INSTITUTO FEDERAL DA PARAÍBA

LEANDERSON COELHO DOS SANTOS
IAN CARNEIRO TEIXEIRA DE ARAÚJO

GESTÃO DE SUPERMERCADO

Cajazeiras - PB
2018

LEANDERSON COELHO DOS SANTOS
IAN CARNEIRO TEIXEIRA DE ARAÚJO

GESTÃO DE SUPERMERCADO

Trabalho apresentado ao Curso Superior Tecnológico em Análise e Desenvolvimento de Sistemas do IFPB – Instituto Federal de Educação, Ciência e Tecnologia da Paraíba, para a disciplina Banco de Dados.

Prof. Dr. Fábio Gomes de Andrade

Sumário

1. Problemática....3
2. Modelo Conceitual....4
2.1 Levantamento de requisitos....4
2.2 Diagrama de entidade e relacionamento....6
2.3 Dicionário conceitual de dados....7
3. Modelo lógico....10
3.1 Esquema Entidade Relacionamento....10
3.2 Refinamento do Esquema Entidade Relacionamento....11
3.3 Dicionario lógico de dados....11
4 Modelo físico....20
4.1 scripts SQL para criação e povoamento das tabelas....20
4.2 Criando índices....30
4.3 Criando visões....31
4.4 scripts SQL para consultas....32
4.5 Criando gatilhos....36
4.6 Criando procedimentos armazenados....37

1. Problemática

O constante crescimento de um supermercado o levou de pequeno para médio porte tornando mais complexo até mesmo aquilo que é corriqueiro no contexto de um supermercado, consequentemente a quantidade de dados gerados todos os dias aumentaram significativamente trazendo a necessidade de que estes dados sejam armazenados, gerenciados, derivados em informações úteis e que tenham integridade, ou seja, que sejam consistentes, duráveis, atômicas e isoladas. Além disso o acesso e a gerência destas informações devem ser restritas a pessoas específicas. Percebe-se então a necessidade de um banco de dados, do contrário é inevitável que surjam problemas como a falta de investimentos futuros e de grande valor para o supermercado, prejuízos que são consequências de uma má gerencia(como por exemplo) do estoque e perda de informações essenciais para o bom funcionamento do estabelecimento.

2. Modelo Conceitual

2.1 Levantamento de requisitos

O supermercado possui um conjunto de funcionários. É armazenado de cada funcionário sua matrícula, nome, CPF, telefone, salário e data de admissão. É também armazenado o setor específico na qual cada funcionário trabalha. Além disso um funcionário pode administrar compras feitas para o abastecimento do supermercado, serão armazenados para compra: código da compra, valor e data que foi efetuada a compra. Um funcionário pode também realizar vendas como caixa, para estas vendas serão armazenados o código de venda, valor, data da venda, forma de pagamento. No momento em que a venda for efetuada será armazenado a quantidade do mesmo produto e o preço unitário. O pagamento da venda pode ser feito de três formas diferentes, em dinheiro, no cartão(onde serão armazenados o número do cartão, titular, bandeira e o número de parcelas) e no cheque (onde será armazenado o número do mesmo). Caso o cliente queira pagar com cheque é necessário que ele seja cadastrado, todo cliente cadastrado deverá ter nome, telefone, endereço e CPF.

O supermercado tem setores que são gerenciados por funcionários, para cada setor é armazenado o nome, e seu respectivo código, também é necessário armazenar a data de início do funcionário que o gerencia e da mesma forma a data do fim da gerência.

As compras do supermercado possuem fornecedores e produtos. Para a compra é preciso armazenar o código da compra, valor, data da compra e produtos fornecidos. Para os produtos fornecidos são arquivados a descrição do produto, seu código de barras e o valor unitário, devem ser armazenados a quantidade de produtos e o preço unitário no momento em que foi realizado a compra, é preciso manter também as informações dos fornecedores, que são: CNPJ, nome, telefones e endereço(número, rua, bairro, cidade). Para o estoque de produtos é necessário guardar o id para os lotes, quantidade de lotes do mesmo produto, quantidade de produtos por lote, data de validade e a data do fornecimento.

