



Universidade do Minho
Escola de Engenharia
Licenciatura em Engenharia Informática

Unidade Curricular de Bases de Dados

Ano Letivo de 2023/2024

Agência de Detetives do Sr. Alfredo

João Daniel da Silva Carvalho	(a94015)
Luis Filipe Araújo Ferreira	(a98286)
Luís Miguel Teixeira Fernandes	(a88539)
Tiago Miguel Carvalho e Cunha	(a87978)

28 de maio de 2024

B D

Agência de Detetives do Sr. Alfredo

João Daniel da Silva Carvalho	(a94015)
Luis Filipe Araújo Ferreira	(a98286)
Luís Miguel Teixeira Fernandes	(a88539)
Tiago Miguel Carvalho e Cunha	(a87978)

28 de maio de 2024

Resumo

Serve o presente documento, elaborado no âmbito da Unidade Curricular de Bases de Dados, para retratar toda a elaboração e execução de uma arquitetura de uma base de dados referente a uma agência de detetives. Esta será usada para armazenamento de dados sobre casos da agência.

O projeto abordado e explicado neste documento é sobre a criação de uma base de dados de uma agência de detetives do Sr. Alfredo, que com o aumento exponencial de casos, e da sua complexidade, começou a ter problemas na organizações dos mesmos, bem como na gestão de informações da própria agência.

Começamos por contextualizar o problema e apresentar o caso de estudo, abordando as motivações e objetivos para a implementação da base de dados. Para uma continuação do projeto, foi analisado se esta seria viável e de seguida foram reunidos todos os recursos e uma equipa de trabalho para este mesmo projeto ser bem sucedido. Para dar como terminada esta fase inicial, procedemos a um levantamento de requisitos relativos ao projeto em causa.

Num fase seguinte, isto é, após o levantamento e tratamento dos requisitos, avançou-se para uma das fases mais importantes do projeto, a Modelação Conceptual. Aqui identificamos e caracterizamos as diferentes entidades necessárias, demonstrando como elas se relacionam, para assim estabelecer uma base de dados sólida, para que execute e cumpra com todos os requisitos e objetivos definidos anteriormente.

Após o Modelo Conceptual, avançou-se para a Modelação Lógica, onde construímos as tabelas que descrevem as diferentes entidades, e relacionamentos presentes no modelo conceptual, numa forma mais detalhada e aprofundada. Por fim, verificámos se o modelo se encontrava normalizado, garantindo assim a integridade e a não redundância dos dados que irão ser inseridos na base de dados.

Com a conclusão da modelação lógica, iniciou-se a parte da implementação física, onde é criada a base de dados modelada até ao momento dentro de um servidor *mysql*. Com a utilização da linguagem *SQL* criaram-se as *queries* responsáveis pela criação das tabelas definidas anteriormente, a criação dos utilizadores baseados nos requisitos de controlo, fez-se o povoamento da base de dados, e a implementação de *queries* que oferecem maneiras de manipular os dados dentro da base de dados consoante os requisitos de manipulação. Por fim, criam-se vários processos, e funções que poderão ser utilizados dentro de outras *queries*, gatilhos que serão acionados após a ocorrência de um dado evento e por últimos algumas vistas. No final calcula-se a evolução da base de dados ao longo do tempo a nível de espaço.

Este trabalho tem como objetivo a construção de um sistema de bases de dados que seja capaz de gerir de uma forma correta a informação relativa a casos pertencentes a uma agência de detetives, permitindo o bom funcionamento da mesma.

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

Palavras-Chave: Caso, Detetive, Cliente, Evidência, Fatura, Tipo, requisitos, entidades, relacionamentos, modelo conceptual, modelo lógico, normalização, modelo físico, MySQL Workbench, bases de dados, SQL Queries.

Índice

1	Definição de Sistema	1
1.1	Contextualização	1
1.2	Motivação e Objectivos	1
1.3	Análise da Viabilidade do processo	4
1.4	Recursos de Equipa de Trabalho	5
1.4.1	Recursos	5
1.4.2	Equipa de Trabalho	5
1.5	Plano de Execução do Projeto	6
1.6	Estrutura do Relatório	6
2	Levantamento e Análise de Requisitos	8
2.1	Método de Levantamento e de Análise de Requisitos Adotado	8
2.2	Organização dos Requisitos Levantados	9
2.2.1	Requisitos de Descrição	9
2.2.2	Requisitos de Manipulação	11
2.2.3	Requisitos de Controlo	13
2.3	Análise e Validação Geral dos Requisitos	14
3	Modelação Conceptual	15
3.1	Apresentação da Abordagem de Modelação Realizada	15
3.2	Apresentação e Explicação do Diagrama ER Produzido	16
3.2.1	ER: Cliente-Caso	17
3.2.2	ER: Caso-Fatura	18
3.2.3	ER: Cliente-Fatura	19
3.2.4	ER: Caso-Detetive	20
3.2.5	ER: Caso-Evidência	21
3.2.6	ER: Caso-Tipo	22
3.2.7	ER: Detetive-Evidência	23
3.2.8	ER: Detetive-Detetive	23
3.3	Identificação e Caracterização das Entidades	24
3.3.1	Entidade - Cliente	24
3.3.2	Entidade - Caso	24
3.3.3	Entidade - Fatura	25
3.3.4	Entidade - Detetive	25
3.3.5	Entidade - Evidência	26
3.3.6	Entidade - Tipo	26

3.4	Dicionário de dados das entidades do modelo	27
3.5	Identificação e Caracterização dos Relacionamentos	27
3.5.1	Relacionamento Cliente-Caso	27
3.5.2	Relacionamento Cliente-Fatura	28
3.5.3	Relacionamento Caso-Fatura	28
3.5.4	Relacionamento Caso-Detetive	29
3.5.5	Relacionamento Caso-Tipo	29
3.5.6	Relacionamento Caso-Evidência	30
3.5.7	Relacionamento Detetive-Evidência	30
3.5.8	Relacionamento Detetive-Detetive	31
3.6	Dicionário dos relacionamentos do modelo	31
3.7	Identificação e Caracterização dos Atributos das Entidades e dos Relacionamentos.	32
3.7.1	Entidade Cliente - Atributos	32
3.7.2	Entidade Fatura - Atributos	33
3.7.3	Entidade Caso - Atributos	34
3.7.4	Entidade Detetive - Atributos	35
3.7.5	Entidade Evidência - Atributos	36
3.7.6	Entidade Tipo - Atributos	37
3.8	Definição das Chaves das Entidades	38
4	Modelação Lógica	39
4.1	Construção e Validação do Modelo de Dados Lógico	39
4.2	Apresentação e Explicação do Modelo Lógico Produzido	40
4.2.1	Cliente	40
4.2.2	Caso	41
4.2.3	Fatura	42
4.2.4	Detetive	43
4.2.5	Evidência	44
4.2.6	Tipo	45
4.2.7	CasoDetetive	45
4.2.8	Modelo Lógico total	45
4.3	Normalização de Dados	46
4.4	Validação do Modelo com Interrogações do Utilizador	47
5	Implementação Física	50
5.1	Apresentação e explicação da base de dados implementada	50
5.1.1	Criação do esquema de bases de dados	51
5.1.2	Implementação da tabela: Clientes	51
5.1.3	Implementação da tabela: Cliente_Telemoveis	52
5.1.4	Implementação da tabela: Casos	53
5.1.5	Implementação da tabela: Tipos	54
5.1.6	Implementação da tabela: Faturas	55
5.1.7	Implementação da tabela: Detetives	56
5.1.8	Implementação da tabela: Caso Detetive	57

5.1.9	Implementação da tabela: Evidências	58
5.1.10	Implementação da tabela: Fotografias	59
5.2	Criação de utilizadores da base de dados	60
5.3	Povoamento da base de dados	63
5.4	Cálculo do espaço da base de dados (inicial e taxa de crescimento anual) . . .	65
5.4.1	Estimativa do crescimento anual	68
5.5	Definição e caracterização de vistas de utilização em SQL	70
5.6	Tradução das interrogações do utilizador para SQL	72
5.7	Indexação do Sistema de Dados	81
5.8	Procedimentos, funções e gatilhos	82
6	Conclusões e Trabalho Futuro	84

Lista de Figuras

1	Diagrama de GANTT	6
1	Diagrama ER	16
2	Entidade-Relacionamento: Cliente-Caso	17
3	Entidade-Relacionamento: Caso-Fatura	18
4	Entidade-Relacionamento: Cliente-Fatura	19
5	Entidade-Relacionamento: Caso-Detetive	20
6	Entidade-Relacionamento: Caso-Evidência	21
7	Entidade-Relacionamento: Caso-Tipo	22
8	Entidade-Relacionamento: Detetive-Evidência	23
9	Entidade-Relacionamento: Detetive-Detetive	23
10	Relacionamento - Cliente-Caso	27
11	Relacionamento - Cliente-Fatura	28
12	Relacionamento - Caso-Fatura	28
13	Relacionamento - Caso-Detetive	29
14	Relacionamento - Caso-Tipo	29
15	Relacionamento - Caso-Evidência	30
16	Relacionamento - Detetive-Evidência	30
17	Relacionamento - Detetive-Detetive	31
18	Atributos da Entidade "Cliente"	32
19	Atributos da Entidade "Fatura"	33
20	Atributos da Entidade "Caso"	34
21	Atributos da Entidade "Detetives"	35
22	Atributos da Entidade "Evidência"	36
23	Atributos da Entidade "Tipo"	37
24	Definição das chaves primárias	38
1	Tabela Cliente	40
2	Tabela Cliente	40
3	Entidade Caso	41
4	Tabela Caso	41
5	Relacionamento Cliente-Caso	41
6	Relacionamento Tipo-Caso	41
7	Entidade Fatura	42
8	Tabela Fatura	42
9	Relacionamento - Caso-Fatura	42

10	Entidade Detetive	43
11	Tabela Detetive	43
12	Entidade Evidencia	44
13	Tabela Evidencia	44
14	Relacionamento Caso-Evidencia	44
15	Relacionamento Detetive-Evidencia	44
16	Entidade Tipo	45
17	Tabela Tipo	45
18	Relacionamento Caso-Detetive	45
19	Tabela CasoDetetive	45
20	Modelo Lógico	46
21	Árvore Algebra	47
22	Arvore Algebra	47
23	Arvore Algebra	48
24	Arvore Algebra	49
1	Esquema - Agência Detetives	51
2	Implementação da tabela: Clientes	51
3	Atributo NIF único	52
4	Implementação da tabela: Cliente_Telemoveis	52
5	Implementação da tabela: Casos	53
6	Implementação da tabela: Tipos	54
7	Implementação da tabela: Faturas	55
8	Implementação da tabela: Detetives	56
9	Implementação da tabela: Caso Detetive	57
10	Implementação da tabela: Evidências	58
11	Implementação da tabela: Evidências	59
12	Criação de Utilizadores	60
13	Administrador com todas as permissões sobre o <i>schema</i>	60
14	RC1	60
15	RC2	60
16	RC3	61
17	RC4	61
18	RC5	61
19	RC6	61
20	RC7	61
21	RC8	62
22	RC9	62
23	RC10	62
24	Entradas na Tabela Detetive	63
25	Relatório Mensal	70
26	Lucro mensal	71
27	Casos - Número de detetives	71
28	Query 1 - RM1	72
29	Query 2 - RM2	72

30	Query 3 - RM3	73
31	Query 4 - RM4	73
32	Query 5 - RM6	73
33	Query 6 - RM7	74
34	Query 7 - RM8	74
35	Query 8 - RM10	75
36	Query 9 - RM11	75
37	Query 10 - RM12	76
38	Query 11 - RM13	76
39	Query 12 - RM14	77
40	Query 13 - RM15	77
41	Query 14 - RM16	78
42	Query 15 - RM17	78
43	Query 16 - RM18	79
44	Query 17 - RM19	79
45	Query 18 - RM20	80
46	Criação de indexação do sistema	81
47	Gatilho para Calcular Faturas	82
48	Calcular a idade de uma entrada	82
49	Adicionar um cliente com um telemóvel	83
1	Processo para RC2	86
2	Processo para RC6	86
3	Processo para RC9	87

Lista de Tabelas

2.1	Requisitos de Descrição	10
2.2	Requisitos de Manipulação	12
2.3	Requisitos de Controlo	14
5.1	Espaço ocupado no disco por cada tipo de dados	65
5.2	Espaço ocupado no disco por cada atributo num Cliente	65
5.3	Espaço ocupado no disco por cada atributo no ClienteTelemoveis	66
5.4	Espaço ocupado no disco por cada atributo num Tipo	66
5.5	Espaço ocupado no disco por cada atributo num Caso	66
5.6	Espaço ocupado no disco por cada atributo numa Faturas	67
5.7	Espaço ocupado no disco por cada atributo num Detetive	67
5.8	Espaço ocupado no disco por cada atributo no Caso Detetive	67
5.9	Espaço ocupado no disco por cada atributo numa Evidência	68
5.10	Espaço ocupado no disco por cada atributo numa Fotografia	68
5.11	Espaço ocupado no disco pela base de dados	68

1 Definição de Sistema

1.1 Contextualização

Alfredo José Oliveira Fernandes era um investigador veterano da policia judiciaria com mais de 20 anos de carreira. Um detetive com muitos casos resolvidos no seu percurso e, acima de tudo, um apoiante feroz do governo e da política portuguesa. Com o passar dos anos, o Sr.Alfredo começou a aperceber-se de um crescente número de casos de corrupção na política portuguesa e assim, movido pelos seu valores pessoais, resolveu lutar pelos seus ideias com as suas próximas mãos.

Sabendo que isso não seria possível mantendo-se na PJ, visto que esta pertencia ao estado e dessa forma inevitavelmente teria bloqueios nas suas investigações, decidiu então fundar a sua própria Agência privada, Integrity Guardians.

Começado de meios humildes, a agência inicialmente aceitou outros casos, para poder financiar as investigações maiores. Para estas arranjou os seus próprios detetives privados, o que levou a um crescimento tanto do número de casos a ser resolvidos como o número de informação necessária a ser armazenada. Devido a isto, ao fim de algum tempo foi-se notando a crescente necessidade de implementar um sistema de Bases de Dados que armazenaria todo o trabalho feito pela agência.

1.2 Motivação e Objectivos

O desenvolvimento deste caso de estudo surge da profunda preocupação do Sr. Alfredo com a integridade do sistema judicial e a crescente incidência de corrupção na política portuguesa. Ao longo da sua carreira na Polícia Judiciária, Alfredo testemunhou diretamente os efeitos negativos da corrupção, percebendo a sua corrosão dos alicerces da justiça e da sociedade. A sua dedicação aos seus valores pessoais e ao sistema judicial impeliu-o a agir perante esta realidade.

Perante esta situação, o Sr. Alfredo optou por tomar uma atitude pro-ativa. Reconhecendo a necessidade de mais autonomia e liberdade para conduzir investigações abrangentes e eficazes contra a corrupção política, ele decidiu fundar a sua própria agência privada de investigação. Ele viu a sua agência não apenas como um meio de conduzir investigações, mas como um instrumento para enfrentar a corrupção de maneira direta, com integridade e dedicação.

Inicialmente, Alfredo entendeu que precisaria garantir financiamento para as suas investigações e recursos para a sua agência. Por isso, aceitou uma variedade de casos, inclusive fora do âmbito da corrupção política, para assegurar o financiamento necessário. À medida que a sua agência crescia em reputação e capacidade, o foco na corrupção política tornava-se mais proeminente. A demanda crescente por investigações desse tipo, juntamente com o crescimento da equipa de detetives privados, levou à necessidade de um sistema de bases de dados para gerir e armazenar eficientemente as informações coletadas.

Objetivos da Implementação do Sistema de Bases de Dados:

Suporte aos Clientes: O sistema de bases de dados irá permitir um registo mais eficiente dos clientes. Estes poderão aceder às suas informações pessoais de forma mais rápida e terão acesso a todas as informações sobre os casos que solicitaram, acompanhando assim o estado do caso sem necessidade de contactar a agência ou deslocarem-se até lá.

Gestão de Lucros: Para cada caso solicitado, o cliente terá que pagar a fatura correspondente. Com um sistema de bases de dados eficiente, o Sr. Alfredo poderá acompanhar de forma mais precisa e detalhada o lucro obtido ao longo do tempo. O sistema irá gerir automaticamente relatórios financeiros, incluindo o registo de todas as faturas emitidas, pagamentos recebidos e despesas associadas a cada caso. Isso facilitará a gestão financeira da agência, fornecendo uma visão clara do desempenho financeiro, identificando tendências de receita e permitindo tomadas de decisão estratégicas para maximizar os lucros.

Suporte à Decisão: Este objetivo destaca a importância de fornecer dados e análises para ajudar na tomada de decisões durante investigações e operações policiais. Isso inclui o uso de informações do sistema para planejar estratégias de vigilância, abordagens de interrogatório e outras ações relacionadas à investigação.

Análise de Evidências: O sistema de bases de dados pode ser uma ferramenta valiosa para organizar e analisar evidências coletadas durante uma investigação. Isso facilita a reconstrução de eventos, a identificação de conexões entre diferentes peças de evidência e o desenvolvimento de uma linha de investigação sólida.

Gestão de Casos: A gestão eficaz de casos é essencial para garantir que as investigações sejam conduzidas de forma eficiente e eficaz. Um sistema de bases de dados pode ser utilizado para registar informações, atribuir tarefas, acompanhar o progresso e gerar relatórios sobre o estado de cada caso.

Centralização das Informações: Centralizar todas as informações relevantes sobre os casos investigados em um único sistema de bases de dados permite uma melhor coordenação e colaboração entre os detetives privados. Isso garante que todos os membros da equipa tenham acesso às mesmas informações atualizadas e facilita a comunicação e a partilha de conhecimento.

Segurança dos Dados: A segurança dos dados sensíveis e confidenciais é uma preocupação crítica para qualquer agência de investigação privada. A implementação de um sistema de

bases de dados proporciona maior segurança aos dados da agência, incluindo medidas para proteger contra acessos não autorizados e a possibilidade de fazer backups regulares para evitar perda de dados.

Aumento da Eficiência: Com todas as informações organizadas e facilmente acessíveis, os detetives privados podem trabalhar de forma mais eficiente, economizando tempo na busca por informações relevantes e na análise de dados. Isso permite que a agência responda de forma mais rápida e eficaz à demanda dos clientes e das investigações em curso.

1.3 Análise da Viabilidade do processo

A viabilidade do projeto de implementação de um sistema de bases de dados para a agência de investigação privada do Sr. Alfredo é alta e justificável por diversos motivos:

Melhoria da Eficiência Operacional: Ao centralizar todas as informações relacionadas aos casos num único sistema de bases de dados, a agência aumentará a sua eficiência operacional. Os detetives terão acesso rápido a informações relevantes, reduzindo o tempo gasto na busca por dados dispersos. Isso permite que a equipa dedique mais tempo à análise e resolução dos casos, aumentando a produtividade geral da agência.

Redução de Erros e Omissões: Com um sistema de bases de dados bem estruturado, há menos probabilidade de erros e omissões na gestão das informações dos casos. Por exemplo, a duplicação de dados é evitada, garantindo a integridade dos registos. Isso aumenta a confiabilidade das informações e melhora a qualidade das investigações conduzidas pela agência.

Facilitação da Colaboração: O sistema de bases de dados facilita a colaboração entre os detetives e outros membros da equipa. Por exemplo, múltiplos detetives podem trabalhar num mesmo caso simultaneamente, atualizando e compartilhando informações em tempo real. Isso promove uma cultura de colaboração e trabalho em equipa dentro da agência.

Suporte à Tomada de Decisão: Um sistema de bases de dados bem desenvolvido pode fornecer análises e relatórios detalhados sobre os casos em andamento. Essas informações são valiosas para apoiar a tomada de decisões estratégicas, como a alocação de recursos, priorização de casos e desenvolvimento de estratégias de investigação.

Redução de Custos a Longo Prazo: Embora haja custos associados à implementação inicial do sistema de bases de dados, os benefícios a longo prazo superam esses custos. Por exemplo, a redução do tempo gasto em tarefas administrativas e a melhoria da eficiência operacional podem resultar numa grande poupança de recursos financeiros e humanos a longo prazo.

