Universidade do Minho
Licenciatura em Engenharia Informática

Unidade Curricular de Bases de Dados

Ano Lectivo de 2023/2024

<<Trabalho de Bases de Dados

"O último a saber" >>

<<Alexandre Miranda (a104445), Diogo Cunha (a100481), Nuno Aguiar(a100480), Rafael Pereira (a104095) >>

<<Março de 2024>>

Data de Recepção	
Responsável	
Avaliação	
Observações	

<<Agência de Detectives "O Último a Saber">>

<<Grupo 50>>
<<Alexandre Miranda (a104445), Diogo Cunha (a100481), Nuno Aguiar(a100480), Rafael Pereira (a104095) >>

<<Março de 2024>>

Resumo (250 palavras)

No âmbito da disciplina de Bases de Dados realizámos um trabalho de análise e planeamento, no qual nos foi proposto o tema “Agência de Detetives”. Este trabalho envolveu a definição de requisitos e a criação de um modelo conceptual que representa de forma clara e estruturada as entidades, relacionamentos e atributos estabelecidos e um modelo lógico, que delineia a estrutura da base de dados e os domínios e características de cada atributo. Tratando-se apenas de uma primeira fase do projecto final, aquilo que aqui se tem por certo terá ainda de ser aplicado na prática, tendo sempre em mente o potencial aparecimento de dificuldades imprevistas, foi feito um esforço por parte do nosso grupo com vista à concepção de objetivos realistas e praticáveis, evitando que no final o trabalho fique aquém do planeado inicialmente.

Encerrados os trabalhos, podemos dizer convictamente que estamos plenamente satisfeitos com os resultados até ao momento, e confiantes para dar seguimento ao projecto na sua segunda fase.

Palavras-chave (no máximo 6 palavras)
Detetive,SBD relacional, mySQL, Conceptual, Lógico

1 Introdução

1.1 Contextualização

A Agência de Detectives O Último a Saber, conhecida por O.U.S., é a mais antiga empresa do ramo em Portugal, tendo sido fundada em 1810 por ordem Régia de D. João VI, que procurava expor a infidelidade de sua esposa, D. Carlota Joaquina. Actualmente a empresa oferece serviços de investigação privada a clientes de alto-padrão e está sediada em Aveiro. Alguns dos maiores escândalos da vida pública portuguesa foram expostos graças à OUS, tais como a casa milionária de José Sócrates em Paris ou a transferência de Arthur Cabral para o Benfica. Foi decidido, por motivos que serão explicados mais à frente, que era necessário o desenvolvimento de uma base de dados interna, neste âmbito, foi contratada uma equipa de engenheiros para facilitar o seu desenvolvimento.

1.2 Apresentação do Caso de Estudo

Durante uma investigação são produzidas grandes quantidades de documentos e provas fotográficas, até hoje estes são mantidos em arquivos na cave da Agência, mas o presente Director da OUS, José Bond, quer mudar este facto, e, através da implementação de um Sistema de Gestão de Bases de Dados, tornar o negócio mais eficiente, reduzindo a necessidade de espaço físico e ainda o risco de fuga de informação. Permitirá ainda designar agentes a determinados casos nos quais estão envolvidos, protegendo a confidencialidade.

1.3 Motivação e Objectivos

Com esta nova tecnologia, pretende-se:
-Reduzir a necessidade de espaço de armazenamento.

-Aumentar a facilidade de pesquisa e acesso à informação.
-Incremento da segurança das informações confidenciais da empresa.

1.4 Recursos e Equipa de Trabalho

- Humanos

-Funcionários da Agência
-Equipa de Engenheiros de Bases de Dados

- Materiais

- Hardware (um servidor e diversos dispositivos seguros de acessos)
- Software (SGBD, interface utilizador-servidor)

- Pessoal Interno

-José Bond, Tiago Connery, Roger Moita, Pierce Aguiar, Timóteo Dalton e Goretti Moreira, respectivamente Director, Agentes secretos e secretária.

- Pessoal Externo

-Equipa de desenvolvimentos de Bases de Dados, responsáveis pelo levantamento de requisitos, modelação do sistema e sua implementação

1.5 Plano de Execução

Houve uma reunião entre os Engenheiros, os Agentes Secretos e o Director, nesta foram delineados todos os pontos previamente enumerados e foi dada luz verde à realização do projecto. Sendo então definido um diagrama de gant que demonstra o cronograma de execução do projeto.

Planeamento previsto

#	Atividade	Inicio	Duração (dias)	Fim
1	Planeamento	15/02/2024	7	22/02/2024
2	Levantamento de requisitos	23/02/2024	14	8/03/2024
3	Modulação Conceptual	9/03/2024	14	23/03/2024
4	Modulação Lógica	24/03/2024	14	7/04/2024

Atividade	Cor
Planeamento	Azul
Levantamento de requisitos	Vermelho
Modulação Conceptual	Verde
Modulação Lógica	Amarelo

Planeamento executado

#	Atividade	Inicio	Duração (dias)	Fim
1	Planeamento	15/02/2024	7	22/02/2024
2	Levantamento de requisitos	23/02/2024	14	15/03/2024
3	Modulação Conceptual	16/03/2024	14	30/03/2024
4	Modulação Lógica	31/03/2024	14	7/04/2024

	Semana 1	Semana 2	Semana 3	Semana 4
Fevereiro				
Março				
Abril				

	Semana 1	Semana 2	Semana 3	Semana 4
Fevereiro				
Março				
Abril				

Planeamento previsto

#	Atividade	Inicio	Duração (dias)	Fim
1	Criação	15/04/2024	7	22/04/2024
2	Povoamento	23/04/2024	7	30/04/2024
3	Vistas	1/05/2024	10	11/05/2024
4	Procedimentos	12/05/2024	7	19/05/2024
5	Queries	20/05/2024	8	28/05/2024

Atividade	Cor
Criação	Azul
Povoamento	Vermelho
Vistas	Verde
Procedimentos	Amarelo
Queries	Roxo

Planeamento executado

#	Atividade	Inicio	Duração (dias)	Fim
1	Criação	15/04/2024	7	22/04/2024
2	Povoamento	23/04/2024	7	30/04/2024
3	Vistas	1/05/2024	14	15/05/2024
4	Procedimentos	16/05/2024	7	23/05/2024
5	Queries	24/05/2024	4	28/05/2024

	Semana 1	Semana 2	Semana 3	Semana 4
Abril				
Maio				

	Semana 1	Semana 2	Semana 3	Semana 4
Abril				
Maio				

2 Levantamento e Análise de Requisitos

De seguida, deu-se o levantamento e análise dos requisitos considerados mais relevantes para o problema. Estes irão delinear todo o tipo de aspectos pertinentes à operacionalidade do sistema de bases de dados que será implementado.

2.1 Método de levantamento e de análise de requisitos adotado

No sentido de fazer o levantamento dos requisitos para a construção da base de dados, foram efectuadas:
-Reuniões entre a equipa de engenheiros e os representantes da Agência.
-Análise da documentação existente sobre o assunto.
-Investigação sobre o modus operandi de um agente secreto.
-Observação do quotidiano da Agência.

2.2 Organização dos requisitos levantados

2.2.1 Requisitos de descrição

Nº	Data e hora	Descrição	Área/Vista	Fonte	Analista
1	23/2/24 11:24	As investigações devem conter um número de identificação, orçamento de pesquisa, despesas de pesquisa, uma descrição,data de inicio,data de conclusao e um estado conclusão.	Investigações	Reunião	Nuno
2	23/2/24 11:33	A ficha de cada Detetive deve conter informação relativa ao seu nome (nome completo) e ao seu nome de código, data de nascimento, contacto(número de telefone e email), género, data de admissão, casos bem sucedidos e o seu número sequencial (id).	Detetives	Reunião	Alexandre
3	23/2/24 12:01	Um administrador deve ter um número sequencial único (id), o seu nome, data de nascimento, contactos (telefone e email), género e data de admissão.	Diretores	Reunião	Nuno
4	26/2/24 9:47	Um Detetive pode estar a trabalhar em vários casos.	Detetives	Quotidiano	Nuno

5	29/2/24 17:34	Uma investigação pode ter várias provas.	Investigações/ Provas	Quotidiano	Rafael
6	27/2/24 13:58	As provas devem conter uma data de recolha,testemunha, local de recolha e o número de identificação da investigação.	Provas	Documentação	Rafael
7	1/3/24 11:02	Os tipos de provas considerados são, provas físicas(ex: objetos),provas de áudio(ex: gravações de áudio) e provas visuais(ex: fotografias, vídeos)	Provas	Investigação do MO	Alexandre
8	25/2/24 16:44	O cliente é definido por nº contribuinte, nome, data de nascimento, morada, género, telemóvel, email e id de cliente.	Clientes	Documentação	Tomás
9	14/3/24 15:04	Uma investigação deve ter uma data que marca o início da investigação.	Investigações	Documentação	Rafael
10	5/3/24 12:01	Uma investigação pode ser investigada por múltiplos detetives	Investigações/ Detetives	Investigação do MO	Tomás
11	5/3/24 12:47	O agente deve conter uma data de alocação a determinada investigação e uma data de remoção caso ele deixe de trabalhar na mesma.	Investigações/ Detetives	Investigação do MO	Nuno

2.2.2 Requisitos de manipulação

Nº	**Data e hora**	**Descrição**	**Area/Vista**	**Fonte**	**Analista**
1	27/2/24 11:55	A cada momento deve ser possível obter uma lista dos detetives que está a trabalhar em determinado caso	Diretores	Quotidiano	Tomás
2	12/3/24 15:14	Deve ser possível obter o número de casos ativos em que a agência está a trabalhar	Diretores	Investigação do MO	Alexandre
3	5/3/24 9:37	Deve ser possível inserir novos detetives.	Detetives	Documentação	Alexandre
4	26/2/24 16:07	Deve ser possível aceder a todos os casos relacionados a um cliente específico.	Clientes	Quotidiano	Rafael

5	1/3/24 11:17	Deve ser possível inserir novos clientes ou remover antigos(caso pedido).	Clientes	Documentação	Nuno
6	8/3/24 14:15	Deve ser possível alterar o valor das despesas de pesquisa caso haja uma despesa extra que não foi considerada inicialmente	Investigações	Quotidiano	Alexandre
7	27/2/24 13:56	Deve ser possível inserir novas investigações.	Investigações	Documentação	Nuno
8	4/3/24 11:27	Deve ser possível inserir novas provas.	Provas	Investigação do MO	Tomás
9	28/2/24 16:08	Deve ser possível alterar as descrição de uma investigação em curso	Investigações	Investigação do MO	Tomás

2.2.3 Requisitos de controlo

Nº	**Data e hora**	**Descrição**	**Area/Vista**	**Fonte**	**Analista**
1	4/3/24 14:55	Apenas os diretores podem alocar casos a Detetives e removê-los dos mesmos.	Diretores	Reunião	Rafael
2	4/3/24 15:02	Os diretores devem ser os únicos a poder criar e eliminar as fichas dos Detetives	Diretores	Reunião	Tomás
3	4/3/24 15:11	Apenas os diretores e os detetives alocados a cada caso podem ter acesso às informações do mesmo.	Diretores/ Detetives	Reunião	Alexandre
4	4/3/24 15:26	Apenas os Diretores podem aumentar o orçamento de pesquisa de uma determinada investigação	Diretores/ Investigações	Reunião	Nuno
5	4/3/24 15:41	Apenas os diretores podem dar uma investigação como encerrada.	Diretores/ Investigações	Reunião	Nuno
6	4/3/24 15:57	Apenas os Directores podem alterar os detalhes/objectivo de uma investigação	Diretores/ Investigações	Reunião	Rafael

3 Modelação Conceptual

3.1 Apresentação da abordagem de modelação realizada

Decidimos fazer um esquema conceptual global, deixando de lado modelos para as restantes vistas de utilização. Em conformidade com a metodologia de Connoly e Begg seria ideal a realização de diversas vistas e a conciliação de sub-esquemas na fase final do processo de modelação conceptual.

