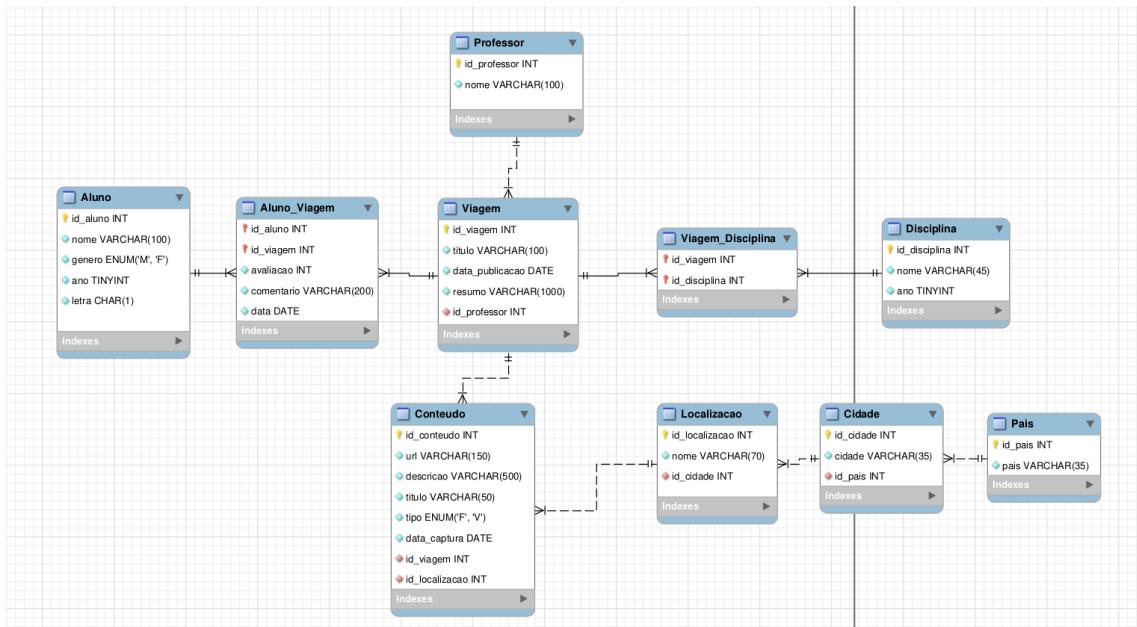


Figura 7 - Modelo Lógico

4.3 Normalização de Dados

O processo de normalização constitui uma etapa fundamental na modelação de bases de dados, cujo principal objetivo é garantir a integridade dos dados, reduzir a redundância e evitar anomalias de inserção, atualização e remoção. Este processo organiza as relações de forma lógica, assegurando que cada tabela representa apenas uma entidade ou relação relevante e que todos os seus atributos estão corretamente estruturados. As formas normais constituem um conjunto de métodos específicos que permitem avaliar e verificar o nível de eficiência do modelo lógico. No presente trabalho, a normalização foi verificada até à Terceira Forma Normal (3FN), garantindo que o sistema desenvolvido mantém uma estrutura consistente.

A *Primeira Forma Normal* (1FN) exige que exista uma chave primária na relação, que todos os atributos de uma relação sejam atómicos, ou seja, não possam ser subdivididos, e que não existam grupos repetitivos ou listas de valores dentro de uma única coluna. No modelo desenvolvido, todas as relações cumprem esta condição. Por exemplo, na tabela “Aluno”, atributos como nome, género são todos valores indivisíveis. Um cenário que violaria a 1FN seria, por exemplo, se existisse uma coluna chamada “contactos_telefone” contendo um conjunto de números telefónicos, algo que não ocorre no modelo final.

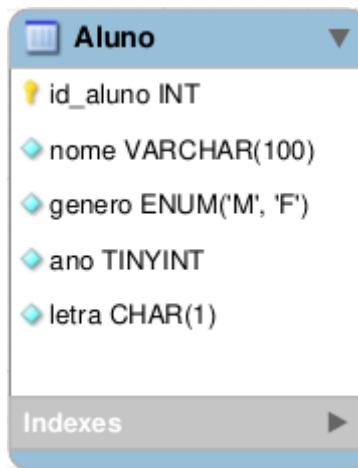


Figura 8 - Tabela Aluno

Outro exemplo relevante surge na relação entre viagens e disciplinas. Inicialmente, poderíamos imaginar um atributo multivalorado “disciplinas_associadas” dentro da tabela “Viagem”, que armazenaria um conjunto de várias disciplinas abordadas nessa viagem, o que violaria a 1FN. No nosso modelo não existem atributos multi-valorados, “Disciplina” é uma entidade autónoma e a associação entre “Viagem” e “Disciplina” é gerida por uma tabela de ligação “Viagem_Disciplina” que representa o relacionamento muitos-para-muitos (N:M) entre ambas entidades, pelo que o modelo cumpre a *Primeira Forma Normal* (1FN).

A *Segunda Forma Normal* (2FN) exige que a tabela esteja na 1FN e coloca-se em causa apenas em tabelas cuja chave primária é composta, exigindo que todos os atributos não-chave devem depender na totalidade dessa chave, de modo a evitar dependências funcionais parciais. No modelo desenvolvido, a maioria das tabelas tem chaves primárias simples, como **Aluno**, **Professor**, **Viagem**, **Conteúdo**, **Disciplina**,

Localizacao, Cidade, e País pelo que cumprem automaticamente esta condição. As verificações mais relevantes incidem, portanto, sobre relações com chave primária composta. No caso de **Aluno_Viagem**, cuja chave primária é composta por (id_aluno, id_viagem), foram analisados os atributos adicionais “avaliação”, “comentário” e “data”. Verifica-se que estes atributos dependem unicamente da combinação dos dois identificadores, isto é, da realização específica de uma viagem por um aluno, e não apenas de um deles isoladamente. O mesmo ocorre na tabela **Viagem_Disciplina**, onde a chave primária composta representa a relação entre uma viagem e uma disciplina, mas não existem quaisquer atributos adicionais suscetíveis à criação de dependências funcionais parciais. Assim, todas as relações que exigem verificação encontram-se corretamente estruturadas segundo as regras da *Segunda Forma Normal* (2FN).

Aluno_Viagem	
!	id_aluno INT
!	id_viagem INT
!	avaliacao INT
!	comentario VARCHAR(200)
!	data DATE
Indexes	

Figura 9 - Tabela Aluno_Viagem

Viagem_aborda_Disciplina	
!	id_viagem INT
!	id_disciplina INT
Indexes	

Figura 10 - Tabela Viagem_Disciplina

A *Terceira Forma Normal* (3FN) exige que uma tabela esteja na 1FN, na 2FN e que não existam dependências funcionais transitivas entre atributos não-chave. Em termos práticos, isto significa que um atributo não-chave não pode depender funcionalmente de outro atributo não-chave, garantindo que cada atributo depende exclusivamente da chave primária. No modelo desenvolvido, todas as relações cumprem a *Terceira Forma Normal* (3FN), a título de exemplo, podemos olhar para a relação **Viagem**: os atributos desta entidade como “*título*”, “*data_publicação*” e “*resumo*” dependem exclusivamente do “*id_viagem*”, que é a sua chave primária.

Conclui-se, portanto, que o modelo lógico desenvolvido cumpre as três primeiras formas normais. Em todas as relações verifica-se que a 1FN, a 2FN e a 3FN são respeitadas, ou seja, o esquema está normalizado até a *Terceira Forma Normal* (3FN).

4.4 Validação do Modelo com Interrogações do Utilizador

A validação do modelo lógico foi realizada com base nos requisitos de manipulação definidos para o sistema, avaliando se a estrutura proposta é capaz de responder corretamente às interrogações colocadas pelos utilizadores. Assim, recorreu-se à álgebra relacional como ferramenta de verificação da consistência do modelo.

Com o objetivo de evitar erros de raciocínio e garantir maior rigor na validação, foi utilizada a calculadora de álgebra relacional, o **RelaX**. Numa fase inicial, o esquema das tabelas foi definido na ferramenta e a base de dados foi posteriormente povoada com dados de teste fictícios, mas de acordo com o modelo desenvolvido, permitindo simular cenários realistas de utilização.

O script com o povoamento utilizado no RelaX encontra-se, na sua totalidade, nos anexos do relatório.

Para a validação do modelo lógico, foram selecionados quatro requisitos de manipulação que variam na sua complexidade. A escolha destes requisitos permitiu testar operações da álgebra relacional, como seleções, projeções, junções, ordenações e agregações, desse modo, diversificando o conjunto de testes realizados para assegurar a capacidade do modelo para responder às necessidades dos utilizadores.

Começando por analisar o **RM7**. O seu objetivo é listar todas as viagens associadas a uma determinada disciplina, dado o seu identificador, neste exemplo consideramos o identificador ‘2’, ordenadas da mais recente para a mais antiga, apresentando o título, o resumo, o identificador da viagem e data de publicação. Este requisito exige inicialmente uma seleção pelo “id_disciplina” na tabela **Viagem_Disciplina**, seguida de uma equijunção com a relação **Viagem**. Para obter as viagens da mais recente para a mais antiga aplicamos uma ordenação decrescente pelo atributo “data_publicação”. No fim, é realizada uma projeção dos atributos desejados, neste caso, o “id_viagem”, o “título”, o “resumo” e a “data_publicacao”.

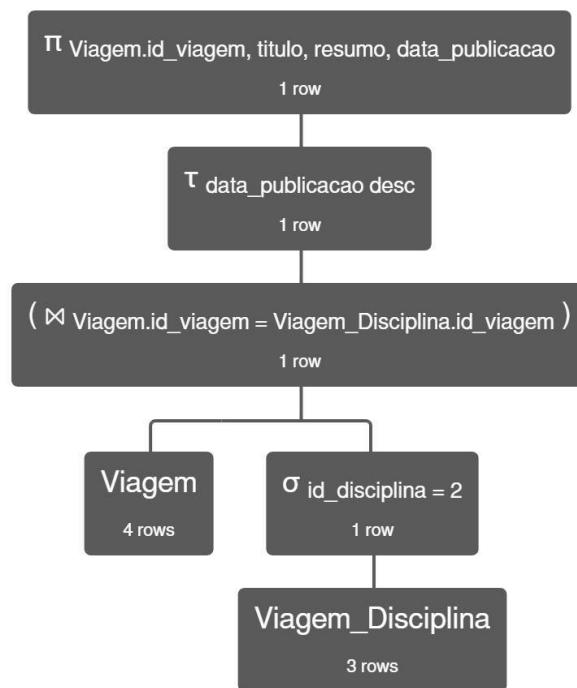


Figura 11 - Árvore da interrogação do RM7

```

Π_{Viagem.id_viagem, titulo, resumo, data_publicacao} ( τ_{data_publicacao desc} (Viagem ⋈
Viagem.id_viagem = Viagem_Disciplina.id_viagem ( σ{id_disciplina = 2
(Viagem_Disciplina) }))
  
```

De seguida, foi analisado o **RM8**, que consiste em listar os alunos, mostrar o seu identificador e nome, que realizaram uma determinada viagem, identificada pelo respetivo ID, e apresentar a avaliação atribuída por cada aluno, ordenando os registo por ordem alfabética do nome do aluno. Este requisito envolve uma operação de seleção sobre a viagem pretendida na relação **Aluno_Viagem**, seguida de uma junção

interna entre a relação com a tabela **Aluno**, bem como uma ordenação. Por fim, basta realizar uma projeção dos atributos que pretendem ser visualizados como resultado, no caso, "id_aluno", "nome" e "avaliacao".

A tradução deste requisito para álgebra relacional resultou na seguinte árvore:

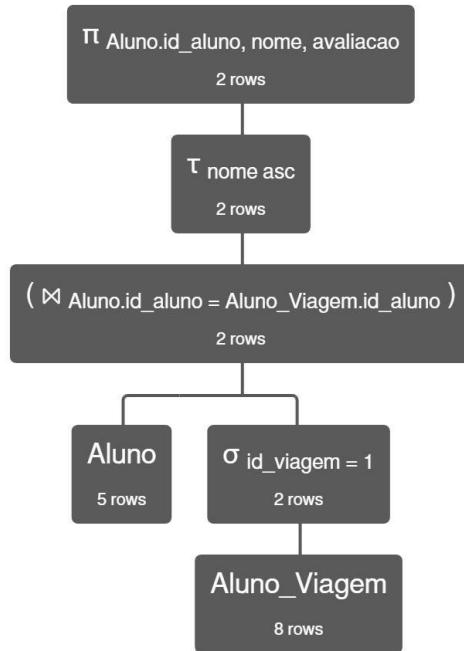


Figura 12 - Árvore da interrogação do RM8

```

Π Aluno.id_aluno, nome, avaliacao ( τ nome asc (
  Aluno ⋈ Aluno.id_aluno = Aluno_Viagem.id_aluno ( σ id_viagem = 1 (Aluno_Viagem) )))
  
```

O **RM10** apresenta a necessidade de utilizar um operador diferente, pois requer o cálculo da classificação média de todas as viagens. A sua tradução para álgebra relacional implica, inicialmente, uma junção natural entre **Viagem** e **Aluno_Viagem**. Sendo que o único atributo que estas relações têm com o mesmo nome é "id_viagem", é baseado nele que a junção será feita. De seguida, aplicamos uma operação de agregação segundo o atributo "id_viagem". Feita a agregação devemos utilizar as funções aplicadas sobre a agregação para os restantes atributos, assim, utilizamos "MIN(Viagem.titulo)" para mostrar o título da viagem e calculamos a média através da função "AVG" sobre o atributo "avaliacao", renomeando os resultados para 'Titulo' e 'Media', respetivamente. Por fim, é aplicada uma operação de ordenação decrescente

sobre o atributo renomeado “Media”, garantindo que as viagens com melhor classificação apareçam no topo da listagem.

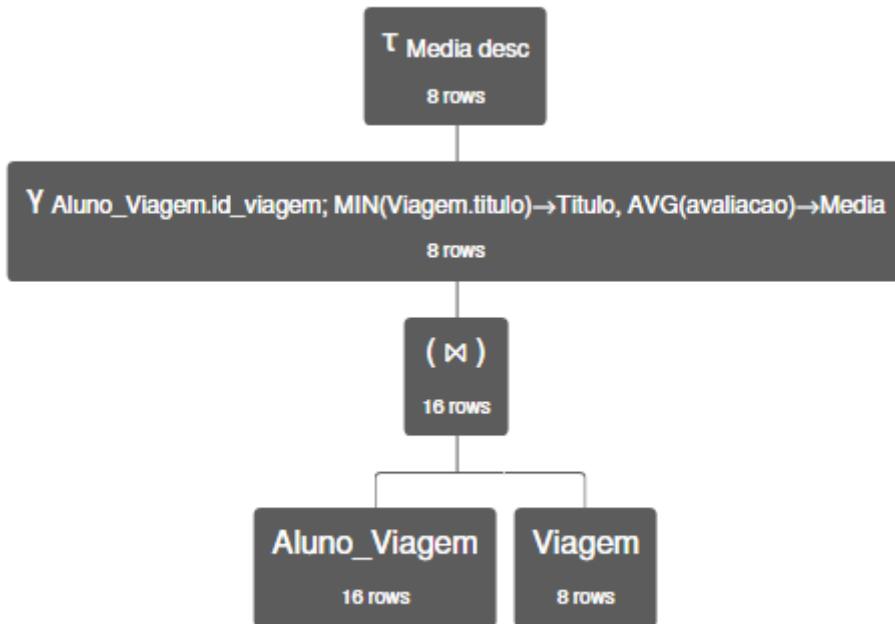


Figura 13 - Árvore da interrogação do RM10

```

T Media desc ( Y Aluno_Viagem.id_viagem; MIN(Viagem.titulo)→Titulo, AVG(avaliacao)→Media
                (Aluno_Viagem ⋈ Viagem) )
  
```

Por fim, o **RM11** corresponde ao requisito mais complexo dos selecionados. Este requisito consiste em gerar uma listagem de todos os professores, acompanhada pelo número total de viagens virtuais criadas por cada um, ordenada de forma decrescente por esse valor, adicionando ainda a média mais alta e mais baixa das suas viagens. A sua implementação em álgebra relacional envolve junções, operações de agrupamento, contagem e ordenação. Começamos por identificar as tabelas necessárias para dar resposta a esta interrogação, no caso, **Professor**, **Viagem** e **Aluno_Viagem**. Realizou-se de seguida uma junção externa à esquerda entre **Professor** e **Viagem** com base no atributo “id_professor”, uma vez que este é comum às duas tabelas, permitindo ficar com todas as instâncias da tabela dos professores, incluindo aqueles que não fizeram viagens. Ao mesmo tempo, na tabela

Aluno_Viagem foi feita uma agregação com base no “id_vagem” e calculamos as médias de todas as viagens com a função “AVG”. Depois realizou-se uma junção externa à direita entre os resultados das operações anteriores, de modo a assegurar que os registos dos professores sem viagens sejam mantidos. Para obter os valores pretendidos para cada professor foi depois necessário agrregar os registos obtidos pelo “id_professor”. Usamos a função “COUNT” para obter o número de viagens total, a função “MAX” para obter a maior nota média e a função “MIN” para obter a menor nota média, assim como o nome do professor. Por fim, apenas foi necessário ordenar por ordem decrescente de número de viagens realizadas para a interrogação ficar completa.

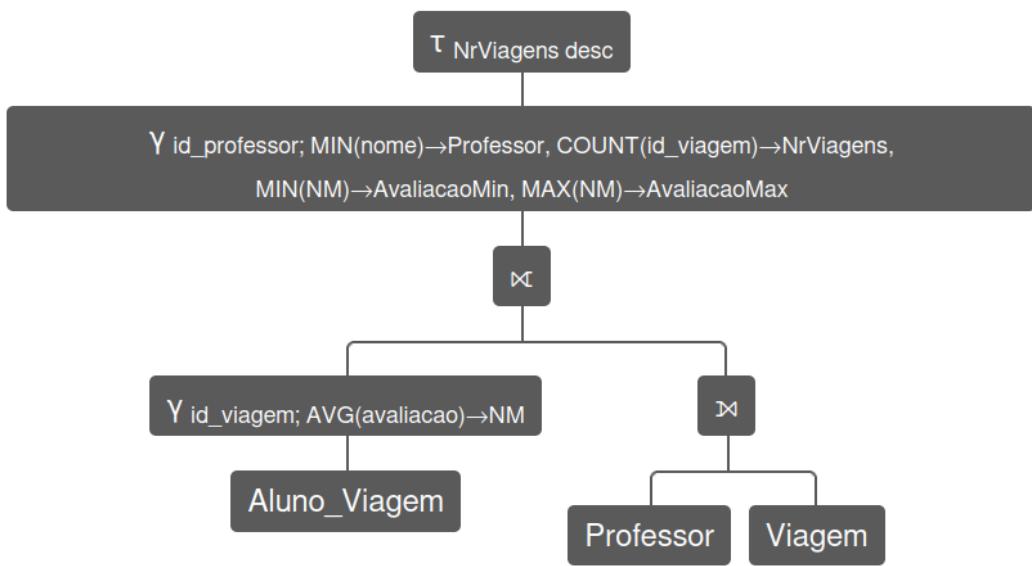


Figura 14 - Árvore da interrogação do RM11

```

    T NrViagens desc ( Y id_professor; MIN(nome)→Professor, COUNT(id_vagem)→NrViagens,
    MIN(NM)→AvaliacaoMin, MAX(NM)→AvaliacaoMax ( Y id_vagem; AVG(avaliacao)→NM (Aluno_Viagem) <-
    (Professor <--> Viagem) ) )
  
```

A correta implementação de todas as interrogações testadas demonstrou que o modelo lógico suporta adequadamente os requisitos definidos, sendo, por isso, considerado válido e apto para avançar para a fase de implementação física.

