



Universidade do Minho

Mestrado Integrado em Engenharia Informática
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Letivo de 2020/2021

SGBD de uma clínica de testagem à Covid-19

Carlos Preto (A89587)

Maria Moreira (A89540)

Pedro Veloso (A89557)

Rui Fernandes (A89138)

Dezembro 2020

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

SGBD de uma clínica de testagem à Covid-19

Carlos Preto (A89587)

Maria Moreira (A89540)

Pedro Veloso (A89557)

Rui Fernandes (A89138)

Grupo 47

Dezembro 2020

Resumo

Numa realidade marcada pela pandemia do vírus covid-19, é imperativo que os Estados consigam rastrear e monitorizar os doentes infetados pelo vírus. Neste contexto, as bases de dados têm um papel fundamental. Neste trabalho, serão apresentados os modelos para construção de um sistema de gestão de bases de dados para uma clínica de realização de testes à covid-19, que tem que gerir marcações, postos de recolha e laboratórios de análise, bem como o registo dos clientes e seus dados.

Assim, foram identificadas várias entidades a colocar na base de dados: cliente, marcação, clínica, teste, posto, laboratório, funcionário e horário. Um cliente realiza uma marcação para realização do teste numa clínica. Na altura de marcação, deve ficar registado o posto onde a colheita será efetuada, bem como a hora e data de realização. Após a colheita, o material é enviado para um laboratório, onde um funcionário o irá analisar e emitir um resultado. A associação do cliente ao resultado e posto de colheita, deve ficar registada na BD.

Um cliente é identificado pelos atributos número de cartão de cidadão (chave primária), data de nascimento, nome, género, morada (composto por rua, localidade e código postal), e por um ou mais contactos (atributo multivalorado). A marcação apresenta um identificador, uma data de realização e, no caso de os testes serem realizados no modo drive-in, a entidade deve ainda ter associada uma matrícula do veículo do cliente, sendo, portanto, este um campo de preenchimento opcional. O posto, apresenta um identificador, um nome, morada, contacto e email. O horário, além do identificador, deve ter um dia, horário de funcionamento da parte da manhã e tarde associados. A entidade teste é caracterizada por um identificador, tipo de teste (monitorização, imunidade e à covid-19), datas de realização e análise, resultado (positivo, negativo ou inconclusivo) e observações, sendo estas opcionais. O laboratório, tal como a clínica, tem um identificador e um nome, e ainda um tipo (interno ou externo, no caso de ser próprio da clínica ou subcontratado). Nestas entidades o identificador é sempre a chave primária. Por fim, o funcionário apresenta um código (chave primária) e um nome.

Com base nestas entidades e nas possíveis relações entre elas, resumidas num Diagrama de Entidades-Relacionamentos, desenvolveu-se um modelo lógico, que criou tabelas para cada entidade. Este modelo, depois de validado, foi implementado usando o MySQL.

Área de Aplicação: Desenho e arquitetura de sistemas de bases de dados

Palavras-Chave: Bases de Dados, Sistema de gestão de bases de dados, Diagramas Entidades-Relacionamentos, Entidades, Atributos, Tabelas do modelo lógico.

Índice

Resumo	i
Índice	ii
Índice de Figuras	iv
Índice de Tabelas	vi
1.Introdução	1
1.1. Contextualização	1
1.2. Apresentação do caso de estudo	1
1.3. Motivação e objetivos	2
1.4. Estrutura do relatório	2
2. Levantamento e análise de requisitos	3
2.1. Método adotado	3
2.2. Análise geral dos requisitos	3
2.3. Requisitos	4
3. Modelação conceptual	6
3.1. Apresentação da abordagem de modelação	6
3.2. Identificação das entidades	6
3.3. Identificação dos relacionamentos	7
3.4. Determinação das multiplicidades	7
3.5. Identificação dos atributos	8
3.6. Definição de chaves candidatas, primárias e alternativas	9
3.7. Apresentação e explicação do diagrama ER	12
3.8. Validação do modelo conceptual	12
4. Modelação Lógica	14
4.1. Construção do modelo de dados lógico	14
4.2. Desenho do modelo lógico	18
4.3. Validação do modelo através da normalização	18
4.4. Validação do modelo com interrogações do utilizador	19
4.5. Revisão do modelo lógico	20
5. Implementação física	21
5.1. Seleção do sistema de gestão de bases de dados	21
5.2. Tradução do esquema lógico para o sistema de gestão de bases de dados	21

5.3. Tradução das interrogações para SQL	30
5.4. Escolha, definição e caracterização de índices	39
5.5. Estimativa do espaço em disco das bases de dados e taxa de crescimento anual	39
5.6. Definição e caracterização das vistas de utilização em SQL	41
5.7. Revisão do sistema implementado	42
6. Conclusão	43
Referências Bibliográficas	44
Lista de Siglas e Acrónimos	45
I.Anexo 1: Código do Povoamento Inicial	46

Índice de Figuras

Figura 1: Diagrama ER	12
Figura 2: Desenho do modelo lógico	18
Figura 3: Tabelas do modelo Físico	21
Figura 4: Conteúdo da Tabela Cliente no modelo físico	21
Figura 5: Tabela representativa da entidade cliente	22
Figura 6: Código de criação da Tabela Cliente	22
Figura 7: Trigger “Genero_OK”	23
Figura 8: Trigger “Nascimento_OK”	23
Figura 9: Tabela representativa do atributo multivalorado contacto cliente	23
Figura 10: Código de criação da Tabela Contacto Cliente	24
Figura 11: Tabela representativa da entidade marcação	24
Figura 12: Código de criação da Tabela Marcação	24
Figura 13: Trigger “DataMarcacao_OK”	25
Figura 14: Tabela representativa da entidade posto	25
Figura 15: Código de criação da Tabela Posto	25
Figura 16: Tabela representativa do atributo multivalorado contacto posto	26
Figura 17: Código de criação da Tabela Contacto Posto	26
Figura 18: Tabela representativa da entidade horário	26
Figura 19: Código de criação da Tabela Horário	27
Figura 20: Tabela representativa do atributo multivalorado posto opera horário	27
Figura 21: Código de criação da Tabela Posto Opera Horário	27
Figura 22: Tabela representativa da entidade teste	28
Figura 23: Código de criação da Tabela Teste	28
Figura 24: Tabela representativa da entidade laboratório	28
Figura 25: Código de criação da Tabela Laboratório	29
Figura 26: Tabela representativa da entidade funcionário	29
Figura 27: Código de criação da Tabela Funcionário	29
Figura 28: Tabela representativa da entidade clínica	29
Figura 29: Código de criação da Tabela Clínica	30
Figura 30: <i>Procedure</i> “agendar_Testes”	30
Figura 31: <i>Procedure</i> “desmarcar_Testes”	31
Figura 32: <i>Procedure</i> “remarcar_Testes”	32

Figura 33: <i>Procedure</i> “resultado_e_dono_Testes”	32
Figura 34: <i>Procedure</i> “lab_e_funcionario_Testes”	33
Figura 35: <i>Procedure</i> “posto_Testes”	33
Figura 36: <i>Procedure</i> “dadosPessoa_Testes”	33
Figura 37: <i>Procedure</i> “testes_Laboratorio”	34
Figura 38: <i>Procedure</i> “testes_Funcionario”	34
Figura 39: <i>Procedure</i> “marcacoes_Periodo”	34
Figura 40: <i>Procedure</i> “infetadas_faixaEtaria”	35
Figura 41: <i>Procedure</i> “infetadas_Genero”	35
Figura 42: <i>Procedure</i> “numero_testesGerais_Laboratorio”	36
Figura 43: <i>Procedure</i> “numero_testesResult_Laboratorio”	36
Figura 44: <i>Procedure</i> “numero_testesGerais_Posto”	36
Figura 45: <i>Procedure</i> “numero_testesResult_Posto”	37
Figura 46: <i>Procedure</i> “tipoTeste_Result”	37
Figura 47: <i>Procedure</i> “pessoas_tipoTeste”	38
Figura 48: <i>Procedure</i> “pessoas_tipoTeste_Periodo”	38
Figura 49: <i>Procedure</i> “pessoas_testePosto”	38
Figura 50: <i>Procedure</i> “pessoas_TestesLocalidade”	39
Figura 51: <i>View</i> “vista_Resultados_Testes”	41
Figura 52: <i>View</i> “pessoas_infetadas”	41
Figura 53: <i>View</i> “marcacoes_Efetoadas”	41

Índice de Tabelas

Tabela 1: Caracterização das relações entre as entidades	8
Tabela 2: Resumo dos atributos das entidades	10
Tabela 3: Tabela Cliente	15
Tabela 4: Tabela Marcação	15
Tabela 5: Tabela Posto	16
Tabela 6: Tabela Teste	17
Tabela 7: Tabela Funcionário	17
Tabela 8: Tamanho médio ocupado por cada registo em cada tabela	40

1. Introdução

O presente trabalho, desenvolvido no âmbito da unidade curricular de Base de Dados, tem como objetivo o desenvolvimento de um sistema de gestão de bases de dados para uma clínica de realização de testes de monitorização e rastreio do vírus covid-19.

1.1. Contextualização

Em tempos de pandemia, o rastreio à covid-19 através da realização de testes é a estratégia mais eficiente para a gestão da doença no país. Com a criação de postos de colheita laboratorial pretendeu-se permitir a toda a população portuguesa o rastreio da doença, evitando, assim, a sua propagação e impedindo a difusão das suas complicações.

Qualquer cidadão, seja ele nacional ou estrangeiro, tem direito à utilização dos serviços prestados num posto de colheita laboratorial, serviços esses que variam desde monitorização (com o controlo da temperatura e do nível das proteínas, bem como um questionário clínico), teste de imunidade (com a realização de um teste serológico) até à realização do teste à covid-19 (com resultados entregues entre 24-72 horas, ou em menos de 90 minutos).

A gestão de um posto de colheita laboratorial envolve bastante informação, não só acerca do posto, funcionários, localização e clientes do mesmo, mas também do laboratório a si associado. Por este motivo se infere a importância da implementação de um sistema de base de dados que possa responder às necessidades básicas do dia-a-dia dos seus colaboradores.

1.2. Apresentação do caso de estudo

Neste âmbito, para gerir uma clínica que realiza diferentes testes de rastreio à covid-19, será realizada uma base de dados (BD) para o efeito.

Uma clínica pode ser considerada uma entidade contactada por um cliente para marcação de um teste virológico. Associado a esta clínica, pode existir um ou vários postos de colheita laboratorial, localizados nas instalações da clínica, em espaços drive-in ou até no domicílio. Após a recolha, a clínica pode, igualmente, enviar o material recolhido para um laboratório, quer seja este interno da clínica ou, em casos de muita afluência, laboratórios externos que lhe prestem serviços.

Para este processo funcionar, é necessário a colaboração de vários funcionários para efetuar a recolha e proceder à análise dos resultados. Além disso, os clientes também são parte integrante do processo, pelo que devem estar devidamente identificados na BD a ser desenvolvida.

De modo a facilitar a realização desta BD, foi criada uma versão simplista, onde apenas se considerou a possibilidade de realização de testes em espaços drive-in, sendo estes os mais utilizados pelos clientes.

1.3. Motivação e objetivos

No contexto atual, é fundamental um funcionamento engrenado dos processos de marcação, recolha e análise, bem como o envio dos resultados ao cliente. Assim sendo, é motivação no desenvolvimento de uma BD de gestão de marcações de testes à covid, garantir o acesso organizado a testes, em tempo útil e coordenado com um baixo tempo de resposta. Seria caótico uma sala de espera de um ponto de recolha desorganizado, com clientes, possivelmente infetados, a chegar todos ao mesmo tempo e a terem de esperar para realizar o teste.

Para prevenir um cenário destes, a BD a desenvolver tem como principal objetivo facilitar o processo de marcação, evitando a formação de filas de clientes à espera para realizar a recolha.

Além disso, é igualmente importante armazenar os resultados dos testes com fiabilidade e segurança, de modo a evitar erros de comunicação ao cliente.

