Universidade do Minho
Mestrado Integrado em Engenharia Informática
Licenciatura em Ciências da Computação

Unidade Curricular de Bases de Dados

Ano Lectivo de 2020/2021

Farmácia NTBH

Bruno Dias a89583

Guilherme Pereira a89479

Luís Sousa a89597

Pedro Barbosa a89529

dezembro 2020

BD

Data de Recepção	
Responsável	
Avaliação	
Observações	

Farmácia NTBH

Bruno Dias a89583

Guilherme Pereira a89479

Luís Sousa a89597

Pedro Barbosa a89529

dezembro 2020

Resumo

Este relatório foi realizado no âmbito da unidade curricular de Base de Dados, descrevendo de forma sucinta mas também detalhada todo o processo de criação de uma base de dados utilizada para gerir o funcionamento de uma farmácia. Este procedimento inicia-se com a conceção seu modelo conceptual e continua com a implementação do seu modelo físico e trabalho sobre o mesmo.

A primeira parte deste relatório diz respeito à definição do sistema onde fizemos uma contextualização de aplicação do sistema, uma fundamentação da implementação da base de dados e analisamos a viabilidade de todo o processo.

Numa segunda fase do projeto fizemos um levantamento e análise dos requisitos necessários para o correto funcionamento da nossa base de dados e também realizamos a validação dos mesmos.

De seguida, e tendo em conta tudo aquilo que definimos na segunda fase do nosso trabalho, prosseguimos para a criação do modelo conceptual onde definimos as diferentes entidades do nosso projeto bem como a maneira como estas se relacionam entre si. Após a conceção do nosso modelo conceptual procedemos à validação do mesmo.

A partir do modelo em cima referido, o nosso grupo construiu então o modelo lógico da nossa farmácia tendo este sido validado através da normalização, interrogações do utilizador e com as transações estabelecidas.

Posteriormente procedemos então à implementação física da nossa base de dados, onde foi feita a escolha do nosso sistema de gestão da base de dados relacionais e a respetiva tradução do esquema lógico para o sistema escolhido.

Por fim, procedemos às conclusões que retiramos após a realização de todas estas fases do nosso projeto e também daquilo que retiramos e esperamos do trabalho futuro.

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

Palavras-Chave: Bases de Dados, Levantamento de Requisitos, Entidades, Relacionamento, Modelo Conceptual, Modelo Lógico, Modelo Físico, MySQL, brModelo, MySQLWorkbench.

Índice

1. Definição do Sistema 8

1.1 Contexto de aplicação do sistema 8

1.2 Fundamentação da implementação da base de dados 8

1.3 Análise da viabilidade do processo 9

2. Levantamento e Análise de Requisitos 10

2.1 Método de levantamento e de análise de requisitos adotado 10

2.2 Requisitos levantados 10

2.2.1 Requisitos de descrição 11

2.2.2 Requisitos de exploração 12

2.2.3 Requisitos de controlo 13

2.3 Análise e validação geral dos requisitos 14

3. Modelação Conceptual 14

3.1 Apresentação da abordagem de modelação realizada 14

3.2 Identificação e caracterização das entidades 14

3.3 Identificação e caracterização dos relacionamentos 15

3.4 Identificação e caracterização da associação dos atributos com as entidades e relacionamentos. 16

3.5 Detalhe ou generalização de entidades 19

3.6 Apresentação e explicação do diagrama ER 19

3.7 Validação do modelo de dados produzido 20

4. Modelação Lógica 23

4.1 Construção e validação do modelo de dados lógico 23

4.1.1 Entidades Fortes 23

4.1.2 Entidades Fracas 24

4.1.3 Relação de um para muitos (1:n) 24

4.1.4 Relacionamento de muitos para muitos (n:m) 25

4.2 Desenho do modelo lógico 26

4.3 Validação do modelo com interrogações do utilizador 26

4.4 Revisão do modelo lógico produzido 29

5. Implementação Física 30

5.1 Seleção do sistema de gestão de bases de dados 30

5.2 Tradução do esquema lógico para o sistema de gestão de bases de dados escolhido em SQL 30

5.3 Tradução das interrogações do utilizador para SQL (alguns exemplos) 36

5.4 Escolha, definição e caracterização de índices em SQL (alguns exemplos) 40

5.5 Estimativa do espaço em disco da base de dados e taxa de crescimento 41

5.6 Definição e caracterização das vistas de utilização em SQL (alguns exemplos) 44

5.7 Definição e implementação de “triggers” em SQL 45

5.8 Revisão do sistema implementado 46

6. Conclusões e Trabalho Futuro 47

Anexos 48

Índice de Figuras

Figura 1 - Modelo Concetual 19
Figura 2 - Modelo Lógico 25
Figura 3 - Código SQL para a criação da tabela cliente 30
Figura 4 - Código SQL para a criação da tabela funcionário 31
Figura 5 - Código SQL para a criação da tabela fatura 32
Figura 6 - Código SQL para a criação da tabela LinhaFatura 33
Figura 7 - Código SQL para a criação da tabela Produto 34
Figura 8 - Código SQL para consultar dados de um cliente 35
Figura 9 - Código SQL para consultar faturas de um cliente 35
Figura 10 - Código SQL para verificar a disponibilidade de um produto 36
Figura 11 - Código SQL para procurar um cliente 36
Figura 12 - Código SQL para consultar os diversos produtos farmacêuticos 37
Figura 13 - Código SQL para consultar o vencimento de um funcionário 37
Figura 14 - Código SQL para consultar quantas vendas foram feitas num certo período de tempo 37
Figura 15 - Código SQL para consultar o total faturado pela farmácia num certo período de tempo 38
Figura 16 - Código SQL para consultar o funcionário que faturou mais num certo período de tempo 38
Figura 17 - Código SQL para consultar os ‘n’ produtos farmacêuticos mais vendidos num certo período de tempo 38
Figura 18 - Código SQL para consultar o cliente que mais compras efetuou num determinado período de tempo 39
Figura 19 - Código SQL para a criação dos índices Nome (no funcionário e no cliente), Designação (no produto) e dataFatura (na fatura) 40
Figura 20 - Código SQL para a implementação da vista das faturas e clientes associados 44
Figura 21 - Vista das faturas e clientes associados 44
Figura 22 - Código SQL para a implementação da vista das linhas de cada fatura e dos produtos associados 45
Figura 23 - Vista das linhas de cada fatura e dos produtos associados 45
Figura 24 - Trigger ativado na inserção de uma linha na linhaFatura 46

Índice de Tabelas

Tabela 1 - Entidades 15
Tabela 2 - Relacionamentos entre Entidades 16
Tabela 3 - Atributos 17
Tabela 4 - Tamanho de cada entrada na respetiva tabela 40

1. Definição do Sistema

1.1. Contexto de Aplicação do Sistema

Nos dias de hoje, com a evolução da tecnologia também as diferentes empresas tendem a evoluir de modo a se manterem competitivas nos seus respetivos mercados e tentarem, ao máximo, uma monopolização dos mesmos.

Há mais de duas décadas que as bases de dados se tornaram uma peça fundamental nos sistemas de informação e segurança sendo de vital importância para todas as empresas. Por exemplo, o armazenamento digital de dados permite acesso rápido e concorrente à informação e também possibilita a descoberta de padrões recorrentes.

Foi com estes ideais em mente que a empresa Farmácias BemEstar requisitou os serviços do nosso grupo de trabalho. A empresa, para além de outras farmácias bem sucedidas ao longo do país como a Farmácia STK em Chelas ou a Farmácia MS em Gaia, pretende agora abrir uma nova farmácia, a Farmácia NTBH, na cidade de Braga, na freguesia de São Victor, junto a um dos pólos da Universidade do Minho.

Tendo em conta a importância das empresas se manterem evoluídas a nível tecnológico como foi acima referido, a empresa requisitou os nossos serviços de forma a ser implementado um sistema de base de dados que lhe permita ser mais competitiva no mercado farmacêutico e diferenciar-se de todas as outras empresas farmacêuticas do país.

1.2. Fundamentação da implementação da base de dados

O corpo administrativo da empresa Farmácias BemEstar, devido à evolução tecnológica e à constante preocupação com o bem estar dos seus clientes, pretende desenvolver um Sistema de Gestão de Base de Dados. Desta forma, é possível

minimizar quaisquer problemas de organização e de recolha de dados que possam ocorrer.

Hoje em dia, o armazenamento digital de dados permite uma resposta rápida aos pedidos de informação, uma melhor gestão dos mesmos e flexibilidade. Pode-se assim concluir que é do maior interesse de qualquer empresa que esteja a iniciar ou a expandir o seu negócio implementar este tipo de tecnologias de forma a terem uma maior organização e eficácia.

Com esta implementação será possível, para a empresa, por exemplo, gerir todas as compras que são efetuadas pelos seus clientes bem como verificar estatisticamente os seus lucros e rapidamente detetar problemas que estejam a ocorrer, conseguindo, assim, reunir as condições necessárias para o resolver.

1.3. Análise da viabilidade do projeto

Com todo o problema da nova realidade que se vive no mundo, uma pandemia mundial, a empresa Farmácias BemEstar procurou o desenvolvimento de um novo sistema que permitisse que a empresa se conseguisse diferenciar de outras que competiam no seu mercado. Desta forma, o seu corpo administrativo recorreu aos nossos serviços para a implementação de uma base de dados que lhes permitisse dar uma rápida e mais organizada resposta aos seus clientes.

O desenvolvimento deste novo sistema tem como principal objetivo servir de suporte ao crescimento desta empresa de Farmácias. A curto prazo deve substituir de uma forma eficaz o antigo sistema por esta utilizado e a longo prazo deve servir de sustento evitando a sua substituição num curto espaço de tempo. Cientes de que este sistema se deve manter funcional e otimizado durante todo o processo de renovação da empresa concluímos que será uma mais valia para a mesma, sendo totalmente viável.

2. Levantamento e Análise de Requisitos

2.1. Método de Levantamento e de Análise de Requisitos Adotado

Para o levantamento dos requisitos, o nosso grupo reuniu com a direção da empresa Farmácias BemEstar e visitou as outras farmácias que esta possui de forma a perceber melhor a maneira como estas funcionavam e proceder ao esclarecimento de qualquer dúvida existente de ambas as partes.

2.2. Requisitos Levantados

Após uma visita a ambas as farmácias da empresa o nosso grupo ficou a perceber na plenitude o seu funcionamento e todas as dúvidas, por mais pequenas que fossem, foram esclarecidas com a direção.