5. Implementação Física

5.1 Apresentação e explicação da base de dados implementada

O modelo físico da base de dados para a escola Novos Horizontes foi materializado através de um *script Structured Query Language* (SQL), desenhado para ser interpretado pelo SGBD MySQL. A construção deste *script* tem como objetivo refletir as decisões estruturais tomadas durante a fase de modelação lógica e, desse modo, assegurar a integridade referencial, os domínios dos dados e o cumprimento dos requisitos levantados.

O processo de implementação iniciou-se com a definição do ambiente da base de dados. Começamos por deixar em comentário a instrução que permite eliminar a base de dados, para em caso de necessidade facilitar a sua utilização. A primeira instrução do *script* assegura a criação da base de dados **ViagensNH**, definindo explicitamente o conjunto de caracteres (*CHARACTER SET*) como **utf8**. Esta decisão técnica é fundamental para garantir o suporte nativo à acentuação da língua portuguesa e a caracteres especiais, essenciais tanto para o registo correto dos nomes dos alunos como para os conteúdos educativos das viagens. Foi adicionado ainda o “*COLLATE utf8mb4_unicode_ci*” que se trata de um conjunto de regras que define como os caracteres são comparados e ordenados. O “*utf8mb4_unicode_ci*” é uma regra de ordenação baseada no padrão *Unicode* e *ci - Case Insensitive* - serve para especificar que as letras maiúsculas e minúsculas não são distinguíveis, isto é, uma procura por “braga” e “Braga” deve produzir o mesmo resultado, permitindo assim evitar inconsistências na base de dados e dar uma maior flexibilidade nas operações de pesquisa e comparação de texto.

Utilizou-se ainda a cláusula “*IF NOT EXISTS*” para conferir robustez ao *script*, prevenindo erros de execução caso a base de dados já se encontre criada no servidor.

```
-- DROP DATABASE ViagensNH;
```

```
CREATE DATABASE IF NOT EXISTS ViagensNH
```

```
CHARACTER SET utf8mb4
```

```
COLLATE utf8mb4_unicode_ci;  
  
USE ViagensNH;
```

O comando “*USE ViagensNH*” permite selecionar a BD **ViagensNH** para ser utilizada, tornando-a a base de dados sobre a qual os comandos SQL vão atuar.

Para a criação das tabelas, utilizamos o comando “*CREATE TABLE IF NOT EXISTS*”, que permite criar a tabela apenas quando ela ainda não existe, evitando conflitos e garantindo que o script de criação pode ser executado várias vezes sem gerar erros.

Depois da criação da base de dados passamos para a criação física das diferentes relações. Para garantir uma correta passagem do modelo lógico para o modelo físico, procedeu-se à conversão de cada entidade e relacionamento definidos no modelo conceptual e posteriormente definidos em relações no modelo lógico em tabelas do MySQL, respeitando as respectivas chaves primárias e chaves estrangeiras. Optamos por manter o nome dados às relações e aos atributos definidos no modelo lógico e também os domínios definidos para cada atributo, de modo a garantir clareza, consistência e um mapeamento direto entre as diferentes fases do projeto.

Nesta fase foi também fundamental estabelecer uma ordem rigorosa na criação das tabelas que respeitasse as dependências das chaves estrangeiras. De modo a respeitar a ordem estipulada pelo princípio de integridade referencial optamos por criar a tabela “filha”, tabela que contém a chave estrangeira, depois da tabela “pai”, tabela que contém a chave primária referenciada. A criação na ordem contrária implicaria a definição das chaves estrangeiras no final da criação das relações com os restantes dos atributos. Deste modo, nesta fase foi fundamental definir e respeitar uma ordem coerente na criação das tabelas, de forma a garantir o cumprimento das dependências impostas pelas chaves estrangeiras e assegurar a integridade referencial da base de dados.

Assim, a criação do esquema iniciou-se por duas entidades base do sistema, o **Professor** e **Aluno**, que não dependem de quaisquer outras tabelas. De seguida, foi criada a entidade **Viagem**, que depende da entidade **Professor**, uma vez que cada viagem está associada ao docente que a criou. Após a existência destas entidades, tornou-se possível definir a tabela **Aluno_Viagem**, que estabelece a relação entre os alunos e as viagens, pelo que depende simultaneamente das tabelas **Aluno** e **Viagem**. Posteriormente, foi criada a entidade **Disciplina**, que tal como as relações **Aluno** e **Professor** são independentes das restantes. A criação da tabela **Disciplina**

permitiu criar a definição da tabela **Viagem_Disciplina**, que surge devido à relação muitos-para-muitos entre as viagens e as disciplinas. De seguida, procedemos à criação das tabelas relacionadas com a componente geográfica do sistema, iniciando pela tabela **Pais**, seguida de **Cidade**, e depois **Localizacao**, respeitando a associação das chaves estrangeiras entre as tabelas. Por fim, foi criada a tabela **Conteudo**, que depende das tabelas **Viagem** e **Localizacao**, sendo, por esse motivo, definida apenas após a existência prévia de ambas. Desta forma, a sequência adotada garante que todas as chaves estrangeiras referenciam tabelas já existentes, assegurando a consistência e a integridade do esquema global da base de dados.

Seguindo esta lógica, começamos pela tabela **Professor**, representativa da entidade responsável pela criação de conteúdos. Como esta tabela não depende de nenhuma outra, foi a primeira a ser instanciada, contendo a chave primária “id_professor” e o “nome” do docente. De acordo com o **RD2**, o identificador do professor diz respeito ao número de docente, pelo que, não se utilizou a propriedade “**AUTO_INCREMENT**” de modo a evitar a inserção de registo sem especificar o número de professor correto.

```
CREATE TABLE IF NOT EXISTS Professor (
    id_professor INT NOT NULL,
    nome VARCHAR (100) NOT NULL,
    PRIMARY KEY (id_professor)
);
```

Segundo a ordem estabelecida, implementou-se a tabela **Aluno**. Tal como na tabela **Professor**, não se utilizou a propriedade “**AUTO_INCREMENT**” para a sua chave primária “id_aluno” uma vez que, segundo o **RD1**, este é atribuído pela escola:

```
CREATE TABLE IF NOT EXISTS Aluno (
    id_aluno INT NOT NULL,
    nome VARCHAR(100) NOT NULL,
    genero ENUM('M', 'F') NOT NULL,
    ano TINYINT NOT NULL,
    letra CHAR(1) NOT NULL,
    PRIMARY KEY (id_aluno)
);
```

Assegurada a existência da tabela dos criadores das viagens, **Professor**, procedemos à criação da tabela central do sistema **Viagem**, com uma chave estrangeira a referenciar a chave primária da tabela **Professor**. Nesta tabela, a chave primária,

“*id_viagem*”, foi definida com a propriedade “*AUTO_INCREMENT*”, atribuindo ao SGBD a responsabilidade de gerar automaticamente identificadores artificiais únicos, através da atribuição de valores incrementais.

No atributo *data_publicacao* optou-se por definir um valor por padrão para que, no caso de não ser especificada a data, seja automaticamente atribuído o valor do dia no qual ela foi inserida.

Foram ainda incluídas as opções “*ON DELETE NO ACTION* e *ON UPDATE CASCADE*” na chave estrangeira “*id_professor*”. A primeira corresponde ao comportamento padrão do SGBD e impede a eliminação de um professor enquanto existirem viagens associadas, evitando a remoção acidental de informação dependente e consequente quebra da integridade referencial na base de dados. A segunda, não sendo o comportamento por omissão, foi utilizada de forma a garantir que eventuais alterações ao código do professor, devido a possíveis reestruturações na escola, são automaticamente refletidas na tabela **Viagem**, mais uma vez, com o objetivo de preservar a integridade referencial da base de dados.

```
CREATE TABLE IF NOT EXISTS Viagem (
    id_viagem INT AUTO_INCREMENT NOT NULL,
    titulo VARCHAR(100) NOT NULL,
    data_publicacao DATE NOT NULL DEFAULT (CURRENT_DATE),
    resumo VARCHAR(1000) NOT NULL,
    id_professor INT NOT NULL,
    PRIMARY KEY (id_viagem),
    FOREIGN KEY (id_professor)
        REFERENCES Professor(id_professor)
        ON DELETE NO ACTION
        ON UPDATE CASCADE
);
```

Seguindo a ordem estabelecida, após a criação da tabela **Viagem**, procedeu-se à implementação da tabela **Aluno_Viagem**, responsável por relacionar os alunos e as suas viagens realizadas. Como se trata de uma relação criada devido a um relacionamento muitos-para-muitos entre as entidades **Aluno** e **Viagem**, a sua chave primária é composta pelo par de atributos “*id_aluno*” e “*id_viagem*”, atributos estes que são, simultaneamente, chaves estrangeiras que referenciam a chave primária da tabela que representam as suas respectivas entidades que participam no relacionamento.

Esta tabela inclui ainda atributos adicionais que caracterizam a relação, nomeadamente a avaliação atribuída pelo aluno, um comentário e a data de realização da viagem. Para o atributo “avaliacao” foi definida uma restrição “CHECK”, limitando os valores possíveis ao intervalo entre 0 e 10, aplicando a regra de negócio definida pela escola, descrita no **RD7**. No atributo “data_realizacao” tal como no atributo “data_publicacao” da viagem, optou-se por definir como valor por padrão a data atual no momento de inserção da viagem.

Na chave estrangeira “id_aluno” optou-se por utilizar a opção “*ON DELETE NO ACTION*” na referência ao aluno, impedindo a eliminação da instância de um aluno que tenha um relacionamento com alguma instância das viagens. Por sua vez, a opção “*ON UPDATE CASCADE*” permite que, caso o código de um aluno seja alterado na tabela **Aluno**, por exemplo, devido a qualquer reestruturação, a alteração seja propagada automaticamente para a tabela **Aluno_Viagem**, preservando a integridade referencial. Na referência à viagem, na chave estrangeira “id_viagem” foi utilizada a opção *ON DELETE CASCADE*, garantindo que, caso uma viagem seja removida do sistema, todas as instâncias da tabela do relacionamento que referenciam a viagem apagada sejam automaticamente eliminadas. Deste modo, é possível eliminar viagens e não pôr em causa a integridade referencial das chaves estrangeiras na tabela **Aluno_Viagem**. A opção “*ON UPDATE NO ACTION*” foi escolhida porque o identificador da viagem é definido como “*AUTO_INCREMENT*” e não existe razão para que este seja alterado. Assim, é gerado um erro caso exista uma tentativa de modificar um “id_viagem” que tenha uma chave estrangeira a referenciá-lo.

```
CREATE TABLE IF NOT EXISTS Aluno_Viagem (
    id_aluno INT NOT NULL,
    id_viagem INT NOT NULL,
    avaliacao INT NOT NULL CHECK (avaliacao BETWEEN 0 AND 10),
    comentario VARCHAR(200) NOT NULL,
    data_realizacao DATE NOT NULL DEFAULT (CURRENT_DATE),
    PRIMARY KEY(id_aluno,id_viagem),
    FOREIGN KEY (id_aluno)
    REFERENCES Aluno(id_aluno)
    ON DELETE NO ACTION
    ON UPDATE CASCADE,
```

```

        FOREIGN KEY (id_viagem)
    REFERENCES Viagem(id_viagem)
        ON DELETE CASCADE
        ON UPDATE NO ACTION
    ) ;

```

De seguida, criamos a tabela **Disciplina**, que representa uma entidade independente das restantes entidades do nosso modelo. A sua chave primária, “*id_disciplina*”, foi definida com a propriedade “*AUTO_INCREMENT*” pelo mesmo motivo citado anteriormente.

```

CREATE TABLE IF NOT EXISTS Disciplina (
    id_disciplina INT NOT NULL AUTO_INCREMENT,
    nome VARCHAR(45) NOT NULL,
    ano TINYINT NOT NULL,
    PRIMARY KEY (id_disciplina)
) ;

```

Posteriormente, criamos a tabela **Viagem_Disciplina**, que resulta da relação muitos-para-muitos entre as entidades **Viagem** e **Disciplina**. À semelhança da tabela **Aluno_Viagem**, foi definida uma chave primária composta por dois atributos, o “*id_viagem*” e o “*id_disciplina*”, que são também chaves estrangeiras que referenciam as chaves primárias das tabelas **Viagem** e **Disciplina**, respectivamente. No que diz respeito às opções associadas às chaves estrangeiras, utilizou-se “*ON DELETE CASCADE*” na referência à tabela **Viagem**, assegurando que a eliminação de uma viagem implica automaticamente a remoção das instâncias que referenciam a mesma na tabela do relacionamento. Por outro lado, na referência à tabela **Disciplina** foi utilizada a opção “*ON DELETE NO ACTION*”, prevenindo a eliminação de uma disciplina enquanto esta se encontrar associada a uma ou mais viagens. Em ambos os casos, a opção “*ON UPDATE NO ACTION*” foi utilizada uma vez que as chaves primárias que são referenciadas foram criadas com “*AUTO_INCREMENT*” pelo que não deverão ser alteradas, de modo a evitar a perda dos relacionamentos entre as viagens e as suas respectivas disciplinas.

```

CREATE TABLE IF NOT EXISTS Viagem_Disciplina (
    id_viagem INT NOT NULL,
    id_disciplina INT NOT NULL,
    PRIMARY KEY (id_viagem, id_disciplina),

```

```

        FOREIGN KEY (id_viagem)
    REFERENCES Viagem(id_viagem)
        ON DELETE CASCADE
        ON UPDATE NO ACTION,
        FOREIGN KEY (id_disciplina)
    REFERENCES Disciplina(id_disciplina)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
    ) ;

```

Segundo a ordem estabelecida segue-se a informação geográfica do sistema. Assim, procedemos à criação das tabelas **Pais** e **Cidade**, estabelecendo uma hierarquia clara entre ambas. A tabela **Pais** foi criada em primeiro lugar, dado que é independente. A tabela **Cidade** foi posteriormente implementada, contendo uma chave estrangeira que referencia a tabela Pais. As opções “*ON DELETE NO ACTION*” e “*ON UPDATE NO ACTION*” foram utilizadas de modo a impedir a eliminação ou alteração de um país que possua cidades associadas, preservando assim a coerência dos dados geográficos.

```

CREATE TABLE IF NOT EXISTS Pais (
    id_pais INT NOT NULL AUTO_INCREMENT,
    pais VARCHAR(35) NOT NULL,
    PRIMARY KEY (id_pais)
);

CREATE TABLE IF NOT EXISTS Cidade (
    id_cidade INT NOT NULL AUTO_INCREMENT,
    cidade VARCHAR(35) NOT NULL,
    id_pais INT NOT NULL,
    PRIMARY KEY (id_cidade),
    FOREIGN KEY (id_pais)
    REFERENCES Pais(id_pais)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);

```

Seguindo a hierarquia, foi criada a tabela **Localizacao**, que depende da tabela **Cidade**. Cada localização está associada a uma única cidade, sendo novamente utilizadas opções para evitar a eliminação cidades que ainda têm localizações associadas a elas, “*ON DELETE NO ACTION*”, e evitar atualizações à chave primária que esteja a ser referenciada, “*ON UPDATE NO ACTION*”.

```
CREATE TABLE IF NOT EXISTS Localizacao (
    id_localizacao INT NOT NULL AUTO_INCREMENT,
    nome VARCHAR(70) NOT NULL,
    id_cidade INT NOT NULL,
    PRIMARY KEY (id_localizacao),
    FOREIGN KEY (id_cidade)
        REFERENCES Cidade(id_cidade)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);
```

Cada uma destas tabelas é composta pelo seu identificador artificial, o “*id*”, e pelo valor associado a esse *id*, o “*país*”, a “*cidade*” e o “*nome*” no caso da localização.

Por fim, foi implementada a tabela **Conteudo**, tabela cujas instâncias só existem se tiver uma instância nas tabelas **Viagem** e **Localizacao** para lhe ser associada. A sua chave primária foi definida com “*AUTO_INCREMENT*”, permitindo ao SGBD a geração automática de identificadores artificiais únicos.

Além disso, a tabela inclui o atributo “*tipo*”, definido como um “*ENUM*”, restringindo os valores possíveis a fotografia, ‘F’, e vídeo,’V’ , para assegurar a validade e identificar os tipos de conteúdo armazenados. No que diz respeito às chaves estrangeiras, foi utilizada a opção “*ON DELETE CASCADE*” na chave estrangeira referente à tabela **Viagem**, garantindo que todos os conteúdos associados são automaticamente eliminados quando uma viagem é removida. Já na referência à tabela **Localizacao**, foi utilizada a opção “*ON DELETE NO ACTION*”, impedindo a eliminação de uma localização enquanto esta estiver associada a conteúdos. Em ambas as chaves estrangeiras foi usada a opção “*ON UPDATE NO ACTION*” evitando mudanças de valor na chave primária “*id_viagem*” da tabela **Viagem** e “*id_localizacao*” da tabela **Localizacao** caso estas estejam a ser referenciadas por alguma instância da tabela **Conteudo**.

```

CREATE TABLE IF NOT EXISTS Conteudo (
    id_conteudo INT NOT NULL AUTO_INCREMENT,
        url VARCHAR(150) NOT NULL,
        descricao VARCHAR(500) NOT NULL,
        titulo VARCHAR(50) NOT NULL,
        tipo ENUM('F', 'V') NOT NULL,
        data_captura DATE NOT NULL,
        id_viagem INT NOT NULL,
        id_localizacao INT NOT NULL,
        PRIMARY KEY (id_conteudo),
        FOREIGN KEY (id_viagem)
        REFERENCES Viagem(id_viagem)
        ON DELETE CASCADE
        ON UPDATE NO ACTION,
        FOREIGN KEY (id_localizacao)
        REFERENCES Localizacao(id_localizacao)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);

```

5.2 Criação de utilizadores da base de dados

A segurança e a integridade dos dados na base de dados "Novos Horizontes" dependem de uma correta definição de acessos. Através da análise dos Requisitos de Controlo (Tabela 3) definimos a existência de três perfis de utilizadores com responsabilidades e níveis de acesso distintos: os alunos que atuam como consumidores de conteúdo, os professores que são os criadores do conteúdo e o bibliotecário que é responsável por auxiliar na administração dos dados e por inserir os dados fixos da base de dados.