Por fim, pretende-se registar o historial clínico do cliente, para posteriores acompanhamentos do estado de saúde.

1.4. Estrutura do relatório

Este trabalho será dividido em 6 capítulos.

O presente capítulo apresentou uma introdução ao tema, bem os objetivos deste projeto e a sua estrutura.

No segundo capítulo é apresentada uma análise de requisitos para o desenvolvimento da BD.

O capítulo seguinte expõe uma explicação aprofundada da modelação conceptual adotada para fazer face ao problema identificado.

No capítulo 4 apresenta-se a modelação lógica, tendo como base o modelo concetual.

O capítulo 5 descreve o detalhe da implementação física da BD.

O último capítulo identifica as principais conclusões do trabalho, analisando os pontos fortes e/ou os menos bem conseguidos.

2. Levantamento e análise de requisitos

2.1. Método adotado

Para desenvolver o trabalho proposto, com o objetivo de criar uma BD mais aproximada à realidade vivida atualmente numa clínica de marcação e realização de testes à covid-19, foi realizada uma pesquisa junto de várias entidades que já realizam testes a este tipo de vírus. Para tal, foram analisadas várias páginas online de clínicas, de modo a perceber o método utilizado na marcação e realização dos testes, bem como identificar os tipos de testes existentes, laboratórios utilizados e postos de colheita disponíveis. Além disso, esta informação foi complementada com uma análise do conteúdo disponível pela direção geral de saúde (DGS), de forma a garantir que os dados utilizados neste trabalho apresentam veracidade.

Numa segunda fase, para que o levantamento de requisitos fosse o mais aproximado possível da realidade, realizamos algumas entrevistas de resposta aberta, a pessoas que já estiveram na situação de necessitar de marcar, realizar e esperar pelo resultado de um teste à covid-19. A amostra utilizada nestas entrevistas não foi suficiente para realizar generalizações, mas para cumprir o objetivo proposto – de perceber o processo de marcação – assumem-se, no âmbito deste trabalho, suficientes.

Da análise de toda a informação recolhida, foi possível compilar uma lista de postos de colheita laboratorial, disponibilizada em https://destinoseguro.azores.gov.pt/wp-content/uploads/2020/10/Listagem-de-Laboratorios-e-Postos-de-colheita_19102020.pdf. Sete destes postos, juntamente com clientes e horários de marcação totalmente fictícios, formam parte da população que irá constituir a BD desenvolvida.

2.2. Análise geral dos requisitos

Para mapear o fluxo de marcação e realização de um teste à covid-19, que sirva de base para a análise dos requisitos necessários a implementar na BD, é necessário identificar as entidades envolvidas e as suas interações.

Um cliente (nome, género, data de nascimento, número de identificação, morada e contacto) contacta uma clínica para realizar o pedido de marcação de um teste, indicando o tipo de teste pretendido. Para a localização do cliente, a clínica indica-lhe os postos de colheita

mais próximos disponíveis (morada, nome, contacto e horário) e o tipo de recolha efetuado (físico, drive-in ou ao domicílio). Após o cliente escolher o posto e o tipo de recolha pretendido, é realizada e registada a data e hora da marcação na BD, para o local selecionado. De ressaltar que, no caso de o tipo de recolha escolhido ser drive-in, pode ser solicitada a matrícula do veículo para facilitar a organização e registo no espaço de recolha.

Após a realização da colheita, o material recolhido é enviado para um laboratório (interno da clínica ou subcontratado) onde será analisado, por um funcionário do mesmo devidamente identificado na BD (nome, e número de funcionário). O resultado do teste – positivo, negativo ou inconclusivo – deverá ser registado na BD, sendo impossível a sua alteração após o registo, de modo a garantir que os dados não possam ser manipulados.

Para cumprir o seu principal objetivo, a BD não deve permitir o agendamento de dois testes em simultâneo no mesmo posto.

2.3. Requisitos

Na análise de requisitos é necessário identificar as características da clínica, dos postos de colheita laboratorial, dos laboratórios de análise, bem como dos clientes. Além disso, é fundamental definir o nível de atuação da BD, ou seja, o que esta pode disponibilizar como ações ao seu utilizador e o que não pode permitir que este faça.

Assim sendo, em seguida serão apresentados, de forma sintética, os requisitos identificados com fundamentais para o desenvolvimento deste trabalho. Estes requisitos foram divididos em três categorias:

1. Requisitos de descrição: para determinar os esquemas da BD. Nesta categoria serão identificados os atributos das entidades envolvidas;
2. Requisitos de exploração: para determinar o modo de povoação da BD e consequente exploração desses dados. Assim, serão apresentados os procedimentos e as funções da BD;
3. Requisitos de administração: para determinar as regras de utilização da BD.

Requisitos de descrição:

- Para que um cliente fique registado na BD é necessário saber o seu nome, data de nascimento, número de cartão de cidadão, contacto, género e morada;
- Para realizar uma marcação é necessário conhecer o cliente e a matrícula do veículo, caso o tipo de teste a realizar seja drive-in. Cada marcação tem um identificador (ID) associado e um horário;
- Para que um funcionário do laboratório esteja registado na base de dados é necessário saber o seu nome e o seu número de funcionário;
- Para que um posto de colheita laboratorial fique registado é necessário saber a sua morada, nome, horário e contacto, bem como o seu código de identificação;

- Um teste tem um cliente, data e hora de realização, posto e um código que o identifica. Posteriormente é constituído por observações e pode ter 3 tipos de resultados (positivo, negativo ou inconclusivo);
- Um laboratório tem um nome, funcionários, tipo (interno ou externo) e um código. Originam os resultados dos testes;
- A clínica tem um nome, vários postos e laboratórios que realizam as análises dos testes, bem como um código identificador.

Requisitos de exploração:

- A BD deve permitir agendar testes;
- Deve permitir desmarcar ou remarcar testes;
- Deve permitir emitir os resultados dos testes;
- Saber o resultado de um determinado teste e o dono do teste;
- Identificar o laboratório que analisou um teste, bem como o funcionário;
- Reconhecer o posto de recolha de um teste;
- Identificar os dados da pessoa a quem pertence um teste;
- A BD deve ser capaz de listar todos os testes realizados num determinado laboratório ou por um determinado funcionário;
- Obter uma lista com as marcações para um período;
- Deve listar as pessoas (clientes) infetadas, identificando a idade e o género da pessoa;
- A BD deve ser capaz de apresentar valores relativas ao número de pessoas infetadas numa determinada faixa etária ou género;
- Conhecer o número total de testes realizados num posto ou laboratório, tendo em conta o seu resultado ou de forma total;
- Deve ser capaz de identificar o tipo de teste a que um resultado se refere;
- A BD deve ser capaz de listar todas as pessoas que realizaram um tipo de teste;
- A BD deve permitir verificar quem realizou um certo tipo de teste num período;
- Deve ser capaz de listar todas as pessoas que realizaram um teste num determinado posto;
- A BD deve ser capaz de identificar as pessoas que realizaram um teste, tendo em conta a sua localidade de residência.

Requisitos de Administração:

- A BD não deve registar clientes sem um género válido associado;
- Não deve permitir registar clientes com datas de nascimento impossíveis – data superior à atual;
- A BD não deve permitir agendar dois testes para o mesmo horário e posto.

3. Modelação conceptual

3.1. Apresentação da abordagem de modelação

Para o desenvolvimento de uma BD é fundamental, numa primeira fase, a elaboração de um modelo conceptual bem conseguido, através da modelação de um Diagrama de Entidades-Relacionamentos (Diagrama ER). Este tipo de diagramas, deve ser constituído por entidades, atributos e pelos relacionamentos existentes entre elas.

Para desenvolver o diagrama ER utilizou-se o *brModelo*, onde as entidades são representadas através de retângulos, caracterizadas através de diferentes atributos, identificados por pequenas circunferências: preenchidas a cor preta ou branca. As circunferências coloridas a preto representam uma chave primária, isto é, um atributo único que distingue esta entidade de todas as outras. Na BD desenvolvida para o exemplo de uma clínica de marcação de testes à covid-19, cada cliente é identificado através de uma chave-primária que é o seu número de cartão de cidadão. Por outro lado, qualquer outro tipo de atributo é representado por uma circunferência branca. Os atributos opcionais, por sua vez, apresentam-se a tracejado.

Além disso, os losangos representam os relacionamentos existentes entre as entidades.

3.2. Identificação das entidades

A partir da análise de requisitos, podem ser identificadas 8 entidades, a saber:

- Cliente: identifica a pessoa que realiza a marcação e um teste à covid-19;
- Marcação: representa o ato de efetuar uma reserva de horário para recolha de material para o teste à covid-19;
- Posto: representa um local de colheita de material para análise;
- Horário: representa o horário de funcionamento, para um determinado dia, de um posto de colheita laboratorial;
- Laboratório: entidade que efetua a análise do material recolhido nos postos;
- Funcionário: pessoa que trabalha num laboratório;
- Teste: representa o teste realizado por um cliente, após analisado;
- Clínica: proprietário de um ou mais postos de colheita.

3.3. Identificação dos relacionamentos

Após identificação das entidades necessárias para a realização do trabalho, podem ser apresentados vários relacionamentos entre elas. Em seguida, descrevem-se os relacionamentos identificados:

- Uma clínica possui um laboratório;
- Uma clínica detém postos;
- Um laboratório emprega funcionários;
- Os funcionários analisam os testes;
- Um posto tem marcações;
- Um posto opera num horário;
- Os clientes efetua marcações;
- Os clientes realizam testes;
- Um posto realiza testes.

3.4. Determinação das multiplicidades

Da análise dos relacionamentos e entidades definidas, temos que um cliente pode efetuar várias marcações. Contudo, cada marcação está associada a apenas um cliente. De ressaltar que, para um cliente estar registrado na BD teve de efetuar, pelo menos, uma marcação e que uma marcação tem de ter um cliente associado, não podendo esta ser registrada na BD sem cliente.

Um cliente pode realizar entre 0 (caso tenha agendado a marcação, mas ainda não tenha efetuado o teste) e N testes, sendo que um teste tem obrigatoriamente um cliente associado e apenas um.

Um posto pode ter 0 ou mais marcações associadas, porém, uma marcação só pode ser realizada para 1 posto.

Um teste é realizado em apenas um posto e um posto pode realizar 0 ou N testes. Um posto encontra-se vinculado a uma clínica, todavia uma clínica pode ter mais do que 1 posto de colheita. Cada posto opera no mínimo num horário, sendo que vários postos podem ter o mesmo horário.

Além de possuir postos, a clínica detém 1 ou mais laboratórios que, por sua vez, empregam vários funcionários (no mínimo 1). Assume-se, neste contexto, que um funcionário não exerce funções em mais nenhum laboratório. Cada um destes funcionários pode efetuar análises, emitindo resultados de um teste. Cada resultado é emitido apenas por um funcionário.

A tabela seguinte, apresenta, de forma resumida, as multiplicidades encontradas entre as entidades do modelo.

Tabela 1: Caracterização das relações entre as entidades

Entidade	Multiplicidade	Relacionamento	Multiplicidade	Entidade
Clínica	(1,1)	<u>Possui</u>	(1,n)	Laboratório
Clínica	(1,1)	<u>Detém</u>	(1,n)	Posto
Posto	(1,n)	<u>Opera</u>	(1,n)	Horário
Posto	(1,1)	<u>Tem</u>	(0,n)	Marcação
Posto	(1,1)	<u>Realiza</u>	(0,n)	Teste
Cliente	(1,1)	<u>Efetua</u>	(1,n)	Marcação
Cliente	(1,1)	<u>Realiza</u>	(0,n)	Teste
Laboratório	(1,1)	<u>Emprega</u>	(1,n)	Funcionário
Funcionário	(1,1)	<u>Analisa</u>	(0,n)	Teste

3.5. Identificação dos atributos

Durante o processo de marcação de um teste, o cliente, caso não exista na BD, deve fornecer as suas informações pessoais, tais como o número de cartão de cidadão, nome, data de nascimento, género, contacto e morada. Uma vez que todos os clientes possuem estas propriedades, estas podem ser vistas então como atributos da entidade. Contudo, como a obtenção da morada é feita através de informações como localidade, código postal e a rua do cliente, considera-se a morada como um atributo composto. Tendo em conta que cada cliente pode possuir mais do que um contacto, considera-se este como um atributo *multivalorado*. Após garantir que o cliente está registado na BD, regista-se a marcação com uma data para a realização do teste, que refere o dia, mês e ano, bem como a hora para a realização do teste. A cada marcação é atribuído um número identificativo que também pode ser visto como um atributo.

