Universidade do Minho

Escola de Engenharia

Mestrado Integrado em Engenharia Informática

Unidade Curricular de

Bases de Dados

Ano Lectivo de 2015/2016

BelaPadaria

Célia Natália Lemos Figueiredo (a67637),

Márcia Maria Fernandes da Costa (a67672),

Pedro Miguel da Rocha Ribeiro (a67717),

Xavier Neves Francisco (a67725)

Janeiro, 2016

Data de Recepção	
Responsável	
Avaliação	
Observações	

BelaPadaria

Célia Natália Lemos Figueiredo (a67637),
Márcia Maria Fernandes da Costa (a67672),
Pedro Miguel da Rocha Ribeiro (a67717),
Xavier Neves Francisco (a67725)

<<Janeiro, 2016>>

<</opcional Dedicatória>>

Resumo

O documento contém toda a informação sobre o processo de elaboração de uma arquitetura de base de dados, que será a base para a informatização do processo de fabrico e venda de uma empresa produtora de pão.

Inicialmente é feita a contextualização do problema apresentado, seguindo-se da análise do caso de estudo em questão. Ainda numa parte introdutória é descrita a motivação que levou a empresa a optar pela mudança do modo como era registada a informação relativa ao processo de produção, assim como os objetivos a atingir com essa mesma mudança.

Esta primeira parte do relatório do projeto incluirá os seguintes pontos: contextualização do projeto, motivação e objetivos, análise de requisitos e a estrutura do relatório.

Durante a fase de modelação conceptual da base de dados serão identificados, com base nos requisitos, as entidades envolvidas no sistema e os seus relacionamentos. De seguida, serão identificados os atributos associados às entidades e/ou aos relacionamentos. Em cada entidade serão apresentadas as suas chaves candidatas e a escolha da sua chave primária. Após concluído o modelo conceptual e antes de prosseguir com o modelo lógico, será feita a eleição do motor de base de dados, onde futuramente será implementada toda a estrutura que derivará dos vários desenhos que serão calculados durante todo o processo.

Com base na segunda fase da metodologia, será derivado o modelo lógico correspondente ao modelo conceptual. De seguida serão validadas as relações através da normalização, a qual corresponde a um dos passos mais importantes da metodologia. Após verificar que o modelo está normalizado serão validadas as relações através do uso de transações do utilizador, assim como verificadas as restrições de integridade.

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

Palavras-Chave: Bases de dados relacionais, análise de requisitos, entidades, atributos, relacionamentos, metodologia, modelo conceptual, modelo lógico, modelo físico, SGBD

Índice

Índice de Figuras

Índice de Tabelas

1. Introdução

Este primeiro capitulo tem como objetivo apresentar uma breve introdução ao projeto a realizar. Sendo assim, é necessário definir o contexto no qual se desenvolve o caso de estudo, seguido da sua descrição. É também importante perceber os motivos que levaram à criação da *BelaPadaria* assim como os objetivos pretendidos.

Ainda numa parte introdutória ao nosso trabalho prático, vamos definir o contexto no qual se desenvolve o caso de estudo e posteriormente descrevê-lo.

Vamos também abordar alguns motivos que nos levaram à escolha e posterior desenvolvimento da *BelaPadaria,* incluindo desta forma, uma análise do caso de estudo.

Relativamente a segunda parte do nosso projeto será apresentado o modelo lógico correspondente ao modelo conceptual. Faremos uma análise e validação das relações através da normalização e ainda de seguida faremos a validação das relações construindo o mapa das transações e respetivas anotações e conclusões finais.

1.1. Contextualização

A panificação é talvez uma das artes culinárias mais antigas, os primeiros relatos sobre a produção e consumo de pão pelo homem datam do ano 8.000 a.C. quando uma massa de farinha de trigo (obtida pela moagem rudimentar com pedras) seria misturada com água e cozinhada em pedras aquecidas. Atualmente o processo de produção de pão está bastante diferente devido à grande evolução e à concorrência no mundo da indústria, existe uma grande oferta tecnológica e inovadora que permite às empresas oferecer um serviço com qualidade e diversificação aos clientes. O facto de as empresas encontrarem equipamentos adaptados à sua necessidade, cujo investimento é muito alto, quase todas as padarias têm os mesmos produtos, sendo artesanal e padronizado, o que fica muito claro que a maioria dos produtos é pré- mistura, com acabamento no ponto de venda, parecendo os mesmos em todas as padarias. É apostando nesses mercados sempre em evolução e procurando disponibilizar produtos a preço mais baixo que a concorrência, que surge a *BelaPadaria*.

A empresa *BelaPadaria* dedica-se ao fabrico de pão e outros derivados para venda no setor alimentar. A empresa efetua todo o processo de fabrico, desde a aquisição da matéria-

prima até à conclusão do processo de fabrico de pão e outros derivados para posterior venda, incluindo ainda, um controlo minucioso sobre a qualidade do mesmo.

Os produtos produzidos destinam-se à venda direta ao consumidor final.

1.2. Apresentação do Caso de Estudo

A padaria efetua todo o processo de fabrico desde a aquisição da matéria-prima até à conclusão do pão em si para posterior venda. Está ainda incluído um controlo minucioso sobre a qualidade do pão. A empresa está dividida em vários sectores nomeadamente o sector financeiro, recursos humanos, compra de matérias-primas, fabrico de pão e entrega. Apenas será alvo de estudo os sectores de fabrico de pão e a sua respetiva entrega.

Será também englobado no problema, a gestão dos operários que fazem parte deste processo de produção. Assim, operários, supervisores e chefes de secção da produção serão abrangidos, mas todos os trabalhadores de outras secções não farão parte do sistema a desenvolver.

O processo de construção do pão segue uma sequência que vamos passar a descrever: na primeira fase, um operário segue uma determinada receita para fazer a massa do pão; após estar concluída, a massa é dividida e modelada na forma final em que o pão é vendido. Seguidamente é colocado no forno onde é cozido à temperatura adequada para garantir a textura característica da padaria em estudo.

Os pães concluídos são enviados para o controlo de qualidade, porém este processo não será alvo de estudo, para posteriormente o pão ser vendido na loja por um funcionário responsável.

Neste documento, todo este processo será alvo de um estudo profundo, documentado através de uma pormenorizada análise de requisitos, para de acordo com a metodologia traçada, ser então desenvolvido um sistema que cumpra todos os objetivos definidos.

1.3. Motivação e Objetivos

Motivação

Os portugueses tal como a maioria das culturas pelo mundo têm o pão presente na sua alimentação e não passam uma refeição sem saborear uma boa fatia. Em Portugal, a tradição de comer pão é já antiga no tempo e é um dos alimentos que está na base da alimentação portuguesa. Feito à base de três cereais – milho, centeio e trigo – é diferente de região para região tanto na forma, como na cor, no gosto ou na textura do miolo, sendo usado em pratos tradicionais ou até em doces.

Foi o facto de ser um alimento simples e tão famoso que nos fez pensar em criar uma base de dados para a gestão mais eficiente de uma padaria. Visto que o pão é do consumo comum da comunidade portuguesa, achamos que uma empresa deste tipo daria lucro e teria margem para crescer.

Devido à grande adesão esperada por parte de potenciais clientes, esta área dá bastante lucro à empresa, e consequentemente, o sucesso pretendido.

Objetivos

Após a conclusão deste projeto e da ingressão da *BelaPadaria* no mercado, é esperado que se consiga atingir um determinado número de objetivos. Assim, um dos objetivos pretendidos é que a empresa consiga progredir no mercado, obtendo sempre um número cada vez mais elevado de clientes ao longo do tempo. Para o aumento da produção esperado, espera-se conseguir atingir um nível elevado no que diz respeito à rapidez e organização de todos os membros da empresa.

Surge também a necessidade de diversificação de ofertas que temos para a comunidade. Pretende-se que sejamos reconhecidos como os mais económicos e com mais qualidade. Pretende-se também que os serviços oferecidos sejam cada vez mais e melhores. Outro objetivo seria proporcionar ao cliente um bom atendimento, para além dos objetivos mencionados, pretende-se que a empresa tenha facilidade no acesso e gestão de informação sobre clientes, ofertas, faturas e empregados.

1.4. Estrutura do Relatório

Este relatório descreve detalhadamente, o processo de estudo, conceção e implementação da base de dados que servirá de suporte à BelaPadaria, seguindo a metodologia proposta no livro Database Systems - A Practical Approach to Design, Implementation and Management 4º Edição (Connolly e Begg, 2005).

Assim sendo o documento está organizado em sete capítulos, sendo que o primeiro está a ser abordado e trata do processo de conhecimento do caso de estudo, mais especificamente, a contextualização do caso de estudo, a apresentação do mesmo, a motivação para que o mesmo seja trabalhado e por fim os objetivos pretendidos com a materialização da solução do caso de estudo.

O segundo capítulo aborda o levantamento de requisitos que será a base para todo o resto do trabalho a desenvolver. Assim no terceiro capítulo teremos o desenvolvimento do modelo conceptual que acontece diretamente a partir do levantamento de requisitos. O modelo

conceptual será devidamente documentado assim como testado para os requisitos considerados.

O quarto capítulo é dedicado ao modelo lógico, sendo que na primeira secção se apresenta a derivação das relações do modelo conceptual para este novo modelo. Validaremos as relações através da normalização. Estas relações são na secção seguinte validadas através das transações. Neste capítulo é feita também a validação das restrições de integridade do sistema e uma revisão do modelo com o utilizador e uma análise do crescimento futuro da base de dados.

No quinto capítulo passa-se para o modelo físico onde na Secção 5.1 é feita a tradução do modelo anterior para o SGBD escolhido. Seguem-se três subsecções com o desenho das relações base do modelo, das representações dos dados derivados e das restrições gerais respetivamente. É também apresentada a organização de ficheiros e índices do modelo. Esta secção está dividida na análise de transações, escolha da organização dos ficheiros e de índices e termina com uma estimativa do espaço em disco necessário para o sistema desenvolvido. Faremos também o desenho das vistas de utilizadores. O capítulo é finalizado apresentado os mecanismos de segurança do sistema, uma consideração da introdução de redundância no modelo e uma monitorização e afinação do sistema operativo.

No sexto capítulo são apresentadas as ferramentas utilizadas ao longo do desenvolvimento deste projeto. Finalmente é realizada uma apreciação crítica sobre o trabalho final, destacando os pontos fortes e fracos do mesmo e o trabalho que futuramente poderá ser desenvolvido. A bibliografia pode ser consultada nas últimas páginas deste documento.

Por último, trataremos das conclusões finais do nosso projeto.

2. Análise e levantamento de requisitos

Esta fase do desenvolvimento é muito importante para definir detalhadamente os requisitos aos quais o nosso sistema deve responder. O sistema em análise corresponde ao processo de fabrico de produtos de panificação. Neste fabrico estão envolvidos funcionários que executam todas as tarefas necessárias para o fabrico e venda do pão.

O domínio da base de dados, que foi pensado para este projeto, abrange os produtos à venda, o próprio registo das vendas, os clientes e os funcionários da empresa.

De seguida apresentamos os requisitos do funcionamento da empresa do caso em estudo:

Produtos

- O sistema atribuirá automaticamente a cada produto registado, um identificador único
- Cada produto tem um nome.
- Todos os produtos deverão estar guardados no sistema e poderão ser posteriormente adicionados novos.
- Cada produto é confecionado por um funcionário
- A cada produto está associado um stock
- Cada produto tem um tempo de duração de confeção
- O preço de um produto deverá ser maior do que 0€.
- Os Produtos farão parte de um pedido solicitado pelo cliente

Clientes

- Cada cliente terá um número sequencial e único, identificando o cliente de forma inequívoca.
- Cada cliente efetua um ou mais pedidos
- Cada cliente receciona todos os produtos que solicitou no seu pedido
- Não existem limites de idades para o atendimento a clientes

- Um cliente pode ou não efetuar registo com os seguintes dados: número do Cartão de Cidadão, número de identificação fiscal (nif), nome completo, morada (rua, código-postal e respetivo concelho), data de nascimento e contactos (telemóvel e e-mail).
- O número do Cartão de Cidadão e o número de identificação fiscal devem ser únicos.

Funcionários

- O sistema atribuirá automaticamente a cada funcionário que se registe, um número de identificação único entre todos os funcionários.
- Um funcionário terá os seguintes dados registados: número do cartão de cidadão único, número de identificação fiscal único, NIB único, nome, morada (número da porta, rua, código-postal, freguesia e respetivo concelho e distrito), data de nascimento e telefone e e-mail e horário.
- Um funcionário terá um salário decidido pelo gestor da empresa.
- Os funcionários podem confecionar mais do que um produto ou a registar mais do que um pedido.
- Todos os funcionários deverão estar registados no sistema.
- Para se registar, um funcionário terá de fornecer todos os seus dados pessoais, nomeadamente todos os requisitos pedidos pelo sistema para registo descritos no primeiro ponto
- Para se registar, um funcionário deverá ter idade igual ou superior a 18 anos e inferior a 65 anos.

Pedidos

- Cada pedido é registado por um número identificador único.
- Os pedidos podem ser feitos por clientes registados ou não
- Os pedidos especificam a quantidade e os tipos de produtos que o cliente está a adquirir.
- O valor de um pedido é a soma dos preços de cada produto.
- Cada pedido tem uma hora e uma data de registo de pedido
- Cada pedido é registado por um funcionário
- A cada pedido está associado um e somente um cliente
- Um pedido tem um ou mais produtos

3. Modelo Conceptual da Base de Dados

A modelação conceptual é a primeira das três principais fases da metodologia utilizada, e consiste no processo de construção de um modelo de dados a partir de uma análise detalhada aos requisitos. Este modelo é completamente independente de todas as considerações físicas, ou seja, é independente dos detalhes de implementação, das linguagens de programação, das plataformas de hardware, etc. O modelo conceptual será de seguida documentado em nove etapas:

1. Identificar o tipo de entidades;
2. Identificar o tipo de relacionamentos;
3. Identificar e associar atributos com entidades ou tipos de relacionamentos;
4. Determinar o domínio dos atributos;
5. Determinar chaves candidatas, primárias e alternadas;
6. Considerar o uso de conceitos de modelagem avançada;
7. Procurar por redundância no modelo;
8. Validar o modelo conceptual com perguntas do utilizador;
9. Rever o modelo conceptual com o utilizador.

De seguida apresenta-se com algum detalhe o desenvolvimento de cada um destes pontos descritos acima, assim como na Figura 1 está representado o esquema conceptual atualizado relativo ao projeto.