Sendo assim, na parte de modelação lógica ficamos apenas com a revisão e melhoria do modelo. É esperado que a implementação física leve a resultados iguais.

3.2 Identificação e caracterização das entidades

Entidade	Descrição	Aliases	Ocorrência
Cliente	Informação acerca dos clientes.	-	Registo único, pode apenas consultar os dados no final da investigação.
Director	Informação acerca do diretor.	-	Registo único, acesso e poder de alteração total sobre o domínio.
Detetive	Informação acerca dos detetives.	Agente	Registo único, acesso apenas aos casos nos quais trabalha.
Investigação	Informação acerca das investigações.	Caso	Agrupa todos os dados pertinentes à investigação e concentra todos os atores que dela fazem parte.
Provas	Provas ligadas a uma investigação		Informação relativa às provas, estando estas sempre vinculadas a uma determinada investigação.
Visual	Tipo de prova visual, podendo esta ser por exemplo uma imagem ou um vídeo.	-	Ocorre quando a prova recolhida para determinado caso é visual.
Audio	Tipo de prova auditiva, podendo esta ser por exemplo uma gravação de voz.	-	Ocorre quando a prova recolhida para determinado caso é auditiva.
Fisica	Tipo de prova física, podendo esta ser um por exemplo um objeto.	-	Ocorre quando a prova recolhida para determinado caso é física.

3.3 Identificação e caracterização dos relacionamentos

Entidade A	Relacionamento	Descrição	Cardinalidade	Participação	Entidade B
Diretores	dirige	os diretores iniciam dirigem as investigações	1:N	T:T	Investigações
Clientes	requisita	os clientes requisitam as investigações	1:N	T:T	Investigações
Detetives	realiza	os detetives trabalham nas investigações	N:N	T:T	Investigações
Investigações	têm	as investigações têm provas	1:N	T:P	Provas

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

Entidade/Relacionamento	Atributo	Descrição	Tipo de dados e tamanho	Nulo	Multi-valor	Chave primária
Cliente	Nome	Nome do cliente	VARCHAR(75)	N	N	N
	DataNascimento	Data de nascimento do cliente	DATE	N	N	N
	NIF	Número de identificação fiscal do cliente	CHAR(9)	N	N	N
	Género	Género do cliente	CHAR(1)	N	N	N
	Morada	Morada do cliente	VARCHAR(100)	N	N	N
	IDCliente	Número interno único que identifica o cliente	INT	N	N	**S**
	Contacto Email	Contactos do cliente Emails do cliente	VARCHAR(150)	N	**S**	N
	Telefone	Lista de números de telefone	CHAR(12)	N	**S**	N

Diretor	IDDiretor	Número interno único que identifica o diretor	INT	N	N	**S**
	Nome	Nome do diretor	VARCHAR(75)	N	N	N
	DataNascimento	Data de nascimento do diretor	DATE	N	N	N
	DataAdmissão	Data de admissão do diretor	DATE	N	N	N
	Contacto	Contactos do diretor				
	Email	Emails do diretor	VARCHAR(150)	N	**S**	N
	Telefone	Lista de números de telefone	CHAR(12)	N	**S**	N
Detetives	IDDetetive	Número interno único que identifica o detetive	INT	N	N	**S**
	Nome	Nome do detetive	VARCHAR(75)	N	N	N
	DataNascimento	Data de nascimento do detetive	DATE	N	N	N
	Género	Género do detetive	CHAR(1)	N	N	N
	Casos bem sucedidos	Número de casos bem sucedidos do detetive	INT	N	N	N
	Nome de código	Apelido do detetive para garantir a sua segurança	VARCHAR(75)	N	N	N
	Contacto	Contactos do detetive				
	Email	Emails do detetive	VARCHAR(150)	**S**	**S**	N
	Telefone	Lista de números de telefone	CHAR(12)	**S**	**S**	N
Investigações	IDInvestigacao	Número interno único que identifica a investigação	INT	N	N	**S**
	DataInicio	Data em que a investigação teve início	DATE	N	N	N

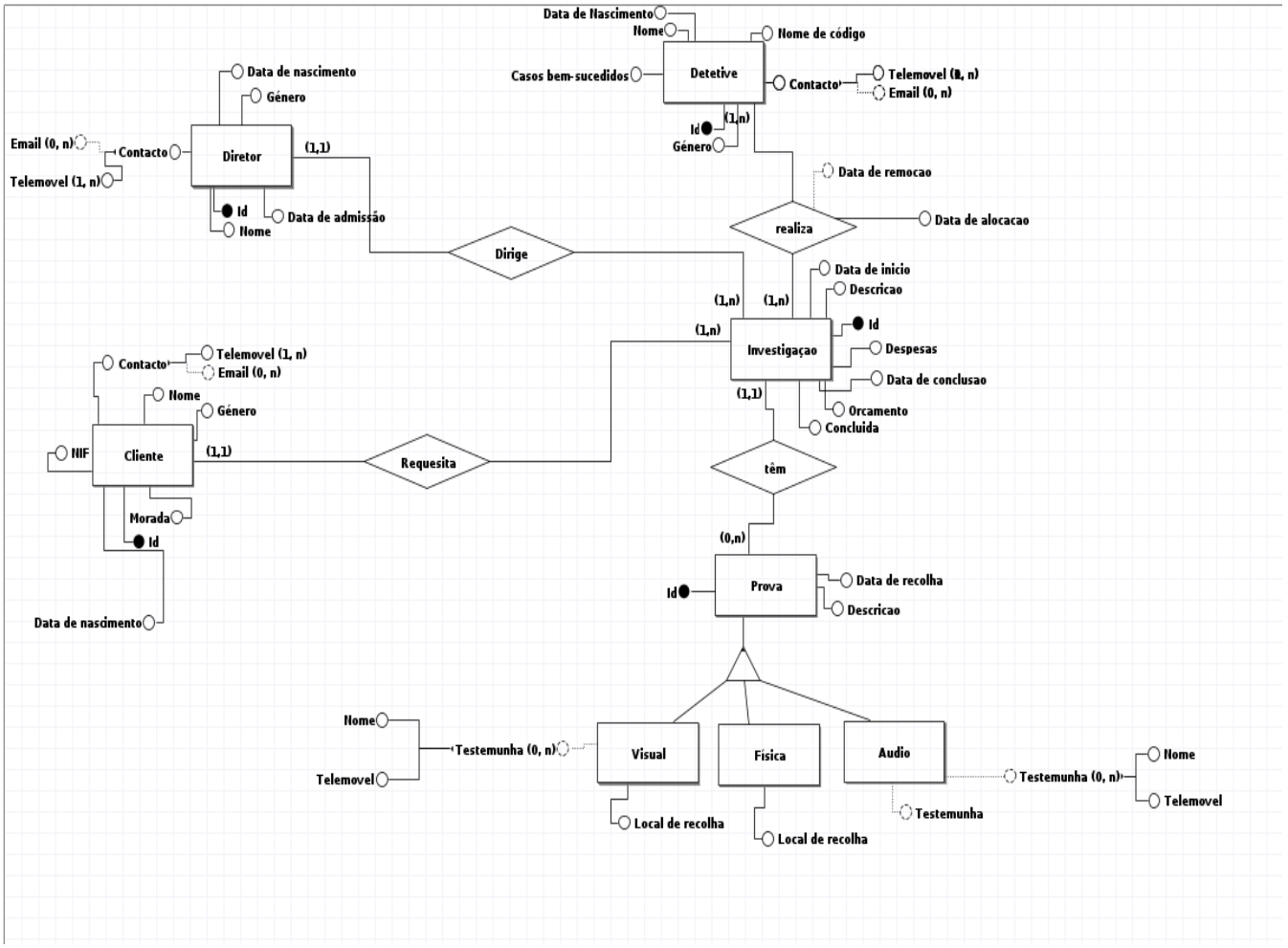
	Despesas	Total de despesas da investigação	DECIMAL(9,2)	N	N	N
	Orçamento	Orçamento inicial da investigação	DECIMAL(9,2)	N	N	N
	Descrição	Observações sobre a investigação	VARCHAR(350)	N	N	N
	Concluida	Valor que indica se a investigação foi concluída.	TINYINT(1)	N	N	N
	DataRemocao	Data em que se concluiu a investigação	DATE	S	N	N
Realiza	DataAlocacao	Data de alocação do detetive à investigação	DATE	N	N	N
	DataRemocao	Data de remoção de um detetive do caso.	DATE	**S**	N	N
Provas	IDProva	Número interno único que identifica uma prova	INT	N	N	**S**
	Data de Recolha	Data de recolha de prova	DATE	N	N	N
	Descrição	Descrição da prova	VARCHAR(500)	N	N	N
Visual	Local de Recolha	Local onde a prova foi recolhida	VARCHAR(100)	N	N	N
	Testemunha	Pessoa ou grupo que testemunhou a prova				
	Nome	Nome da testemunha	VARCHAR(75)	**S**	**S**	N
	Telemóvel	Telemóvel da testemunha	CHAR(12)	**S**	**S**	N
Física	Local de Recolha	Local onde a prova foi recolhida	VARCHAR(100)	N	N	N
Audio	Testemunha	Pessoa que testemunhou a prova				
	Nome	Nome da testemunha	VARCHAR(75)	**S**	**S**	N

	Telefone	Telemóvel da testemunha	CHAR(12)	**S**	**S**	N

3.5 Apresentação e explicação do diagrama ER produzido

Para realizarmos a construção do diagrama Entidade-Relacionamento começamos por criar as entidades e atribuir-lhes os atributos que tínhamos definido nos nossos requisitos,e eventualmente mudando algo,que acreditávamos fazer sentido.
De seguida criamos os relacionamentos,tendo em mente a orientação e o deste modelo(de cima para baixo e da esquerda para a direita),neste passo inicialmente tivemos dúvidas sobre certos aspetos do diagrama,as quais decidimos levar aos docentes.