O posto deve indicar o seu nome, morada, contacto e email, bem como o seu número identificativo, sendo todos estes considerados atributos desta entidade. No entanto, uma vez mais a morada é considerado um atributo composto pelos atributos localidade, código postal e rua e o contacto é considerado um atributo *multivalorado*. Já o email considera-se um atributo opcional. Cada posto apenas opera num determinado horário, identificado pelo identificador, dia a que se refere, bem como a hora a que se inicia e termina o trabalho na parte da manhã e da tarde. Estes atributos relativos ao horário podem ser opcionais quando o posto se encontrar fechado.

A uma marcação de um teste num determinado posto pode ser associada a matrícula do veículo do cliente, originando o atributo opcional matrícula na entidade marcação.

Um teste, tal como as restantes entidades, deve possuir um identificador, além de ter que ser identificado pelo tipo (monitorização, imunidade e à covid-19) e data de realização, que constituem atributos simples da entidade.

O laboratório de análise também possui um identificador, um tipo (interno ou externo) e o seu nome. Tal como a anterior, esta entidade é caracterizada por atributos simples e opcionais (no caso do tipo). Um laboratório emprega funcionários, que devem ser identificados pelo seu código de funcionários, bem como o seu nome. A relação do funcionário com um teste origina atributos como o resultado do teste, data de realização da análise ou outras observações pertinentes, na entidade teste. O resultado e a data de análise podem ser considerados atributos simples, já a observação é um atributo opcional.

A clínica apresenta um identificador e um nome, sendo, portanto, identificada através de dois atributos simples.

3.6. Definição de chaves candidatas, primárias e alternativas

No desenvolvimento da BD, é necessário identificar as chaves candidatas, primárias e alternativas das entidades.

As chaves primárias são aquelas que identificam, sem forma de dúvida, uma entidade na BD. Por exemplo, cada carro específico de uma BD será identificado através da chave primária “matrícula”. As chaves candidatas são todas aquelas que apresentam características para serem chaves primárias de uma entidade. No exemplo anterior, a matrícula e o número do quadro do chassi poderiam ser consideradas chaves candidatas para identificar um carro. As chaves candidatas não escolhidas podem ser consideradas chaves alternativas.

Para identificar o cliente, identifica-se como chave candidata o número de cartão de cidadão. Neste caso a chave candidata deu origem à chave primária.

Para a entidade marcação não se encontrou nenhuma chave candidata, uma vez que a data pode ser repetida em diferentes postos. Desta forma, foi adicionado um atributo extra – o ID – que pretende transformar cada marcação num registo único na BD. Este atributo será, então, a chave primária.

No caso do posto de recolha, apesar da morada ser chave candidata para identificar o posto, optou-se por criar uma chave primária – ID – pois, considerou-se mais fácil o utilizador utilizar o ID do posto para o identificar.

Para o horário, tal como aconteceu no caso da marcação, teve que se acrescentar uma chave primária através de um atributo artificial, pois a altura do dia e o próprio dia não podem ser consideradas chaves candidatas a identificar esta entidade.

Para o teste também foi acrescentado o ID, pois podem ser realizados vários testes no mesmo dia e hora, mas em postos diferentes.

No caso do laboratório e da clínica, o nome pode ser classificado como chave candidata, uma vez que em Portugal não podem ser registadas duas empresas com o mesmo nome. Contudo este será um campo textual potencialmente composto por vários caracteres. Assim, ao se criar um ID para ser usado como chave primária desta entidade, facilita a utilização por parte do utilizador, sendo, portanto, o nome uma chave alternativa.

Os funcionários têm como chave candidata e primária o seu número de funcionário.

A tabela seguinte resume a informação sobre os atributos das entidades apresentadas anteriormente.

Tabela 2: Resumo dos atributos das entidades

Entidade	Atributo	Chave	Descrição	Domínio	Nulo	Multivalorado
Cliente	Nº de Cartão de Cidadão	Primária	Número de identificação do cidadão	Inteiro	Não	Não
	Data de nascimento		Data de nascimento do cliente	Data	Não	Não
	Nome		Nome do cliente	String	Não	Não
	Género		Género do cliente	Char	Não	Não
	Morada		Rua, localidade e código postal do cliente	String	Não	Não
	Contacto		Contacto telefónico do cliente	String	Não	Sim
Marcação	ID	Primária	Nº identificador do registo no sistema	Inteiro	Não	Não
	Data Realização		Dia e hora de realização	Data	Não	Não
	Matrícula		Identificado do veículo do cliente	String	Sim	Não
Posto	ID	Primária	Nº identificador do posto no sistema	Inteiro	Não	Não
	Nome		Nome do posto	String	Não	Não
	Morada	Candidata	Rua, localidade e código postal do posto	String	Não	Não
	Contacto		Contacto telefónico do posto	String	Não	Sim
	Email		Contacto eletrónico do posto	String	Sim	Não

Horário	ID	Primária	Nº identificador do horário no sistema	Inteiro	Não	Não
	Dia		Dia da semana	String	Não	Não
	Início da manhã		Hora de abertura do posto da parte da manhã	Hora	Sim	Não
	Fim da manhã		Hora de encerramento da parte da manhã	Hora	Sim	Não
	Início da tarde		Hora de abertura do posto da parte da tarde	Hora	Sim	Não
	Fim da tarde		Hora de encerramento da parte da tarde	Hora	Sim	Não
Teste	ID	Primária	Nº identificador do teste no sistema	Inteiro	Não	Não
	Tipo		Tipo de teste: monitorização, imunidade e à covid-19	Char	Não	Não
	Data de realização		Data de realização do teste	Data	Não	Não
	Data de análise		Data de análise do teste	Data	Não	Não
	Resultado		Resultado do teste: positivo, negativo ou inconclusivo	Char	Não	Não
	Observações		Notas adicionais ao resultado	String	Sim	Não
Laboratório	ID	Primária	Nº identificador do laboratório no sistema	Inteiro	Não	Não
	Nome	Candidata	Nome do laboratório	String	Não	Não
	Tipo		Interno ou externo	Char	Sim	Não
Funcionário	Código	Primária	Código do funcionário	Inteiro	Não	Não
	Nome		Nome do funcionário	String	Não	Não
Clínica	ID	Primária	Nº identificador da clínica no sistema	Inteiro	Não	Não
	Nome	Candidata	Nome da clínica	String	Não	Não

3.7. Apresentação e explicação do diagrama ER

As entidades, os relacionamentos entre as mesmas e os seus atributos anteriormente mencionados podem ser compilados num Diagrama ER.

O Diagrama ER que representa o esquema da BD a desenvolver para uma clínica de realização de testes de monitorização e rastreio da covid-19 é apresentada na Figura 1.

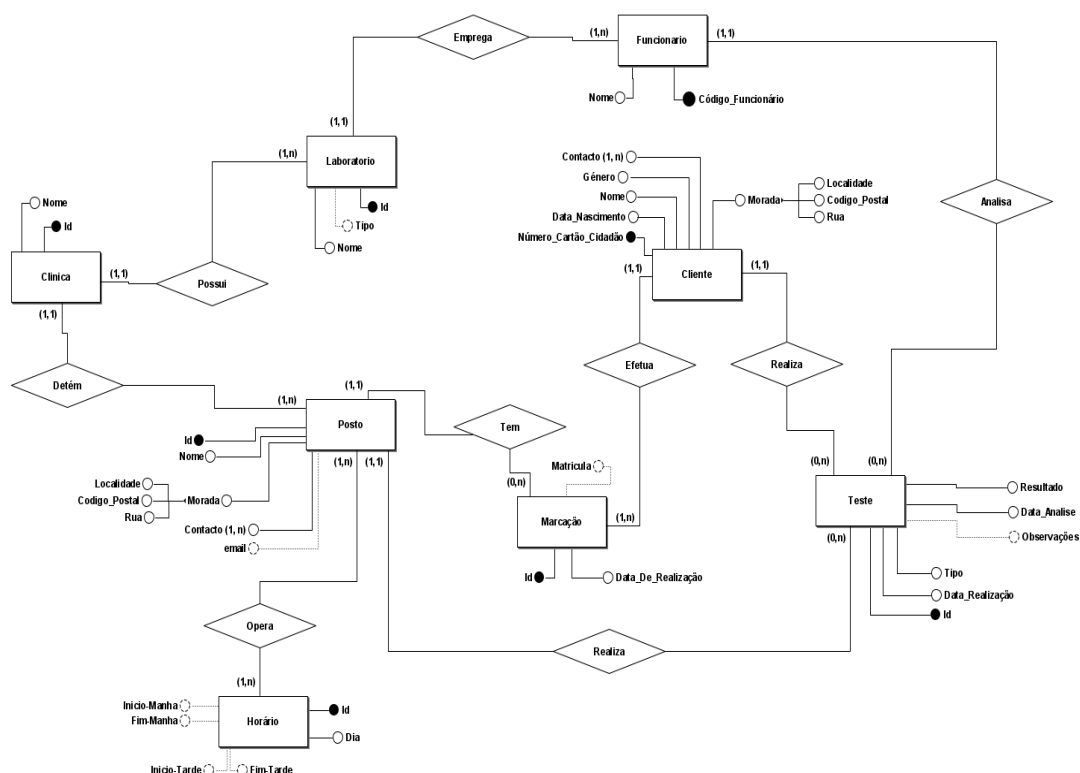


Figura 1: Diagrama ER

3.8. Validação do modelo conceptual

Após o desenvolvimento do modelo conceptual, é indispensável realizar uma análise do mesmo, de modo a verificar se é possível realizar transações e interrogações, pois, só deste modo, o modelo reflete um correto funcionamento.

Para avaliar a validade de um modelo concetual, optou-se por utilizar o método “Descrição da Transação”. Este método consiste em verificar que toda a informação sobre as entidades, relacionamentos entre as mesmas e atributos é fornecida pelo modelo.

São exemplo de transações e interrogações as seguintes:

1. Os atributos de cada entidade possibilitam o controlo de todos os dados necessários para a validação de cada tipo de registo. Por exemplo, para registar um cliente na BD é necessário conhecer o seu nome, número de cartão de

cidadão, contacto, género, morada e data de nascimento. Como a entidade cliente possui todos os atributos acima mencionados, esta transação é possível;

2. Obter listas com informações sobre os testes, clientes a eles associados ou funcionários responsáveis pela sua análise é possível devido às relações existentes entre as entidades;
3. Filtrar testes por data de realização, data de análise, resultado ou tipo de teste é possível devido à existência desses atributos na entidade teste e na sua relação com a entidade funcionário;
4. É possível obter as marcações de um determinado posto devido à existência de duas entidades posto e marcação, que se encontram relacionadas.

Com todas as validações realizadas ao modelo conceptual, pode-se avançar para o desenvolvimento do modelo lógico.

O modelo lógico pode ser encontrado no capítulo seguinte.

4. Modelação Lógica

Numa segunda fase do desenvolvimento de uma BD é necessária a criação de um modelo lógico (ML), tendo por base o modelo conceptual apresentado anteriormente. O ML é construído através de processos de normalização, que, tendo em conta os requisitos dos utilizadores, permite a validação do modelo. Este processo dá origem a diversas tabelas de registo de dados. Nestas existem chaves primárias ou estrangeiras, que são chaves primárias de outras tabelas. Os atributos, bem como o seu tipo de dados e tamanho estão ainda presentes neste tipo de tabelas.

