BLUE TECHNOLOGY

Banco de Dados Software

MASSACHEF

Carlos Henrique F. L. da Fonseca
Cleisson de Jesus Di Lauro
Luiz Guilherme Rodrigues
Lucas Caires Sanches

São Mateus
2019

Histórico de Revisões

Versão	Comentário	Data
1.0	Descrição e Especificação dos requisitos	15/11/2018
1.2	Descrição detalhada com banco de dados	16/11/2019

Sumário

1 Definição de Requisitos
2 Requisitos de Usuário
3 Diagramas
  3.1 Modelo Conceitual
  3.2 Modelo Lógico
  3.3 Modelo Físico
4. Execução dos Arquivos
5. VIEWS
6. Restrições de Integridade
6.1 TRIGGERS
6.2 Stored Procedures
7. Dicionário de Dados

1. Definição de Requisitos

Visando aprimorar seus serviços, o gerente da Pizzaria solicitou que desenvolvesse um sistema que permite o usuário realizar tarefas administrativas de forma simples e rápida, além de oferecer o atendimento ao cliente que frequenta o estabelecimento por meio de um comanda digital e auxiliar no controle de estoque. Abaixo estão descritos os requisitos que o sistema deve atender:

1 - Para melhoria do gerenciamento, todos os pedidos devem ser registrados no sistema. Este por sua vez além dos itens do pedidos, contém o valor total, a data atual e o status do pedido.

2 - Dos produtos comercializados, deve ser armazenado seu nome/descrição e preço. Um produto pode ser pronto ou a preparar, e estes compõem o estoque em que contém a quantidade de cada produto e data de validade. De um produto a preparar deve ser armazenado os ingredientes que são necessários para o seu preparo.

3 - Os funcionários devem ser registrados no sistema. Estes devem conter as seguintes informações: Nome, número da carteira de trabalho, CPF, data de admissão, data de nascimento, rua, número da residência, CEP, cargo, e-mail, conta salário.

4 - Os fornecedores devem ser registrados no sistema. Estes devem conter as seguintes informações: Razão social, nome fantasia, rua, número da sede da empresa, CEP, CNPJ, telefone, e-mail.

5 - Cada entrega do fornecedor deve ser registrada no sistema, com a data de entrega e a quantidade de produtos entregues.

6 - Não permitir que um produto fora da sua data de validade seja incluído em um pedido.