Para evitar a gestão ineficiente e propensa a erros que resultaria da atribuição individual de privilégios a cada novo utilizador, a equipa de desenvolvimento optou pela implementação de um sistema baseado em cargos, RBAC. O comando “*CREATE ROLE*” foi utilizado para criar os diferentes cargos no sistema de gestão da base de

dados. Esta abordagem permite agrupar permissões por função, facilitando a administração da segurança e evitando a atribuição manual de privilégios a cada utilizador. Foram criados três cargos no SGBD MySQL – ‘aluno’ , ‘professor’ e ‘bibliotecario’ – que agregam os conjuntos de permissões necessários para cada função. A atribuição de permissões aos diferentes cargos teve em conta os requisitos de controlo.

Abaixo apresenta-se o script SQL que materializa esta estratégia:

```
CREATE ROLE IF NOT EXISTS 'aluno';

CREATE ROLE IF NOT EXISTS 'professor';

CREATE ROLE IF NOT EXISTS 'bibliotecario';
```

A atribuição de permissões é feita com o comando “*GRANT <permissões>*”. Através destes comandos foi possível especificar, de forma rigorosa, quais as operações que cada perfil terá sobre as tabelas do sistema, isto é, as operações de leitura (“*SELECT*”), inserção (“*INSERT*”), atualização (“*UPDATE*”) e remoção (“*DELETE*”). Apenas foram consideradas estas permissões uma vez que, de acordo com o **RC11**, nenhum elemento da escola deve ter acesso a inserir novas relações, removê-las ou alterar as existentes. É importante notar que o papel de bibliotecário foi criado mesmo podendo conter apenas um utilizador, o Sr. Bruno. Isso facilita futuras alterações, pois todas as permissões associadas ao papel podem ser geridas centralmente, permitindo adicionar ou remover utilizadores sem necessidade de redefinir permissões individuais. Esta abordagem garante a aplicação do princípio do menor privilégio, assegurando que cada utilizador dispõe apenas das permissões estritamente necessárias ao desempenho das suas funções. Além disso, dado o **RC10**, foi implementado como medida de segurança retirar a capacidade a todos os cargos de conceder os privilégios a eles concedidos, através do comando:

“*REVOKE GRANT OPTION ON viagensnh.* FROM 'cargo'*”. Apesar dos cargos não terem esse privilégio por pré-definição de criação no MySQL consideramos uma boa prática refletir explicitamente a restrição do requisito. Por fim, considerando as funções, procedimentos e vistas criados para satisfazer os requisitos de manipulação definidos, também é necessário decidir que cargos terão acesso aos mesmos. Para isso foram utilizados os comandos:

Para as vistas: "GRANT (permissoes) ON ViagensNH.'nome_vista' TO 'cargo'"

Para as funções: "GRANT EXECUTE ON FUNCTION ViagensNH.'nome_funcao' TO 'cargo'"

Para os procedimentos: "GRANT EXECUTE ON PROCEDURE ViagensNH.'nome_procedimento' TO 'cargo'"

Começando pelo cargo atribuído aos alunos, o **RC1** estipula que estes devem ter acesso de leitura em todas as relações da base de dados, exceto na tabela de **Professor e Aluno**. Para dar cumprimento a esta regra, foi atribuído ao 'aluno' o privilégio "SELECT" nas tabelas **Viagem, Conteudo, Localizacao, Cidade, Pais, Disciplina, Viagem_Disciplina e Aluno_Viagem**. Adicionalmente, para satisfazer o **RC2**, que exige a submissão da avaliação e do comentário após a realização de uma viagem, concedeu-se a permissão de "INSERT" na tabela associativa **Aluno_Viagem**.

Criou-se então a seguinte tabela com as diferentes permissões correspondentes ao cargo aluno e que facilitará a organização para atribuição destas no próprio SQL:

	Professor	Aluno	Disciplina	Viagem	Conteudo	Localização	Cidade	País	Aluno_Viagem	Viagem_Disciplina
INSERT									X	
DELETE										
UPDATE										
SELECT			X	X	X	X	X	X	X	X

Tabela 22 - Permissões do cargo aluno

Tendo em atenção as informação contida na tabela anterior foi criado o seguinte *script* para atribuição de permissões:

```
GRANT INSERT,SELECT ON ViagensNH.Aluno_Viagem TO 'aluno';
GRANT SELECT ON ViagensNH.Viagem TO 'aluno';
GRANT SELECT ON ViagensNH.Conteudo TO 'aluno';
GRANT SELECT ON ViagensNH.Viagem_Disciplina TO 'aluno';

GRANT SELECT ON ViagensNH.Disciplina TO 'aluno';
GRANT SELECT ON ViagensNH.Localizacao TO 'aluno';
```

```
GRANT SELECT ON ViagensNH.Cidade TO 'aluno';
GRANT SELECT ON ViagensNH.Pais TO 'aluno';
```

Além disso, os alunos devem ter permissões de ver a vista sobre os detalhes das viagens, “v_detalhes_viagem”, e a vista dos melhores dez alunos, “v_top_alunos”, tal como descrito nos **RC12** e **RC13**, respetivamente. Além disso, os alunos devem ter permissões de execução da função “f_total_vagens_aluno” e dos procedimentos “sp_perfil_preferencias_aluno” e “sp_pesquisar_vagem”, tal como definido nos **RC14**, **RC16** e **RC17**, respetivamente.

```
GRANT SELECT ON ViagensNH.v_detalhes_vagem TO 'aluno';
GRANT SELECT ON ViagensNH.v_top_alunos TO 'aluno';
GRANT EXECUTE ON FUNCTION ViagensNH.f_total_vagens_aluno TO
    'aluno';
GRANT EXECUTE ON PROCEDURE
    ViagensNH.sp_perfil_preferencias_aluno TO 'aluno';
GRANT EXECUTE ON PROCEDURE ViagensNH.sp_pesquisar_vagem TO
    'aluno';
```

Por fim, asseguramos que os alunos estão impedidos de partilhar os seus privilégios.

```
REVOKE GRANT OPTION ON ViagensNH.* FROM 'aluno';
```

Para o perfil de professor, a análise do **RC4** indica a necessidade dos professores poderem inserir autonomamente os conteúdos pedagógicos relacionados com as viagens. Assim, ao cargo ‘**professor**’ foram concedidos privilégios de “**INSERT**” nas tabelas **Viagem**, **Conteudo**, **Localização**, **Cidade**, **Pais** e **Viagem_Disciplina**. Este cargo possui ainda permissões de leitura sobre todas as relações existentes na base de dados, tal como definido no **RC3**. Segundo o **RC5**, este cargo deve ter permissões para alterar os dados da tabela das viagens e dos conteúdos. Por fim, considerando o **RC6** o cargo dos professores não têm qualquer permissão de remoção de registo na BD.

Considerando os requisitos mencionados acima, construímos a tabela 26 para representar as permissões do cargo destinado aos utilizadores dos professores.

	Professor	Aluno	Disciplina	Viagem	Conteudo	Localização	Cidade	País	Aluno_Viagem	Viagem_Disciplina
INSERT				X	X	X	X	X		X
DELETE										
UPDATE				X	X					
SELECT	X	X	X	X	X	X	X	X	X	X

Tabela 23 - Permissões do cargo professor

Traduzindo a tabela para um script SQL obtemos:

```

GRANT SELECT ON ViagensNH.* TO 'professor';
GRANT INSERT,UPDATE ON ViagensNH.Viagem TO 'professor';
GRANT INSERT,UPDATE ON ViagensNH.Conteudo TO 'professor';
GRANT INSERT ON ViagensNH.Localizacao TO 'professor';
GRANT INSERT ON ViagensNH.Cidade TO 'professor';
GRANT INSERT ON ViagensNH.Pais TO 'professor';

```

Pela análise dos requisitos de controlo 12 (**RC12**) ao requisito de controlo 17 (**RC17**) concluímos que os professores serão capazes de ver todas as vistas criadas, e de executar todas as funções e procedimentos, logo:

```

GRANT SELECT ON ViagensNH.v_detalhes_viagem TO 'professor';
GRANT SELECT ON ViagensNH.v_top_alunos TO 'professor';
GRANT EXECUTE ON FUNCTION ViagensNH.f_total_vagens_aluno TO
    'professor';
GRANT EXECUTE ON FUNCTION ViagensNH.f_taxa_sucesso_vagem TO
    'professor';
GRANT EXECUTE ON PROCEDURE
    ViagensNH.sp_perfil_preferencias_aluno TO 'professor';
GRANT EXECUTE ON PROCEDURE ViagensNH.sp_pesquisar_vagem TO
    'professor';

```

Mais uma vez, aplicamos a medida de segurança e retiramos a capacidade de partilha de privilégios.

```
REVOKE GRANT OPTION ON ViagensNH.* FROM 'professor';
```

Para finalizar, falta criar o perfil ‘bibliotecario’ atribuído ao utilizador do Sr. Bruno. Baseado nos **RC8** e **RC9**, atribuímos ao ‘bibliotecário’ permissão de “*SELECT*” e “*DELETE*” em todas as tabelas da base de dados. Deste modo, asseguramos que este cargo possui as permissões necessárias para eliminar dados e limitamos a possibilidade de ocorrerem erros de utilização que impliquem a perda de dados. Além disso, tal como definido no **RC7**, o cargo de ‘bibliotecario’ tem a capacidade de inserir dados de novos alunos, professores e disciplinas, centralizando e simplificando a gestão dos dados académicos.

Seguindo os requisitos explicados acima, desenvolvemos a seguinte tabela de permissões.

	Professor	Aluno	Disciplina	Viagem	Conteudo	Localização	Cidade	País	Aluno_Viagem	Viagem_Disciplina
INSERT	X	X	X							
DELETE	X	X	X	X	X	X	X	X	X	X
UPDATE	X	X	X							
SELECT	X	X	X	X	X	X	X	X	X	X

Tabela 24 - Permissões do cargo bibliotecario

A tradução destas permissões para SQL geraram o seguinte script:

```

GRANT SELECT ON ViagensNH.* TO 'bibliotecario';
GRANT DELETE ON ViagensNH.* TO 'bibliotecario';
GRANT INSERT,UPDATE ON ViagensNH.Aluno TO 'bibliotecario';
GRANT INSERT,UPDATE ON ViagensNH.Professor TO 'bibliotecario';
GRANT INSERT,UPDATE ON ViagensNH.Disciplina TO
    'bibliotecario';

```

Tal como os professores, o perfil de bibliotecario consegue ver todas as vistas criadas, e de executar todas as funções e procedimentos.

```

GRANT SELECT ON ViagensNH.v_detalhes_viagem TO
    'bibliotecario';
GRANT SELECT ON ViagensNH.v_top_alunos TO 'bibliotecario';
GRANT EXECUTE ON FUNCTION ViagensNH.f_total_viagens_aluno TO
    'bibliotecario';
GRANT EXECUTE ON FUNCTION ViagensNH.f_taxa_sucesso_viagem TO

```

```

        'bibliotecario';

GRANT EXECUTE ON PROCEDURE
ViagensNH.sp_perfil_preferencias_aluno TO 'bibliotecario';
GRANT EXECUTE ON PROCEDURE ViagensNH.sp_pesquisar_viagem TO
        'bibliotecario';

```

E por último asseguramos que o papel de bibliotecario não tem a capacidade de partilhar os privilégios.

```
REVOKE GRANT OPTION ON ViagensNH.* FROM 'bibliotecario';
```

Finalizada a criação dos perfis, para efeitos de validação e demonstração prática das políticas de segurança implementadas, procedemos à criação de utilizadores. Esta etapa é fundamental para assegurar que a segregação de funções opera conforme o planeado antes da entrada em produção.

Foram definidos três utilizadores de teste, cada um representando um dos perfis identificados:

- **sr_bruno**: Receberá o cargo de '**bibliotecario**', pelo que permite ser utilizado para validar as operações de manutenção de dados académicos.
- **orlando_belo**: Receberá o cargo de '**professor**', pelo que permite testar o ciclo de vida dos conteúdos (criação, edição e remoção de viagens), confirmando que os docentes têm autonomia pedagógica sem comprometer a estrutura do sistema.
- **francisco_luciano**: Receberá o cargo de '**aluno**', pelo que permite verificar a eficácia das restrições de leitura, assegurando que os alunos conseguem consultar materiais e submeter avaliações, mas estão tecnicamente impedidos de alterar dados sensíveis.

É importante ressalvar que, num ambiente de produção real, a criação de utilizadores seria um processo dinâmico, desencadeado pela aplicação de gestão escolar sempre que um novo membro fosse matriculado. No entanto, para o âmbito deste projeto, a criação estática destes utilizadores permite uma auditoria direta e eficaz das permissões.

O comando “*CREATE USER IF NOT EXISTS*” foi utilizado para criar utilizadores de teste na base de dados, evitando um erro no caso de já estarem criados e permitindo a validação prática das políticas de segurança implementadas. O *script* SQL

apresentado de seguida concretiza a criação destes utilizadores e a respetiva atribuição dos cargos previamente configurados, finalizando a implementação da camada de segurança.

```
CREATE USER IF NOT EXISTS 'sr_bruno'@'localhost' IDENTIFIED BY  
    'bibliotecario';  
  
CREATE USER IF NOT EXISTS 'orlando_belo'@'localhost'  
    IDENTIFIED BY 'belo20';  
  
CREATE USER IF NOT EXISTS 'francisco_luciano'@'localhost'  
    IDENTIFIED BY 'xico'';  
  
GRANT 'bibliotecario' TO 'sr_bruno'@'localhost';  
GRANT 'professor' TO 'orlando_belo'@'localhost';  
GRANT 'aluno' TO 'francisco_luciano'@'localhost';  
SET DEFAULT ROLE ALL TO 'sr_bruno'@'localhost';  
SET DEFAULT ROLE ALL TO 'orlando_belo'@'localhost';  
SET DEFAULT ROLE ALL TO 'francisco_luciano'@'localhost';
```

A associação dos utilizadores aos respetivos cargos foi efetuada através do comando “*GRANT <role> TO <user>*”. Este mecanismo permite que cada utilizador herde automaticamente todas as permissões definidas para o seu perfil, garantindo a correta atribuição de permissões entre alunos, professores e administração. Foi ainda utilizado “*SET DEFAULT ROLE ALL*” para que os utilizadores fiquem logo com os seus roles atribuídos no momento que se conectem à sua conta.

5.3 Povoamento da base de dados

O *script* de povoamento desenvolvido tem como objetivo inicializar a base de dados “Novos Horizontes” com um conjunto de dados consistente, realista e semanticamente válido. Esta etapa é fundamental para facilitar a realização de testes de carga, a validação das *queries* de exploração e a verificação dos mecanismos de integridade referencial implementados.

Para realizar a inserção de registos nas tabelas, utilizamos a instrução:

```
"INSERT INTO 'tabela' (atr1, atr2, ...) VALUES (reg1atr1,  
reg1atr2, ...), ..., (regnatr1, regnatr2, ...);"
```

Onde ‘tabela’ corresponde ao nome da tabela onde os dados serão inseridos. O esquema, “(atr1, atr2, ...)”, é opcional e representa a lista de colunas que irão receber os valores, mas caso seja especificado devemos garantir que cada valor é inserido na coluna correta. Por outro lado, se este estiver omissos é obrigatório fornecer valores para todas as colunas da tabela, respeitando a ordem em que foram definidas na sua criação. Podem ser inseridos vários registo por cada instrução de “*INSERT INTO*”. No povoamento presente neste tópico apenas foi utilizado o comando “*INSERT INTO*” com a especificação do esquema de forma a ficar mais claro aquilo que estava a ser inserido e para poder omitir os valores que são automaticamente definidos, como por exemplo, as chaves primárias artificiais.

Tal como na fase de implementação física da base de dados, a ordem no processo de povoamento deve respeitar rigorosamente as dependências existentes entre as diferentes tabelas. Estas dependências surgem devido às chaves estrangeiras que limitam a ordem pela qual os registo podem ser inseridos. Inserir dados numa tabela que referencia outra ainda não povoadas conduziria à violação da integridade referencial, resultando em erros de inserção ou inconsistências nos dados. Nesse sentido, optou-se por utilizar a mesma ordem definida na implementação física das tabelas, garantindo assim que todas as dependências entre entidades fossem respeitadas desde o início do processo. Ficou então a seguinte ordem: Professor, Aluno, Viagem, Aluno_Viagem, Disciplina, Viagem_Disciplina, Pais, Cidade, Localizao, Conteudo.