Escalabilidade: Um sistema de bases de dados bem projetado é escalável e pode crescer conforme as necessidades da agência. À medida que a agência expande a sua equipa e aumenta o número de casos, o sistema de bases de dados pode ser facilmente adaptado para lidar com esse crescimento, garantindo que a agência possa continuar a operar de forma eficiente e eficaz.

1.4 Recursos de Equipa de Trabalho

1.4.1 Recursos

Recursos Humanos:

1. Clientes

Funções: Requisitam casos à empresa.

Recursos Materiais:

1. Servidores Instalados nos escritórios da empresa

Funções: Sistema de Gestão de Bases de Dados

1.4.2 Equipa de Trabalho

Pessoal Interno:

1. Alfredo José Oliveira Fernandes

Funções: Administração da empresa

2. Detetives

Funções: Receber e Resolver casos

Pessoal Externo

1. Engenheiros especializados em criar e manter a base de dados da empresa.

Funções: Levantamento de requisitos, modelação e implementação da base de dados, e a sua respetiva manutenção

1.5 Plano de Execução do Projeto

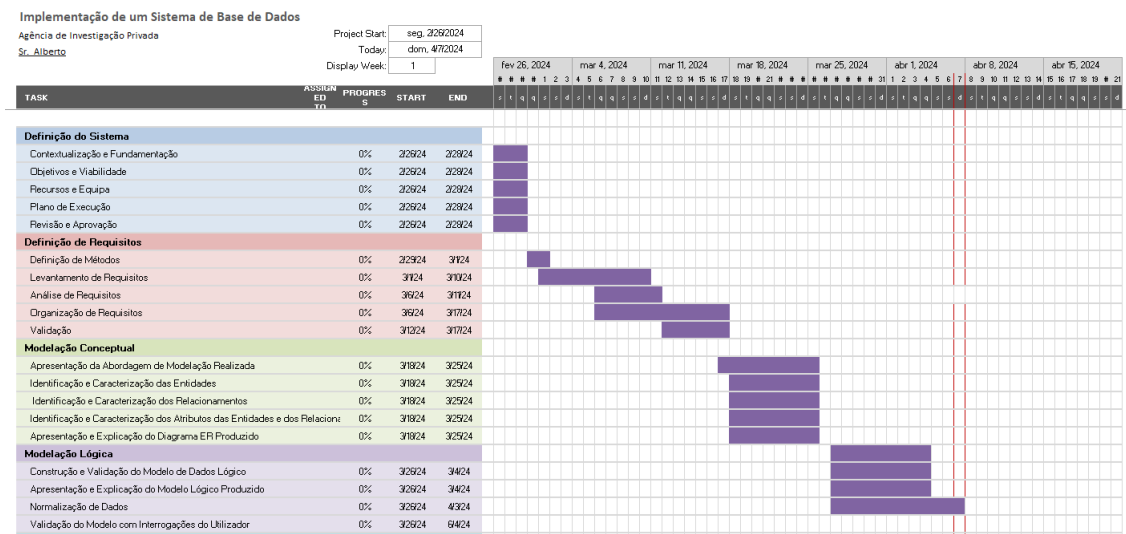


Tabela 1: Diagrama de GANTT

O Diagrama de GANTT ajuda a visualizar claramente as tarefas necessárias para realizar o trabalho de grupo, ao estabelecer uma sequência lógica de atividades, ajuda a atribuir responsabilidades a cada membro do grupo. Além disso, permite acompanhar o progresso do trabalho ao longo do tempo, identificando possíveis atrasos e ajustando-os no plano conforme necessário para cumprir os prazos. Apesar de não termos conseguido cumprir todos os prazos estabelecidos no Diagrama de GANTT, foi possível realizar o trabalho necessário para alcançar o "checkpoint". No entanto, conseguimos adaptar-nos às circunstâncias e priorizar as tarefas essenciais para garantir que o progresso do trabalho de grupo não fosse comprometido.

1.6 Estrutura do Relatório

Nesta secção, foi feita uma pequena descrição da estrutura que será abordada nos próximos capítulos deste relatório.

Capítulo 2 - Levantamento e Análise de Requisitos:

- Neste capítulo, iremos começar por explicar o método de como foram levantados os requisitos, em seguida iremos apresentar todos os tipos de requisitos levantados, por fim, neste capítulo iremos explicar como foi feita a análise e validação destes requisitos

Capítulo 3 - Modelo Conceptual:

- Neste capítulo, iremos começar por apresentar e explicar o nosso modelo construído, em seguida, é feita a identificação e caracterização de cada entidade e os seus relaciona-

mentos. Além desta explicação detalhada, vamos também identificar e caracterizar os atributos que cada entidade possui.

Capítulo 4 - Modelo Lógico:

- Neste capítulo, iremos realizar a construção e validação do modelo lógico. Inicialmente, iremos analisar os relacionamentos existentes para determinar as melhores tabelas a serem criadas. Para assegurar a validade do modelo lógico, iremos normalizar os dados. Depois, apresentaremos o modelo lógico e iremos validá-lo através das interrogações do utilizador.

Capítulo 5 - Modelo Físico:

- Neste capítulo, abordamos a implementação da nossa base de dados através do MySQL Workbench, começando pela tradução do modelo lógico para SQL. Foram configurados os utilizadores e definimos as permissões de acesso, e de seguida utilizamos scripts em Python ou diretamente em SQL para povoar a base de dados. Foram convertidas as interrogações do utilizador para SQL, calculámos o espaço ocupado pela base de dados e foi estimado o seu crescimento anual. Finalmente, implementamos vistas de utilização, procedimentos, funções e gatilhos para otimizar a gestão e segurança dos dados.

2 Levantamento e Análise de Requisitos

2.1 Método de Levantamento e de Análise de Requisitos Adotado

Na agência de investigação privada liderada pelo Sr. Alfredo, a identificação precisa dos requisitos é crucial para o sucesso das investigações realizadas pela equipa. Com o intuito de melhorar os métodos de levantamento de requisitos, o Sr. Alfredo solicitou a ajuda do experiente analista Jorge Jesus para desenvolver estratégias eficazes que assegurem uma compreensão abrangente das necessidades dos clientes e dos objetivos das investigações.

Jorge Jesus, com a sua vasta experiência em análise de requisitos em diferentes contextos, iniciou um processo colaborativo com a equipa da agência. Optaram por uma abordagem multifacetada, integrando uma variedade de técnicas de levantamento de requisitos para capturar uma diversidade de perspetivas e 'insights'.

O primeiro passo foi identificar as partes interessadas envolvidas nas investigações, incluindo os gerentes da agência, os investigadores e, naturalmente, os clientes. Jorge conduziu entrevistas detalhadas com os clientes para compreender as suas preocupações, objetivos e requisitos específicos. Estas entrevistas forneceram uma base sólida para compreender as necessidades dos clientes e estabelecer os objetivos das investigações.

Além disso, Jorge realizou uma análise minuciosa da documentação fornecida pelos clientes, como relatórios policiais, registos financeiros e comunicações relevantes. Esta análise ajudou a identificar padrões, tendências e informações cruciais que informaram o desenvolvimento dos requisitos.

Reuniões de 'brainstorming' foram realizadas com a equipa da agência para explorar ideias, discutir abordagens de investigação e identificar potenciais desafios e soluções. Diagramas de atividade foram elaborados para visualizar o fluxo de trabalho das investigações, desde a recolha de dados até à entrega dos resultados.

Para complementar estas abordagens, foram enviados inquéritos padronizados aos clientes para recolher feedback adicional sobre os serviços da agência e as suas experiências anteriores. A análise dos resultados dos inquéritos forneceu 'insights' valiosos que ajudaram a refinar ainda

mais os requisitos levantados. Jorge e a equipa da agência trabalharam em estreita colaboração para validar os requisitos com os clientes, garantindo que todas as partes estivessem alinhadas quanto aos objetivos da investigação e às expectativas do cliente.

No final do processo, todos os requisitos foram documentados de forma clara e concisa num documento formal, fornecendo uma base sólida para orientar as investigações futuras da agência. A abordagem colaborativa com Jorge Jesus permitiu que a agência desenvolvesse estratégias robustas de levantamento de requisitos, garantindo que as investigações sejam conduzidas de forma eficaz e eficiente, atendendo às necessidades e expectativas dos clientes.

2.2 Organização dos Requisitos Levantados

Depois de uma análise detalhada dos requisitos levantados para a agência de investigação privada, procedemos à cuidadosa organização de todas as informações obtidas. Este processo envolveu a compilação e seleção dos requisitos considerados mais relevantes para o desenvolvimento da base de dados, com o objetivo de criar uma estrutura sólida que satisfizesse as necessidades da agência do Sr. Alfredo.

2.2.1 Requisitos de Descrição

Os requisitos de descrição são essenciais para garantir a precisão, consistência e compreensão dos dados em uma base de dados, facilitando a sua manutenção e conformidade com padrões, estando estes associados à criação de objetos na base de dados, para isso temos na tabela 2.1 os seguintes requisitos recolhidos:

Nº	Data e Hora	Descrição	Fonte	Analista
RD1	25/03/2024 17:00	Cada cliente da agência deve ser registado com o seu número de identificação.	Chefe	Luís Ferreira
RD2	25/03/2024 17:10	Um cliente deve possuir um identificador, um nome, um email, um ou mais telemóveis, género, data de nascimento, morada composta por código postal, rua e localidade, nif e terá associado a si uma lista dos casos que este solicitou, bem como uma lista das faturas pagas.	Chefe	Luís Ferreira
RD3	25/03/2024 17:40	Cada fatura deve incluir um número identificativo, o valor pago pelo caso, uma data de emissão e a data de pagamento.	Chefe	João Daniel

Nº	Data e Hora	Descrição	Fonte	Analista
RD4	25/03/2024 18:25	Um caso deve possuir um identificador único, um estado dizendo se o caso se encontra em execução, suspenso ou terminado, uma designação, data de início, data de conclusão, o custo que este caso teve para a agência, o caso deve ter associado a si o seu tipo de caso, bem como a fatura originada pelo caso, uma lista de detetives que participam neste caso, bem como uma lista das evidências encontradas.	Chefe	Jorge Jesus
RD5	25/03/2024 19:50	Um detetive deve possuir um identificador único, um nome, formas de contacto sendo estas telemóvel e email, um estado dizendo se o detetive se encontra no ativo ou não e por fim o seu cargo, o detetive também deve ter associado a si uma lista com as evidências que este encontrou, bem como uma lista dos casos a que este está associado.	Chefe	Tiago Cunha
RD6	25/03/2024 20:30	Uma evidência deve possuir um identificador, uma designação, uma descrição, data de coleta e uma fotografia opcional.	Chefe	João Daniel
RD7	25/03/2024 21:00	Cada tipo de caso deve ter associado, um número identificativo, uma designação e a descrição.	Chefe	Luís Ferreira

Tabela 2.1: Requisitos de Descrição

2.2.2 Requisitos de Manipulação

A base de dados da agência de investigação privada deve ser capaz de realizar operações de povoamento e exploração de dados de forma eficiente. Isso inclui a execução de consultas para extrair informações específicas, bem como a implementação de procedimentos, funções e gatilhos para automatizar tarefas e garantir a integridade dos dados.

Por exemplo, poderia ser necessário criar uma função para calcular o número de casos por detetive em determinado período, ou um gatilho para atualizar automaticamente o estado de um caso quando uma evidência é recolhida. Essas capacidades são essenciais para garantir que a base de dados seja utilizada de forma eficaz e que forneça informações úteis para as operações da agência, para isso temos na tabela 2.2 os seguintes requisitos de manipulação recolhidos:

Nº	Data e Hora	Descrição	Fonte	Analista
RM1	26/03/2024 09:00	A qualquer momento, o sistema deve apresentar um relatório de faturas mensais, com o número de casos analisados, o número de detetives envolvidos e o valor total faturado.	Chefe	João Daniel
RM2	26/03/2024 09:10	A qualquer momento, o sistema deve listar todos os casos solicitados por um cliente.	Chefe	Jorge Jesus
RM3	26/03/2024 09:20	Deve ser possível saber quantos casos houve num mês.	Chefe	Luís Ferreira
RM4	26/03/2024 09:30	A qualquer momento, o sistema deve listar todos os clientes da agência, ordenados pelo o seu número de identificação fiscal.	Chefe	Tiago Cunha
RM5	26/03/2024 09:40	A qualquer momento deve ser possível consultar, adicionar, editar e eliminar informações de um cliente.	Chefe	João Daniel
RM6	26/03/2024 09:50	Deve ser possível consultar todos os detetives envolvidos num caso.	Chefe	Jorge Jesus
RM7	26/03/2024 10:00	Deve ser possível saber quantos casos um detetive esteve envolvido num mês.	Chefe	Luís Fernandes
RM8	26/03/2024 11:00	Deve ser possível consultar as evidências recolhidas por um detetive.	Chefe	João Daniel
RM9	26/03/2024 11:10	A qualquer momento deve ser possível consultar, adicionar, editar e eliminar informações de um detetive.	Chefe	Luís Ferreira
RM10	26/03/2024 11:20	Deve ser possível saber quantas evidências um caso teve.	Chefe	Luís Fernandes

Nº	Data e Hora	Descrição	Fonte	Analista
RM11	26/03/2024 11:30	A qualquer momento, o sistema deve listar todos os casos da agência, ordenados pelo o seu tipo.	Chefe	Luís Ferreira
RM12	26/03/2024 11:40	Deve ser possível consultar o histórico de casos de um detetive.	Chefe	Tiago Cunha
RM13	26/03/2024 11:50	A qualquer momento, o sistema deve listar todas as evidências, ordenadas pela data de coleta	Chefe	João Daniel
RM14	26/03/2024 12:00	Deve ser possível saber os casos que envolveram mais detetives.	Chefe	Luís Ferreira
RM15	26/03/2024 12:10	A qualquer momento, o sistema deve listar todos os detetives pelo o seu estado.	Chefe	Luís Fernandes
RM16	26/03/2024 12:20	A qualquer momento, o sistema deve listar todos os casos ordenados por estado (em execução, suspenso ou terminado)	Chefe	Tiago Cunha
RM17	26/03/2024 12:30	A qualquer momento, o sistema deve listar todas as faturas já emitidas	Chefe	João Daniel
RM18	26/03/2024 12:40	A qualquer momento, o sistema deve listar todas as faturas pagas	Chefe	João Daniel
RM19	26/03/2024 12:50	A qualquer momento, o sistema deve ser capaz de calcular o lucro obtido num mês	Chefe	Jorge Jesus
RM20	26/03/2024 13:00	A qualquer momento, o sistema deve listar todas as faturas por pagar	Chefe	Luís Fernandes

Tabela 2.2: Requisitos de Manipulação

2.2.3 Requisitos de Controlo

É crucial que a base de dados da agência de investigação privada seja capaz de controlar o acesso dos utilizadores de acordo com os seus perfis e permissões, isso implica definir várias formas de monitorizar e restringir o acesso aos dados, garantindo que cada utilizador apenas tenha acesso às informações pertinentes para o seu papel na organização.

Por exemplo, um detetive pode ter permissão para visualizar os seus casos e evidências, enquanto um administrador pode ter acesso total a todas as funcionalidades da base de dados. É fundamental implementar mecanismos de gestão de utilizadores que permitam atribuir e revogar permissões de forma granular, garantindo assim a segurança e integridade dos dados da agência. Para isso, temos na tabela 2.3 os seguintes requisitos de controlo organizados:

Nº	Data e Hora	Descrição	Fonte	Analista
RC1	26/03/2024 15:00	Só o administrador pode gerar as faturas.	Chefe	Luís Ferreira
RC2	26/03/2024 15:10	Cada cliente tem apenas acesso aos dados dos seus casos solicitados.	Chefe	Tiago Cunha
RC3	26/03/2024 15:20	Só o administrador pode criar/editar/eliminar as informações de um caso.	Chefe	Luís Fernandes
RC4	26/03/2024 15:30	Só o administrador pode criar/consultar/editar/eliminar as informações de qualquer cliente. Cada cliente pode realizar o mesmo, mas apenas das suas informações.	Chefe	Jorge Jesus
RC5	26/03/2024 15:40	Só o administrador e os detetives podem consultar as informações dos clientes.	Chefe	Luís Fernandes
RC6	26/03/2024 15:50	Cada cliente tem apenas acesso aos seus dados pessoais.	Chefe	Luís Ferreira
RC7	26/03/2024 16:00	Só o administrador e os detetives podem consultar as informações de qualquer caso.	Chefe	Luís Fernandes
RC8	26/03/2024 16:10	Só os detetives podem registar/consultar/editar evidências de um caso.	Chefe	Tiago Cunha
RC9	26/03/2024 17:00	Só o administrador pode criar/consultar/editar/eliminar as informações de qualquer detetive. Cada detetive pode realizar o mesmo, mas apenas das suas informações.	Chefe	João Daniel
RC10	26/03/2024 17:10	Só o administrador e os detetives podem consultar o histórico de casos solicitados por qualquer cliente.	Chefe	Luís Fernandes

Nº	Data e Hora	Descrição	Fonte	Analista
RC11	26/03/2024 17:30	Cada cliente pode consultar o seu histórico de casos solicitados.	Chefe	João Daniel

Tabela 2.3: Requisitos de Controlo

2.3 Análise e Validação Geral dos Requisitos

O levantamento de requisitos é um processo extremamente importante, permitindo que haja uma noção da totalidade de informação contida na base de dados. Como tal, os nossos requisitos devem estar de acordo com aquilo que torne mais ágil e simples a resolução de um caso, tanto para o lado do cliente como para o detetive.

Para fazer o levantamento, foi marcada uma reunião pelo Sr.Alfredo com os detetives para perceber a ótica de todos sobre os requisitos, onde foi pedido que dessem a sua opinião tanto como detetives e clientes. No final do levantamento, o Sr.Alfredo e os seus detetives voltaram a rever os requisitos um por um para chegarem a um consenso. Quando este foi alcançado, o Sr.Alfredo acabou a sua reunião com os detetives.

Mais tarde reuniu-se com os Engenheiros para lhes apresentar os requisitos para estes poderem proceder a fase de Modelação conceptual.

3 Modelação Conceptual

3.1 Apresentação da Abordagem de Modelação Realizada

Após a minuciosa etapa de levantamento, análise e validação dos requisitos, iniciou-se a fase de construção do modelo conceptual para o desenvolvimento da base de dados. Nesta etapa, adotou-se a elaboração de um diagrama entidade-relacionamento (ER), um modelo de dados de alto nível amplamente utilizado no projeto de bases de dados conceptual e lógico.

A modelação conceptual é uma fase crucial no desenvolvimento de sistemas de base de dados, pois proporciona uma compreensão detalhada do domínio do sistema, identificando entidades, atributos e relacionamentos. A partir das informações levantadas e validadas, foi possível visualizar como as entidades estão relacionadas entre si, garantindo que o modelo desenvolvido atenda às necessidades e expectativas do cliente e dos utilizadores finais.

O processo de construção do diagrama ER iniciou-se com a identificação das entidades participantes e dos seus atributos, seguido pelo estabelecimento dos relacionamentos entre elas. Essa abordagem proporcionou uma representação visual clara das relações entre as entidades, facilitando a compreensão do sistema por parte de todos os envolvidos no projeto.

Assim, a construção do modelo conceptual baseou-se numa análise criteriosa dos requisitos levantados, garantindo que o sistema de base de dados projetado seja capaz de cumprir todos os requisitos e necessidades do Sr. Alfredo.