Depois de obtermos ajuda dos docentes,este é o modelo final.



Ao analisar o modelo algumas coisas podem ser verificadas:

1.Todas as entidades têm uma chave primária,a que chamamos “Id”,é um atributo que terá um valor distinto para todas as instâncias de um entidade,e que será utilizado para as identificar.

2.O relacionamento “realiza” apresenta os atributos,”data de alocação” e “data de remoção”,estes não são propriamente atributos exclusivos da entidade Detetive ,nem da entidade Investigação,por isso aparecem no relacionamento entre os dois(o que apenas é possível porque esta relação é do tipo N:N).

3.A entidade “Prova” apresenta uma característica disjuntiva e obrigatória,isto é, uma prova deve ser obrigatoriamente de um dos tipos:Visual,Física ou Áudio,cada uma destas novas entidades tem também o seu conjunto de atributos próprio.

4.Temos vários tipos de relacionamentos,como:

Cliente:Investigação (1:(1,N))
Um único cliente pode requisitar várias investigações.

Diretor:Investigação (1:(1,N))
A agência tem apenas um diretor,este dirige todas as investigações.

Detetive:Investigação ((1,N):(1,N))
Uma investigação pode ter vários a trabalhar nela e um detetive pode ser alocado a várias investigação.

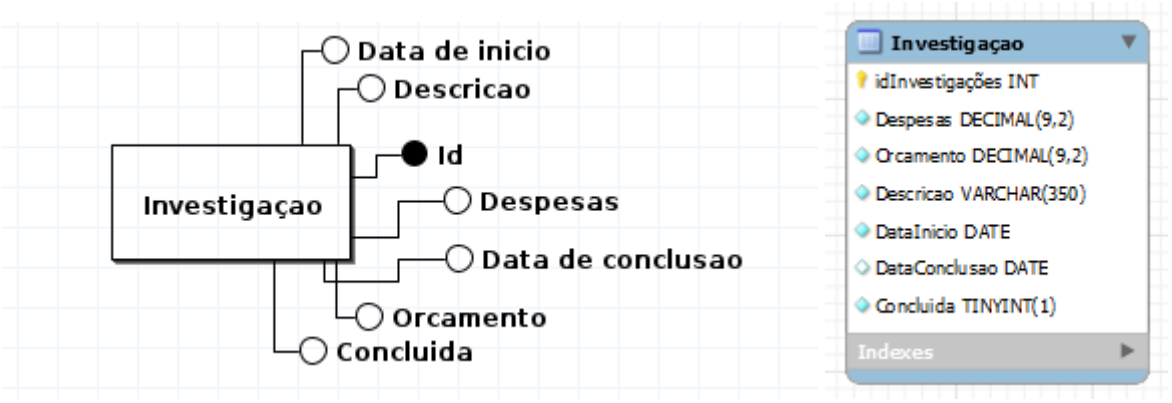
Investigação:Prova (1:(0,N)
Uma investigação pode apresentar várias provas,mas também pode não apresentar nenhuma,por exemplo no início de uma investigação,pode ainda não ter sido encontrada nenhuma prova.

4. Modelação Lógica

4.1 Construção e validação do modelo de dados lógico
(Na criação deste modelo utilizamos mySQL.)
Os atributos da maior parte das entidades foram omitidos por simplicidade.

Entidades - Primeiro, cada entidade foi traduzida para uma tabela de acordo com o nome e o tipo dos seus atributos, especificados anteriormente.

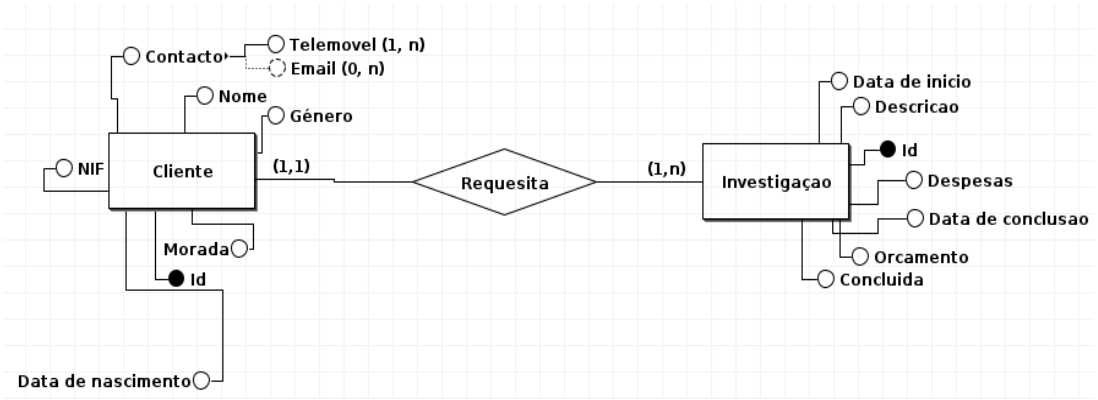
A entidade “Investigação” é representada da seguinte forma no modelo conceptual:



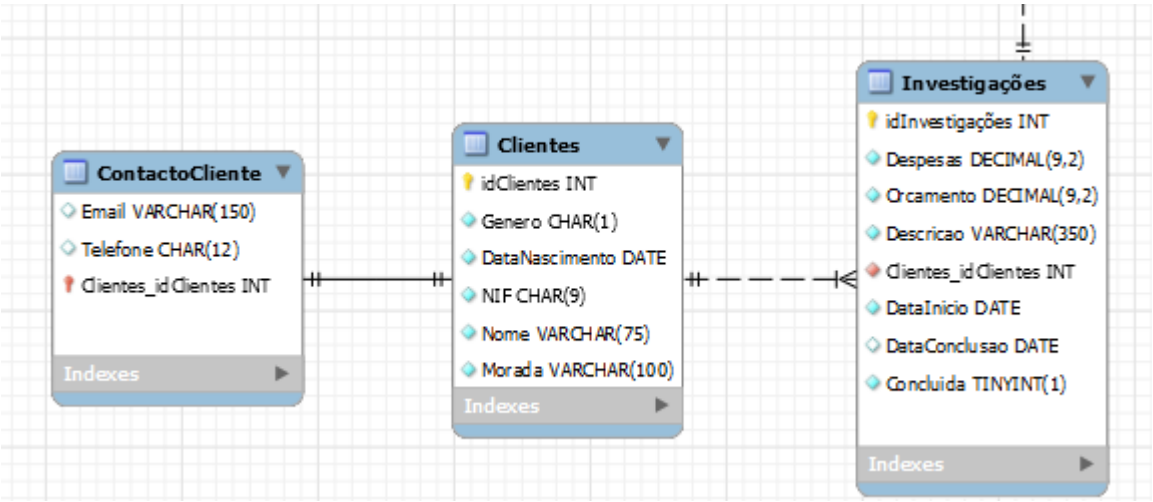
Para convertermos essa entidade na tabela resultante do modelo lógico passámos o id como chave primária e adicionamos os restantes atributos.

Relacionamentos - Depois, os relacionamentos foram traduzidos em novas tabelas e chaves estrangeiras

Devido à cardinalidade 1:N no relacionamento entre as entidades cliente e investigação no modelo conceptual

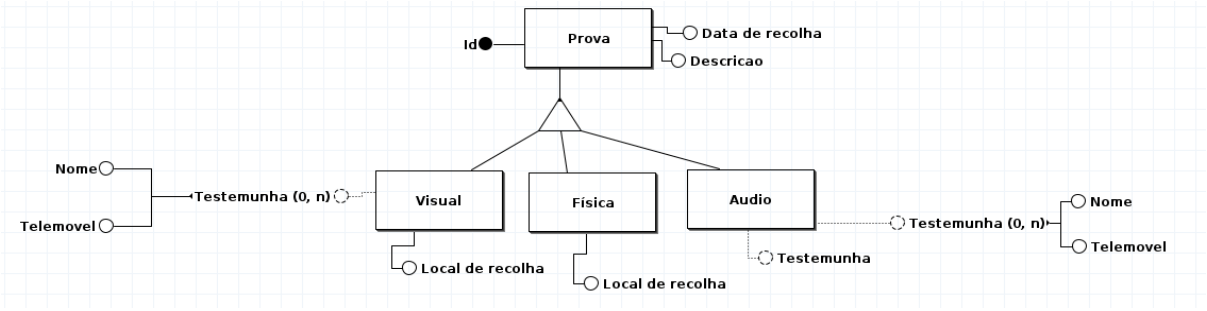


A sua conversão para relacionamento do modelo lógico é a seguinte:

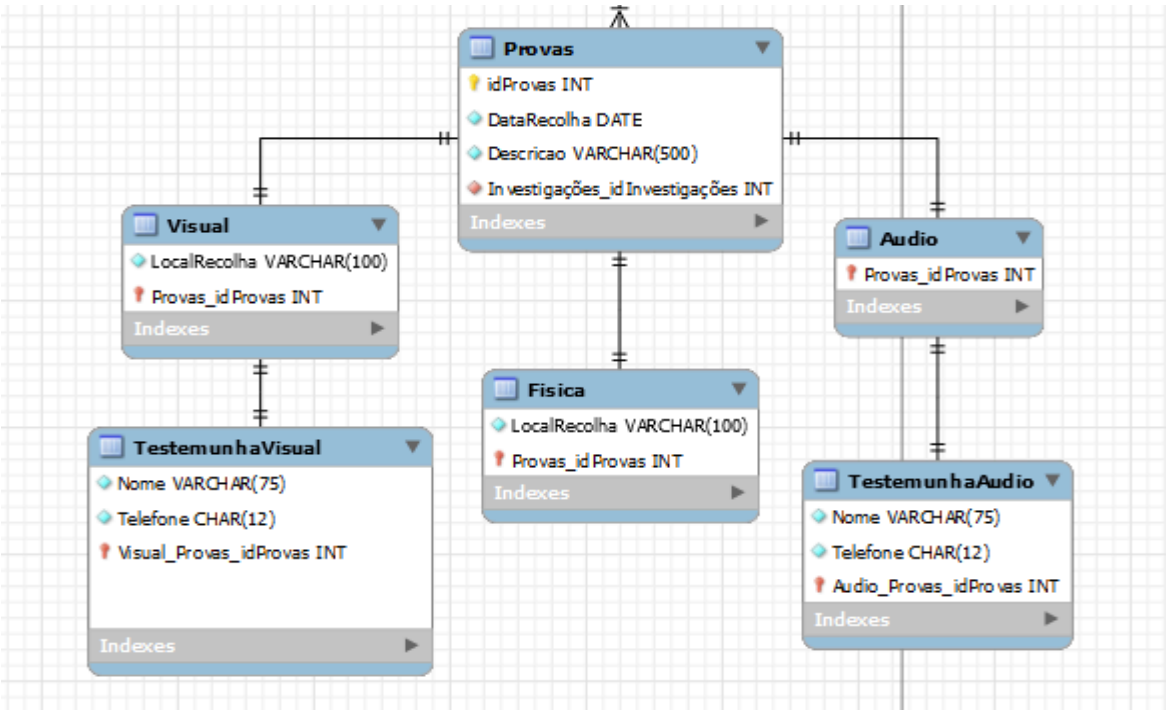


Sendo então chave primária da entidade cliente passada como chave estrangeira para a tabela de investigações.