Começamos então por introduzir registo nas tabelas Professor e Aluno.

```
INSERT INTO Professor (id_professor, nome) VALUES  
(5001, 'Marcelo Almeida'),  
(5002, 'Sofia Nunes'),  
(5003, 'Rita Carvalho'),  
(5004, 'Pedro Mendes');  
  
INSERT INTO Aluno (id_aluno, nome, genero, ano, letra) VALUES  
(2001, 'Ana Silva', 'F', 9, 'A'),  
(2002, 'Bruno Costa', 'M', 6, 'B'),  
(2003, 'Carla Rocha', 'F', 11, 'A'),  
(2004, 'Diogo Pereira', 'M', 7, 'C'),  
(2005, 'Eva Martins', 'F', 6, 'B'),  
(2006, 'Fábio Santos', 'M', 12, 'A'),
```

(2007, 'Gabriela Lopes', 'F', 10, 'C'),
(2008, 'Henrique Dias', 'M', 12, 'A'),
(2009, 'Inês Faria', 'F', 7, 'B'),
(2010, 'João Neves', 'M', 6, 'C');

Posteriormente, povoamos o núcleo do sistema: as **Viagens**. São registadas diversas viagens virtuais, associadas aos professores que as criaram. De seguida, inserimos também os registo da tabela **Aluno-Viagem**.

```
INSERT INTO Viagem  
(titulo, data_publicacao, resumo, id_professor) VALUES  
('Lisboa Histórica', '2024-03-12', 'Exploração aprofundada da  
história de Lisboa, analisando os seus monumentos, evolução  
urbana e impacto cultural ao longo dos séculos.', 5001),  
('Porto Animal', '2024-03-20', 'Viagem educativa dedicada ao  
estudo da fauna, abordando biodiversidade, habitats naturais e  
a importância da preservação das espécies.', 5002),  
('Madrid Artística', '2024-04-05', 'Percorso cultural pelos  
principais museus e espaços artísticos de Madrid,  
contextualizando as obras e os seus autores.', 5003),  
('Paris Monumental', '2024-04-18', 'Viagem interdisciplinar  
que relaciona matemática, arquitetura e história através da  
análise dos monumentos icónicos de Paris.', 5001),  
('Roma Antiga', '2024-05-01', 'Exploração detalhada da  
civilização romana, destacando arquitetura, política e legado  
cultural da Roma Antiga.', 5004),  
('Berlim Contemporânea', '2024-05-15', 'Análise da história  
recente de Berlim, com foco na divisão da cidade, na Guerra  
Fria e na reunificação alemã.', 5003),  
('Ciência em Munique', '2024-05-25', 'Viagem virtual dedicada  
à ciência e tecnologia, explorando museus científicos e  
inovações desenvolvidas em Munique.', 5002),  
('Barcelona Criativa', '2024-06-05', 'Exploração artística e  
arquitetónica da cidade de Barcelona, com destaque para as  
obras de Antoni Gaudí.', 5001)  
;
```

```
INSERT INTO Aluno_Viagem
(id_aluno, id_viagem, avaliacao, comentario, data_realizacao)
VALUES
    (2001, 1, 10, 'Viagem extremamente enriquecedora que ajudou a compreender melhor a história e identidade cultural de Lisboa.', '2024-03-13'),
    (2001, 4, 8, 'Gostei muito da ligação entre matemática e arquitetura, especialmente na análise dos monumentos parisienses.', '2024-04-19'),
    (2002, 1, 9, 'Atividade bastante interessante que reforçou os meus conhecimentos sobre a história de Portugal.', '2024-03-13'),
    (2002, 2, 10, 'Experiência educativa e envolvente, com conteúdos bem explicados sobre os animais e os seus habitats.', '2024-03-21'),
    (2003, 2, 8, 'Aprendi bastante sobre biodiversidade e preservação ambiental ao longo da viagem.', '2024-03-21'),
    (2003, 3, 9, 'Viagem cultural muito inspiradora que despertou ainda mais o meu interesse pela arte.', '2024-04-06'),
    (2004, 3, 7, 'Conteúdos bem estruturados que ajudaram a compreender melhor o contexto artístico de Madrid.', '2024-04-06'),
    (2005, 5, 10, 'Excelente abordagem à história romana, com explicações claras e exemplos bem escolhidos.', '2024-05-02'),
    (2005, 8, 7, 'Gostei especialmente da arquitetura de Gaudí e da forma criativa como os conteúdos foram apresentados.', '2024-06-06'),
    (2006, 2, 8, 'Viagem muito bem organizada e educativa, com exemplos claros e interessantes.', '2024-03-21'),
    (2006, 7, 8, 'Experiência bastante interessante que mostrou a importância da ciência e da inovação.', '2024-05-26'),
```

(2007, 4, 9, 'A ligação entre diferentes disciplinas tornou a viagem muito mais interessante.', '2024-04-19'),

(2008, 3, 8, 'Viagem cultural rica e bem contextualizada, com bons exemplos artísticos.', '2024-04-06'),

(2008, 6, 9, 'Gostei muito da forma como a história recente de Berlim foi explicada.', '2024-05-16'),

(2009, 5, 7, 'Aprendi bastante sobre a civilização romana e o seu impacto na atualidade.', '2024-05-02'),

(2010, 1, 8, 'Viagem interessante e bem organizada, com conteúdos claros e acessíveis.', '2024-03-13')

;

Segue-se o povoamento da relação **Disciplina**, que é uma entidade que não depende de nenhuma outra, e da **Viagem_Disciplina**, uma vez que as instâncias de **Disciplina** e de **Viagem** já foram criadas neste ponto.

```
INSERT INTO Disciplina (nome, ano) VALUES
('Educação Visual', 5),
('História', 6),
('Matemática', 7),
('Ciências Naturais', 8),
('Física e Química', 9),
('Português', 10)
;
```

```
INSERT INTO Viagem_Disciplina (id_viagem, id_disciplina)
VALUES
(1,2),(2,4),(3,1),(3,2),(4,2),(4,3),(5,2),(6,2),(7,5),(8,1);
```

Posteriormente, realizamos o povoamento relativo aos dados geográficos. Começamos a partir da tabela **Pais**, seguida da tabela **Cidade** e finalizamos com a tabela **Localização**. Esta organização na inserção dos dados permitiu a definição de uma chave estrangeira na tabela **Cidade** que referencia um valor válido da chave primária da tabela **Pais**. Seguindo a mesma linha de raciocínio o povoamento na tabela **Cidade**, permitiu inserir os dados na tabela **Localizacao**.

```
INSERT INTO Pais (pais)
VALUES
('Portugal'), ('Espanha'), ('França'), ('Itália'), ('Alemanha');
```

```
INSERT INTO Cidade (cidade, id_pais)
VALUES
('Lisboa', 1), ('Porto', 1),
('Madrid', 2), ('Barcelona', 2),
('Paris', 3), ('Marselha', 3),
('Roma', 4), ('Milão', 4),
('Berlim', 5), ('Munique', 5)
; 
```

```
INSERT INTO Localizacao (nome, id_cidade)
VALUES
('Torre de Belém', 1), ('Zoológico', 1),
('Ribeira', 2), ('Sé do Porto', 2),
('Museu do Prado', 3),
('Parque Güell', 4),
('Torre Eiffel', 5), ('Louvre', 5),
('Coliseu', 7),
('Porta de Brandemburgo', 9);
```

Por fim, povoamos a tabela de Conteúdos. Esta foi preenchida com referências a vídeos e descrições, ligando cada item multimédia à sua respetiva viagem e localização.

```
INSERT INTO Conteudo
(url,    descricao,    titulo,    tipo,    data_captura,    id_viagem,
```

```
    id_localizacao)
VALUES
    ('https://img1.jpg', 'Fotografia detalhada da Torre
de Belém, evidenciando os elementos arquitetónicos
manuelinos.', 'Torre de Belém', 'F', '2024-03-12', 1, 1),
    ('https://vid1.mov', 'Vídeo educativo sobre o
Oceanário de Lisboa, mostrando diversas espécies marinhas.',
'Oceanário de Lisboa', 'V', '2024-03-12', 1, 2),
    ('https://img2.jpg', 'Imagen panorâmica da Ribeira do
Porto, destacando a arquitetura tradicional.', 'Ribeira do
Porto', 'F', '2024-03-20', 2, 3),
    ('https://img3.jpg', 'Fotografia da Sé do Porto ao
entardecer, evidenciando o seu valor histórico.', 'Sé do
Porto', 'F', '2024-03-20', 2, 4),
    ('https://img4.jpg', 'Imagen do Museu do Prado com destaque
para a sua importância artística.', 'Museu do Prado', 'F',
'2024-04-05', 3, 5),
    ('https://vid2.mov', 'Vídeo panorâmico do Parque
Güell, destacando a arquitetura modernista.', 'Parque Güell',
'V', '2024-06-05', 8, 6),
    ('https://img5.jpg', 'Fotografia noturna da Torre
Eiffel iluminada.', 'Torre Eiffel', 'F', '2024-04-18', 4, 7),
    ('https://vid3.mov', 'Vídeo da entrada do Museu do
Louvre, mostrando a pirâmide central.', 'Museu do Louvre',
'V', '2024-04-18', 4, 8),
    ('https://vid4.mov', 'Vídeo ilustrativo do Coliseu
Romano, explorando a sua estrutura.', 'Coliseu Romano', 'V',
'2024-05-01', 5, 9),
    ('https://img6.jpg', 'Fotografia da Porta de
Brandemburgo, símbolo histórico de Berlim.', 'Porta de
Brandemburgo', 'F', '2024-05-15', 6, 10)
;
```

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

Para garantir que o sistema implementado para a escola é adequado ao seu funcionamento e sustentabilidade a longo prazo, procedeu-se a uma estimativa teórica do espaço de armazenamento necessário para a base de dados desenvolvida. Esta estimativa tem como objetivo avaliar a dimensão inicial da base de dados e analisar de que forma o seu tamanho poderá evoluir ao longo do tempo, tendo em conta o crescimento expectável do volume de informação armazenada. O cálculo foi realizado com base no modelo físico implementado em *MySQL*, utilizando o motor de armazenamento *InnoDB*, e assenta em valores médios considerados adequados para o caso.

O primeiro passo consistiu na identificação do espaço ocupado pelos diferentes tipos de dados utilizados no projeto. Alguns tipos de dados apresentam um tamanho fixo em disco, como é o caso do tipo “*INT*”, cujo espaço ocupado é constante independentemente do valor armazenado. No entanto, outros tipos, nomeadamente os de comprimento variável, como “*VARCHAR(N)*” e “*DECIMAL*”, dependem diretamente do tamanho máximo definido e do conteúdo efetivamente armazenado. Para determinar o espaço ocupado por cada tipo de dados, recorreu-se à documentação oficial do *MySQL*, que descreve a forma como o motor *InnoDB* armazena cada domínio em disco. Com base nessa informação, foi construída uma tabela de correspondência entre os tipos de dados utilizados no modelo e o respetivo espaço ocupado, permitindo obter valores de referência consistentes para os cálculos seguintes.

Domínios	Tamanho (em bytes)
<i>INT</i>	4
<i>TINYINT</i>	1
<i>DATE</i>	3
<i>CHAR(N)</i>	N*4
<i>ENUM('val1', ...)</i>	1 ou 2 dependendo do número de opções
<i>VARCHAR(N)</i>	Espaço máximo: N*4 + 1 (se N*4 < 255) N*4 + 2 (se N*4 > 255)

Tabela 25 - Domínios utilizados e respetivo tamanho em *bytes*

Alguns dos tipos envolveram algumas considerações:

- o tipo “CHAR” é utilizado para armazenar *strings* de tamanho fixo. Cada coluna “CHAR(N)” reserva um espaço constante em *bytes*, independentemente do tamanho real do conteúdo armazenado. Com o charset *utf8mb4*, que foi especificado na criação da base dados, cada caractere pode ocupar até 4 *bytes*. Portanto, uma coluna “CHAR(N)” ocupa sempre $4 \times N$ *bytes*. Sendo que apenas utilizamos “CHAR(1)” conclui-se que para este caso apenas são utilizados 4 *bytes*.
- o tipo “VARCHAR” é de tamanho variável. Ele armazena apenas os *bytes* necessários para os caracteres efetivamente utilizados, mais um pequeno overhead de 1 ou 2 *bytes* para registrar o comprimento da *string*. Se o total de *bytes* armazenados ultrapassar 255, o overhead passa a ser de 2 *bytes*. Essa característica torna o “VARCHAR” mais eficiente para campos onde o comprimento da *string* pode variar significativamente. Desta forma, foi necessário realizar estimativas para a média da percentagem que é ocupada em relação à capacidade máxima de cada valor.
- o tipo “ENUM” para até 255 valores diferentes, ocupa apenas 1 *byte*; para mais de 255 valores, são usados 2 *bytes*. No nosso projeto, cada coluna “ENUM” possui apenas duas opções possíveis, o que significa que cada valor ocupa apenas 1 *byte*.

A partir dos valores associados a cada domínio, foi possível estimar o tamanho médio de um registo em cada uma das relações da base de dados. Para tal, analisaram-se individualmente todas as tabelas do modelo, somando o espaço ocupado por cada atributo que compõe um registo. No caso dos atributos de tamanho variável, como títulos, descrições, URLs ou comentários, não se considerou o tamanho máximo definido no esquema, mas sim um valor médio esperado, de forma a obter uma estimativa mais realista do espaço efetivamente utilizado pelo sistema. Considerou-se então que em média é utilizada 70% da capacidade para estes atributos. Para cada relação foi construída uma tabela auxiliar que apresenta, segundo o estudo em análise, o espaço ocupado por cada registo de cada entidade.

Relação Aluno		
Atributo	Domínio	Tamanho (em bytes)
id_aluno	INT	4
nome	VARCHAR(100)	$100 * 4 * 0.70 + 2 = 282$
genero	ENUM('M','F')	1
ano	TINYINT	1
letra	CHAR(1)	$4*1 = 4$
TOTAL	-	292

Tabela 26 - Espaço ocupado por um registo da relação Aluno

Relação Professor		
Atributo	Domínio	Tamanho (em bytes)
id_professor	INT	4
nome	VARCHAR(100)	$100 * 4 * 0.70 + 2 = 282$
TOTAL	-	286

Tabela 27 - Espaço ocupado por um registo da relação Professor

Relação Disciplina		
Atributo	Domínio	Tamanho (em bytes)
id_viagem	INT	4
nome	VARCHAR(45)	$45*4*0.7 + 1 = 126$
ano	TINYINT	1
TOTAL	-	131

Tabela 28 - Espaço ocupado por um registo da relação Disciplina

Relação Viagem		
Atributo	Domínio	Exemplo
id_viagem	INT	4
titulo	VARCHAR(100)	$100 * 4 * 0.70 + 2 = 282$
data_publicacao	DATE	3
resumo	VARCHAR(1000)	$1000 * 4 * 0.70 + 2 = 2802$
id_professor	INT	4
TOTAL	-	3095

Tabela 29 - Espaço ocupado por um registo da relação Viagem

Relação Conteudo		
Atributo	Domínio	Tamanho (em bytes)
id_conteudo	INT	4
url	VARCHAR(150)	$150*4*0.7 + 2 = 422$
descricao	VARCHAR(500)	$50*4*0.7 + 2 = 1402$
titulo	VARCHAR(50)	$50*4*0.7 + 1 = 141$
tipo	ENUM('F', 'V')	1
data_captura	DATE	3
id_viagem	INT	4
id_localizacao	INT	4
TOTAL	-	1981

Tabela 30 - Espaço ocupado por um registo da relação Conteudo

Relação Aluno_Viagem		
Atributo	Domínio	Tamanho (em bytes)
id_aluno	INT	4
id_viagem	INT	4
avaliacao	INT	4
comentario	VARCHAR(200)	$200*4*0.7 + 2 = 562$
data_realizacao	DATE	3
TOTAL	-	577

Tabela 31 - Espaço ocupado por um registo da relação Aluno_Viagem

Relação Viagem_Disciplina		
Atributo	Domínio	Tamanho (em bytes)
id_viagem	INT	4
id_disciplina	INT	4
TOTAL	-	8

Tabela 32 - Espaço ocupado por um registo da relação Viagem_Disciplina

Relação Localizacao		
Atributo	Domínio	Tamanho (em bytes)
id_localizacao	INT	4
nome	VARCHAR(70)	$70*4*0.7 + 1 = 197$
Id_cidade	INT	4
TOTAL	-	205

Tabela 33 - Espaço ocupado por um registo da relação Localizacao

Relação Cidade		
Atributo	Domínio	Tamanho (em bytes)
id_cidade	INT	4
cidade	VARCHAR(35)	$35*4*0.7 + 1 = 99$
Id_pais	INT	4
TOTAL	-	107

Tabela 34 - Espaço ocupado por um registo da relação Cidade

Relação País		
Atributo	Domínio	Tamanho (em bytes)
id_pais	INT	4
país	VARCHAR(35)	$35*4*0.7 + 1 = 99$
TOTAL	-	103

Tabela 35 - Espaço ocupado por um registo da relação País

Para estimar o tamanho total da base de dados, tornou-se necessário definir valores realistas para o número de registos esperados em cada tabela. Para esse efeito, foi realizada uma nova reunião com a direção da escola e com o Sr. Bruno, com o objetivo de recolher informação aproximada sobre o funcionamento normal da instituição. Nesta reunião foram discutidos aspetos como o número médio de alunos e professores que entram na escola por ano letivo e o número expectável de viagens criadas por cada professor ao longo do tempo, bem como a quantidade média de

conteúdos associados a cada viagem. Foi ainda validada a estimativa de ocupação média de 70% para os campos de tamanho variável.

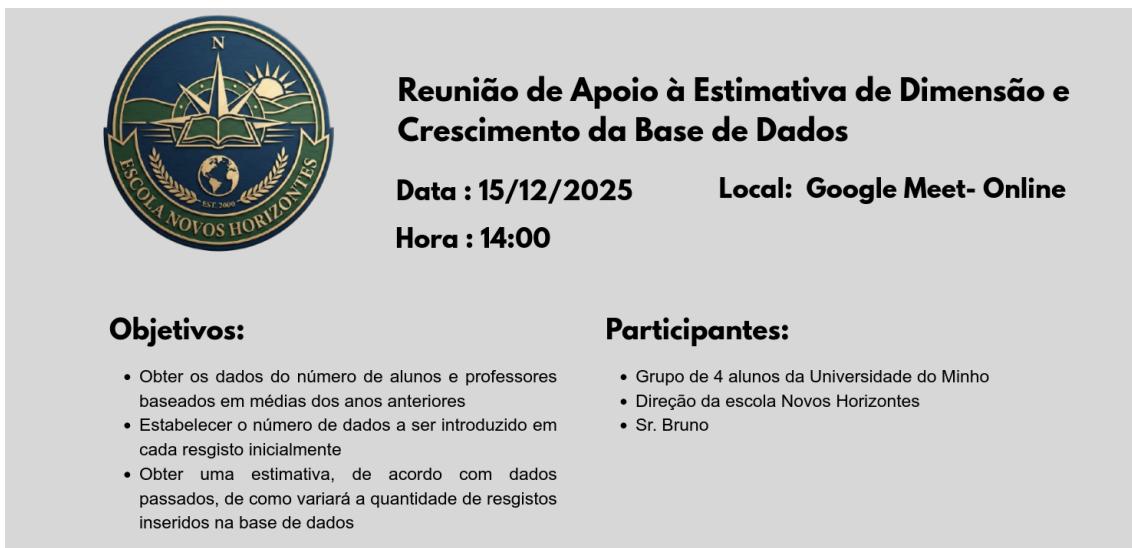


Figura 15- Cabeçalho da ata da reunião para obter dados para fazer estimativas sobre o tamanho inicial e taxa de crescimento da BD

Desta reunião foram retiradas as seguintes notas:

- O número de alunos na escola mantém-se praticamente constante com uma média dos últimos 10 anos de 1056 alunos por ano na escola. Saindo e entrando em média 88 alunos por ano. Os alunos que saíram devem continuar registados na base de dados de forma a ficarem registadas as suas avaliações das viagens. Ficou ainda estabelecido que o registo de um aluno que tenha saído da escola deve ser alterado para ter ficar na turma com ano 0 e letra A.
- O corpo docente, incluindo todos os níveis de ensino e disciplinas, é, de acordo com as médias dos últimos 5 anos, constituído por 80 professores por ano letivo. Verifica-se também que o número de professores ativos por ano mantém-se aproximadamente constante, entrando cerca 5 professores novos por ano e saindo outros 5. Os professores que saem da escola devem se manter registados para existir sempre um criador das viagens. Serão desconsideradas situações onde seria possível eliminar o registo do professor por falta de atividade do mesmo.