3.2 Apresentação e Explicação do Diagrama ER Produzido

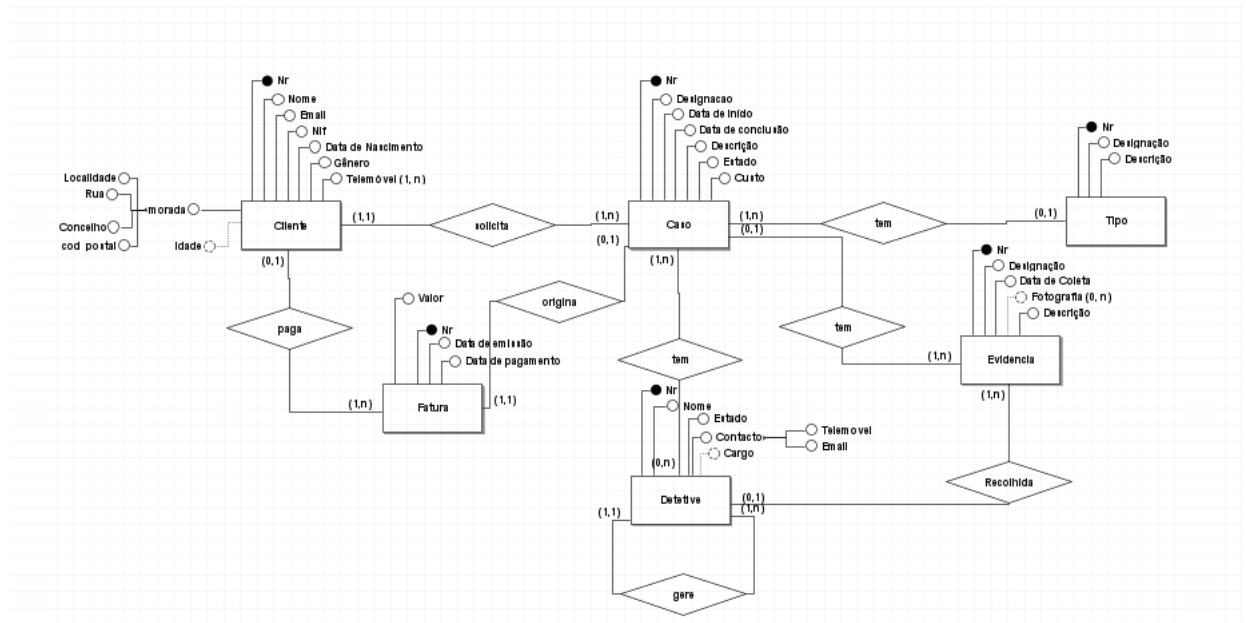


Tabela 1: Diagrama ER

O processo de construção do modelo ER é fundamental para representar de forma clara e organizada as entidades e relacionamentos identificados anteriormente, juntamente com seus atributos. Começamos por identificar as principais entidades envolvidas no sistema, cada uma dessas entidades é representada por retângulos no diagrama ER. Em seguida, atribuímos os atributos relevantes a cada entidade. Após caracterizar as entidades com seus atributos, identificamos os relacionamentos entre elas, esses relacionamentos são representados por linhas conectando as entidades no diagrama ER, com a cardinalidade indicando a natureza do relacionamento.

3.2.1 ER: Cliente-Caso

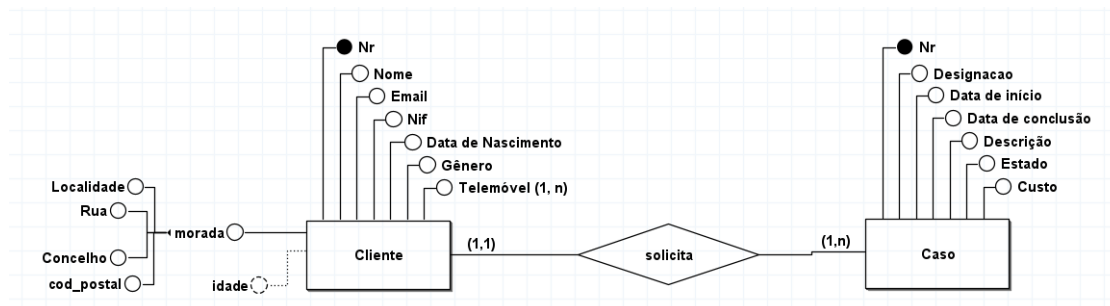


Tabela 2: Entidade-Relacionamento: Cliente-Caso

Na entidade "Cliente", temos os atributos, "Nr", "Nome", "Email", "Nif", "Data de nascimento", "Gênero", que são atributos simples, o atributo "idade" também é simples mas opcional, já o atributo "Telemovel" é um atributo multivalorado, uma vez que o cliente pode possuir mais que um número de contacto telefónico. Também contém o atributo composto "Morada", que contém "localidade", "Rua", "Concelho" e "cod postal". Esta entidade permite o registo de clientes na agência, podendo assim também saber quantos clientes estão registados na agência.

Na entidade "Caso", temos os atributos, "Nr", "Designação", "Data de início", "Data de conclusão", "Descrição", "Estado" e "Custo", sendo todos estes atributos simples. Esta entidade permite o registo de casos na agência, podendo assim também saber quantos casos estão registados na agência.

Após a caracterização destas entidades com os seus atributos, identificamos os relacionamentos entre elas, por exemplo, um cliente pode solicitar um ou vários casos (1,N), mas um caso só pode ser solicitado por um e um só cliente (1,1). Através desta relação irá ser possível obter informação sobre os casos que o cliente já solicitou.

Os atributos e relacionamentos destas entidades tiveram origem nos seguintes requisitos de descrição: RD1 e RD4.

3.2.2 ER: Caso-Fatura

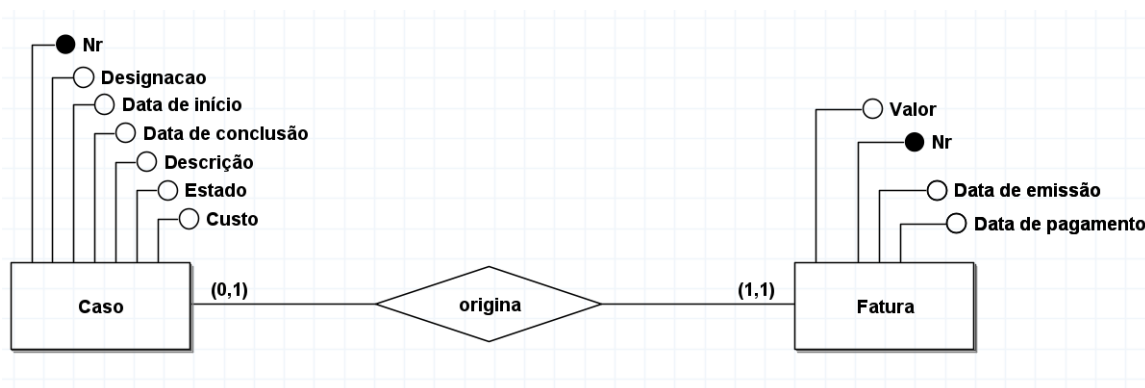


Tabela 3: Entidade-Relacionamento: Caso-Fatura

Como a entidade "Caso" já foi explicada, vamos observar agora a entidade "Fatura", esta tem como atributos, "Nr", "Data de emissão", "Data de pagamento", "Valor", atributos estes que são simples. Esta entidade permite a existência de faturas no sistema, podendo também saber as faturas que já foram pagas e as que estão por pagar.

Após a caracterização desta entidade com os seus atributos, identificamos os relacionamentos entre a entidade "Caso" e "Fatura", por exemplo, cada caso origina a sua fatura, enquanto que uma fatura pode não ser originada de um caso ou originada de um só caso. Através desta relação é possível obter informação sobre os casos que originaram faturas registadas no sistema.

Os atributos e relacionamentos destas entidades tiveram origem nos seguintes requisitos de descrição: RD3 e RD4.

3.2.3 ER: Cliente-Fatura

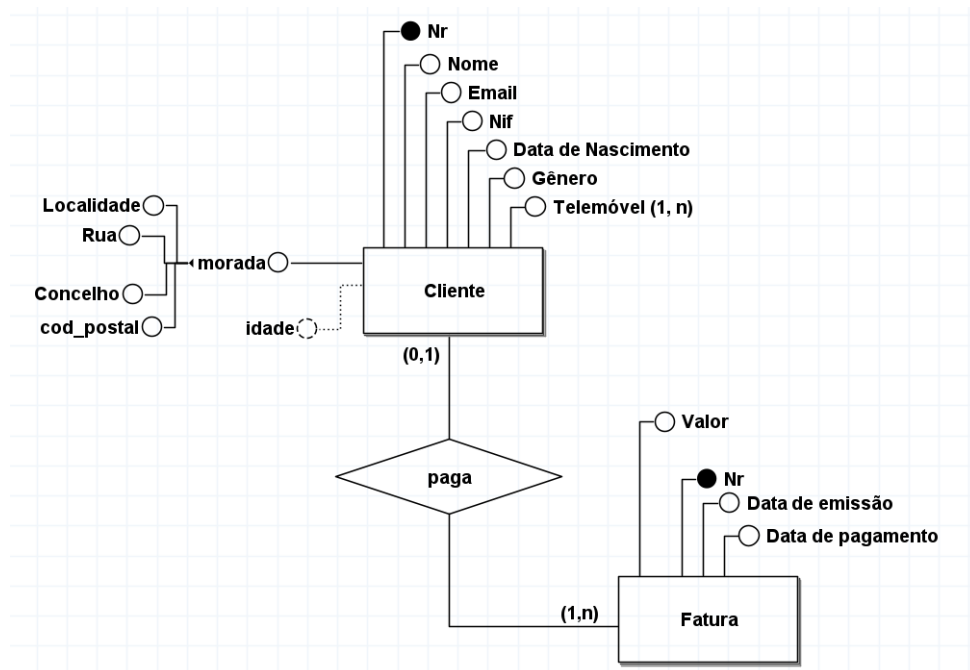


Tabela 4: Entidade-Relacionamento: Cliente-Fatura

Como ambas as entidades já foram explicadas anteriormente, vamos focar na identificação dos relacionamentos entre estas entidades, por exemplo, um cliente pode pagar uma ou várias faturas, por ter solicitado um só caso ou vários casos, mas uma fatura pode não ser paga por um cliente ou paga por um cliente.

Os atributos e relacionamentos destas entidades tiveram origem nos seguintes requisitos de descrição: RD1 e RD3.

3.2.4 ER: Caso-Detetive

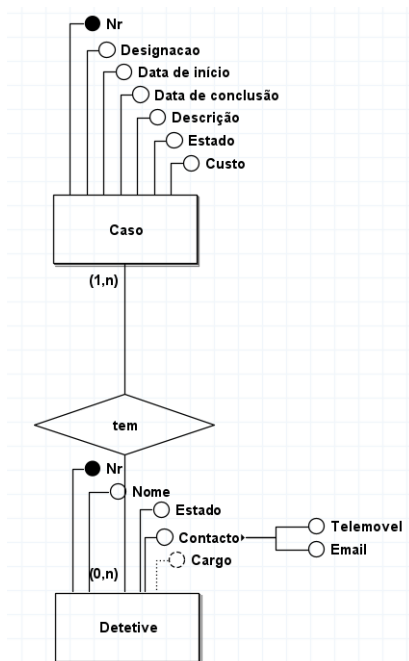


Tabela 5: Entidade-Relacionamento: Caso-Detetive

Como a entidade caso já foi falada, vamos falar da entidade "Detetive". Esta tem como atributos simples "Nr", "Nome" e "Estado". O atributo "cargo" é simples mas opcional. Além disso também possui o atributo composto "Contacto", que contém "Telemóvel" e "email". Esta entidade permite o registo de detetives, e também saber quantos detetives existem na agência.

Os relacionamentos entre as entidades "Caso" e "Detetive" são tais que um detetive pode ter um ou vários casos (1,N), mas um caso pode ter zero ou mais detetives (0,n).

Os atributos e relacionamentos destas entidades tiveram origem nos seguintes requisitos de descrição: RD4 e RD5.

3.2.5 ER: Caso-Evidência

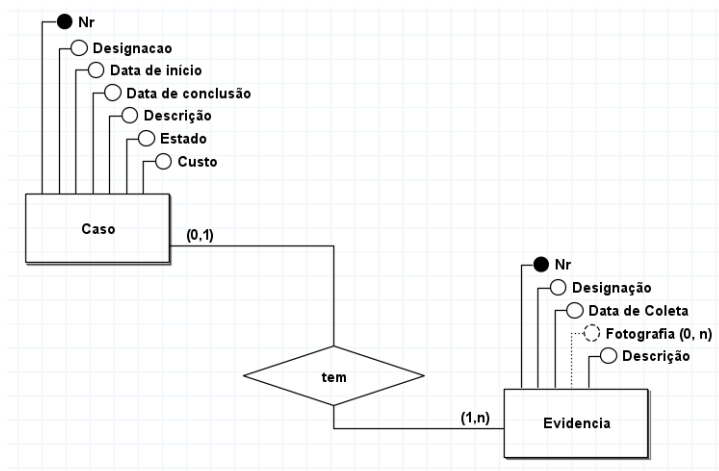


Tabela 6: Entidade-Relacionamento: Caso-Evidência

Falemos agora da entidade "Evidência". Esta contém 4 atributos simples: "Nr", "Designação", "Data de coleta" e "Descrição". Também possui um atributo multivalorado opcional em "Fotografia", uma vez que uma evidência pode não ter foto ou até pode ter várias. Esta entidade permite o registo de evidências, e também saber quantas evidências há.

O relacionamento entre "Caso" e "Evidência" vem de uma evidência poder não estar ou estar associada a um caso (0,1), e um caso ter uma ou várias evidências (1,N).

Os atributos e relacionamentos destas entidades tiveram origem nos seguintes requisitos de descrição: RD4 e RD6.

3.2.6 ER: Caso-Tipo

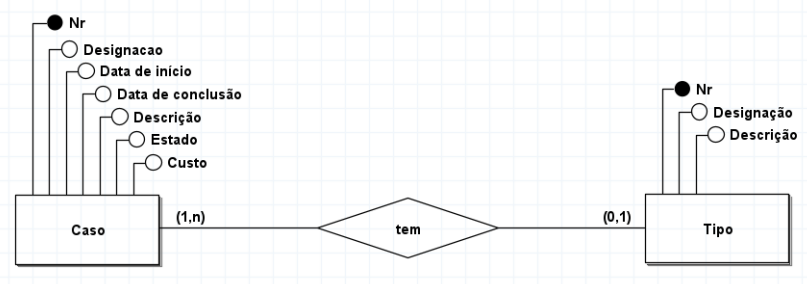


Tabela 7: Entidade-Relacionamento: Caso-Tipo

A entidade "Tipo" possui três atributos simples, sendo estes "Nr", "Designação" e "Descrição". Esta entidade permite que os casos sejam agrupados por "Tipo" e também para poder saber quantos casos com um certo tipo existem.

O relacionamento "Caso" e "Tipo" vem de um caso ter um tipo ou não (0,1), e um certo tipo ter estar associado a um ou mais casos (1,n).

Os atributos e relacionamentos destas entidades tiveram origem nos seguintes requisitos de descrição: RD4 e RD7.

3.2.7 ER: Detetive-Evidência

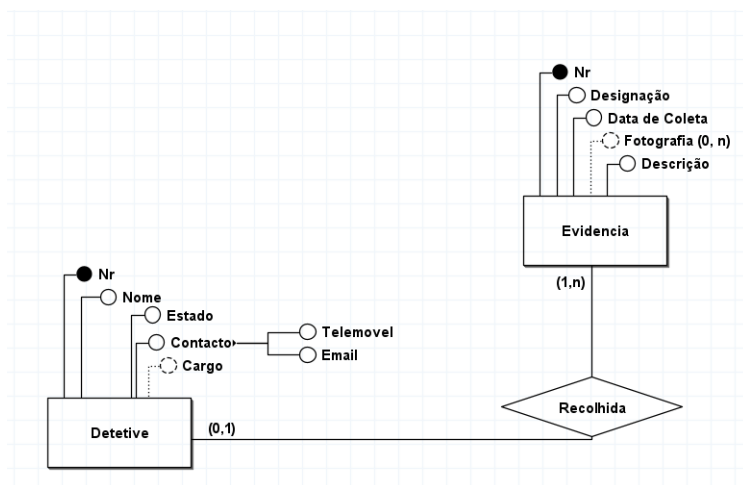


Tabela 8: Entidade-Relacionamento: Detetive-Evidência

O relacionamento "Detetive" e "Evidência" vem de um detetive recolher uma ou mais evidências, e uma evidência poder ter sido recolhida por um detetive ou não. Este relacionamento tem origem do requisito de descrição RD5 e RD6.

3.2.8 ER: Detetive-Detetive

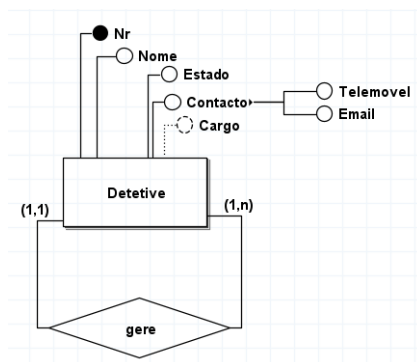


Tabela 9: Entidade-Relacionamento: Detetive-Detetive

Este relacionamento vem de um detetive ser gerido por um e só por um detetive seu superior, mas um detetive pode gerir 1 ou mais detetives de cargo mais baixo. Este relacionamento tem origem do requisito de descrição RD5.

3.3 Identificação e Caracterização das Entidades

Analisando os requisitos definidos foram identificadas as seguintes entidades:

3.3.1 Entidade - Cliente

Cliente refere-se a uma pessoa que utiliza os serviços da agência do Sr. Alfredo. Esta entidade possui os seguintes atributos:

- "Nr": Número de identificação único constituído por 6 dígitos numéricos.
- "Nome": Nome completo do cliente.
- "Email": Endereço eletrónico do cliente.
- "NIF": Número de identificação fiscal do cliente.
- "Data de Nascimento": Data de nascimento do cliente.
- "Género": representa o género do cliente.
- "Telémovei": representa o contacto telefónico do cliente, trata-se de um atributo multivalorado, uma vez que, um cliente pode ter um ou mais contactos.
- "Idade": representa a idade do cliente, trata-se de um atributo derivado, pois queremos calcular a idade do cliente.

Os atributos desta relação tiveram origem no requisito de descrição RD1 e RD2.

3.3.2 Entidade - Caso

Caso refere-se a um caso que esteja a ser investigado pela agência. Esta entidade possui os seguintes atributos:

- "Nr": número de identificação único constituído por 6 dígitos numéricos.
- "Designação": trata-se de um código do caso a ser investigado.
- "Data de Início": data de quando é que o caso foi iniciado.
- "Data de Conclusão": data de quando é que o caso foi concluído
- "Descrição": é uma descrição do que é que se trata o caso.
- "Estado": representa o estado atual do caso.

- "Custo": representa o custo que agência teve para um caso solicitado.

Os atributos desta relação tiveram origem no requisito de descrição RD4.

3.3.3 Entidade - Fatura

Fatura é um comprovativo do serviço prestado pela agência. Esta entidade possui os seguintes atributos:

- "Nr": número de identificação único constituído por 6 dígitos numéricos.
- "Valor": trata-se do valor a ser pago pelo serviço prestado pela agência.
- "Data de emissão": data de quando a fatura foi emitida.
- "Data de pagamento": data de quando a fatura foi paga pelo cliente.

Os atributos desta relação tiveram origem no requisito de descrição RD3.

3.3.4 Entidade - Detetive

Detetive refere-se a uma pessoa que desempenha funções de investigação na agência do Sr. Alfredo. Esta entidade possui os seguintes atributos:

- "Nr": Número de identificação único constituído por 6 dígitos numéricos.
- "Nome": Nome completo do detetive da agência.
- "Estado": indica se o detetive encontra-se ativo ou inativo.
- "Contacto": é um atributo composto por:
 - "Telemóvel": representa o contacto telefónico do detetive.
 - "Email": endereço eletrónico do detetive.
- "Cargo": trata-se da hierarquia da agência entre detetives (atributo opcional).

Os atributos desta relação tiveram origem no requisito de descrição RD5.

3.3.5 Entidade - Evidência

Evidência refere-se a qualquer tipo de informação, objeto, documento, ou outro elemento relevante que forneça indícios ou prova relacionada ao caso em questão. Esta entidade possui os seguintes atributos:

- "Nr": Número de identificação único constituído por 6 dígitos numéricos.
- "Designação": trata-se da categoria ou tipo específico de evidência que está sendo identificada.
- "Data de Coleta": data de quando a evidência foi recolhida.
- "Fotografia": prova visual da evidência em questão.
- "Descrição": descrição detalhada da evidência encontrada.