4.1. Construção do modelo de dados lógico

Em seguida, serão identificadas as tabelas do ML oriundas das entidades, dos atributos multivalorados e da relação de muitos para muitos existente entre as entidades posto e horário.

4.1.1. Tabela Cliente

Esta tabela é proveniente da entidade cliente e identifica todos os clientes que se encontram registados na BD. É composta por 7 atributos, sendo o número de cartão de cidadão a chave primária.

De notar ainda que o atributo composto morada se encontra representado através dos seus atributos simples – localidade, código postal e rua.

O atributo multivalorado contacto não se encontra associado a esta tabela, pois dará origem a uma tabela própria (apresentada no ponto seguinte).

A tabela seguinte identifica os atributos da tabela do ML do cliente, bem como a identificação das chaves primárias e estrangeiras existentes na mesma.

Tabela 3: Tabela Cliente

Atributos	Chave Primária	Chave Estrangeira
Número de cartão de cidadão	X	
Data de nascimento		
Nome		
Género		
Localidade		
Código postal		
Rua		

4.1.2. Tabela Contacto Cliente

O atributo multivalorado contacto associado à entidade cliente deu origem à Tabela Contacto Cliente. Nesta tabela podem ser encontradas 2 chaves primárias – o número de telefone e o identificador do cliente. Contudo, a segunda também é considerada uma chave estrangeira. Neste caso, a esta tabela tem uma chave primária composta.

4.1.3. Tabela Marcação

Os atributos, chaves primárias e estrangeiras da tabela marcação podem ser encontrados na tabela seguinte. Esta tabela é resultado da entidade marcação.

Tabela 4: Tabela Marcação

Atributos	Chave Primária	Chave Estrangeira
ID	X	
Data de realização		
Matrícula		
Posto		X
Cliente		X

4.1.4. Tabela Posto

A Tabela Posto, resultado da entidade posto, segue a seguinte lógica:

Tabela 5: Tabela Posto

Atributos	Chave Primária	Chave Estrangeira
ID	X	
Localidade		
Código postal		
Rua		
Nome		
Email		
Clínica		X

O atributo email pode apresentar um valor *null*.

4.1.5. Tabela Contacto Posto

O atributo multivalorado contacto da entidade posto permitiu a criação da Tabela Contacto Posto. A chave primária desta tabela é o número de telefone e tem como chave estrangeira o identificador do posto, com o mesmo domínio de dados que a chave primária do posto. Contudo, a chave identificadora do posto também é uma chave primária, pelo que esta tabela apresenta uma chave primária composta.

4.1.6. Tabela Horário

A Tabela Horário tem como atributos o dia, o início da manhã, fim da manhã, início da tarde, fim da tarde e a chave primária ID. O início e fim da manhã ou tarde podem ser valores *null*, caso o posto esteja fechado num dos blocos horários ou até em ambos. Esta tabela é fruto da entidade horário.

4.1.7. Tabela Posto Opera Horário

A relação de N para M existente entre as entidades posto e horário, deu origem à Tabela Posto Opera Horário. Esta tabela apresenta a particularidade de ter apenas dois atributos que, simultaneamente, são chaves primárias e estrangeiras.

4.1.8. Tabela Teste

A Tabela Teste, proveniente da entidade teste, apresenta as seguintes características:

Tabela 6: Tabela Teste

Atributos	Chave Primária	Chave Estrangeira
ID	X	
Data de realização		
Tipo		
Data de análise		
Resultado		
Observações		
Posto		X
Cliente		X
Funcionário		X

O atributo observações pode apresentar um valor *null*.

4.1.9. Tabela Laboratório

A Tabela Laboratório, resultante da entidade laboratório, apresenta como chave primária o ID e como chave estrangeira o atributo clínica. Contém ainda os atributos nome e tipo. Uma vez que o tipo é um atributo opcional, tal como identificado no Diagrama ER, este pode não estar registado na BD (pode ser um valor *null*).

4.1.10. Tabela Funcionário

A Tabela Funcionário é resultado da entidade funcionário e pode ser caracterizada da seguinte forma:

Tabela 7: Tabela Funcionário

Atributos	Chave Primária	Chave Estrangeira
Código de funcionário	X	
Nome		
Laboratório		X

4.1.11. Tabela Clínica

A Tabela Clínica contém 2 atributos: o ID, como chave primária, e o nome. O ID, por sua vez, também vai ser chave estrangeira de outras tabelas. Esta tabela provém da entidade clínica.

4.2. Desenho do modelo lógico

Neste trabalho, o ML foi desenvolvido utilizando o MySQL Workbench e encontra-se presente na figura seguinte. Este modelo segue as tabelas acima explicitadas.

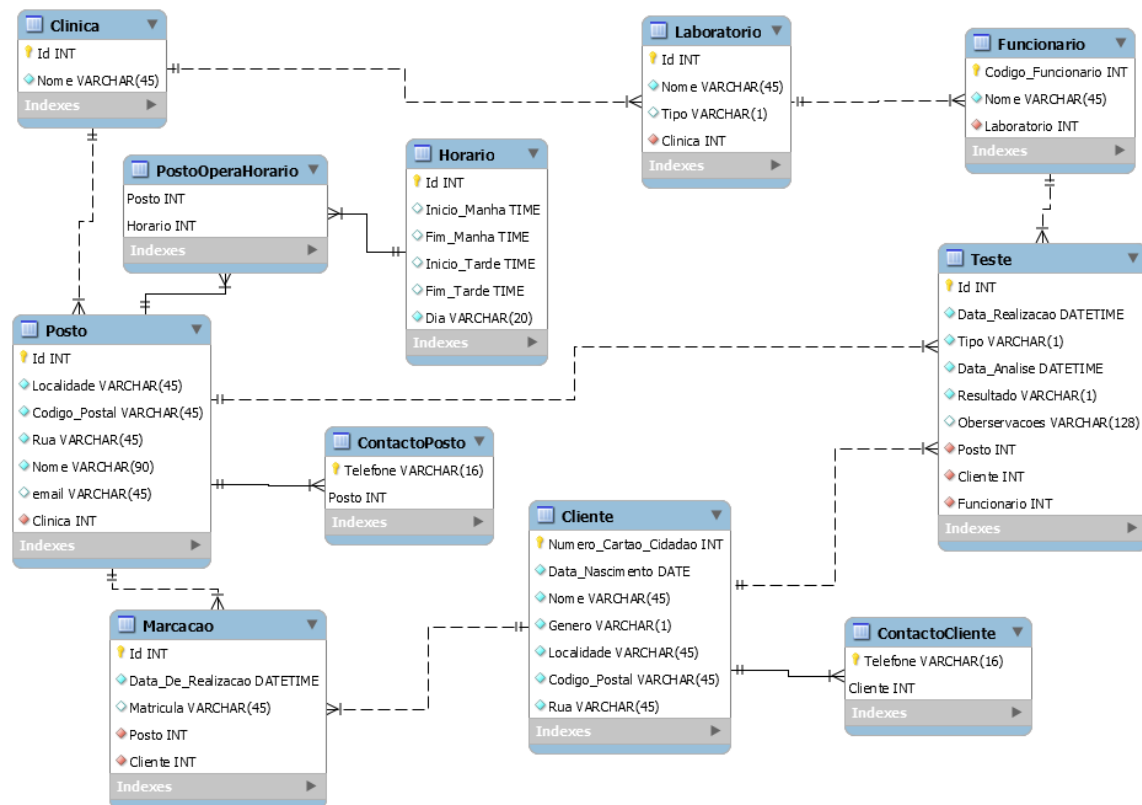


Figura 2: Desenho do modelo lógico

4.3. Validação do modelo através da normalização

Para implementação de um modelo eficaz, é necessário validar a sua normalização, pois desta forma é possível evitar erros entre as relações.

A validação das relações através da normalização requer a análise das 3 formas normais.

A primeira forma normal implica que os atributos na BD não possam ser compostos ou multivalorados. Para garantir esta forma normal, o atributo composto morada (associado às entidades cliente e posto) não foi representado nas tabelas, tendo sido dividido em 3 atributos simples, já identificados anteriormente. Além disso, o atributo multivalorado contacto (das entidades cliente e posto) deu origem a duas tabelas com uma chave primária composta.

A segunda forma normal pretende garantir que os atributos normais (qualquer atributo que não seja chave primária) dependem apenas da chave primária da tabela. Assim, se existir um atributo não dependente de uma chave primária da tabela, esse atributo deve ser

desassociado da tabela e colocado numa outra, onde se possa verificar uma dependência direta com a chave primária dessa tabela. Deste modo, a não redundância de dados no modelo fica assegurada.

Por último, a terceira forma normal deve assegurar que os atributos apenas dependem da chave primária e são completamente independentes entre si, ou seja, não existem dependências transitivas.

Analisando o modelo desenvolvido, é possível perceber que não existem redundâncias de dados no modelo, uma vez que este cumpre todas as formas normais.

4.4. Validação do modelo com interrogações do utilizador

Após a validação do modelo através da normalização, é fundamental verificar se este cumpre com todos os requisitos inicialmente definido pelos utilizadores.

Os requisitos relativos à marcação de testes e registo dos respetivos resultados, são garantidos através das tabelas marcação e teste, definidas no modelo.

A relação existente entre as tabelas cliente e teste, permite saber o resultado de um teste e associá-lo ao respetivo dono, bem como aos seus dados base. Esta mesma relação permite fazer uma análise, por idade e género, dos clientes com testes positivos, conseguindo-se assim, valores estatísticos. Com o atributo tipo na tabela teste, é possível identificar a espécie de teste a que um resultado se refere. Através deste atributo e da relação já identificada entre o cliente e o teste, pode-se obter uma lista com todas as pessoas que realizaram um certo tipo de teste, de uma forma geral ou num certo intervalo de tempo. Esta análise temporal só é possível através dos atributos data de realização e data de análise da tabela teste. A relação entre as tabelas cliente e teste, juntamente com o atributo localidade da tabela cliente, permite ainda identificar a localidade de residência das pessoas que realizaram um teste. É ainda possível obter o contacto de um cliente que realizou um determinado teste, através da relação das tabelas cliente-contato cliente e cliente-teste.

A relação entre posto e teste permite listar todos os testes realizados num certo posto e conhecer o número total de testes realizados num posto. O atributo resultado, presente na tabela teste, permite tratar os resultados de forma individual.

Das tabelas funcionário e teste pode-se extrair a informação acerca do funcionário que analisou um determinado teste, sendo possível, deste modo, listar todos os testes analisados por um funcionário. Quando se analisa a relação das tabelas teste, funcionário e laboratório, pode-se identificar o laboratório que realizou um determinado teste, bem como listar todos os testes analisados num determinado laboratório.

Por fim, a relação entre posto-marcação e marcação-cliente pode identificar todos os clientes que realizaram uma marcação e que vão realizar um teste num determinado horário (devido ao atributo data de marcação na tabela marcação). Para um posto é ainda possível

identificar o seu horário, através da relação das tabelas posto-posto opera horário e posto opera horário-horário.

De notar que, os atributos de cada tabela possibilitam o controlo de todos os dados necessários para a validação de cada tipo de registo.

4.5. Revisão do modelo lógico

Efetuada a validação do ML através da normalização e, posteriormente, com as interrogações do utilizador, por parte do grupo de trabalho, é necessário avaliar o mesmo modelo com os utilizadores finais da BD, de modo a perceber se o modelo criado vai de encontro às suas expectativas de utilização.

Assim sendo, após a aprovação do modelo por parte destes, pode-se iniciar o desenvolvimento do modelo físico, com base nos modelos conceptual e lógico definidos.

O modelo físico será apresentado no capítulo seguinte.

5. Implementação física

5.1. Seleção do sistema de gestão de bases de dados

Uma BD é gerida por um sistema de gestão de bases de dados (SGBD). Existem diferentes SGBD diferenciados em função dos requisitos e funcionalidades que proporcionam, como segurança, integridade dos dados e intolerância a falhas.