2 Requisitos de Usuário

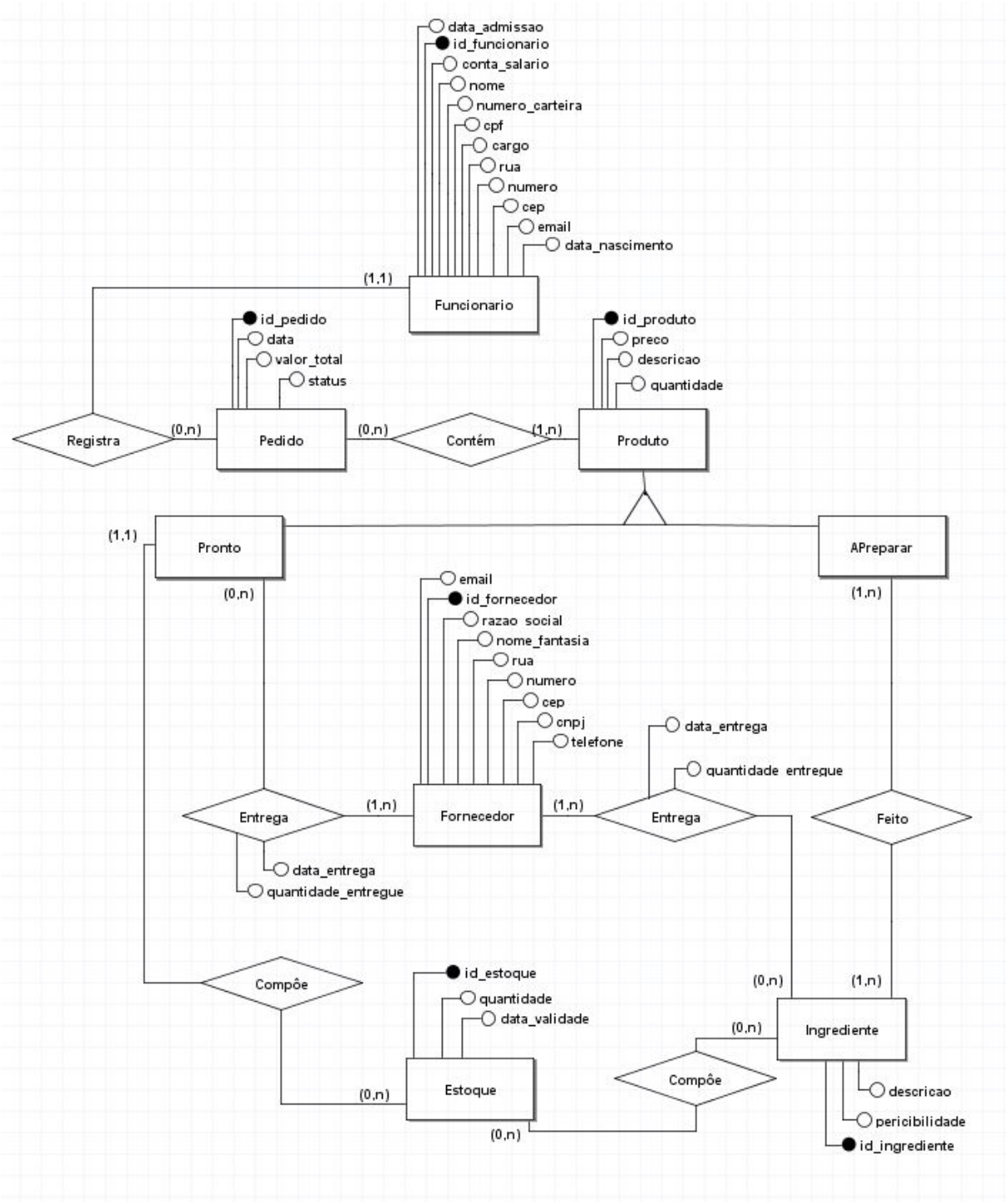
Requisitos Aprovados

Lista de Requisitos

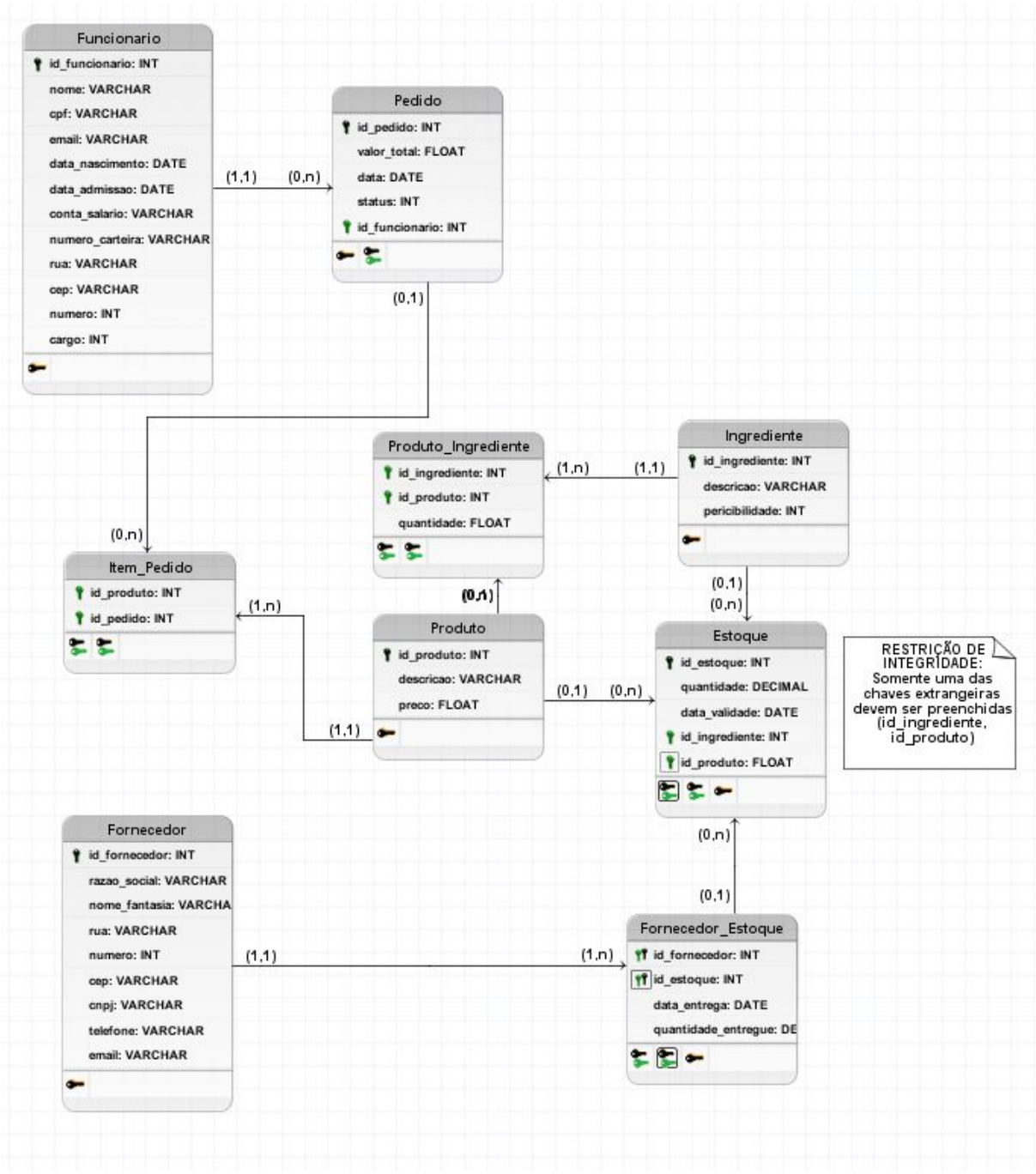
Número	Tipo	Descrição
1	Funcional	Permitir cadastrar, remover, alterar e listar produtos. De cada produto deve conter as seguintes informações: Nome, código, valor.
2	Funcional	Permitir gerenciar itens do estoque, contendo as seguintes ações: adicionar, remover, alterar e listar estoque. De cada item deve conter as seguintes informações: Nome, código, perecibilidade, data de validade, quantidade.
3	Funcional	Permitir cadastrar, remover, alterar e listar fornecedores, estes devem conter as seguintes informações: Razão social, nome fantasia, endereço, CNPJ, telefone, e-mail.
4	Funcional	Permitir a abertura e fechamento do caixa diário.
5	Funcional	Permitir através de um sistema administrativo realizar controle de estoque, gerar relatórios referente ao estoque e fluxo de caixa.
6	Funcional	Permitir criar, alterar e fechar pedido. Este por sua vez além dos itens do pedido, contém o valor total, a data atual, e o status do pedido.
7	Funcional	Permitir cadastrar, remover, alterar e listar funcionários, estes devem conter as seguintes informações: Nome, número da carteira de trabalho, CPF, data de admissão, data de nascimento, endereço, telefone de contato, e-mail, conta salário.
8	Não funcional	O sistema deverá possuir uma interface amigável e de fácil utilização.
09	Não funcional	O sistema poderá ser utilizado em desktops ou dispositivos móveis.