Os atributos desta relação tiveram origem no requisito de descrição RD6.

3.3.6 Entidade - Tipo

Tipo refere-se à categoria ou classificação específica do caso que está a ser investigado, por exemplo, o tipo de caso pode ser um roubo, homicídio, etc. Esta entidade possui os seguintes atributos:

- "Nr": Número de identificação único constituído por 6 dígitos numéricos.
- "Designação": trata-se do tipo do caso a ser investigado.
- "Descrição": é uma descrição detalhada do tipo de caso.

Os atributos desta relação tiveram origem no requisito de descrição RD7.

3.4 Dicionário de dados das entidades do modelo

Entidade	Descrição	Sinónimos	Onde ocorre
Cliente	Pessoa que solicita o serviço de investigação.	Comprador	Um Cliente solicita um Caso
Caso	Investigação solicitada por um cliente, relativa a um crime ou então a uma curiosidade do cliente	Investigação	Um Caso vai ser solicitado para investigação por um Cliente
Fatura	Documento que discrimina uma transação.	Conta	Uma fatura gerada para o pagamento do Caso
Detetive	Pessoa responsável por realizar a investigação de um crime.	Investigador	O Detetive trata da investigação do Caso
Evidencia	Algo usado para comprovar o Caso em questão.	Prova	Prova encontrada aquando da resolução do Caso
Tipo	Género atribuído a um determinado Caso.	Género	Cada caso ira ter um Tipo associado

3.5 Identificação e Caracterização dos Relacionamentos

Nesta modelação foram criados alguns relacionamentos entre entidades, de modo a identificá-los e caracterizá-los foi feita uma análise e explicação de cada relacionamento:

3.5.1 Relacionamento Cliente-Caso

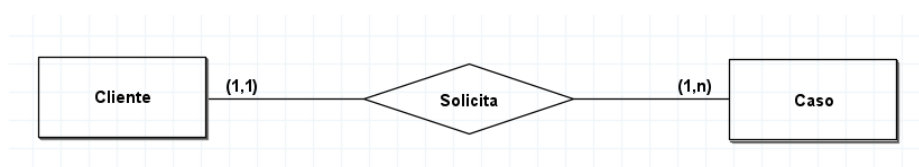


Tabela 10: Relacionamento - Cliente-Caso

Relacionamento: Cliente solicita Caso.

Descrição: Foi estipulada a relação entre o cliente e o caso para saber quantos casos é que o cliente já solicitou.

Multiplicidade: Cliente (1,1)- Caso (1,N)

Um cliente pode solicitar um ou mais casos, mas um caso só pode ser solicitado por um e um só cliente.

Requisito: Este relacionamento foi originado através do requisito de descrição RD2.

3.5.2 Relacionamento Cliente-Fatura

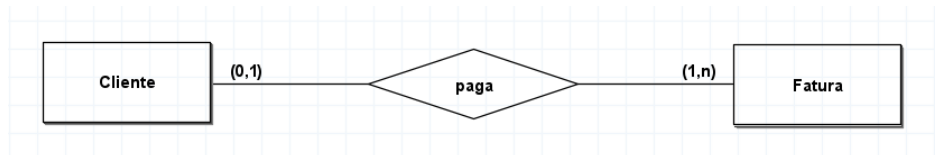


Tabela 11: Relacionamento - Cliente-Fatura

Relacionamento: Cliente paga Fatura

Descrição: Foi estipulada a relação entre o cliente e a fatura para saber as faturas que já foram emitidas e pagas pelo cliente.

Multiplicidade: Cliente (0,1) - Fatura (1,N)

Um cliente pode pagar varias faturas associadas a casos solicitados. Uma fatura pode ser não ser paga ou paga por um cliente.

Requisito: Este relacionamento foi originado através do requisito de descrição RD2.

3.5.3 Relacionamento Caso-Fatura

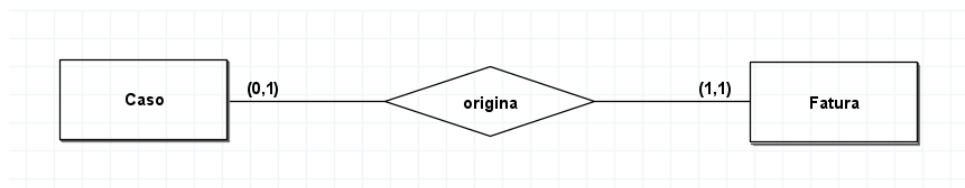


Tabela 12: Relacionamento - Caso-Fatura

Relacionamento: Caso origina Fatura

Descrição: Foi estipulada a relação entre caso e fatura para saber que casos é que originaram uma fatura.

Multiplicidade: Caso (0,1) - Fatura (1,1)

Um caso originar uma e apenas uma fatura. Uma fatura é originada por nenhum ou apenas um caso.

Requisito: Este relacionamento foi originado através dos requisitos RD7 e RD9.

3.5.4 Relacionamento Caso-Detetive

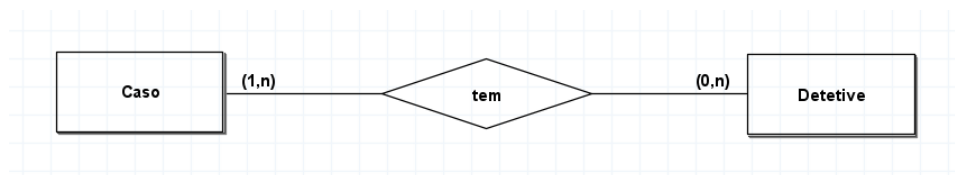


Tabela 13: Relacionamento - Caso-Detetive

Relacionamento: Caso tem Detetive

Descrição: Foi estipulada a relação entre caso e detetive para obter informação sobre quais casos um certo detetive está envolvido e sobre que detetives é que estão num caso.

Multiplicidade: Caso $(1,N)$ - Detetive $(0,N)$

Um caso pode ter nenhum ou vários detetives envolvidos. Um detetive pode estar em um ou mais casos.

Requisito: Este relacionamento foi originado através do requisito de descrição RD4.

3.5.5 Relacionamento Caso-Tipo

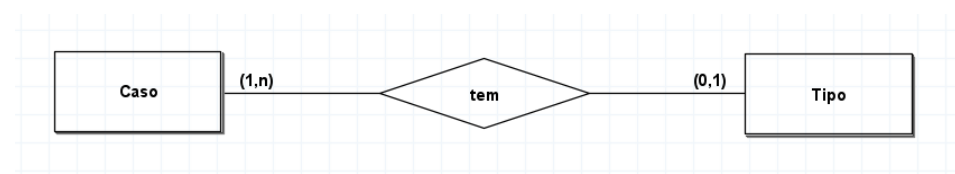


Tabela 14: Relacionamento - Caso-Tipo

Relacionamento: Caso tem Tipo

Descrição: Foi estipulada a relação entre caso e tipo para poder organizar os vários tipos de casos, permitindo identificar que casos é que tem um certo tipo.

Multiplicidade: Caso $(1,N)$ - Tipo $(0,1)$

Um caso pode não ter um tipo ou ter um e só um tipo. Um tipo pode estar num caso ou em vários casos.

Requisito: Este relacionamento foi originado através do requisito de descrição RD4.

3.5.6 Relacionamento Caso-Evidência

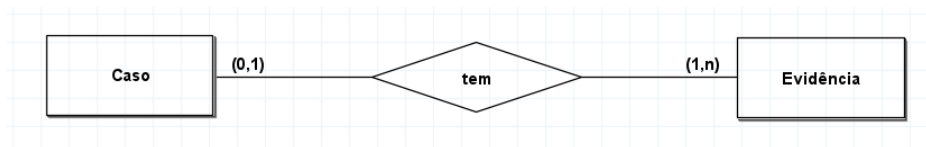


Tabela 15: Relacionamento - Caso-Evidência

Relacionamento: Caso tem Evidência

Descrição: Foi estipulada a relação entre caso e evidência para saber que evidências um caso tem.

Multiplicidade: Caso $(0,1)$ - Evidência $(1,N)$

Um caso tem uma evidência ou varias evidências. Uma evidência está em nenhum caso ou em um só caso.

Requisito: Este relacionamento foi originado através do requisito de descrição RD4.

3.5.7 Relacionamento Detetive-Evidência

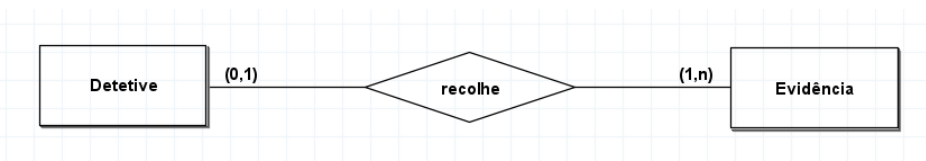


Tabela 16: Relacionamento - Detetive-Evidência

Relacionamento: Detetive recolhe Evidência

Descrição: Foi estipulada a relação entre detetive e evidência para obter informação sobre quantas evidências é que um detetive recolheu.

Multiplicidade: Detetive $(0,1)$ - Evidência $(1,N)$

Um detetive pode recolher uma várias evidências. Uma evidência pode não ser recolhida por um detetive ou recolhida por um e apenas um detetive.

Requisito: Este relacionamento foi originado através do requisito de descrição RD5.

3.5.8 Relacionamento Detetive-Detetive

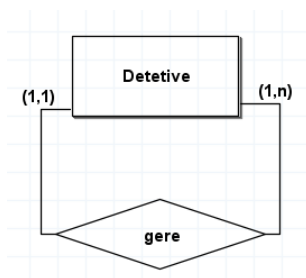


Tabela 17: Relacionamento - Detetive-Detetive

Relacionamento: Detetive gere Detetive

Descrição: Foi estipulada a relação recursiva entre detetives para poder gerir os casos para cada detetive, uma vez que existe um cargo associado ao detetive.

Multiplicidade: Detetive (1,1) - Detetive (1,N)

Um detetive gere um ou varios detetives. Um detetive é gerido por um e apenas um detetive

Requisito: Este relacionamento foi originado através do requisito de descrição RD5.

3.6 Dicionário dos relacionamentos do modelo

Entidade	Multiplicidade	Relacionamento	Multiplicidade	Entidade
Cliente	(1,1)	Solicita	(1,N)	Caso
Cliente	(0,1)	Paga	(1,N)	Fatura
Caso	(0,1)	Origina	(1,1)	Fatura
Caso	(1,N)	Tem	(0,N)	Detetive
Caso	(0,1)	Tem	(1,N)	Evidência
Caso	(1,N)	Tem	(0,1)	Tipo
Detetive	(0,1)	Recolhe	(1,N)	Evidência
Detetive	(1,1)	Gere	(1,N)	Detetive

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

3.7.1 Entidade Cliente - Atributos

Atributo	Descrição	Tipo	NULL	Composto	Multivalorado	Derivado	Exemplo
ID	Número único identificativo (Chave Primária)	INT	X	X	X	X	1
Nome	Nome completo do Cliente	VARCHAR (100)	X	X	X	X	José Carlos da Silva Duarte
Gênero	Gênero do Cliente	VARCHAR (45)	✓	X	X	X	Masculino
Data de Nascimento	Data de nascimento do Cliente	DATE	✓	X	X	X	18-04-1989
Endereço:	Informações sobre o endereço:	-----	✓	✓	X	X	-----
-Localidade;	Localidade;	VARCHAR(250)	✓	X	X	X	Braga
-Cocelho;	Concelho;	VARCHAR (100)	✓	X	X	X	Barcelos
-Morada;	Morada;	VARCHAR (100)	✓	X	X	X	Rua das Barrocas
Código-Postal;	Código Postal;	VARCHAR (9)	✓	X	X	X	4750-290
Endereço Eletrónico	Endereço Eletrónico do Cliente	VARCHAR(100)	X	X	X	X	jose_carlos@gmail.com
NIF	Número de identificação fiscal	INT	X	X	X	X	213456789
Telemóvel	Lista de telemóveis que o cliente possui	VARCHAR(15)	X	X	✓	X	912345987
Idade	Idade do Cliente	INT	✓	X	X	✓	35

Tabela 18: Atributos da Entidade "Cliente"

Os atributos da entidade "Cliente" foram originados através do requisito de descrição RD1 e RD2.

3.7.2 Entidade Fatura - Atributos

Atributo	Descrição	Tipo	NULL	Composto	Multivalorado	Derivado	Exemplo
ID	Número único identificativo da Fatura (Chave Primária)	INT	X	X	X	X	1
Valor	Valor da fatura a pagar.	INT	X	X	X	X	50,00
Data de emissão	Data de emissão da fatura.	DATETIME	X	X	X	X	16-03-2024
Data de pagamento	Data de pagamento da fatura.	DATETIME	✓	X	X	X	27-03-2024

Tabela 19: Atributos da Entidade "Fatura"

Os atributos da entidade "Fatura" foram originados através do requisito de descrição RD3.

3.7.3 Entidade Caso - Atributos

Atributo	Descrição	Tipo	NULL	Composto	Multivalorado	Derivado	Exemplo
ID	Número único identificativo (Chave Primária)	INT	X	X	X	X	1
Estado	Estado do Caso (Em execução, Congelado, Terminado)	VARCHAR (45)	X	X	X	X	Em execução
Designação	Designação do Caso	VARCHAR (100)	X	X	X	X	INV-2024-001
Descrição	Descrição do Caso	TEXT	X	X	X	X	O caso INV-2024-001 envolve a investigação de um suposto desvio de fundos em uma empresa de tecnologia de médio porte chamada Tech Innovate.
Data de Início	Data de quando é que o Caso foi iniciado	DATE	X	X	X	X	01-01-2024
Data de Conclusão	Data de quando é que o Caso foi terminado	DATE	✓	X	X	X	27-03-2024
Custo	Custo que a agência teve para este caso	INT	X	X	X	X	300,00

Tabela 20: Atributos da Entidade "Caso"

Os atributos da entidade "Caso" foram originados através do requisito de descrição RD4.

3.7.4 Entidade Detetive - Atributos

Atributo	Descrição	Tipo	NULL	Composto	Multivalorado	Derivado	Exemplo
ID	Número único identificativo (Chave Primária)	INT	X	X	X	X	1
Nome	Nome completo do Detetive	VARCHAR(200)	X	X	X	X	Mario Costa Araújo Ferreira
Contacto: Telemóvel; Email;	Informações sobre o contacto do Detetive: Número telefónico do Detetive; Endereço Eletrónico do Detetive;	----- INT VARCHAR(100)	----- X X	----- X X	----- X X	----- X X	----- 932167894 mario.agencia.priv@ agencia.com
Estado	Estado do Detetive (Ativo/Inativo)	VARCHAR(45)	X	X	X	X	Ativo
Cargo	Cargo do Detetive (Chefe/Detetive)	INT	✓	X	X	X	Detetive

Tabela 21: Atributos da Entidade "Detetives"

Os atributos da entidade "Detetive" foram originados através do requisito de descrição RD5.

3.7.5 Entidade Evidência - Atributos


Atributo	Descrição	Tipo	NULL	Composto	Multivalorado	Derivado	Exemplo
ID	Número único identificativo (Chave Primária)	INT	X	X	X	X	1
Designação	Designação da evidência	VARCHAR(75)	X	X	X	X	Impressão Digital
Descrição	Descrição da evidência recolhida	VARCHAR(300)	X	X	X	X	Impressão digital numas chaves, encontrada na rua garcia da orta.
Data de Coleta	Data de quando foi recolhida a evidência	DATE	X	X	X	X	25-03-2024
Fotografia	Fotografia/as associadas à evidência	IMAGE	✓	X	✓	X	

Tabela 22: Atributos da Entidade "Evidência"

Os atributos da entidade "Evidência" foram originados através do requisito de descrição RD6.

3.7.6 Entidade Tipo - Atributos

Atributo	Descrição	Tipo	NULL	Composto	Multivalorado	Derivado	Exemplo
ID	Número único identificativo (Chave Primária)	INT	X	X	X	X	1
Designação	Designação do tipo de caso	VARCHAR(45)	X	X	X	X	Fraude Fiscal
Descrição	Descrição do tipo de caso	VARCHAR(200)	X	X	X	X	Este tipo de caso envolve a investigação de atividades fraudulentas relacionadas a finanças, como fraude bancária, falsificação de documentos, lavagem de dinheiro, ou desvio de fundos corporativos.

Tabela 23: Atributos da Entidade "Tipo"

Os atributos da entidade "Tipo" foram originados através do requisito de descrição RD7.

3.8 Definição das Chaves das Entidades

Entidade	Chave Primária	Motivo
Cliente	idCliente (ID)	Número único identificativo
Fatura	idFatura (ID)	Número único identificativo.
Caso	idCaso (ID)	Número único identificativo.
Detetive	idDetetive (ID)	Número único identificativo.
Evidência	idEvidencia (ID)	Número único identificativo.
Tipo	idTipo (ID)	Número único identificativo

Tabela 24: Definição das chaves primárias

4 Modelação Lógica

4.1 Construção e Validação do Modelo de Dados Lógico

Por forma a construirmos um bom modelo lógico que possuisse o menor número possível de erros, optámos por cuidadosamente inspecionar o modelo conceptual que havíamos desenvolvido e, de seguida, aplicar um conjunto de regras por forma a converter um modelo noutro. Inicialmente, optámos por verificar quantas tabelas o nosso modelo lógico possuiria, para isso, fomos analisar o modelo conceptual, procurando por entidades, relações N para N, e atributos multivalorados. Após esta análise, concluímos que o nosso modelo possuiria 9 tabelas. Após esta análise, passámos para a ferramenta de modelação que optámos por usar, o MySQL Workbench, e, de seguida, começámos a fazer a caracterização de cada uma das tabelas, analisando nesta fase de caracterização cada um dos atributos que as nossas tabelas iriam possuir, baseando-nos no modelo lógico. Após a caracterização estar completa, começámos a olhar para os relacionamentos que possuíamos no modelo conceptual. Estes relacionamentos no modelo conceptual vão passar a ser relações entre tabelas, sendo que cada relacionamento 1 para N dá origem a uma relação 1 para N. Para esta relação ser definida de forma correta, temos de adicionar numa das tabelas, neste caso do lado dos "muitos", uma chave estrangeira que nos remete para a tabela com a qual estamos a estabelecer a relação. De seguida, reparamos que possuíamos uma relação de 1 para 1; estas relações são bastante peculiares, pois tanto podem dar origem a uma única tabela, no caso de as duas entidades possuírem uma relação total, ou duas tabelas, no caso de uma delas não ser estritamente necessária na relação. No nosso caso concreto, tratava-se do segundo caso, pelo que a nossa relação 1 para 1 originou duas tabelas. De seguida, tivemos de determinar quem seria o pai e o filho desta relação, por forma a colocarmos a chave estrangeira no local correto. Usando estes métodos de conversão, asseguramos que as relações entre tabelas são bem definidas e mantemos a integridade referencial.

4.2 Apresentação e Explicação do Modelo Lógico Produzido

Como referido anteriormente o modelo lógico foi construído baseado em algumas regras de conversão de um modelo conceptual para um lógico, de seguida iremos explicar em pormenor cada uma das conversões que foram feitas.