As farmácias da empresa BemEstar encontram-se abertas 24 horas por dia e 365 dias por ano (366, se este for bissexto).

Em relação aos clientes, se for a primeira vez que estes realizam uma compra na farmácia, o funcionário irá criar uma ficha do cliente que irá conter o seu nome, o NIF, o telemóvel, o e-mail, a palavra chave e a morada (por sua vez constituída por cidade, código postal e rua).

Se o cliente quiser comprar um produto sujeito a receita médica esta terá obrigatoriamente de ser apresentada pelo mesmo quer em papel quer numa mensagem recebida no seu telemóvel.

A venda de produtos pressupõe a emissão de uma fatura no sistema por parte do funcionário. Às faturas emitidas estará associado a sua data de emissão, o desconto, o IVA, o preço total e o código das receitas.

Quanto aos produtos farmacêuticos adquiridos pelos clientes é estritamente necessário saber a quantidade por embalagem, o número de embalagens existentes em stock, a designação, o preço, o modo de administração, o laboratório onde são fabricados, se é necessária receita médica, a validade e a quantidade por dose.

Em relação aos funcionários que trabalham na empresa BemEstar, a direção pretende ter as seguintes informações sobre cada um, o seu e-mail, o telemóvel, o

NISS, o IBAN, a palavra chave, o nome, o salário e a morada (constituída por cidade, código postal e rua).

2.2.1. Requisitos de Descrição

Cliente

Um cliente necessita de estar registado no sistema para assim ser identificado e efetuar as suas compras na farmácia. Este registo é feito aquando da primeira vez que o cliente efetua uma compra na farmácia. A cada cliente é atribuído um id único e registado dados como o seu nome, NIF, número de telemóvel, email, palavra chave e morada (que inclui cidade, código postal e rua).

Fatura

Cada fatura terá um id único, tal como acontece com os clientes. A descrição de cada fatura irá conter a data da mesma, o desconto, o IVA, o preço total da compra e o código das receitas utilizadas pelo cliente.

Produto Farmacêutico

Cada produto terá um id único. A cada produto estará associado a quantidade por embalagem, as embalagens existentes em stock, a sua designação, o seu preço de venda, o modo de administração, o laboratório onde foi fabricado, um indicador caso seja necessária receita médica, a validade do mesmo e a quantidade por dose.

Funcionário

Tal como as outras entidades também o funcionário terá um id único. Para além disso as informações necessárias sobre o funcionário da farmácia são o seu email, o número de telemóvel, a sua password, o seu número de identificação da segurança social (NISS), o seu IBAN para ser possível fazer a transferência do seu salário, a palavra chave, o seu nome, o salário que recebe e a sua morada (que inclui cidade, código postal e rua).

2.2.2. Requisitos de exploração

Cliente

Um cliente da farmácia deve conseguir realizar as seguintes tarefas:

- Consultar os seus dados
- Consultar as suas faturas
- Consultar o stock da farmácia para saber se um determinado produto farmacêutico se encontra disponível

Funcionário

Um funcionário da farmácia deve conseguir realizar as seguintes tarefas:

- Procurar clientes
- Consultar os diversos produtos farmacêuticos

Administrador

O administrador da farmácia deve conseguir realizar as seguintes tarefas:

- Consultar dados dos funcionários
  - Consultar os dados de pagamento de vencimentos aos funcionários
- Consultar as faturas
  - Consultar quantas vendas foram feitas num certo período de tempo
  - Consultar o total faturado pela farmácia num certo período de tempo
  - Consultar quem faturou mais num certo período de tempo
  - Consultar os 'n' produtos farmacêuticos mais vendidos
- Consultar os dados dos clientes da farmácia
  - Consultar o cliente que mais compras efetuou num certo período de tempo
- Consultar os produtos farmacêuticos

2.2.3. Requisitos de controlo

Cliente

O Cliente da farmácia deve conseguir:

- Alterar as suas informações pessoais

Funcionário

O Funcionário deve conseguir:

- Registar novos clientes
- Alterar as informações pessoais dos clientes já existentes
- Registar novas faturas

Administrador

O Administrador deve conseguir:

- Registar novos funcionários
- Adicionar as informações dos novos funcionários registados
- Alterar os dados pessoais dos funcionários já existentes
- Apagar faturas
- Inserir produtos farmacêuticos
- Alterar os dados dos produtos farmacêuticos

2.3. Análise e Validação Geral dos Requisitos

Os requisitos de descrição, exploração e controlo acima referidos revelam ser bastante descritivos acerca da maneira como a empresa Farmácias BemEstar funciona, uma vez que conseguem cobrir e descrever todas as componentes que constituem o ambiente em que esta se encontra inserida e descrevem detalhadamente as interações entre estes. Assim, torna-se evidente o papel que cada um dos utilizadores da base de dados vai desempenhar e o que se pretende com esta.

Desta forma, e após nova reunião com o corpo administrativo da empresa recebemos o seu avante para podermos prosseguir com o nosso trabalho.

3. Modelação Conceptual

3.1. Apresentação da Abordagem de Modelação Realizada

Após uma análise de requisitos, avançamos para a construção de um modelo conceptual , que representa a base de dados de forma independente dos detalhes de implementação, tendo em conta as relações entre entidades e os dados que cada uma armazena, de forma a ter um modelo coerente.

Optamos por uma abordagem conceitual focada na emissão e análise de faturas após a compra de produtos cujas entidades e relacionamentos destinam-se a facilitar a compreensão e a análise do sistema virtual elaborado.

3.2. Identificação e Caracterização das Entidades

De maneira a começar a construção do nosso modelo começamos por definir os tipos de entidades existentes e as suas possíveis relações.
Assim surge a entidade Cliente, onde o nosso modelo se inicia, que efetua a compra de Produtos obtendo então uma Fatura, desta forma conseguimos relacionar Faturas

com Produtos que são duas das nossas entidades modeladas. Prosseguindo com a análise de requisitos temos ainda o Funcionário uma vez que é necessária uma entidade para o atendimento de clientes e para a emissão da fatura correspondente. Obtemos então:

Entidade	Descrição
Cliente	Termo geral para descrever as pessoas responsáveis pela compra de produtos
Fatura	Termo geral que descreve a fatura que se obtém após uma compra
Produto	Termo geral para descrever aquilo que se compra
Funcionário	Termo geral que descreve as pessoas que realizam as vendas

Tabela 1 - Entidades

3.3. Identificação e Caracterização dos Relacionamentos

Uma vez caracterizadas as entidades procedeu-se novamente à análise dos requisitos registados com o intuito de estabelecer relações entre as entidades apresentadas. Desta forma, conseguimos apontar os seguintes relacionamentos :

Cliente - Fatura

Este relacionamento representa a compra de produtos por parte de um cliente. O cliente pode optar por comprar um ou mais produtos ou por não comprar nenhum. A fatura contém a informação do cliente, estando então associada apenas a um cliente, e contém também o valor total a ser pago pelo cliente. Estamos perante uma relação de 1 para N visto que um cliente obtém uma fatura cada vez que adquire um produto na farmácia (uma no mínimo, pois este é registado na sua primeira compra).

Fatura - Produto

Neste relacionamento temos uma cardinalidade de N para N já que existe a possibilidade de passar várias faturas para um dado produto, bem como podem ser vendidos vários produtos numa única fatura. Uma fatura também contém o preço total da compra de um ou mais produtos e guarda dados de forma a permitir uma atualização do stock dos mesmos.

Funcionário - Fatura

Um funcionário é responsável pela emissão de faturas pertencentes a clientes, o que nos permite assumir que a fatura é uma forma de relacionar um funcionário e os clientes. A fatura permite-nos analisar informação relativa ao funcionário tal como o seu número de vendas. Em termos de cardinalidade, um funcionário pode emitir várias faturas e uma fatura é lançada por apenas um funcionário.

Entidade	Cardinalidade	Relacionamento	Cardinalidade	Entidade
Cliente	(1,1)	Possui	(1,n)	Fatura
Fatura	(0,n)	Relaciona-se	(1,n)	Produto Farmacêutico
Funcionário	(1,1)	Emite	(0,n)	Fatura

Tabela 2 - Relacionamentos entre Entidades

3.4. Identificação e Caracterização da Associação dos Atributos com as Entidades e Relacionamentos

De forma a representar efetivamente os objetos, relativos a cada entidade, que se pretende guardar no sistema foi necessário fornecer a cada um os devidos atributos. São facilmente identificados uma vez que a informação foi guardada em detalhe nos requisitos do sistema.

Apresenta-se de seguida uma tabela que representa os atributos relativos a cada entidade:

Entidade	Atributo	Descrição	Multivalorado	Derivado
Cliente	idCliente	Identificador único de um cliente	Não	Não
	Nome	Nome do cliente	Não	Não
	NIF	NIF do cliente	Não	Não
	Telemovel	Número de telemóvel do cliente	Não	Não
	eMail	Email do cliente	Não	Não
	Pass	Palavra-chave do cliente	Não	Não
	Cidade	Cidade do cliente	Não	Não
	CodigoPostal	Código Postal do cliente	Não	Não
	Rua	Rua onde mora o cliente	Não	Não
Fatura	idFatura	Identificador único de uma fatura	Não	Não
	Funcionario	Funcionário associado a uma fatura	Não	Não
	Cliente	Cliente associado a uma fatura	Não	Não
	DataFatura	Data de emissão da fatura	Não	Não
	Desconto	Desconto aplicado à compra	Não	Não
	IVA	IVA aplicado à compra	Não	Não
	PrecoTotal	Preço total da compra	Não	Não
Funcionário	idFuncionario	Identificador único de um funcionário	Não	Não
	eMail	Email do funcionário	Não	Não
	Telemovel	Telemóvel do funcionário	Não	Não
	NISS	Número de Identificação de Segurança Social de um funcionário	Não	Não

	IBAN	IBAN do funcionário	Não	Não
	Pass	Palavra-chave do funcionário	Não	Não
	Nome	Nome do funcionário	Não	Não
	Salario	Salário do funcionário	Não	Não
	Cidade	Cidade do funcionário	Não	Não
	CodigoPostal	Código Postal do funcionário	Não	Não
	Rua	Rua onde mora o funcionário	Não	Não
Produto	idProduto	Identificador único de um produto	Não	Não
	Designação	Nome do produto	Não	Não
	Laboratorio	Laboratório onde foi desenvolvido o produto	Não	Não
	Administracao	Modo de administração do produto	Não	Não
	ReceitaMedica	Indicador caso o produto necessite de receita médica	Não	Não
	Dosagem	Quantidade por dose do produto	Não	Não
	QuantidadeEmbalagem	Quantidade de doses do produto	Não	Não
	Preco	Preco do produto	Não	Não
	Validade	Validade do produto	Não	Não
	EmbalagensEmStock	Número de embalagens em stock do produto	Não	Não

Tabela 3 - Atributos

3.5. Detalhe ou Generalização de Entidades

As entidades identificadas representam objetos distintos com diferentes funções o que permite concluir que não há necessidade de recorrer ao detalhe de entidades já que as entidades de um mesmo tipo terão ocorrências com características semelhantes e representam objetos idênticos. Também não recorremos à generalização uma vez que todos os atributos associados a uma entidade não variam , ou seja , não existem entidades com um conjunto de atributos idênticos.