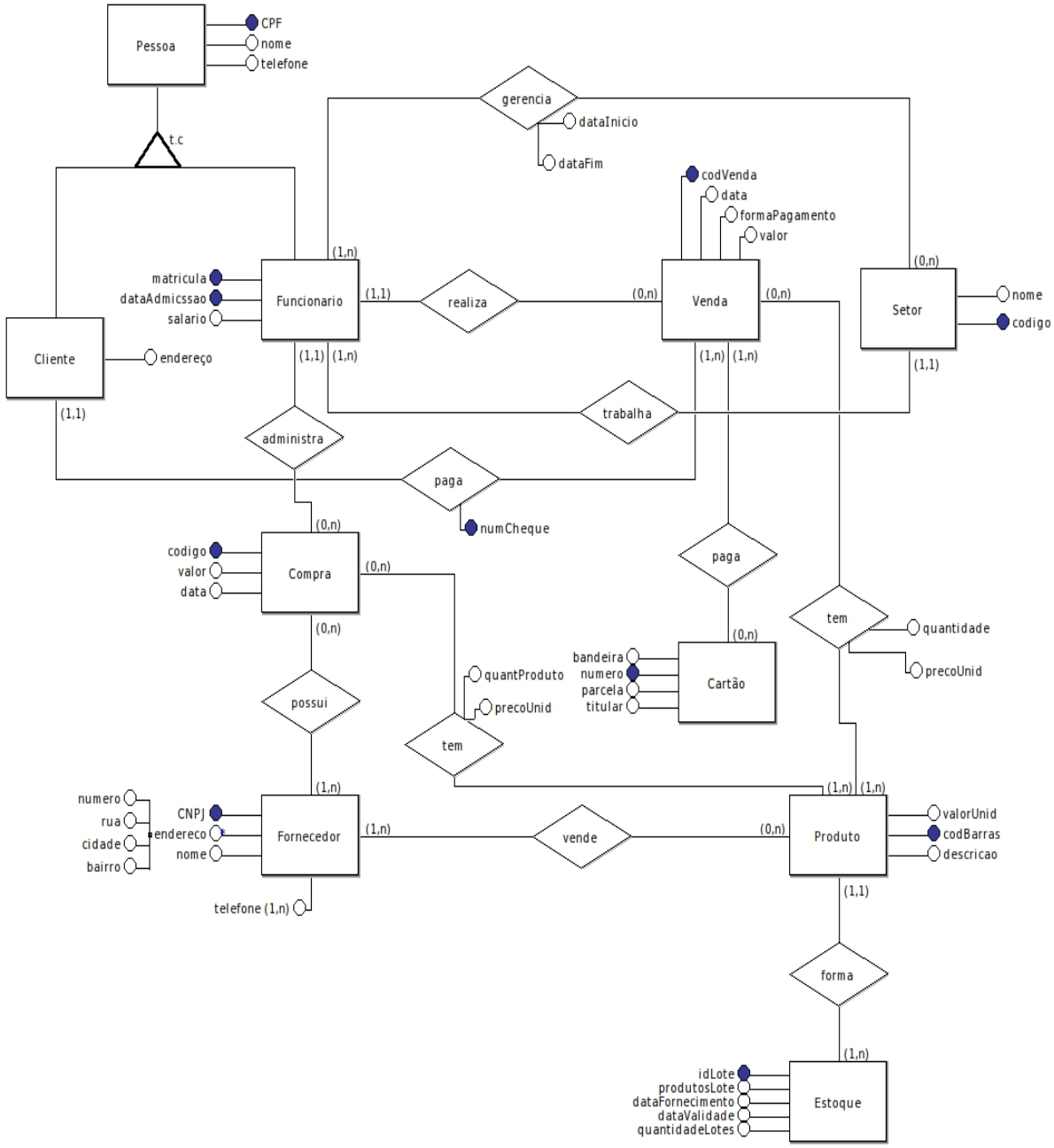
O banco de dados conta com um sistema de consultas, onde estas consultas trarão informações relevantes para situações que são corriqueiras no contexto do supermercado. Para fins de conhecimento e gerenciamento de produtos vendidos é necessário agrupar as vendas por um intervalo de datas. O supermercado tem centenas de produtos em estoque, para saber a possibilidade de falta de produtos nas próximas semanas é importante que seja recuperado o código de barras, o nome e a quantidade de todos os produtos em estoque ordenados de forma crescente pela quantidade.