4.2.1 Cliente

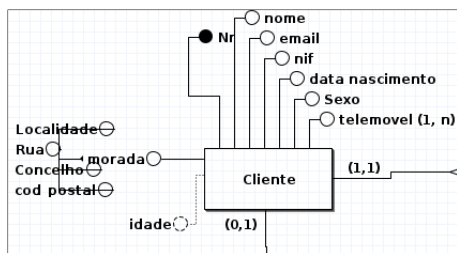


Tabela 1: Tabela Cliente

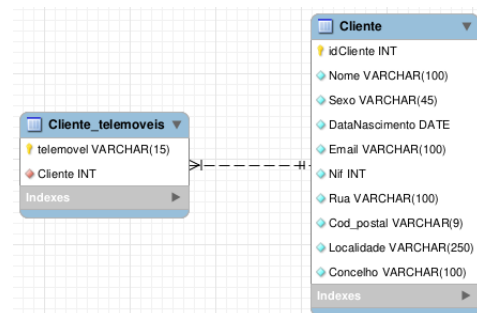


Tabela 2: Tabela Cliente

Transformações do Modelo Conceptual para o Modelo Lógico:

- Criação da Tabela Cliente e atribuir chave Primária
- Identificação de Atributos Multivalorados
- Tratamento do Atributo Multivalorado (Criação da Tabela Cliente_telemoveis)
- Definição das Chaves na Tabela Cliente_telemoveis]
- Conversão e caracterização de Atributos Simples da Entidade Cliente

Começando pela entidade Cliente, podemos ver na Tabela 1 a entidade que havíamos idealizado no modelo conceptual. De seguida, vemos a conversão desta entidade para uma tabela no modelo lógico. A conversão desta entidade foi feita seguindo as seguintes regras: Primeiramente, criamos uma tabela Cliente no nosso modelo, de seguida, verificamos se a nossa entidade possuía ou não atributos multivalorados. Como podemos ver na Tabela 1, temos um atributo multivalorado que, neste caso, seria 'telemovel'. Este atributo origina outra tabela, tabela esta que nos diz que cada cliente poderá possuir um ou mais telemóveis. Após a criação desta tabela, optamos, por simplicidade, por tratar logo dos atributos da mesma, que no nosso caso serão uma chave primária e uma chave estrangeira. A chave primária será o próprio número de telemóvel, visto que queremos que cada número seja único e pertença a apenas um cliente. A chave estrangeira será o número do cliente, por forma a conseguirmos relacionar o Cliente com o seu número de telemóvel. Após a criação desta tabela, começamos

a definir as entradas na tabela Cliente. Como a nossa entidade agora apenas possuía uma chave e atributos simples, esta conversão foi relativamente simples, passando por ser apenas converter cada atributo para uma entrada da tabela, indicando as suas características.

4.2.2 Caso

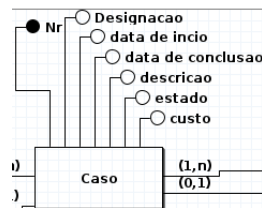


Tabela 3: Entidade Caso



Tabela 4: Tabela Caso



Tabela 5: Relacionamento Cliente-Caso

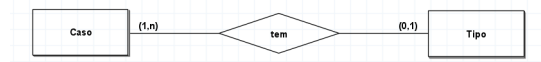


Tabela 6: Relacionamento Tipo-Caso

Transformações do Modelo Conceptual para o Modelo Lógico:

- Criação da Tabela Caso e atribuir Chave Primária
- Conversão e caracterização de Atributos Simples da Entidade Caso
- Identificação de Relacionamentos 1 para N
- Criação de Atributos para Chaves Estrangeiras
- Ligar Chaves Estrangeiras

No caso da entidade Caso, podemos ver a partir das Tabelas 5 e 6 que a conversão desta não será tão simples, devido ao facto de esta ter dois relacionamentos a chegar a si de 1 para N, o que fará com que esta Tabela possua duas chaves estrangeiras a referenciar outras duas tabelas. Mas analisando uma coisa de cada vez, primeiramente iremos começar por analisar a entidade Caso em si, e olhando para ela podemos desde logo ver que esta apenas possui uma chave primária e atributos simples, o que nesta fase nos simplifica a conversão, sendo esta apenas criar a Tabela Caso e fazer a conversão respetiva de cada atributo para uma linha da tabela, introduzindo também todas as características destes atributos. De seguida, como referido anteriormente, iremos ter de olhar para os relacionamentos 1 para N que chegam a esta entidade. No nosso caso, esta possui relacionamentos 1 para N com outras duas entidades,

Tipo e Cliente. Como já referi anteriormente, estes relacionamentos irão dar origem a chaves estrangeiras na tabela Caso, mas de momento apenas iremos criar um atributo com o nome Cliente e outro com o nome Tipo. Estes atributos irão ter o mesmo tipo das chaves das tabelas Cliente e Evidência, e posteriormente, quando tivermos todas as tabelas geradas, iremos fazer a ligação deste atributo ao atributo id nas tabelas respetivas, originando assim uma chave estrangeira.

4.2.3 Fatura

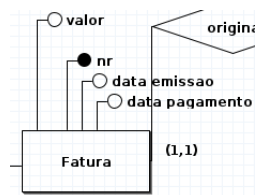


Tabela 7: Entidade Fatura

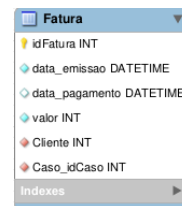


Tabela 8: Tabela Fatura

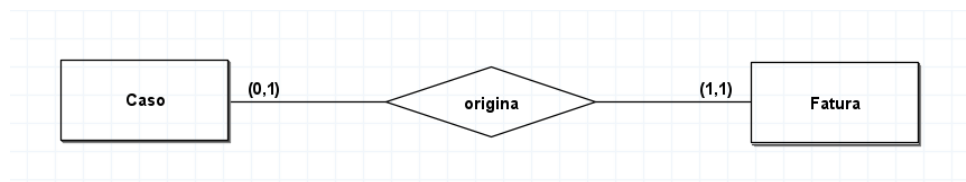


Tabela 9: Relacionamento - Caso-Fatura

Transformações do Modelo Conceptual para o Modelo Lógico:

- Verificação de Atributos Multivalorados
- Criação da Tabela e atribuir Chave Primária
- Conversão e caracterização de Atributos Simples
- Gestão do Relacionamento 1 para N com Cliente
- Análise do Relacionamento 1 para 1
- Determinação e Inclusão da Chave Estrangeira do Relacionamento 1 para 1

No caso da Fatura, tal como no caso anterior, vemos através da tabela 9 que esta possui a chegar a si um relacionamento 1 para N com o cliente, então essa parte da conversão será bastante similar à referida anteriormente. Mas, mais uma vez começando pelo mais básico, iremos verificar se esta entidade possui atributos multivalorados, o que não se verifica, e então esta entidade será bastante fácil de criar, tendo apenas uma chave primária e atributos simples. Após caracterizarmos estes atributos, iremos olhar mais uma vez para os relacionamentos que

esta relação possui, e vemos que, para além do relacionamento que chega a si de 1 para N, que irá originar uma chave estrangeira de Cliente, vemos também que esta está envolvida num relacionamento 1 para 1. Ao olharmos para estes relacionamentos, temos de estudar caso a caso, vendo se o relacionamento é total por parte das duas entidades ou se alguma das entidades é opcional. No nosso caso em particular, encontramos-nos perante um caso do segundo tipo, pelo que teremos de analisar quem será o pai e o filho da relação para vermos onde colocar de forma correta uma chave estrangeira. Neste caso, o pai será a entidade Fatura, logo a tabela desta entidade irá possuir também uma chave estrangeira para a Tabela Caso.

4.2.4 Detetive

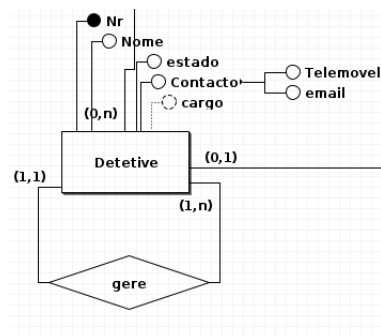


Tabela 10: Entidade Detetive

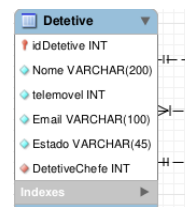


Tabela 11: Tabela Detetive

Transformações do Modelo Conceptual para o Modelo Lógico:

- Criação da Tabela Detetive e atribuir Chave Primária
- Conversão e caracterização de Atributos Simples
- Tratamento de Atributo Composto
- Análise do relacionamento Recursivo
- Definir Chave Estrangeira para Relacionamento Recursivo

A conversão da entidade Detetive para a Tabela Detetive possui algumas particularidades, começando pelo facto de esta entidade possuir um atributo composto, atributo este que na prática irá desdobrar-se em dois atributos simples aquando a conversão para a Tabela. Mas o mais interessante é que ainda não tínhamos visto antes é o facto de esta entidade possuir um relacionamento recursivo, por forma a determinar o Detetive que gere outros detetives. A conversão desta entidade para uma tabela será, na sua generalidade, bastante simples, no entanto, irá ter a particularidade de possuir uma chave estrangeira que irá apontar para a própria Tabela Detetive.

4.2.5 Evidência

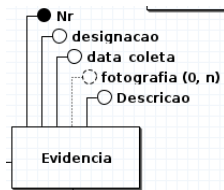


Tabela 12: Entidade Evidencia

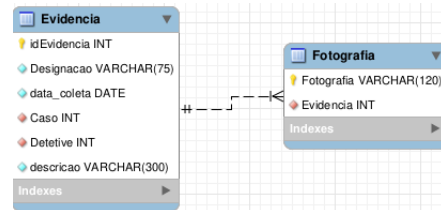


Tabela 13: Tabela Evidencia

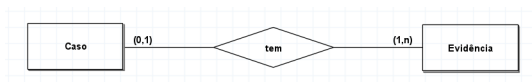


Tabela 14: Relacionamento
Evidencia

Caso-

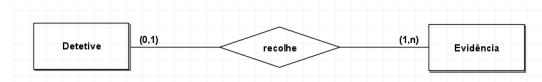


Tabela 15: Relacionamento
Evidencia

Detetive-

Transformações do Modelo Conceptual para o Modelo Lógico:

- Criação da Tabela Evidência e atribuir Chave Primária
- Tratamento do Relacionamentos 1 para N
- Analise do Atributo Multivalorado
- Criação e Caracterização da Tabela Fotografia
- Conversão e caracterização de Atributos Simples da Entidade Evidência

Na conversão da entidade Evidência para a Tabela Evidência, iremos ter, como anteriormente, a chegada a esta entidade de dois relacionamentos 1 para N. Como já vimos anteriormente, quando uma entidade tem a chegar a si estes relacionamentos, a tabela da entidade em si irá ter a adição de chaves estrangeiras que nos remetem para a tabela da qual a relação provém. Irá ter também um atributo multivalorado, o que dará origem a uma nova tabela, neste caso, a tabela Fotografia. Esta tabela irá possuir uma chave primária que corresponderá ao path único da imagem e uma chave estrangeira que nos remete para a Evidência à qual estas fotos pertencem. Em termos de conversão, esta era a parte mais complicada ao converter esta entidade, visto que, de resto, a entidade apenas possui uma chave primária e atributos simples que, como já vimos, são de fácil conversão, tendo nós apenas de ter cuidado em atribuir as características corretas a cada atributo.

4.2.6 Tipo

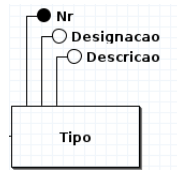


Tabela 16: Entidade Tipo



Tabela 17: Tabela Tipo

Transformações do Modelo Conceptual para o Modelo Lógico:

- Criação da Tabela Tipo e atribuir Chave Primária
- Conversão e caracterização de Atributos Simples

A conversão da entidade Tipo para a Tabela Tipo será das mais simples, visto que esta entidade apenas possui uma chave primária e dois atributos simples. Sendo assim, apenas será necessária criar a tabela e fazer a caracterização destes atributos, estando assim a conversão feita.

4.2.7 CasoDetetive

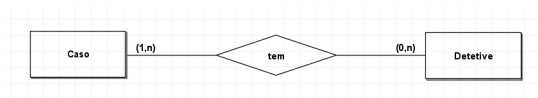


Tabela 18: Relacionamento Caso-Detetive

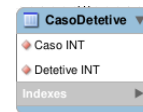


Tabela 19: Tabela CasoDetetive

Transformações do Modelo Conceptual para o Modelo Lógico:

- Criação da Tabela do Relacionamento N para N entre Caso e Detetive
- Criação de duas Chaves Estrangeiras para fazerem a ligação entre as duas tabelas.

Esta tabela é bastante particular, pois reflete o relacionamento entre duas entidades. Para esta conversão, usamos a regra de que um relacionamento N para N origina uma tabela. Por forma a esta tabela estar bem construída, esta deve possuir duas chaves estrangeiras. No nosso caso, iremos possuir uma chave estrangeira para a tabela Caso e outra chave estrangeira para a tabela Detetive.

4.2.8 Modelo Lógico total

Por fim, apresentamos o Modelo Lógico completo. Para obtermos este modelo, ainda tivemos de estabelecer as relações entre algumas das tabelas através da criação efetiva das chaves estrangeiras.

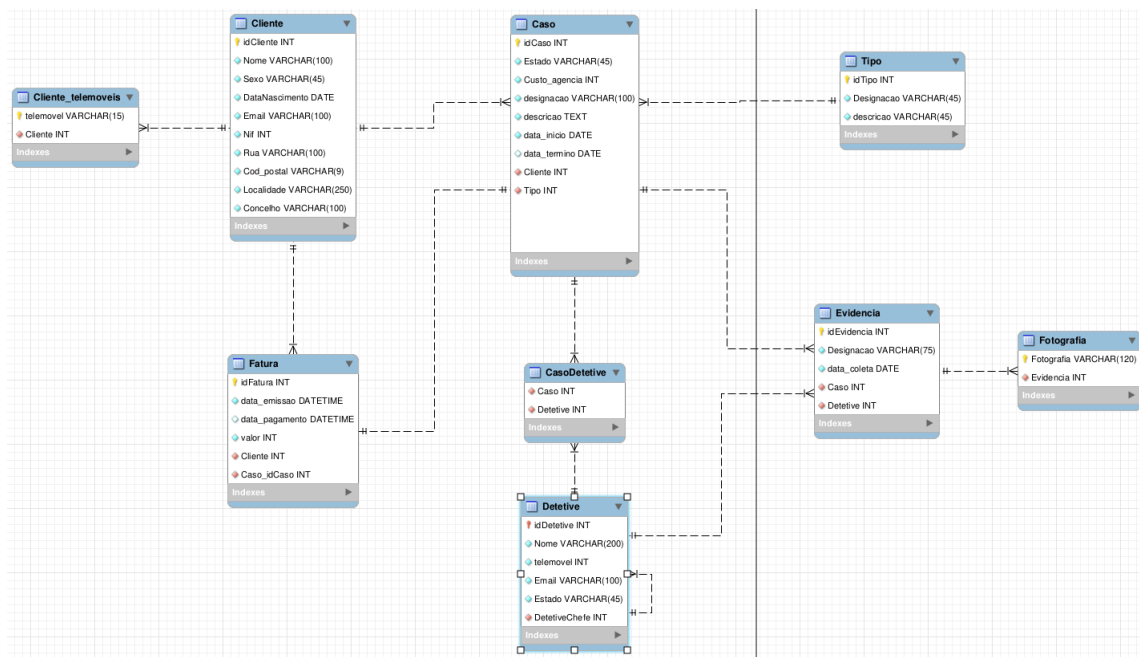


Tabela 20: Modelo Lógico

4.3 Normalização de Dados

O modelo lógico produzido encontra-se nas 3 primeiras formas normais. Passando agora a explicar o porquê de o nosso modelo se encontrar em cada uma das formas normais. A primeira forma normal informa-nos que cada tabela deve ter uma chave primária única, que não devemos ter informações repetidas e que os valores dos atributos devem ser atômicos. Perante estas regras e após análise do nosso modelo lógico, podemos facilmente concluir que este se encontra na primeira forma normal, algo que era expectável caso a conversão do modelo conceptual para o lógico fosse feita de forma adequada. Quanto à segunda forma normal, esta aplica-se maioritariamente a tabelas que possuam chaves compostas e indica-nos que não deve haver dependência parcial das colunas em relação à nossa chave. No nosso caso, facilmente verificamos que o nosso modelo se encontra na segunda forma normal pelo facto de não possuímos chaves compostas. Por fim, temos a terceira forma normal que nos diz que não devemos ter dependências transitivas; estas referem-se a um atributo não chave estar dependente de outro. No nosso caso, mais uma vez isto não acontece, pois não possuímos atributos não chave que sejam referência para outras tabelas. Após esta análise, podemos concluir que o nosso modelo se encontra nas 3 primeiras formas normais.

4.4 Validação do Modelo com Interrogações do Utilizador

Ordena os clientes por nif de forma ascendente:

- "Query": τ nif asc (Cliente)

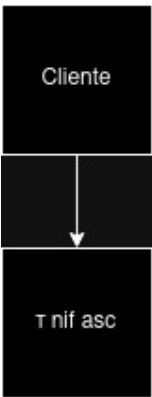


Tabela 21: Árvore Algebra

Retorna todos os casos solicitados por um Cliente:

- "Query": $\sigma_{idCliente=1}((Cliente) \bowtie idCliente=Cliente(Caso))$

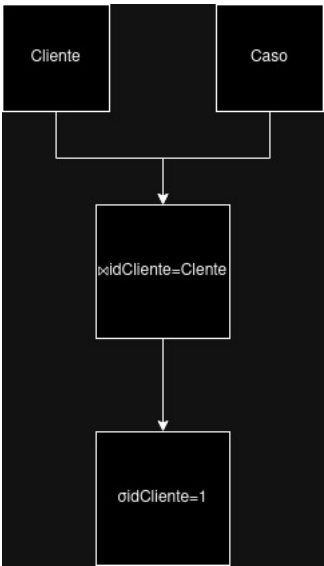


Tabela 22: Arvore Algebra

Retorna todas as evidências de um determinado Caso:

- "Query": $\sigma_{idCaso=1}((Caso) \bowtie_{idCaso=Caso} (Evidencia))$

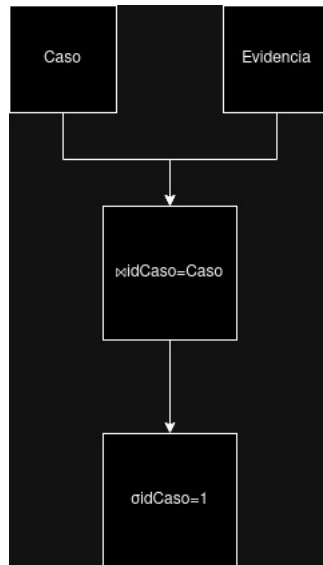


Tabela 23: Arvore Algebra

Conta o número de casos de um dado cliente:

- "Query": $\rho(\text{contaCasos}) (\text{count} (\sigma_{\text{idCliente}=1} ((\text{Cliente}) \bowtie_{\text{idCliente}=\text{Cliente}} (\text{Caso}))))$

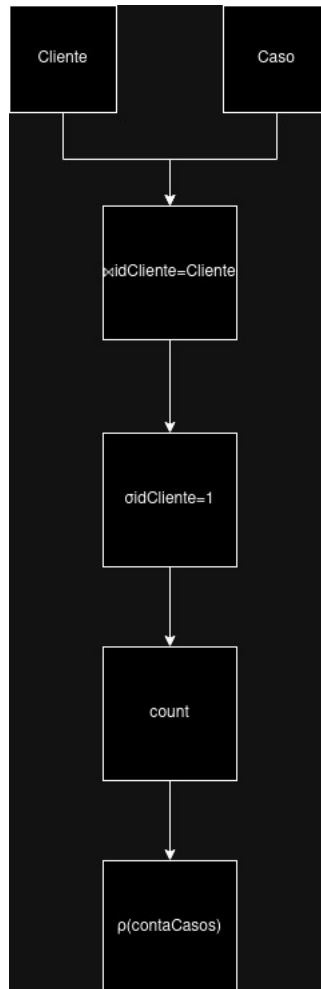


Tabela 24: Arvore Algebra

5 Implementação Física

5.1 Apresentação e explicação da base de dados implementada

Com base no modelo lógico produzido na fase anterior, é possível identificar as dependências existentes entre as diferentes tabelas, o que gera uma ordem pela qual as tabelas devem ser criadas. Assim sendo, as primeiras tabelas a serem criadas são as tabelas que não apresentam chaves estrangeiras, isto porque, quando uma dada tabela apresenta uma chave estrangeira, essa tabela está dependente da tabela da qual essa chave estrangeira é referenciada (sendo que nessa tabela, a variável será a sua chave primária). Contudo, no caso de detetive, ele apresenta uma chave estrangeira proveniente de si mesmo, isto não causa problema na construção das tabelas, mas pode provocar problemas no povoamento caso os detetives chefes não sejam inseridos primeiro na base de dados, antes dos detetives os quais estão subordinados eles.