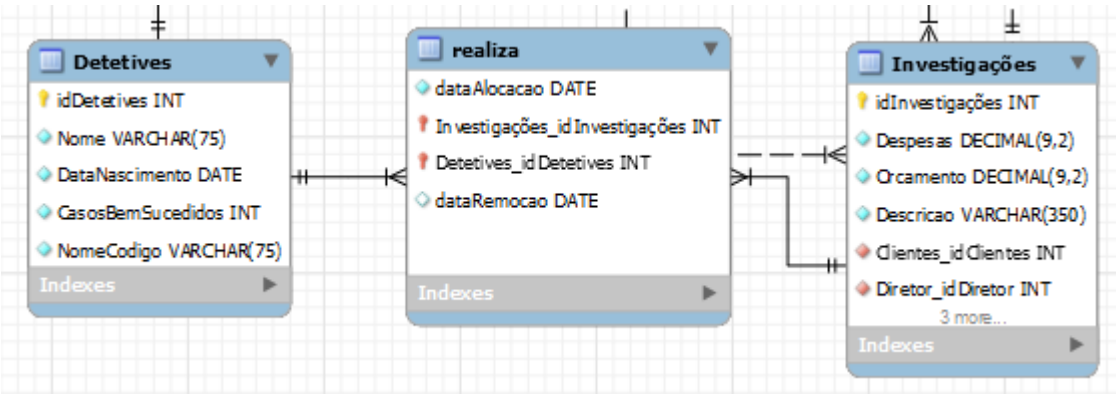
Diferentes tipos de provas:



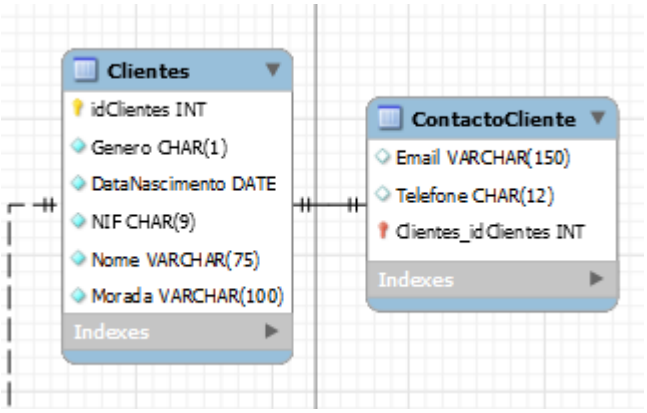
Devido ao relacionamento de superclasse/subclasse entre a entidade Provas e as suas subclasses, foram criadas 4 tabelas. Foi também criada uma tabela para as testemunhas, pois estas são atributos multivalorados.

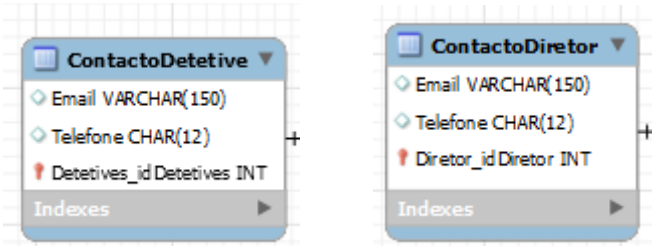


Como o relacionamento entre as entidades Detetives e Investigação é do tipo N:N,foi introduzida uma tabela intermédia,a que chamamos “realiza” sendo então a chave primária da mesma o id_investigação, id_detetive e tendo como chaves estrangeiras o id_investigação e id_detetive sendo assim uma chave estrangeira e primária.

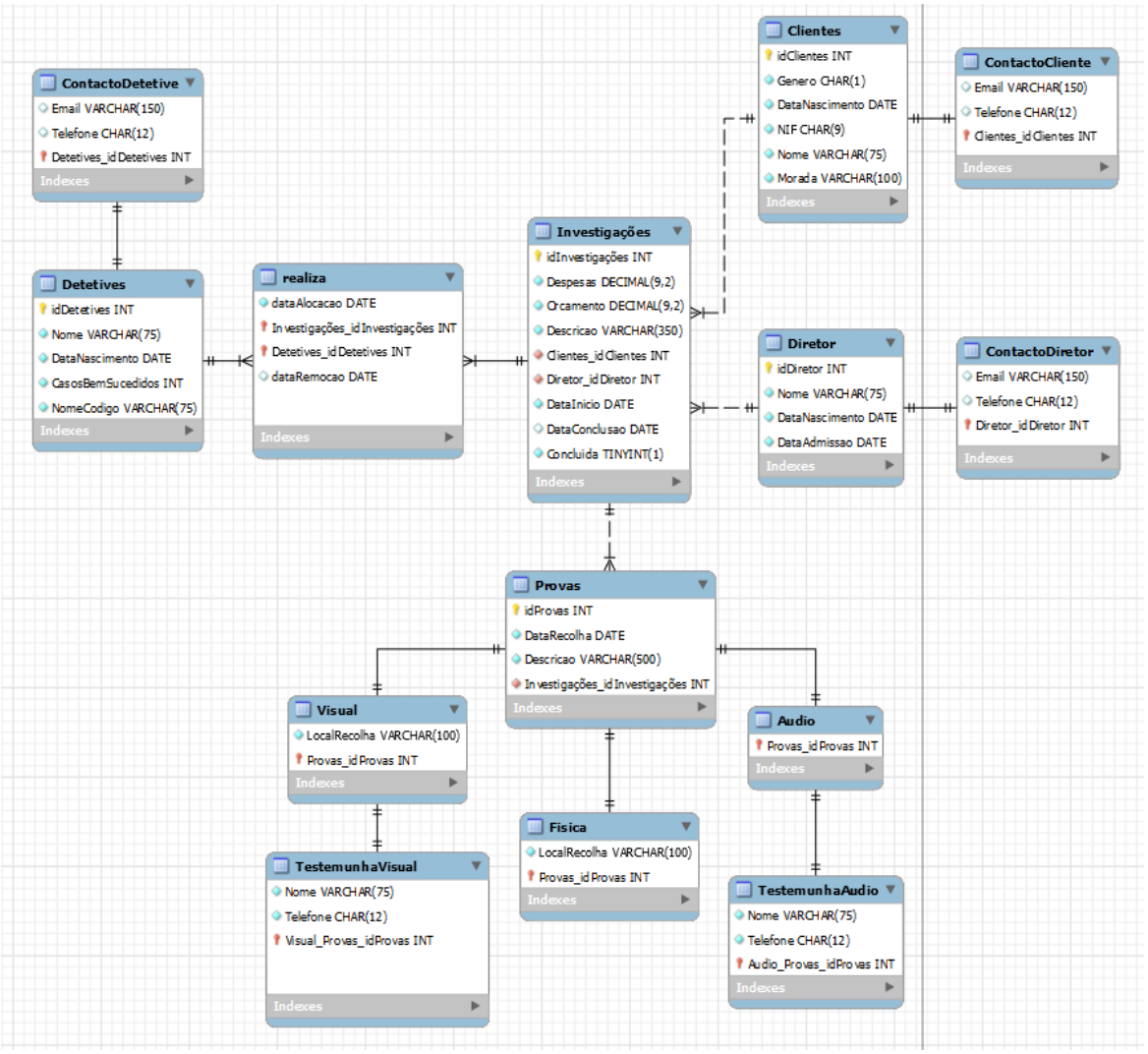


Para os emails e telefones dos clientes,diretor e detetives,foram criadas as seguintes tabelas, onde podemos observar que têm uma chave composta sendo então a chave primária a mesma que a estrangeira devido ao facto de serem atributos multivalorados.





4.2 Apresentação do modelo lógico produzido



4.3 Normalização de Dados

Após análise do modelo lógico,é possível concluir que:

- A primeira forma normal é obedecida, pois não há repetições de dados entre tabelas, todos os atributos de uma determinada tabela têm sempre valores atómicos, não existem grupos de dados repetidos nas tabelas e todas têm chave primária.
- A segunda forma normal é obedecida, pois todos os atributos de todas as tabelas dependem totalmente da chave primária, nunca dependendo apenas de uma parte da mesma
- A terceira forma normal é obedecida, pois não existe nenhum atributo que dependa não só da chave primária como também de outro atributo.

4.4 Validação do Modelo com Interrogações do Utilizador

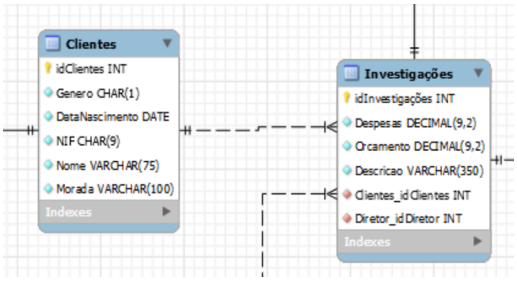
Com base nos requisitos de manipulação, desenvolvemos algumas queries para verificar se as ações de exploração da base de dados podem ou não ser satisfeitas:

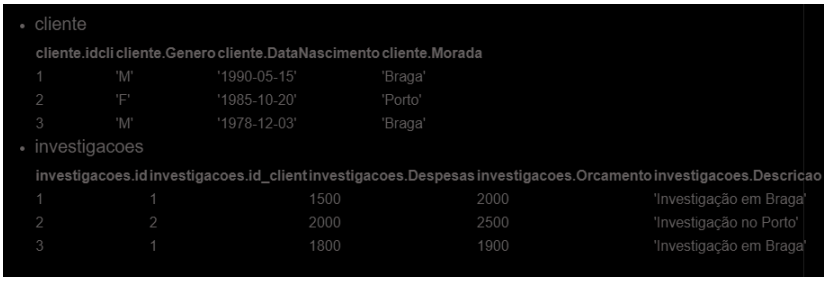
1. Criar uma lista de investigações onde o cliente que as requisitou mora em Braga?