3.6. Apresentação e Explicação do diagrama ER

Com as entidades , relacionamentos e atributos definidos conseguimos chegar ao seguinte diagrama ER :

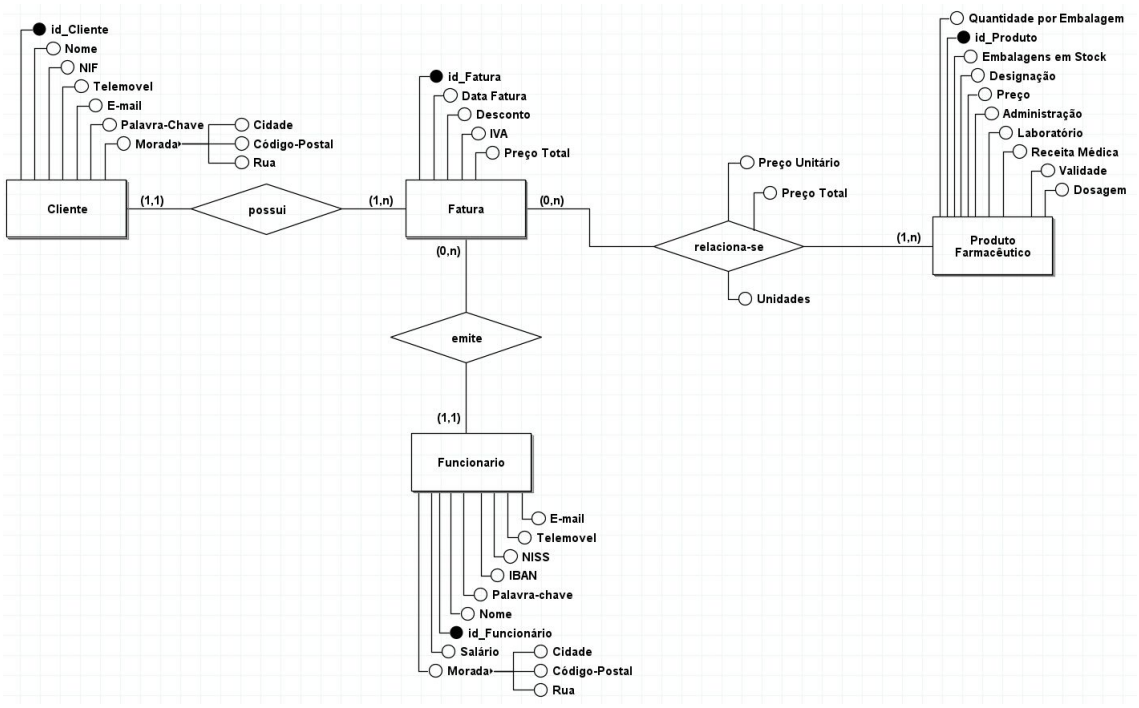

Figura 1 - Modelo Conceptual

No diagrama apresentado conseguimos verificar que as entidades e os seus respectivos atributos correspondem aos requisitos de exploração enunciados.O mesmo acontece para os relacionamentos.

As multiplicidades atribuídas já se encontram explicadas na seção 3.3.

A escolha das chaves primárias foi feita tendo em conta a mais eficiente para os casos como a entidade “Cliente” em que verifica-se a existência de 4 chaves candidatas uma vez que é possível identificar a entidade através do seu “NIF” , "telemóvel", “email” e “id_cliente” , contudo o cliente pode alterar o telemóvel e o email por isso estes atributos nao sao as melhores escolhas para chave primária. Entre as outras chaves candidatas escolhe-se o “id_cliente” dado que o NIF terá, certamente, um valor numérico superior.

Em casos com uma única forma de identificação como a entidade “Fatura” foi escolhida a opção existente para chave primária.

3.7. Validação do modelo de dados produzido

Dando por finalizado o modelo conceptual da base de dados, foi necessário validar o modelo de dados com o utilizador.

Para isto foi realizada uma reunião com a administração da Farmácia NTBH na qual procuramos conseguir explicar como satisfazer todos os requisitos necessários ao bom funcionamento da farmácia .

A nível do cliente, este deve ser capaz de consultar os registos das suas compras que devem conter a data das mesmas , quanto gastou em cada compra, qual o produto mais comprado e se obteve algum desconto. Para tal apresentamos a seguinte solução :

- Para possibilitar o acesso às datas de compras são necessárias as entidades Fatura e Funcionário, uma vez que a data de cada compra é referida na fatura e é o funcionário que a emite no seguimento do processo de compra.

- Com o acesso à entidade Fatura e com a relação desta com a entidade Produto, o cliente pode consultar o gasto em cada compra e se obteve um desconto já que na emissão da Fatura consta o preço total resultante da soma dos preços dos produtos, referidos pela entidade Produto, e se os mesmos estão , ou não, em desconto.
- Como cada Fatura contém o registo de cada compra , o cliente pode facilmente verificar qual o produto que adquire mais frequentemente e com quanta regularidade.

A nível do administrador, este deve ter acesso ao número de vendas de cada produto , à ficha de cada cliente , ao salário de cada funcionário , aos produtos em stock, às vendas num dado período de tempo, ao funcionário que efetuou mais vendas e aos lucros da farmácia. Por isso prosseguimos com as seguintes afirmações:

- Como cada Fatura contém os produtos vendidos a cada compra efetuada é possível listar o número de vezes que um dado produto é vendido.
- Cada cliente ao efetuar a sua primeira compra na farmácia é registado pelo Funcionário sendo-lhe associada uma ficha com todas as suas informações. Desta forma, é garantido o acesso à ficha individual de cada cliente.
- Para poder verificar o salário de cada Funcionário basta aceder à entidade Funcionario onde este lhe é atribuído.
- Cada fatura tem uma data de emissão por isso facilmente conseguimos quantificar o número de vendas realizadas num dado período de tempo.
- Como é o Funcionário a fazer a emissão da fatura esta fica registada nas suas vendas sendo possível verificar qual o Funcionário com maior número.
- Cada Fatura contém o valor total da compra e a entidade Produtos possui o valor individual de cada produto. Desta forma, é possível calcular o lucro adquirido pela farmácia.
- Como é atribuído o número de embalagens em stock de cada produto à entidade Produto podemos facilmente verificar o stock existente dos mesmos.

No fim da apresentação, a administração aprovou as justificações submetidas e validou o modelo de dados apresentado , uma vez que permite satisfazer todas as necessidades dos utilizadores.

4. Modelação Lógica

Após a conclusão da conceptualização do problema apresentado pelo modelo conceptual, anteriormente engendrado, necessitamos agora de construir um modelo lógico. A construção deste modelo segue o rumo de um modelo de dados relacional, tendo como base o modelo conceptual apresentado anteriormente. Neste capítulo serão apresentadas todas as etapas da construção e validação do modelo em questão, bem como o seu produto final.

4.1. Construção e validação do modelo de dados lógico

Para a construção deste modelo ser iniciada, necessitamos de estabelecer os elementos que vamos criar. Temos de perceber quais as relações a criar que representam as entidades, os relacionamentos e os atributos, todos eles presentes no modelo conceptual anteriormente construído. Esta construção pode ser feita faseadamente. A derivação das relações para o modelo lógico irá apenas percorrer passos que sejam necessários e que existam. Assim, teremos os passos apresentados de seguida:

4.1.1. Entidades Fortes

Uma entidade forte entende-se como uma entidade que possui uma chave primária e que não depende de uma outra entidade para que a mesma exista. Assim, pela análise do modelo conceptual, é percetível que todas as entidades são fortes, já que nenhuma depende de outra para a sua existência. Assim, no modelo lógico teremos as seguintes relações, representativas das entidades do modelo conceptual:

- **Cliente** (idCliente, Nome, NIF, Telemovel, eMail, Pass, Cidade, CodigoPostal, Rua)
    - **Chave Primária -** idCliente

- **Fatura** (idFatura, DataFatura, Desconto, IVA, PrecoTotal)
    - **Chave Primária -** idFatura

- **Funcionário** (idFuncionario, Nome, IBAN, NISS, Telemovel, Salario, Pass, eMail, Cidade, CodigoPostal, Rua)
    - **Chave Primária -** idFuncionario

- **Produto** (idProduto, Designacao, Laboratorio, ReceitaMedica, Dosagem, QuantidadeEmbalagem, Preco, Validade, EmbalagensEmStock)
    - **Chave Primária -** idProduto

É importante perceber que, no caso de termos atributos compostos no modelo conceptual, estes apenas se vão desdobrar e ser todos eles um atributo da relação no modelo lógico. Temos o exemplo de Morada, que passa no modelo lógico para três atributos distintos: Cidade, CodigoPostal e Rua.

4.1.2. Entidades Fracas

Ao contrário do conceito de entidade forte, entidade fraca é uma entidade que irá depender de uma outra entidade (entidade forte) para a sua existência ser possível. A sua chave primária é derivada pela entidade forte que é sua "dona". Desse modo, no caso em estudo, não nos deparamos com nenhuma entidade fraca.