Um outro ponto importante é especificar o comportamento de certos atributos, como restrições sobre os valores que estes podem tomar. De forma a simplificar, atributos que tomam valores enumerados (exemplo: estado), restringimos os seus dados a um conjunto de valores (estado apenas pode tomar o valor de 'C' ou 'E' ou 'T'). Um outro exemplo é especificar como uma dado individuo de uma tabela deve reagir a alterações da sua chave estrangeira, isto é, o caso de haver alterações ou de ser removida.

Identificando a ordem das tabelas e as restrições que devam existir, foram criadas as *SQL queries* do tipo *DDL (Data Definition Language)*, onde estão especificado o nome da Base de Dados "*AgenciaDetetivesSr.Alfredo*", a especificação das diferentes tabelas, e também os seus atributos.

Com o auxílio do MySQL, o grupo decidiu recorrer ao MySQL Workbench como ferramenta de desenho, desenvolvimento e administração de base de dados.

Procedemos então à conversão das tabelas do modelo lógico anteriormente apresentado, resultando nas seguintes implementações:

5.1.1 Criação do esquema de bases de dados

```
-----  
-- Schema AgenciaDetetivesSr.Alfredo  
-----  
CREATE SCHEMA IF NOT EXISTS `AgenciaDetetivesSr.Alfredo` DEFAULT CHARACTER SET utf8 ;  
-- drop database `AgenciaDetetivesSr.Alfredo`;  
USE `AgenciaDetetivesSr.Alfredo` ;
```

Tabela 1: Esquema - Agência Detetives

De modo, a organizar as tabelas da nossa Agência de Detetives foi criado um esquema, que irá organizar também views e procedimentos, facilitando assim a gestão da base de dados.

5.1.2 Implementação da tabela: Clientes

```
-----  
-- Table `AgenciaDetetivesSr.Alfredo`.`clientes`  
-----  
CREATE TABLE IF NOT EXISTS `AgenciaDetetivesSr.Alfredo`.`clientes` (  
  `idCliente` INT NOT NULL AUTO_INCREMENT,  
  `nome` VARCHAR(100) NOT NULL,  
  `sexo` VARCHAR(45) NOT NULL,  
  `dta_nascimento` DATE NOT NULL,  
  `email` VARCHAR(100) NOT NULL,  
  `nif` INT NOT NULL,  
  `rua` VARCHAR(100) NOT NULL,  
  `cod_postal` VARCHAR(9) NOT NULL,  
  `localidade` VARCHAR(250) NOT NULL,  
  `concelho` VARCHAR(100) NOT NULL,  
  -- Restrições sobre formato e valores de alguns parâmetros  
  CONSTRAINT `chk_nif` CHECK(LENGTH(`nif`) = 9),  
  CONSTRAINT `chk_sexo` CHECK(`sexo` IN ('F','M','I')),  
  CONSTRAINT `chk_cod_postal` CHECK (`cod_postal` REGEXP '^[0-9]{4}-[0-9]{3}$'),  
  PRIMARY KEY (`idCliente`));
```

Tabela 2: Implementação da tabela: Clientes

Nesta tabela de cliente temos uma série de atributos que constituem a informação pessoal do cliente, o ID do cliente representa uma chave primária, uma vez que se trata de um identificador único de cada cliente, neste atributo foi configurado um auto incremento para garantir que cada novo cliente tenha um ID único. No que toca ao nif, sexo e código postal, foram definidas restrições sobre o formato e valores, o nif do cliente deverá ter 9 dígitos respeitando o formato convencional dos nifs, por exemplo, 123654789, o sexo do cliente está compreendido entre 3 valores, 'F' feminino, 'M' masculino e 'I' intersexo, por fim, o código postal deverá respeitar

um formato com 4 dígitos numéricos, um hífen e 3 dígitos numéricos seguidos deste hífen, por exemplo, 4705-245.

Para garantir que um NIF seja único foi adicionada uma restrição no atributo do NIF:

```
ALTER TABLE clientes
ADD CONSTRAINT unq_nif UNIQUE (nif);
```

Tabela 3: Atributo NIF único

5.1.3 Implementação da tabela: Cliente_Telemoveis

```
-----
-- Table `AgenciaDetetivesSr.Alfredo`.`cliente_telemoveis`
-----
CREATE TABLE IF NOT EXISTS `AgenciaDetetivesSr.Alfredo`.`cliente_telemoveis` (
  `telemovel` VARCHAR(15) NOT NULL,
  `idCliente` INT NOT NULL,
  PRIMARY KEY (`telemovel`),
  CONSTRAINT `fk_cltelemoveis_clinte`
    FOREIGN KEY (`idCliente`)
    REFERENCES `AgenciaDetetivesSr.Alfredo`.`clientes` (`idCliente`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);
```

Tabela 4: Implementação da tabela: Cliente_Telemoveis

A tabela 'cliente_telemoveis' foi criada para armazenar os números de telemóvel associados a cada cliente, nesta tabela temos o telemóvel definido como chave primária e não nulo, o que significa que cada registo nesta tabela deve incluir um número de telemóvel. Temos o ID do cliente que se trata de uma chave estrangeira que faz referência ao 'idCliente' da tabela do Cliente, isto estabelece uma relação entre o cliente e os seus números de telefone, indicando que cada número de telefone pertence a um cliente específico.

As restrições de ação 'ON DELETE NO ACTION' e 'ON UPDATE NO ACTION' são especificadas, o que significa que se um registo na tabela 'Clientes' for removido ou atualizado, nenhuma ação será tomada nesta tabela de telemóveis do cliente, isto mantém a integridade dos dados, assegurando que os números de telemóvel não serão automaticamente removidos ou atualizados quando um cliente é removido ou atualizado.

5.1.4 Implementação da tabela: Casos

```
-- Table `AgenciaDetetivesSr.Alfredo`.`Caso`
-----
CREATE TABLE IF NOT EXISTS `AgenciaDetetivesSr.Alfredo`.`casos` (
  `idCaso` INT NOT NULL AUTO_INCREMENT,
  `estado` CHAR(1) NOT NULL,
  `custo_agencia` INT NOT NULL,
  `designacao` VARCHAR(100) NOT NULL,
  `descricao` TEXT NOT NULL,
  `dta_inicio` DATE NOT NULL,
  `dta_conclusao` DATE NULL,
  `Cliente` INT NOT NULL,
  `Tipo` INT NOT NULL,
  PRIMARY KEY(`idCaso`),
  -- Adicionar uma check constraint para limitar os valores de inserção na coluna estado
  -- E-Em Execução, C-Congelado, T-Terminado
  CONSTRAINT `chk_estado` CHECK(`estado` IN ('E', 'C', 'T')),
  -- Adicionar uma check constraint para garantir que a dta_conclusao é sempre posterior à dta_inicio
  CONSTRAINT `chk_dtas` CHECK(`dta_conclusao` > `dta_inicio`),
  CONSTRAINT `fk_casos_cliente`
    FOREIGN KEY (`Cliente`)
    REFERENCES `AgenciaDetetivesSr.Alfredo`.`clientes` (`idCliente`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_casos_tipo`
    FOREIGN KEY (`Tipo`)
    REFERENCES `AgenciaDetetivesSr.Alfredo`.`tipos` (`idTipo`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION
);
```

Tabela 5: Implementação da tabela: Casos

A tabela Casos foi criada para gerir informações sobre casos tratados pela agência de detetives, em que temos alguns atributos que tratam da informação de um caso, o ID do caso foi definido como chave primária, uma vez que se trata de um identificador único para cada caso, neste atributo foi configurado um auto incremento para garantir que cada novo Caso tenha um ID único.

No que toca ao estado de um caso, foi definida uma restrição que limita os valores a 'E' Em Execução, 'C' Congelado e 'T' Terminado, garantindo a consistência dos dados inseridos. Para a data de início e conclusão foi definida uma restrição para garantir que a data de conclusão seja sempre posterior à data de início, evitando assim inconsistências temporais.

A chave estrangeira 'Cliente' estabelece a relação com o 'idCliente' da tabela 'Clientes', isto é, cada caso está associado a um cliente, deste modo, será possível verificar quem solicitou um dado caso. As ações 'ON DELETE NO ACTION' e 'ON UPDATE NO ACTION', indicam que nenhuma ação automática será realizada nesta tabela 'Caso' quando um registo da tabela 'Clientes' é removido ou atualizado, preservando assim a integridade dos registos de casos. A chave estrangeira 'Tipo' faz referência ao 'idTipo' da tabela 'Tipos' que se trata de um Tipo de Caso, permitindo assim a categorização dos casos por tipo, as ações da chave estrangeira 'Cliente' aplicam-se também para esta chave estrangeira 'Tipo', garantindo que atualizações ou remoções na tabela 'Tipos' não afetem os dados presentes na tabela 'Casos'.

5.1.5 Implementação da tabela: Tipos

```
-----  
-- Table `AgenciaDetetivesSr.Alfredo`.`Tipo`  
-----  
CREATE TABLE IF NOT EXISTS `AgenciaDetetivesSr.Alfredo`.`tipos` (  
  `idTipo` INT NOT NULL AUTO_INCREMENT,  
  `Designacao` VARCHAR(45) NOT NULL,  
  `descricao` VARCHAR(45) NOT NULL,  
  PRIMARY KEY (`idTipo`)  
);
```

Tabela 6: Implementação da tabela: Tipos

A tabela 'Tipos' foi criada para organizar os casos da agência, em diversos tipos de casos, nesta tabela temos informação relevante de cada tipo de caso, o ID do tipo do caso é a chave primária desta tabela, pelo facto de ser um identificador único de cada tipo de caso, sendo configurado um auto incremento para garantir que cada novo tipo de caso tenha um ID único.

5.1.6 Implementação da tabela: Faturas

```
-- -----  
-- Table `AgenciaDetetivesSr.Alfredo`.`faturas`  
-- -----  
CREATE TABLE IF NOT EXISTS `AgenciaDetetivesSr.Alfredo`.`faturas` (  
  `idFatura` INT NOT NULL AUTO_INCREMENT,  
  `dta_emissao` DATETIME NOT NULL,  
  `dta_pagamento` DATETIME NULL,  
  `valor` INT NOT NULL,  
  `cliente` INT NOT NULL,  
  `caso` INT NOT NULL,  
  -- Adicionar uma check constraint para garantir que a dta_pagamento é sempre posterior à dta_emissao  
  CONSTRAINT `chk_dtas_faturas` CHECK(`dta_pagamento` > `dta_emissao`),  
  -- Adicionar uma check constraint para impedir a inserção de valores negativos  
  CONSTRAINT `chk_valor_fatura` CHECK(`valor` >= 0),  
  CONSTRAINT `fk_fatura_cliente`  
    FOREIGN KEY (`cliente`)  
    REFERENCES `AgenciaDetetivesSr.Alfredo`.`clientes` (`idCliente`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  CONSTRAINT `fk_fatura_caso`  
    FOREIGN KEY (`caso`)  
    REFERENCES `AgenciaDetetivesSr.Alfredo`.`casos` (`idCaso`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  PRIMARY KEY(`idFatura`)  
);
```

Tabela 7: Implementação da tabela: Faturas

A tabela 'Faturas' foi criada para organizar as faturas dos casos solicitados por cada cliente da agência, o ID da fatura é a chave primária desta tabela, uma vez que se trata de um identificador único de cada fatura, este foi configurado com um incremento automático para garantir que cada nova fatura tenha um ID único.

Além disso, foram estabelecidas restrições na data de pagamento e emissão para garantir que a data de pagamento seja sempre posterior à data de emissão, evitando assim inconsistências temporais, foi estabelecida também uma restrição no valor da fatura para impedir a inserção de valores negativos.

A chave estrangeira 'cliente' faz referência ao 'idCliente' da tabela do cliente que solicitou um caso, a tabela de faturas possui também uma outra chave estrangeira 'caso' que faz referência ao 'idCaso' da tabela 'Casos' que originou esta fatura. Ambas as chaves estrangeiras são configuradas com as ações 'ON DELETE NO ACTION' e 'ON UPDATE NO ACTION', o que significa que nenhuma ação será realizada na tabela 'Faturas' quando os dados correspondentes das tabelas 'Clientes' e 'Casos' forem atualizados ou removidos. Preservando assim a integridade do histórico das faturas, mesmo se os casos ou clientes forem alterados.

5.1.7 Implementação da tabela: Detetives

```
-----  
-- Table `AgenciaDetetivesSr.Alfredo`.`detetives`  
-----  
CREATE TABLE IF NOT EXISTS `AgenciaDetetivesSr.Alfredo`.`detetives`(  
  `idDetetive` INT NOT NULL AUTO_INCREMENT,  
  `Nome` VARCHAR(200) NOT NULL,  
  `telemovel` INT NOT NULL,  
  `email` VARCHAR(100) NOT NULL,  
  `estado` CHAR(1) NOT NULL,  
  `detetiveChefe` INT NOT NULL,  
  -- Adicionar uma check constraint para limitar os valores de inserção na coluna estado  
  -- A-Ativo I-Inativo  
  CONSTRAINT `chk_estado_detetive` CHECK(`estado` IN ('A', 'I')),  
  CONSTRAINT `fk_detetive_detetive`  
    FOREIGN KEY (`detetiveChefe`)  
    REFERENCES `AgenciaDetetivesSr.Alfredo`.`detetives` (`idDetetive`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  PRIMARY KEY(`idDetetive`)  
);
```

Tabela 8: Implementação da tabela: Detetives

Nesta tabela de detetives temos uma série de atributos que constituem a informação pessoal/profissional do detetive da agência, o ID do detetive representa uma chave primária, uma vez que se trata de um identificador único de cada detetive, foi configurado neste atributo um incremento automático para garantir que cada novo detetive tenha um ID único. No que toca ao estado de um detetive, foi definida uma restrição que limita os valores a 'A' Ativo e 'I' Inativo, garantindo a consistência dos dados inseridos.

A chave estrangeira 'detetiveChefe' faz referência à própria tabela 'Detetives' uma vez que os detetives podem ser geridos por um detetive Chefe, esta chave foi configurada com as ações 'ON DELETE NO ACTION' e 'ON UPDATE NO ACTION', garantindo que a exclusão ou atualização de um detetive chefe não altere automaticamente os dados dos detetives que estão a ser geridos, preservando assim a integridade dos dados.

5.1.8 Implementação da tabela: Caso Detetive

```
-----  
-- Table `AgenciaDetetivesSr.Alfredo`.`detetives`  
-----  
CREATE TABLE IF NOT EXISTS `AgenciaDetetivesSr.Alfredo`.`detetives`(  
  `idDetetive` INT NOT NULL AUTO_INCREMENT,  
  `Nome` VARCHAR(200) NOT NULL,  
  `telemovel` INT NOT NULL,  
  `email` VARCHAR(100) NOT NULL,  
  `estado` CHAR(1) NOT NULL,  
  `detetiveChefe` INT NOT NULL,  
  -- Adicionar uma check constraint para limitar os valores de inserção na coluna estado  
  -- A-Ativo I-Inativo  
  CONSTRAINT `chk_estado_detetive` CHECK(`estado` IN ('A', 'I')),  
  CONSTRAINT `fk_detetive_detetive`  
    FOREIGN KEY (`detetiveChefe`)  
    REFERENCES `AgenciaDetetivesSr.Alfredo`.`detetives` (`idDetetive`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  PRIMARY KEY(`idDetetive`)  
);
```

Tabela 9: Implementação da tabela: Caso Detetive

A tabela Caso Detetive foi criada para gerir os casos designados a cada detetive, nesta agência de detetives, vários detetives podem estar envolvidos em vários casos, nesta tabela temos duas chaves estrangeiras, o 'caso' que faz referência ao 'idCaso' da tabela 'Casos', o que assegura que qualquer caso ao qual um detetive está designado seja um caso existente e registado na tabela de 'Casos', o 'detetive' faz referência ao 'idDetetive' da tabela 'Detetives', o que garante que qualquer detetive associado a um caso através desta tabela seja também um detetive existente na tabela de 'Detetives'.

Ambas as chaves estrangeiras estão configuradas com as ações 'ON DELETE NO ACTION' e 'ON UPDATE NO ACTION', o que significa que se um detetive ou caso for removido ou atualizado, nenhuma ação será realizada na tabela 'Caso Detetive', mantendo a integridade dos dados.

5.1.9 Implementação da tabela: Evidências

```
-- Table `AgenciaDetetivesSr.Alfredo`.`evidencias`  
-----  
CREATE TABLE IF NOT EXISTS `AgenciaDetetivesSr.Alfredo`.`evidencias`(  
  `idEvidencia` INT NOT NULL AUTO_INCREMENT,  
  `designacao` VARCHAR(75) NOT NULL,  
  `data_coleta` DATE NOT NULL,  
  `caso` INT NOT NULL,  
  `detetive` INT NOT NULL,  
  CONSTRAINT `fk_evidencia_detetive`  
    FOREIGN KEY (`detetive`)  
    REFERENCES `AgenciaDetetivesSr.Alfredo`.`detetives` (`idDetetive`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  CONSTRAINT `fk_evidencia_caso`  
    FOREIGN KEY (`caso`)  
    REFERENCES `AgenciaDetetivesSr.Alfredo`.`casos` (`idCaso`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  PRIMARY KEY (`idEvidencia`)  
)
```

Tabela 10: Implementação da tabela: Evidências

A tabela Evidências foi criada para gerir as evidências recolhidas de um detetive sobre um determinado caso, nesta tabela temos algumas informações relevantes sobre uma evidência, o 'idEvidencia' representa a chave primária da tabela, uma vez que se trata de um identificador único, esta chave foi configurada com um auto incremento para garantir que cada nova evidência recolhida tenha um ID único.

Estão inseridas nesta tabela duas chaves estrangeiras, a do detetive que faz referência ao 'idDetetive' da tabela de Detetives e a do caso que faz referência ao 'idCaso' da tabela de Casos, deste modo, podemos visualizar numa dada evidência que detetive a recolheu e o caso correspondente.

Ambas as chaves estrangeiras foram configuradas com as ações 'ON DELETE NO ACTION' e 'ON UPDATE NO ACTION', o que significa que se um detetive ou caso for removido ou atualizado, nenhuma ação será realizada na tabela 'Evidências', assim, será possível manter o histórico de evidências, preservando a integridade destes dados.

5.1.10 Implementação da tabela: Fotografias

```
-- Table `AgenciaDetetivesSr.Alfredo`.`fotografias`  
  
CREATE TABLE IF NOT EXISTS `AgenciaDetetivesSr.Alfredo`.`fotografias` (  
  `fotografia` VARCHAR(75) NOT NULL,  
  `evidencia` INT NOT NULL,  
  CONSTRAINT `fk_fotografia_evidencia`  
    FOREIGN KEY (`evidencia`)  
      REFERENCES `AgenciaDetetivesSr.Alfredo`.`evidencias` (`idEvidencia`)  
      ON DELETE NO ACTION  
      ON UPDATE NO ACTION,  
  PRIMARY KEY(`fotografia`)  
);
```

Tabela 11: Implementação da tabela: Evidências

A tabela Fotografias foi criada, devido ao facto de uma evidência possuir uma lista de fotografias que são importantes para reforçar a evidência de um determinado caso, nesta tabela temos inserida a chave estrangeira 'evidencia' que faz referência ao 'idEvidencia' da tabela 'Evidencias', deste modo, podemos ter conhecimento das fotografias de uma dada evidência.

Esta chave estrangeira foi configurada com as ações 'ON DELETE NO ACTION' e 'ON UPDATE NO ACTION', o que significa que se uma evidência for removida ou atualizada, nenhuma ação será realizada na tabela 'Fotografias', assim, será possível manter o histórico de fotografias, preservando a integridade destes dados.