O SGBD a utilizar deve ser o mais adequado ao problema em causa e o que possibilite a realização de várias operações por vários utilizadores ao mesmo tempo.

Neste trabalho, foi utilizado SGBD relacional MySQL, para desenvolvimento da BD, baseado na linguagem SQL, uma vez que é um sistema simples, rápido e seguro.

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

Na implementação física do modelo lógico, desenvolveram-se tabelas representativas das entidades, compostas por chaves primárias, estrangeiras e atributos, identificados nas colunas. Estas tabelas suportam o modelo físico.

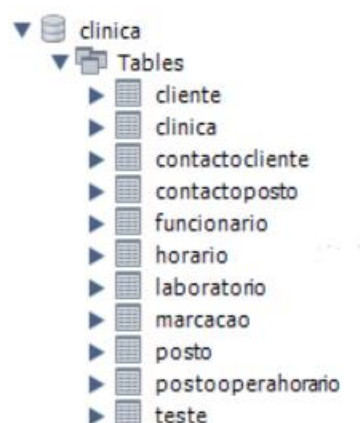


Figura 3: Tabelas do modelo Físico

Table: **cliente**

Columns:

<u>Numero_Cartao_Cidadao</u>	int PK
Data_Nascimento	datetime
Nome	varchar(45)
Genero	varchar(1)
Localidade	varchar(45)
Codigo_Postal	varchar(45)
Rua	varchar(45)

Figura 4: Conteúdo da Tabela Cliente no modelo físico

5.2.1. Tabela Cliente

O cliente é identificado através do número do seu cartão de cidadão, sendo este atributo a chave primária da Tabela Cliente. Esta chave é não nula e única, pois garante que para cada cliente não existem dois números de cartão de cidadão iguais.

Os restantes atributos do cliente são não nulos.








Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 Numero_Cartao_Cidadao	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Data_Nascimento	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Nome	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Genero	VARCHAR(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Localidade	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Codigo_Postal	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Rua	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 5: Tabela representativa da entidade cliente

Para a criação da Tabela Cliente, foi utilizado o seguinte código SQL:

```
CREATE TABLE `cliente` (  
  `Numero_Cartao_Cidadao` int NOT NULL,  
  `Data_Nascimento` date NOT NULL,  
  `Nome` varchar(45) NOT NULL,  
  `Genero` varchar(1) NOT NULL,  
  `Localidade` varchar(45) NOT NULL,  
  `Codigo_Postal` varchar(45) NOT NULL,  
  `Rua` varchar(45) NOT NULL,  
  PRIMARY KEY (`Numero_Cartao_Cidadao`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 6: Código de criação da Tabela Cliente

Para garantir que não existem incoerências no género dos clientes, foi definido o Trigger “Genero_OK” (Figura 7), que impossibilita a introdução de valores diferentes de ‘M’ ou ‘F’ neste campo.

```

DELIMITER $$
CREATE TRIGGER Genero_OK BEFORE INSERT ON cliente
FOR EACH ROW
BEGIN
    IF(new.Genero <> 'F' AND new.Genero <> 'M')
    THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Genero escolhido é invalido, use F ou M';
    END IF;
END $$
DELIMITER $$

```

Figura 7: Trigger “Genero_OK”

Nesta entidade, foi ainda criado o Trigger “Nascimento_OK” (Figura 8), que permite verificar que todas as datas de nascimento introduzidas não são superiores à data atual.

```

DELIMITER $$
CREATE TRIGGER Nascimento_OK BEFORE INSERT ON cliente
FOR EACH ROW
BEGIN
    IF(new.Data_Nascimento > DATE(now()))
    THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Data de nascimento invalida.';
    END IF;
END $$
DELIMITER $$

```

Figura 8: Trigger “Nascimento_OK”

5.2.2. Tabela Contacto Cliente

O atributo multivalorado contacto associado à entidade cliente deu origem à Tabela Contacto Cliente. Nesta tabela podem ser encontradas 2 chaves primárias – o número de telefone e o identificador do cliente. As chaves primárias apresentam valores não nulos e únicos.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
☛ Telefone	VARCHAR(16)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
☛ Cliente	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 9: Tabela representativa do atributo multivalorado contacto cliente

Para criação desta tabela, usou-se o seguinte código SQL:

```
CREATE TABLE `contactocliente` (
  `Telefone` varchar(16) NOT NULL,
  `Cliente` int NOT NULL,
  PRIMARY KEY (`Telefone`, `Cliente`),
  KEY `fk_ContactoCliente_Cliente_idx` (`Cliente`),
  CONSTRAINT `fk_ContactoCliente_Cliente` FOREIGN KEY (`Cliente`) REFERENCES `cliente` (`Numero_Cartao_Cidadao`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 10: Código de criação da Tabela Contacto Cliente

5.2.3. Tabela Marcação

O ID identifica a marcação e é considerado a chave primária da Tabela Marcação. Nesta tabela deve ainda estar identificado de forma obrigatória os atributos data de realização, posto e cliente, sendo por isso atributos classificados como não nulos. O atributo matrícula, por sua vez, pode ser classificado como nulo.






Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 Id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Data_De_Realizacao	DATETIME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Matricula	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Posto	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Cliente	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 11: Tabela representativa da entidade marcação

A Tabela Marcação foi criada seguindo o código SQL abaixo:

```
CREATE TABLE `marcacao` (
  `Id` int NOT NULL,
  `Data_De_Realizacao` datetime NOT NULL,
  `Matricula` varchar(45) DEFAULT NULL,
  `Posto` int NOT NULL,
  `Cliente` int NOT NULL,
  PRIMARY KEY (`Id`),
  KEY `fk_Marcacao_Posto1_idx` (`Posto`),
  KEY `fk_Marcacao_Cliente1_idx` (`Cliente`),
  CONSTRAINT `fk_Marcacao_Cliente1` FOREIGN KEY (`Cliente`) REFERENCES `cliente` (`Numero_Cartao_Cidadao`),
  CONSTRAINT `fk_Marcacao_Posto1` FOREIGN KEY (`Posto`) REFERENCES `posto` (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 12: Código de criação da Tabela Marcação

Para garantir a coerência nas marcações de um posto, foi definido o *trigger* “DataMarcacao_OK” que impossibilita a marcação de um teste no mesmo horário para o mesmo posto, encontrado na figura seguinte.


```

DELIMITER $$
CREATE TRIGGER DataMarcacao_Ok BEFORE INSERT ON marcacao
FOR EACH ROW
BEGIN
    DECLARE soma INT;
    SELECT count(*) INTO soma
    FROM marcacao
    WHERE marcacao.Data_De_Realizacao = new.Data_De_Realizacao AND marcacao.Posto = new.Posto;

    IF(soma > 0)
    THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Data de marcação Inválida.';
    END IF;
END $$
DELIMITER $$

```

Figura 13: Trigger “DataMarcacao_OK”

5.2.4. Tabela Posto

O posto tem como chave primária o seu ID e apresenta atributos não nulos como localidade, código postal, rua, nome e clínica. Por sua vez, o email, não sendo obrigatório, pode adquirir um valor *null*.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Localidade	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Codigo_Postal	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Rua	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Nome	VARCHAR(90)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
email	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Clinica	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 14: Tabela representativa da entidade posto

A Tabela Posto foi criada através do seguinte código SQL:

```

CREATE TABLE `posto` (
  `Id` int NOT NULL,
  `Localidade` varchar(45) NOT NULL,
  `Codigo_Postal` varchar(45) NOT NULL,
  `Rua` varchar(45) NOT NULL,
  `Nome` varchar(90) NOT NULL,
  `email` varchar(45) DEFAULT NULL,
  `Clinica` int NOT NULL,
  PRIMARY KEY (`Id`),
  KEY `fk_Posto_Clinica1_idx` (`Clinica`),
  CONSTRAINT `fk_Posto_Clinica1` FOREIGN KEY (`Clinica`) REFERENCES `clinica` (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8

```

Figura 15: Código de criação da Tabela Posto

5.2.5. Tabela Contacto Posto

O atributo multivalorado contacto da entidade posto permitiu a criação da Tabela Contacto Posto. A chave primária desta tabela é o número de telefone e tem como chave estrangeira o identificador do posto. Estas chaves têm valores não nulos e únicos.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
🔔 Telefone	VARCHAR(16)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔔 Posto	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 16: Tabela representativa do atributo multivalorado contacto posto

Para criação desta tabela, usou-se o código SQL seguinte:

```
CREATE TABLE `contactoposto` (
  `Telefone` varchar(16) NOT NULL,
  `Posto` int NOT NULL,
  PRIMARY KEY (`Telefone`, `Posto`),
  KEY `fk_ContactoPosto_Posto1_idx` (`Posto`),
  CONSTRAINT `fk_ContactoPosto_Posto1` FOREIGN KEY (`Posto`) REFERENCES `posto` (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 17: Código de criação da Tabela Contacto Posto

5.2.6. Tabela Horário

A Tabela Horário tem como atributos o dia, início da manhã, fim da manhã, início da tarde, fim da tarde e a chave primária ID. O início e fim da manhã ou tarde podem ser valores *null*. O dia é obrigatório, logo terá que ser preenchido na BD.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
🔔 Id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔔 Inicio_Manha	TIME	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔔 Fim_Manha	TIME	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔔 Inicio_Tarde	TIME	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔔 Fim_Tarde	TIME	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
🔔 Dia	VARCHAR(20)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 18: Tabela representativa da entidade horário

Esta Tabela Horário foi criada pelo código SQL abaixo:

```
CREATE TABLE `horario` (
  `Id` int NOT NULL,
  `Inicio_Manha` time DEFAULT NULL,
  `Fim_Manha` time DEFAULT NULL,
  `Inicio_Tarde` time DEFAULT NULL,
  `Fim_Tarde` time DEFAULT NULL,
  `Dia` varchar(20) NOT NULL,
  PRIMARY KEY (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 19: Código de criação da Tabela Horário

5.2.7. Tabela Posto Opera Horário

Nesta tabela, os atributos posto e horário formam uma chave primária composta e, como tal, são valores não nulos e únicos.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Posto	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Horario	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 20: Tabela representativa do atributo multivalorado posto opera horário

Para criação desta tabela, usou-se o seguinte código SQL:

```
CREATE TABLE `postooperahorario` (
  `Posto` int NOT NULL,
  `Horario` int NOT NULL,
  PRIMARY KEY (`Posto`, `Horario`),
  KEY `fk_Opera_Posto1_idx` (`Posto`),
  KEY `fk_Opera_Horario1_idx` (`Horario`),
  CONSTRAINT `fk_Opera_Horario1` FOREIGN KEY (`Horario`) REFERENCES `horario` (`Id`),
  CONSTRAINT `fk_Opera_Posto1` FOREIGN KEY (`Posto`) REFERENCES `posto` (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 21: Código de criação da Tabela Posto Opera Horário

5.2.8. Tabela Teste

O teste tem como chave primária o ID, sendo esta única e não nula. Na Tabela Teste estão ainda presentes 3 chaves estrangeiras: posto, cliente e funcionário, classificadas como não nulas. Dos restantes atributos presentes, apenas o atributo observações pode assumir um valor *null*.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Data_Realizacao	DATETIME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Tipo	VARCHAR(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Data_Analise	DATETIME	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Resultado	VARCHAR(1)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Oberservacoes	VARCHAR(128)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Posto	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Cliente	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Funcionario	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 22: Tabela representativa da entidade teste