3 Diagramas

3.1 Modelo Entidade Relacionamento



3.2 Modelo Lógico



3.3 Modelo Físico

```
CREATE DATABASE `pizzaria`;
USE `pizzaria`;

SET NAMES utf8 ;
SET character_set_client = utf8mb4 ;

CREATE TABLE IF NOT EXISTS Fornecedor (
   id_fornecedor INT AUTO_INCREMENT NOT NULL,
   razao_social VARCHAR(90) NOT NULL,
   nome_fantasia VARCHAR(45) NOT NULL,
   rua VARCHAR(45) NOT NULL,
   numero INT NOT NULL,
   cep VARCHAR(9) NOT NULL,
   cnpj VARCHAR(18) NOT NULL,
   telefone VARCHAR(14) NOT NULL,
   email VARCHAR(45) NOT NULL,
   PRIMARY KEY (id_fornecedor)
);

CREATE TABLE IF NOT EXISTS Produto (
   id_produto  INT AUTO_INCREMENT NOT NULL,
   descricao VARCHAR(45) NOT NULL,
   preco DECIMAL NOT NULL,
   PRIMARY KEY (id_produto)
);


CREATE TABLE IF NOT EXISTS Funcionario (
   id_funcionario  INT AUTO_INCREMENT NOT NULL,
   nome VARCHAR(60) NOT NULL,
   cpf VARCHAR(14) NOT NULL,
   email VARCHAR(45) NOT NULL,
   data_nascimento DATE NOT NULL,
   data_admissao DATE NOT NULL,
```

```
    conta_salario VARCHAR(13) NOT NULL,
    numero_carteira VARCHAR(7) NOT NULL,
    rua VARCHAR(45) NOT NULL,
    cep VARCHAR(9) NOT NULL,
    numero INT NOT NULL,
    cargo INT NOT NULL,
    PRIMARY KEY (id_funcionario)
);

CREATE TABLE IF NOT EXISTS Pedido (
    id_pedido  INT AUTO_INCREMENT NOT NULL,
    valor_total FLOAT NOT NULL,
    data DATE NOT NULL,
    status INT NOT NULL,
    id_funcionario INT NOT NULL,
    PRIMARY KEY (id_pedido),
    FOREIGN KEY (id_funcionario) REFERENCES Funcionario (id_funcionario)
);

CREATE TABLE IF NOT EXISTS Item_Pedido (
    id_produto INT NOT NULL,
    id_pedido INT NOT NULL,
    FOREIGN KEY (id_produto) REFERENCES Produto (id_produto),
    FOREIGN KEY (id_pedido) REFERENCES Pedido (id_pedido)
);


CREATE TABLE IF NOT EXISTS Ingrediente (
    id_ingrediente  INT AUTO_INCREMENT NOT NULL,
    descricao VARCHAR(45) NOT NULL,
    perecibilidade INT NOT NULL,
    PRIMARY KEY (id_ingrediente)
);
```

```
CREATE TABLE IF NOT EXISTS Estoque (
    id_estoque  INT AUTO_INCREMENT NOT NULL,
    quantidade FLOAT NOT NULL,
    data_validade DATE NOT NULL,
    id_ingrediente INT,
    id_produto INT,
    PRIMARY KEY (id_estoque),
    FOREIGN KEY (id_produto) REFERENCES Produto (id_produto),
    FOREIGN KEY (id_ingrediente) REFERENCES Ingrediente (id_ingrediente)
);

CREATE TABLE IF NOT EXISTS Fornecedor_Estoque(
    id_fornecedor INT NOT NULL,
    id_estoque INT NOT NULL,
    data_entrega DATE NOT NULL,
    quantidade_entrega DECIMAL NOT NULL,
    PRIMARY KEY (id_fornecedor, id_estoque),
    FOREIGN KEY (id_fornecedor) REFERENCES Fornecedor (id_fornecedor),
    FOREIGN KEY (id_estoque) REFERENCES Estoque (id_estoque)
);

CREATE TABLE IF NOT EXISTS Produto_Ingrediente (
        id_produto INT NOT NULL,
    id_ingrediente INT NOT NULL,
    quantidade FLOAT NOT NULL,
    FOREIGN KEY (id_ingrediente) REFERENCES Ingrediente (id_ingrediente),
    FOREIGN KEY (id_produto) REFERENCES Produto (id_produto)
);
```

4. Execução dos Arquivos

A partir daqui, precisamos que seja executado os arquivos **criaBD**, **cargaBD**, **proceduresBD**, **triggersBD** e **viewsBD**, nesta respectiva ordem.

OBS: Caso não consiga abrir os arquivos diretamente no workbench, abra-os em qualquer editor de texto e copie todas as instruções presentes no arquivo, em seguida cole e execute no workbench. Realize esse procedimento em cada um dos arquivos: **criaBD**, **cargaBD**, **proceduresBD**, **triggersBD** e **viewsBD**, nesta respectiva ordem.

5. VIEWS

Lista todos os ingredientes disponiveis em estoque.

```
CREATE VIEW ingredientesDisponiveis AS

   (SELECT DISTINCT descricao, e.id_ingrediente

    FROM Estoque e

    JOIN Ingrediente i

    ON i.id_ingrediente = e.id_ingrediente

    WHERE e.id_estoque IS NOT NULL);
```

Lista todos os produtos disponíveis em estoque.

```
CREATE VIEW produtosDisponiveis AS

     (SELECT DISTINCT descricao, e.id_produto

     FROM Estoque e

     JOIN Produto p

      ON e.id_produto = p.id_produto

     WHERE id_estoque IS NOT NULL);
```