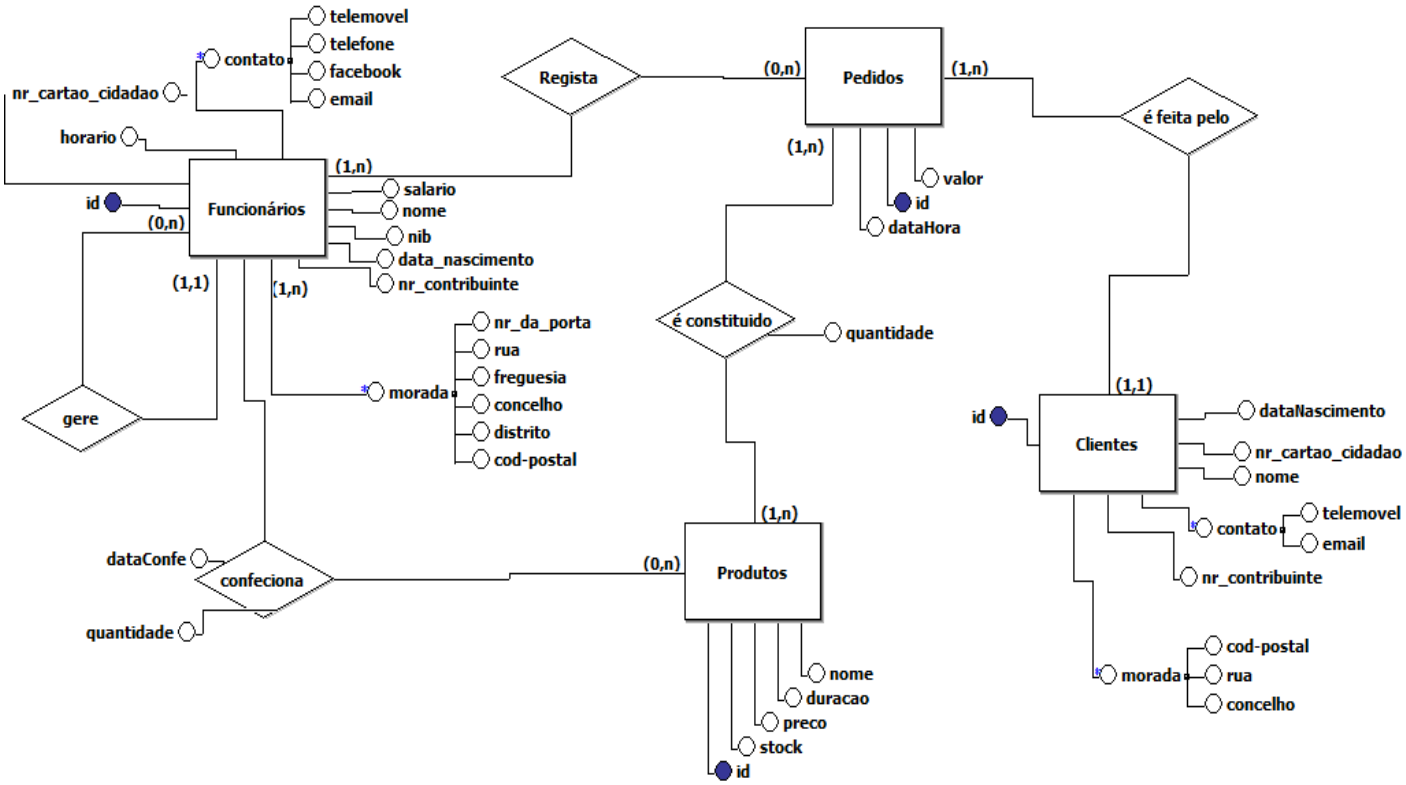

Figura 1 - Esquema conceptual

3.1. Identificar as entidades do problema

O primeiro passo da construção do modelo conceptual passa por identificar as entidades relevantes do problema, que serão os principais objetos em que o utilizador está interessado. A classificação dos objetos do problema como entidades do modelo, ou não, deve ser um processo cuidadoso e metódico. O facto de um objeto do problema ter uma importância considerável não implica que este seja classificado como entidade no modelo conceptual. Inicialmente serão apresentadas as entidades importantes do problema e a justificação da sua existência. De seguida serão documentadas todas as entidades num dicionário de dados com uma descrição, um sinónimo e a justificação pela ocorrência dessa mesma entidade.

3.1.1 Cliente

O cliente é uma das entidades mais importantes no processo de venda, pois vai adquirir os produtos que a *BelaPadaria* produz. O processo de venda torna-se mais rápido e prático se os dados do cliente forem adquiridos previamente, no entanto, o registo de cada cliente não é obrigatório. Posto isto é notável a necessidade da criação da entidade clientes.

3.1.2 Funcionário

Uma outra personagem importante no processo de venda e fabrico será a entidade funcionário. Esta surge na venda como representante desta fábrica de produzir pão, ainda que, nem todos os funcionários desta empresa estejam em contato direto com o público. Surge então a entidade funcionários.

3.1.3 Pedido

A entidade ***Pedido*** servirá para registar todos os produtos e as quantidades pedidas por cada entidade ***Cliente***. Como tal, o registo pormenorizado dos pedidos são cruciais no nosso problema.

3.1.4 Produto

Os diferentes tipos de produtos devem ser caracterizados com detalhe de modo a serem facilmente identificados. Surgem portanto a entidade produto. Existem vários tipos de pães e ainda alguns tipos de bolos, como baguetes, pão bijou, pão centeio, broa de milho, pão com chouriço, bolas de Berlim, tarte de morango poderão ser alguns dos exemplos dos produtos presentes no sistema.

3.1.5 Dicionário de dados das entidades

Nome da Entidade	Descrição	Sinónimos	Ocorrência
Clientes	Esta entidade representa as várias pessoas que compram produtos na BelaPadaria.	Consumidor	Cliente é das entidades mais importantes e principais pois são quem efetuam compras na padaria.
Produtos	Esta entidade representa os produtos finais que são vendidos pela BelaPadaria.	Artigo	Produto é outra das entidades mais relevantes do problema. É o produto que é vendido ao cliente.
Funcionários	Entidade que representa a várias pessoas que trabalham na BelaPadaria.	Empregado	O Funcionário pode ter vários cargos e são os funcionários que fazem a empresa e que a representam.
Pedidos	Esta entidade representa o pedido que o cliente faz a BelaPadaria.	Pedido	O pedido é composto por um ou mais produtos que o cliente pretende comprar.

Tabela 1 - Dicionário de dados das entidades

3.2. Identificar tipos de relacionamentos

Após identificar as entidades é necessário identificar os seus relacionamentos. Em primeiro lugar será apresentado um diagrama ER do modelo para ser mais fácil perceber as entidades e a forma como elas se relacionam entre si. A acompanhar o modelo ER será apresentado o Dicionário de dados dos relacionamentos.

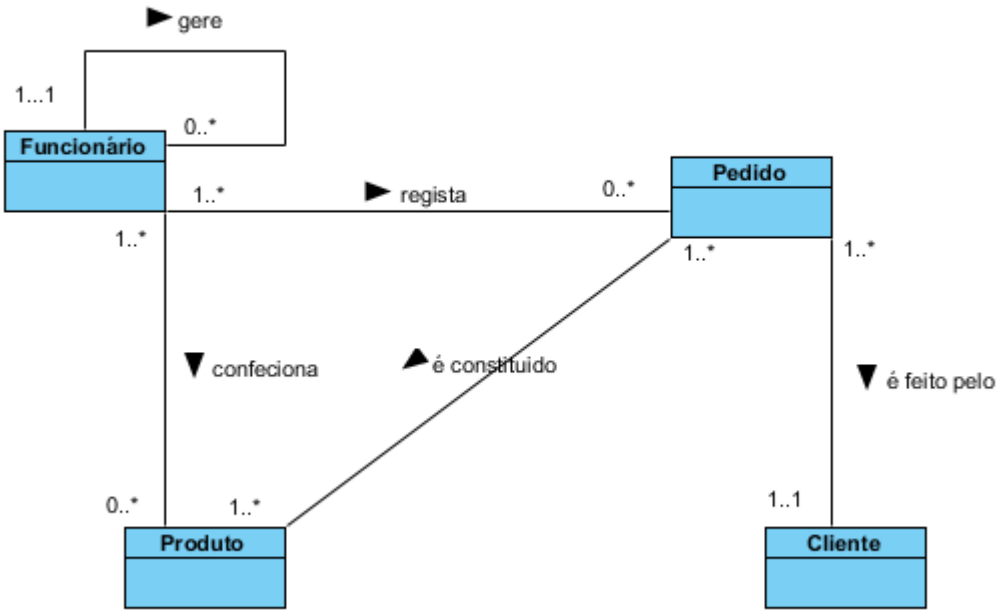

Figura 2 - Relacionamento entre Entidades

3.2.1 Dicionário de dados dos relacionamentos

Nome Entidade	Multiplicidade	Relacionamento	Multiplicidade	Nome Entidade
Funcionário	(1, 1)	Gere	(0, n)	Funcionário
Funcionário	(1, n)	Regista	(0, n)	Pedido
Funcionário	(1, n)	confeciona	(0, n)	Produto
Pedido	(1, n)	é feito pelo	(1, 1)	Cliente
Pedido	(1, n)	é constituído	(1, n)	Produto

Tabela 2 - Dicionário de dados dos relacionamentos

3.3. Identificar e relacionar atributos com as entidades e tipos de relacionamentos/ Determinar domínios de atributo

O passo seguinte da metodologia passa por identificar os atributos de cada entidade e de cada relacionamento. Para tal identificação é necessário analisar os requisitos e perguntar "Qual a informação que preciso para guardar esta entidade ou este relacionamento?". Será também feita com esta fase a atribuição dos domínios dos atributos, ou seja, atribuir o intervalo de valores que os atributos podem tomar.

Apresentar-se-á então de seguida o dicionário de dados com os nomes das entidades, os seus atributos, a descrição de cada atributo, o domínio de valores, se o atributo pode ser nulo ou não e o tipo de atributo (simples, composto, derivado ou multi-valor).

3.3.1 Dicionário de dados dos atributos das entidades

Entidade	Atributo	Descrição	Tipo de dados e tamanho	Nulo	Chaves
Clientes	IdCliente	Código que identifica univocamente um cliente.	Int	Não	Primária
	Nome	Nome completo do cliente.	Varchar 75	Sim	
	NrCartaoCidadao	Número do cartão de cidadão que identifica o cliente.	Numeric 8	Sim	
	NrContribuinte	Número de Identificação Fiscal.	Numeric 9	Sim	
	DataNascimento	Data de nascimento do cliente.	DATE	Sim	
	CódigoPostal	Código postal da residência do cliente	Varchar 8	Sim	
	Rua	Nome da rua do cliente, onde reside	Varchar 75	Sim	
	Concelho	Nome do concelho do cliente.	Varchar 75	Sim	
	Telemóvel	Número de telemóvel do cliente.	Numeric 9	Sim	
	E-mail	E-mail do cliente.	Varchar 75	Sim	
Produtos	idProduto	Número que identifica o produto.	Int	Não	Primária

	nome	Designação do produto, o seu tipo. (Broa de milho, baguete, bijou…)	Varchar 75	Não	
	preço	Preço do produto.	Numeric 7, 2	Não	
	stock	Stock do produto.	Int (10)	Não	
	duração	Tempo que demora a um produto estar pronto.	TIME	Sim	
Funcionários	idFuncionario	Número que identifica o funcionário.	Numeric 8	Não	Primária
	Nome	Nome do funcionário.	Varchar 75	Não	
	Salário	Salário que o funcionário recebe por parte da empresa.	Numeric 8, 2	Não	
	NIB	Número de identificação bancária, para a qual recebe o salário.	Numeric 22	Não	
	NrCartaoCidadao	Número do cartão de cidadão do funcionario.	Numeric 8	Não	
	NrContribuinte	Número de identificação fiscal do funcionário.	Numeric 9	Não	
	DataNascimento	Data de nascimento do funcionário.	DATE	Não	
	CódigoPostal	Código Postal do funcionário.	Numeric 8	Não	
	Freguesia	Nome da freguesia do funcionário.	Varchar 75	Não	
	NºPorta	Número da porta do funcionário.	Varchar 10	Sim	
	Rua	Nome da rua do funcionário.	Varchar 75	Sim	
	Concelho	Nome do concelho do funcionário.	Varchar 75	Não	
	Distrito	Nome do distrito do funcionário.	Varchar 75	Não	
	Telemóvel	Número de telemóvel do funcionário.	Numeric 9	Não	
	Telefone	Número de telefone do funcionário.	Numeric 9	Sim	
	Facebook	Endereço de facebook do funcionário.	Varchar 75	Sim	
	E-mail	Endereço de e-mail do funcionário.	Varchar 75	Sim	

	horario	Número inteiro correspondente a um horário que determina o número de horas que um funcionário trabalha	Número inteiro	Não	
Pedidos	id	Número que identifica o pedido efetuado pelo cliente/recebido pelo funcionário.	Número inteiro	Não	Primária
	valor	Valor total dos produtos constituintes do pedido (soma dos preços)	DECIMAL 7, 2	Não	
	dataHora	Data e hora a que o pedido foi entregue e devidamente pago.	DATETIME	Não	

Tabela 3 - Dicionário de dados dos atributos das entidades

3.3.2 Dicionário de dados dos atributos das relações

Relação	Atributo	Descrição	Tipo de dados e tamanho	Nulo
Pedido é constituído por produto	quantidade	A quantidade de cada produto que constitui o produto	Numeric 8	Não
Produto é confecionado por funcionário	quantidade	A quantidade de produtos que foram confecionados pelo funcionário	Numeric 8	Não
	dataConfe	A data de confeção de um produto por um funcionário	DATETIME	Sim

Tabela 4 - Dicionário de dados dos atributos dos relacionamentos

3.4. Determinar chaves candidatas, primárias e alternadas

Nesta etapa é necessário determinar qual o conjunto das chaves candidatas do conjunto de atributos de uma entidade. A partir dessas chaves candidatas é escolhida a chave primária e as restantes poderão ser chaves alternadas. Uma chave alternada é então uma chave que é candidata, não foi eleita primária, mas sobre a qual se irão fazer várias pesquisas. É importante eleger-se chaves alternadas se se verificar que se irão fazer várias pesquisas sobre esse atributo, pois leva à criação um índice ordenado por esse mesmo atributo aumentando assim a eficiência dessas pesquisas. Começamos então por demostrar um diagrama de relacionamento de entidades com as chaves primárias adicionadas.

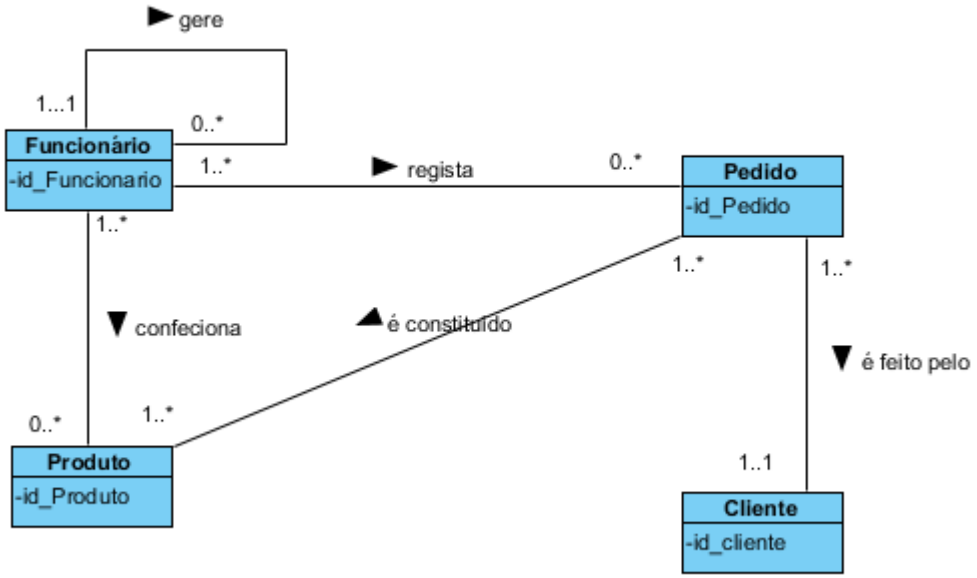

Figura 3 - Diagrama ER com as chaves primárias adicionadas

Documentar-se-á de seguida as escolhas e as justificações das chaves para cada entidade:

Clientes:

Chaves candidatas: {idCliente, nr_cartao_cidadao, nr_contribuinte}

Chave primária: A chave primária escolhida será o id pois, apesar de todos serem do mesmo domínio (números inteiros), é o que tem menor ordem de grandeza.

Chaves Alternadas: nr_cartao_cidadao,nr_contribuinte

Pedidos:

Chaves candidatas: {idPedidos}

Chave primária: A escolha da chave primária recai na única chave candidata, id.

Chaves Alternadas: Nenhuma.

Funcionários:

Chaves candidatas: {idFuncionarios, nr_cartao_cidadao, nr_contribuinte}

Chave primária: A chave primária escolhida será o id pois, apesar de todos serem do mesmo domínio (números inteiros), é o que tem menor ordem de grandeza.

Chaves Alternadas: nr_cartao_cidadao, nr_contribuinte

Produtos:

Chaves candidatas: {idproduto, Nome}

Chave primária: A chave primária escolhida será o id pois, comparando com o nome, requer menor carga computacional pois o nome é uma string e o id um inteiro.

Chaves Alternadas: Nenhuma.

3.5. Considerar o uso de modelação avançada

Nesta etapa da metodologia é necessário verificar se é aplicável o uso de conceitos de modelação avançada tais como agregações ou composições. No entanto, como este se trata de um passo opcional da Modelação Conceptual, e dada a elevada importância que possui representar claramente as entidades e relacionamentos mais importantes, optamos por não implementar esta generalização e manter assim a simplicidade do diagrama ER.