Esta tabela foi criada com base no seguinte código SQL:

```
CREATE TABLE `teste` (
  `Id` int NOT NULL,
  `Data_Realizacao` datetime NOT NULL,
  `Tipo` varchar(1) NOT NULL,
  `Data_Analise` datetime NOT NULL,
  `Resultado` varchar(1) NOT NULL,
  `Oberservacoes` varchar(128) DEFAULT NULL,
  `Posto` int NOT NULL,
  `Cliente` int NOT NULL,
  `Funcionario` int NOT NULL,
  PRIMARY KEY (`Id`),
  KEY `fk_Testes_Posto1_idx` (`Posto`),
  KEY `fk_Testes_Cliente1_idx` (`Cliente`),
  KEY `fk_Testes_Funcionario1_idx` (`Funcionario`),
  CONSTRAINT `fk_Testes_Cliente1` FOREIGN KEY (`Cliente`) REFERENCES `cliente` (`Numero_Cartao_Cidadao`),
  CONSTRAINT `fk_Testes_Funcionario1` FOREIGN KEY (`Funcionario`) REFERENCES `funcionario` (`Codigo_Funcionario`),
  CONSTRAINT `fk_Testes_Posto1` FOREIGN KEY (`Posto`) REFERENCES `posto` (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 23: Código de criação da Tabela Teste

5.2.9. Tabela Laboratório

A Tabela Laboratório apresenta como chave primária o ID e como chave estrangeira o atributo clínica. Apresenta ainda os atributos nome e tipo. Todos os atributos são não nulos com exceção do tipo.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Nome	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Tipo	VARCHAR(1)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Clinica	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 24: Tabela representativa da entidade laboratório

O código SQL abaixo permite criar esta tabela:

```
CREATE TABLE `laboratorio` (
  `Id` int NOT NULL,
  `Nome` varchar(45) NOT NULL,
  `Tipo` varchar(1) DEFAULT NULL,
  `Clinica` int NOT NULL,
  PRIMARY KEY (`Id`),
  KEY `fk_Laboratorio_Clinica1_idx` (`Clinica`),
  CONSTRAINT `fk_Laboratorio_Clinica1` FOREIGN KEY (`Clinica`) REFERENCES `clinica` (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 25: Código de criação da Tabela Laboratório

5.2.10. Tabela Funcionário

A tabela Funcionário apresenta como chave primária o código do funcionário e tem como atributos não nulos o nome e laboratório, sendo este último uma chave estrangeira.




Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 Codigo_Funcionario	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Nome	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Laboratorio	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 26: Tabela representativa da entidade funcionário

Esta Tabela Funcionário foi criada pelo seguinte código SQL:

```
CREATE TABLE `funcionario` (
  `Codigo_Funcionario` int NOT NULL,
  `Nome` varchar(45) NOT NULL,
  `Laboratorio` int NOT NULL,
  PRIMARY KEY (`Codigo_Funcionario`),
  KEY `fk_Funcionario_Laboratorio1_idx` (`Laboratorio`),
  CONSTRAINT `fk_Funcionario_Laboratorio1` FOREIGN KEY (`Laboratorio`) REFERENCES `laboratorio` (`Id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 27: Código de criação da Tabela Funcionário

5.2.11. Tabela Clínica

A Tabela Clínica contém 2 atributos: o ID, como chave primária, e o nome.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
 Id	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
 Nome	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Figura 28: Tabela representativa da entidade clínica

Esta tabela foi criada através do seguinte código SQL:

```
CREATE TABLE `clinica` (  
  `Id` int NOT NULL,  
  `Nome` varchar(45) NOT NULL,  
  PRIMARY KEY (`Id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8
```

Figura 29: Código de criação da Tabela Clínica

5.3. Tradução das interrogações para SQL

De maneira a obter um funcionamento correto da BD, é necessário ter procedimentos que se adequem ao contexto atual de uma clínica. Para tal, tem de ser possível agendar e desmarcar testes, bem como poder alterar as informações relativas a um dado teste.

5.3.1. Agendar um teste

Para ser possível agendar um teste, tem de se ter acesso aos diferentes atributos da tabela de marcações, à exceção do ID. O ID de cada nova marcação será sempre dado pela incrementação do número de marcações na BD, de maneira a que cada ID seja único e não haja diferentes marcações com o mesmo identificador.

O procedimento para agendar um teste encontra-se na Figura 29.

```
delimiter $$  
CREATE PROCEDURE agendar_Testes(IN dataTeste DATETIME, matricula VARCHAR(45), posto INT, cliente INT)  
BEGIN  
  -- Declarar Variáveis  
  DECLARE i INT;  
  DECLARE ErroTransacao BOOL DEFAULT 0;  
  DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET ErroTransacao = 1;  
  
  START TRANSACTION;  
  
  -- ID Teste  
  SET i = (SELECT count(*) FROM marcacao);  
  
  INSERT  
    INTO marcacao (Id, Data_De_Realizacao, Matricula, Posto, Cliente)  
    VALUES (i+1, dataTeste, matricula, posto, cliente);  
  
  -- Verificação ocorrência de um erro  
  IF ErroTransacao THEN  
    BEGIN  
      SELECT `Não foi possível registar teste` AS Mensagem_Retorno;  
      ROLLBACK;  
    END;  
  ELSE  
    BEGIN  
      SELECT `Operação concluída com sucesso` AS Mensagem_Retorno;  
      COMMIT;  
    END;  
  END IF;  
END; $$
```

Figura 30: Procedure “agendar_Testes”

5.3.2. Desmarcar um teste

Se é possível marcar um teste, então também tem de ser possível desmarcar o mesmo. Com este procedimento estamos a considerar casos onde a realização de um teste já não poderá ser realizada.

```
delimiter $$
CREATE PROCEDURE desmarcar_Teste(IN marcacao_x INT)
BEGIN
    -- Declaração de handler para tratamento de Erros
    DECLARE ErroTransacao BOOL DEFAULT 0;
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET ErroTransacao = 1;

    -- Início Transação
    START TRANSACTION;

    -- Desmarcar Teste
    DELETE FROM marcacao
        WHERE Id = marcacao_x;

    -- Verificação ocorrência de um erro
    IF ErroTransacao THEN
        -- Desfazer operações realizadas
        BEGIN
            SELECT `Não foi possível desmarcar teste` AS Mensagem_Retorno;
            ROLLBACK;
        END;
    ELSE
        -- Confirmar as operações realizadas
        BEGIN
            SELECT `Operação concluída com sucesso` AS Mensagem_Retorno;
            COMMIT;
        END;
    END IF;
END;$$
```

Figura 31: *Procedure* “desmarcar_Teste”

5.3.3. Remarcar um teste

É possível alterar os campos dos atributos de uma marcação, caso seja necessário. Para isso terão de ser fornecidos todos os valores dos novos campos da tabela de marcação, sendo que apenas será alterada a marcação com a chave primária idêntica à chave passada como parâmetro.

Para remarcar um teste, é invocado o seguinte *Procedure*:

```

delimiter $$
CREATE PROCEDURE remarcar_Teste(IN marcacao_x INT, dataTeste DATETIME, matricula VARCHAR(45), posto INT, cliente INT)
BEGIN
    -- Declaração de handler para tratamento de Erros
    DECLARE ErroTransacao BOOL DEFAULT 0;
    DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET ErroTransacao = 1;

    START TRANSACTION;

    UPDATE marcacao
        SET Data_De_Realizacao = dataTeste,
            Matricula = matricula,
            Posto = posto,
            Cliente = cliente
        WHERE Id = marcacao_x;

    -- Verificação ocorrência de um erro
    IF ErroTransacao THEN
        BEGIN
            SELECT `Não foi possível remarcar teste` AS Mensagem_Retorno;
            ROLLBACK;
        END;
    ELSE
        BEGIN
            SELECT `Operação concluída com sucesso` AS Mensagem_Retorno;
            COMMIT;
        END;
    END IF;
END; $$

```

Figura 32: *Procedure* “remarcar_Teste”

5.3.4. Saber resultados de um dado teste

É importante poder saber qual o resultado de um dado teste. Para tal, deve-se invocar, no *procedure* abaixo, qual o cliente que realizou esse mesmo teste.

```

delimiter $$
CREATE PROCEDURE resultado_e_dono_Teste(IN teste_x INT)
BEGIN
    SELECT Nome, Resultado
        FROM teste T JOIN cliente C ON T.cliente = C.Numero_Cartao_Cidadao
        WHERE T.Id = teste_x;
END; $$

```

Figura 33: *Procedure* “resultado_e_dono_Teste”

5.3.5. Saber laboratório que analisou um dado teste

Para invocar este procedimento, que informa qual o laboratório que analisou um determinado teste, é necessário introduzir como parâmetro o número do teste referido. Desta forma, consegue-se obter os nomes do laboratório e do funcionário onde este foi realizado.


```

delimiter $$
CREATE PROCEDURE lab_e_funcionario_Teste(IN teste_x INT)
BEGIN
    SELECT L.Nome AS nomeLaboratorio, F.Nome AS nomeFuncionario
    FROM teste T JOIN funcionario F ON T.funcionario = F.Codigo_Funcionario
    JOIN laboratorio L ON F.laboratorio = L.Id
    WHERE T.Id = teste_x;
END;$$

```

Figura 34: *Procedure* “lab_e_funcionario_Teste”

5.3.6. Saber posto de recolha de um dado teste

O procedimento da Figura 34, informa o nome do posto que recolheu o material para realização de um determinado teste.

```

delimiter $$
CREATE PROCEDURE posto_Teste(IN teste_x INT)
BEGIN
    SELECT Nome
    FROM teste T JOIN posto P ON T.posto = P.ID
    WHERE T.Id = teste_x;
END;$$

```

Figura 35: *Procedure* “posto_Teste”

5.3.7. Dados pessoais de cliente que realizou um teste

O procedimento abaixo, devolve todos os dados relativos a um cliente que realizou um determinado teste.

```

delimiter $$
CREATE PROCEDURE dadosPessoa_Teste(IN teste_x INT)
BEGIN
    SELECT Nome, Data_Nascimento, Genero, Localidade, Codigo_Postal, Rua
    FROM teste T JOIN cliente C ON T.cliente = C.Numero_Cartao_Cidadao
    WHERE T.Id = teste_x;
END;$$

```

Figura 36: *Procedure* “dadosPessoa_Teste”

5.3.8. Testes realizados num dado laboratório

Para obter todos testes realizados num laboratório registado no sistema, representados pelo seu ID, invocou-se o seguinte *procedure*.

```
delimiter $$
CREATE PROCEDURE testes_Laboratorio(IN laboratorio_x INT)
BEGIN
    SELECT T.Id AS IdTeste
        FROM teste T JOIN funcionario F ON T.funcionario = F.Codigo_Funcionario
            JOIN laboratorio L On F.laboratorio = L.Id
        WHERE L.Id = laboratorio_x;
END;$$
```

Figura 37: *Procedure* “testes_Laboratorio”

5.3.9. Testes realizados por um funcionário

Com o *procedure* seguinte pretende-se obter todos os testes realizados por um funcionário registado no sistema, representado pelo seu ID.

```
delimiter $$
CREATE PROCEDURE testes_Funcionario(IN funcionario_x INT)
BEGIN
    SELECT Id
        FROM teste T
        WHERE T.Funcionario = funcionario_x;
END;$$
```

Figura 38: *Procedure* “testes_Funcionario”

5.3.10. Marcações num período

O *procedure* abaixo devolve todas as marcações agendadas para um dado período, compreendido entre duas datas diferentes. A marcação será representada pelo seu ID, a sua data de realização e a matrícula.

```
delimiter $$
CREATE PROCEDURE marcacoes_Periodo(inicio DATETIME, fim DATETIME)
BEGIN
    SELECT Id, Data_De_Realizacao, Matricula
        FROM marcacao
        WHERE Data_De_Realizacao >= inicio
            AND Data_De_Realizacao <= fim;
END;$$
```

Figura 39: *Procedure* “marcacoes_Periodo”

5.3.11. Pessoas infetadas numa faixa etária

Através do *procedure* seguinte pode-se obter o número de cartão de cidadão, o nome e a idade de todas as pessoas, pertencentes a uma determinada faixa etária que se encontram infetadas.