De forma a validar este pedido, teremos de fazer a junção entre as tabelas de ‘investigacoes’ com a tabela dos ‘clientes’ através do id do cliente que é uma chave estrangeira na tabela de ‘investigacoes’ e a chave primária na tabela dos clientes. Depois seria necessário verificar o atributo Morada do cliente para se saber se este reside em Braga. Desta forma a função em Álgebra Relacional seria a seguinte:

π id, Despesas, Orcamento, Descricao σ Morada = 'Braga' (investigacoes ⋈ id_client = idcli cliente)

E podemos observar este pedido em funcionamento nas imagens seguintes(à esquerda as tabelas no modelo lógico relevantes a este pedido, à direita um exemplo das tabelas com os dados e em baixo o resultado expectado):





- cliente

cliente.idcli	cliente.Genero	cliente.DataNascimento	cliente.Morada
1	'M'	'1990-05-15'	'Braga'
2	'F'	'1985-10-20'	'Porto'
3	'M'	'1978-12-03'	'Braga'

- investigacoes

investigacoes.id	investigacoes.id_client	investigacoes.Despesas	investigacoes.Orcamento	investigacoes.Descricao
1	1	1500	2000	'Investigação em Braga'
2	2	2000	2500	'Investigação no Porto'
3	1	1800	1900	'Investigação em Braga'

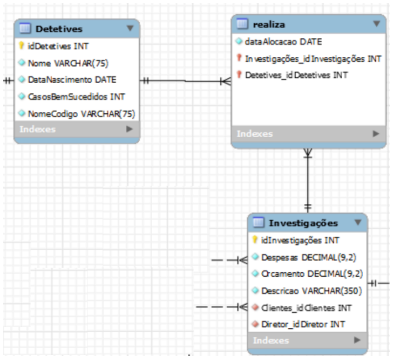
investigacoes.id	investigacoes.Despesas	investigacoes.Orcamento	investigacoes.Descricao
1	1500	2000	'Investigação em Braga'
3	1800	1900	'Investigação em Braga'

2. Criar uma lista dos detetives que começaram a trabalhar na investigação 1 desde o primeiro dia, sendo que estão ordenados do mais novo para o mais velho:

De forma a obter o resultado desta querie seria necessário fazer a junção de 3 tabelas diferentes, sendo estas a tabela do ‘realiza’, a do ‘detetives’ e a da ‘investigacoes’. Ao fazer a junção entre a tabela ‘detetives’ e a tabela ‘realiza’ na chave primária da tabela ‘detetives’ com o atributo chave composta da tabela ‘realiza id_detetives conseguimos obter então todas as informações relativas aos detetives cujo o atributo id_investigacoes que é uma chave composta na tabela ‘realiza’ e é a chave primária na tabela ‘investigacoes’, com este atributo podemos então procurar a investigação com id 1 e verificar a sua data de início

sendo que por fim iremos selecionar os detetives cujo o atributo na tabela ‘realiza’ DataAlocacao é igual ao atributo DataInicio. Por fim ordenamos o resultado pela data de nascimento dos detetives de ordem decrescente. Desta forma a função obtida em Álgebra Relacional seria a seguinte:

τ DataNascimento desc π id_detetives, NomeCodigo, DataNascimento σ DataAlocacao = DataInicio and id_investigacoes_FK = 1 ((detetives $\bowtie$ id_detetives_FK = id_detetives realiza) $\bowtie$ id_investigacoes_FK = id investigacoes)



detetives.id_detetives	detetives.NomeCodigo	detetives.DataNascimento
1	'Maca'	'1980-05-10'
2	'Uva'	'1975-03-15'

3.Buscar todas as provas físicas da investigação com id=2

Para obtermos o resultado desta querie termos que fazer a juncao das tabelas Investigacoes,Provas e Fisica (através das chaves estrangeiras) e “filtrar” as linhas para que apenas aquelas com idInvestigacao=2.

πidInvestigacoes,idProvas,DataRecolha,Provas.Descricao,Fisica.LocalRecolha(σidInvestigacoes=2(Investigacoes$\bowtie_{idInvestigacoes=Investigacoes_idInvestigacoes}$(Provas$\bowtie_{idProvas= Provas_idProvas}$Fisicas)))

4.Detetives que participaram numa investigação com Data de início depois 2023 e com 10 ou mais casos bem sucedidos,ordenados por número decrescente de casos bem sucedidos.

Para obtermos o resultado desta querie termos que fazer a juncao das tabelas Investigacoes,realiza e Detetives(através das chaves estrangeiras) e “filtrar” as linhas para que apenas aquelas com DataInicio>'2022-12-31' e com CasosBemSucedidos>=10 sejam escolhidas e depois sejam ordenadas por ordem decrescente de CasosBemSucedidos.

τCasosBemSucedidos desc(

πidDetetives,NomeCodigo,CasosBemSucedidos,idInvestigacoes,Investigacoes.DataInicio

(σDataInicio>'2022-12-31'∧CasosBemSucedidos>=10(Investigacoes⋈idInvestigacoes=Investigacoes_idInvestigacoes**(realiza⋈idDetetives=Detetives_idDetetives Detetives))))**

5.Detetives que trabalham numa investigação do cliente de id 3

Para obtermos o resultado desta querie termos que fazer a juncao das tabelas Clientes,Investigacoes,realiza e Detetives (através das chaves estrangeiras) e “filtrar” as linhas para que apenas aquelas com idClientes=3 sejam escolhidas.

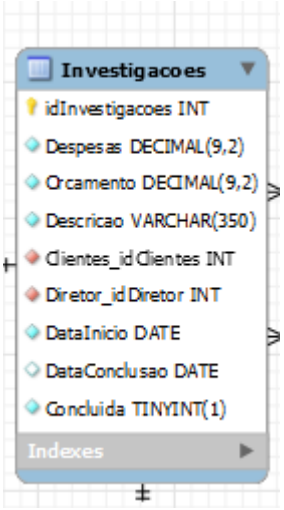
πidDetetives,Detetives.Nome,NomeCodigo,CasosBemSucedidos,idClientes,Clientes.Nome,IdInvestigacoes,Descricao(**σ**idClientes=3(Clientes⋈idClientes=Clientes_idClientes(Investigacoes⋈idInvestigacoes=Investigacoes_idInvestigacoes(realiza⋈idDetetives=Detetives_idDetetives Detetives))))

5. Implementação Física

5.1 Apresentação e explicação da base de dados implementada

A passagem do esquema lógico para o físico teve por base os atributos que cada entidade tinha, referindo o tipo de dados das mesmas e as chaves, tanto primárias como estrangeiras, referenciadas às respectivas tabelas, estabelecendo os relacionamentos entre as diferentes tabelas de dados. Um exemplo disso é a tabela ‘Investigações’ criada conforme o exemplo mencionado abaixo. Nesta tabela, vemos os atributos *(idInvestigações, Despesas, Orçamento, Descrição, Clientes_idClientes, Diretor_idDiretor, DataInicio, DataConclusao, Concluída)* com os seus tipos de dados (ex: INT) e as suas restrições (ex: NOT NULL). Destacam-se também os atributos ‘idInvestigações’ que representa a chave primária desta tabela e as chaves ‘Clientes_idClientes’ e ‘Diretor_idDiretor’ que são chaves estrangeiras nesta tabela e fazem referência às chaves primárias das tabelas mencionadas por elas.

```
CREATE TABLE IF NOT EXISTS `Ultimo_a_Saber`.`Investigacoes` (
  `idInvestigacoes` INT NOT NULL,
  `Despesas` DECIMAL(9,2) NOT NULL,
  `Orcamento` DECIMAL(9,2) NOT NULL,
  `Descricao` VARCHAR(350) NOT NULL,
  `Clientes_idClientes` INT NOT NULL,
  `Diretor_idDiretor` INT NOT NULL,
  `DataInicio` DATE NOT NULL,
  `DataConclusao` DATE NULL,
  `Concluida` TINYINT(1) NOT NULL,
  PRIMARY KEY (`idInvestigacoes`),
  INDEX `fk_Investigações_Clientes1_idx` (`Clientes_idClientes` ASC) VISIBLE,
  INDEX `fk_Investigações_Diretor1_idx` (`Diretor_idDiretor` ASC) VISIBLE,
  CONSTRAINT `fk_Investigações_Clientes1`
    FOREIGN KEY (`Clientes_idClientes`)
    REFERENCES `Ultimo_a_Saber`.`Clientes` (`idClientes`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Investigações_Diretor1`
    FOREIGN KEY (`Diretor_idDiretor`)
    REFERENCES `Ultimo_a_Saber`.`Diretor` (`idDiretor`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```



5.2 Criação de utilizadores da base de dados

Tendo por base os requisitos definidos no início deste projeto foram criados dois utilizadores da base de dados:

1.O diretor

```
use Ultimo_a_Saber;

    -- Diretor Jose Bond
create user 'pauloB'@'localhost' identified by 'Password_2';

```

2.Um detetive

```
use Ultimo_a_Saber;
    -- detetive Rafael Leao
create user 'rafaleao'@'localhost' identified by 'Password_1';
```

De seguida concederam-se as permissões que correspondiam a cada tipo de utilizador:

-O diretor tem acesso para alterar,selecionar,eliminar e atualizar as tabelas Clientes,Investigações,realiza e Detetives.

```
-- O diretor controla a tabela clientes
GRANT select,insert,update,delete
ON Ultimo_a_Saber.Clientes
TO 'pauloB'@'localhost';
FLUSH PRIVILEGES;

-- O diretor controla a tabela Contacto_clientes
GRANT select,insert,update,delete
ON Ultimo_a_Saber.ContactoCliente
TO 'pauloB'@'localhost';
FLUSH PRIVILEGES;

-- O diretor controla a tabela Detetives
GRANT select,insert,update,delete
ON Ultimo_a_Saber.Detetives
TO 'pauloB'@'localhost';
FLUSH PRIVILEGES;
```

```
GRANT select,insert,update,delete
ON Ultimo_a_Saber.realiza
TO 'pauloB'@'localhost';
FLUSH PRIVILEGES;
```

-O diretor apenas pode selecionar as tabelas Provas,Visuais,Audio,Fisica,TestemunhaAudio e TestemunhaVisual.