4.1.3. Relacionamentos de um para muitos (1:n)

Um relacionamento de um para muitos, ou seja de (1:n) irá fazer com que a relação que apresenta multiplicidade N receba uma chave estrangeira, que será a chave primária da relação que tem multiplicidade 1. Possuímos dois relacionamentos de 1 para muitos (Cliente - Fatura e Funcionário - Fatura). Assim as seguintes relações iriam receber as seguintes chaves estrangeiras:

- **Fatura** (idFatura, DataFatura, Desconto, IVA, PrecoTotal)
    - **Chave Estrangeira** - Funcionario
        - **Origem em Funcionario** - idFuncionario (Primary key)
    - **Chave Estrangeira** - Cliente
        - **Origem em Cliente** - idCliente (Primary key)

4.1.4. Relacionamentos de muitos para muitos (n:m)

Um relacionamento de muitos para muitos irá gerar uma necessidade de uma nova relação intermédia entre ambas as relações em estudo. A chave primária desta relação, será um chave composta pelas duas chaves primárias das tabelas envolvidas na questão. Em conjunto com esta nova relação, podemos encontrar atributos na mesma que eram atributos de relacionamento no modelo conceptual. Este tipo de relacionamento acontece entre a Fatura e o Produto. Assim, as seguintes tabelas seriam criadas:

- **LinhaFatura** (PrecoUnitario, PrecoTotal, Quantidade)
    - **Chave Primária -** Fatura, Produto

4.2. Desenho do modelo lógico

Após estabelecidos os parâmetros a criar, surge então o desenho do esquema lógico:

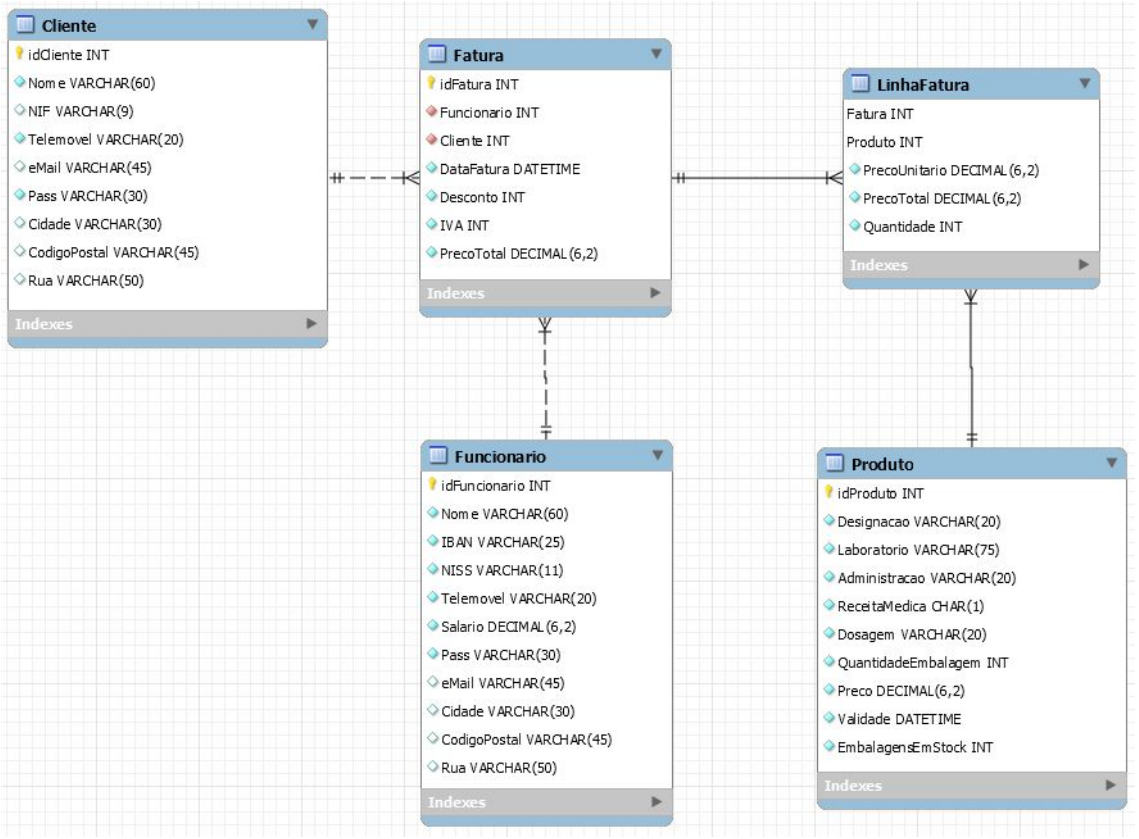

Figura 2 - Modelo Lógico

4.3. Validação do modelo com interrogações do Utilizador

Após construído e normalizado o modelo lógico, temos de verificar se o modelo produzido é capaz de responder aos requisitos do Utilizador. Para esta validação, utilizamos questões/interrogações feitas pelo Utilizador.

Deste modo, para os Clientes da Farmácia, o modelo consegue responder às interrogações da seguinte maneira:

- **Posso consultar os meus dados?**

  Apenas precisamos de selecionar na tabela dos Clientes, o Cliente em questão, e apresentar os seus dados.

- **Posso consultar as minhas faturas?**

  Para esta questão, teremos de fazer uma junção das tabelas dos clientes e das faturas. Assim, iremos selecionar todas as faturas de um dado cliente.

- **Posso consultar o stock da farmácia para saber se um determinado produto farmacêutico se encontra disponível?**

  Para consultarmos o stock de um certo produto, basta ir à tabela dos Produtos e procurar pelo produto examinado (pela linha). Depois de termos o produto, iremos aceder à coluna que corresponde à quantidade do mesmo em stock.

Para os Funcionários da Farmácia, o modelo consegue responder às interrogações da seguinte maneira:

- **Posso procurar por clientes?**

  Para procurar por clientes, apenas é necessário ir à tabela dos Clientes e procurar pelo cliente requerido.

- **Posso consultar os diversos produtos farmacêuticos?**

  Para consultar todos os produtos farmacêuticos, apenas é necessário aceder à tabela dos Produtos. No entanto, se quiser consultar um produto em específico, basta selecionar o mesmo da tabela dos Produtos.

Para o Administrador da Farmácia, o modelo consegue responder às interrogações da seguinte maneira:

- **Posso consultar os dados de pagamento do salário aos funcionários?**

  Para responder a esta questão, basta aceder à tabela dos funcionários, selecionar um funcionário e selecionar a coluna relativa ao seu salário.

- **Posso consultar quantas vendas foram feitas num certo período de tempo?**

  Para consultar o número de vendas feitas num certo período de tempo, basta selecionar da tabela das faturas, as faturas que se encontram no período de tempo estabelecido e fazer uma contagem das mesmas.

- **Posso consultar o total faturado pela farmácia num certo período de tempo?**

  Para consultar o total faturado pela farmácia num certo período de tempo, basta selecionar da tabela das faturas, as faturas que se encontram no período de tempo dado e somar todos os valores respectivos às colunas que representam o valor total de uma venda (fatura).

- **Posso consultar quem faturou mais num certo período de tempo?**

  Inicialmente é feita uma junção das tabelas dos funcionários e das faturas. Aqui faz-se uma seleção de apenas para apenas os que estão no período de tempo dado. A partir daí é calculado o valor total de produtos vendidos por cada funcionário. Após isso, os funcionários são ordenados decrescentemente segundo os valores referidos e é selecionada a primeira linha, que representa o funcionário que faturou mais no período de tempo dado.

- **Posso consultar o produto farmacêutico mais vendido?**

  Inicialmente é feita uma junção das tabelas dos produtos e das faturas. A partir das entradas de produtos associados a cada fatura, é calculado o total de quantidade vendida de cada produto. Após isso, os produtos são ordenados decrescentemente segundo esse valor total, e retiramos a primeira linha, que corresponde ao produto mais vendido.

- **Posso consultar o cliente que mais compras efetuou num certo período de tempo?**

  Inicialmente é feita uma junção das tabelas dos funcionários e das faturas. Aqui faz-se uma seleção de apenas para apenas os que estão no período de tempo dado. A partir daí é calculado o número total de faturas associado a cada cliente. Após isso, os clientes são ordenados decrescentemente segundo os valores obtidos e é selecionada a primeira linha, que representa o cliente que mais compras fez no período de tempo dado.

4.4. Revisão do Modelo Lógico produzido

Na fase final da modulação lógica da base de dados em estudo, é necessária a revisão desta mesma por parte do Utilizador. Esta revisão tem como grande objetivo garantir que toda a estrutura criada satisfaz os requisitos desejados para a mesma. Após esta revisão, damos o modelo lógico como aprovado, já que cumpre todos os requisitos pré-estabelecidos pelo Utilizador, como visto nos tópicos acima referidos.

5. Implementação Física

Chegando à fase final da construção do sistema de gestão de base de dados, é necessário fazer a implementação física da mesma, realizar o seu povoamento, fazer a sua exploração e ainda controlar a informação armazenada. Começamos por traduzir o esquema lógico para o sistema de gestão de base de dados escolhido. Após implementarmos o esquema físico, passamos a implementar as interrogações do Utilizador para SQL. Definimos ainda índices e vistas de utilização em SQL. Além disso, é ainda calculada uma estimativa do espaço em disco da base de dados ocupado.

No final de tudo, e analogamente ao que aconteceu anteriormente, é feita a revisão do sistema para concluir o processo.

5.1. Seleção do Sistema de Gestão de Base de Dados

Para a realização desta parte do projeto foi utilizado como Sistema de Gestão de Base de Dados o MySQL. Este foi o Sistema escolhido dado que era com o qual estávamos mais familiarizados. Tem também uma interface de fácil compreensão, com um bom desempenho, sendo ainda um serviço gratuito. Para além das caraterísticas referidas, torna-se muito relevante para a escolha este ser o sistema utilizado nas aulas.