Apresenta a quantidade total dos produtos no estoque.

```
CREATE VIEW quant_produtos_estoque AS

SELECT

p.id_produto,

COALESCE(SUM(e.quantidade), 0) AS quant_em_estoque

FROM Estoque e

RIGHT JOIN produto p

      ON p.id_produto = e.id_produto

GROUP BY p.id_produto;
```

Apresenta a quantidade total de ingredientes no estoque.

```
CREATE VIEW quant_ingredientes_estoque AS

SELECT

i.id_ingrediente,

COALESCE(SUM(e.quantidade), 0) AS quant_em_estoque

FROM Estoque e

RIGHT JOIN ingrediente i

     ON i.id_ingrediente = e.id_ingrediente

GROUP BY i.id_ingrediente;
```

Apresenta os pedidos com status de a fazer.

```
CREATE VIEW pedidos_afazer AS

SELECT*

FROM

     Pedido

WHERE STATUS = 0;
```

Retorna o faturamento do dia até o momento.

```
CREATE VIEW faturamento_dia AS

SELECT

sum(p.valor_total) AS faturamento

FROM pedido p

WHERE p.data = curdate() AND p.status = 1;



select * from faturamento_diaingredientesdisponiveis
```

6. Restrições de Integridade

6.1 TRIGGERS

1)Trigger para gerar o valor total do pedido, a cada pedido adicionado ao pedido, o preço do pedido é somado ao valor total.

```
DELIMITER $

CREATE TRIGGER Tgr_Item_Pedido AFTER INSERT

ON Item_Pedido

FOR EACH ROW

BEGIN

     UPDATE  Pedido pe

          SET valor_total = valor_total + (SELECT p.preco FROM
Produto P WHERE NEW.id_produto = p.id_produto)

     WHERE pe.id_pedido = NEW.id_pedido;

END $

DELIMITER ;
```

2) Trigger verifica se um lote de um estoque esgotou, e caso tenha esgotado aloca o lote de validade mais proxima para uso.

```
DELIMITER $

CREATE TRIGGER tgr_estoque_negativo

AFTER UPDATE ON Estoque

FOR EACH ROW

BEGIN

     DECLARE validade_mais_proxima DATE;

    IF NEW.quantidade < 0 THEN

           IF NEW.id_produto IS NOT NULL THEN

                SET validade_mais_proxima = (SELECT MIN(data_validade) FROM Estoque e WHERE e.id_produto = NEW.id_produto) AND NEW.quantidade > 0;

                UPDATE Estoque e SET e.quantidade = e.quantidade - NEW.quantidade WHERE e.data_validade = validade_mais_proxima AND e.id_produto = NEW.id_produto;

                UPDATE Estoque e SET e.quantidade = 0 WHERE e.data_validade = NEW.data_validade AND e.id_produto = NEW.id_produto;

           ELSE

                SET validade_mais_proxima = (SELECT MIN(data_validade) FROM Estoque e WHERE e.id_ingrediente = NEW.id_ingrediente) AND NEW.quantidade > 0;

                UPDATE Estoque e SET e.quantidade = e.quantidade - NEW.quantidade WHERE e.data_validade = validade_mais_proxima AND e.id_produto = NEW.id_ingrediente;

                  UPDATE Estoque e SET e.quantidade = 0 WHERE e.data_validade = NEW.data_validade AND e.id_produto = NEW.id_ingrediente;

           END IF;

    END IF;

END $
```

6.2 Stored Procedures

1) Garante que id de produto e id de ingrediente, não podem coexistir, ou seja somente uma pode ser válida.

```
DELIMITER //

CREATE PROCEDURE SP_INSERE_ESTOQUE (IN quantidade INT, IN
validade DATE, IN tipo INT, IN id INT)

BEGIN

    IF (tipo = 1) THEN

          INSERT                                          INTO
Estoque(quantidade,data_validade,id_ingrediente,id_produto)
VALUES(quantidade,validade,null,id);

     ELSE                                INSERT             INTO
Estoque(quantidade,data_validade,id_ingrediente,id_produto)
VALUES(quantidade,validade,id,null);

END IF;

END //

DELIMITER ;
```