3.6. Verificar que não existem redundâncias no modelo

Nesta etapa do projeto é necessário analisar o modelo conceptual em busca de redundância e eliminá-la caso exista. Num primeiro passo devem ser examinados todos os relacionamentos de multiplicidade um-para-um. Neste passo, o objetivo é rever o modelo de forma a identificar se existe alguma redundância presente e no caso de existir, removê-la. Observando o modelo elaborado, constata-se que não existem relacionamentos do tipo um-para-um, logo não existe a possibilidade de haver redundância neste aspeto.

No segundo passo do processo devem ser removidos os relacionamentos redundantes. Um relacionamento é redundante quando é possível obter a mesma informação à custa de outros relacionamentos. É relativamente fácil verificar a existência de múltiplos

caminhos entre duas entidades do modelo, contudo isso não implica que existam relacionamentos redundantes entre elas, pois tais relacionamentos podem representar associações distintas.

3.7. Validar Modelo com perguntas do utilizador

Neste momento está definido um modelo conceptual que representa os requisitos impostos pela *BelaPadaria*. Assim, justifica-se então a verificação de todo o modelo de forma a garantir que este consegue dar resposta a todas as interrogações requeridas pelo utilizador. Se o modelo não conseguir responder a alguma das interrogações analisadas, então significa que houve algum erro na modelação, e poderão ter de ser repetidos alguns passos da modelação conceptual já realizados. Apresentar-se-ão de seguida as perguntas e as suas respostas tendo em conta as entidades e relacionamentos que serão usados para responder.

Quantos produtos foram fabricados num dia de uma determinada semana?

Para saber quantos produtos foram fabricados num dia de uma determinada semana, apenas necessitamos da entidade Produto, a partir da qual conseguimos obter toda a informação necessária para a obtenção do resultado pretendido.

Qual o preço de determinado produto?

Para sabermos o preço de determinado produto, basta acedermos a entidade produto que contem o atributo preço que nos diz exatamente qual o preço do produto em causa.

Qual o stock existente de determinado produto?

Para sabermos o stock existente, temos como atributo da entidade produto o stock que nos diz exatamente o valor que temos em stock. Desta forma evitamos a quebra na venda por falta de produto a ser comercializado.

Qual o número de pedidos efetuados entre duas datas?

Esta pergunta é essencial para perceber o volume de vendas em determinadas épocas, para sabermos quantos pedidos foram efetivamente efetuados entre duas datas basta acedermos a entidade pedidos que possui o atributo data e daí fazer a pesquisa pretendida.

Qual o número de pedidos que determinado cliente fez entre duas datas?

Esta pergunta é importante, pois percebe-se desta forma, se o cliente está ou não satisfeito com os nossos produtos, para sabermos quantos pedidos foram efetivamente efetuados entre

duas datas por um determinado cliente, temos que inicialmente saber quais os clientes que efetuaram pedidos e depois basta somar o número de pedidos nas respetivas datas.

Que funcionário registou determinado pedido?

Esta pergunta é importante no sentido de, se houver algum problema com algum pedido de algum cliente, sabe-se, a qual funcionário nos devemos de dirigir para resolver a situação. Desta forma para sabermos esta informação acedemos a entidade funcionário que registou determinado pedido e conseguimos toda a informação necessária para identificar o individuo.

Quanto tempo demora a fazer um certo produto?

Saber o tempo necessário que irá demorar a confeção de determinado produto para informar o cliente é importante e então para obtermos tal informação acedemos a entidade produto que contem o atributo duração que nos informa diretamente quanto ao tempo que efetivamente pretendemos saber.

Qual a quantidade de vendas de um certo produto?

É necessário saber quais são os produtos que se vendem mais, para que a padaria os confecione ou compre em maiores quantidades; reduzindo desta forma a compra ou fabrico dos produtos que são menos vendidos.

Quais foram os produtos confecionados por um determinado funcionário cujo pedido foi efetuado numa determinada data?

Acedemos a entidade pedido que contem o atributo data no qual fazemos uma pesquisa pelo dia pedido, e daqui resultam todos os pedidos referentes ao dia em causa. De seguida para cada um desses pedidos consultamos todos os produtos que o constituem e para cada um desses produtos consultamos o funcionário que o confecionou fazendo uma pesquisa pelo id do funcionário.

Quais foram os clientes que foram atendidos por determinado funcionário?

Para determinar quais foram os clientes que foram atendidos por determinado funcionário acedemos a entidade pedido e dessa forma obtemos todos os pedidos que foram registados pelo funcionário em questão. De seguida para cada um desses pedidos conseguimos o cliente que o efetuou.

Que funcionário registou determinado pedido?

Para responder a esta pergunta é necessário cruzar a informação da entidade funcionário e entidade pedido.

3.8. Rever e validar o Modelo com o Utilizador

Para finalizar esta fase, o modelo conceptual desenhado deve ser revisto com o utilizador. Este processo é extremamente importante na medida em que o utilizador deve ser capaz de reconhecer, assinando, que o modelo de dados concebido é uma representação real dos requisitos da empresa a ser modelada. Este passo é também uma forma de avaliar o trabalho feito face aos requisitos do utilizador. Foi avaliada toda a documentação associada ao modelo de dados, ou seja, o dicionário de dados de entidades, o dicionário de dados de relacionamentos, o dicionário de dados de atributos e o diagrama ER.

Na revisão do dicionário de entidades foi possível identificar todas as entidades importantes no modelo e constatar com o cliente que correspondem aos objetos considerados importantes no problema e às necessidades da empresa. Na revisão do dicionário de relacionamentos foi justificada a forma como as entidades estão associadas e o utilizador verificou que estes relacionamentos correspondem à realidade do problema. Já na revisão do dicionário de atributos o utilizador pode verificar mais concretamente a informação associada a cada entidade e validar o domínio de cada atributo, constatando que estes correspondem aos requisitos anteriormente apresentados. Por fim, na revisão do diagrama ER foi possível obter uma visão geral do modelo de dados e realizar a verificação de algumas das perguntas mais frequentes no problema de modo a testar as capacidades do mesmo. O modelo de dados foi aprovado pelo cliente como sendo uma solução que dá resposta a todas as necessidades da empresa.

4. Modelo Lógico

Terminada a modelação conceptual, procedemos à fase de modelação lógica. Esta consiste na tradução do modelo conceptual anteriormente desenvolvido para um modelo lógico capaz de representar os requisitos definidos, assim como a validação do mesmo. Após a conclusão desta fase, deveremos obter um modelo lógico único representativo dos requisitos do sistema. Este modelo está apresentado na figura abaixo:

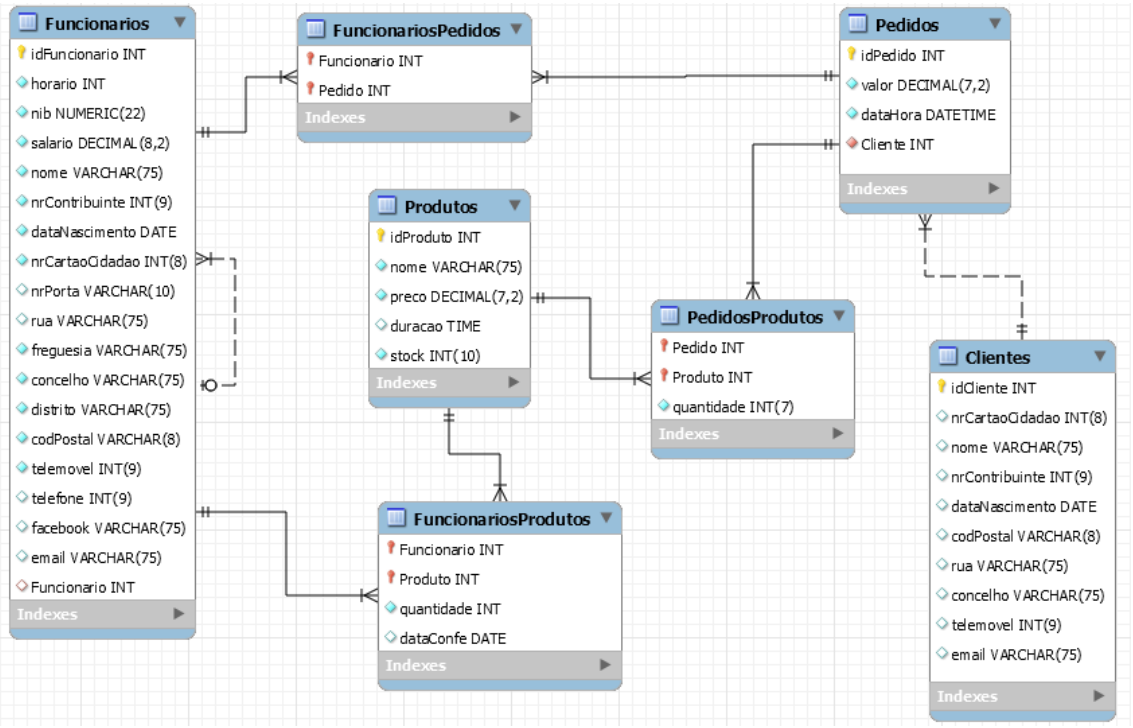

Figura 4 - Modelo Lógico

4.1. Derivação das relações para o modelo lógico

Neste primeiro passo, é necessário derivar as entidades, relacionamentos e atributos considerados no modelo conceptual para as relações do modelo lógico. Foi usada a DDL

(Database Definition Language), onde se descreve a composição de cada uma das relações, tendo por base as seguintes estruturas que se podem encontrar no modelo conceptual:

1 - Entidades Fortes

2- Relacionamentos muitos-para-muitos

3- Relacionamentos um-para-muitos Recursivos

4- Relacionamentos um-para-muitos

Para cada um dos pontos acima listados será descrito como foram abordados no contexto do problema de forma a derivar as relações e os relacionamentos alcançados no modelo lógico.

1 - Entidades Fortes

Uma entidade forte é uma entidade que não depende da existência de outra entidade e que pode por si só formar uma chave primária com os seus atributos. No modelo desenvolvido todas as nossas entidades são fortes.

- **Clientes** (idCliente, nrCartaoCIdadao, nome, nrContribuinte, dataNascimento, codPostal, rua, concelho, telemóvel, email)
  Chave Primária: idCliente

- **Funcionarios** (idFuncionarios, horario, nib, salario, nome, nrContribuinte, dataNascimento, nrCartaoCidadao, nrPorta,rua, freguesia, concelho, distrito, codPostal, telemóvel, telefone, facebook, email, Funcionario)
  Chave Primária: idFuncionarios
  Chave Estrangeira: Funcionario

- **Pedidos** (idPedido, valor, dataHora)
  Chave Primária: idPedido

- **Produtos** (idProduto, nome, preço, duracao, stock)
  Chave Primária: idProduto

2- Relacionamentos muitos-para-muitos

Quando este tipo de relacionamentos acontece é criada uma nova relação que conterá todos os atributos que fazem parte do relacionamento. A nova relação terá uma cópia das chaves primárias das entidades que participam no relacionamento e funcionam como chaves estrangeiras. Uma dessas chaves ou mesmo as duas formarão a chave primária da nova relação ou ainda em combinações com outros atributos da nova relação. Desta forma surgem as seguintes relações:

- **Funcionarios - Pedidos**

  Resulta:

  FuncionariosPedidos (Funcionario, Pedido)
  Chaves Estrangeiras: Funcionario correspondente a idFuncionario

Pedido correspondente a idPedido
Chave Primária: (Funcionario, Pedido)

- **Funcionarios – Produtos**

  Resulta:
  FuncionariosProdutos (Funcionario, Produto, quantidade)
  Chaves Estrangeiras: Funcionario correspondente a idFuncionario
  Produto correspondente a idProduto
  Chaves Primárias: (Funcionario, Produto)

- **Pedidos – Produtos**

  Resulta:
  PedidosProdutos (Pedido, Produto, quantidade)
  Chaves Estrangeiras: Pedido correspondente a idPedido
  Produto correspondente a idProduto
  Chaves Primárias: (Pedido, Produto)

3- Relacionamentos um-para-muitos Recursivos

Neste tipo de relacionamentos, a entidade de multiplicidade "muitos" fica com um novo atributo (chave estrangeira) que é a chave primária representante da outra entidade. Temos no nosso modelo a seguinte:

- **Funcionarios** (idFuncionarios, horário, nib, salario, nome, nrContribuinte, dataNascimento, nrCartaoCidadao, nrPorta, rua, freguesia, concelho, distrito, codPostal, telemovel, telefone, facebook, email, Funcionario)

  Chave Primária: idFuncionarios
  Chave Estrangeira: Funcionario

4- Relacionamentos um-para-muitos

Neste tipo de relacionamentos, a entidade de multiplicidade "muitos" fica com um novo atributo (chave estrangeira) que é a chave primária representante da outra entidade. Temos no nosso modelo a seguinte

- **Pedidos** (idPedido,valor, dataHora, Cliente)
  Chaves Primárias: (idPedido)
  Chaves Estrangeiras: Cliente correspondente a idCliente

- **Clientes** (idCliente, nrCartaoCidadao, nome, nrContribuinte, datNascimento, codPostal, rua, concelho, telemóvel, email)
  Chaves Primárias: (idCliente)

4.2. Validação das relações através da normalização

Esta fase do processo tem como objetivo decidir quais os atributos que pertencem em conjunto numa relação. Serão validados os atributos presentes em cada relação de modo a assegurar que estes são os necessários para sustentar os requisitos de dados. As relações devem da mesma forma ser validadas para evitar redundância e situações de inconsistência aquando de inserções ou atualizações. Devem ser identificadas as dependências funcionais entre atributos em cada relação e todas as chaves primárias. De notar que estas dependências funcionais identificam os relacionamentos do modelo de dados. O resultado da normalização é um modelo lógico estruturalmente consistente e que possui mínima redundância. No entanto, às vezes é discutido que um desenho de uma base de dados normalizada não oferece o máximo de eficiência de processamento. A normalização força-nos a perceber completamente cada atributo e o que representa na base de dados. O processo de normalização será executado até à terceira forma normal.

4.2.1 1FN – 1ª Forma Normal

Para aplicar a 1FN numa tabela desnormalizada, é necessário identificar os atributos ou grupos de atributos repetidos. Um atributo, ou grupo de atributos repetidos, é aquele que ocorre numa tabela várias vezes para uma mesma chave primária.

Sendo que as regras da primeira forma normal já foram previamente aplicadas e que todos os atributos já foram devidamente identificados, resta-nos apenas verificar que todos os atributos presentes nas relações se tratam de atributos atómicos.

Como o nosso modelo relacional não possui atributos multi-valor nem grupos repetidos, concluímos que respeita a 1ªForma Normal.

4.2.2 2FN – 2ª Forma Normal

Para que a segunda forma normal seja verificada é necessário que o modelo de dados se encontre já na primeira forma normal e que os atributos não-chave de relações que contenham chaves primárias compostas sejam funcionalmente dependentes da totalidade da chave, ou seja, não podem existir atributos numa relação que sejam dependentes apenas de uma parte da chave, pois caso isso se verificasse, o modelo iria sofrer de anomalias de atualização.

Neste caso como também já foram tomadas considerações acerca das dependências funcionais nas relações, também já não é possível encontrar no modelo casos em que a segunda forma normal seja desrespeitada.

Assim como os atributos nessa tabela são dependentes da totalidade da chave, a segunda forma normal já se verifica.

4.2.3 3FN – 3ª Forma Normal

Para que uma relação se encontre na terceira forma normal, é necessário que se encontre já na segunda forma normal e que para além disso não existam nela atributos não-chave-primária que sejam transitivamente dependentes da chave primária, ou seja, não pode existir um atributo que seja ao mesmo tempo funcionalmente dependente da chave primária e de um outro atributo. Assim, para que seja verificada a terceira forma normal, é necessário eliminar as dependências transitivas presentes nas relações do modelo. Sendo assim, concluímos que todas as relações respeitam a 3ª Forma Normal pois todos os atributos não-chave dependem, inteira e exclusivamente, da totalidade da chave.

Após a verificação de todas as relações do nosso modelo relacional, concluímos que este se encontra normalizado até à 3ªForma Normal.

4.3. Validação das relações através das transações dos utilizadores

Finalizada a derivação do Modelo Lógico, é necessário assegurar que a partir deste é possível obter resposta a todas as perguntas que um utilizador possa realizar. Assim, similarmente à verificação efetuada na Modelação Conceptual, foi testada se as transações requisitadas pelo utilizador são também suportadas pelo Modelo Lógico.