5.2 Tradução do esquema lógico para o sistema de gestão de base de dados escolhido em SQL

Nesta fase do projeto, conseguimos facilmente, com recurso à ferramento de *forward engineer* do MySQL Workbench, converter o modelo lógico que possuíamos para o modelo físico. Após uma revisão e aprimoramentos de minuciosidades, o processo de tradução foi então concluído. Assim, as relações base produzidas na definição do modelo lógico são:

- **Relação Cliente**

Domínio ID Cliente (**idCliente**) : inteiro

Domínio Nome (**Nome**): string de comprimento variável, comprimento 60

Domínio Número Identificação Fiscal (**NIF**): string de comprimento variável, comprimento 9

Domínio Telemóvel (**Telemovel**): string de comprimento variável, comprimento 20

Domínio EMail (**eMail**): string de comprimento variável, comprimento 45

Domínio Password (**Pass**): string de comprimento variável, comprimento 30

Domínio Cidade (**Cidade**): string de comprimento variável, comprimento 30

Domínio Código Postal (**CodigPostal**): string de comprimento variável, comprimento 45

Domínio Rua (**Rua**): string de comprimento variável, comprimento 50

CHAVE PRIMÁRIA (**idCliente**)

CHAVE ALTERNATIVA (**NIF**)

CHAVE ALTERNATIVA (**eMail**)

```
-- -----------------------------------------------------
-- Table `ProjetoFarmacia`.`Cliente`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ProjetoFarmacia`.`Cliente` (
  `idCliente` INT UNSIGNED NOT NULL,
  `Nome` VARCHAR(60) NOT NULL,
  `NIF` VARCHAR(9) NULL,
  `Telemovel` VARCHAR(20) NOT NULL,
  `eMail` VARCHAR(45) NULL,
  `Pass` VARCHAR(30) NOT NULL,
  `Cidade` VARCHAR(30) NULL,
  `CodigoPostal` VARCHAR(45) NULL,
  `Rua` VARCHAR(50) NULL,
  PRIMARY KEY (`idCliente`),
  UNIQUE INDEX `idCliente_UNIQUE` (`idCliente` ASC) VISIBLE,
  UNIQUE INDEX `NIF_UNIQUE` (`NIF` ASC) VISIBLE,
  UNIQUE INDEX `eMail_UNIQUE` (`eMail` ASC) VISIBLE)
ENGINE = InnoDB;
```

Figura 3 - Código SQL para a criação da tabela cliente

- **Relação Funcionario**

Domínio ID Funcionário (**idFuncioario**) : inteiro

Domínio Nome (**Nome**): string de comprimento variável, comprimento 60

Domínio IBAN (**IBAN**): string de comprimento variável, comprimento 25

Domínio NISS (**NISS**): string de comprimento variável, comprimento 11

Domínio Telemóvel (**Telemovel**): string de comprimento variável, comprimento 20

Domínio Salário (**Salario**): decimal (precisão = 2 e escala = 6)

Domínio EMail (**eMail**): string de comprimento variável, comprimento 45

Domínio Password (**Pass**): string de comprimento variável, comprimento 30

Domínio Cidade (**Cidade**): string de comprimento variável, comprimento 30

Domínio Código Postal (**CodigPostal**): string de comprimento variável, comprimento 45

Domínio Rua (**Rua**): string de comprimento variável, comprimento 50

CHAVE PRIMÁRIA (**idFuncionario**)

CHAVE ALTERNATIVA (**IBAN**)

CHAVE ALTERNATIVA (**NISS**)

CHAVE ALTERNATIVA (**eMail**)

```
-- -----------------------------------------------------
-- Table `ProjetoFarmacia`.`Funcionario`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ProjetoFarmacia`.`Funcionario` (
  `idFuncionario` INT NOT NULL,
  `Nome` VARCHAR(60) NOT NULL,
  `IBAN` VARCHAR(25) NOT NULL,
  `NISS` VARCHAR(11) NOT NULL,
  `Telemovel` VARCHAR(20) NOT NULL,
  `Salario` DECIMAL(6,2) NOT NULL,
  `Pass` VARCHAR(30) NOT NULL,
  `eMail` VARCHAR(45) NULL,
  `Cidade` VARCHAR(30) NULL,
  `CodigoPostal` VARCHAR(45) NULL,
  `Rua` VARCHAR(50) NULL,
  PRIMARY KEY (`idFuncionario`),
  UNIQUE INDEX `idFuncionario_UNIQUE` (`idFuncionario` ASC) VISIBLE,
  UNIQUE INDEX `IBAN_UNIQUE` (`IBAN` ASC) VISIBLE,
  UNIQUE INDEX `NISS_UNIQUE` (`NISS` ASC) VISIBLE,
  UNIQUE INDEX `eMail_UNIQUE` (`eMail` ASC) VISIBLE)
ENGINE = InnoDB;
```

Figura 4 - Código SQL para a criação da tabela funcionário

- **Relação Fatura**

Domínio ID Fatura (**idFatura**) : inteiro

Domínio Funcionário (**Funcionario**): inteiro

Domínio Cliente (**Cliente**): inteiro

Domínio Data da Fatura (**DataFatura**): temporal (Datatime)

Domínio Desconto (**Desconto**): inteiro

Domínio IVA (**IVA**): inteiro

Domínio Preço Total (**PrecoTotal**): decimal (precisão = 2 e escala = 6)

CHAVE PRIMÁRIA (**idFatura**)

CHAVE ESTRANGEIRA (**Funcionario**)

CHAVE ESTRANGEIRA (**Cliente**)

```
-- -----------------------------------------------------
-- Table `ProjetoFarmacia`.`Fatura`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ProjetoFarmacia`.`Fatura` (
  `idFatura` INT NOT NULL,
  `Funcionario` INT NOT NULL,
  `Cliente` INT NOT NULL,
  `DataFatura` DATETIME NOT NULL,
  `Desconto` INT NOT NULL,
  `IVA` INT NOT NULL,
  `PrecoTotal` DECIMAL(6,2) NOT NULL,
  PRIMARY KEY (`idFatura`),
  INDEX `fk_Fatura_Funcionário_idx` (`Funcionario` ASC) VISIBLE,
  INDEX `fk_Fatura_Cliente1_idx` (`Cliente` ASC) VISIBLE,
  UNIQUE INDEX `idFatura_UNIQUE` (`idFatura` ASC) VISIBLE,
  CONSTRAINT `fk_Fatura_Funcionário`
    FOREIGN KEY (`Funcionario`)
    REFERENCES `ProjetoFarmacia`.`Funcionario` (`idFuncionario`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Fatura_Cliente1`
    FOREIGN KEY (`Cliente`)
    REFERENCES `ProjetoFarmacia`.`Cliente` (`idCliente`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Figura 5 - Código SQL para a criação da tabela fatura

- **Relação LinhaFatura**

Domínio Fatura (**Fatura**) : inteiro

Domínio Produto (**Produto**): inteiro

Domínio Preço Unitário (**PrecoUnitario**): decimal (precisão = 2 e escala = 6)

Domínio Preço Total (**PrecoTotal**): decimal (precisão = 2 e escala = 6)

Domínio Quantidade (**Quantidade**): inteiro

CHAVE PRIMÁRIA (**Fatura,Produto**)

CHAVE ESTRANGEIRA (**Fatura**)

CHAVE ESTRANGEIRA (**Produto**)

```
-- -----------------------------------------------------
-- Table `ProjetoFarmacia`.`LinhaFatura`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ProjetoFarmacia`.`LinhaFatura` (
  `Fatura` INT NOT NULL,
  `Produto` INT NOT NULL,
  `PrecoUnitario` DECIMAL(6,2) NOT NULL,
  `PrecoTotal` DECIMAL(6,2) NOT NULL,
  `Quantidade` INT NOT NULL,
  INDEX `fk_Linha-Fatura_Produto-Famacêutico1_idx` (`Produto` ASC) VISIBLE,
  PRIMARY KEY (`Fatura`, `Produto`),
  CONSTRAINT `fk_Linha-Fatura_Fatura1`
    FOREIGN KEY (`Fatura`)
    REFERENCES `ProjetoFarmacia`.`Fatura` (`idFatura`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_Linha-Fatura_Produto-Famacêutico1`
    FOREIGN KEY (`Produto`)
    REFERENCES `ProjetoFarmacia`.`Produto` (`idProduto`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

Figura 6 - Código SQL para a criação da tabela LinhaFatura

- **Relação Produto**

Domínio ID Produto (**idProduto**) : inteiro

Domínio Designação (**Designacao**): string de comprimento variável, comprimento 20

Domínio Laboratório (**Laboratorio**): string de comprimento variável, comprimento 75

Domínio Administração (**Administracao**): string de comprimento variável, comprimento 20

Domínio Receita Médica (**ReceitaMedica**): char

Domínio Dosagem (**Dosagem**): string de comprimento variável, comprimento 20

Domínio Quantdade Embalagens (**QuantidadeEmbalagens**): inteiro

Domínio Preço Unitário (**Preco**): decimal (precisão = 2 e escala = 6)

Domínio Prazo de Validade (**Validade**): temporal (Datetime)

Domínio Embalagens em Stock (**EmbalagensEmStock**): inteiro

CHAVE PRIMÁRIA (**idProduto**)

```
-- -----------------------------------------------------
-- Table `ProjetoFarmacia`.`Produto`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ProjetoFarmacia`.`Produto` (
  `idProduto` INT NOT NULL,
  `Designacao` VARCHAR(20) NOT NULL,
  `Laboratorio` VARCHAR(75) NOT NULL,
  `Administracao` VARCHAR(20) NOT NULL,
  `ReceitaMedica` CHAR(1) NOT NULL,
  `Dosagem` VARCHAR(20) NOT NULL,
  `QuantidadeEmbalagem` INT NOT NULL,
  `Preco` DECIMAL(6,2) NOT NULL,
  `Validade` DATETIME NOT NULL,
  `EmbalagensEmStock` INT NOT NULL,
  PRIMARY KEY (`idProduto`),
  UNIQUE INDEX `idProduto_UNIQUE` (`idProduto` ASC) VISIBLE)
ENGINE = InnoDB;
```

Figura 7 - Código SQL para a criação da tabela Produto

5.3 Tradução das Interrogações do Utilizador para SQL

Nesta secção serão apresentados alguns exemplos de código SQL que permitem obter respostas às interrogações do utilizador, expressas pelos requisitos de exploração.