2)Insere um produto em um determinado pedido e, caso seja um produto pronto, debita-o na tabela do estoque. Caso seja um produto que é composto por outros ingredientes, verifica quais e a quantidade de cada ingrediente e posteriormente debita-os da tabela de estoque. Em ambos os casos, o debito da quantidade é feito no estoque do produto ou ingrediente com a data de validade mais próxima da atual.

```
CREATE PROCEDURE insere_produto_pedido(
     IN id_produto INT,
    IN id_pedido INT)
BEGIN
     DECLARE fim INT DEFAULT FALSE;
     DECLARE quant FLOAT;
    DECLARE ingr INT;
    DECLARE validade_mais_proxima DATE;
    DECLARE cursorQuant CURSOR FOR
          SELECT
               id_ingrediente,
```

```
                quantidade
            FROM Produto_Ingrediente pi
            WHERE id_produto = pi.id_produto;
      DECLARE CONTINUE HANDLER FOR NOT FOUND SET fim = 1;
      IF ((id_produto IN (SELECT id_produto FROM produto)) AND
(id_pedido IN (SELECT id_pedido FROM pedido))) THEN
      INSERT INTO Item_Pedido VALUES (id_produto,id_pedido);
             IF (id_produto IN (SELECT pi.id_produto FROM
Produto_Ingrediente pi)) THEN
                OPEN cursorQuant;
             loopQuant: LOOP
                FETCH cursorQuant INTO ingr,quant;
                IF fim THEN
                     LEAVE loopQuant;
                END IF;
                     IF (quant <= (SELECT quant_em_estoque FROM
quant_ingredientes_estoque qie WHERE qie.id_ingrediente = ingr))
THEN
                     SET validade_mais_proxima = (SELECT
MIN(data_validade) FROM Estoque e WHERE (e.id_ingrediente = ingr
AND data_validade > curdate()));
                     UPDATE Estoque es
                          SET es.quantidade = es.quantidade -
quant
                     WHERE
                          es.data_validade = validade_mais_proxima
AND es.id_ingrediente = ingr;
                END IF;
             END LOOP;
      ELSE
           IF ((SELECT quant_em_estoque FROM quant_produtos_estoque
qpe WHERE id_produto = qpe.id_produto) > 0) THEN
                SET validade_mais_proxima = (SELECT
MIN(data_validade) FROM Estoque e WHERE e.id_produto = id_produto
AND data_validade > curdate());
                UPDATE Estoque es
                     SET es.quantidade = es.quantidade - 1
                WHERE
                     es.data_validade = validade_mais_proxima AND
id_produto = es.id_produto;
           END IF;
      END  IF;
    END IF;
END $
```

7. Dicionário de Dados

Tabela: ***Funcionário***

PK	FK	Nome	Descrição	Tipo	Tamanho	Formato	Restrições
X		id_funcionario	Código que identifica unicamente os dados de um funcionário no banco de dados.	Inteiro	-	-	-
		nome	Nome completo do funcionário.	Texto	60	-	-
		cpf	Número do Cadastro de Pessoa Física do funcionário.	Texto	14	999.999.999-99	O CPF deve ser válido.
		email	Endereço de email do funcionário.	Texto	45	-	-
		data_nascimento	Data de Nascimento do funcionário.	Data	10	DD/MM/AAAA	O funcionário deve ser maior de idade.
		data_admissao	Data em que o funcionário foi admitido na empresa.	Data	10	DD/MM/AAAA	-
		conta_salario	Número da conta bancário do funcionário para depósito do salário.	Texto	13	99999999999-9	-
		numero_carteira	Número da carteira de trabalho do funcionário.	Texto	7	99999999	-
		rua	Rua de residência do funcionário.	Texto	45	-	-
		cep	Número do código postal da residência do funcionário.	Texto	9	99999-999	O CEP deve ser válido.
		numero	Número da residência do funcionário.	Inteiro	-	-	-
		cargo	Código que identifica o cargo atual do funcionário 1- Gerente 2 -Caixa 3 - Garçom 4 - Chef 5 - Subchefe	Inteiro	-	-	Valor deve estar entre 1 e 5.