1ª Transação: Inserir pedido, associar funcionário ao pedido e adicionar produto ao pedido

Nesta transação pretende-se demonstrar o caminho do registo de um pedido. Para tal será inserido um novo pedido aos pedidos existentes na nossa base de dados, com a inserção deste novo pedido irá ser necessário definir o funcionário, o cliente e a quantidade dos produtos constituintes do pedido. Será feita a inserção na tabela FuncionariosPedidos o Funcionario correspondente ao idPedido, assim como a inserção na tabela PedidosProdutos o

produto e a respetiva quantidade. Sendo desta forma executada a transação de registo de um pedido.

2ª Transação: Atualização do stock de um produto aquando da confeção de um produto por um funcionário

Nesta transação pretende-se que o valor do stock se mantenha consistente, para tal sempre que um funcionário confeciona uma quantidade de produtos o stock desse produto necessita de ser atualizado. Será feita a inserção na tabela FuncionariosProdutos o funcionário responsável pela confeção do produto, a quantidade e data de confeção. Desta forma será feita uma atualização ao stock na tabela Produtos, mantendo assim a coerência de dados.

3ª Transação – Substituir o funcionário despedido na tabela FuncionariosPedidos e eliminar o funcionário despedido na tabela Funcionarios

Nesta transação é atualizada a nossa lista de funcionarios, será feita uma atualização na tabela FuncionarioPedidos onde ocorre a substituição do funcionário despedido pelo novo funcionário. De seguida será efetuada a eliminação do funcionário despedido da tabela Funcionarios.

4.4. Análise das restrições de integridade

O objetivo agora é garantir que o modelo lógico que foi construído representa de forma correta os dados pretendidos para a empresa *BelaPadaria*. Para que o modelo lógico esteja de acordo com o pretendido é preciso garantir que todas as restrições de integridade são respeitadas e portanto o que está feito é o pretendido para a base de dados da *BelaPadaria.*

Se os pontos apresentados a seguir forem respeitados garante-se que a base de dados contem dados credíveis e que a sua manipulação é consistente e sem falhas nas regras de negócio.

1 - Necessidade de valores não-nulos:

As chaves primárias são valores que não podem ser nulos mas podem existir outros que por regras de negócio também se enquadram neste ponto. Todos os atributos que podem ter esta característica podem ser visualizados no dicionário de dados lógico.

2 - Restrições de domínio do atributo:

Todos os atributos tem a si associado um domínio que define a gama ou tipo de valores que cada atributo pode ter. Desta forma tenta-se dar algum significado a cada atributo guardado de forma a diminuir possíveis erros futuros do utilizador. O domínio dos atributos foi apurada no desenvolvimento já efetuado e pode ser visualizado no dicionário de dados do modelo conceptual.

3 – Multiplicidade:

A multiplicidade é uma restrição que diz respeito ao relacionamento entre entidades. É esta regra que identifica a cardinalidade que ocorre entre duas relações, ou seja, identifica a forma como os dados se relacionam. A multiplicidade está presente desde logo no modelo conceptual e consequentemente irá passar para o modelo lógico. O modelo lógico gerado está dependente da multiplicidade presente no modelo conceptual e portanto para cada multiplicidade diferente é gerado uma solução diferente.

4 - Integridade da entidade:

Esta integridade é garantida pelo uso de uma chave primária que seja única e não nula. Desta forma garante-se que cada tuplo é identificado univocamente por uma chave na relação. A forma como a chave primária é formada depende do caso em estudo mas desde que se cumpra as duas regras apresentada esta integridade será cumprida. Esta restrição foi considerada na identificação das chaves primárias e explicita no dicionário de dados.

5 - Integridade referencial:

Esta regra de integridade diz respeito a forma como as chaves estrangeiras são utilizadas. Independentemente da sua utilização é obrigatório garantir que os relacionamentos entre as relações sejam preservados e o modelo relacional seja respeitado, ou seja, esta restrição pretende garantir que o valor de uma chave estrangeira de uma relação “filho” seja o mesmo e exista na chave primária da relação “pai” associada. Sempre que uma chave estrangeira sofra alterações devido a remoções e modificações esta integridade deve ser verificada. O dicionário de dados explicita que as chaves estrangeiras são eleitas chaves primárias de outras tabelas. Esta restrição supervisiona a consistência da base de dados, uma vez que uma alteração numa chave primária deve ser refletida nas chaves estrangeiras a esta associada.

Esta restrição ajuda na consistência da base de dados e deve portanto ser verificada sempre que uma relação "pai" sofra qualquer tipo de alteração ou remoção.

- **Inserir um registo na relação filho –** Apenas se verifica se a chave estrangeira na relação filho têm alguma correspondência com o valor da chave primária na relação pai.

- **Remover um registo da relação filho –** Uma remoção na relação filho não afeta a integridade em qualquer aspeto.

- **Atualização de um registo na relação filho –** O procedimento é totalmente igual ao de inserir um registo na relação filho.

- **Inserir um registo na relação pai –** Uma inserção na relação pai não interfere com a integridade referencial. Contudo de modo manter a informação atualizada e consistente é necessário proceder à atualização nas respetivas relações filho.

- **Remover registo na relação pai –** No caso da remoção de um registo na relação pai a integridade referencial pode ser posta em causa, na medida em que pode existir um registo numa relação filho sem uma entidade pai. Neste sentido, foi decidido que iriamos utilizar a funcionalidade *NO ACTION* que impede que a remoção de um registo na relação pai se existir algum registo filho. Para além disso, foi também utilizada a função *CASCADE* de modo a atualizar as chaves estrangeiras associadas a esta chave primária.

- **Atualizar a chave primária num registo pai –** Da mesma forma que a situação anterior, as atualizações da chave primária num registo pai devem ser refletidas na relação filho. Por isso foram novamente consideradas as estratégias de *NO ACTION* e *CASCADE*.

6 - Restrições gerais:

Por fim, é necessário ver representadas no modelo as restrições que dizem respeito a regras de negócio. Todas as restrições deste tipo serão apresentadas abaixo, agrupadas por relação. Como todas as restrições acima enunciadas estas também têm de ser refletidas no modelo de forma a que o modelo represente a organização da BelaPadaria de forma correta.

Clientes

• O número do BI, NIF, email são únicos

Esta restrição está definida no script de criação das tabelas, passaremos a mostrar na imagem a baixo um exemplo:

```
53  CREATE TABLE IF NOT EXISTS `belapadaria`.`Clientes` (
54    `idCliente` INT NOT NULL AUTO_INCREMENT ,
55    `nrCartaoCidadao` INT(8) NULL UNIQUE,
56    `nome` VARCHAR(75) NULL,
57    `nrContribuinte` INT(9) NULL UNIQUE,
58    `dataNascimento` DATE NULL,
59    `codPostal` VARCHAR(8) NULL,
60    `rua` VARCHAR(75) NULL,
61    `concelho` VARCHAR(75) NULL,
62    `telemovel` INT(9) NULL,
63    `email` VARCHAR(75) NULL UNIQUE,
64    PRIMARY KEY (`idCliente`))
65  ENGINE = InnoDB;
```

Figura 5 - Excerto da script criação da tabela Clientes

Funcionários

- Para se registar, um funcionário deverá ter idade igual ou superior a 18 anos e inferior a 65 anos, o seu salário deverá ser superior a 0 €
- O número do BI, NIF, email, nib, facebook são únicos

```
20  CREATE TABLE IF NOT EXISTS `belapadaria`.`Funcionarios` (
21    `idFuncionario` INT NOT NULL AUTO_INCREMENT,
22    `horario` INT NOT NULL,
23    `nib` DECIMAL(22) NOT NULL UNIQUE,
24    `salario` DECIMAL(8,2) NOT NULL CHECK (salario>0),
25    `nome` VARCHAR(75) NOT NULL,
26    `nrContribuinte` INT(9) NOT NULL UNIQUE,
27    `dataNascimento` DATE NOT NULL,
28    `nrCartaoCidadao` INT(8) NOT NULL UNIQUE,
29    `nrPorta` VARCHAR(10) NULL,
30    `rua` VARCHAR(75) NULL,
31    `freguesia` VARCHAR(75) NOT NULL,
32    `concelho` VARCHAR(75) NOT NULL,
33    `distrito` VARCHAR(75) NOT NULL,
34    `codPostal` VARCHAR(8) NOT NULL,
35    `telemovel` INT(9) NOT NULL,
36    `telefone` INT(9) NULL,
37    `facebook` VARCHAR(75) NULL UNIQUE,
38    `email` VARCHAR(75) NULL UNIQUE,
39    `Funcionario` INT NULL,
40    PRIMARY KEY (`idFuncionario`),
41    INDEX `fk_Funcionarios_Funcionarios_idx` (`Funcionario` ASC),
42    CONSTRAINT `fk_Funcionarios_Funcionarios`
43      FOREIGN KEY (`Funcionario`)
44      REFERENCES `belapadaria`.`Funcionarios` (`idFuncionario`)
45      ON UPDATE CASCADE
46      ON DELETE SET NULL )
```

Figura 6 - Excerto da script criação da tabela Funcionarios

Produtos

• Os preços dos produtos deverão ser superior a zero euros

```
89  CREATE TABLE IF NOT EXISTS `belapadaria`.`Produtos` (
90    `idProduto` INT NOT NULL AUTO_INCREMENT,
91    `nome` VARCHAR(75) NOT NULL,
92    `preco` DECIMAL(7,2) NOT NULL CHECK (preco>0),
93    `duracao` TIME NULL,
94    `stock` INT(10) NOT NULL CHECK (stock>=0),
95    PRIMARY KEY (`idProduto`))
96  ENGINE = InnoDB;
```

Figura 7 – Excerto da Criação da script criação da tabela Produtos

Pedidos

• O valor do pedido deverá ter um valor superior a zero

```
71  CREATE TABLE IF NOT EXISTS `belapadaria`.`Pedidos` (
72    `idPedido` INT NOT NULL AUTO_INCREMENT ,
73    `valor` DECIMAL(7,2) NOT NULL CHECK (valor>0),
74    `dataHora` DATETIME NOT NULL,
75    `Cliente` INT NOT NULL,
76    PRIMARY KEY (`idPedido`),
77    INDEX `fk_Pedidos_Clientes1_idx` (`Cliente` ASC),
78    CONSTRAINT `fk_Pedidos_Clientes1`
79      FOREIGN KEY (`Cliente`)
80      REFERENCES `belapadaria`.`Clientes` (`idCliente`)
81      ON UPDATE CASCADE
82      ON DELETE NO ACTION)
83  ENGINE = InnoDB;
```

Figura 8 - Excerto da script de criação da tabela Pedidos

4.5. Revisão do modelo lógico com o utilizador

A revisão do modelo lógico junto do utilizador tem como principal objetivo validar o desenho lógico de acordo com as especificações do cliente.

Primeiramente foi apresentado a evolução da arquitetura da sua fase inicial, conceptual, para a fase lógica. Recorrendo ao dicionário lógico, foi possível caracterizar toda a estrutura a nível da sua robustez, de garantias de consistência de dados, e identificar de que forma a estrutura vai de encontro com a realidade pretendida pelos requisitos do cliente. Aqui foi tomado especial foco em relação a todas restrições que o dicionário lógico definia para cada uma das relações da estrutura.

Numa segunda fase, foi demonstrado através do mapa de transações, que as operações críticas pretendidas podem ser facilmente satisfeitas, com segurança e segundo todos os requisitos explicitados já numa fase anterior. Ainda com recurso ao mapa de transações, ficaram explícitas as áreas da arquitetura mais atacadas com as transações pretendidas. O trabalho desenvolvido nesta fase visou satisfazer todas as necessidades do cliente, sendo que, este não expressou qualquer necessidade de retificações no modelo ou de inclusão de novos elementos.

4.6. Análise do crescimento futuro

O objetivo deste tópico é apresentar uma estimativa do crescimento da base de dados com base em dados que a organização da *BelaPadaria* disponibilizou. O cálculo irá ser feito tendo em conta o motor da base de dados escolhido para implementar o projeto, o '*sql server 2014'*. Inicialmente será calculado o tamanho inicial que a base de dados terá de ter para a *BelaPadaria* começar a funcionar e só posteriormente, prever o futuro crescimento.

Tamanho inicial da base de dados:

Tendo em conta os requisitos do sistema previamente analisados temos que inicialmente a base de dados terá:

- Cerca de 8 funcionários
- Cerca de 22 produtos
- Cerca de 100 clientes
- Cerca de 30 pedidos

Para se poder continuar a analise é necessário saber-se o tamanho que o '*sql server 2014'* utiliza para cada tipo de dados. De seguida, apresenta-se um quadro resumo para os tipos de dados utilizados no projeto e o seu respetivo tamanho.

Tipos de dados	Tamanho
Integer	4 bytes
Varchar(N)	2 + N bytes
DATE	3 bytes
DATETIME	8 bytes

DECIMAL (N, M)	N + M < 10 => 5 bytes N + M < 20 => 9 bytes N + M < 29 => 13 bytes N + M < 39 => 17 bytes
TIME	3 bytes

Tabela 5 - Tipos de dados e tamanhos em Sql Server

	Funcionários	Espaço Ocupado
Atributos	idFuncionario INT	4
	horario INT	4
	nib NUMERIC (22)	13
	salario DECIMAL (8,2)	9
	nome VARCHAR (75)	77
	nrContribuinte INT (9)	36
	dataNascimento DATE	3
	nrCartaoCidadao INT (8)	32
	NrPorta VARCHAR(10)	12
	rua VARCHAR(75)	77
	freguesia VARCHAR(75)	77
	concelho VARCHAR(75)	77
	distrito VARCHAR(75)	77
	codPostal VARCHAR(8)	10
	telemovel INT(9)	36
	telefone INT(9)	36
	facebook VARCHAR (75)	77
	email VARCHAR(75)	77
	Funcionario INT	4
	Total:	734

Como no tamanho inicial da base de dados temos 8 funcionários, basta multiplicar 8*734= 5872 bytes.

Tabela 6 - Tamanho inicial para a tabela Funcionarios

Clientes	Espaço Ocupado
idCliente INT	4
nrCartaoCidadao INT(8)	32
nome VARCHAR(75)	77
nrContribuinte INT(9)	36
dataNascimento DATE	3
codPostal VARCHAR (8)	10
rua VARCHAR (75)	77
concelho VARCHAR (75)	77
telemovel INT(9)	36
email VARCHAR (75)	77
	429

Como no tamanho inicial da base de dados temos 100 clientes, basta multiplicar 100*429= 4290 bytes.

Tabela 7 - Tamanho inicial para a tabela Clientes

Produtos	Espaço Ocupado
idProduto INT	4
nome VARCHAR (75)	77
preco DECIMAL(7,2)	5
stock INT (10)	40
duracao TIME	3
	129

Como no tamanho inicial da base de dados temos 22 produtos, basta multiplicar 22*129= 2838 bytes.

Tabela 8 - Tamanho inicial para a tabela Produtos

Pedidos	Espaço Ocupado
idPedido INT	4
valor DECIMAL(7,2)	5
dataHora DATETIME	8
	17

Como no tamanho inicial da base de dados temos 30 pedidos, basta multiplicar 30*17= 510 bytes.

Tabela 9 - Tamanho inicial para a tabela Pedidos

FuncionariosProdutos	Espaço Ocupado
Funcionario INT	4
Produto INT	4
quantidade INT	4
dataConfe DATE	3
	15

Como no tamanho inicial da base de dados temos 26 entradas na tabela FuncionariosProduto, basta multiplicar 15*26= 390 bytes.

Tabela 10 - Tamanho inicial para a tabela FuncionariosProdutos

FuncionariosPedidos	Espaço Ocupado
Funcionario INT	4
Pedido INT	4
	8

Como no tamanho inicial da base de dados temos 30 entradas na tabela FuncionariosPedidos, basta multiplicar 8*30= 240 bytes.

Tabela 11 - Tamanho inicial para a tabela FuncionariosPedidos

PedidosProdutos	Espaço Ocupado
Pedido INT	4
Produto INT	4
quantidade DECIMAL (7,2)	5
	13

Como no tamanho inicial da base de dados temos 29 entradas na tabela PedidosProdutos, basta multiplicar 13*29= 377 bytes

Tabela 12 - Tamanho inicial para a tabela PedidosProdutos

O tamanho inicial da base de dados será aproximadamente 41755 bytes (40,77Kbytes) para que a *BelaPadaria* posso começar a funcionar.