- Consultar dados de um cliente

```sql
-- Consultar os dados de um cliente
DROP PROCEDURE IF EXISTS dados_cliente;

DELIMITER //
CREATE PROCEDURE dados_cliente (IN nome VARCHAR(60))
BEGIN
        SELECT * FROM cliente WHERE cliente.Nome = nome;
END //
DELIMITER ;
```

Figura 8 - Código SQL para consultar dados de um cliente

- Consultar as faturas de um cliente

```sql
-- Consultar as faturas de um cliente
DROP PROCEDURE IF EXISTS faturas_cliente;

DELIMITER //
CREATE PROCEDURE faturas_cliente (IN nome VARCHAR(60))
BEGIN
        SET @id = (SELECT idCliente FROM cliente WHERE cliente.Nome = nome);
        SELECT * FROM fatura WHERE fatura.Cliente = @id;
END //
DELIMITER ;
```

Figura 9 - Código SQL para consultar as faturas de um cliente

- Verificar disponibilidade de um produto

```
-- Verificar disponibilidade de um produto
DROP PROCEDURE IF EXISTS verificar_disponibilidade_produto;

DELIMITER //
CREATE PROCEDURE verificar_disponibilidade_produto (IN nome_produto VARCHAR(20))
BEGIN
        SELECT * FROM produto WHERE designacao = nome_produto;
END //
DELIMITER ;
```

Figura 10 - Código SQL para verificar a disponibilidade de um produto

- Procurar um cliente

```
-- Procurar um cliente
DROP PROCEDURE IF EXISTS procurar_cliente;

DELIMITER //
CREATE PROCEDURE procurar_cliente (IN nome VARCHAR(60))
BEGIN
        SELECT * FROM cliente WHERE cliente.Nome = nome;
END //
DELIMITER ;
```

Figura 11 - Código SQL para procurar um cliente

- Consultar os diversos produtos farmacêuticos

```
-- Consultar os diversos produtos
DROP PROCEDURE IF EXISTS consultar_produtos;

DELIMITER //
CREATE PROCEDURE consultar_produtos ()
BEGIN
        SELECT * FROM produto;
END //
DELIMITER ;
```

Figura 12 - Código SQL para consultar os diversos produtos farmacêuticos

- Consultar o vencimento de um funcionário

```
-- Consultar o vencimento de um funcionário
DROP PROCEDURE IF EXISTS consultar_vencimento_funcionario;

DELIMITER //
CREATE PROCEDURE consultar_vencimento_funcionario (IN nome VARCHAR(60))
BEGIN
       SELECT idFuncionario, Nome, Salario FROM funcionario f WHERE f.Nome = nome;
END //
DELIMITER ;
```

Figura 13 - Código SQL para consultar o vencimento de um funcionário

- Consultar quantas vendas foram feitas num certo período de tempo

```
-- Consultar quantas vendas foram feitas num certo período de tempo
DROP FUNCTION IF EXISTS numero_vendas_entre;

DELIMITER //
CREATE FUNCTION numero_vendas_entre (dataInicial DATETIME, dataFinal DATETIME) RETURNS INT
BEGIN
       RETURN (SELECT COUNT(*) AS TotalVendas FROM fatura f WHERE f.DataFatura >= dataInicial AND f.DataFatura <= dataFinal);
END //
DELIMITER ;
```

Figura 14 - Código SQL para consultar quantas vendas foram feitas num certo período de tempo

- Consultar o total faturado pela farmácia num certo período de tempo

```sql
-- Consultar o total faturado pela farmácia num certo período de tempo
DROP FUNCTION IF EXISTS total_faturado_entre;

DELIMITER //
CREATE FUNCTION total_faturado_entre (dataInicial DATETIME, dataFinal DATETIME) RETURNS DECIMAL(6,2)
BEGIN
        RETURN (SELECT SUM(PrecoTotal) AS TotalFaturado FROM fatura f WHERE f.DataFatura >= dataInicial AND f.DataFatura <= dataFinal);
END //
DELIMITER ;
```

Figura 15 - Código SQL para consultar o total faturado pela farmácia num certo período de tempo

- Consultar que funcionário que faturou mais num certo período de tempo

```sql
-- Consultar quem faturou mais num certo período de tempo
DROP PROCEDURE IF EXISTS quem_faturou_mais_entre;

DELIMITER //
CREATE PROCEDURE quem_faturou_mais_entre (IN dataInicial DATETIME, IN dataFinal DATETIME)
BEGIN
        SELECT f.Funcionario AS funcionario, SUM(f.PrecoTotal) AS total FROM fatura f WHERE f.DataFatura >= dataInicial AND f.DataFatura <= dataFinal
    GROUP BY funcionario ORDER BY total DESC LIMIT 1;
END //
DELIMITER ;
```

Figura 16 - Código SQL para consultar o funcionário que faturou mais num certo período de tempo

- Consultar os ‘n’ produtos farmacêuticos mais vendidos

```sql
-- Consultar os 'n' produtos farmacêuticos mais vendidos
DROP PROCEDURE IF EXISTS produtos_mais_vendidos;

DELIMITER //
CREATE PROCEDURE produtos_mais_vendidos (IN n INT)
BEGIN
        SELECT lf.Produto AS IDProduto, p.Designacao AS Designacao, SUM(lf.Quantidade) AS totalVendido
    FROM linhaFatura lf, produto p
    WHERE lf.Produto = p.idProduto
    GROUP BY IDProduto ORDER BY totalVendido DESC LIMIT n;
END //
DELIMITER ;
```

Figura 17 - Código SQL para consultar os ‘n’ produtos farmacêuticos mais vendidos

- Consultar o cliente que mais compras efetuou num determinado período de tempo

```sql
-- Consultar o cliente que mais compras efetuou num certo período de tempo
DROP PROCEDURE IF EXISTS quem_comprou_mais_entre;

DELIMITER //
CREATE PROCEDURE quem_comprou_mais_entre (IN dataInicial DATETIME, IN dataFinal DATETIME)
BEGIN
        SELECT f.Cliente AS IDCliente, c.Nome AS nomeCliente, COUNT(f.Cliente) AS total
    FROM fatura f, cliente c
    WHERE f.Cliente = c.idCliente AND f.DataFatura >= dataInicial AND f.DataFatura <= dataFinal
    GROUP BY IDCliente ORDER BY total DESC LIMIT 1;
END //
DELIMITER ;
```

Figura 18 - Código SQL para consultar o cliente que mais compras efetuou num determinado período de tempo

5.4. Escolha, definição e caracterização de índices em SQL

No seguinte tópico iremos realizar a escolha, a definição e a caracterização de índices do modelo em SQL. Para isso é preciso perceber o que são os índices em SQL. Os índices são uma funcionalidade que permite a ordenação das tabelas e dos seus dados consoante um certo parâmetro da mesma, ou seja, um atributo, que não seja no entanto a sua chave primária. Se tivermos, por exemplo, a lista de alunos inscritos na Universidade do Minho, será bastante espontâneo encontrar um aluno pelo seu número de identificação na Universidade (assumindo que esta é a chave primária da tabela que constitui). No entanto, será difícil e muito menos eficiente encontrar um aluno pelo seu nome, já que este não é o atributo pela qual a tabela se encontra ordenada.

Assim, conclui-se que a utilização destes índices será fulcral no que toca à otimização de queries de consulta de tabelas da base de dados. Posteriormente, estes índices criam estruturas tais como Árvores Binárias ou Árvores AVL que irão organizar as tabelas segundo o atributo contido no índice.

Analisando o nosso trabalho, todas as diferentes tabelas e os seus atributos, bem como as diferentes queries que queremos implementar, decidimos quais atributos colocar em índice, já que serão os que mais serão utilizados.

Assim, achamos por bem utilizar esta funcionalidade tanto nos Clientes como nos Funcionários, indexando as tabelas pelo atributo correspondente ao Nome em ambas, já que é o atributo que mais utilizamos (sem que seja a chave primária). Além destes, colocamos ainda em índice a Designação do Produto. Por último atribuímos ainda um índice à coluna que guarda a Data na tabela da Fatura, já que este atributo é muitas vezes utilizado nas diversas queries.

```
CREATE INDEX idx_nomeCli ON Cliente(Nome);
CREATE INDEX idx_nomeFunc ON Funcionario(Nome);

CREATE INDEX idx_desig ON Produto(Designacao);

CREATE INDEX idx_data ON Fatura(dataFatura);
```

Figura 19 - Código SQL para a criação dos índices sobre as colunas Nome (no Funcionário e no Cliente), Designação (no Produto) e dataFatura (na Fatura)

5.5. Estimativa do espaço em disco da base de dados e taxa de crescimento

Seguidamente é apresentada uma estimativa daquele que será o espaço utilizado pela base de dados no disco. Dado que podem existir vários casos possíveis, colocamos em estudo todos os piores casos. Com estes casos referimos-nos aos casos que possuem informação em todas as colunas das tabelas, mesmo naquelas que podem assumir o valor de NULL. Além disso, assumimos também que qualquer atributo que utilize o VARCHAR irá usufruir do máximo tamanho possível de ser aproveitado (já que neste tipo de atributos os bytes diferem consoante o número de chars utilizados).

Para melhor perceção dos casos em estudo foi construída uma tabela que respeita os parâmetros anunciados anteriormente. Assim, depois de calculados os valores, obtemos a seguinte tabela:

Relação	Atributo	Data Type	Tamanho MAX	Total MAX
Cliente	idCliente	INT	4 bytes	300 bytes
	Nome	VARCHAR(60)	61 bytes	
	NIF	VARCHAR(9)	10 bytes	

	Telemovel	VARCHAR(20)	21 bytes	
	eMail	VARCHAR(45)	46 bytes	
	Pass	VARCHAR(30)	31 bytes	
	Cidade	VARCHAR(30)	31 bytes	
	CodigoPostal	VARCHAR(45)	46 bytes	
	Rua	VARCHAR(50)	51 bytes	
Fatura	idFatura	INT	4 bytes	31 bytes
	Funcionario	INT	4 bytes	
	Cliente	INT	4 bytes	
	DataFatura	DATETIME	8 bytes	
	Desconto	INT	4 bytes	
	IVA	INT	4 bytes	
	PrecoTotal	DECIMAL(6,2)	3 bytes	
LinhaFatura	Fatura	INT	4 bytes	18 bytes
	Produto	INT	4 bytes	
	PrecoUnitario	DECIMAL(6,2)	3 bytes	
	PrecoTotal	DECIMAL(6,2)	3 bytes	
	Quantidade	INT	4 bytes	
Produto	idProduto	INT	4 bytes	163 bytes
	Designacao	VARCHAR(20)	21 bytes	
	Laboratorio	VARCHAR(75)	76 bytes	
	Administracao	VARCHAR(20)	21 bytes	
	ReceitaMedica	CHAR(1)	1 byte	
	Dosaagem	VARCHAR(20)	21 bytes	
	QuantidadeEmbalagem	INT	4 bytes	
	Preco	DECIMAL(6,2)	3 bytes	
	Validade	DATETIME	8 bytes	
	EmbalagensEmStock	INT	4 bytes	
Funcionario	idFuncionario	INT	4 bytes	331 bytes
	Nome	VARCHAR(60)	61 bytes	
	Salario	DECIMAL(6,2)	3 bytes	
	Telemovel	VARCHAR(20)	21 bytes	
	eMail	VARCHAR(45)	46 bytes	
	Pass	VARCHAR(30)	31 bytes	
	Cidade	VARCHAR(30)	31 bytes	
	CodigoPostal	VARCHAR(45)	46 bytes	
	Rua	VARCHAR(50)	51 bytes	
	IBAN	VARCHAR(25)	26 bytes	
	NISS	VARCHAR(11)	12 bytes	

Tabela 4 - Tamanho de cada entrada na respetiva tabela

Para uma previsão do espaço de disco utilizado no momento inicial, iremos estudar os casos de povoamento da base de Dados que foram construídos para o

projeto. Neste caso inicial iremos ter 11 Clientes, 4 Funcionários, 29 Produtos, 7 Faturas e 9 linhas de Fatura.