5.2 Criação de utilizadores da base de dados

Os utilizadores criados para interagir com a base de dados, e as permissões que cada um recebeu foram dadas consoantes os requisitos de controlo, isto é, o que cada um pode fazer na base de dados. Para isto foram criados 3 utilizadores diferentes:

```
CREATE USER 'admin_user'@'localhost' IDENTIFIED BY 'admin123';
CREATE USER 'client_user'@'localhost' IDENTIFIED BY 'client123';
CREATE USER 'detective_user'@'localhost' IDENTIFIED BY 'detective123';
```

Tabela 12: Criação de Utilizadores

Como se pode verificar no exemplo de código anterior, os utilizadores criados foram um administrador, um detetive e um cliente, sendo que as permissões que cada um tem estão relacionadas com os requisitos de controlo, como foi dito anteriormente.

O administrador, será um utilizador que poderá interagir com a base de dados com todos os privilégios, isso é dado através do seguinte pedaço de código.

```
GRANT ALL PRIVILEGES ON `AgenciaDetetivesSr.Alfredo`.* TO 'admin_user'@'localhost';
```

Tabela 13: Administrador com todas as permissões sobre o *schema*

As restantes permissões, de cada utilizador, vão ser agora enumeradas consoante os requisitos de controlo.

RC1 - Só o administrador pode gerar as faturas.

```
-- RC1: Only admin can generate invoices
GRANT ALL PRIVILEGES ON `AgenciaDetetivesSr.Alfredo`.`faturas` TO 'admin_user'@'localhost';
```

Tabela 14: RC1

RC2 - Cada cliente tem apenas acesso aos dados dos seus casos solicitados.

```
-- Granting execution permissions on procedures to clients
GRANT EXECUTE ON PROCEDURE `AgenciaDetetivesSr.Alfredo`.`getClientCases` TO 'client_user'@'localhost';

-- Granting basic permissions to allow login and use of database to clients
GRANT USAGE ON *.* TO 'client_user'@'localhost';
GRANT SELECT ON `AgenciaDetetivesSr.Alfredo`.`casos` TO 'client_user'@'localhost';
GRANT SELECT ON `AgenciaDetetivesSr.Alfredo`.`clientes` TO 'client_user'@'localhost';
```

Tabela 15: RC2

O processo *GetClientCases* está definido em Anexo 1.

RC3 - Só o administrador pode criar/editar/eliminar as informações de um caso.

```
-- RC3: Only admin can create/edit/delete case information
GRANT ALL PRIVILEGES ON `AgenciaDetetivesSr.Alfredo`.`casos` TO 'admin_user'@'localhost';
```

Tabela 16: RC3

RC4 - Só o administrador pode criar/consultar/editar/eliminar as informações de qualquer cliente. Cada cliente pode realizar o mesmo, mas apenas das suas informações.

```
-- RC4: Only admin can create/view/edit/delete any client information.
-- Each client can do the same but only for their own information
GRANT ALL PRIVILEGES ON `AgenciaDetetivesSr.Alfredo`.`clientes` TO 'admin_user'@'localhost';
```

Tabela 17: RC4

RC5 - Só o administrador e os detetives podem consultar as informações dos clientes.

```
-- RC5: Only admin and detectives can view client information
GRANT SELECT ON `AgenciaDetetivesSr.Alfredo`.`clientes` TO 'admin_user'@'localhost';
GRANT SELECT ON `AgenciaDetetivesSr.Alfredo`.`clientes` TO 'detective_user'@'localhost';
```

Tabela 18: RC5

RC6 - Cada cliente tem apenas acesso aos seus dados pessoais.

```
GRANT EXECUTE ON PROCEDURE `AgenciaDetetivesSr.Alfredo`.`GetClientInfo` TO 'client_user'@'localhost';
```

Tabela 19: RC6

O processo *GetClientInfo* está definida em Anexo 2.

RC7 - Só o administrador e os detetives podem consultar as informações de qualquer caso.

```
-- RC7: Only admin and detectives can view any case information
GRANT SELECT ON `AgenciaDetetivesSr.Alfredo`.`casos` TO 'admin_user'@'localhost';
GRANT SELECT ON `AgenciaDetetivesSr.Alfredo`.`casos` TO 'detective_user'@'localhost';
```

Tabela 20: RC7

RC8 - Só os detetives podem registrar/consultar/editar evidências de um caso.

```
-- RC8: Only detectives can register/view/edit case evidence
GRANT ALL PRIVILEGES ON `AgenciaDetetivesSr.Alfredo`.`evidencias` TO 'detective_user'@'localhost';
```

Tabela 21: RC8

RC9 - Só o administrador pode criar/consultar/editar/eliminar as informações de qualquer detetive. Cada detetive pode realizar o mesmo, mas apenas das suas informações.

```
-- RC9: Only admin can create/view/edit/delete any detective information.
-- Each detective can do the same but only for their own information
GRANT ALL PRIVILEGES ON `AgenciaDetetivesSr.Alfredo`.`detetives` TO 'admin_user'@'localhost';
-- Granting execution permissions on procedures to detective
GRANT EXECUTE ON PROCEDURE `AgenciaDetetivesSr.Alfredo`.`UpdateDetectiveInfo` TO 'detective_user'@'localhost';
```

Tabela 22: RC9

O processo *UpdateDetectiveInfo* está definido em Anexos 3.

RC10 - Só o administrador e os detetives podem consultar o histórico de casos solicitados por qualquer cliente.

```
-- RC10: Only admin and detectives can view the history of cases requested by any client
GRANT SELECT ON `AgenciaDetetivesSr.Alfredo`.`casos` TO 'admin_user'@'localhost';
GRANT SELECT ON `AgenciaDetetivesSr.Alfredo`.`casos` TO 'detective_user'@'localhost';
```

Tabela 23: RC10

5.3 Povoamento da base de dados

O povoamento da base de dados, pode ser efetuado de duas maneiras, sendo uma delas através de um *script SQL* que contém **SQL queries** que serão executadas diretamente na bases de dados, uma outra forma é através de um *script* em *Python* que povoa a base de dados através de uma conexão externa com a base de dados.

Tal como é referenciado no secção 5.1, é necessário um cuidado quando criamos as tabelas devido às dependências entre elas. O mesmo acontece no povoamento, quando se tenta inserir uma nova entrada numa tabela, na qual apresenta uma dependência de uma outra tabela, primeiro tem que se adicionar a entrada que é responsável pela chave estrangeira, e posteriormente adicionar a entrada que se relaciona com essa entrada. Um exemplo é os detetives, que apresentam um detetive chefe, antes de adicionar os detetives comuns, o detetive chefe tem que ser o primeiro a ser adicionado para não causar problemas com as chaves estrangeiras.

```
/* Inserindo dados na tabela detetives */  
INSERT INTO detetives (Nome, telemovel, email, estado, detetiveChefe)  
VALUES  
( 'Alfredo Sousa', 911223344, 'alfredo.sousa@gmail.com', 'A', 1),  
( 'Miguel Costa', 922334455, 'miguel.costa@gmail.com', 'A', 1),  
( 'Ana Pereira', 933445566, 'ana.pereira@gmail.com', 'I', 1);
```

Tabela 24: Entradas na Tabela Detetive

Analisando o caso anterior, os detetives adicionados na Base de Dados, terão todos o mesmo detetive chefe, que por sua vez já terá um registo relativamente a si na base de dados.

Relativamente às maneiras de inserir dados, ou seja, povoar a base de dados, a utilização de um *script SQL*, faz com que neles as *queries* do tipo *INSERT* estejam já criadas, especificando a tabela na qual desejam inserir a entrada, como também que parâmetros deseja inicializar, e posteriormente os seus valores iniciais (valores iniciais, pois posteriormente, alguns desses valores podem ser alterados). O exemplo anterior é um caso de inserção de uma entrada na tabela de detetives, onde são criadas três entradas, uma por cada linha após *VALUES*. O ficheiro em si pode ter múltiplas *queries* que adicionaram cada uma, várias novas entradas nas tabelas que selecionarem.

Sobre o povoamento externo, isto é, utilizando o programa em *Python* que povoa a base de dados através de uma conexão. Para efetuar a conexão com a base de dados foi usada a biblioteca *mysql.connector* que fornece uma função que dando o ip, número de porta, nome da base de dados e as credencias do utilizador é possível estabelecer uma conexão com um sistema de base de dados, e como passamos o nome da base de dados na qual queremos comunicar, a conexão fica diretamente ligada a base de dados em si, neste caso, *'AgenciaDetetivesSr.Alfredo'* (O exemplo a seguir, demonstra um pedaço de código para a conexão e a inserção).

Listing 5.1: Estabelecimento de conexão com a base de dados

```
conn = mysql.connector.connect(  
    host='localhost',  
    user='root',  
    password='',  
    database='AgenciaDetetivesSr.Alfredo'  
)
```

Listing 5.2: Inserção de dados através do script em Python

```
cursor = conn.cursor()  
  
cursor.execute("""  
    INSERT INTO clientes (nome, sexo, dta_nascimento, email, nif, rua, cod_pos)  
    VALUES  
    ('Paulo Oliveira', 'M', '1975-01-15', 'paulo.oliveira@gmail.com', 223456789,  
    ('Fernanda Costa', 'F', '1982-04-12', 'fernanda.costa@gmail.com', 334567890,  
    ('Ricardo Pereira', 'M', '1992-07-21', 'ricardo.pereira@gmail.com', 445678901)  
    """)
```

Posteriormente à conexão, basta estruturar *SQL queries* e enviar as *queries* para a base de dados. No caso de erro, o programa imprime na linha de comando qual foi o erro que ocorreu.

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

Um aspeto fundamental na construção de uma base de dados é a alocação e gestão do espaço físico no hardware. Este aspeto é crucial, pois tem um impacto direto nos custos associados à manutenção da base de dados. Com isto em mente, realizou-se uma análise do modelo desenvolvido com o objetivo de determinar o espaço necessário para garantir um funcionamento eficiente da base de dados.

Tabela 5.1: Espaço ocupado no disco por cada tipo de dados

Tipo de Dados	Tamanho (Bytes)
INT	4
VARCHAR(N)	N+1
DATE	3
DATETIME	8
DOUBLE	8
TEXT	L+2, onde $L < L^{216} = \text{Length}$
DECIMAL	4

Tendo em conta a tabela acima representada, usando como referência o manual do MySQL [1], foi então calculado o espaço da nossa base de dados. Foi considerado um número de entradas de dados que seria realista num estado inicial da agência.

Neste momento a agência tem 17 clientes registados.

Tabela 5.2: Espaço ocupado no disco por cada atributo num Cliente

	Atributos	Tipo de Dados	Espaço no disco
Clientes	Contribuinte	INT	4
	Nome	VARCHAR(75)	76
	DataNascimento	DATE	3
	Genero	INT	4
	Contacto	INT	4
	Email	VARCHAR(100)	101
	Rua	VARCHAR(100)	101
	Localidade	VARCHAR(30)	31
	Código-Postal	VARCHAR(8)	9
Total			$333 \times 17 = 5661$ bytes

Neste momento, existem 20 contactos telefónicos de clientes registados na agência.

Tabela 5.3: Espaço ocupado no disco por cada atributo no ClienteTelemoveis

	Atributos	Tipo de Dados	Espaço no disco
ClienteTelemoveis	telemovel	VARCHAR(15)	16
	idCliente	INT	4
Total			$20 \times 20 = 400$ bytes

Na agência existem 5 tipos de casos definidos até o momento.

Tabela 5.4: Espaço ocupado no disco por cada atributo num Tipo

	Atributos	Tipo de Dados	Espaço no disco
Tipos	idTipo	INT	4
	Designação	VARCHAR(45)	46
	Descrição	VARCHAR(45)	46
Total			$96 \times 5 = 480$ bytes

Neste momento, existem 19 casos solicitados a serem investigados ou que já foram resolvidos na agência.

Tabela 5.5: Espaço ocupado no disco por cada atributo num Caso

	Atributos	Tipo de Dados	Espaço no disco
Casos	idCaso	INT	4
	Estado	VARCHAR(1)	2
	Custo Agência	INT	4
	Designação	VARCHAR(100)	101
	Descrição	TEXT	$L+2$
	Data de Inicio	DATE	3
	Data de Conclusão	DATE	3
	idCliente	INT	4
	idTipo	INT	4
Total			$19 \times (127 + L) \leq 2413 + 19L$ bytes

Neste momento, existem 9 faturas emitidas sobre um caso solicitado por um cliente na agência.

Tabela 5.6: Espaço ocupado no disco por cada atributo numa Faturas

	Atributos	Tipo de Dados	Espaço no disco
Faturas	idFatura	INT	4
	Data de Emissão	DATETIME	8
	Data de Pagamento	DATETIME	8
	Valor	INT	4
	idCliente	INT	4
	idCaso	INT	4
Total			$32 \times 9 = 288$ bytes

Na agência existem 10 detetives registados até o momento.

Tabela 5.7: Espaço ocupado no disco por cada atributo num Detetive

	Atributos	Tipo de Dados	Espaço no disco
Detetives	idDetetive	INT	4
	Nome	VARCHAR(200)	201
	Telemovel	INT	4
	Email	VARCHAR(100)	101
	Estado	VARCHAR(1)	2
	detetiveChefe	INT	4
Total			$316 \times 10 = 3160$ bytes

Neste momento, existem 10 detetives envolvidos em casos, alguns deles até em dois ou mais, para esta tabela temos atualmente 18 entradas de dados, que correspondem a casos que possuem um ou mais detetives envolvidos.

Tabela 5.8: Espaço ocupado no disco por cada atributo no Caso Detetive

	Atributos	Tipo de Dados	Espaço no disco
Caso Detetive	idCaso	INT	4
	idDetetive	INT	4
Total			$8 \times 18 = 144$ bytes

Neste momento, existem 27 evidências recolhidas na agência.

Tabela 5.9: Espaço ocupado no disco por cada atributo numa Evidência

	Atributos	Tipo de Dados	Espaço no disco
Evidências	idEvidencia	INT	4
	Designação	VARCHAR(75)	76
	Data de Coleta	DATE	3
	idCaso	INT	4
	idDetetive	INT	4
Total			$91 \times 27 = 2457$ bytes

Neste momento, existem 27 fotografias associadas às evidências

Tabela 5.10: Espaço ocupado no disco por cada atributo numa Fotografia

	Atributos	Tipo de Dados	Espaço no disco
Fotografias	Fotografia	VARCHAR(75)	76
	idEvidencia	INT	4
Total			$91 \times 27 = 2457$ bytes

Temos então uma base de dados com o seguinte tamanho:

Tabela 5.11: Espaço ocupado no disco pela base de dados

Tabela	Espaço no Disco
Cientes	5661 bytes
CientesTelemoveis	400 bytes
Casos	$413 + 19L$ bytes
Faturas	288 bytes
Tipos	480 bytes
Detetives	3160 bytes
CasoDetetive	144 bytes
Evidência	2457 bytes
Fotografias	2457 bytes
Total	$\leq 15\,460 + 19L$ bytes

5.4.1 Estimativa do crescimento anual

Perspetivando como poderá ser o crescimento da agência do Sr. Alfredo ao final de um ano, idealizamos um cenário hipotético. A partir deste, iremos retirar algumas conclusões do impacto deste crescimento na base de dados.

De acordo, com o cenário hipotético idealizado, vemos que foi verificado um crescimento anual de 10%, com isto, temos que o espaço ocupado no disco pela base de dados cresceu para 17006 + 21L bytes, o que atualmente não demonstra impacto significativo nos servidores instalados na agência.

Contudo, o crescimento do espaço ocupado pela base de dados ao final de cerca de 10 anos, poderá implicar uma atualização dos dados da base de dados ou um aumento dos recursos materiais (mais servidores).

5.5 Definição e caracterização de vistas de utilização em SQL

As vistas em SQL são ferramentas essenciais num Sistema de Gestão de Base de Dados (SGBD) que permitem a criação de tabelas virtuais a partir de consultas especificadas, essas vistas simplificam o acesso a informações complexas e facilitam consultas repetitivas, melhorando a eficiência e a segurança dos dados. Elas permitem ocultar informações sensíveis e garantir que os utilizadores apenas tenham acesso ao que é relevante consoante as suas permissões.

Além disso, as vistas ajudam na manutenção da base de dados, permitindo que alterações nas tabelas subjacentes não afetem as aplicações que dependem desses dados. Portanto, as vistas não só otimizam o desempenho e a recuperação de dados, mas também são essenciais para a segurança e o controlo de acesso dentro da agência.

Em seguida temos as vistas de utilização do nosso modelo:

- Tabela que mostra em cada mês quantos casos foram investigados na agência, o número de detetives envolvidos e o valor total faturado desses casos.

```
CREATE VIEW MonthlyInvoiceReport AS
SELECT
    DATE_FORMAT(f.dta_emissao, '%Y-%m') AS mes_ano,
    COUNT(DISTINCT f.caso) AS numero_casos_analisados,
    COUNT(DISTINCT cd.detetive) AS numero_detetives_envolvidos,
    SUM(f.valor) AS valor_total_faturado
FROM
    faturas f
LEFT JOIN caso_detetive cd ON f.caso = cd.caso
GROUP BY
    DATE_FORMAT(f.dta_emissao, '%Y-%m');
```

Tabela 25: Relatório Mensal

- Tabela que mostra o lucro mensal da agência

```
CREATE VIEW MonthlyProfit AS
SELECT
    DATE_FORMAT(f.dta_pagamento, '%Y-%m') AS mes_ano,
    SUM(f.valor) - SUM(c.custo_agencia) AS lucro
FROM
    faturas f
    INNER JOIN casos c ON f.caso = c.idCaso
WHERE
    f.dta_pagamento IS NOT NULL
GROUP BY
    DATE_FORMAT(f.dta_pagamento, '%Y-%m')
ORDER BY
    mes_ano;
```

Tabela 26: Lucro mensal

- Tabela que mostra para cada caso, juntamente com as suas informações, o número de detetives envolvidos

```
CREATE VIEW CasesWithMostDetectives AS
SELECT
    c.idCaso,
    c.designacao AS designacao_caso,
    c.descricao,
    COUNT(cd.detetive) AS numero_detetives
FROM
    casos c
    INNER JOIN caso_detetive cd ON c.idCaso = cd.caso
GROUP BY
    c.idCaso,
    c.designacao,
    c.descricao
ORDER BY
    numero_detetives DESC;
```

Tabela 27: Casos - Número de detetives

5.6 Tradução das interrogações do utilizador para SQL

A tradução das interrogações do utilizador para SQL refere-se à transformação de perguntas ou solicitações feitas pelos utilizadores que possam ser executadas numa base de dados. Essencialmente, este processo inclui a interpretação das intenções do utilizador e a sua tradução em operações que permitam recuperar, manipular ou atualizar informações na base de dados.