Tabela: ***Fornecedor***

PK	FK	Nome	Descrição	Tipo	Tamanho	Formato	Restrições
X		id_fornecedor	Código que identifica unicamente os dados de um fornecedor no banco de dados.	Inteiro	-	-	-
		razao_social	Nome devidamente registrado sob o qual o fornecedor exerce suas atividades.	Texto	90	-	-
		nome_fantasia	Nome de designação popular de título de estabelecimento do fornecedor.	Texto	45	-	-
		cnpj	Número do Cadastro Nacional da Pessoa Jurídica.	Texto	18	99.999.999/9999-99	O CNPJ deve ser válido.
		email	Endereço de email do fornecedor.	Texto	45	-	-
		telefone	Número de telefone do fornecedor.	Texto	14	(27)99999-9999	-
		rua	Rua em que se localiza a empresa do fornecedor.	Texto	45	-	-
		cep	Número do código postal da empresa do fornecedor.	Texto	9	99999-999	O CEP deve ser válido.
		numero	Número da sede da empresa do fornecedor.	Inteiro	-	9999	-

Tabela: ***Pedido***

PK	FK	Nome	Descrição	Tipo	Tamanho	Formato	Restrições
X		id_pedido	Código que identifica unicamente os dados de um pedido no Banco de Dados.	Inteiro	-	-	-
		data	Data de registro do pedido.	Data	10	DD/MM/AAAA	-
		valor_total	Valor total do pedido.	Decimal	-	99,99	-
		status	Código que identifica a condição em que o pedido se encontra 1 - Finalizado 2 - Cancelado 3 - Em aberto	Inteiro	-	-	Valor deve estar entre 1 e 3.
	X	id_funcionario	Código que identifica unicamente os dados de um funcionário no banco de dados.	Inteiro	-	-	-

Tabela: ***item_Pedido***

PK	FK	Nome	Descrição	Tipo	Tamanho	Formato	Restrições
	X	id_pedido	Código que identifica unicamente os dados de um pedido no banco de dados.	Inteiro	-	-	-
	X	Id_produto	Código que identifica unicamente os dados de um produto no banco de dados.	Inteiro	-	-	-

Tabela: ***Produto***

PK	FK	Nome	Descrição	Tipo	Tamanho	Formato	Restrições
	X	id_produto	Código que identifica unicamente os dados de um produto no banco de dados.	Inteiro	-	-	-
		descricao	Nome ou designação do produto.	Texto	45	-	-
		preco	Valor do produto.	Decimal	-	99,99	-

Tabela: ***Produto_Ingrediente***

PK	FK	Nome	Descrição	Tipo	Tamanho	Formato	Restrições
	X	id_produto	Código que identifica unicamente os dados de um produto no banco de dados.	Inteiro	-	-	-
	X	id_ingrediente	Código que identifica unicamente os dados de um ingrediente no banco de dados.	Inteiro	-	-	-
		quantidade	Quantidade de ingrediente necessário	Decimal	-	-	-

Tabela: ***Ingrediente***

PK	FK	Nome	Descrição	Tipo	Tamanho	Formato	Restrições
X		id_ingrediente	Código que identifica unicamente os dados de um ingrediente no banco de dados.	Inteiro	-	-	-
		descricao	Nome ou designação do ingrediente.	Texto	45	-	-
		pericibilidade	Código que indica o grau de perecibilidade de um produto. 1 - Baixo 2 - Médio 3 - Alto	Inteiro	-	-	Valor deve estar entre 1 e 3.

Tabela: ***Estoque***

PK	FK	Nome	Descrição	Tipo	Tamanho	Formato	Restrições
X		id_estoque	Código que identifica unicamente os dados de um estoque no banco de dados.	Inteiro	-	-	-
		quantidade	Valor total de itens em um lote no estoque.	Decimal	-	-	Quantidade deve ser sempre maior ou igual a zero.
		data_validade	Data de validade de um lote no estoque.	Data	10	DD/MM/AAAA	-
	X	id_produto	Código que identifica unicamente os dados de um produto no banco de dados.	Inteiro	-	-	-
	X	id_ingrediente	Código que identifica unicamente os dados de um ingrediente no banco de dados.	Inteiro	-	-	-

Tabela: ***Funcionario_Estoque***

PK	FK	Nome	Descrição	Tipo	Tamanho	Formato	Restrições
X		id_estoque	Código que identifica unicamente os dados de um estoque no banco de dados.	Inteiro	-	-	-
		data_entrega	Data de entrega de um lote no estoque.	Data	10	DD/MM/AAAA	-
		quantidade_entrega	Valor total de itens em um lote entregues no estoque.	Decimal	-	-	Quantidade deve ser sempre maior ou igual a zero.
	X	id_fornecedor	Código que identifica unicamente os dados de um fornecedor no banco de dados.	Inteiro	-	-	-