Feito este passo, é agora importante calcular o crescimento futuro da base de dados. Utilizando os requisitos do sistema previamente levantados:

• Um crescimento mensal de 20 clientes.
• Fazem-se 200 vendas diárias.
• Contrata-se em média 2 funcionários novos por ano
• Adicionam-se 1 nova distribuidora por ano
• Adicionam-se, em média, 5 produtos por mês
• Os pedidos aumentam cerca de 5% por mês

No final do primeiro ano o crescimento da *BelaPadaria* será aproximadamente de 0.2653 Megabytes (278237 bytes) de dados.

5. Modelo Físico

Nesta etapa é efetuada a tradução do modelo lógico desenvolvido anteriormente para um modelo físico a ser implementado num SGBD. Esta etapa é também desenvolvida no sentido de otimizar o modo como os dados são guardados e acedidos. Existem diversas estruturas de armazenamento dados. Ao passo que algumas delas são extremamente eficientes na tarefa de guardar dados, umas são mais eficientes no modo como os dados são acedidos. A escolha das estruturas de armazenamento é naturalmente condicionada pelo SGBD, já que estamos dependentes da disponibilização ou não de estruturas alternativas de armazenamento. O modelo de dado físico deve ser totalmente dirigido para a natureza dos dados e aquilo que se pretende realizar com eles.

Assim, são seguidamente descritas as relações base, organização de ficheiros e índices usados para alcançar um uso eficiente dos dados, e todas as restrições de integridade e medidas de segurança.

5.1. Tradução do modelo lógico para o SGBD escolhido

Neste passo foi decidido como representar as relações base identificadas no modelo lógico, de modo a que as relações e as suas restrições possam ser suportadas pelo SGBD escolhido. Para tal ser possível os três passos seguintes terão de ser corretamente efetuados: desenhar as relações base, desenhar as representações dos dados derivados e desenhar as restrições gerais (de negócio), que descreveremos nos pontos seguintes.

5.1.1 Desenho das relações base

Inicialmente é preciso colocar à disposição toda a informação sobre as relações apresentadas no modelo lógico de dados. Informação que diga respeito a domínios, valores por defeito, valores nulos e todo o tipo de restrições devem ser apresentados nesta fase. Para

cada relacionamento, e de forma muito parecido com o que foi feito para o modelo lógico, será apresentada a informação necessária mas agora tendo em consideração o SGBD escolhido.

Relação Pedidos

Domínio ID:	Inteiro
Domínio Valor:	Decimal
Domínio DataHora	DateTime
Domínio Cliente	Inteiro

Pedidos(

idPedido	ID	NOT NULL,
valor	Valor	NOT NULL,
dataHora	DataHora	NOT NULL,
Cliente	Cliente	NOT NULL

PRIMARY KEY (idPedido)
FOREIGN KEY (Cliente) REFERENCES (idCliente) ON UPDATE CASCADE ON DELETE NO ACTION);

Relação Clientes:

Domínio ID	Inteiro
Domínio NrCartãoCidadão	String de tamanho fixo, tamanho 8
Domínio Nome	String de tamanho variável, tamanho 75
Domínio NrContribuinte	String de tamanho fixo, tamanho 9
Domínio DataNascimento	Data
Domínio CodPostal	String de tamanho variável, tamanho 8
Domínio Rua	String de tamanho variável, tamanho 75
Domínio Concelho	String de tamanho variável, tamanho 75
Domínio Telemóvel	String de tamanho fixo, tamanho 9
Domínio Email	String de tamanho variável, tamanho 75

Clientes(

idCliente	ID	NOT NULL,
nrCartaoCidadao	NrCartãoCidadão	NULL,
nome	Nome	NULL,
nrContribuinte	NrContrinuinte	NULL,
dataNascimento	DataNascimento	NULL,
codPostal	CodPostal	NULL,
rua	Rua	NULL,

concelho	Concelho	NULL,
telemovel	Telemóvel	NULL,
email	Email	NULL,
PRIMARY KEY (idCliente)		
);		

Relação Funcionários

Domínio ID	Inteiro
Domínio Horário	Inteiro
Domínio NIB	Número de vírgula fixa, tamanho 22
Domínio Salário	Decimal positivo com 2 casas decimais
Domínio Nome	String de tamanho variável, tamanho 75
Domínio nrContribuinte	String de tamanho fixo, tamanho 9
Dominio DataNascimento	Data
Domínio NrCartãoCidadão	Inteiro, tamanho 8
Domínio NrPorta	String de tamanho variável, tamanho 10
Domínio Rua	String de tamanho variável, tamanho 75
Domínio Freguesia	String de tamanho variável, tamanho 75
Domínio Concelho	String de tamanho variável, tamanho 75
Domínio Distrito	String de tamanho variável, tamanho 75
Domínio CodPostal	String de tamanho variável, tamanho 8
Domínio Telemóvel	Inteiro de tamanho fixo, tamanho 9
Domínio Telefone	Inteiro de tamanho fixo, tamanho 9
Domínio Facebook	String de tamanho variável, tamanho 75
Domínio Email	String de tamanho variável, tamanho 75
Domínio Funcionário	Inteiro

Funcionários(		
idFuncionario	ID	NOT NULL,
horario	Horário	NOT NULL,
nib	NIB	NOT NULL,
salario	Salário	NOT NULL,
nome	Nome	NOT NULL,
nrContribuinte	NrContribuinte	NOT NULL,
dataNascimento	DataNascimento	NOT NULL,
nrCartaoCidadao	NrCartãoCidadão	NOT NULL,
nrPorta	NrPorta	NULL,
rua	Rua	NULL,

freguesia Freguesia NOT NULL,
concelho Concelho NOT NULL,
distrito Distrito NOT NULL,
codPostal CodPostal NOT NULL,
telemovel Telemóvel NOT NULL,
telefone Telefone NULL,
facebook Facebook NULL,
email Email NULL,
Funcionario Funcionário NULL,
PRIMARY KEY (idFuncionario),
FOREIGN KEY (Funcionario) REFERENCES (idFuncionario) ON UPDATE CASCADE ON DELETE SET NULL);

Relação Produtos:

Domínio ID Inteiro
Domínio Nome String de tamanho variável, tamanho 45
Domínio Preço Decimal positivo com 2 casas decimais
Domínio Duração TEMPO
Domínio Stock Inteiro

Produto(
idProduto ID NOT NULL,
nome Nome NOT NULL,
preco Preço NOT NULL,
duracao Duração NULL,
stock Stock NOT NULL,
PRIMARY KEY (idProduto)
);

Relação FuncionáriosPedidos:

Domínio FuncionárioID Inteiro
Domínio PedidoID Inteiro

FuncionáriosPedidos (
Funcionario FuncionárioID NOT NULL,
Pedido PedidoID NOT NULL,
PRIMARY KEY (Funcionario, Pedido),

FOREIGN KEY (Funcionario) REFERENCES (idFuncionario) ON UPDATE CASCADE ON DELETE CASCADE,
FOREIGN KEY (Pedido) REFERENCES (idPedido) ON UPDATE CASCADE ON DELETE CASCADE);

Relação FuncionáriosProdutos

Domínio FuncionárioID	Inteiro
Domínio ProdutoID	Inteiro
Domínio Quantidade	Inteiro
Domínio dataConfe	Data

FuncionáriosProdutos (

Funcionario	FuncionárioID	NOT NULL,
Produto	ProdutoID	NOT NULL,
quantidade	Quantidade	NOT NULL,
dataConfe	Data Confeção	NULL

PRIMARY KEY (Funcionario, Produto),
FOREIGN KEY (Funcionario) REFERENCES (idFuncionario) ON UPDATE CASCADE ON DELETE CASCADE,
FOREIGN KEY (Produto) REFERENCES (idProduto) ON UPDATE CASCADE ON DELETE CASCADE);

Relação PedidosProdutos:

Domínio PedidoID	Inteiro
Domínio ProdutoID	Inteiro
Domínio Quantidade	Inteiro

PedidosProdutos(

Pedido	PedidoID	NOT NULL,
Produto	ProdutoID	NOT NULL,
quantidade	Quantidade	NOT NULL,

PRIMARY KEY (Pedido, Produto),
FOREIGN KEY (Pedido) REFERENCES (idPedido) ON UPDATE CASCADE ON DELETE NO ACTION,
FOREIGN KEY (Produto) REFERENCES (idProduto) ON UPDATE CASCADE ON DELETE CASCADE);

5.1.2 Desenho das representações dos dados derivados

Nesta secção é discutida a utilização dos atributos derivados. Estes são atributos cujo valor pode ser calculado através de outros atributos.

Um exemplo de um atributo cujo valor poderia ser derivado seria o "Valor" de um determinado “Pedido”. Este valor poderia ser calculado através dos preços e da quantidade de cada produto pertencente a esse “Pedido”.

Contudo, este cálculo teria de ser efetuado cada vez que se pretende efetuar a consulta do valor desse dado atributo, sendo que esta operação requer um custo mais elevado quantos mais produtos de um dado pedido estiverem associados. Assim, a opção tomada foi acrescentar um atributo na entidade Pedido denominado "Valor" que será atualizado sempre que se associar um novo produto a um dado pedido. Desta forma, dado que o custo de armazenamento deste atributo não é particularmente significativo, é possível aceder diretamente ao seu valor aumentando a performance de execução.

Foi criado um gatilho para que sempre que fosse registado um pedido o valor do pedido fosse a multiplicação do preço do produto pela quantidade, que demonstramos na imagem abaixo.

```
26  delimiter $$
27  create trigger atualizarValor
28  after insert on PedidosProdutos
29  for each row
30  begin
31  declare total decimal (7,2);
32  select (New.Quantidade*preco)into total
33      from Produtos
34      where idProduto=New.Produto;
35   update Pedidos
36   set Valor =total+valor
37   where idPedido=New.Pedido;
38  end $$
```

Figura 9 - Script da criação do gatilho "AtualizaValor"

5.1.3 Desenho das restrições gerais

Cada uma das restrições gerais têm de ser implementadas no SGBD e o pretendido agora é exatamente isso. Para cada uma das restrições de negócio será apresentada uma forma de a implementar tendo em consideração o motor escolhido. As soluções encontradas

podem ser simplesmente implementadas com SQL standard ou utilizando mecanismos específicos do SGBD como por exemplo, um trigger. A escolha da solução vai depender muito das capacidades do SGBD e caso alguma das restrições não possam ser implementado por ela, a solução será implementada pela aplicação. De seguida, cada uma das restrições irá ser apresentada e a sua respetiva solução.

Nesta fase são definidas as restrições gerais (ou regras de negócio) que servem para garantir a coerência no "mundo real" dos dados armazenados. Por exemplo, não faz sentido existir uma tarefa cuja data de início seja superior à data em que foi finalizada. Assim, é necessário impor regras que impeçam a ocorrência destes casos.

Relação Funcionário

- A idade de um funcionário deve ser igual ou superior a 18 anos e inferior a 65

```
3   delimiter $$
4   create trigger anoFuncionario
5   before insert on Funcionarios
6   for each row
7   begin
8   declare msg varchar(255);
9
10  if((datediff(now(),new.DataNascimento))/365<18)
11  Then
12   set msg= 'Não tem idade minima para Exercer funções';
13            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = msg;
14
15  elseif ((datediff(now(),new.DataNascimento))/365>65)
16  Then
17   set msg= 'EXECEDEU A IDADE PARA EXERCER FUNÇÕES';
18            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = msg;
19  end if;
20  end
21  $$
```

Figura 10 - Criação de um gatilho que permite verificar o limite de idades

- O salário de um funcionário tem de ser maior que 0 unidades monetárias.

Para esta restrição ser garantida, foi implementada na script de criação das tabelas o mecanismo CHECK (salario>0), como pode ser verificado na Figura 9.

Relação produto

-O preço de um certo produto deverá ser igual ou superior a 0,0€

-O stock existente desse produto deverá ser superior ou igual a zero

Para estas restrições serem respeitadas implementaram-se na script de criação das tabelas os mecanismos CHECK (preco > 0) e CHECK (stock >= 0), como pode ser observado na Figura 7.

Relação Pedido

- O valor do pedido terá de ser maio que 0 €.

Para estas restrições serem respeitadas implementaram-se na script de criação das tabelas os mecanismos CHECK (valor > 0)

5.2. Organização dos ficheiros e índices

5.2.1 Análise de transações

De forma a aumentar a performance da base de dados, foi necessária uma análise das transações que ocorrem entre as relações. A partir desta análise, foi possível perceber a sua funcionalidade e identificar as mais importantes. Como foi referido anteriormente, na análise de requisitos, foi elaborada uma lista com algumas das principais queries. Através da análise dessa lista, concluímos que a maior parte das delas apenas implicam operações de leitura. No entanto, também existem operações frequentes que implicam a inserção/atualização de valores nas relações. De qualquer das maneiras, optamos por criar uma script que identifica o tipo de operações que cada querie implica.

1ª Transação - Inserir pedido, associar funcionário ao pedido e adicionar produto ao pedido

```
delimiter $$
create procedure registarPedido (IN Funcionario INT, IN Produto INT, IN Quantidade INT, in Cliente int)
BEGIN
declare PedidoID int;
declare erro bool default 0;
declare continue handler for sqlexception set erro=1;
start transaction;

insert into Pedidos (idPedido,valor,dataHora, Cliente)
        values (idPedido,0,now(), Cliente);
select idPedido into PedidoID
from Pedidos
order by idPedido DESC
LIMIT 1;

insert into FuncionariosPedidos (Funcionario,Pedido)
            values (Funcionario, PedidoID);

insert into PedidosProdutos (Pedido, Produto, quantidade)
            values (PedidoID, Produto, Quantidade);
IF ERRO
THEN rollback;
ELSE COMMIT;
END IF;
END $$
```

Figura 11 – Transação 1 – Criação do procedimento “registarPedido”

<u>2ª Transação:</u> Atualização do stock de um produto aquando da confeção de um produto por um funcionário

```
delimiter $$
create procedure atualizarStockFromFunc (in Funcionario int, in Produto int, in Quantidade int)
BEGIN

declare erro bool default 0;
declare continue handler for sqlexception set erro=1;
start transaction;

Insert into FuncionariosProdutos
(Funcionario, Produto, quantidade, dataConfe)
VALUES
    (Funcionario, Produto, Quantidade, now());

    Update Produtos set stock=stock+Quantidade
    WHERE idProduto=Produto;

IF ERRO
THEN rollback;
ELSE COMMIT;
END IF;
 end $$
```

Figura 12 – Criação do procedimento “atualizarStockFromFunc”

3ª Transação – Substituir o funcionário despedido na tabela FuncionariosPedidos e eliminar o funcionário despedido na tabela Funcionarios

```
74  -- drop procedure substiuFun
75  delimiter $$
76  create procedure substiuFun (in Funcionariodes int, in Funcionariosub int)
77
78  BEGIN
79  declare erro bool default 0;
80  declare continue handler for sqlexception set erro=1;
81  start transaction;
82
83  Update FuncionariosPedidos set Funcionario=Funcionariosub
84  where Funcionario=Funcionariodes;
85
86  delete from Funcionarios
87  where idFuncionario=Funcionariodes;
88
89  IF ERRO
90  THEN rollback;
91  ELSE COMMIT;
92  END IF;
93   end $$
```

Figura 13 – Criação do procedimento “substiuFun”

5.2.2 Escolha da organização dos ficheiros

A base de dados em armazenamento secundário é organizada num ou mais ficheiros, em que cada ficheiro contém vários registos e cada registo é composto por vários campos. Por norma, a cada registo está associado uma entidade e um campo de um atributo.

Guardar e aceder a dados de uma forma eficiente, é um dos principais objetivos, no desenho do esquema físico de uma base de dados.

5.2.3 Escolha dos índices

Um índice é uma estrutura de dados que permite ao SGBD localizar registos num ficheiro mais rapidamente e, consequentemente, melhorar o tempo de resposta a queries do utilizador. Todas as tabelas InnoDB têm um indice especial chamado indice "clustered" (agrupado) onde a informação sobre os registos é guardado. Normalmente, este indice é sinónimo da chave primária da tabela.