Seguem as interrogações do utilizadores desenvolvidas em SQL:

RM1 - A qualquer momento, o sistema deve apresentar um relatório de faturas mensais, com o número de casos analisados, o número de detetives envolvidos e o valor total faturado.

```
SELECT
    DATE_FORMAT(f.dta_emissao, '%Y-%m') AS mes_ano,
    COUNT(DISTINCT f.caso) AS numero_casos_analisados,
    COUNT(DISTINCT cd.detetive) AS numero_detetives_envolvidos,
    SUM(f.valor) AS valor_total_faturado
FROM
    faturas f
    LEFT JOIN caso_detetive cd ON f.caso = cd.caso
GROUP BY
    DATE_FORMAT(f.dta_emissao, '%Y-%m');
```

Tabela 28: Query 1 - RM1

RM2 - A qualquer momento, o sistema deve listar todos os casos solicitados por um cliente.

```
SELECT
    cl.nome,
    c.idCaso,
    c.designacao AS designacao,
    c.descricao,
    c.estado,
    c.custo_agencia,
    c.dta_inicio,
    c.dta_conclusao,
    t.Designacao AS tipo
FROM
    casos c
    INNER JOIN clientes cl ON c.Cliente = cl.idCliente
    INNER JOIN tipos t ON c.Tipo = t.idTipo
WHERE
    cl.idCliente = 1;
```

Tabela 29: Query 2 - RM2

RM3 - Deve ser possível saber quantos casos houve num mês.

```
SELECT
    DATE_FORMAT(dta_inicio, '%Y-%m') AS mes_ano,
    COUNT(*) AS numero_casos
FROM
    casos
WHERE
    DATE_FORMAT(dta_inicio, '%Y-%m') = '2024-01'
GROUP BY
    DATE_FORMAT(dta_inicio, '%Y-%m');

-- lista de clientes ordenada por nif
SELECT *FROM clientes
ORDER BY
    nif;
```

Tabela 30: Query 3 - RM3

RM4 - A qualquer momento, o sistema deve listar todos os clientes da agência, ordenados pelo o seu número de identificação fiscal

```
SELECT *FROM clientes
ORDER BY
    nif;
```

Tabela 31: Query 4 - RM4

RM6 - Deve ser possível consultar todos os detetives envolvidos num caso

```
SELECT
    d.idDetetive,
    d.Nome,
    d.telemovel,
    d.email,
    d.estado,
    d.detetiveChefe
FROM
    caso_detetive cd
    INNER JOIN detetives d ON cd.detetive = d.idDetetive
WHERE
    cd.caso = 2;
```

Tabela 32: Query 5 - RM6

RM7 - Deve ser possível saber quantos casos um detetive esteve envolvido num mês

```
SELECT
    d.idDetetive,
    d.Nome,
    DATE_FORMAT(c.dta_inicio, '%Y-%m') AS mes_ano,
    COUNT(DISTINCT c.idCaso) AS numero_casos
FROM
    caso_detetive cd
    INNER JOIN detetives d ON cd.detetive = d.idDetetive
    INNER JOIN casos c ON cd.caso = c.idCaso
GROUP BY
    d.idDetetive,
    DATE_FORMAT(c.dta_inicio, '%Y-%m')
ORDER BY
    d.idDetetive;
```

Tabela 33: Query 6 - RM7

RM8 - Deve ser possível consultar as evidências recolhidas por um detetive.

```
SELECT
    e.idEvidencia,
    e.designacao,
    e.data_coleta,
    e.caso,
    e.detetive
FROM
    evidencias e
    INNER JOIN detetives d ON e.detetive = d.idDetetive
WHERE
    d.idDetetive = 1;
```

Tabela 34: Query 7 - RM8

RM10 - Deve ser possível saber quantas evidências um caso teve.

```
SELECT
    c.idCaso,
    c.designacao AS designacao,
    COUNT(e.idEvidencia) AS numero_evidencias
FROM
    casos c
    LEFT JOIN evidencias e ON c.idCaso = e.caso
GROUP BY
    c.idCaso, c.designacao;
```

Tabela 35: Query 8 - RM10

RM11 - A qualquer momento, o sistema deve listar todos os casos da agência, ordenados pelo o seu tipo.

```
SELECT
    c.idCaso,
    c.designacao AS designacao,
    c.descricao,
    c.estado,
    c.custo_agencia,
    c.dta_inicio,
    c.dta_conclusao,
    t.designacao AS tipo
FROM
    casos c
    INNER JOIN tipos t ON c.Tipo = t.idTipo
ORDER BY
    t.designacao;
```

Tabela 36: Query 9 - RM11

RM12 - Deve ser possível consultar o histórico de casos de um detetive.

```
SELECT
    d.idDetetive,
    d.Nome AS nome_detetive,
    c.idCaso,
    c.designacao AS designacao_caso,
    c.descricao,
    c.estado,
    c.custo_agencia,
    c.dta_inicio,
    c.dta_conclusao,
    t.Designacao AS tipo
FROM
    detetives d
    INNER JOIN caso_detetive cd ON d.idDetetive = cd.detetive
    INNER JOIN casos c ON cd.caso = c.idCaso
    INNER JOIN tipos t ON c.Tipo = t.idTipo
WHERE
    d.idDetetive = 1
ORDER BY
    c.dta_inicio DESC;
```

Tabela 37: Query 10 - RM12

RM13 - A qualquer momento, o sistema deve listar todas as evidências, ordenadas pela data de coleta

```
SELECT
    e.idEvidencia,
    e.designacao,
    e.data_coleta,
    e.caso,
    e.detetive
FROM
    evidencias e
ORDER BY
    e.data_coleta;
```

Tabela 38: Query 11 - RM13

RM14 - Deve ser possível saber os casos que envolveram mais detetives.

```
SELECT
    c.idCaso,
    c.designacao AS designacao_caso,
    c.descricao,
    COUNT(cd.detetive) AS numero_detetives
FROM
    casos c
    INNER JOIN caso_detetive cd ON c.idCaso = cd.caso
GROUP BY
    c.idCaso,
    c.designacao,
    c.descricao
ORDER BY
    numero_detetives DESC;
```

Tabela 39: Query 12 - RM14

RM15 - A qualquer momento, o sistema deve listar todos os detetives pelo o seu estado.

```
SELECT
    d.idDetetive,
    d.Nome,
    d.telemovel,
    d.email,
    d.estado,
    d.detetiveChefe
FROM
    detetives d
ORDER BY
    d.estado;
```

Tabela 40: Query 13 - RM15

RM16 - A qualquer momento, o sistema deve listar todos os casos ordenados por estado (em execução, suspenso ou terminado)

```
SELECT
    c.idCaso,
    c.designacao AS designacao,
    c.descricao,
    c.estado,
    c.custo_agencia,
    c.dta_inicio,
    c.dta_conclusao,
    t.Designacao AS tipo
FROM
    casos c
    INNER JOIN tipos t ON c.Tipo = t.idTipo
ORDER BY
    c.estado;
```

Tabela 41: Query 14 - RM16

RM17 - A qualquer momento, o sistema deve listar todas as faturas já emitidas

```
SELECT
    f.idFatura,
    f.dta_emissao,
    f.dta_pagamento,
    f.valor,
    c.nome AS nome_cliente,
    cs.designacao AS designacao_caso
FROM
    faturas f
    INNER JOIN clientes c ON f.cliente = c.idCliente
    INNER JOIN casos cs ON f.caso = cs.idCaso;
```

Tabela 42: Query 15 - RM17

RM18 - A qualquer momento, o sistema deve listar todas as faturas pagas

```
SELECT
    f.idFatura,
    f.dta_emissao,
    f.dta_pagamento,
    f.valor,
    c.nome AS nome_cliente,
    cs.designacao AS designacao_caso
FROM
    faturas f
    INNER JOIN clientes c ON f.cliente = c.idCliente
    INNER JOIN casos cs ON f.caso = cs.idCaso
WHERE
    f.dta_pagamento IS NOT NULL
ORDER BY
    f.dta_pagamento DESC;
```

Tabela 43: Query 16 - RM18

RM19 - A qualquer momento, o sistema deve ser capaz de calcular o lucro obtido num mês

```
SELECT
    DATE_FORMAT(f.dta_pagamento, '%Y-%m') AS mes_ano,
    SUM(f.valor) - SUM(c.custo_agencia) AS lucro
FROM
    faturas f
    INNER JOIN casos c ON f.caso = c.idCaso
WHERE
    f.dta_pagamento IS NOT NULL
GROUP BY
    DATE_FORMAT(f.dta_pagamento, '%Y-%m')
ORDER BY
    mes_ano;
```

Tabela 44: Query 17 - RM19

RM20 - A qualquer momento, o sistema deve listar todas as faturas por pagar

```
SELECT
    f.idFatura,
    f.dta_emissao,
    f.valor,
    c.nome AS nome_cliente,
    cs.desginacao AS designacao_caso
FROM
    faturas f
    INNER JOIN clientes c ON f.cliente = c.idCliente
    INNER JOIN casos cs ON f.caso = cs.idCaso
WHERE
    f.dta_pagamento IS NULL
ORDER BY
    f.dta_emissao DESC;
```

Tabela 45: Query 18 - RM20

5.7 Indexação do Sistema de Dados

Afim de melhorar algumas funções de procura e visualização, foram criados alguns índices nas tabelas Clientes, Cliente_Telemoveis, Casos, Faturas, Detetives, Caso_Detetive, Evidencias e Fotografias.

```
-- clientes
CREATE INDEX idx_email ON clientes(email);
CREATE INDEX idx_nif ON clientes(nif);

-- cliente_telemoveis
CREATE INDEX idx_idCliente ON cliente_telemoveis(idCliente);

-- casos
CREATE INDEX idx_Cliente ON casos(Cliente);
CREATE INDEX idx_Tipo ON casos(Tipo);

-- faturas table
CREATE INDEX idx_cliente ON faturas(cliente);
CREATE INDEX idx_caso ON faturas(caso);

-- detetives
CREATE INDEX idx_detetiveChefe ON detetives(detetiveChefe);

-- caso_detetive
CREATE INDEX idx_caso ON caso_detetive(caso);
CREATE INDEX idx_detetive ON caso_detetive(detetive);

-- evidencias
CREATE INDEX idx_caso ON evidencias(caso);
CREATE INDEX idx_detetive ON evidencias(detetive);

-- fotografias
CREATE INDEX idx_evidencia ON fotografias(evidencia);
```

Tabela 46: Criação de indexação do sistema

O propósito destas indexações é otimizar os procedimentos que envolvem os atributos email e nif de um Cliente, o ID do cliente e do número telefónico de um Cliente, o ID do cliente e tipo de um Caso, o ID do cliente e do caso de uma Fatura, o atributo que permite identificar se o Detetive é chefe de um Detetive, o ID do caso e do Detetive de um Caso - Detetive (que permite identificar em que casos os detetives estão envolvidos), o ID do caso e do detetive de uma Evidência e o ID da evidência de uma Fotografia.

5.8 Procedimentos, funções e gatilhos

O gatilho criado para a base de dados, serve para que no momento em que um novo caso for adicionado à base de dados (evento), ele seja acionado para assim criar uma fatura associada ao cliente que pediu o caso e o caso em si. Assim sendo, no momento em que o novo caso seja criado, o gatilho irá pegar na data atual, calcular o preço correspondente, tendo em conta alguns dados do caso, e as chaves primárias de Cliente e Caso, e assim criar uma nova fatura associada a ambas as entradas responsáveis pelas chaves primárias.

```
CREATE TRIGGER AfterInsertCaso
AFTER INSERT ON casos
FOR EACH ROW
BEGIN
    INSERT INTO faturas (dta_emissao, valor, cliente, caso)
    VALUES (NOW(), NEW.custo_agencia * 1.5, NEW.Cliente, NEW.idCaso);
END$$
```

Tabela 47: Gatilho para Calcular Faturas

Relativamente à função criada, o seu objetivo é calcular a idade a partir da data fornecida. Esta função é criada com o intuito de simplificar o calculo das idades, pegando na data dada como *input* e calcular a sua idade atual. Esta função pode ser utilizada para calcular a idade de um cliente, como também à quanto tempo um dado foi dado como concluído ou inicializado.

```
-- funcao para calcular a idade
DELIMITER $$
CREATE FUNCTION CalcularIdade(dta_nascimento DATE)
RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE idade INT;
    SET idade = TIMESTAMPDIFF(YEAR, dta_nascimento, CURDATE());
    RETURN idade;
END$$
DELIMITER ;
```

Tabela 48: Calcular a idade de uma entrada

Para finalizar, foram criados vários processos, sendo cada um criado para um dado efeito. Como foi visto no sub-capítulo 5.2, algumas restrições foram efetuadas sobre quem pode executar um processo. Outros exemplos, estão relacionados com a adição de novas entradas às tabelas, como pode ser visto no exemplo seguinte:

```

CREATE PROCEDURE AdicionarClienteComTelemoveis(
    IN nome VARCHAR(100),
    IN sexo VARCHAR(45),
    IN dta_nascimento DATE,
    IN email VARCHAR(100),
    IN nif INT,
    IN rua VARCHAR(100),
    IN cod_postal VARCHAR(9),
    IN localidade VARCHAR(250),
    IN concelho VARCHAR(100),
    IN telemovel1 VARCHAR(15)
)
BEGIN
    DECLARE EXIT HANDLER FOR SQLEXCEPTION
    BEGIN
        ROLLBACK;
    END;

    START TRANSACTION;

    INSERT INTO clientes (nome, sexo, dta_nascimento, email, nif, rua, cod_postal, localidade, concelho)
    VALUES (nome, sexo, dta_nascimento, email, nif, rua, cod_postal, localidade, concelho);

    SET @idCliente = LAST_INSERT_ID();

    INSERT INTO cliente_telemoveis (telemovel, idCliente) VALUES (telemovel1, @idCliente);

    COMMIT;
END$$

```

Tabela 49: Adicionar um cliente com um telemóvel

O exemplo anterior é um processo que recebe como entrada os campos necessários para criar um novo cliente e o seu número de telefone, assim, quando haver necessidade de adicionar uma nova entrada à base de dados, em vez de construir a *query* de inserir nova entrada, basta apenas chamar o processo, dar os valores de entrada e a inserção é efetuada. Com esta abordagem, é garantida a coerência na adição de entradas. É de notar, que o processo descrito anteriormente, suporta caso de falha graças ao *TRANSACTION*. *TRANSACTION* assegura que caso o corpo do processo falhe, seja possível voltar ao estado antes de qualquer umas das instruções tenha sido executada, mantendo assim a base de dados consistente.

6 Conclusões e Trabalho Futuro

Durante este trabalho realizámos todos os passos necessários para efetuar a contextualização do nosso caso de estudo, bem como a preparação necessária para a elaboração do modelo conceptual, por meio da definição de requisitos e, subsequentemente, da conversão dos mesmos para o modelo efetivo. Seguidamente, definimos os dicionários de dados das entidades, bem como dos seus atributos e dos relacionamentos. Esta fase revelou-se bastante útil, pois facilitou significativamente a conversão para o modelo lógico. Essa foi, efetivamente, a etapa seguinte: começar a estudar as regras de conversão de um modelo conceptual para um modelo lógico. Após termos estudado e interiorizado estas regras, procedemos à conversão de um modelo para o outro. Para a segunda fase deste trabalho, iniciaremos a parte da implementação física, que se revela extremamente importante, pois reflete tudo o que fizemos anteriormente e ajudar-nos-á também a compreender que melhorias poderíamos ter implementado nas duas primeiras fases, de modo a tornar a implementação física mais eficiente e robusta. Em resumo, esta primeira fase do trabalho ajudou-nos a compreender a importância de estudarmos a fundo o problema, para que possamos realizar uma boa modelação do mesmo, a qual, se não for executada corretamente, poderá criar obstáculos nas fases subsequentes do projeto. Concluimos, portanto, que esta é uma das partes mais cruciais de todo o trabalho, pois constitui a base do mesmo.

Na segunda fase deste projeto, o nosso grupo realizou a implementação física da nossa base de dados. Esta implementação permitiu-nos fortalecer os nossos conhecimentos da linguagem *SQL*, através da realização de *queries* e até *scripts* para o povoamento da base de dados. Com isto também voltamos a relembrar a importância da primeira fase deste projeto, pois sem uma boa modelação o nosso trabalho teria sido bem mais arduo. Também foi preciso considerar problemas como o crescimento da base de dados, para garantir a sua viabilidade no futuro. Concluindo, esta fase permitiu-nos aprimorar os nossos conhecimentos e também ter uma "noção" dos problemas ao implementar uma base de dados num contexto real.

Referências

- [1] MySQL. (2018). *MySQL 8.0 Reference Manual*. Disponível em: <https://dev.mysql.com/doc/refman/8.0/en/>

Anexos

```
DELIMITER $$

CREATE PROCEDURE GetClientCases()
BEGIN
    DECLARE clientId INT;
    DECLARE currentUser VARCHAR(100);
    SET currentUser = USER();

    IF currentUser LIKE 'client_user%localhost' THEN
        SET clientId = CAST(SUBSTRING_INDEX(SUBSTRING_INDEX(currentUser, 'client_user', -1), '@', 1) AS UNSIGNED);

        IF clientId > 0 THEN
            SELECT * FROM casos WHERE Cliente = clientId;
        ELSE
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Invalid client ID format';
        END IF;
    ELSE
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Unauthorized access';
    END IF;
END $$

DELIMITER ;
```

Tabela 1: Processo para RC2

```
DELIMITER $$

CREATE PROCEDURE GetClientInfo()
BEGIN
    DECLARE clientId INT;
    DECLARE currentUser VARCHAR(100);
    SET currentUser = USER();

    IF currentUser LIKE 'client_user%localhost' THEN
        SET clientId = CAST(SUBSTRING_INDEX(SUBSTRING_INDEX(currentUser, 'client_user', -1), '@', 1) AS UNSIGNED);

        IF clientId > 0 THEN
            SELECT * FROM clientes WHERE idCliente = clientId;
        ELSE
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Invalid client ID format';
        END IF;
    ELSE
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Unauthorized access';
    END IF;
END $$

DELIMITER ;
```

Tabela 2: Processo para RC6

```

CREATE PROCEDURE UpdateDetectiveInfo(IN name VARCHAR(200), IN email VARCHAR(100))
BEGIN
    DECLARE detectiveId INT;
    DECLARE currentUser VARCHAR(100);
    SET currentUser = USER();

    IF currentUser LIKE 'detective_user%@localhost' THEN
        SET detectiveId = CAST(SUBSTRING_INDEX(SUBSTRING_INDEX(currentUser, 'detective_user', -1), '@', 1) AS UNSIGNED);

        IF detectiveId > 0 THEN
            UPDATE detetives SET Nome = name, email = email WHERE idDetetive = detectiveId;
        ELSE
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Invalid detective ID format';
        END IF;
    ELSE
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Unauthorized access';
    END IF;
END $$

DELIMITER ;

```

Tabela 3: Processo para RC9

Listing 6.1: Povoamento da Base de dados

```
USE 'AgenciaDetetivesSr.Alfredo';

/* Inserindo dados na tabela clientes */
INSERT INTO clientes (nome, sexo, dta_nascimento, email, nif, rua, cod_postal,
VALUES
('João_Silva', 'M', '1980-01-01', 'joao.silva@gmail.com', 123456789, 'Rua_Anto
('Maria_Santos', 'F', '1985-02-02', 'maria.santos@gmail.com', 987654321, 'Rua_
('Carlos_Lima', 'M', '1990-03-03', 'carlos.lima@gmail.com', 112233445, 'Rua_Pe

/* Inserindo dados na tabela cliente_telemoveis */
INSERT INTO cliente_telemoveis (telemovel, idCliente)
VALUES
('912345678', 1),
('923456789', 2),
('934567890', 3);

/* Inserindo dados na tabela tipos */
INSERT INTO tipos (Designacao, descricao)
VALUES
('Investigação_Criminal', 'Casos_relacionados_a_crimes'),
('Busca_e_Vigilância', 'Serviços_de_busca_e_vigilância'),
('Segurança_Pessoal', 'Serviços_de_segurança_pessoal');

/* Inserindo dados na tabela casos */
INSERT INTO casos (estado, custo_agencia, desginacao, descricao, dta_inicio, d
VALUES
('E', 5000, 'Investigação_Roubo', 'Investigação_sobre_roubo_de_joias', '2024-0
('C', 3000, 'Vigilância_Suspeito', 'Vigilância_de_um_suspeito_de_fraude', '202
('T', 7000, 'Segurança_Executiva', 'Serviço_de_segurança_para_executivo', '202

/* Inserindo dados na tabela detetives */
INSERT INTO detetives (Nome, telemovel, email, estado, detetiveChefe)
VALUES
('Alfredo_Sousa', 911223344, 'alfredo.sousa@gmail.com', 'A', 1),
('Miguel_Costa', 922334455, 'miguel.costa@gmail.com', 'A', 1),
('Ana_Pereira', 933445566, 'ana.pereira@gmail.com', 'I', 1);

/* Inserindo dados na tabela caso_detetive */
INSERT INTO caso_detetive (caso, detetive)
VALUES
(1, 1),
(2, 2),
(3, 3);

/* Inserindo dados na tabela evidencias */
INSERT INTO evidencias (designacao, data_coleta, caso, detetive)
```

VALUES

```
('Impressões_Digitais', '2024-01-10', 1, 1),  
( 'Vídeo_de_Vigilância', '2024-02-15', 2, 2),  
( 'Documentos', '2024-03-20', 3, 3);
```

/ Inserindo dados na tabela fotografias */*

INSERT INTO fotografias (fotografia , evidencia)

VALUES

```
('foto1.jpg', 1),  
( 'foto2.jpg', 2),  
( 'foto3.jpg', 3);
```