```
GRANT Select
ON Ultimo_a_Saber.Audio
TO 'pauloB'@'localhost';
FLUSH PRIVILEGES;

GRANT Select
ON Ultimo_a_Saber.TestemunhaAudio
TO 'pauloB'@'localhost';
FLUSH PRIVILEGES;

GRANT Select
ON Ultimo_a_Saber.TestemunhaVisual
TO 'pauloB'@'localhost';
FLUSH PRIVILEGES;
```

```
-- O diretor pode ver a tabela provas,a
GRANT Select
ON Ultimo_a_Saber.Provas
TO 'pauloB'@'localhost';
FLUSH PRIVILEGES;

GRANT Select
ON Ultimo_a_Saber.Fisica
TO 'josebond'@'localhost';
FLUSH PRIVILEGES;

GRANT Select
ON Ultimo_a_Saber.Visual
TO 'pauloB'@'localhost';
FLUSH PRIVILEGES;
```

Já os detetives podem alterar,selecionar,eliminar e atualizar as tabelas Provas,Visuais,Audio,Fisica,TestemunhaAudio e TestemunhaVisual.

```sql
GRANT select,insert,update,delete
ON Ultimo_a_Saber.Provas
TO  'rafaleao'@'localhost';
FLUSH PRIVILEGES;

GRANT select,insert,update,delete
ON Ultimo_a_Saber.Audio
TO  'rafaleao'@'localhost';
FLUSH PRIVILEGES;

GRANT select,insert,update,delete
ON Ultimo_a_Saber.Visual
TO  'rafaleao'@'localhost';
FLUSH PRIVILEGES;
```

```sql
GRANT select,insert,update,delete
ON Ultimo_a_Saber.Fisica
TO  'rafaleao'@'localhost';
FLUSH PRIVILEGES;

GRANT select,insert,update,delete
ON Ultimo_a_Saber.TestemunhaAudio
TO  'rafaleao'@'localhost';
FLUSH PRIVILEGES;

GRANT select,insert,update,delete
ON Ultimo_a_Saber.TestemunhaVisual
TO  'rafaleao'@'localhost';
FLUSH PRIVILEGES;
```

Os detetives apenas podem selecionar as linhas da tabela Investigacoes(excluindo as colunas orçamento e as despesas) que lhes compete.

```sql
-- O detetive pode ver as informaçoes sobre os casos
GRANT SELECT
ON Ultimo_a_Saber.DescricoesInvRafa
TO 'rafaleao'@'localhost';
FLUSH PRIVILEGES;
```

```sql
-- View para o Detetive Rafael Leao
Create view DescricoesInvRafa as
select Investigacoes.idInvestigacoes,Descricao,DataInicio,Concluida,DataConclusao
from Investigacoes inner join (realiza inner join Detetives
on idDetetives=Detetives_idDetetives)
on idInvestigacoes=Investigacoes_idInvestigacoes
where idDetetives=7;
```

5.3 Povoamento da base de dados

De forma a realizar o povoamento da Base de Dados “O Último a Saber” diretamente, realizámos um script em SQL que irá inserir os dados nas respectivas tabelas. Estes serão os dados a ser utilizados ao longo dos testes na Base de Dados.
Como podemos observar pelas imagens que se seguem, executamos um *INSERT INTO* em todas as tabelas criadas.

```sql
CREATE TABLE IF NOT EXISTS `Ultimo_a_Saber`.`Investigacoes` (
  `idInvestigacoes` INT NOT NULL,
  `Despesas` DECIMAL(9,2) NOT NULL,
  `Orcamento` DECIMAL(9,2) NOT NULL,
  `Descricao` VARCHAR(350) NOT NULL,
  `Clientes_idClientes` INT NOT NULL,
  `Diretor_idDiretor` INT NOT NULL,
  `DataInicio` DATE NOT NULL,
  `DataConclusao` DATE NULL,
  `Concluida` TINYINT(1) NOT NULL,
  PRIMARY KEY (`idInvestigacoes`),
  INDEX `fk_Investigações_Clientes1_idx` (`Clientes_idClientes` ASC) VISIBLE,
  INDEX `fk_Investigações_Diretor1_idx` (`Diretor_idDiretor` ASC) VISIBLE,
  CONSTRAINT `fk_Investigações_Clientes1`
    FOREIGN KEY (`Clientes_idClientes`)
    REFERENCES `Ultimo_a_Saber`.`Clientes` (`idClientes`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Investigações_Diretor1`
    FOREIGN KEY (`Diretor_idDiretor`)
    REFERENCES `Ultimo_a_Saber`.`Diretor` (`idDiretor`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```sql
-- Povoar Investigações
INSERT INTO `Ultimo_a_Saber`.`Investigacoes` (`idInvestigacoes`, `Despesas`, `Orcamento`, `Descricao`, `Clientes_idClientes`, `Diretor_idDiretor`, `DataInicio`, `DataConclusao`, `Concluida`)
VALUES
(1, 500.00, 1000.00, 'Investigação de Pessoa Desaparecida - Clínica Navarro', 1, 1, '2024-03-15', NULL, 0),
(2, 1000.00, 2000.00, 'Investigação de Fraude Financeira - Simão Navarro', 2, 1, '2024-04-20', NULL, 0),
(3, 800.00, 1500.00, 'Investigação de Homicídio - Ponte de Lima', 3, 1, '2024-05-10', NULL, 0),
(4, 600.00, 1200.00, 'Investigação de Pessoa Desaparecida - Colégio da Barra', 4, 1, '2024-06-05', NULL, 0),
(5, 1200.00, 2500.00, 'Investigação de Homicídio - Francisca', 5, 1, '2024-07-12', NULL, 0);
```

Para inserir os dados com um programa decidimos utilizar a linguagem python. O código que realizámos tinha como objetivo adicionar dados tanto via ficheiro como via input direto, mas tivemos dificuldades a adicionar os dados lidos pelo csv parser nas tabelas correspondentes. Dado isto o programa que implementámos apenas nos permite adicionar os dados via input direto, como podemos observar:

```
Input from file? (y/n): n
Enter the table name: clientes
Parameters:
Nome            Tipo          Pode ser null
--------------  ------------  ---------------
Genero          char(1)       NO
DataNascimento  date          NO
NIF             char(9)       NO
Nome            varchar(75)   NO
Morada          varchar(100)  NO
Enter value for Genero: M
Enter value for DataNascimento: 2003-01-11
Enter value for NIF: 654123
Enter value for Nome: Nuno
Enter value for Morada: Rua Braga, Braga
Commit? (y/n): y
Input from file? (y/n): n
Enter the table name: ContactoCliente
Parameters:
Nome                Tipo          Pode ser null
------------------  ------------  ---------------
Email               varchar(150)  YES
Telefone            char(12)      YES
Clientes_idClientes  int          NO
Enter value for Email: nuno@email.com
Enter value for Telefone: 99119921
Enter value for Clientes_idClientes: 6
Commit? (y/n): y
```

idClientes	Genero	DataNascimento	NIF	Nome	Morada	Email	Telefone
1	M	1980-05-12	123456789	Ana Silva	Rua da Amizade, Lisboa	ana.silva@example.com	123456789
2	F	1975-09-20	987654321	José Santos	Avenida Central, Porto	jose.santos@example.com	987654321
3	F	1992-03-10	456789123	Manuela Costa	Travessa das Flores, Braga	manuela.costa@example.com	456789123
4	M	1988-07-08	321654987	Miguel Ferreira	Rua das Oliveiras, Coimbra	miguel.ferreira@example.com	321654987
5	F	1985-11-25	789123456	Sofia Martins	Praceta dos Girassóis, Faro	sofia.martins@example.com	789123456
6	M	2003-01-11	654123	Nuno	Rua Braga, Braga	nuno@email.com	99119921

5.4 Cálculo do espaço da base de dados

Recorremos ao mySQL Workbench para estimar o tamanho inicial da base de dados. Na aba SCHEMAS, clicamos com o botão direito em cima da nossa BD 'O Último a Saber' e selecionamos a opção "Schema Inspector". Concluímos que o tamanho base da Schema da nossa BD era de aproximadamente 336.0 KiB. Criámos um script em que é calculado o tamanho em bytes e Kib de cada tabela, e comprovamos que o valor final era o mesmo, mais especificamente 344064 Bytes. Em seguida mostram-se os valores descritos e o código SQL desse script:

```sql
SELECT
  TABLE_NAME AS `Table`,
  DATA_LENGTH AS `Data_length`, -- tamanho dos dados em cada tabela (em bytes)
  INDEX_LENGTH AS `Index_length`, -- tamanho do ficheiro de indíce para a tabela (em bytes)
  (DATA_LENGTH + INDEX_LENGTH) AS `Size (Bytes)`,
  ROUND((data_length + index_length) / 1024, 3) AS `Size (KiB)`
FROM
  information_schema.TABLES
WHERE
  TABLE_SCHEMA = "ultimo_a_saber"
UNION
SELECT
    'TOTAL',
    SUM(data_length),
    SUM(index_length),
    round(SUM(data_length + index_length)),
    ROUND((SUM(data_length) + SUM(index_length)) / 1024, 3)
FROM
  information_schema.TABLES
WHERE
  TABLE_SCHEMA = "ultimo_a_saber"
ORDER BY (DATA_LENGTH + INDEX_LENGTH) DESC
```

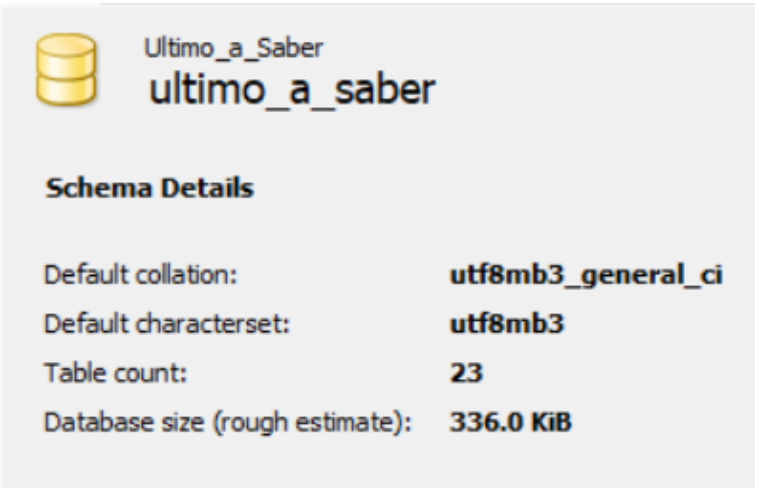


Table	Data_length	Index_length	Size (Bytes)	Size (KiB)
TOTAL	229376	114688	344064	336.000
investigacoes	16384	32768	49152	48.000
contactocliente	16384	16384	32768	32.000
contactodetetive	16384	16384	32768	32.000
contactodiretor	16384	16384	32768	32.000
provas	16384	16384	32768	32.000
realiza	16384	16384	32768	32.000
audio	16384	0	16384	16.000
clientes	16384	0	16384	16.000
detetives	16384	0	16384	16.000
diretor	16384	0	16384	16.000
fisica	16384	0	16384	16.000
testemunhaaudio	16384	0	16384	16.000
testemunhavisual	16384	0	16384	16.000
visual	16384	0	16384	16.000

5.5 Definição e caracterização de vistas de utilização em SQL

De forma a simplificar a perceção que um utilizador têm sobre a base de dados, foram criadas diversas vistas de utilização para as áreas necessárias de acordo com os requisitos previamente estabelecidos. Para tal foram criadas as vistas ‘DetetivesInvestigacaoId’ sendo que o id é o identificador da investigação pretendida. Irá existir uma vista como esta para cada investigação.