- Serão oferecidas cerca de 55 disciplinas diferentes no momento de implementação da base de dados. Segundo a direção da escola este valor tende a aumentar cerca de duas disciplina em cada 3 anos ($\frac{2}{3} = 0.667$ por ano).
- Estão previstas, inicialmente, a criação de 200 viagens e um aumento deste número em cerca de 1 por professor por ano (cerca de 80 por ano).
- Cada aluno participa, em média, em 6 viagens por ano, resultando em aproximadamente $1056 * 5 = 6336$ registos na relação Aluno_Viagem.
- Cada viagem aborda, em média, 2 disciplinas, perfazendo 400 registos na relação Viagem_Disciplina com um aumento de $80 * 2 = 160$ por ano.
- Cada viagem terá em média 6 conteúdos diferentes, resultando em cerca de 1200 registos na relação Conteúdo, inicialmente. O crescimento esperado por ano será $80 * 6 = 480$ novos registos na relação Conteudo por ano.
- Considerando as viagens realizadas, estima-se que existam 1000 locais distintos visitados, registados na tabela Localizacao, com um crescimento de cerca de 200 registos por ano.
- A abrangência geográfica das viagens corresponde a 60 cidades diferentes e 10 países, estimativa baseada nos registos históricos de viagens escolares disponíveis. Espera-se um aumento no número de 6 cidades por ano e 1 país por ano.

Notar que estes valores foram apenas estimativas definidas durante a reunião, com as quais todos os membros da mesma acharam plausíveis e que, por essa razão, poderão não ir de encontro à real taxa de crescimento da base de dados.

Com base nos tamanhos médios dos registos e no número estimado de registos por relação, foi então possível calcular o tamanho teórico inicial da base de dados, através da multiplicação do tamanho médio de cada registo pelo número de registos correspondente. Obteve-se então o seguinte valor:

$$292 * 1056 + 286 * 80 + 131 * 55 + 3095 * 200 + 1981 * 1200 + 205 * 1000 + 8 * 400 + 107 * 60 + 103 * 10 = 3550287 \text{ bytes}$$

Nota-se que o número de registos para a tabela Aluno_Viagem foi 0 uma vez que inicialmente nenhum aluno vai ter viagens realizadas. Para um valor inicial do sistema obteve-se cerca de 3.55 *MegaByte* (MB). Consideramos que 1 MB = 1 000 000 bytes.

A partir deste valor inicial, analisou-se também o aumento previsto para o armazenamento devido a um ano , considerando as estimativas feitas:

$$292 * 88 + 286 * 5 + 0.667 * 131 + 80 * 3092 + 6336 * 577 + 160 * 8 + 480 * 1981 + 200 * 205 + 6 * 107 + 103 * 1 = 4919310 \text{ bytes}$$

Obteve-se uma taxa de crescimento anual de cerca de 4.92MB. Pode-se então concluir que o armazenamento ocupado pelo sistema implementado, ao fim de n anos, seja de: $3.55 + 4.92 * n$ MB.

Os cálculos mostram que o espaço total ocupado pela base de dados é consideravelmente reduzido e perfeitamente compatível com os recursos de armazenamento disponíveis em qualquer infraestrutura atual.

Importa, no entanto, referir que os valores obtidos correspondem a uma estimativa teórica e não refletem de forma exata o espaço real ocupado em disco. O motor *InnoDB* utiliza mecanismos internos adicionais que não foram considerados neste cálculo, como a paginação dos dados, o armazenamento de índices, e a existência de cabeçalhos e metadados associados a cada registo. Os dados são armazenados em páginas de tamanho fixo, o que pode levar a algum desperdício de espaço quando as páginas não se encontram totalmente preenchidas. Além disso, os índices criados para garantir a integridade referencial e melhorar o desempenho das consultas também ocupam espaço adicional em disco. Apesar destas limitações, a estimativa realizada é suficiente para concluir que a base de dados apresenta uma dimensão reduzida e que os requisitos de armazenamento são pouco exigentes, não constituindo qualquer limitação para a adoção e utilização do sistema no contexto da escola.

5.5 Definição e caracterização de vistas de utilização em SQL

As Vistas funcionam como tabelas virtuais, definidas a partir de instruções “*SELECT*”. Estas não armazenam dados fisicamente, sendo o seu conteúdo calculado dinamicamente sempre que são consultadas. O comando utilizado para a sua criação é o “*CREATE VIEW nome AS SELECT ...*”, que permite associar um nome a uma

consulta em SQL, tornando possível reutilizar essa consulta de forma simples e consistente.

A utilização de Vistas no sistema "Novos Horizontes" justifica-se pela necessidade de abstração e segurança, bem como para simplificar o acesso a dados que se encontram normalizados e dispersos por várias tabelas. Ao invés de os utilizadores (ou a aplicação) escreverem junções complexas repetidamente, criaram-se camadas lógicas virtuais.

Foi identificada a necessidade de uma vista que agregasse toda a informação detalhada de uma viagem num único registo. A vista "v_detalhes_viagem" realiza a junção entre a viagem, o nome do professor responsável e a disciplina associada, facilitando a apresentação de dados sem expor diretamente a estrutura física das tabelas.

```
CREATE VIEW v_detalhes_viagem AS SELECT
v.id_viagem, v.titulo, v.data_publicacao, p.nome AS Professor,
GROUP_CONCAT(CONCAT(d.nome, ' ', d.ano, 'º ano') SEPARATOR ',',
') AS Disciplinas FROM Viagem AS v
INNER JOIN Professor AS p ON v.id_professor = p.id_professor
INNER JOIN Viagem_Disciplina AS vd
ON v.id_viagem = vd.id_viagem
INNER JOIN Disciplina AS d
ON vd.id_disciplina = d.id_disciplina
GROUP BY v.id_viagem
ORDER BY v.id_viagem;
```

Adicionalmente, para satisfazer o requisito **RM9**, realizou-se permitindo a visualização rápida do "Top 10 Alunos" que realizaram mais viagens, criou-se a vista "v_top_alunos". Esta vista abstrai o cálculo da contagem de viagens, apresentando uma lista ordenada dos 10 alunos mais ativos, pronta a ser consumida por qualquer utilizador com permissões de leitura.

```
CREATE VIEW v_top_alunos AS SELECT
a.id_aluno, a.nome, CONCAT(a.ano, 'º', ' ', a.letra) AS turma,
COUNT(av.id_viagem) AS total_viagens
FROM Aluno AS a
INNER JOIN Aluno_Viagem AS av ON a.id_aluno = av.id_aluno
```

```

        GROUP BY a.id_aluno, a.nome
        ORDER BY total_vagens DESC, MAX(av.data_realizacao) ASC
        LIMIT 10;

```

5.6 Tradução das interrogações do utilizador para SQL

Nesta fase, procedeu-se à tradução direta das expressões de Álgebra Relacional validadas na secção 4.4 para a linguagem SQL. Estas instruções constituem a base das operações de manipulação de dados do sistema, permitindo a implementação prática das consultas e funcionalidades definidas anteriormente. Foi ainda comparado os resultados obtidos nas expressões de álgebra relacional, com as queries desenvolvidas nesta fase de forma a confirmar uma correta tradução para SQL.

RM7: Listar todas as viagens disponíveis para uma determinada disciplina dado o seu identificador, ordenadas da mais recente para a mais antiga (apresentando o id, o título, o resumo e a data de publicação).

Para efeitos de demonstração consideramos que era preendido listar as viagens para a disciplina com id 2.

```

SELECT v.id_vagem, v.titulo, v.resumo, v.data_publicacao
      FROM Viagem AS v
INNER JOIN Viagem_Disciplina AS vd
        ON v.id_vagem = vd.id_vagem
INNER JOIN Disciplina AS d
        ON vd.id_disciplina = d.id_disciplina
       WHERE d.id_disciplina = 2
ORDER BY v.data_publicacao DESC;

```

RM8: Listar os alunos (id e nome) que realizaram uma determinada viagem, ordenados alfabeticamente, apresentando a sua respetiva avaliação.

Para efeitos de demonstração vamos considerar que queremos listar os alunos que realizaram a viagem com id igual a 1:

```

SELECT a.id_aluno, a.nome, av.avaliacao
      FROM Aluno AS a

```

```

    INNER JOIN Aluno_Viagem AS av
    ON a.id_aluno = av.id_aluno
    WHERE av.id_viagem = 1
    ORDER BY a.nome ASC;

```

RM10: Consultar a classificação média atual de todas as viagens, apresentando o respetivo identificador, título e o valor da média calculada.

```

SELECT v.id_viagem, v.titulo, AVG(av.avaliacao) AS Media
    FROM Viagem AS v
    INNER JOIN Aluno_Viagem AS av
    ON v.id_viagem = av.id_viagem
    GROUP BY v.id_viagem;

```

RM11: Listar o desempenho dos professores, apresentando o id do professor, nome do professor e o número total de viagens virtuais criadas por cada um, ordenado por ordem decrescente da quantidade de viagens criadas. Além disso, devem ser apresentadas as classificações médias mínimas e máximas das suas viagens.

```

SELECT p.id_professor, p.nome, COUNT(v.id_viagem) AS
total_viagens, MIN(r.nm) AS AvaliacaoMin, MAX(r.nm) AS
AvaliacaoMax
    FROM Professor AS p
    LEFT JOIN Viagem AS v
    ON p.id_professor = v.id_professor
    LEFT JOIN (
        SELECT id_viagem, AVG(avaliacao) AS nm
            FROM Aluno_Viagem
            GROUP BY id_viagem) AS r
        ON v.id_viagem = r.id_viagem
        GROUP BY p.id_professor
        ORDER BY total_viagens DESC;

```

5.7 Indexação do Sistema de Dados

A estratégia de indexação foi desenhada para otimizar o desempenho das interrogações mais frequentes (identificadas na secção 5.6), equilibrando a velocidade de leitura (“*SELECT*”) com os custos adicionais que os índices implicam: maior tempo nas operações de escrita (“*INSERT/UPDATE*”) e maior consumo de memória e armazenamento. Por esta razão, optou-se por indexar apenas os atributos mais relevantes, evitando impactos desnecessários no desempenho e no uso de recursos do sistema.

O MySQL (motor *InnoDB*) cria automaticamente índices para as Chaves Primárias e Chaves Estrangeiras. Contudo, para atributos utilizados frequentemente em cláusulas de filtragem (“*WHERE*”) e ordenação (“*ORDER BY*”), é necessário criar índices explícitos, de forma a melhorar a velocidade das buscas sobre eles. Devido à dimensão relativamente reduzida do projeto e ao volume moderado de dados previstos para a base de dados da Escola Novos Horizontes, é provável que o impacto do *overhead* de índices seja pouco significativo.

A instrução utilizada para implementar os índices foi:

```
CREATE INDEX nome_do_indice ON nome_da_tabela(coluna)
```

Onde “*nome_do_indice*” é o nome atribuído ao índice, “*nome_da_tabela*” é a tabela na qual o índice será criado, e *coluna* indica o(s) atributo(s) a indexar.

Implementaram-se índices para os nomes dos alunos e dos professores e também para os títulos das viagens e para a data de publicação das viagens, uma vez que o sistema prevê funcionalidades de pesquisa textual nestes campos.

Índice para otimizar a ordenação por data (RM7):

```
CREATE INDEX idx_viagem_data ON Viagem(data_publicacao);
```

Índice para acelerar a ordenação por título (RM13):

```
CREATE INDEX idx_viagem_titulo ON Viagem(titulo);
```

Índice para acelerar a pesquisa e ordenação de alunos por nome (RM8):

```
CREATE INDEX idx_aluno_nome ON Aluno(nome);
```

Em suma, a implementação destes índices estratégicos assegura uma melhoria significativa no desempenho global do Sistema de Base de Dados da Escola Novos Horizontes. Ao evitar a necessidade de leituras sequenciais completas das tabelas nas pesquisas mais frequentes e ao agilizar os processos de ordenação temporal e alfabética, reduz-se drasticamente o custo computacional das interrogações. Desta forma, garante-se que o sistema permanece eficiente e com tempos de resposta curtos, mesmo perante o crescimento expectável do volume de dados.

5.8 Implementação de procedimentos, funções e gatilhos

A implementação de lógica do lado do servidor (*server-side logic*) é fundamental para garantir que as regras de negócio mais importantes sejam sempre aplicadas diretamente na base de dados, independentemente da aplicação utilizada. Esta abordagem centraliza a validação e o processamento de dados, assegurando a integridade e a consistência do sistema implementado para a escola “Novos Horizontes”. Para tal, utilizam-se **procedimentos**, que permitem executar um conjunto de operações de forma organizada; **funções**, que realizam cálculos ou retornam valores derivados em tempo real; e **gatilhos**, que executam automaticamente ações em resposta a eventos na base de dados, garantindo que alterações nos dados seguem sempre as regras definidas.

Optou-se por não armazenar o atributo derivado “nr_vagens_realizadas”, que originalmente indicaria o número total de viagens realizadas por cada aluno, na entidade Aluno. Tendo em conta o tamanho relativamente reduzido do sistema desenvolvido, calcular este valor através de junções entre as tabelas envolvidas é pouco dispendioso em termos de desempenho. Assim, em vez de manter um campo adicional que exigiria atualização constante sempre que um aluno realiza uma viagem, decidiu-se privilegiar a simplicidade da estrutura da base de dados, recorrendo a **funções** para calcular o número de viagens em tempo real. Adicionalmente, utilizaram-se **gatilhos** para assegurar a consistência temporal dos dados inseridos, garantindo que todas as alterações seguem corretamente as regras definidas.

Gatilhos

Um **gatilho** é um mecanismo que executa automaticamente instruções SQL em resposta a eventos como inserções, atualizações ou remoções numa tabela. A notação básica de um gatilho no *MySQL* consiste, inicialmente em criá-lo com “*CREATE TRIGGER nome_gatilho*”, depois define-se um nome para o gatilho, indicar se deve ser executado antes (“*BEFORE*”) ou depois (“*AFTER*”) de um evento, especificar o tipo de evento que o dispara (“*INSERT*”, “*UPDATE*” ou “*DELETE*”), e indicar que ele se aplica a cada linha afetada pela operação (“*FOR EACH ROW*”). Dentro do gatilho, é possível colocar um bloco de instruções SQL delimitado por “*BEGIN ... END*”, onde podem ser realizadas verificações, cálculos ou operações que garantam a integridade e a consistência dos dados.

Mesmo considerando que o sistema implementado não necessitaria de nenhum *trigger*, optamos por implementar um de forma ilustrativa de como seria. Para isso, apesar da tabela *Aluno_Viagem* possuir um valor por defeito para a data de realização (“*CURRENT_DATE*”), o sistema permite a inserção manual de datas. Para evitar inconsistências lógicas — como um aluno realizar uma viagem antes de esta ter sido sequer publicada pelo professor — implementou-se o gatilho “*t_valida_cronologia*”.

Este gatilho verifica, antes de cada inserção, se a data de realização é válida face à data de publicação da viagem, rejeitando a operação caso a cronologia seja impossível.

```
DELIMITER $$

CREATE TRIGGER t_valida_cronologia BEFORE INSERT ON
    Aluno_Viagem
    FOR EACH ROW
    BEGIN
        DECLARE data_pub DATE;
        SELECT data_publicacao INTO data_pub FROM Viagem
            WHERE id_viagem = NEW.id_viagem;
        IF NEW.data_realizacao < data_pub THEN
            SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'Erro de Integridade: A viagem não pode ser
            realizada antes da sua data de publicação.';
```

```

        END IF;
    END $$

DELIMITER ;

```

Funções

As funções foram implementadas neste sistema para automatizar cálculos frequentes e garantir a integridade dos indicadores de desempenho. Ao contrário dos procedimentos, as funções são desenhadas para devolver um valor único, permitindo a sua integração direta em instruções “*SELECT*”, “*WHERE*” ou mesmo dentro de outras estruturas lógicas. Esta estratégia promove a reutilização de código e evita a redundância de dados, uma vez que valores estatísticos e contagens são calculados em tempo real com base nos dados mais recentes.

Em MySQL, uma função é criada utilizando o comando “*CREATE FUNCTION*”, seguida do nome da função, dos parâmetros de entrada e do tipo de valor que será retornado. Dentro do corpo da função, delimitado por “*BEGIN ... END*”, podem ser executadas instruções SQL, cálculos e verificações, e o resultado final é devolvido através do comando “*RETURN*”. Uma vez criada, a função pode ser chamada diretamente em consultas SQL, integrando-se como qualquer outro valor ou expressão. Por exemplo, é possível utilizá-la numa cláusula “*SELECT*” para calcular a contagem de viagens realizadas por um aluno.

Contagem de Atividade do Aluno

Para suprir a ausência da coluna “nr_vagens_realizadas” na tabela Aluno, criou-se a função “f_total_vagens_aluno”. Esta função recebe o identificador do aluno e devolve o número total de viagens concluídas. A adoção desta abordagem dinâmica elimina a necessidade de colunas derivadas, que exigem a utilização de gatilhos para se manterem atualizadas.

```

DELIMITER $$

CREATE FUNCTION f_total_vagens_aluno (p_id_aluno INT)
RETURNS INT DETERMINISTIC READS SQL DATA
BEGIN
    DECLARE v_total INT;
    SELECT COUNT(*) INTO v_total FROM Aluno_Viagem
    WHERE id_aluno = p_id_aluno;
    RETURN v_total;

```

```
        END $$  
DELIMITER ;
```

Medição da Taxa de Sucesso de Viagens

No contexto da avaliação de qualidade das atividades extracurriculares, foi desenvolvida a função “f_taxa_sucesso_viagem”. O seu objetivo é converter as avaliações individuais dos alunos num indicador percentual de sucesso para cada viagem (a percentagem de alunos que deu avaliação positiva relativamente ao total de alunos que realizou a viagem).

A função filtra as participações com classificação positiva (superior ou igual a 5) e calcula a sua proporção em relação ao total de avaliações recebidas. Inclui ainda uma salvaguarda lógica para evitar divisões por zero em viagens que ainda não possuam *feedbacks*. Este indicador é fundamental para a coordenação escolar decidir sobre a manutenção ou ajuste do formato das viagens disponibilizadas.

```
DELIMITER $$  
CREATE FUNCTION f_taxa_sucesso_viagem (p_id_viagem INT)  
RETURNS DECIMAL(5,2) DETERMINISTIC READS SQL DATA  
BEGIN  
    DECLARE v_total INT;  
    DECLARE v_positivas INT;  
    SELECT COUNT(*) INTO v_total FROM Aluno_Viagem  
        WHERE id_viagem = p_id_viagem;  
    IF v_total = 0 THEN RETURN 0.00;  
    END IF;  
    SELECT COUNT(*) INTO v_positivas FROM Aluno_Viagem  
        WHERE id_viagem = p_id_viagem AND avaliacao >= 5;  
    RETURN (v_positivas * 100.0 / v_total);  
END $$  
DELIMITER ;
```

Procedimentos

Os Procedimentos Armazenados foram adotados neste projeto para centralizar a lógica de negócio complexa no servidor de base de dados. Esta abordagem oferece

duas grandes vantagens: parametrização, permitindo filtrar resultados de forma dinâmica sem expor a complexidade das tabelas, e encapsulamento, garantindo que as regras de inserção e manipulação de dados são cumpridas rigorosamente, independentemente da aplicação externa que interage com o sistema.