Feitas as contas, chegamos à conclusão que a base de dados ocupa inicialmente 9730 bytes (que ficará próximo dos 9.73KBytes).

No que toca ao crescimento da base de dados, é importante compreender que nem todos os componentes da base de dados têm uma evolução proporcional. Exemplificando, o número de funcionários irá manter-se o tanto que possível constante, dado que não serão contratados novos funcionários com grande frequência. Por sua vez, o número de clientes é um parâmetro que tende sempre a aumentar, mesmo que não exageradamente (de uma forma linear). O facto de o número de clientes crescer irá fazer com que o número de Faturas também cresça, bem como o número de Linhas de Fatura (que aparecem uma ou mais vezes em cada fatura). Não só o crescimento de clientes tem influência, mas também a compra de produtos na farmácia por parte de clientes já registrados na base de dados. Assim, tanto as Faturas como as Linhas de Fatura irão ter um comportamento mais parecido com um comportamento exponencial. Desse modo, toda a Base de Dados irá crescer consideravelmente ao longo do tempo. Isto deve-se ao facto de tanto os Clientes como as Faturas e as linhas de Fatura serem os componentes que irão crescer em maior quantidade no sistema e que possuem inclusive um número razoável de bytes. Se no entanto não forem inseridos clientes novos e a farmácia seja regida apenas por clientes já existentes que voltem para fazer compras, o crescimento da base de dados irá ser muito mais baixo, devido ao facto da criação de novos clientes possuir um custo tão alto e esta deixar de existir.

Ao contrário do que pode acontecer em outros projetos e empresas, como por exemplo um hotel em que são alugados todos os quartos em todas as noites sempre por clientes novos, que podemos claramente ver qual será o pior caso, o mesmo não acontecerá no âmbito de uma Farmácia, já que é praticamente impossível calcular uma estimativa do mesmo. No entanto, pode ser feita uma pequena perceção daquilo que iria acontecer, mesmo que sem qualquer cálculos, como referido anteriormente.

5.6. Definição e criação das vistas de utilização em SQL

As vistas de utilização são bastantes úteis já que passam por ser tabelas virtuais que aumentam a proteção e segurança dos dados, já que a informação apenas é mostrada a utilizadores autorizados.

Analisando o trabalho, achamos por bem apresentar duas das possíveis vistas de utilização. Uma delas, para cada fatura, relaciona o nome do Cliente e do Funcionário referentes à mesma. Apresenta ainda a data de emissão da fatura, o IVA imposto à compra e o preço total:

```
-- VIEW das faturas e clientes associados
DROP VIEW IF EXISTS view_faturas;

DELIMITER //
CREATE VIEW view_faturas
AS
    SELECT c.Nome AS Nome_Cliente, f.Nome AS Nome_Funcionario, fat.DataFatura AS Data_Compra, fat.IVA AS IVA, fat.PrecoTotal AS Dinheiro_Gasto
    FROM fatura fat
    INNER JOIN cliente AS c ON fat.Cliente = c.idCliente
    INNER JOIN funcionario AS f ON fat.Funcionario = f.idFuncionario;
//
DELIMITER ;

SELECT * FROM view_faturas;
```

Figura 20 - Código SQL para a implementação da vista das faturas e clientes associados

	Nome_Cliente	Nome_Funcionario	Data_Compra	IVA	Dinheiro_Gasto
▶	Ana Teresa Gião Gomes	Jaime Oliveira	2020-12-04 10:10:32	6	12.34
	Maria Beatriz Araújo Lacerda	Jaime Oliveira	2020-12-04 10:15:17	6	13.98
	Maria Quintas Barros	Jaime Oliveira	2020-12-04 11:47:52	6	7.75
	Francisco Correia Franco	Jaime Oliveira	2020-12-04 11:51:05	6	17.98
	Maria Beatriz Araújo Lacerda	Tibúrcio Mantorras	2020-12-04 15:45:27	6	7.35
	Maria da Trindade Pascoal	Tibúrcio Mantorras	2020-12-04 15:51:11	6	22.98
	João da Costa e Campos	Tibúrcio Mantorras	2020-12-04 17:35:20	6	7.95

Figura 21 - Vista das faturas e clientes associados

A outra vista discrimina, para cada fatura, os produtos associados a esta. Associado a este produto encontramos ainda informação sobre a sua forma de administração, o preço por embalagem e ainda o número de embalagens compradas:

```sql
-- VIEW das linhas duma fatura e dos produtos associados
DROP VIEW IF EXISTS view_linhas_fatura;

DELIMITER //
CREATE VIEW view_linhas_fatura
AS
    SELECT lf.Fatura AS Fatura, p.Designacao AS Designacao_Produto, p.Administracao AS Administracao, lf.PrecoUnitario AS Preco_Por_Embalagem, lf.Quantidade AS Embalagens
    FROM linhaFatura lf
    INNER JOIN produto AS p ON lf.Produto = p.idProduto
    ORDER BY Fatura ASC;
//
DELIMITER ;

SELECT * FROM view_linhas_fatura;
```

Figura 22 - Código SQL para a implementação da vista das linhas de cada fatura e dos produtos associados

Fatura	Designacao_Produto	Administracao	Preco_Por_Embalagem	Embalagens
1	Bepanthene	Uso cutâneo	4.99	1
1	Brufen	Via oral	7.35	1
2	Rosilan	Via oral	6.99	2
3	Halibut	Uso cutâneo	7.75	1
4	Cetix	Via oral	8.99	2
5	Brufen	Via oral	7.35	1
6	Urispas	Via oral	7.99	1
6	Esomeprazol Alter	Via oral	13.99	1
7	Bisoltussin	Via bucal	7.95	1

Figura 23 - Vista das linhas de cada fatura e dos produtos associados

5.7. Definição e Implementação de “Triggers” em SQL

Na implementação de uma base de dados, por vezes, é necessário reagir a eventos que não são acionados pelo Utilizador. Desta forma surgem os triggers/gatilhos que são ativados automaticamente quando um dado evento acontece.

Neste trabalho em concreto decidimos usar um trigger após a inserção de linha na tabela LinhaFatura. Este trigger tem duas partes: primeiramente começa por atualizar a tabela fatura nomeadamente o atributo PrecoTotal; Posteriormente, este atualiza o stock existente do Produto na Base de Dados, mais especificamente atualiza o atributo EmbalagensEmStock na tabela do Produto em questão.

```
DROP TRIGGER IF EXISTS insere_na_fatura;

DELIMITER //
CREATE TRIGGER insere_na_fatura AFTER INSERT ON linhafatura FOR EACH ROW
BEGIN
        -- Update da fatura
    UPDATE fatura SET precoTotal = precoTotal + (NEW.PrecoTotal * (1-(Desconto/100))) WHERE idFatura = NEW.Fatura;

    -- Update do stock
    UPDATE produto SET EmbalagensEmStock = EmbalagensEmStock - NEW.Quantidade WHERE idProduto = NEW.Produto;
END //
DELIMITER ;
```

Figura 24 - Trigger ativado na inserção de uma linha na linhaFatura

5.8. Revisão do sistema implementado

Dada a conclusão da última etapa de construção do Sistema de Gestão de Base de Dados, procede-se então à revisão da mesma e de toda a sua implementação por parte do Utilizador. Nesta revisão foram exploradas as várias funcionalidades propostas para o sistema, e inspecionadas a implementação dos requisitos especificados no modelo físico. Após essa retificação de todo o sistema, o mesmo foi aprovado e aceite com bastante satisfação.

6. Conclusões e Trabalho Futuro

Através de um devido faseamento ao longo do desenvolvimento da base de dados, o nosso grupo pensa que isso teve uma grande contribuição para a realização de um trabalho conciso e bem estruturado. Este tipo de metodologia fez com que a quantidade de erros ao longo do projeto diminuísse e que ao existir foco, concentração e todo o devido cuidado em cada uma das fases de implementação fosse possível a sua realização sem que quase não fosse preciso retroceder a uma fase anterior.

Fazendo uma análise geral de todo o trabalho realizado pensamos que a fase mais complicada foi o desenvolvimento do modelo concetual pois este torna-se a base do nosso trabalho e qualquer erro cometido nesta fase poderia tornar toda a nossa base de dados obsoleta. Para se perceber a importância deste modelo, é através deste que se deriva o modelo lógico, de onde mais tarde se desenvolve o modelo lógico. Assim, é possível perceber a importância que existe na conceção do modelo concetual de modo a todo o resto do trabalho. Deste modo, obtivemos uma melhor perceção do funcionamento da nossa empresa e de como todas as entidades existentes se relacionam entre si, sendo que houve uma maior qualidade dos requisitos levantados.

No futuro, poderíamos adicionar mais entidades ao nosso projeto de forma a enriquecer ainda mais a nossa base de dados e conseguir uma melhor aproximação daquilo que é a realidade do funcionamento de uma cadeia de farmácias de uma empresa.

Concluímos assim que todos os objetivos propostos na realização deste trabalho foram cumpridos, sendo que todo este processo culminou numa base de dados totalmente funcional.