Todos os índices InnoDB são B+Tree, onde os registos dos índices são guardados nas páginas-folha da árvore. O tamanho default de uma página é 16KB. Quando novos registos

são inseridos, o InnoDB tenta deixar 1/16 da página livre para futuras inserções e updates nos registos do índice. Se os registos do índice são inseridos de forma ordenada, as páginas resultantes estarão praticamente preenchidas. Caso as inserções sejam feitas de forma aleatório, as páginas estarão entre metade e praticamente preenchidas. Se o fator de preenchimento da página de um índice baixar para menos de metade, o InnoDB tenta contrair a árvore para libertar a página. Como foi referido, o InnoDB utiliza a chave primária de cada tabela como índice. Sendo assim, vamos apresentar alguns índices:

```
3  Create INDEX FuncionarioNib on Funcionarios (Nome);
4
5  Create Index ClienteNome on Clientes (nome);
6
7  Create Index ProdutoPreco on Produtos (nome);
8
9  Create index PedidoCliente on Pedidos (idPedido);
10
```

Figura 14 - Criação de índices

5.2.4 Estimativa de espaço em disco necessário

Como já foi referido anteriormente, utilizaremos o motor de armazenamento InnoDB do MySQL. Visto isto, precisamos de saber, o espaço necessário para cada tipo de domínio utilizado na base de dados. O InnoDB utiliza o “COMPACT Row Format” por defeito, e os tamanhos do domínio são os seguintes:

- Integer 4 bytes
- Date 3 bytes
- Varchar(N) 2+N bytes
- Decimal (7,2) 5 bytes
- Decimal (8,2) 9 bytes

Após alguns cálculos, concluímos que o espaço necessário para cada registo nas tabelas são os seguintes:

- Funcionarios 614
- FuncionariosPedidos 8
- Pedidos 17
- Produtos 63

- PedidosProdutos 13
- FuncionariosProdutos 15
- Clientes 337

Agora, de acordo com a informação da nossa base de dados o espaço necessário para cada registo numa tabela a multiplicar pelo número de registos totais:

- Funcionario 8*614 = 4912
- FuncionariosPedidos 30*8 = 240
- Pedidos 30*17 = 510
- Produtos 22*63 = 1386
- PedidosProdutos 13*29 = 377
- FuncionariosProdutos 26*15 = 390
- Clientes 100*337 = 33700

Somando todos os valores de cada tabela, estima-se que o tamanho inicial necessário para a base de dados ronde 41755 bytes (40,77Kbytes).

5.3. Desenho das vistas de utilizadores

Uma das questões importantes no desenvolvimento de um Sistema de Gestão de Base de Dados é a restrição das vistas de cada utilizador. A nossa implementação tem em conta que existirão N tipos de utilizadores:

- Administrador – Tem acesso a toda a Base de Dados e pode executar todas as ações que entender.

- Chefe de padaria – Tem acesso a toda a informação exceto os dados pessoais dos clientes.

- Funcionário – Tem os mesmos privilégios do que o chefe de secção, no entanto, com a exceção de não conhecer os fornecedores.

- Cliente – Pode consultar os produtos que estão expostos, ver as compras que lhe dizem respeito e os seus dados pessoais.

Em baixo apresentamos, para as várias vistas possíveis, o código SQL respetivo

```
5  CREATE VIEW Preçario AS
6  SELECT nome, preco AS 'Preço por unidade'
7  FROM Produtos
8  order by nome;
9  -- SELECT * FROM Preçario;
10
```

Figura 15 - Criação de uma Vista com o preçario, visivel para os clientes

```
12 -- Lista de Funcionário com respetivos horarios
13 CREATE VIEW Horarios AS
14 SELECT nome, horario
15 FROM Funcionarios;
16 -- SELECT * FROM Horarios;
17
```

Figura 16 - Criação de uma vista coma informação dos horarios dos funcionários

```
19 -- Informação dos Clientes
20 -- drop view InformacaoCliente
21 CREATE VIEW InformacaoCliente AS
22 SELECT nome AS 'Nome', nrCartaoCidadao AS 'CC', nrContribuinte as 'NIF', concelho, telemovel
23 FROM Clientes;
24 -- Select * from InformacaoCliente
```

Figura 17 - Criação de uma vista contendo a informação dos clientes

```
28 -- Compras dos clientes
29 -- drop view compraCliente
30 create view compraCliente as
31 select distinct idCliente, c.nome as 'Nome Cliente', pp.quantidade, p.nome as 'Nome Produto',
32 p.preco, pe.idPedido as 'Pedido'
33 from Clientes c  inner join Pedidos pe on pe.Cliente=c.idCliente
34                 inner join PedidosProdutos pp on pp.Pedido=pe.idPedido
35                 inner join Produtos p on pp.Produto=p.idProduto
36                 Order by pe.idPedido;
37 -- Select * from compraCliente;
```

Figura 18 - Criação de uma vista contendo a informação da relação cliente compra

5.4. Mecanismos de segurança

Nesta fase temos o objetivo de mostrar os mecanismos de segurança especificados pelos utilizadores durante a definição dos requisitos.

Na análise de requisitos definimos a existência de três tipos de utilizadores: Funcionario, Administradores e Clientes estes com a mesma vista mas com permissões diferentes.

Funcionários

Tem permissões de consulta e inserção para todas as tabelas, mas este utilizador não pode remover ou alterar nenhuma entrada de qualquer tabela.

Permissões para Funcionario:

- Necessárias para consulta

GRANT SELECT ON Pedidos TO Funcionario

GRANT SELECT ON Produtos TO Funcionario

GRANT SELECT ON Clientes TO Funcionario

GRANT SELECT ON Funcionarios TO Funcionario

- Necessárias para inserção

GRANT INSERT ON Pedidos TO Funcionario

GRANT INSERT ON PedidosProdutos TO Funcionario

GRANT INSERT ON Clientes TO Funcionario

- Necessárias para não permitir alterações

DENY UPDATE ON Produtos TO Funcionario

DENY UPDATE ON Clientes TO Funcionario

DENY UPDATE ON Funcionarios TO Funcionario

-Necessárias para não permitir remoções

DENY DELETE ON Clientes TO Funcionario

DENY DELETE ON Produtos TO Funcionario

DENY DELETE ON Funcionarios TO Funcionario

Administrador

Este tipo de utilizador deve ter permissões para realizar todas as tarefas sobre a base de dados.

Permissões para Administrador

- Necessárias para poder realizar qualquer comando

GRANT ALL ON Funcionarios TO Administrador

GRANT ALL ON FuncionariosPedidos TO Administrador

GRANT ALL ON Pedidos TO Administrador

GRANT ALL ON PedidosProdutos TO Administrador

GRANT ALL ON Produtos TO Administrador

```
GRANT ALL ON Clientes TO Administrador
GRANT ALL ON FuncionariosProdutos TO Administrador
```

5.5. Consideração de introdução de redundância controlada

A introdução de redundância controlada apenas deve ser considerada quando se prevê que a Base de Dados criada não cumprirá os requisitos de performance pretendidos, dado que esta contribui para o surgimento de dados inconsistentes. Quando se pretende evitar o processamento simultâneo de várias relações, manter um qualquer sistema de arquivo ou manter as relações a um nível (simples) que permita satisfazer as “queries” dos seus utilizadores, desnormalizar o sistema em causa pode ser uma opção viável. Apesar de a desnormalização contribuir positivamente para o desempenho das operações de pesquisa, acontece o contrário relativamente às operações de atualização. Deste modo, poderá fazer sentido considerar a realização deste processo para relações cuja frequência de atualizações seja pequena e bastante menor do que a frequência de pesquisas efetuadas. Neste sistema optou-se por não introduzir nenhum tipo de redundância.

5.6. Monitorizar e afinar o sistema operativo

Um dos principais objetivos de uma base de dados é guardar e aceder à informação de uma forma eficiente. Há portanto, uma série de fatores a ter em conta, nomeadamente, o espaço de armazenamento, o tempo de resposta do sistema ou o número de transações que se consegue processar num certo intervalo de tempo. Ao fazer com que uma Base de Dados seja eficiente, é necessário conjugar todos estes fatores de tal modo que se atinja o melhor equilíbrio possível entre os mesmos, ainda que tal implique priorizar uns em detrimento dos outros. Desta forma, afinar um sistema operativo, é uma atividade que nunca se completa. Existe, ao longo da vida do sistema, a necessidade de monitorizar o seu desempenho, de forma a explicar as mudanças no ambiente, e as necessidades dos utilizadores. No entanto, uma alteração numa área de um sistema para melhorar o sistema pode piorar o desempenho noutra área.

6. Ferramentas utilizadas

Para a elaboração deste documento foram utilizadas várias ferramentas:

- Microsoft Word 2010: Elaboração deste documento.
- brModelo: Desenho do diagrama ER e do modelo conceptual.
- MySQL Workbench: Desenvolvimento do modelo lógico e físico.
- Visual Paradigm 12.2: Desenho de mapa transacional.
- Microsoft SQL Server - Sistema de gestão de base de dados relacional

7. Conclusões e Trabalho Futuro

O desenvolvimento de uma base de dados é um processo complicado e demorado e portanto, sendo feito de forma incorreta pode provocar elevados custos a uma organização. Na tentativa de diminuir possíveis erros, deve ser seguida uma metodologia de forma a sistematizar todo o processo de desenvolvimento.

Apesar dos vários pontos positivos da utilização de uma metodologia, o processo de modelação torna-se bastante intensivo e demorado, podendo então, ser desnecessário para casos simples. Já para problemas complexos e com vários utilizadores, a sua utilização é fundamental para que se possa assegurar que o sistema desenvolvido é o pedido pela organização.

A metodologia sugerida pelo livro Database Systems foi seguida à risca na tentativa de que o resultado satisfizesse com sucesso as necessidades da *BelaPadaria*. Durante a primeira etapa de desenvolvimento, os futuros utilizadores da base de dados da *BelaPadaria* foram consultados para que cada passo fosse validado e portanto considerado de acordo com os requisitos.

De seguida implementamos o modelo lógico e o esquema físico da base de dados e um conjunto de testes, que serviu como ponto de partida para iniciar o funcionamento do sistema. Futuramente também se pretende acompanhar a utilização da base de dados por parte da *BelaPadaria,* onde se poderá intervir sempre que necessário, fazendo a manutenção do sistema.

No entanto, qualquer Base de Dados necessita de manutenção regular, de modo a garantir que funcione bem e sem anomalias ao longo do tempo, caso isso não seja satisfeito, será diminuída lentamente a sua performance, deixando de ser funcional e, tornando-se portanto, inútil.

Por fim, achamos desta forma que conseguimos cumprir os objetivos definidos no início do projeto.

Bibliografia

Thomas M. Connolly, Carolyn E. Begg - Database Systems: A Practical Approach to Design, Implementation and Management - 4th Edition.

Lista de Siglas e Acrónimos

BD Base de Dados

ER Entidade-Relacionamento

SGBD Sistema de Gestão de Base de Dados

Anexos

I. Anexo 1 – Script completo da criação do esquema

```
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';

-- -----------------------------------------------------
-- Schema belapadaria
-- -----------------------------------------------------
drop Schema if exists belapadaria;
-- -----------------------------------------------------
-- Schema belapadaria
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `belapadaria` DEFAULT CHARACTER SET utf8 ;
USE `belapadaria` ;

-- -----------------------------------------------------
-- Table `belapadaria`.`Funcionarios`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `belapadaria`.`Funcionarios` (
  `idFuncionario` INT NOT NULL AUTO_INCREMENT,
  `horario` INT NOT NULL,
  `nib` DECIMAL(22) NOT NULL UNIQUE,
  `salario` DECIMAL(8,2) NOT NULL CHECK (salario>0),
  `nome` VARCHAR(75) NOT NULL,
  `nrContribuinte` INT(9) NOT NULL UNIQUE,
  `dataNascimento` DATE NOT NULL,
  `nrCartaoCidadao` INT(8) NOT NULL UNIQUE,
  `nrPorta` VARCHAR(10) NULL,
```

```
  `rua` VARCHAR(75) NULL,
  `freguesia` VARCHAR(75) NOT NULL,
  `concelho` VARCHAR(75) NOT NULL,
  `distrito` VARCHAR(75) NOT NULL,
  `codPostal` VARCHAR(8) NOT NULL,
  `telemovel` INT(9) NOT NULL,
  `telefone` INT(9) NULL,
  `facebook` VARCHAR(75) NULL UNIQUE,
  `email` VARCHAR(75) NULL UNIQUE,
  `Funcionario` INT NULL,
  PRIMARY KEY (`idFuncionario`),
  INDEX `fk_Funcionarios_Funcionarios_idx` (`Funcionario` ASC),
  CONSTRAINT `fk_Funcionarios_Funcionarios`
    FOREIGN KEY (`Funcionario`)
    REFERENCES `belapadaria`.`Funcionarios` (`idFuncionario`)
    ON UPDATE CASCADE
    ON DELETE SET NULL )
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `belapadaria`.`Clientes`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `belapadaria`.`Clientes` (
  `idCliente` INT NOT NULL AUTO_INCREMENT ,
  `nrCartaoCidadao` INT(8) NULL UNIQUE,
  `nome` VARCHAR(75) NULL,
  `nrContribuinte` INT(9) NULL UNIQUE,
  `dataNascimento` DATE NULL,
  `codPostal` VARCHAR(8) NULL,
  `rua` VARCHAR(75) NULL,
  `concelho` VARCHAR(75) NULL,
  `telemovel` INT(9) NULL,
  `email` VARCHAR(75) NULL UNIQUE,
  PRIMARY KEY (`idCliente`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `belapadaria`.`Pedidos`
```

```
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `belapadaria`.`Pedidos` (
  `idPedido` INT NOT NULL AUTO_INCREMENT ,
  `valor` DECIMAL(7,2) NOT NULL CHECK (valor>0),
  `dataHora` DATETIME NOT NULL,
  `Cliente` INT NOT NULL,
  PRIMARY KEY (`idPedido`),
  INDEX `fk_Pedidos_Clientes1_idx` (`Cliente` ASC),
  CONSTRAINT `fk_Pedidos_Clientes1`
    FOREIGN KEY (`Cliente`)
    REFERENCES `belapadaria`.`Clientes` (`idCliente`)
    ON UPDATE CASCADE
    ON DELETE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `belapadaria`.`Produtos`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `belapadaria`.`Produtos` (
  `idProduto` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(75) NOT NULL,
  `preco` DECIMAL(7,2) NOT NULL CHECK (preco>0),
  `duracao` TIME NULL,
  `stock` INT(10) NOT NULL CHECK (stock>=0),
  PRIMARY KEY (`idProduto`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `belapadaria`.`FuncionariosPedidos`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `belapadaria`.`FuncionariosPedidos` (
  `Funcionario` INT NOT NULL,
  `Pedido` INT NOT NULL,
  PRIMARY KEY (`Funcionario`, `Pedido`),
  INDEX `fk_Funcionarios_has_Pedidos_Pedidos1_idx` (`Pedido` ASC),
  INDEX `fk_Funcionarios_has_Pedidos_Funcionarios1_idx` (`Funcionario` ASC),
  CONSTRAINT `fk_Funcionarios_has_Pedidos_Funcionarios1`
    FOREIGN KEY (`Funcionario`)
```

```
    REFERENCES `belapadaria`.`Funcionarios` (`idFuncionario`)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
  CONSTRAINT `fk_Funcionarios_has_Pedidos_Pedidos1`
    FOREIGN KEY (`Pedido`)
    REFERENCES `belapadaria`.`Pedidos` (`idPedido`)
    ON UPDATE CASCADE
    ON DELETE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `belapadaria`.`PedidosProdutos`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `belapadaria`.`PedidosProdutos` (
  `Pedido` INT NOT NULL,
  `Produto` INT NOT NULL,
  `quantidade` INT(7) NOT NULL,
  PRIMARY KEY (`Pedido`, `Produto`),
  INDEX `fk_Pedidos_has_Produtos_Produtos1_idx` (`Produto` ASC),
  INDEX `fk_Pedidos_has_Produtos_Pedidos1_idx` (`Pedido` ASC),
  CONSTRAINT `fk_Pedidos_has_Produtos_Pedidos1`
    FOREIGN KEY (`Pedido`)
    REFERENCES `belapadaria`.`Pedidos` (`idPedido`)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
  CONSTRAINT `fk_Pedidos_has_Produtos_Produtos1`
    FOREIGN KEY (`Produto`)
    REFERENCES `belapadaria`.`Produtos` (`idProduto`)
    ON UPDATE CASCADE
    ON DELETE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `belapadaria`.`ClientesProdutos`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `belapadaria`.`ClientesProdutos` (
  `Cliente` INT NOT NULL,
  `Produto` INT NOT NULL,
```

```
  PRIMARY KEY (`Cliente`, `Produto`),
  INDEX `fk_Clientes_has_Produtos_Produtos1_idx` (`Produto` ASC),
  INDEX `fk_Clientes_has_Produtos_Clientes1_idx` (`Cliente` ASC),
  CONSTRAINT `fk_Clientes_has_Produtos_Clientes1`
    FOREIGN KEY (`Cliente`)
    REFERENCES `belapadaria`.`Clientes` (`idCliente`)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
  CONSTRAINT `fk_Clientes_has_Produtos_Produtos1`
    FOREIGN KEY (`Produto`)
    REFERENCES `belapadaria`.`Produtos` (`idProduto`)
          ON UPDATE CASCADE
    ON DELETE CASCADE)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `belapadaria`.`FuncionariosProdutos`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `belapadaria`.`FuncionariosProdutos` (
  `Funcionario` INT NOT NULL,
  `Produto` INT NOT NULL,
  `quantidade` INT NOT NULL,
  `dataConfe` DATE NULL,
  PRIMARY KEY (`Funcionario`, `Produto`),
  INDEX `fk_Funcionarios_has_Produtos_Produtos2_idx` (`Produto` ASC),
  INDEX `fk_Funcionarios_has_Produtos_Funcionarios2_idx` (`Funcionario` ASC),
  CONSTRAINT `fk_Funcionarios_has_Produtos_Funcionarios2`
    FOREIGN KEY (`Funcionario`)
    REFERENCES `belapadaria`.`Funcionarios` (`idFuncionario`)
    ON UPDATE CASCADE
    ON DELETE CASCADE,
  CONSTRAINT `fk_Funcionarios_has_Produtos_Produtos2`
    FOREIGN KEY (`Produto`)
    REFERENCES `belapadaria`.`Produtos` (`idProduto`)
    ON UPDATE CASCADE
    ON DELETE CASCADE)
ENGINE = InnoDB;
```

```
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

II. Anexo – Script completo do Povoamento da 'BelaPadaria'