Em MySQL, um procedimento é criado utilizando o comando “*CREATE PROCEDURE*”, seguida do nome do procedimento, dos parâmetros de entrada ou saída, e do bloco de instruções SQL delimitado por “*BEGIN ... END*”. Uma vez criado, o procedimento pode ser invocado através do comando:

```
CALL nome_do_procedimento(parametros)
```

Executando automaticamente as operações definidas e permitindo a reutilização de código em diferentes partes do sistema.

Análise do Perfil do Aluno (RM12)

No âmbito do requisito de manipulação 12, que visa gerar o "Perfil de Preferências de Disciplinas" de um aluno, optou-se pela implementação de um procedimento armazenado em detrimento de uma Vista. A principal razão para esta escolha técnica prende-se com a necessidade de parametrização: as vistas não aceitam parâmetros de entrada, o que obrigaria a carregar todos os dados de todos os alunos para depois filtrar na aplicação, um processo ineficiente.

O procedimento “sp_perfil_preferencias_aluno” recebe o identificador do aluno como argumento e processa os dados agregados, nomeadamente a média de avaliações e contagem de viagens, apenas para esse utilizador. O resultado é devolvido ordenado de forma decrescente as médias obtidas e, como critério de desempate, pelo volume de viagens realizadas, permitindo identificar rapidamente as áreas de maior interesse do estudante.

```
DELIMITER $$  
CREATE PROCEDURE sp_perfil_preferencias_aluno  
    (IN p_id_aluno INT)  
    BEGIN  
        SELECT a.id_aluno, a.nome, d.nome AS disciplina,  
              AVG(av.avaliacao) AS media_avaliacao,  
              COUNT(av.id_viagem) AS total_vagens  
        FROM Disciplina AS d  
        JOIN Aluno AS a ON d.id_disciplina = a.id_disciplina  
        JOIN Avaliacao AS av ON a.id_aluno = av.id_aluno  
        WHERE a.id_aluno = p_id_aluno  
        ORDER BY media_avaliacao DESC, total_vagens DESC;
```

```

        INNER JOIN Viagem_Disciplina AS vd
        ON d.id_disciplina = vd.id_disciplina
            INNER JOIN Aluno_Viagem AS av
            ON vd.id_viagem = av.id_viagem
                INNER JOIN Aluno AS a
                ON a.id_aluno = av.id_aluno
                WHERE a.id_aluno = p_id_aluno
            GROUP BY a.id_aluno, a.nome, d.id_disciplina, d.nome
            ORDER BY media_avaliacao DESC, total_viagens DESC;
        END $$

    DELIMITER ;

```

Pesquisa de Viagens (RM13)

Seguindo a mesma lógica, para responder ao requisito de manipulação 13 , optou-se igualmente pela utilização de um procedimento armazenado. Tal como no caso anterior, a necessidade de receber um termo de texto variável para filtrar os resultados justifica o uso de um procedimento em vez de uma vista estática.

O procedimento “sp_pesquisar_viagem” encapsula a lógica de pesquisa textual (“*LIKE*”), verificando a ocorrência do termo fornecido tanto no título como no resumo da viagem, e devolve a listagem ordenada alfabeticamente, abstraindo a complexidade da consulta para o utilizador final.

```

    DELIMITER $$

CREATE PROCEDURE sp_pesquisar_viagem (IN p_termo VARCHAR(100))
BEGIN
    SELECT id_viagem, titulo, resumo, data_publicacao FROM Viagem
        WHERE titulo LIKE CONCAT('%', p_termo, '%')
        OR resumo LIKE CONCAT('%', p_termo, '%')
        ORDER BY titulo ASC;
    END $$

    DELIMITER ;

```

Com a implementação destes Gatilhos, Funções e Procedimentos, garantimos que a base de dados faz mais do que apenas guardar informação: ela também a valida e processa. Ao centralizar as regras importantes no próprio SQL (como impedir datas erradas ou calcular médias), evitamos erros e garantimos que os dados estão sempre coerentes. Isto torna o sistema mais seguro e facilita muito o trabalho de quem for desenvolver a aplicação, pois as operações mais complexas já estão resolvidas dentro da base de dados.

6. Conclusões e Trabalho Futuro

O presente trabalho teve como objetivo o desenvolvimento de um Sistema de Base de Dados para a Escola Novos Horizontes, capaz de suportar a gestão de viagens virtuais, os seus intervenientes e os respetivos conteúdos e avaliações. Ao longo da realização do projeto, procuramos seguir o ciclo de vida de uma base de dados, acompanhado a matéria que ia sendo lecionada durante as aulas. Inicialmente, conseguimos cumprir os prazos estabelecidos no Diagrama de Gantt. Contudo, na fase de levantamento e análise de requisitos e, posteriormente na modelação conceptual, fomos incapazes de cumprir as datas pré-definidas. A fase de organização dos requisitos foi mais demorada do que o esperado. Primeiro porque tivemos dificuldade em definir e porque reconhecemos que esta era uma fase fundamental do trabalho que iria sustentar todo o projeto desenvolvido daí para a frente, desse modo, optamos por aumentar o período que havíamos definido para esta fase para garantir que os requisitos ficavam bem definidos e devidamente classificados. Do mesmo modo, a modelação conceitual foi mais demorada que o esperado, dado que, é nesta fase que trabalhamos diretamente com os requisitos de descrição definidos e procuramos assegurar que respeitamos os requisitos de descrição que havíamos definido anteriormente. Tivemos esse cuidado, pois o modelo conceptual atua como a base da nossa base de dados e, por esse motivo, se este fosse mal definido isso colocaria em causa a qualidade da base de dados que estávamos a construir.

Nas fases de implementação do modelo lógico e físico da base de dados, tivemos alguns desvios face ao planeamento inicial. Este atraso justificou-se sobretudo devido à desconsideração por parte dos alunos de um período de avaliação tão comprido. Apesar disso, a equipa conseguiu-se organizar no tempo restante de forma a conseguir deixar o trabalho pronto atempadamente, deixando ainda tempo para revisões do trabalho realizado. Além disso, notamos que o cuidado tido anteriormente na fase de modelação conceptual foi valiosa, no sentido de facilitar a modelação lógica do nosso trabalho.

Quanto ao relatório, consideramos que o mesmo reflete de forma fiel o trabalho desenvolvido ao longo do semestre. Tivemos o cuidado de justificar todas as decisões tomadas, tanto ao nível da modelação como da implementação, explicamos não só as soluções adotadas, mas também as razões que estiveram na sua base.

Consideramos que o relatório está bem estruturado e coerente, assim como reflete a evolução natural do projeto e permite uma leitura clara do processo seguido.

Em termos críticos, reconhecemos que uma melhor estimativa temporal para as fases poderia ter evitado alguns dos desvios verificados face ao diagrama de Gantt. Ainda assim, estes contribuíram para uma aprendizagem importante sobre a necessidade de flexibilidade no planeamento. Além disso, com o desenvolvimento deste projeto conseguimos compreender as diferentes fases do desenvolvimento de uma base de dados. Foi também possível perceber melhor a importância da análise cuidada dos requisitos, da modelação correta dos dados e da validação contínua das soluções implementadas. Para além dos aspectos técnicos, o trabalho contribuiu também para o desenvolvimento de competências transversais, como a gestão do tempo, a organização do trabalho, capacidade de adaptação a imprevistos e ainda o trabalho em equipa, fundamentais para a realização de projetos de maior dimensão e para o contexto profissional futuro. Para além disso, o acompanhamento do projeto facilitou a consolidação da matéria lecionada por criar um exemplo prático onde podíamos explorar e aplicar o que fomos aprendendo ao longo das aulas.

Deste modo, temos então uma base de dados funcional pronta para ser utilizada pela escola Novos Horizontes que poderá, no futuro, necessitar de manutenção e receber novas funcionalidades de acordo com o feedback tanto dos alunos como dos professores.

Em conclusão, consideramos que o projeto foi concluído com sucesso, apesar das dificuldades sentidas ao longo do seu desenvolvimento. Os objetivos inicialmente definidos foram alcançados e as soluções implementadas respondem de forma adequada aos requisitos estabelecidos, refletindo o empenho da equipa e a aprendizagem adquirida ao longo do trabalho.

7. Bibliografia

7.1 Referências

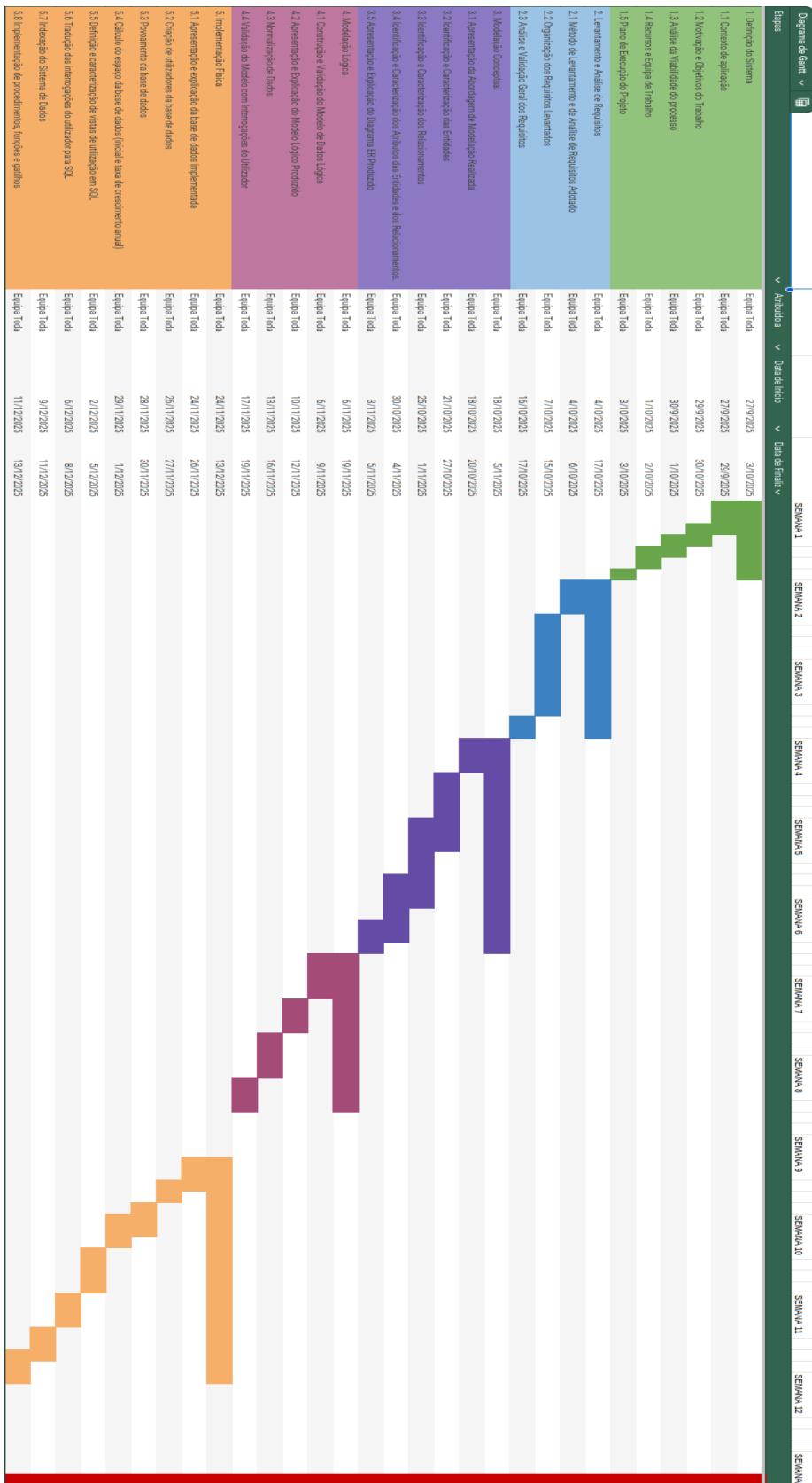
- Connolly, T.M. and Begg, C. (2015). Database systems : a practical approach to design, implementation and management. Harlow: Pearson Education Limited.
- <https://dev.mysql.com/doc/refman/8.4/en/storage-requirements.html#data-types-storage-reqs-innodb>

7.2 Lista de Siglas e Acrónimos

- SDB Sistema de Base de Dados
- RBAC Controlo de Acessos Baseado em Cargos
- BD Base de Dados
- RD Requisitos de Descrição
- URL Uniform Resource Locator
- RM Requisitos de Manipulação
- RC Requisitos de Controlo
- ER Entidade-Relacionamento
- SGBD Sistema de Gestão de Base de Dados
- PK Chave Primária
- FK Chave Estrangeira
- 1FN Primeira Forma Normal
- 2FN Segunda Forma Normal
- 3FN Terceira Forma Normal
- SQL *Structured Query Language*
- MB *MegaByte*

Anexos

I. Diagrama de Gantt



II. Povoamento RelaX

```
group: ViagensNH

Professor = {
    id_professor:number, nome:string
    5001, 'Marcelo Almeida'
    5002, 'Sofia Nunes'
    5003, 'Rita Carvalho'
    5004, 'Pedro Duarte'
}

Aluno = {
    id_aluno:number, nome:string, genero:string, ano:number,
letra:string
    2001, 'Ana Silva', 'F', 9, 'A'
    2002, 'Bruno Costa', 'M', 6, 'B'
    2003, 'Carla Rocha', 'F', 11, 'A'
    2004, 'Diogo Pereira', 'M', 7, 'C'
    2005, 'Eva Martins', 'F', 6, 'B'
    2006, 'Fábio Santos', 'M', 12, 'A'
    2007, 'Gabriela Lopes', 'F', 10, 'C'
    2008, 'Henrique Dias', 'M', 12, 'A'
    2009, 'Inês Faria', 'F', 7, 'B'
    2010, 'João Neves', 'M', 6, 'C'
}

Viagem = {
    id_viagem:number, titulo:string, data_publicacao:string,
resumo:string, id_professor:number
    1, 'Lisboa Histórica', '2024-03-12', 'Exploração aprofundada
da história de Lisboa, analisando os seus monumentos, evolução
urbana e impacto cultural ao longo dos séculos.', 5001
```

2, 'Porto Animal', '2024-03-20', 'Viagem educativa dedicada ao estudo da fauna, abordando biodiversidade, habitats naturais e a importância da preservação das espécies.', 5002

3, 'Madrid Artística', '2024-04-05', 'Percurso cultural pelos principais museus e espaços artísticos de Madrid, contextualizando as obras e os seus autores.', 5003

4, 'Paris Monumental', '2024-04-18', 'Viagem interdisciplinar que relaciona matemática, arquitetura e história através da análise dos monumentos icónicos de Paris.', 5001

5, 'Roma Antiga', '2024-05-01', 'Exploração detalhada da civilização romana, destacando arquitetura, política e legado cultural da Roma Antiga.', 5004

6, 'Berlim Contemporânea', '2024-05-15', 'Análise da história recente de Berlim, com foco na divisão da cidade, na Guerra Fria e na reunificação alemã.', 5003

7, 'Ciência em Munique', '2024-05-25', 'Viagem virtual dedicada à ciência e tecnologia, explorando museus científicos e inovações desenvolvidas em Munique.', 5002

8, 'Barcelona Criativa', '2024-06-05', 'Exploração artística e arquitetónica da cidade de Barcelona, com destaque para as obras de Antoni Gaudí.', 5001

}

```
Aluno_Viagem = {  
    id_aluno:number,      id_viagem:number,      avaliacao:number,  
    comentario:string,    data_realizacao:string  
    2001, 1, 10, 'Viagem extremamente enriquecedora que ajudou a  
    compreender melhor a história e identidade cultural de  
    Lisboa.', '2024-03-13'  
    2001, 4, 8, 'Gostei muito da ligação entre matemática e  
    arquitetura, especialmente na análise dos monumentos  
    parisienses.', '2024-04-19'  
    2002, 1, 9, 'Atividade bastante interessante que reforçou os  
    meus conhecimentos sobre a história de Portugal.',  
    '2024-03-13'
```

2002, 2, 10, 'Experiência educativa e envolvente, com conteúdos bem explicados sobre os animais e os seus habitats.', '2024-03-21'

2003, 2, 8, 'Aprendi bastante sobre biodiversidade e preservação ambiental ao longo da viagem.', '2024-03-21'

2003, 3, 9, 'Viagem cultural muito inspiradora que despertou ainda mais o meu interesse pela arte.', '2024-04-06'

2004, 3, 7, 'Conteúdos bem estruturados que ajudaram a compreender melhor o contexto artístico de Madrid.', '2024-04-06'

2005, 5, 10, 'Excelente abordagem à história romana, com explicações claras e exemplos bem escolhidos.', '2024-05-02'

2005, 8, 7, 'Gostei especialmente da arquitetura de Gaudí e da forma criativa como os conteúdos foram apresentados.', '2024-06-06'

2006, 2, 8, 'Viagem muito bem organizada e educativa, com exemplos claros e interessantes.', '2024-03-21'

2006, 7, 8, 'Experiência bastante interessante que mostrou a importância da ciência e da inovação.', '2024-05-26'

2007, 4, 9, 'A ligação entre diferentes disciplinas tornou a viagem muito mais interessante.', '2024-04-19'

2008, 3, 8, 'Viagem cultural rica e bem contextualizada, com bons exemplos artísticos.', '2024-04-06'

2008, 6, 9, 'Gostei muito da forma como a história recente de Berlim foi explicada.', '2024-05-16'

2009, 5, 7, 'Aprendi bastante sobre a civilização romana e o seu impacto na atualidade.', '2024-05-02'

2010, 1, 8, 'Viagem interessante e bem organizada, com conteúdos claros e acessíveis.', '2024-03-13'

}

```
Disciplina = {
    id_disciplina:number, nome:string, ano:number
1, 'Educação Visual', 5
2, 'História', 6
3, 'Matemática', 7
```

```

4, 'Ciências Naturais', 8
5, 'Física e Química', 9
6, 'Português', 10
}

Viagem_Disciplina = {
    id_viagem:number, id_disciplina:number
1,2
2,4
3,1
3,2
4,2
4,3
5,2
6,2
7,5
8,1
}

País = {
    id_pais:number, pais:string
1, 'Portugal'
2, 'Espanha'
3, 'França'
4, 'Itália'
5, 'Alemanha'
}

Cidade = {
    id_cidade:number, cidade:string, id_pais:number
1, 'Lisboa', 1
2, 'Porto', 1
3, 'Madrid', 2
4, 'Barcelona', 2
5, 'Paris', 3
6, 'Marselha', 3
}