ANEXOS

```sql
--
-- Esquema: "Farmácia"
USE `ProjetoFarmacia` ;

--
-- Permissão para fazer operações de remoção de dados.
SET SQL_SAFE_UPDATES = 0;


-- %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
-- Povoamento da tabela "Cliente"
INSERT INTO cliente
    (idCliente, Nome, NIF, Telemovel, eMail, Pass, Cidade, CodigoPostal, Rua)
    VALUES
        (1, 'João da Costa e Campos',        '123456789', '911111111', 'jcc@ntbh.pt',  'JonyCosta83',  'Aguada do Queixo',
            '9999-99-99 Queijadas',          'Rua das Adegas Felizes, 12, 1ª Cave'),
        (2, 'Josefina Vivida da Paz',        '122133144', '912222222', 'josefina@ntbh.pt', 'JosfPazz', 'Friso do Eixo',
            '7799-77-77 Friso do Eixo',      'Av dos Castros Reais, 122, 3º'),
        (3, 'Ana Santa do Carmo',            '876543298', '913333333', 'saca@ntbh.pt',  'SantaCarmo123',  'Abre o Tacho',
            '4534-54-21 Vale do Tacho',      'Travessa do Jacob, 21'),
        (4, 'Jesualdo Peza-Mor',             '564352786', '914444444', 'pezamor@ntbh.pt',  'JesusJ88',  'Vale dos Lençóis',
            '1245-64-11 Camadas',            'Estrada do Sossego, Km10'),
        (5, 'Maria da Trindade Pascoal',     '777666555', '915555555', 'trindade@ntbh.pt',  'Teletubbies2000',  'Aguada do Queixo',
            '9999-99-99 Queijadas',          'Rua das Adegas da Rua, 15, 10 Esq/T'),
        (6, 'Florindo Teixo Figueirinha',    '787676651', '916666666', 'teixof@ntbh.pt',  'Flores6969',  'Veloandro',
            '5555-59-55 Veloandro',          'Autódromo das Vagas, Garagem 123'),
        (7, 'Carminho Cunha Bastos',         '543234111', '917777777', 'cbastos@ntbh.pt',  'CunhaBoy1976',  'Vitalis do Sousa',
            '6532-A2-43 Vitalis',            'Rua do Mus-Vitalis, 56, r/c '),
        (8, 'Francisco Correia Franco',      '152152152', '969007408', 'sanic@gmail.com',  'Gottagofast123',  'Povoação',
            '1976-152 Povoação',             'Rua do Povo, 152'),
        (9, 'Maria Quintas Barros',          '044044044', '967399483', 'mosarte@gmail.com',  'Tamtamtam123',  'Povoação',
            '1976-044 Povoção',              'Rua do Povo, 44'),
        (10,'Maria Beatriz Araújo Lacerda',  '136136136', '961150609', 'eskilo@gmail.com',  '123Macacochines',  'Povoação',
            '1976-136 Povoação',             'Rua do Povo, 136'),
        (11,'Ana Teresa Gião Gomes',         '158158158', '916841002', 'babbles@gmail.com',  'MantorrasPrimo2000',  'Povoação',
            '1976-158 Povoação',             'Rua do Povo, 158')
    ;

-- %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
-- Povoamento da tabela "Produto"
INSERT INTO produto
    (idProduto, Designacao, Laboratorio, Administracao, ReceitaMedica, Dosagem, QuantidadeEmbalagem, Preco, Validade, EmbalagensEmStock)
    VALUES
        (1, 'Zemalex', 'Italfarmaco - Produtos Farmacêuticos, Lda.', 'Uso cutâneo', '1', '100 ml', 1, 5.99, '2021/07/31', 10),
        (2, 'Bepanthene', 'Bayer Portugal, S.A.', 'Uso cutâneo', '0', '100 g', 1, 4.99, '2021/05/01', 10),
        (3, 'Imodium', 'Johnson & Johnson, Lda.', 'Via oral', '1', '2 mg', 20, 14.99, '2021/05/10', 6),
        (4, 'Voltaren', 'Novartis Farma - Produtos Farmacêuticos, S.A.', 'Via oral', '1', '50 mg', 20, 12.99, '2021/08/01', 4),
        (5, 'Voltaren Emulgel', 'Novartis Consumer Health - Produtos Farmacêuticos e Nutrição Lda.', 'Uso cutâneo', '0', '100 g', 1, 11.50, '2021/04/01', 4),
        (6, 'Voltaren Rapid', 'Novartis Farma - Produtos Farmacêuticos, S.A.', 'Via oral', '1', '25 mg', 20, 14.99, '2021/08/01', 5),
        (7, 'Urispas', 'Jaba Recordati, S. A.', 'Via oral', '1', '200 mg', 60, 7.99, '2021/07/01', 3),
        (8, 'Brufen', 'Abbott Laboratórios, Lda.', 'Via oral', '1', '400 mg', 20, 7.35, '2021/09/01', 15),
        (9, 'Brufen', 'Abbott Laboratórios, Lda.', 'Via oral', '1', '600 mg', 20, 10.50, '2021/09/01', 15),
        (10, 'Esomeprazol Alter', 'Alter, S.A.', 'Via oral', '1', '20 mg', 56, 7.99, '2021/04/20', 7),
        (11, 'Esomeprazol Alter', 'Alter, S.A.', 'Via oral', '1', '40 mg', 56, 13.99, '2021/04/20', 4),
        (12, 'Fenistil', 'Novartis Consumer Health - Produtos Farmacêuticos e Nutrição Lda.', 'Via oral', '0', '1 mg', 20, 7.50, '2021/05/31', 5),
        (13, 'Fenistil', 'Novartis Consumer Health - Produtos Farmacêuticos e Nutrição Lda.', 'Via oral', '0', '4 mg', 20, 11.50, '2021/05/31', 5),
        (14, 'Fenistil Gel', 'Novartis Consumer Health - Produtos Farmacêuticos e Nutrição Lda.', 'Uso cutâneo', '0', '50 g', 1, 9.75, '2021/04/01', 7),
        (15, 'Furadantina MC', 'Goldshield Pharmaceuticals, Ltd.', 'Via oral', '1', '50 mg', 20, 8.99, '2021/09/30', 3),
        (16, 'Furadantina MC', 'Mercury Pharmaceuticals Ltd', 'Via oral', '1', '100 mg', 50, 13.99, '2021/08/31', 4),
        (17, 'Unisedil', 'Laquifa - Laboratórios, S.A.', 'Via oral', '1', '5 mg', 40, 5.99, '2021/08/01', 6),
        (18, 'Rosilan', 'Laboratórios Vitória, S.A.', 'Via oral', '1', '6 mg', 20, 6.99, '2021/10/01', 6),
        (19, 'Rosilan', 'Laboratórios Vitória, S.A.', 'Via oral', '1', '30 mg', 10, 9.99, '2021/10/01', 3),
        (20, 'Rinialer', 'Bialfar - Produtos Farmacêuticos, S.A.', 'Via oral', '1', '10 mg', 15, 7.35, '2021/10/31', 5),
        (21, 'Bilaxten', 'Laboratórios Vitória, S.A.', 'Via oral', '1', '20 mg', 10, 8.50, '2021/08/01', 4),
        (22, 'Cetix', 'Medinfar Consumer Health - Produtos Farmacêuticos, Lda.', 'Via oral', '0', '10 mg', 10, 8.99, '2021/07/01', 6),
        (23, 'Bisoltussin', 'Boehringer Ingelheim, Lda.', 'Via bucal', '0', '10.5 mg', 20, 7.95, '2021/06/01', 5),
        (24, 'Arnigel', 'Laboratoires Boiron, S.A.', 'Uso cutâneo', '0', '45 g', 1, 7.50, '2021/05/01', 3),
        (25, 'Halibut', 'Grünenthal, S.A.', 'Uso cutâneo', '0', '100 g', 1, 7.75, '2021/06/01', 7),
        (26, 'Halibut', 'Grünenthal, S.A.', 'Uso cutâneo', '0', '50 g', 1, 4.35, '2021/06/01', 5),
        (27, 'Ben-U-Ron', 'Bene Farmacêutica, Lda.', 'Via oral', '0', '500 mg', 20, 5.35, '2021/08/01', 10),
        (28, 'Ben-U-Ron', 'Bene Farmacêutica, Lda.', 'Via oral', '0', '1000 mg', 20, 8.95, '2021/08/01', 15),
        (29, 'Ben-U-Ron', 'Bene Farmacêutica, Lda.', 'Via oral', '0', '1000 mg', 36, 14.95, '2021/08/01', 8)
    ;
```

```sql
-- %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
-- Povoamento da tabela "Fatura"
INSERT INTO fatura
    (idFatura, Funcionario, Cliente, DataFatura, Desconto, IVA, PrecoTotal)
    VALUES
        (1, 4, 11, '2020/12/04 10:10:32', 5, 6, 12.34),
        (2, 4, 10, '2020/12/04 10:15:17', 0, 6, 13.98),
        (3, 4, 9,  '2020/12/04 11:47:52', 0, 6, 7.75),
        (4, 4, 8,  '2020/12/04 11:51:05', 0, 6, 17.98),
        (5, 3, 10, '2020/12/04 15:45:27', 0, 6, 7.35),
        (6, 3, 5,  '2020/12/04 15:51:11', 0, 6, 22.98),
        (7, 3, 1,  '2020/12/04 17:35:20', 0, 6, 7.95)
    ;
```

```sql
-- %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
-- Povoamento da tabela "LinhaFatura"
INSERT INTO linhafatura
    (Fatura, Produto, PrecoUnitario, PrecoTotal, Quantidade)
    VALUES
        (1, 2, 4.99, 4.99, 1),
        (1, 8, 7.35, 7.35, 1),
        (2, 18, 6.99, 13.98, 2),
        (3, 25, 7.75, 7.75, 1),
        (4, 22, 8.99, 17.98, 2),
        (5, 8, 7.35, 7.35, 1),
        (6, 11, 13.99, 13.99, 1),
        (6, 7, 7.99, 7.99, 1),
        (7, 23, 7.95, 7.95, 1)
    ;
```

```sql
-- %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
-- Povoamento da tabela "Funcionario"
INSERT INTO funcionario
    (idFuncionario, Nome, IBAN, NISS, Telemovel, Salario, Pass, eMail, Cidade, CodigoPostal, Rua)
    VALUES
        (1, 'Álvaro de Campos',      'PT50447190141202842236851', '82953993927', '921111111', 1000, 'AlvarIX',
            'alvaro@pessoa.pt',      'Encosta do Sabonete', '1111-999 Sabonete', 'Rua Limpa, 123, 4º Esq'),
        (2, 'Joaquim Arnaldo',       'PT50066364552108015180912', '44995242528', '921111112', 1200, 'Escopetas123',
            'arnalds@gmail.com',     'Porto', '2222-999 Boavista', 'Travessa da Boavista, 69'),
        (3, 'Tibúrcio Mantorras',    'PT50836833437107073483503', '54556336057', '921111113', 1200, 'Tibititi90',
            'tiburcio@gmail.com',    'Porto', '3333-999 Boavista', 'Rua do Oculista, 456'),
        (4, 'Jaime Oliveira',        'PT50513575430934735563824', '11533672286', '963812190', 2000, '5SherkBoy',
            'jaiminho@gmail.com',    'Braga', '4710-405 Enguardas', 'Rua Sr. da Paz, 5')
    ;
```