```
-- Ver detetives na Investigação 1
CREATE VIEW `Ultimo_a_Saber`.`DetetivesInvestigacao1` AS
SELECT i.Descricao, i.DataInicio, r.DataAlocacao,
CASE
    WHEN r.dataRemocao IS NOT NULL THEN r.dataRemocao
    ELSE 'Ainda Ativo'
END AS DataRemocao,
CASE i.concluida
    WHEN 1 THEN 'Sim'
    ELSE 'Não'
END AS Concluida,
d.IdDetetives, d.Nome, d.NomeCodigo, d.CasosBemSucedidos
FROM `Ultimo_a_Saber`.`Detetives` AS d
INNER JOIN `Ultimo_a_Saber`.`Realiza` AS r
ON d.idDetetives = r.Detetives_idDetetives
INNER JOIN `Ultimo_a_Saber`.`investigacoes` AS i
ON i.idInvestigacoes = r.Investigacoes_idInvestigacoes
WHERE r.Investigacoes_idInvestigacoes = 1;
```

Foram também criadas vistas para ser possível ver todas as investigações, todos os detetives que trabalham na agência e de todos os clientes.

```sql
-- Criar vista dos Clientes e os seus contactos
CREATE VIEW `Ultimo_a_Saber`.`ClientesView` AS
SELECT cli.*, c.Email, c.Telefone
FROM `Ultimo_a_Saber`.`Clientes` AS cli
INNER JOIN `Ultimo_a_Saber`.`ContactoCliente` AS c
ON cli.idClientes = c.Clientes_idClientes;
```

```sql
-- Criar vista dos Detetives e os seus contactos
CREATE VIEW `Ultimo_a_Saber`.`DetetivesView` AS
SELECT d.*, c.Email, c.Telefone
FROM `Ultimo_a_Saber`.`Detetives` AS d
INNER JOIN `Ultimo_a_Saber`.`ContactoDetetive` AS c
ON d.idDetetives = c.Detetives_idDetetives;
```

```sql
-- Criar vista de todas as Investigações
CREATE VIEW `Ultimo_a_Saber`.`InvestigacoesView` AS
SELECT *
FROM `Ultimo_a_Saber`.`Investigacoes`;
```

idInvestigacoes	Despesas	Orcamento	Descricao	Clientes_idClientes	Diretor_idDiretor	DataInicio	DataConclusao	Concluida
1	500.00	1000.00	Investigação de Pessoa Desaparecida - Clínica ...	1	1	2024-03-15	2024-05-27	1
2	1000.00	2000.00	Investigação de Fraude Financeira - Simão Nav...	2	1	2024-04-20	NULL	0
3	800.00	1500.00	Investigação de Homicídio - Ponte de Lima	3	1	2024-05-10	NULL	0
4	600.00	1200.00	Investigação de Pessoa Desaparecida - Colégio ...	4	1	2024-06-05	NULL	0
5	1200.00	2500.00	Investigação de Homicídio - Francisca	5	1	2024-07-12	NULL	0

5.6 Tradução das interrogações do utilizador para SQL

Procedendo à implementação das interrogações do utilizador para SQL, desenvolvemos a tradução a partir das queries que tínhamos traduzido previamente em álgebra relacional. De seguida apresentam-se as soluções das mesmas:

1. Criar uma lista de investigações onde o cliente que as requisitou mora em Braga.

```sql
-- Listar as investigações onde o cliente mora em Braga
SELECT i.idInvestigacoes, i.despesas, i.orcamento, i.descricao, i.datainicio, i.dataconclusao, c.morada
FROM `Ultimo_a_Saber`.`investigacoes` AS i
INNER JOIN  `Ultimo_a_Saber`.`clientes` AS c
WHERE c.morada LIKE "%, Braga";
```

idInvestigacoes	despesas	orcamento	descricao	datainicio	dataconclusao	morada
1	500.00	1000.00	Investigação de Pessoa Desaparecida - Clínica ...	2024-03-15	2024-05-28	Travessa das Flores, Braga
2	1000.00	2000.00	Investigação de Fraude Financeira - Simão Nav...	2024-04-20	NULL	Travessa das Flores, Braga
3	800.00	1500.00	Investigação de Homicídio - Ponte de Lima	2024-05-10	NULL	Travessa das Flores, Braga
4	600.00	1200.00	Investigação de Pessoa Desaparecida - Colégio ...	2024-06-05	NULL	Travessa das Flores, Braga
5	1200.00	2500.00	Investigação de Homicídio - Francisca	2024-07-12	NULL	Travessa das Flores, Braga

2. Criar uma lista dos detetives que começaram a trabalhar na investigação 1 desde o primeiro dia, sendo que estão ordenados do mais novo para o mais velho.

idDetetives	Nome	DataNascimento	CasosBemSucedidos	NomeCodigo
1	José Bernardo	1990-05-15	10	Maça
2	Sofio Santos	1985-09-20	15	Banana

```sql
-- Criar uma lista dos detetives que começaram a trabalhar na investigação 1 desde o primeiro dia
-- Ordenados do mais novo para o mais velho

SELECT d.*
FROM `Ultimo_a_Saber`.`detetives` AS d
INNER JOIN `Ultimo_a_Saber`.`realiza` AS r
ON d.idDetetives = r.Detetives_idDetetives
INNER JOIN `Ultimo_a_Saber`.`investigacoes` AS i
ON i.idInvestigacoes = r.Investigacoes_idInvestigacoes
WHERE i.idInvestigacoes = 1 AND r.dataAlocacao = i.DataInicio
ORDER BY d.dataNascimento DESC;
```

3.Buscar todas as provas físicas da investigação com id=2

```sql
SELECT i.idInvestigacoes, i.Descricao, p.idProvas,  p.Descricao, p.DataRecolha, f.LocalRecolha
FROM `Ultimo_a_Saber`.`investigacoes` AS i
INNER JOIN  `Ultimo_a_Saber`.`provas` AS p
ON i.idInvestigacoes = p.Investigacoes_idInvestigacoes
INNER JOIN `Ultimo_a_Saber`.`fisica` AS f
ON p.idProvas = f.Provas_idProvas
WHERE idInvestigacoes=2;
```

idInvestigacoes	Descricao	idProvas	Descricao	DataRecolha	LocalRecolha
2	Investigação de Fraude Financeira - Simão Nav...	7	Equipamento de espionagem encontrado.	2023-07-20	Local do Crime, Porto

4.Detetives que participaram numa investigação com Data de início depois 2023 e com 10 ou mais casos bem sucedidos,ordenados por número decrescente de casos bem sucedidos.

```sql
SELECT d.idDetetives, d.Nome, d.NomeCodigo, d.CasosBemSucedidos, i.idInvestigacoes
FROM `Ultimo_a_Saber`.`investigacoes` AS i
INNER JOIN `Ultimo_a_Saber`.`realiza` AS r
ON i.idInvestigacoes = r.Investigacoes_idInvestigacoes
INNER JOIN `Ultimo_a_Saber`.`detetives` AS d
ON d.idDetetives= r.Detetives_idDetetives
WHERE i.DataInicio > '2022-12-31' And d.CasosBemSucedidos >= 10
ORDER BY d.CasosBemSucedidos DESC;
```

idDetetives	Nome	NomeCodigo	CasosBemSucedidos
8	Darwin Nuñez	Flop	20
5	Nuno Aveirense	Beira-Mar	18
10	Lisa Ann	Ananás	17
2	Sofio Santos	Banana	15
6	Miguel de Bragança	Pêssego	14
3	Ze Manel o taxista	Taxi	12
7	Rafael Leão	RL10	11
1	José Bernardo	Maça	10

5.Detetives que trabalham numa investigação do cliente de id 3

```
SELECT c.idClientes, c.Nome, d.idDetetives, d.Nome, d.NomeCodigo, d.CasosBemSucedidos,  i.idInvestigacoes, i.Descricao
FROM `Ultimo_a_Saber`.`Clientes` AS c
INNER JOIN `Ultimo_a_Saber`.`investigacoes` AS i
ON c.idClientes = i.Clientes_idClientes
INNER JOIN `Ultimo_a_Saber`.`realiza` AS r
ON i.idInvestigacoes = r.Investigacoes_idInvestigacoes
INNER JOIN `Ultimo_a_Saber`.`detetives` AS d
ON d.idDetetives = r.Detetives_idDetetives
WHERE c.idClientes=3;
```

idClientes	Nome	idDetetives	Nome	NomeCodigo	CasosBemSucedidos	idInvestigacoes	Descricao
3	Manuela Costa	5	Nuno Aveirense	Beira-Mar	18	3	Investigação de Homicídio - Ponte de Lima
3	Manuela Costa	6	Miguel de Bragança	Pêssego	14	3	Investigação de Homicídio - Ponte de Lima

5.7 Indexação do Sistema de Dados

Para acelerar procuras em tabelas que são frequentemente acedidas foi usada a criação de índices de forma automática do forward engineering para chaves estrangeiras das várias tabelas. Por exemplo, na tabela 'realiza' que aparece abaixo, é criado um índice 'fk_realiza2_Detetives1_idx' que otimiza a pesquisa de Detetives ordenando pelo ID de Detetive e também é criado um índice 'fk_realiza2_Investigacoes1_idx' que otimiza a pesquisa de Investigações ordenando pelo ID de Investigações.

```
CREATE TABLE IF NOT EXISTS `Ultimo_a_Saber`.`realiza` (
  `dataAlocacao` DATE NOT NULL,
  `Investigacoes_idInvestigacoes` INT NOT NULL,
  `Detetives_idDetetives` INT NOT NULL,
  `dataRemocao` DATE NULL,
  PRIMARY KEY (`Investigacoes_idInvestigacoes`, `Detetives_idDetetives`),
  INDEX `fk_realiza2_Detetives1_idx` (`Detetives_idDetetives` ASC) VISIBLE,
  INDEX `fk_realiza2_Investigacoes1_idx` (`Investigacoes_idInvestigacoes` ASC) VISIBLE,
  CONSTRAINT `fk_realiza2_Investigações1`
    FOREIGN KEY (`Investigacoes_idInvestigacoes`)
    REFERENCES `Ultimo_a_Saber`.`Investigacoes` (`idInvestigacoes`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_realiza2_Detetives1`
    FOREIGN KEY (`Detetives_idDetetives`)
    REFERENCES `Ultimo_a_Saber`.`Detetives` (`idDetetives`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

5.8 Implementação de procedimentos, funções e gatilhos

De forma a facilitar o uso de funções recorrentes, implementámos diversos procedimentos, funções e gatilhos que consideramos necessários.