```

```

    7, 'Roma', 4
    8, 'Milão', 4
    9, 'Berlim', 5
    10, 'Munique', 5
}

Localizacao = {
    id_localizacao:number, nome:string, id_cidade:number
    1, 'Torre de Belém', 1
    2, 'Zoológico', 1
    3, 'Ribeira', 2
    4, 'Sé do Porto', 2
    5, 'Museu do Prado', 3
    6, 'Parque Güell', 4
    7, 'Torre Eiffel', 5
    8, 'Louvre', 5
    9, 'Coliseu', 7
    10, 'Porta de Brandemburgo', 9
}

Conteudo = {
    id_conteudo:number, url:string, descricao:string,
    titulo:string, tipo:string, data_captura:string,
    id_viagem:number, id_localizacao:number
    1, 'https://img1.jpg', 'Fotografia detalhada da Torre de Belém, evidenciando os elementos arquitetónicos manuelinos.', 'Torre de Belém', 'F', '2024-03-12', 1, 1
    2, 'https://vid1.mov', 'Vídeo educativo sobre o Oceanário de Lisboa, mostrando diversas espécies marinhas.', 'Oceanário de Lisboa', 'V', '2024-03-12', 1, 2
    3, 'https://img2.jpg', 'Imagen panorâmica da Ribeira do Porto, destacando a arquitetura tradicional.', 'Ribeira do Porto', 'F', '2024-03-20', 2, 3
    4, 'https://img3.jpg', 'Fotografia da Sé do Porto ao entardecer, evidenciando o seu valor histórico.', 'Sé do Porto', 'F', '2024-03-20', 2, 4
}

```

5, 'https://img4.jpg', 'Imagen do Museu do Prado com destaque para a sua importância artística.', 'Museu do Prado', 'F', '2024-04-05', 3, 5

6, 'https://vid2.mov', 'Vídeo panorâmico do Parque Güell, destacando a arquitetura modernista.', 'Parque Güell', 'V', '2024-06-05', 8, 6

7, 'https://img5.jpg', 'Fotografia noturna da Torre Eiffel iluminada.', 'Torre Eiffel', 'F', '2024-04-18', 4, 7

8, 'https://vid3.mov', 'Vídeo da entrada do Museu do Louvre, mostrando a pirâmide central.', 'Museu do Louvre', 'V', '2024-04-18', 4, 8

9, 'https://vid4.mov', 'Vídeo ilustrativo do Coliseu Romano, explorando a sua estrutura.', 'Coliseu Romano', 'V', '2024-05-01', 5, 9

10, 'https://img6.jpg', 'Fotografia da Porta de Brandemburgo, símbolo histórico de Berlim.', 'Porta de Brandemburgo', 'F', '2024-05-15', 6, 10

}

III. Expressões Álgebra Relacional

RM7:

```
Π Viagem.id_viagem, titulo, resumo, data_publicacao ( τ data_publicacao desc (Viagem ×  
Viagem.id_viagem = Viagem_Disciplina.id_viagem ( σ id_disciplina = 2  
(Viagem_Disciplina))))
```

RM8:

```
Π Aluno.id_aluno, nome, avaliacao ( τ nome asc (Aluno ×  
Aluno.id_aluno = Aluno_Viagem.id_aluno ( σ id_viagem = 1  
(Aluno_Viagem))))
```

RM10:

```
τ Media desc ( ∀ Aluno_Viagem.id_viagem; MIN(Viagem.titulo) → Titulo, AVG(avaliacao) → Media  
(Aluno_Viagem × Viagem))
```

RM11:

```
τ NrViagens desc ( ∀ id_professor; MIN(nome) → Professor, COUNT(id_viagem) → NrViagens,  
MIN(NM) → AvaliacaoMin, MAX(NM) → AvaliacaoMax ( ∀ id_viagem; AVG(avaliacao) → NM (Aluno_Viagem)  
× (Professor × Viagem)))
```


IV. Script criação SQL

```
-- --  
-- Universidade do Minho  
-- Disciplina de Bases de Dados  
-- A Linguagem SQL  
-- Trabalho Prático: Viagens virtuais da escola Novos  
Horizontes  
-- Implementação do Esquema Global  
-- MySQL 8.0.28 (MySQL Community Server GPL)  
-- 2025, Novembro/Dezembro  
-- --  
  
-- Criação da Base de Dados  
-- DROP DATABASE ViagensNH;  
CREATE DATABASE IF NOT EXISTS ViagensNH  
CHARACTER SET utf8mb4  
COLLATE utf8mb4_unicode_ci;  
  
-- Selecionar a base de dados a utilizar  
USE ViagensNH;  
  
-- Criação das tabelas da base de dados  
--  
-- Criação da Tabela Professor  
CREATE TABLE IF NOT EXISTS Professor (  
    id_professor INT NOT NULL,  
    nome VARCHAR (100) NOT NULL,  
    PRIMARY KEY (id_professor)  
);  
  
-- Criação da tabela Aluno  
CREATE TABLE IF NOT EXISTS Aluno (  
    id_aluno INT NOT NULL,  
    nome VARCHAR(100) NOT NULL,  
    genero ENUM('M', 'F') NOT NULL,  
    ano TINYINT NOT NULL,
```

```

        letra CHAR(1) NOT NULL,
        PRIMARY KEY (id_aluno)
    ) ;

-- Criação da tabela Viagem
CREATE TABLE IF NOT EXISTS Viagem (
    id_viagem INT AUTO_INCREMENT NOT NULL,
    titulo VARCHAR(100) NOT NULL,
    data_publicacao DATE NOT NULL DEFAULT (CURRENT_DATE),
    resumo VARCHAR(1000) NOT NULL,
    id_professor INT NOT NULL,
    PRIMARY KEY (id_viagem),
    FOREIGN KEY (id_professor)
        REFERENCES Professor(id_professor)
        ON DELETE NO ACTION
        ON UPDATE CASCADE
) ;

-- Criação da tabela Aluno_Viagem
CREATE TABLE IF NOT EXISTS Aluno_Viagem (
    id_aluno INT NOT NULL,
    id_viagem INT NOT NULL,
    avaliacao INT NOT NULL CHECK (avaliacao BETWEEN 0 AND 10),
    comentario VARCHAR(200) NOT NULL,
    data_realizacao DATE NOT NULL DEFAULT (CURRENT_DATE),
    PRIMARY KEY(id_aluno,id_viagem),
    FOREIGN KEY (id_aluno)
        REFERENCES Aluno(id_aluno)
        ON DELETE NO ACTION
        ON UPDATE CASCADE,
    FOREIGN KEY (id_viagem)
        REFERENCES Viagem(id_viagem)
        ON DELETE CASCADE
        ON UPDATE NO ACTION
) ;

```

```

-- Criação da tabela Disciplina
CREATE TABLE IF NOT EXISTS Disciplina (
    id_disciplina INT NOT NULL AUTO_INCREMENT,
    nome VARCHAR(45) NOT NULL,
    ano TINYINT NOT NULL,
    PRIMARY KEY (id_disciplina)
);

-- Criação da tabela Viagem_Disciplina
CREATE TABLE IF NOT EXISTS Viagem_Disciplina (
    id_viagem INT NOT NULL,
    id_disciplina INT NOT NULL,
    PRIMARY KEY (id_viagem, id_disciplina),
    FOREIGN KEY (id_viagem)
        REFERENCES Viagem(id_viagem)
        ON DELETE CASCADE
        ON UPDATE NO ACTION,
    FOREIGN KEY (id_disciplina)
        REFERENCES Disciplina(id_disciplina)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
);

```

-- Criação da tabela País

```

CREATE TABLE IF NOT EXISTS Pais (
    id_pais INT NOT NULL AUTO_INCREMENT,
    pais VARCHAR(35) NOT NULL,
    PRIMARY KEY (id_pais)
);

```

-- Criação da tabela Cidade

```

CREATE TABLE IF NOT EXISTS Cidade (
    id_cidade INT NOT NULL AUTO_INCREMENT,
    cidade VARCHAR(35) NOT NULL,
    id_pais INT NOT NULL,

```

```

        PRIMARY KEY (id_cidade),
        FOREIGN KEY (id_pais)
            REFERENCES Pais(id_pais)
            ON DELETE NO ACTION
            ON UPDATE NO ACTION
    ) ;

-- Criação da tabela Localizacao
CREATE TABLE IF NOT EXISTS Localizacao (
    id_localizacao INT NOT NULL AUTO_INCREMENT,
    nome VARCHAR(70) NOT NULL,
    id_cidade INT NOT NULL,
    PRIMARY KEY (id_localizacao),
    FOREIGN KEY (id_cidade)
        REFERENCES Cidade(id_cidade)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
) ;

-- Criação da tabela Conteudo
CREATE TABLE IF NOT EXISTS Conteudo (
    id_conteudo INT NOT NULL AUTO_INCREMENT,
    url VARCHAR(150) NOT NULL,
    descricao VARCHAR(500) NOT NULL,
    titulo VARCHAR(50) NOT NULL,
    tipo ENUM('F', 'V') NOT NULL,
    data_captura DATE NOT NULL,
    id_viagem INT NOT NULL,
    id_localizacao INT NOT NULL,
    PRIMARY KEY (id_conteudo),
    FOREIGN KEY (id_viagem)
        REFERENCES Viagem(id_viagem)
        ON DELETE CASCADE
        ON UPDATE NO ACTION,
    FOREIGN KEY (id_localizacao)
        REFERENCES Localizacao(id_localizacao)
)

```

```
        ON DELETE NO ACTION
        ON UPDATE NO ACTION
    ) ;

-- <fim>
-- Afonso Barros a112178
-- Luís Pedrosa a112081
-- Diogo Pedrosa a103178
-- Matheus Azevedo a111430
-- DROP DATABASE viagensnh;
```


V. Script povoamento SQL

```
-- --
-- Universidade do Minho
-- Disciplina de Bases de Dados
-- A Linguagem SQL
-- Trabalho Prático: Viagens virtuais da escola Novos
Horizontes
-- Povoamento da Base de Dados
-- MySQL 8.0.28 (MySQL Community Server GPL)
-- 2025, Novembro/Dezembro
-- --

-- Povoamento da tabela Professor
INSERT INTO Professor (id_professor, nome)
VALUES
    (5001, 'Marcelo Almeida'),
    (5002, 'Sofia Nunes'),
    (5003, 'Rita Carvalho'),
    (5004, 'Pedro Mendes')
;

-- Povoamento da tabela Aluno
INSERT INTO Aluno (id_aluno, nome, genero, ano, letra)
VALUES
    (2001, 'Ana Silva', 'F', 9, 'A'),
    (2002, 'Bruno Costa', 'M', 6, 'B'),
    (2003, 'Carla Rocha', 'F', 11, 'A'),
    (2004, 'Diogo Pereira', 'M', 7, 'C'),
    (2005, 'Eva Martins', 'F', 6, 'B'),
    (2006, 'Fábio Santos', 'M', 12, 'A'),
    (2007, 'Gabriela Lopes', 'F', 10, 'C'),
    (2008, 'Henrique Dias', 'M', 12, 'A'),
    (2009, 'Inês Faria', 'F', 7, 'B'),
    (2010, 'João Neves', 'M', 6, 'C')
;
```

```

-- Povoamento da tabela Viagem
INSERT INTO Viagem (titulo, data_publicacao, resumo,
id_professor)
VALUES
    ('Lisboa Histórica', '2024-03-12', 'Exploração
aproximada da história de Lisboa, analisando os seus
monumentos, evolução urbana e impacto cultural ao longo dos
séculos.', 5001),
    ('Porto Animal', '2024-03-20', 'Viagem educativa
dedicada ao estudo da fauna, abordando biodiversidade,
habitats naturais e a importância da preservação das
espécies.', 5002),
    ('Madrid Artística', '2024-04-05', 'Percorso cultural
pelos principais museus e espaços artísticos de Madrid,
contextualizando as obras e os seus autores.', 5003),
    ('Paris Monumental', '2024-04-18', 'Viagem
interdisciplinar que relaciona matemática, arquitetura e
história através da análise dos monumentos icónicos de
Paris.', 5001),
    ('Roma Antiga', '2024-05-01', 'Exploração detalhada da
civilização romana, destacando arquitetura, política e legado
cultural da Roma Antiga.', 5004),
    ('Berlim Contemporânea', '2024-05-15', 'Análise da
história recente de Berlim, com foco na divisão da cidade, na
Guerra Fria e na reunificação alemã.', 5003),
    ('Ciência em Munique', '2024-05-25', 'Viagem virtual
dedicada à ciência e tecnologia, explorando museus científicos
e inovações desenvolvidas em Munique.', 5002),
    ('Barcelona Criativa', '2024-06-05', 'Exploração
artística e arquitetónica da cidade de Barcelona, com destaque
para as obras de Antoni Gaudí.', 5001)
;

-- Povoamento da tabela Aluno_Viagem

```

```
INSERT INTO Aluno_Viagem (id_aluno, id_viagem, avaliacao,
comentario, data_realizacao)

VALUES

(2001, 1, 10, 'Viagem extremamente enriquecedora que
ajudou a compreender melhor a história e identidade cultural
de Lisboa.', '2024-03-13'),

(2001, 4, 8, 'Gostei muito da ligação entre matemática
e arquitetura, especialmente na análise dos monumentos
parisienses.', '2024-04-19'),

(2002, 1, 9, 'Atividade bastante interessante que
reforçou os meus conhecimentos sobre a história de Portugal.',
'2024-03-13'),

(2002, 2, 10, 'Experiência educativa e envolvente, com
conteúdos bem explicados sobre os animais e os seus
habitats.', '2024-03-21'),

(2003, 2, 8, 'Aprendi bastante sobre biodiversidade e
preservação ambiental ao longo da viagem.', '2024-03-21'),

(2003, 3, 9, 'Viagem cultural muito inspiradora que
despertou ainda mais o meu interesse pela arte.',
'2024-04-06'),

(2004, 3, 7, 'Conteúdos bem estruturados que ajudaram
a compreender melhor o contexto artístico de Madrid.',
'2024-04-06'),

(2005, 5, 10, 'Excelente abordagem à história romana,
com explicações claras e exemplos bem escolhidos.',
'2024-05-02'),

(2005, 8, 7, 'Gostei especialmente da arquitetura de
Gaudí e da forma criativa como os conteúdos foram
apresentados.', '2024-06-06'),

(2006, 2, 8, 'Viagem muito bem organizada e educativa,
com exemplos claros e interessantes.', '2024-03-21'),

(2006, 7, 8, 'Experiência bastante interessante que
mostrou a importância da ciência e da inovação.',
'2024-05-26'),

(2007, 4, 9, 'A ligação entre diferentes disciplinas
tornou a viagem muito mais interessante.', '2024-04-19'),
```

```

(2008, 3, 8, 'Viagem cultural rica e bem
contextualizada, com bons exemplos artísticos.',

'2024-04-06'),
(2008, 6, 9, 'Gostei muito da forma como a história
recente de Berlim foi explicada.', '2024-05-16'),
(2009, 5, 7, 'Aprendi bastante sobre a civilização
romana e o seu impacto na atualidade.', '2024-05-02'),
(2010, 1, 8, 'Viagem interessante e bem organizada,
com conteúdos claros e acessíveis.', '2024-03-13')
;

-- Povoamento da tabela Disciplina
INSERT INTO Disciplina (nome, ano)
VALUES
('Educação Visual', 5),
('História', 6),
('Matemática', 7),
('Ciências Naturais', 8),
('Física e Química', 9),
('Português', 10)
;

-- Povoamento da tabela Viagem_Disciplina
INSERT INTO Viagem_Disciplina (id_viagem, id_disciplina)
VALUES
(1,2),
(2,4),
(3,1),
(3,2),
(4,2),
(4,3),
(5,2),
(6,2),
(7,5),
(8,1)
;
```

```

-- Povoamento da tabela Pais
INSERT INTO Pais (pais)
VALUES
    ('Portugal'),
    ('Espanha'),
    ('França'),
    ('Itália'),
    ('Alemanha')

;

-- Povoamento da tabela Cidade
INSERT INTO Cidade (cidade, id_pais)
VALUES
    ('Lisboa', 1),
    ('Porto', 1),
    ('Madrid', 2),
    ('Barcelona', 2),
    ('Paris', 3),
    ('Marselha', 3),
    ('Roma', 4),
    ('Milão', 4),
    ('Berlim', 5),
    ('Munique', 5)

;

-- Povoamento da tabela Localizacao
INSERT INTO Localizacao (nome, id_cidade)
VALUES
    ('Torre de Belém', 1),
    ('Zoológico', 1),
    ('Ribeira', 2),
    ('Sé do Porto', 2),
    ('Museu do Prado', 3),
    ('Parque Güell', 4),
    ('Torre Eiffel', 5),

```

```

        ('Louvre', 5),
        ('Coliseu', 7),
        ('Porta de Brandemburgo', 9)
;

-- Povoamento da tabela Conteudo
INSERT INTO Conteudo (url, descricao, titulo, tipo,
data_captura, id_viaj, id_localizacao)
VALUES
    ('https://img1.jpg', 'Fotografia detalhada da Torre de Belém, evidenciando os elementos arquitetónicos manuelinos.', 'Torre de Belém', 'F', '2024-03-12', 1, 1),
    ('https://vid1.mov', 'Vídeo educativo sobre o Oceanário de Lisboa, mostrando diversas espécies marinhas.', 'Oceanário de Lisboa', 'V', '2024-03-12', 1, 2),
    ('https://img2.jpg', 'Imagen panorâmica da Ribeira do Porto, destacando a arquitetura tradicional.', 'Ribeira do Porto', 'F', '2024-03-20', 2, 3),
    ('https://img3.jpg', 'Fotografia da Sé do Porto ao entardecer, evidenciando o seu valor histórico.', 'Sé do Porto', 'F', '2024-03-20', 2, 4),
    ('https://img4.jpg', 'Imagen do Museu do Prado com destaque para a sua importância artística.', 'Museu do Prado', 'F', '2024-04-05', 3, 5),
    ('https://vid2.mov', 'Vídeo panorâmico do Parque Güell, destacando a arquitetura modernista.', 'Parque Güell', 'V', '2024-06-05', 8, 6),
    ('https://img5.jpg', 'Fotografia noturna da Torre Eiffel iluminada.', 'Torre Eiffel', 'F', '2024-04-18', 4, 7),
    ('https://vid3.mov', 'Vídeo da entrada do Museu do Louvre, mostrando a pirâmide central.', 'Museu do Louvre', 'V', '2024-04-18', 4, 8),
    ('https://vid4.mov', 'Vídeo ilustrativo do Coliseu Romano, explorando a sua estrutura.', 'Coliseu Romano', 'V', '2024-05-01', 5, 9),