```
-- Universidade do Minho
-- Mestrado Integrado em Engenharia Informática
-- Unidade Curricular de Bases de Dados
-- 2015/2016

-- drop database belapadaria;

-- DELETE FROM funcionarios;

USE `belapadaria` ;

-- SET SQL_SAFE_UPDATES = 0;

-- DELETE FROM produtos;
INSERT INTO produtos
(idProduto, nome, preco, duracao, stock)
VALUES
(1, 'pão de milho', 3.2 , 1000, 200),
(idProduto,'bijou',0.2,1500,600),
(idProduto, 'cacete', 0.80, 1800, 630),
(idProduto, 'broa de centeio', 2.5, 1000, 310),
(idProduto, 'baguete',0.40, 1800, 500),
(idProduto, 'trigo', 0.10, 950, 1000),
(idProduto, 'pao de forma', 1.20, 1321, 100),
(idProduto, 'pão preto', 2.85, 2000, 150),
(idProduto, 'pao sirio',4.0, 1000, 320),
(idProduto, 'pão mistura', 2.4, 1800, 125),
(idProduto, 'croissants', 0.75, 2000, 46 ),
(idProduto, 'bolas de berlim',1.10, 1256, 65),
(idProduto, 'muffin', 0.7, 1900, 23),
```

```
(idProduto, 'bolo de arroz', 0.5, 1327, 13),
(idProduto, 'bolo brigadeiro', 1.20, 1658, 46),
(idProduto, 'panike', 0.8, 623, 100),
(idProduto, 'pão de ló de Ovar', 10.00, 2500, 50 ),
(idProduto, 'bolas integrais', 0.85, 2653, 210),
(idProduto, 'pão com chouriço', 1.00, 1856, 65),
(idProduto, 'pastel de nata', 1.00, 2100, 150),
(idProduto, 'tarte de morango', 6.00, 3215, 20),
(idProduto, 'tarte de amendoa', 7.00, 2500, 15);
-- SELECT * FROM produtos;


-- DELETE FROM clientes;
INSERT INTO clientes
(idCliente,nrCartaoCidadao,nome,
nrContribuinte,dataNascimento,codPostal,rua,concelho,telemovel,email)
VALUES
(1,14344154,'Célia Figueiredo', 262646080, '1994-05-21', '4755-128', 'Rua da Costa',
'Barcelos',933337717,'celianlf@hotmail.com'),
(idCliente,14256546,'Silvia Figueiredo', 262646170, '1990-08-01', '4755-128', 'Rua da Coutada',
'Barcelos',934885969,'silvialfg@hotmail.com'),
(idCliente,14513924,'Daniel Rodrigues', 243519145, '1994-01-31', '4830-274', 'Rua dos
Condes', 'Povoa de Lanhoso',945434241,'danysuica@hotmail.com'),
(idCliente,14628648,'Patricia Barros', 252636450, '1994-02-24', '4970-129', 'Rua da Cachada',
'Arcos de Valdevez',934947881,'padb7@hotmail.com'),
(idCliente,14646469,'Márcia Costa', 246464646, '1994-01-21', '4710-416', 'Rua dos
Peões','Braga',933337681,'mmcostinha@gmail.com'),
(idCliente,14425150,'Xavier Francisco', 258903384, '1994-11-10', '2495-183', 'Rua da Central',
'Leiria',912845563,'xavier.n.francisco@gmail.com'),
(idCliente,14256545,'Maria Linda', 562345089, '1989-12-25', '4755-117', 'Rua da Quintão',
'Barcelos',919568697, 'marialinda@hotmail.com'),
(idCliente,13142598,'Pedro Miguel', 232524089, '1994-06-06', '4890-123', 'Rua da Cortina',
'Guimarães',936575075,'celianlfg@hotmail.com'),
(idCliente, 65752855, 'Acacio Reino', 236896573, '1990-06-01','4789-456', 'Rua das Camélias',
'Braga', 936545623, 'acacioreino@gmail.com'),
(idCliente,41988997, 'Afonso Felipe', 255590195, '1978-06-16', '4659-489','Rua Doutor
Fernando Santos', 'Braga', 919636489, 'afonsofilipe@hotmail.com'),
(idCliente,66813274, 'Aida Figueira', 234117754 , '1950-01-31', '4632-488','Rua Martins Sá',
'Guimarães', 969596989, 'aidafigueira@gmail.com'),
```

(idCliente, 21166885,'Alzira Branco', 212100348, '1965-02-05','4698-420', 'Rua das Manteigas', 'Povoa de Lanhoso', 919596638, 'alzbranco@gmail.com'),
(idCliente, 59289948, 'Amanda Collaço', 248562163, '1978-02-04', '4963-023', 'Rua do Areal', 'Braga', 936569638, 'amanda@gmail.com'),
(idCliente, 82069345, 'Araribóia Lages', 230799642, '1989-02-06','4652-365','Rua Principal', 'Ponte da Barca', 936598045, 'lages@gmail.com'),
(idCliente, 87303557 , 'Aurélio Alencar', 216271968, '1979-02-07', '4569-123', 'Rua Santa Clara', 'Braga', 919895941, 'aurelioalencar@hotmail.com'),
(idCliente, 77850320 , 'Barnabé Maia', 258732863 , '1965-05-05', '4658-046', 'Rua Carris', 'Ponte da Barca', 910023659, 'barnabe@gmail.com'),
(idCliente, 87234908, 'Camilo Castelo Branco', 254983529, '1957-09-06', '4986-056', 'Rua da Costa ', 'Ponte de Lima', 929695963, 'camilo@gmial.com'),
(idCliente, 75490443, 'Carmem Eiró', 211043896, '1968-09-06', '4755-118', 'Rua do computador', 'Ponte de Lima', 919796364, 'carmeneiro@gmail.com'),
(idCliente, 14906449, 'Cecília Moita', 225505091,'1945-06-06', '4755-136', 'Rua dos cornudos', 'Braga', 919895032, 'cecilia@gmial.com'),
(idCliente, 73530459,'Celso Nieves', 238394802, '1978-07-07', '4769-123', 'Rua de Remelhe', 'Braga', 963126799, 'celsoneves@gmail.com'),
(idCliente, 55905558, 'Cecília Amorín', 228393850, '1946-05-05', '4754-145', 'Rua dos Olivais', 'Braga', 926364501, 'ceciliamoites@gmail.com'),
(idCliente, 70364642, 'Caím Barateiro', 245843773, '1976-05-06', '4659-145', 'Rua do Ramalhete', 'Vila Verde', 925468456, 'caimbarateiro@gmail.com'),
(idCliente, 59193671, 'Brenda Eiró', 216255827,'1978-09-25','4758-118', 'Rua da Angola', 'Vila Verde', 923659874, 'brendaeiro@gmail.com'),
(idCliente, 17604812,'Berengária Capistrano', 244020132, '1980-03-26', '4632-089', 'Rua dos Congregados', 'Braga', 919362032, 'capistrano@gmail.com'),
(idCliente, 77563243 ,'Cleiton Dâmaso', 230040906,'1946-04-06','4789-063', 'Rua dos pimentos', 'Braga', 919666320, 'cleitondamaso@gmail.com'),
(idCliente, 59263645, 'Constança Bezerra', 219881747, '1989-09-09', '4569-230', 'Rua dos livros', 'Braga', 919632555, 'constanca@gmail.com'),
(idCliente, 64891283 , 'Cristiana Camelo', 249082277 , '1955-03-01', '4980-053', 'Rua da Fonte', 'Arcos de Valdevez', 963222032, 'cristianocamelo@gmail.com'),
(idCliente, 42690311,'Cândido Cordero', 225553173, '1946-06-04', '4986-012', 'Rua da Fiona', 'Braga', 912333650, 'cordero@gmail.com'),
(idCliente, 38243651, 'Dinis Silveira dos Açores', 243459649 , '1963-06-06','4695-569', 'Travessa Cruz da Argola','Braga', 912365445, 'dinisacores@gmail.com'),
(idCliente, 48779571, 'Dorindo Bautista', 235346313, '1989-12-12', '4610-050', 'Rua do Montinho' , 'Guimarães', 932336520, 'dorindo@gmail.com'),
(idCliente, 21458549, 'Else Costa', 224673769, '1981-07-26', '4810-050', 'Rua José Faria Martins', 'Guimarães', 938971033, 'elsetintascompinta@hotmail.com'),

(idCliente, 63302315, 'Délio Valadares', 259684559, '1966-11-11', '4695-023', 'Rua Adamastor', 'Guimarães', 963126755, 'deliovaladares@gmail.com'),
(idCliente, 24630583 , 'Enia Godinho', 240731124, '1967-12-15', '4932-563', 'Rua dos lafões', 'Vila Nova de Famalicão', 936333014, 'eniagod@hotmail.com'),
(idCliente, 58402102,'Fausto Bezerril', 245911111 , '1965-05-24', '4965-365', 'Rua das Camélias', 'Braga', 931125563, 'faustobezerril@gmail.com'),
(idCliente, 29678505, 'Felisbela Barroqueiro0', 251677164, '1945-03-03', '4765-012', 'Travessa da igreja', 'Braga', 923666320, 'felisbelabar@gmail.com'),
(idCliente, 20737619 ,'Fernão Mesquita', 258487592 , '1940-09-15', '4978-026', 'Travessa dos mosqueteiros', 'Braga', 936363203, 'fernaomesq@gmail.com'),
(idCliente, 54660207,'Filipa Moita', 231508952 , '1949-03-06', '4965-001', 'Rua Camões', 'Braga', 933331236, 'filipamota@gmail.com'),
(idCliente, 61590065 , 'Filomena Vicario', 228505066, '1976-10-05', '4755-129', 'Rua D.João IV', 'Porto', 919998632, 'filomenavica@gmail.com'),
(idCliente, 37447762 , 'Fulvio Álvarez', 212368844 , '1954-04-29', '4321-603', 'Rua Padre Henrique', 'Porto', 936569874, 'fulviolavarez@gmail.com'),
(idCliente, 74539809 , 'Fátima Quinzeiro', 237361232, '1973-10-23', '4895-236', 'Rua das Marias', 'Porto', 964456320, 'fatimaquinz@gmail.com'),
(idCliente, 31543274, 'Gedeão Piteira', 223094652, '1968-08-08', '4769-961', 'Rua Milionário', 'Porto ', 912223650, 'gedeaopiteira@gmail.com'),
(idCliente, 53034102, 'Genoveva Ataíde', 215529801, '1978-09-06', '4956-025', 'Rua da Luz', 'Porto', 926565320, 'genovevaataide@gmail.com'),
(idCliente, 61953014 , 'Gertrudes Serralheiro', 226703667, '1975-06-03', '4563-029', 'Rua dos pneus', 'Porto', 912236603, 'gertrudesserral@gmail.com'),
(idCliente, 48819960 , 'Gilberto Aragão', 248185990, '1989-07-08', '4569-147', 'Rua Gil Eanes', 'Porto', 932635632, 'gilbertoaragao@gmail.com'),
(idCliente, 75892799, 'Gilberto Hidalgo', 249343033, '1959-09-28', '4956-110', 'Rua Vasco da Gama', 'Aveiro', 963332145, 'gilbertohidalgo@gmail.com'),
(idCliente, 28212437, 'Graça Lameiras', 249680555, '1978-10-11', '4112-456', 'Praceta Goa', 'Aveiro', 912366658, 'gracalameira@gmail.com'),
(idCliente, 40421771, 'Guaraci Henriques', 221772118, '1957-12-10','4756-012', 'Rua Guiné Bissau', 'Aveiro', 933365569, 'guaracihenr@gmail.com'),
(idCliente, 10258345, 'Guterre Nóbrega', 231098939, '1978-10-21', '3800-510', 'Rua da Liberdade','Aveiro', 912256302, 'guterresno@hotmail.com'),
(idCliente, 85644722, 'Heloísa Barros', 243678919, '1981-02-24', '3845-520', 'Rua 1 de Maio', 'Póvoa de Varzim', 936632014, 'heloisabarros@gmail.com'),
(idCliente, 62789072, 'Herculano Rosário', 26634814, '1967-06-08', '3846-456', 'Rua de milho', 'Póvoa de Varzim', 912256301, 'herculanorosa@gmail.com'),
(idCliente, 92883492 , 'Heriberto Vega', 247026870 , '1979-05-26', '3465-489', 'Rua dos Combatentes', 'Fafe', 919898638, 'heribertovega@hotmail.com'),

(idCliente, 28459026 , 'Irene Cayubi', 256196917, '1970-05-13', '3956-102', 'Rua das Flores', 'Aveiro', 912365698, 'irenecay@gmail.com'),
(idCliente, 31133069, 'Isaura Fraga', 216794139 , '1971-09-29', '3789-036', 'Rua dos Canteiros da Igreja', 'Porto', 936632220, 'isaurafraga@gmail.com'),
(idCliente, 55387861 , 'Isilda Louzada', 220006450 , '1978-08-25', '4698-456','Rua das Camélias','Braga', 915569874, 'isildalos@hotmail.com'),
(idCliente, 28066071, 'Jandira Fuentes', 238509303, '1979-09-06', '4956-113', 'Rua dos Campeões', 'Porto', 936523652, 'jandirafuentes@gmail.com'),
(idCliente, 86288833, 'Jonas Saltão', 214500324, '1972-10-30', '4980-005', 'Ruas das Catarinas', 'Porto', 963166320, 'jonasaltao@gmail.com'),
(idCliente, 29896435, 'João Murtinho', 214754691, '1979-10-10', '4896-123', 'Rua dos mosqueteiros', 'Porto', 936565230, 'joaomurtinho@gmial.com'),
(idCliente, 32089958, 'Jéssica Abasto', 230792216, '1980-10-06', '4979-125', 'Rua dos aquecedores', 'Porto', 912223547, 'jessicaabasto@gmail.com'),
(idCliente, 53165146, 'Levi Caiapó', 210508385, '1958-10-13', '4700-125', 'Rua dos sofás', 'Porto', 936656320, 'levicaia@gmail.com'),
(idCliente, 86804958, 'Lina Bulhosa', 214973412, '1973-12-05', '4752-478', 'Rua da Boavista', 'Porto', 919696320, 'linabulhosa@gmial.com'),
(idCliente, 97588097, 'Luzia Barretto', 219322680, '1986-12-03', '4563-333', 'Rua da Silvia', 'Braga', 929696301, 'luziabarreto@gmail.com'),
(idCliente, 49686083, 'Luís Lamego', 209103457, '1962-01-03', '4963-160', 'Rua das televisões', 'Braga', 919596989, 'luislamego@hotmial.com'),
(idCliente, 57264248 , 'Léia Godoi', 218642117, '1974-05-27', '4698-456', 'Rua dos patetas', 'Porto', 919696324, 'leiagodi@gmail.com'),
(idCliente, 41363446, 'Lívia Beserril', 244078522, '1971-03-26', '4698-114', 'Rua Antonio Campos', 'Braga', 933365447, 'liviabase@gmail.com'),
(idCliente, 26081466, 'Marcelo Colaço', 230013263, '1995-10-06', '4750-63', 'Rua Martins de Sá', 'Barcelos', 934885620, 'marceloscolaco@gmail.com'),
(idCliente, 50091477, 'Marco Assis', 234882607, '1990-01-06', '4796-118' ,'Rua do Marcelo Marcelino', 'Barcelos', 936654102, 'marcoassis@hotmail.com'),
(idCliente, 77434812,'Moisés Mourato', 250448213, '1976-06-28', '4950-005', 'Rua das candeias', 'Braga', 919633210, 'moisesmourato@gmail.com'),
(idCliente, 54203565, 'Moisés Paião', 258916186, '1959-12-26', '4520-220', 'Rua das sapatilhas', 'Braga', 919987441, 'moisespaiao@gmail.com'),
(idCliente, 14993103, 'Morgana Nieto', 258979151, '1979-06-28', '4693-009', 'Rua dos paus', 'Porto', 919952012, 'morgananieto@hotmail.com'),
(idCliente, 42325186, 'Máxima Gallindo', 210239403, '1989-09-09', '3900-145', 'Rua dos pessegos', 'Braga', 936523023, 'maximagall@gmail.com'),
(idCliente, 83964441, 'Nicolau Pessoa', 256107368, '1976-05-10', '3956-056', 'Rua dos palhaços', 'Porto', 966654230, 'nicolaupessoa@hotmail.com'),

(idCliente, 72563775, 'Noé Amaro', 239787750, '1970-12-31', '3900-009', 'Rua das Portas Partidas', 'Porto', 912223658, 'noemaoro@gamil.com'),
(idCliente, 84869974, 'Noé Matoso', 222031717, '1976-06-09', '3795-026', 'Rua Santa Bárbara', 'Porto', 918856320, 'noematoso@hotmail.com'),
(idCliente, 64530325 , 'Olavo Regalado', 249041833, '1988-10-12', '4006-026', 'Avenida da Liberdade', 'Braga', 936666523, 'olavoregalo@hotmail.com'),
(idCliente, 97987526, 'Ondina Fonseca', 230863795, '1989-10-01', '4700-060', 'Rua das Canetas', 'Porto', 919192936, 'ondinafonseca@gmail.com'),
(idCliente, 82403162, 'Parcidio Santos', 210384949, '1987-07-08', '3926-056', 'Rua dos Abades', 'Braga', 933365478, 'parcisiosantos@gmail.com'),
(idCliente, 80122799, 'Plínio Souto', 247125019, '1975-05-25', '4900-118', 'Rua do Bairro Frio', 'Barcelos', 919666547, 'pliniosouto@gmail.com'),
(idCliente, 22068561,'Raul Abranches', 255587146, '1980-05-29', '4962-089', 'Rua do cão lindo', 'Porto', 929632301, 'raulabranches@gmail.com'),
(idCliente, 19955704, 'Ricardo Abreu', 258299270, '1976-10-21', '4760-220', 'Rua dos meninos lindos', 'Porto', 919693201, 'ricardoabreu@gmail.com'),
(idCliente, 49415844, 'Rita Granjeiro', 217637687, '1980-01-28', '3500-110', 'Rua do Sol do cão', 'Porto', 919654487, 'ritagrangeiro@gmail.com'),
(idCliente, 67642994, 'Ronaldo Neto', 225708936, '1993-06-29', '4750-042', 'Rua do Sado', 'Aveiro', 939669321, 'ronaldoneto@gmail.com'),
(idCliente, 28876313, 'Rubim Lopes', 252634214, '1990.09-26', '4569-117', 'Rua do Lobarinhas', 'Barcelos', 939632114, 'rubimlopes@gmail.com'),
(idCliente, 32385660, 'Selma Rico', 245116940, '1954-10-01', '4750-154', 'Rua do Hospital', 'Barcelos', 929632145, 'selmarico@gmail.com'),
(idCliente, 30025843, 'Simeão Paula', 237751261, '1988-08-28', '4755-125', 'Rua da Julia Pinheiro', 'Braga', 919995874, 'simeaopaula@gmail.com'),
(idCliente, 82040713, 'Sofia Meneses', 211857717, '1990-10-18', '3923-256', 'Rua das Carteiras', 'Barcelos', 929696325, 'sofiameneses@gmail.com'),
(idCliente,69715916,'Júlio Figos',239580157,'2002-12-11','4835-052','Rua das Violetas','Guimarães',969654555,'jujufigos@hotmail.com'),
(idCliente,28778883,'Amália Costa',223050562,'1989-10-10','3800-032','Rua do Rei','Viana do Castelo',939564525,'amaliacostinha@yahoo.com'),
(idCliente,76460503,'Henrique Alburqueque',252339592, '1979-11-26', '3455-489', 'Rua dos Combatentes', 'Fafe', 912211662, 'heriqueque@hotmail.com'),
(idCliente,54808239,'Susana Freitas',228058672,'1990-02-18','3820-416','Rua Mousinho da Silveira','Aveiro',939998452,'sufrei@yahoo.com'),
(idCliente,12723975,'Cláudia Mendes',245217670,'2000-02-03','4035-025','Rua das Amoreiras','Lisboa',912103254,'mendeslau9@hotmail.com'),
(idCliente,72520958,'Joel Freitas',220720397,'1992-06-01','4800-050','Rua do Rossio','Lisboa',969654148,'jufreitas@gmail.com'),

(idCliente,35078862,'Celina Melo',241573795,'1990-03-02','4810-023','Rua 25 de Abril','Porto',966659032,'melo11@gmail.com'),
(idCliente,50007458,'Eduardo Almeida',232152526,'1999-09-12','3835-023','Rua Central 3','Fafe',939365472,'edu_al_meida@gmail.com'),
(idCliente,24038347,'Ana João Castro',246336500,'1994-06-15','4835-005','Rua 1 de Janeiro','Braga',918542013,'castro_ana34@hotmail.com'),
(idCliente,97767537,'Zita Cerqueira', 217587087, '2000-01-01','3805-155','Avenida da Guerra','Porto',939352012,'zzitac_5@gmail.com'),
(idCliente,72682483, 'Álvaro Passos', 217484621 , '1988-09-13', '4800-212', 'Rua Soares dos Santos ', 'Famalicão', 912296364, 'passos88@hotmail.com'),
(idCliente,72682480, 'Manuel Mendes', 249807888 , '1998-09-13', '4850-211', 'Rua Sol do Ave ', 'Fafe', 932296304, 'manuelfmendes8@hotmail.com'),
(idCliente,67939759, 'Ártur Fonseca', 240020325 , '1978-10-13', '4610-561', 'Rua Soares Abreu ', 'Felgueiras', 966296364, 'arturfonfon@hotmail.com'),
(idCliente, 66097075, 'Vitor Ferreira', 232536798, '1992-03-03', '4800-321', 'Rua dos Santos Populares ', 'Famalicão', 922206364, 'ferreira65@hotmail.com'),
(idCliente, 23656282, 'Diogo Costa', 252918603, '1984-08-11', '4710-417', 'Rua Soares dos Santos ', 'Braga', 912246364, 'dicosta@hotmail.com');

-- SELECT * FROM clientes;

-- DELETE FROM pedidos
INSERT INTO pedidos
(idPedido, valor, dataHora, Cliente)
VALUES
(1, 0, '2015-10-01 10:00:00', 1),
(idPedido, 0, '2015-11-03 11:01:23', 2),
(idPedido, 0, '2015-12-03 12:12:53', 3),
(idPedido, 0, '2015-12-10 14:10:05', 4),
(idPedido, 0, '2015-12-20 15:21:23', 50),
(idPedido, 0, '2015-12-21 15:25:00', 5),
(idPedido, 0, '2015-12-23 16:20:56', 6),
(idPedido, 0, '2015-12-24 16:20:47', 7),
(idPedido, 0, '2015-12-24 17:05:56', 8),
(idPedido, 0, '2015-12-24 17:09:53', 9),
(idPedido, 0, '2015-12-24 18:01:02', 10),
(idPedido, 0, '2015-12-24 18:04:36', 12),
(idPedido, 0, '2015-12-24 18:08:39', 13),
(idPedido, 0, '2015-12-24 18:15:49',14),
(idPedido, 0, '2015-12-24 18:20:50',15),