```
delimiter $$
CREATE PROCEDURE infetadas_faixaEtaria(inicio INT, fim INT)
BEGIN
    SELECT C.Numero_Cartao_Cidadao AS IdCliente, Nome, (YEAR(T.Data_Realizacao) - YEAR(C.Data_Nascimento)) AS Idade
    FROM teste T JOIN cliente C ON T.cliente = C.Numero_Cartao_Cidadao
    WHERE (YEAR(T.Data_Realizacao) - YEAR(C.Data_Nascimento) >= inicio)
        AND (YEAR(T.Data_Realizacao) - YEAR(C.Data_Nascimento) <= fim)
        AND Resultado = 'P';
END;$$
```

Figura 40: *Procedure* “infetadas_faixaEtaria”

5.3.12. Pessoas infetadas de um dado género

O *procedure* seguinte indica número de cartão de cidadão, o nome e o género de todas os clientes de um dado género que realizaram testes com resultado positivo.

```
delimiter $$
CREATE PROCEDURE infetadas_genero(IN genero_x VARCHAR(1))
BEGIN
    SELECT C.Numero_Cartao_Cidadao AS IdCliente, Nome, Genero
    FROM teste T JOIN cliente C on T.cliente = C.Numero_Cartao_Cidadao
    WHERE C.Genero = genero_x AND Resultado = 'P';
END;$$
```

Figura 41: *Procedure* “infetadas_Genero”

5.3.13. Número de testes realizados num laboratório

O *procedure* que permite obter o número total de testes que foram realizados num dado laboratório, independentemente do resultado, encontra-se representado na Figura 41.

```

CREATE PROCEDURE numero_testesGerais_Laboratorio(IN laboratorio_x INT)
BEGIN
    SELECT count(*) AS numeroTestesLaboratorio
    FROM teste T JOIN funcionario F ON T.funcionario = F.Codigo_Funcionario
        JOIN laboratorio L ON F.Laboratorio = L.Id
    WHERE L.Id = laboratorio_x;
END;$$

```

Figura 42: *Procedure* “numero_testesGerais_Laboratorio”

5.3.14. Número de testes realizados num laboratório, tendo em conta qual o seu resultado

O seguinte *procedure* indica o número de testes que foram realizados num dado laboratório para um determinado resultado.

```

delimiter $$
CREATE PROCEDURE numero_testesResult_Laboratorio(IN laboratorio_x INT, result_x VARCHAR(1))
BEGIN
    SELECT count(*) AS numeroTestesLaboratorioResult
    FROM teste T JOIN funcionario F ON T.funcionario = F.Codigo_Funcionario
        JOIN laboratorio L ON F.Laboratorio = L.Id
    WHERE L.Id = laboratorio_x AND Resultado = result_x;
END;$$

```

Figura 43: *Procedure* “numero_testesResult_Laboratorio”

5.3.15. Número de testes realizados num posto, independente do resultado

O procedimento da Figura 43 especifica o número total de testes que foram realizados num dado posto, independentemente do resultado desse mesmo teste.

```

delimiter $$
CREATE PROCEDURE numero_testesGerais_Posto(IN posto_x INT)
BEGIN
    SELECT count(*) AS numeroTestesPosto
    FROM teste T
    WHERE T.Posto = posto_x;
END;$$

```

Figura 44: *Procedure* “numero_testesGerais_Posto”

5.3.16. Número de testes realizados num posto, por resultado

O *procedure* seguinte indica o número de testes que foram realizados num dado posto para um resultado específico.

```
delimiter $$
CREATE PROCEDURE numero_testesResult_Posto(IN posto_x INT, result_x VARCHAR(1))
BEGIN
    SELECT count(*) AS numeroTestesPostoResultado
    FROM teste T
    WHERE T.Posto = posto_x AND T.Resultado = result_x;
END;$$
```

Figura 45: *Procedure* “numero_testesResult_Posto”

5.3.17. Tipo de teste dado um resultado

Dado um resultado, o *procedure* seguinte irá indicar qual o tipo de teste (‘M’ – monitorização, ‘I’ – imunidade, ‘C’ – teste à covid-19), a que esse resultado está associado.

```
delimiter $$
CREATE PROCEDURE tipoTeste_Result(result_x VARCHAR(1))
BEGIN
    SELECT tipo
    FROM teste
    WHERE resultado = result_x;
END;$$
```

Figura 46: *Procedure* “tipoTeste_Result”

5.3.18. Pessoas que realizaram um tipo de teste

O *procedure* apresenta-nos o número de cartão de cidadão, o nome, a data de nascimento, o género, a localidade, o código postal e a rua de todas as pessoas que realizaram um dado tipo de teste.

```

delimiter $$
CREATE PROCEDURE pessoas_tipoTeste(IN tipo_x VARCHAR(1))
BEGIN
    SELECT Numero_Cartao_Cidadao, Nome, Data_Nascimento, Genero, Localidade, Codigo_Postal, Rua
    FROM teste T JOIN cliente C ON T.cliente = C.Numero_Cartao_Cidadao
    WHERE Tipo = tipo_x;
END;$$

```

Figura 47: *Procedure* “pessoas_tipoTeste”

5.3.19. Pessoas que realizaram um tipo de teste num dado período

Dado um certo período de tempo, este procedimento, indica quais as pessoas que realizaram um certo tipo de teste nesse mesmo período. Serão apresentados o número de cartão de cidadão, o nome e o género dessas pessoas.

```

delimiter $$
CREATE PROCEDURE pessoas_tipoTeste_Periodo(IN tipo_x INT, inicio DATETIME, fim DATETIME)
BEGIN
    SELECT Numero_Cartao_Cidadao, Nome, Genero
    FROM teste T JOIN cliente C ON T.cliente = C.Numero_Cartao_Cidadao
    WHERE Tipo = tipo_x
    AND T.Data_Realizacao >= inicio
    AND T.DATA_Realizacao <= fim;
END;$$

```

Figura 48: *Procedure* “pessoas_tipoTeste_Periodo”

5.3.20. Pessoas que realizaram um Teste num dado Posto

Para um determinado posto, o *procedure* seguinte indica quais as pessoas que realizaram qualquer tipo de teste nesse posto, sendo que serão apresentados o número de cartão de cidadão, o nome e o género dessas pessoas.

```

delimiter $$
CREATE PROCEDURE pessoas_testePosto(IN posto_x INT)
BEGIN
    SELECT Numero_Cartao_Cidadao, Nome, Genero
    FROM teste T JOIN cliente C ON T.cliente = C.Numero_Cartao_Cidadao
    WHERE Posto = posto_x;
END;$$

```

Figura 49: *Procedure* “pessoas_testePosto”

5.3.21. Pessoas de uma dada localidade que realizaram um teste

Dada uma certa localidade, obtém-se, através do seguinte *procedure*, as pessoas que realizaram qualquer tipo de teste e que são residentes nessa localidade, sendo que serão apresentados o número de cartão de cidadão, o nome e o género das mesmas.

```
delimiter $$
CREATE PROCEDURE pessoas_TestLocalidade(IN localidade_x VARCHAR(45))
BEGIN
    SELECT Numero_Cartao_Cidadao, Nome, Genero
    FROM teste T JOIN cliente C ON T.cliente = C.Numero_Cartao_Cidadao
    WHERE C.Localidade = localidade_x;
END;$$
```

Figura 50: *Procedure* “pessoas_TestLocalidade”

5.4. Escolha, definição e caracterização de índices

Num SGBD os índices são utilizados para facilitar a procura de dados em tabelas de grandes dimensões, pois determinam de forma rápida a posição dos dados indexados, não necessitando de percorrer as tabelas de forma sequencial. Este processo permite reduzir o número de acessos à memória central e o tempo de procura.

As chaves primárias e estrangeiras foram usadas como índices, uma vez que é requisito do mecanismo de armazenamento do MySQL, o InnoDB, que requer a indexação das chaves primárias e estrangeiras. Estes índices são criados de forma automática numa tabela. Optou-se por não criar mais nenhum índice, pois a dimensão da BD não o justifica. Sendo esta uma BD de pequena dimensão, a obtenção de dados já é realizada de uma forma rápida. Neste caso, a inserção de mais índices iria prejudicar o desempenho da BD, pois o tempo de inserção e alteração de dados poderia aumentar.

5.5. Estimativa do espaço em disco das bases de dados e taxa de crescimento anual

Um passo importante no desenvolvimento de uma BD é perceber o seu desempenho no futuro, nomeadamente o espaço em disco que irá ocupar. Para tal, calculou-se o tamanho em disco de cada uma das tabelas da BD, tendo por base uma estimativa de crescimento previsto do tamanho das mesmas com a introdução de dados. A estimativa é calculada tendo o domínio dos atributos de cada tabela.

Assumindo, o máximo de caracteres possíveis para os atributos textuais e a não existências de valores nulos, calculou-se os seguintes resultados por cada inserção numa tabela em *bytes*, segundo a documentação do MySQL.

Tabela 8: Tamanho médio ocupado por cada registo em cada tabela

Tabela	Tamanho médio (<i>bytes</i>)
Cliente	193
Contacto Cliente	24
Marcação	66
Posto	239
Contacto Posto	24
Posto Opera Horário	8
Horário	37
Teste	165
Laboratório	56
Funcionário	54
Clínica	50

Após o povoamento inicial, onde foram introduzidos 1 clínica, 7 postos, 7 contactos posto, 5 laboratórios, 12 funcionários, 10 clientes, 10 contactos cliente, 8 horários, 49 horários opera posto, 10 marcações e 5 testes na BD, obteve-se uma ocupação em disco de aproximadamente 7162 *bytes*.

Num futuro, assume-se que apenas os clientes, os contactos dos clientes, as marcações e os testes emitidos poderão ter a sua quantidade alterada na BD.

De acordo com <https://www.sns.gov.pt/noticias/2020/10/09/testes-covid-19-novo-posto-fixo-em-lisboa/>, um posto tem uma capacidade de realização por volta de 1 500 testes diários.

Desta forma, assume-se que podem ser realizadas 1 500 marcações em cada um dos 7 postos, o que perfaz um total de 10 500 testes emitidos por dia. Ao fim de um ano, serão ocupados 252,945 *megabytes* (MB), referentes à Tabela Marcação e 632,362 MB relativos à Tabela Teste.

Admitindo a possibilidade de, por ano, serem registados 383 250 novos clientes (10% do número de marcações de testes do primeiro ano), o que se reflete num total de aproximadamente 74 MB de espaço em disco. Este valor será acrescido de mais 10 MB para os contactos dos clientes.

Em suma, no fim de um ano, serão necessários cerca de 969 MB de espaço em disco para suportar a BD. Contudo, tendo em conta que determinados campos podem ser nulos ou que nem todos podem utilizar os caracteres máximos disponíveis, este valor pode ser inferior.

5.6. Definição e caracterização das vistas de utilização em SQL

Para facilitar o acesso aos dados que podem ser encontrados em diferentes tabelas da BD, podem ser definidas vistas.

Uma vista pode ser considerada uma tabela virtual constituída por linhas e colunas com dados provenientes de tabelas relacionadas em *queries*. Os dados apresentados pela vista são gerados de forma dinâmica no momento em que a vista é invocada.

Neste projeto foram definidas as seguintes vistas:

5.6.1. Vista resultados de todos os testes realizados

Esta vista apresenta o nome da pessoa que realizou o teste, bem como o resultado deste, constituindo, assim, uma forma mais perceptível de saber qual o resultado de cada pessoa na realização do teste.

```
114 CREATE VIEW vista_Resultados_Testes AS
115     SELECT Nome, Resultado
116     FROM teste T JOIN cliente C ON T.cliente = C.Numero_Cartao_Cidadao;
```

Figura 51: View “vista_Resultados_Testes”

5.6.2. Vista pessoas infetadas

Uma das *views* mais importantes nesta BD é a das pessoas infetadas, que apresentará uma lista com o nome e género das pessoas que testaram positivo.

```
190 CREATE VIEW pessoas_infetadas AS
191     SELECT Nome, Genero
192     FROM teste T JOIN cliente C ON T.cliente = C.Numero_Cartao_Cidadao
193     WHERE T.Resultado = 'P';
```

Figura 52: View “pessoas_infetadas”

5.6.3. Vista todas as marcações efetuadas

No âmbito deste trabalho, poderia ser útil aceder a todas as marcações registadas no sistema, sabendo os dados respetivos a cada uma das marcações.