Relativamente aos procedimentos e com base nos requisitos, definimos os seguintes:

- adicionarDetetiveAInvestigacao: Este procedimento necessita de um id do detetive que queremos adicionar e do id da investigação ao qual o iremos adicionar. De seguida é criada uma entrada na tabela realiza com os ids que recebe e marca a dataAlocacao com a data do dia em que foi adicionado.

```
DELIMITER $$
CREATE PROCEDURE adicionarDetetiveAInvestigacao(
    IN idDetective INT,
    IN idInvestigacao INT
)
BEGIN
    INSERT INTO `Ultimo_a_Saber`.`realiza` (`dataAlocacao`, `Investigacoes_idInvestigacoes`, `Detetives_idDetetives`)
    VALUES (CURDATE(), idInvestigacao, idDetective);
END $$
DELIMITER ;
```

- removerDetetiveInvestigacao: Este procedimento é utilizado para remover um determinado detetive de uma determinada investigação, tal como o anterior é necessário receber o id de ambos. De seguida atualiza a data de remoção para a data atual sendo que futuramente qualquer detetive que contenha uma data de remoção não é considerado como ativo na investigação.

```
DELIMITER $$
CREATE PROCEDURE removerDetetiveInvestigacao(
    IN idDetective INT,
    IN idInvestigacao INT
)
BEGIN
    UPDATE `Ultimo_a_Saber`.`realiza`
    SET `dataRemocao` = CURDATE()
    WHERE `Detetives_idDetetives` = idDetective
    AND `Investigacoes_idInvestigacoes` = idInvestigacao;
END $$
DELIMITER ;
```

- adicionarProvaInvestigacao: Este procedimento recebe as informações necessárias que constituem uma prova e de seguida irá inseri-las na tabela Prova. Receberá também um tipo que especifica que tipo de prova é, para poder inserir os dados na tabela correta.

```
DELIMITER $$
CREATE PROCEDURE adicionarProvaInvestigacao(
    IN descricao VARCHAR(500),
    IN investigacao_id INT,
    IN local_de_recolha VARCHAR(100), -- pode ser null
    IN tipo INT -- tipo = 1 é visual tipo = 0 é fisica
)
BEGIN
    DECLARE data_recolha DATE;
    DECLARE prova_id INT;

    SET data_recolha = CURDATE();

    INSERT INTO `Ultimo_a_Saber`.`Provas` (DataRecolha, Descricao, Investigacoes_idInvestigacoes)
    VALUES (data_recolha, descricao, investigacao_id);

    SET prova_id = LAST_INSERT_ID();

    IF local_de_recolha IS NULL THEN
        INSERT INTO `Ultimo_a_Saber`.`Audio` (Provas_idProvas)
        VALUES (prova_id);
    ELSEIF tipo = 1 THEN
        INSERT INTO `Ultimo_a_Saber`.`Visual` (Provas_idProvas, LocalRecolha)
        VALUES (prova_id, local_de_recolha);
    ELSE
        INSERT INTO `Ultimo_a_Saber`.`Fisica` (Provas_idProvas, LocalRecolha)
        VALUES (prova_id, local_de_recolha);
    END IF;
END $$
DELIMITER ;
```

- concluirInvestigacao: Este procedimento recebe um id de uma determinada investigação, muda o estado da mesma para concluido (1 = concluido, 0 = ativa) e define a data de conclusão com a data atual.

```
DELIMITER $$
CREATE PROCEDURE concluirInvestigacao(
    IN id_Investigacao INT
)
BEGIN
    UPDATE `Ultimo_a_Saber`.`investigacoes`
    SET concluida = 1, dataConclusao = CURDATE()
    WHERE idInvestigacoes = id_Investigacao;
END $$
DELIMITER ;
```

Relativamente aos gatilhos foi definido apenas um:

- atualizarDetetivesTrigger: Este gatilho tem como objetivo atualizar todos os detetives que estão a trabalhar numa determinada investigação. Quando a

mesma é concluída este gatilho é ativado e irá percorrer na tabela do realiza com o intuito de encontrar todos os detetives que trabalham na investigação que encerrou e ainda não foram removidos da mesma(dataRemocao = null). Assim este gatilho ativa-se sempre que o atributo “concluída” de uma determinada entrada muda de 0 para 1.

```
DELIMITER $$
CREATE TRIGGER atualizarDetetivesTrigger AFTER UPDATE ON `Ultimo_a_Saber`.`Investigacoes`
FOR EACH ROW
BEGIN
    IF NEW.Concluida = 1 THEN
        UPDATE `Ultimo_a_Saber`.`realiza`
        SET `dataRemocao` = IF(`dataRemocao` IS NULL, CURDATE(), `dataRemocao`)
        WHERE `Investigacoes_idInvestigacoes` = NEW.idInvestigacoes;
    END IF;
END $$
DELIMITER ;
```

Relativamente às funções, foram definidas as seguintes:

- calcularDuracaoInvestigacao: Esta função recebe o id de uma determinada investigação e calcula o número de dias que esta investigação durou/dura. No caso de a data de conclusão ser null é utilizada a data atual como data final.

```
DELIMITER $$
CREATE FUNCTION `Ultimo_a_Saber`.calcularDuracaoInvestigacao(idInvestigacao INT) RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE duracao INT;
    DECLARE inicio DATE;
    DECLARE fim DATE;

    SELECT dataInicio INTO inicio FROM `Ultimo_a_Saber`.`investigacoes` WHERE idInvestigacoes = idInvestigacao;
    SELECT dataConclusao INTO fim FROM `Ultimo_a_Saber`.`investigacoes` WHERE idInvestigacoes = idInvestigacao;

     IF inicio IS NULL THEN
        return -1; -- If dataInicio is NULL, return NULL
    END IF;

    IF fim IS NOT NULL THEN
        SET duracao = DATEDIFF(fim, inicio);
    ELSE
        SET fim = CURDATE();
        SET duracao = DATEDIFF(fim, inicio);
    END IF;

    RETURN duracao;
END $$
DELIMITER ;
```

- contaInvestigacoesAtivasDetetive: dado o id de um detetive conta o número de investigações em que este está a trabalhar ativamente.Ou seja a sua dataRemocao na tabela realiza é null.

```
DELIMITER $$
CREATE FUNCTION `Ultimo_a_Saber`.contaInvestigacoesAtivasDetetive(detective_id INT) RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE nInvestigacoes INT;
    SELECT COUNT(*)
    INTO nInvestigacoes
    FROM `Ultimo_a_Saber`.`Realiza` AS r
    INNER JOIN `Ultimo_a_Saber`.`investigacoes` AS i
    ON i.idInvestigacoes = r.Investigacoes_idInvestigacoes
    WHERE r.Detetives_idDetetives = detective_id
    AND (r.dataRemocao IS NULL AND i.concluida = 0);

    RETURN nInvestigacoes;
END $$
DELIMITER ;
```

- contaInvestigacoesAtivas: Esta função conta todas as investigações ativas na base de dados da agência. Para tal verifica se a dataConclusao ainda é null se for conta essa investigação.

```
DELIMITER $$
CREATE FUNCTION `Ultimo_a_Saber`.contaInvestigacoesAtivas() RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE nInvestigacoes INT;
    SELECT COUNT(*)
    INTO nInvestigacoes
    FROM `Ultimo_a_Saber`.`Investigacoes`
    WHERE `Investigacoes`.DataConclusao IS NULL;

    RETURN nInvestigacoes;
END $$
DELIMITER ;
```

6.Conclusões e Trabalho Futuro

Durante o desenvolvimento do projeto de base de dados, passamos por várias etapas importantes que nos permitiram criar uma solução funcional e robusta, apesar dos desafios encontrados.

Na 1a fase desenvolvemos uma série de etapas fundamentais.

Começamos pela fundamentação,nesta,procuramos entender o contexto no qual iriamos desenvolver a base de dados,assim como perceber se era viável realizar este projeto tendo em conta os recursos que tínhamos disponíveis.Também definimos requisitos(tendo em conta as necessidades dos usuários finais),que nos serviriam de base durante o desenvolvimento deste projeto.

De seguida, elaboramos modelos conceituais para representar de forma genérica a estrutura da base de dados que viria a ser desenvolvida e ajudar a sua compreensão. Esses modelos permitem ter uma visão ampla de todo o sistema.

Posteriormente, avançamos para os modelos lógicos, nos quais traduzimos o modelo conceitual,definindo as tabelas, chaves primárias e estrangeiras,no entanto percebemos mais tarde que este modelo poderia ter sido construído de uma melhor forma.

2a Fase

Uma das fases mais complexas da segunda parte do projeto foi refazer o modelo lógico. Identificamos problemas relacionados à normalização que precisavam ser corrigidos para garantir a integridade e eficiência da base de dados. Este passo foi crucial para evitar redundâncias, assegurando que a base de dados fosse escalável e mantivesse dados consistentes.

A transição do modelo lógico para o modelo físico foi realizada com sucesso. Através de forward engineering foram criadas todas as tabelas necessárias,com os seus atributos,chaves primárias e secundárias e todas as suas restrições.

Durante a implementação, percebemos que os requisitos iniciais não eram suficientemente abrangentes para uma base de dados completa. Isso nos levou a revisitar e aprimorar esses requisitos, tivemos que criar queries e procedimentos que não tinham sido pensados previamente, aumentando assim as funcionalidades da base de dados.

A criação de um pequeno programa em Java para interagir com a base de dados apresentou dificuldades. Problemas com a configuração do driver JDBC e a conexão à base de dados,estes foram superados quando se mudou de linguagem de programação e se desenvolveu o programa em Python. Desta forma foi possível criar uma conexão com a base de dados e começar a compreender como interagir com a base de dados utilizando outras linguagens de programação.

Apesar dos desafios e ajustes necessários, o projeto de base de dados foi concluído com êxito. As diversas etapas do processo — desde a fundamentação até ao modelo físico— demonstraram a eficácia do método de desenvolvimento de bases de dados. Conseguimos criar um sistema funcional, capaz de atender a uma variedade de necessidades e com potencial para ser expandido conforme novos requisitos surgirem.

Em resumo, o desenvolvimento desta base de dados não apenas atingiu seus objetivos iniciais, como também nos permitiu adquirir capacidades que poderão ser aplicadas em futuros projetos.