```
(idPedido, 0, '2015-12-24 18:25:46',16),
(idPedido, 0, '2015-12-24 18:30:32',17),
(idPedido, 0, '2015-12-24 18:45:46',18),
(idPedido, 0, '2015-12-24 18:50:54',19),
(idPedido, 0, '2015-12-26 09:08:03',20),
(idPedido, 0, '2015-12-26 09:20:06',21),
(idPedido, 0, '2015-12-26 10:21:09',53),
(idPedido, 0, '2015-12-26 15:15:15',22),
(idPedido, 0, '2015-12-26 18:25:39',23),
(idPedido, 0, '2015-12-27 15:46:48',24),
(idPedido, 0, '2015-12-28 09:06:03',25),
(idPedido, 0, '2015-12-29 11:11:11',26),
(idPedido, 0, '2015-12-30 15:06:09',27),
(idPedido, 0, '2015-12-31 17:56:58',28),
(idPedido, 0, '2016-01-03 09:25:00',29);

-- SELECT * FROM pedidos


-- DELETE FROM pedidosprodutos
INSERT INTO pedidosprodutos
(Pedido, Produto, quantidade)
VALUES
(1,1, 2),
(1,6, 2),
(2,2, 5),
(2,10, 9),
(2,8, 5),
(3,1, 1),
(4,3, 10),
(5,8, 12),
(6,14, 2),
(7, 4, 3),
(8, 21, 1),
(8, 2, 1),
(8,5,4),
(9,13, 5),
(9, 12, 5),
(10,14,2),
(10,13,2),
```

```
(11, 22, 1),
(11, 6, 3),
(12, 8, 1),
(13, 9, 6),
(14, 10, 6),
(15, 19, 2),
(16, 16, 4),
(17,6, 3),
(18, 9, 8),
(19, 9, 8),
(30, 21, 2),
(30, 2, 3);
-- SELECT * FROM pedidosprodutos



-- DELETE FROM funcionarios;
INSERT INTO funcionarios
(idFuncionario,horario,nib,salario,nome,nrContribuinte,dataNascimento,nrCartaoCidadao,
nrPorta,rua,freguesia,concelho,distrito,codPostal,telemovel,telefone,facebook,email,Funcionario
)
VALUES
(1, '1',00358965763256987 4569, 2000, 'Célia Figueiredo', 262646080, '1993-02-
02',14384155,27,' Rua da Costa',
'Chorente', 'Barcelos', 'Braga','4755-118', 919568697, '252957073',
'https://www.facebook.com/celianlfg','celianlgh@hotmail.com', null),
(idFuncionario, '2',03256565845698569 8563, 1100, 'José João', '262646210', '1993-03-
29',16254896,2569,' Rua da Quintã', 'Remelhe',
'Barcelos', 'Braga', '4625-485', 926354789, 253808909, 'https://www.facebook.com/JoséJoão',
'jj@gmail.com', null),
(idFuncionario, '1',12356846656987 4563210, 850, 'Carlos Daniel', 264646200, '1993-09-22','
14384154', 93,'Rua da Calçada',
'Courel','Barcelos', 'Braga', '4589-129', 911171046,
'252957056','https://www.facebook.com/carlosDaniel', 'carlosdaniel@live.com.pt', 1),
(idFuncionario, '1',65324568989996 3200124, 850, 'Adriana Pereira', 262646200, '1993-08-
08',14365163,1200, 'Rua Padre Confucio da Silva',
'Covas do Barroso', 'Boticas', 'Vila Real', '4068-110', 934885521,
258657079,'https://www.facebook.com/AdrianaPereira', 'adrianpereira_7@hotmail.com', 2),
(idFuncionario, '1',89632477811811811 8118, 650, 'Marcus Costa', 262646118, '1990-02-
01',14568974,118, 'Rua Central',
```

'Carvalhas','Amares', 'Braga','4720-118',963126744, 257527118,'https://www.facebook.com/MarcusCosta','marcuscosta@gmail.com', 2),
(idFuncionario, '3',12312312312312312312, 1090, 'Luís Oliveira', 245689056, '1989-06-22','13459687',56,'Rua das Cancelinhas',
'Oliveira Santa Maria','Vila Nova Famalicao','Porto', '4899-123', 915325568, 252951987, 'https://www.facebook.com/LuisCarlosRoqueOliveira','luis.roqueoliveira@gmail.com', 1),
(idFuncionario, '2', 123654789632023654569, 1500, 'Márcia Costa', 252646260, '1994-01-21',14513294, 46, 'Rua do Rio Alto', 'Alto',
'Póvoa de Varzim', 'Porto', '4490-002', 919299987, 253419557, 'https://www.facebook.com/marciacosta46', 'mmcostinha@gmail.com', 1),
(idFuncionario, '3', 003565477800012365478, 1090, 'Manuel Faria', 262516046, '1992-07-18', 14984561, 1569, 'Rua da Igreja Velha',
'Tabuadelo', 'Guimarães', 'Braga', '4835-599', 933365236, 252369896,'https://www.facebook.com/manuelfaria', 'manuelfaria@gmail.com', 2);
-- SELECT * FROM funcionarios;

-- DELETE FROM funcionariosprodutos;
INSERT INTO funcionariosprodutos
(Funcionario, Produto, quantidade, dataConfe)
VALUES
(4, 1, 200, '2015-12-24'),
(4, 2, 700, '2015-12-29'),
(4, 3, 700, '2015-12-30'),
(4, 11,30, '2015-12-30'),
(4, 12, 70, '2016-01-02'),
(5, 13, 25, '2016-01-02'),
(5, 4, 400, '2016-01-02'),
(5, 5, 500,'2016-01-02'),
(5, 6, 1500, '2016-01-02'),
(5, 7, 150, '2016-01-02'),
(5, 8, 200, '2016-01-03'),
(5, 17, 60, '2016-01-03'),
(6, 9, 400, '2016-01-03'),
(6, 10, 190, '2016-01-03'),
(6, 14, 30, '2016-01-03'),
(6, 15, 50, '2016-01-04'),
(6, 16, 150, '2016-01-04'),
(3, 18, 250, '2016-01-04'),
(3, 19, 70, '2016-01-04'),

```
(3, 20, 160, '2016-01-04'),
(3, 21, 25, '2016-01-04'),
(3, 22, 30, '2016-01-05'),
(7, 21, 25, '2016-01-06'),
(7, 6, 1400, '2016-01-06'),
(7, 8, 150, '2016-01-06'),
(7, 4, 350, '2016-01-06');

-- SELECT * FROM funcionariosprodutos;

-- DELETE FROM funcionariospedidos;
INSERT INTO funcionariospedidos
(Funcionario,Pedido)
VALUES
(1, 1), (1, 4), (1, 3),
(1, 5), (2, 7), (3, 8),
(1,9), (1,10), (2, 11),
(3, 12), (1, 13), (2,14),
(1,15), (2, 16), (3,17),
(1, 18), (2, 19), (1,20),
(2,21), (3, 22), (8, 23),
(2, 24), (1, 25), (2,26),
(3,27), (1,28), (2,29),
(3,30), (1, 2), (2, 6);

-- SELECT * FROM funcionariospedidos;


-- DELETE FROM clientesprodutos;
INSERT INTO clientesprodutos
(Cliente, Produto)
VALUES
(1, 1),
(1, 2),
(2, 20),
(3, 3),
(4, 9),
(5, 21),
(6, 7),
(7, 18),
```

(8, 19),
(9, 6),
(10, 21),
(11, 9),
(12, 10),
(13,10),
(14, 1),
(15, 5),
(16, 15),
(16, 1);