```
326 • CREATE VIEW marcaoes_Efetuadas AS
327     SELECT *
328     FROM marcacao
```

Figura 53: View “marcaoes_Efetuadas”

5.7. Revisão do sistema implementado

Para finalizar o processo de desenvolvimento da BD é necessária uma última validação do modelo físico criado pelos utilizadores finais da mesma.

Depois de garantida a aprovação por parte destes, o projeto fica concluído e pronto a ser implementado num contexto real.

6. Conclusão

No contexto atual, é fundamental gerir de forma eficiente a pandemia instalada, pelo que a aplicação de bases de dados eficazes e bem estruturadas é imperativa, de modo a prevenir as filas nos centros de marcação e colheita de testes virológicos. Assim sendo, a realização deste trabalho tinha como principal objetivo o desenvolvimento de uma BD capaz de facilitar o processo de marcação de testes à covid-19, evitando as filas dos clientes, bem como armazenar os resultados dos testes de cada cliente de forma segura.

O alcance destes objetivos foi facilitado pela criação de oito entidades estruturais na BD e pelas relações existentes entre as mesmas.

A relação entre as entidades marcação, caracterizada por um ID, data de realização e matrícula, posto, que apresenta um nome, ID, morada e contactos e cliente, permite organizar todos os clientes que realizaram uma marcação e que vão realizar um teste num determinado horário.

A entidade cliente, caracterizada pelo número de cartão de cidadão, data de nascimento, nome, género, morada e contacto, aliada ainda à entidade teste, que contém um ID, tipo (monitorização, imunidade e à covid-19), datas de realização e análise, resultado (positivo, negativo ou inconclusivo) e observações, permite conhecer os resultados de um teste e associá-lo a uma pessoa.

Além disto, é ainda possível identificar o laboratório (ID, nome e tipo), bem como o funcionário (ID e nome), através dos relacionamentos destas duas entidades com a entidade teste.

Com todos os objetivos alcançados faz-se um balanço positivo do trabalho desenvolvido. Os *triggers* criados permitem o controlo dos dados introduzidos. Através dos vários procedimentos e das vistas criados, pode-se obter e inserir todos os valores para satisfazerem os requisitos dos utilizadores.

Contudo, o povoamento da BD poderia ter sido mais extenso, para permitir uma melhor análise e teste dos vários procedimentos implementados, verificando a eficiência da BD.

Referências Bibliográficas

Governo dos Açores, 2020. *ENTIDADES CONVENCIONADAS COM A REGIÃO AUTÓNOMA DOS AÇORES PARA DESPISTE À INFEÇÃO POR CORONAVÍRUS SARS-CoV-2* (COVID-19). [Online]

Available at: https://destinoseguro.azores.gov.pt/wp-content/uploads/2020/10/Listagem-de-Laboratorios-e-Postos-de-colheita_19102020.pdf

[Acedido em dezembro 2020].

OLIBONI, D., 2016. *O que é um SGBD?*. [Online]

Available at: <https://www.oficinadanet.com.br/post/16631-o-que-e-um-sgbd>

[Acedido em dezembro 2020].

SNS, 2020. *Testes Covid-19 | Posto fixo em Lisboa*. [Online]

Available at: <https://www.sns.gov.pt/noticias/2020/10/09/testes-covid-19-novo-posto-fixo-em-lisboa/>

[Acedido em dezembro 2020].

Universidade Nova de Lisboa, 2016. *Modelação conceptual, logica e fisica*. [Online]

Available at: <https://www.studocu.com/pt/document/universidade-nova-de-lisboa/bases-de-dados/resumos/modelacao-conceptual-logica-e-fisica/2058195/view>

[Acedido em dezembro 2020].

Wellington, 2011. *Conceitos e criação de views no SQL Server*. [Online]

Available at: <https://www.devmedia.com.br/conceitos-e-criacao-de-views-no-sql-server/22390>

[Acedido em dezembro 2020].

Lista de Siglas e Acrónimos

BD	Base de Dados
DGS	Direção Geral de Saúde
ID	Identificador
Diagrama ER	Diagrama de Entidades-Relacionamentos
ML	Modelo Lógico
SGBD	Sistema De Gestão De Bases De Dados
MB	<i>Megabytes</i>

I. Anexo 1: Código do Povoamento Inicial

```
USE `clinicabd`;
```

```
INSERT INTO clinica
```

```
(id,nome)
```

```
VALUES
```

```
(1,'Synlabhealth II');
```

```
INSERT INTO posto
```

```
(Id,Localidade,Codigo_Postal,Rua,Nome,email,Clinica)
```

```
VALUES
```

```
(1,'Almeirim','2080-141','Rua General Humberto Delgado','Posto de Análises Clínicas do  
Almeirim',NULL,1),
```

```
(2,'Amadora','2700-458','Largo Doutor Dário Gandra Nunes','Posto de Análises Clínicas da  
Amadora','posto.venteira@synlab.pt',1),
```

```
(3,'Barreiro','2830-003','Avenida do Bocage','Posto de Análises Clínicas do Barreiro',NULL,1),
```

```
(4,'Braga','4710-102','Rua Ambrósio Santos','Posto de Análises Clínicas de  
Braga','posto.ambrosio-santos@synlab.pt',1),
```

```
(5,'Coimbra','3020-479','Praceta Professor Robalo Cordeiro','Posto de Análises Clínicas de  
Coimbra','laboratorio.coimbra@synlab.pt',1),
```

```
(6,'Covilhã','6200-506','Av.a Infante D. Henrique','Posto de Análises Clínicas da  
Covilhã',NULL,1),
```

```
(7,'Évora','7005-259','Praceta Horta do Bispo','Posto de Análises Clínicas de  
Évora','laboratorio.evora@synlab.pt',1);
```

```
INSERT INTO contactoposto
```

```
(Telefone,Posto)
```

```
VALUES
```

```
('243592604',1),
```

```
('914155759',2),
```

```
('910728308',3),
```

```
('253248805',4),
```

```
('227860743',4),
```

```
('935465241',4),
```

```
('239701512',5),
```

```
('275313383',6),
```

```
('266759590',7);
```

```
INSERT INTO laboratorio
      (Id, Nome, Tipo, Clinica)
```

```
VALUES
```

```
(1, 'Hemobiolab', 'E', 1),
(2, 'AVELAB', 'E', 1),
(3, 'Hormofuncional', 'E', 1),
(4, 'Cintramédica', 'E', 1),
(5, 'Labocentro', 'I', 1);
```

```
INSERT INTO funcionario
```

```
      (Codigo_Funcionario, Nome, Laboratorio)
```

```
VALUES
```

```
(1, 'Adelino', 1),
(2, 'Alfredo', 1),
(3, 'Alexandre', 1),
(4, 'Dinis', 2),
(5, 'Duarte', 2),
(6, 'Emílio', 3),
(7, 'Gilberto', 3),
(8, 'Gustavo', 3),
(9, 'Inácio', 4),
(10, 'Leopoldo', 5),
(11, 'Marcelo', 5),
(12, 'Mauro', 5);
```

```
INSERT INTO cliente
```

```
VALUES
```

```
(1, '1972/12/14', 'Simão', 'M', 'Mafra', '2640-465', 'Largo Coronel Brito Gorjão'),
(2, '1967/12/27', 'Telmo', 'M', 'Lourinhã', '2534-500', 'Praça José Máximo da Costa'),
(3, '2000/11/06', 'Tiago', 'M', 'Almeirim', '2080-142', 'Rua dos 3 Vales'),
(4, '1953/06/23', 'Vasco', 'M', 'Amadora', '2700-459', 'Largo Doutor Dário Gandra Nunes'),
(5, '1953/12/09', 'Raul', 'M', 'Amadora', '2700-457', 'Rua Real Fábrica do Vidro'),
(6, '1933/12/12', 'Ana', 'F', 'Barreiro', '2830-703', 'R. Prof. Francisco Gentil'),
(7, '1982/12/17', 'Augusta', 'F', 'Braga', '4710-112', 'Rua S. Tomás de Aquino'),
(8, '1980/12/07', 'Carlota', 'F', 'Coimbra', '3020-458', 'Largo Coronel Brito Gorjão'),
(9, '1980/12/22', 'Cristina', 'F', 'Covilhã', '6200-512', 'Rua Dr. António José de Almeida'),
(10, '1962/12/03', 'Eva', 'F', 'Évora', '7005-250', 'Rua Assis Leão');
```

```
INSERT INTO contactocliente
```

```
      (Telefone, Cliente)
```

```
VALUES
```

```
('930551476', 1),
('961773477', 2),
('968895013', 3),
```

```

('961773477',4),
('918851034',5),
('935465241',6),
('966148612',7),
('930568014',8),
('930570152',8),
('935465241',10);

```

INSERT INTO horario

(Id,Inicio_Manha,Fim_Manha,Inicio_Tarde,Fim_Tarde,Dia)

VALUES

```

(1,NULL,NULL,'14:30:00','19:00:00','Segunda-Feira'),
(2,NULL,NULL,'14:00:00','18:30:00','Terça-Feira'),
(3,NULL,NULL,'14:00:00','18:30:00','Quarta-Feira'),
(4,NULL,NULL,'14:00:00','18:30:00','Quinta-Feira'),
(5,NULL,NULL,'14:00:00','18:30:00','Sexta-Feira'),
(6,'09:30:00','12:30:00',NULL,NULL,'Sábado-Feira'),
(7,NULL,NULL,NULL,NULL,'Domingo-Feira'),
(8,'10:00:00','13:00:00',NULL,NULL,'Domingo-Feira');

```

INSERT INTO postoooperahorario

(Posto,Horario)

VALUES

```

(1,1),
(1,2),
(1,3),
(1,4),
(1,5),
(1,6),
(1,7),
    (2,1),
    (2,2),
    (2,3),
    (2,4),
    (2,5),
    (2,6),
    (2,7),
        (3,1),
        (3,2),
        (3,3),
        (3,4),
        (3,5),
        (3,6),
        (3,8),

```



```

(4,1),
(4,2),
(4,3),
(4,4),
(4,5),
(4,6),
(4,8),
(5,1),
(5,2),
(5,3),
(5,4),
(5,5),
(5,6),
(5,7),
(6,1),
(6,2),
(6,3),
(6,4),
(6,5),
(6,6),
(6,7),
(7,1),
(7,2),
(7,3),
(7,4),
(7,5),
(7,6),
(7,8);

```

INSERT INTO marcacao

VALUES

```

(1,'2020-11-30 16:30:00','44-FH-82',4,1),
(2,'2020-11-30 16:40:00','43-SL-56',4,2),
(3,'2020-11-30 16:40:00','82-SP-75',1,3),
(4,'2020-11-30 16:40:00','73-RI-57',2,4),
(5,'2020-11-30 16:50:00','18-SI-97',2,5),
(6,'2020-11-30 16:40:00','21-SK-81',3,6),
(7,'2020-11-30 16:50:00','37-SL-19',4,7),
(8,'2020-11-30 16:40:00',NULL,5,8),
(9,'2020-11-30 16:40:00',NULL,6,9),
(10,'2020-11-30 16:40:00',NULL,7,10);

```

```

INSERT INTO teste
VALUES
(1,'2020-11-30 16:34:00','C','2020-12-01 10:30:00','P',NULL,4,1,1),
(2,'2020-11-30 16:47:00','C','2020-12-01 10:30:00','N',NULL,4,2,2),
(3,'2020-11-30 16:44:00','C','2020-12-01 10:30:00','P',NULL,1,3,3),
(4,'2020-11-30 16:41:00','C','2020-12-01 11:30:00','I','Baixa concentração de globulos
vermelhos',2,4,1),
(5,'2020-11-30 16:52:00','I','2020-12-01 10:30:00','P',NULL,2,5,4);

```