```

```
('https://img6.jpg', 'Fotografia da Porta de Brandemburgo, símbolo histórico de Berlim.', 'Porta de Brandemburgo', 'F', '2024-05-15', 6, 10)
;

-- <fim>
-- Afonso Barros a112178
-- Luís Pedrosa a112081
-- Diogo Pedrosa a103178
-- Matheus Azevedo a111430
```


VI. ***Script Queries, Índices, Vistas, Funções, Gatilhos, Procedimentos***

```
-- --
-- Universidade do Minho
-- Disciplina de Bases de Dados
-- A Linguagem SQL
-- Trabalho Prático: Viagens virtuais da escola Novos
Horizontes
-- Implementação das definições (vistas, queries, índices,
gatilhos, funções e procedimentos)
-- MySQL 8.0.28 (MySQL Community Server GPL)
-- 2025, Novembro/Dezembro
-- --
USE ViagensNH;

--
=====
=====
-- 1. VISTAS (VIEWS) - Ponto 5.5
--
=====
=====

-- Vista para simplificar a apresentação dos detalhes das
viagens
CREATE OR REPLACE VIEW v_detalhes_viagem AS
    SELECT v.id_viagem, v.titulo, v.data_publicacao, p.nome AS Professor,
    GROUP_CONCAT(CONCAT(d.nome, ' ', d.ano, '° ano'))
    SEPARATOR ', ') AS Disciplinas FROM Viagem AS v
        INNER JOIN Professor AS p ON v.id_professor =
p.id_professor
```

```

        INNER JOIN Viagem_Disciplina AS vd ON v.id_viagem =
vd.id_viagem
            INNER JOIN Disciplina AS d ON vd.id_disciplina =
d.id_disciplina
                GROUP BY v.id_viagem
                ORDER BY v.id_viagem;

-- Vista para consultar o Top 10 de Alunos com mais viagens
realizadas
CREATE OR REPLACE VIEW v_top_alunos AS
    SELECT a.id_aluno, a.nome, CONCAT(a.ano, 'º', ' ', a.letra)
AS turma, COUNT(av.id_viagem) AS total_vagens FROM Aluno AS a
        INNER JOIN Aluno_Viagem AS av ON a.id_aluno =
av.id_aluno
        GROUP BY a.id_aluno, a.nome
        ORDER BY total_vagens DESC, MAX(av.data_realizacao) ASC
        LIMIT 10;

-- =====
-- 2. INTERROGAÇÕES DO UTILIZADOR (QUERIES) - Ponto 5.6
-- =====

-- RM7: Listar todas as viagens disponíveis para uma
determinada disciplina dado o seu identificador, ordenadas da
mais recente para a mais antiga (apresentando o id, o título,
o resumo e a data de publicação)
-- Neste caso, para efeito de demonstração utilizamos
id_disciplina = 2
SELECT v.id_viagem, v.titulo, v.resumo, v.data_publicacao FROM
Viagem AS v
        INNER JOIN Viagem_Disciplina AS vd ON v.id_viagem =
vd.id_viagem

```

```

        INNER JOIN Disciplina AS d ON vd.id_disciplina =
d.id_disciplina
WHERE d.id_disciplina = 2
ORDER BY v.data_publicacao DESC;

-- RM8: Listar os alunos (id e nome) que realizaram uma
determinada viagem, ordenados alfabeticamente, apresentando a
sua respetiva avaliação
-- Neste caso, para efeito de demonstração utilizamos
id_viagem = 1
SELECT a.id_aluno, a.nome, av.avaliacao FROM Aluno AS a
        INNER JOIN Aluno_Viagem AS av ON a.id_aluno = av.id_aluno
WHERE av.id_viagem = 1
ORDER BY a.nome ASC;

-- RM10: Consultar a classificação média atual de todas as
viagens, apresentando o respetivo identificador, título e o
valor da média calculada
SELECT v.id_viagem, v.titulo, AVG(av.avaliacao) AS Media FROM
Viagem AS v
        INNER JOIN Aluno_Viagem AS av ON v.id_viagem =
av.id_viagem
GROUP BY v.id_viagem;

-- RM11: Listar o desempenho dos professores, apresentando o
id do professor, nome do professor e o número total de viagens
virtuais criadas por cada um, ordenado por ordem decrescente
da quantidade de viagens criadas. Além disso, devem ser
apresentadas as classificações médias mínimas e máximas das
suas viagens.
SELECT p.id_professor, p.nome, COUNT(v.id_viagem) AS
total_viagens, MIN(r.nm) AS AvaliacaoMin, MAX(r.nm) AS
AvaliacaoMax
FROM Professor AS p
LEFT JOIN Viagem AS v ON p.id_professor = v.id_professor
LEFT JOIN (

```

```

        SELECT id_viagem, AVG(avaliacao) AS nm
        FROM Aluno_Viagem
        GROUP BY id_viagem) AS r ON v.id_viagem = r.id_viagem
GROUP BY p.id_professor
ORDER BY total_vagens DESC;

-- =====
-- 3. ÍNDICES (INDEXES) - Ponto 5.7
-- =====
-- Índice para otimizar a ordenação cronológica (RM7)
CREATE INDEX idx_viagem_data ON Viagem(data_publicacao);

-- Índice para acelerar a ordenação por título (RM13)
CREATE INDEX idx_viagem_titulo ON Viagem(titulo);

-- Índice para otimizar a pesquisa e ordenação alfabética dos
alunos (RM8)
CREATE INDEX idx_aluno_nome ON Aluno(nome);

-- =====
-- 4. GATILHOS, FUNÇÕES E PROCEDIMENTOS - Ponto 5.8
-- =====
-- -----
-- -----
-- GATILHOS (TRIGGERS)

```

```

--  

-----  

----  
  

-- Gatilho: t_valida_cronologia  

-- Objetivo: Impedir que uma viagem seja realizada antes da  

sua data de publicação.  

DELIMITER $$  

CREATE TRIGGER t_valida_cronologia BEFORE INSERT ON  

Aluno_Viagem  

FOR EACH ROW  

BEGIN  

DECLARE data_pub DATE;  
  

SELECT data_publicacao INTO data_pub FROM Viagem  

WHERE id_viagem = NEW.id_viagem;  
  

IF NEW.data_realizacao < data_pub THEN  

    SIGNAL SQLSTATE '45000'  

        SET MESSAGE_TEXT = 'Erro de Integridade: A viagem não  

pode ser realizada antes da sua data de publicação.';  

END IF;  

END $$  

DELIMITER ;  
  

--  

-----  

----  

-- FUNÇÕES (FUNCTIONS)  

--  

-----  

----  
  

-- Função: f_total_vagens_aluno  

-- Objetivo: Calcular dinamicamente o nº de viagens realizados  

por um aluno.

```

```

DELIMITER $$

CREATE FUNCTION f_total_vagens_aluno (p_id_aluno INT)
    RETURNS INT DETERMINISTIC READS SQL DATA
BEGIN
    DECLARE v_total INT;

    SELECT COUNT(*) INTO v_total FROM Aluno_Viagem
        WHERE id_aluno = p_id_aluno;

    RETURN v_total;
END $$

DELIMITER ;

-- Função: f_taxa_sucesso_viagem
-- Objetivo: Calcular percentagem de avaliações positivas (>=
5) de uma viagem.

DELIMITER $$

CREATE FUNCTION f_taxa_sucesso_viagem (p_id_viagem INT)
    RETURNS DECIMAL(5,2) DETERMINISTIC READS SQL DATA
BEGIN
    DECLARE v_total INT;
    DECLARE v_positivas INT;

    SELECT COUNT(*) INTO v_total FROM Aluno_Viagem WHERE
id_viagem = p_id_viagem;

    IF v_total = 0 THEN RETURN 0.00;
    END IF;

    SELECT COUNT(*) INTO v_positivas FROM Aluno_Viagem
        WHERE id_viagem = p_id_viagem AND avaliacao >= 5;

    RETURN (v_positivas / v_total) * 100;
END $$

DELIMITER ;

```

```

--  

-----  

-- PROCEDIMENTOS (STORED PROCEDURES)  

--  

-----  

--  

-----  

-- Procedimento: sp_perfil_preferencias_aluno (RM12)  

-- Objetivo: Listar as preferências de um aluno específico.  

DELIMITER $$  

CREATE PROCEDURE sp_perfil_preferencias_aluno (IN p_id_aluno  

INT)  

BEGIN  

    SELECT a.id_aluno, a.nome, d.nome AS disciplina,  

AVG(av.avaliacao) AS media_avaliacao, COUNT(av.id_viagem) AS  

total_vagens FROM Disciplina AS d  

    INNER JOIN Viagem_Disciplina AS vd ON d.id_disciplina  

= vd.id_disciplina  

    INNER JOIN Aluno_Viagem AS av ON vd.id_viagem =  

av.id_vagem  

    INNER JOIN Aluno AS a ON a.id_aluno = av.id_aluno  

WHERE a.id_aluno = p_id_aluno  

GROUP BY a.id_aluno, a.nome, d.id_disciplina, d.nome  

ORDER BY media_avaliacao DESC, total_vagens DESC;  

END $$  

DELIMITER ;  

-- Procedimento: sp_pesquisar_viagem (RM13)  

-- Objetivo: Pesquisar viagens por texto (título ou resumo).  

DELIMITER $$  

CREATE PROCEDURE sp_pesquisar_viagem (IN p_termo VARCHAR(100))  

BEGIN  

    SELECT id_vagem, titulo, resumo, data_publicacao FROM  

Viagem

```

```
        WHERE titulo LIKE CONCAT('%', p_termo, '%') OR resumo
        LIKE CONCAT('%', p_termo, '%')
        ORDER BY titulo ASC;
END $$

DELIMITER ;

-- <fim>
-- Afonso Barros a112178
-- Luís Pedrosa a112081
-- Diogo Pedrosa a103178
-- Matheus Azevedo a111430
```

VII. *Script Roles e Users*

```
-- Criação dos roles

CREATE ROLE IF NOT EXISTS 'aluno';

CREATE ROLE IF NOT EXISTS 'professor';

CREATE ROLE IF NOT EXISTS 'bibliotecario';

-- Dar permissões ao role aluno

GRANT INSERT,SELECT ON ViagensNH.Aluno_Viagem TO 'aluno';

GRANT SELECT ON ViagensNH.Viagem TO 'aluno';

GRANT SELECT ON ViagensNH.Conteudo TO 'aluno';

GRANT SELECT ON ViagensNH.Viagem_Disciplina TO 'aluno';

GRANT SELECT ON ViagensNH.Disciplina TO 'aluno';

GRANT SELECT ON ViagensNH.Localizacao TO 'aluno';

GRANT SELECT ON ViagensNH.Cidade TO 'aluno';

GRANT SELECT ON ViagensNH.Pais TO 'aluno';

-- Permissões vistas, procedimentos, funções ao cargo aluno

GRANT SELECT ON ViagensNH.v_detalhes_viagem TO 'aluno';

GRANT SELECT ON ViagensNH.v_top_alunos TO 'aluno';

GRANT EXECUTE ON FUNCTION ViagensNH.f_total_vagens_aluno TO
' aluno';

GRANT EXECUTE ON PROCEDURE
ViagensNH.sp_perfil_preferencias_aluno TO 'aluno';

GRANT EXECUTE ON PROCEDURE ViagensNH.sp_pesquisar_vagem TO
' aluno';

-- REVOKE GRANT OPTION ON ViagensNH.* FROM 'aluno';
```

```

-- Dar permissões ao role professor

GRANT SELECT ON ViagensNH.* TO 'professor';

GRANT INSERT,UPDATE ON ViagensNH.Viagem TO 'professor';

GRANT INSERT,UPDATE ON ViagensNH.Conteudo TO 'professor';

GRANT INSERT ON ViagensNH.Localizacao TO 'professor';

GRANT INSERT ON ViagensNH.Cidade TO 'professor';

GRANT INSERT ON ViagensNH.Pais TO 'professor';

-- Permissões vistas, procedimentos, funções ao cargo
professor

GRANT SELECT ON ViagensNH.v_detalhes_viagem TO 'professor';

GRANT SELECT ON ViagensNH.v_top_alunos TO 'professor';

GRANT EXECUTE ON FUNCTION ViagensNH.f_total_vagens_aluno TO
'professor';

GRANT EXECUTE ON FUNCTION ViagensNH.f_taxa_sucesso_vagem TO
'professor';

GRANT EXECUTE ON PROCEDURE
ViagensNH.sp_perfil_preferencias_aluno TO 'professor';

GRANT EXECUTE ON PROCEDURE ViagensNH.sp_pesquisar_vagem TO
'professor';

-- REVOKE GRANT OPTION ON ViagensNH.* FROM 'professor';

-- Dar permissões ao role bibliotecario

GRANT SELECT ON ViagensNH.* TO 'bibliotecario';

GRANT DELETE ON ViagensNH.* TO 'bibliotecario';

GRANT INSERT,UPDATE ON ViagensNH.Aluno TO 'bibliotecario';

```

```

GRANT INSERT,UPDATE ON ViagensNH.Professor TO 'bibliotecario';

GRANT INSERT,UPDATE ON ViagensNH.Disciplina TO
'bibliotecario';

-- Permissões vistas, procedimentos, funções ao cargo aluno

GRANT SELECT ON ViagensNH.v_detalhes_viagem TO
'bibliotecario';

GRANT SELECT ON ViagensNH.v_top_alunos TO 'bibliotecario';

GRANT EXECUTE ON FUNCTION ViagensNH.f_total_vagens_aluno TO
'bibliotecario';

GRANT EXECUTE ON FUNCTION ViagensNH.f_taxa_sucesso_viagem TO
'bibliotecario';

GRANT EXECUTE ON PROCEDURE
ViagensNH.sp_perfil_preferencias_aluno TO 'bibliotecario';

GRANT EXECUTE ON PROCEDURE ViagensNH.sp_pesquisar_viagem TO
'bibliotecario';

-- REVOKE GRANT OPTION ON ViagensNH.* FROM 'bibliotecario';

-- Criação de users

CREATE USER IF NOT EXISTS 'sr_bruno'@'localhost' IDENTIFIED BY
'bibliotecario';

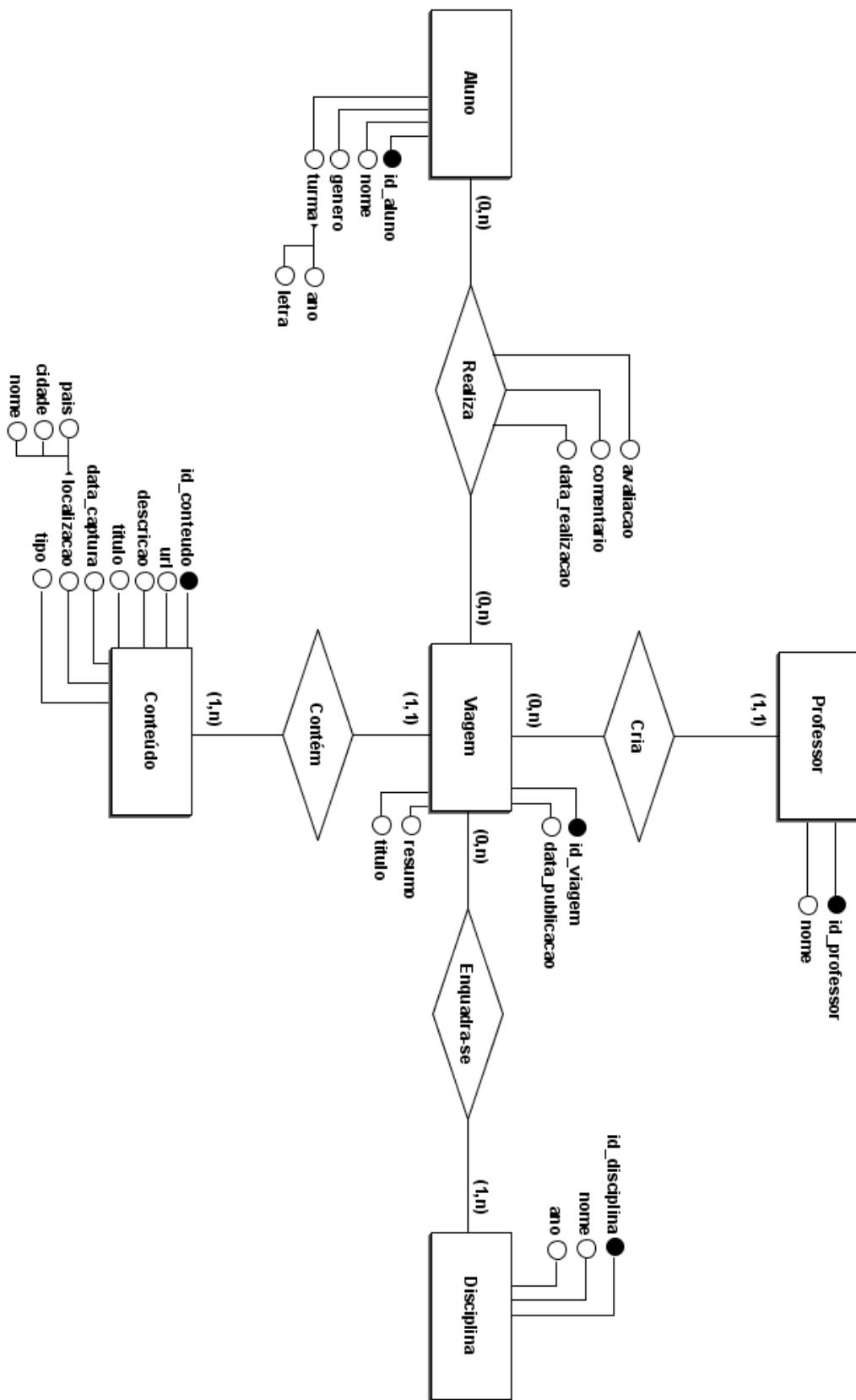
CREATE USER IF NOT EXISTS 'orlando_belo'@'localhost'
IDENTIFIED BY 'belo20';

CREATE USER IF NOT EXISTS 'francisco_luciano'@'localhost'
IDENTIFIED BY 'xico';

GRANT 'bibliotecario' TO 'sr_bruno'@'localhost';
GRANT 'professor' TO 'orlando_belo'@'localhost';
GRANT 'aluno' TO 'francisco_luciano'@'localhost';
SET DEFAULT ROLE ALL TO 'sr_bruno'@'localhost';
SET DEFAULT ROLE ALL TO 'orlando_belo'@'localhost';
SET DEFAULT ROLE ALL TO 'francisco_luciano'@'localhost';

```


VIII. Modelo Conceptual



IX. Modelo Lógico