Para que as prateleiras não forneçam produtos fora do prazo de validade é importante que ao informar um intervalo de datas, sejam apresentados os produtos que tem sua data de validade nesse período de tempo ordenados pela data como primeiro critério de ordenação e depois pela descrição do produto. O supermercado precisa ordenar por data os produtos que foram comprados para estoque. Os funcionários responsáveis pelas compras do supermercado precisam de informações importantes fornecidas de forma rápida, para isso quando for especificado o nome do fornecedor, o sistema retornará os telefones para contato. O dono também preza por um controle de informações dos funcionários que administraram as compras para abastecimento e estoque, por isto, ao inserir o código da compra deve ser retornado o nome e a matrícula do funcionário que ficou responsável além do valor total da compra.

O supermercado é divido em setores, é necessário que ao informar o setor seja exibido todos os funcionários que trabalham no mesmo, também é essencial saber o único funcionário que gerencia um determinado setor. É preciso ter conhecimento sobre o salário de cada funcionário.

Para a gestão financeira do supermercado, pode ser informado uma forma de pagamento e um intervalo de datas e será retornado a soma dos valores das vendas do supermercado nesse período e nesta forma de pagamento. É essencial que haja agilização no calculo do valor total da venda de produtos, é preciso que o funcionário consiga filtrar o valor e a descrição de um produto específico pelo código de barras.

2.2 Diagrama de entidade e relacionamento



2.3 Dicionário conceitual de dados.

Entidade Pessoa:

- Generalização da entidade Cliente e Funcionário. Guarda atributos comum a uma pessoa.

Atributo(s):

- CPF: armazena o número do CPF de uma pessoa.
- telefone: armazena o número de telefone de uma pessoa.
- nome: armazena o nome de uma pessoa.

Entidade Funcionário:

- Especialização da entidade Pessoa. Criada para armazenar as informações específicas de um Funcionário.

Atributo(s):

- salario: armazena o salário do funcionário.
- dataadmissao: armazena a data de admissão do funcionário.
- matricula: armazena o número da matrícula do funcionário.

Entidade Cliente:

- Especialização da entidade Pessoa. Criada para armazenar informações de um cliente cadastrado.

Atributos(s):

- endereço: armazena o endereço do cliente cadastrado.

Entidade Cartão:

- Criada para armazenar as informações do cartão utilizado em uma compra.

Atributos(s):

- parcela: armazena a quantidade de parcelas do valor total.
- bandeira: armazena a bandeira do cartão.
- numero: armazena o número do cartão.
- titular: armazena o nome do titular do cartão.

Entidade Setor:

- Criado para armazenar informações relacionadas aos setores do supermercado.

Atributos(s):

- nome: armazena o nome do setor.
- código: armazena o código do setor.

Entidade Venda:

- Criada para armazenar as informações de uma venda feita pelo supermercado.

Atributos(s):

- codVenda: armazena o código da venda.
- data: armazena a data que a venda foi efetuada.
- formaPagamento: armazena a forma de pagamento da venda.
- quantProduto: armazena a quantidade de produtos de uma venda.
- valor: armazena o valor total da venda.

Entidade Produto:

- Criada para armazenar informações dos produtos.

Atributo(s):

- valorUnid: armazena o valor por unidade do produto.
- codBarras: armazena o código de barras do produto.
- descricao: armazena a descrição do produto.

Entidade Fornecedor:

- Criada para armazenar informações dos fornecedores.

Atributo(s):

- CNPJ: armazena o CNPJ da empresa que o fornecedor trabalha.
- endereço: armazena o endereço da empresa que o fornecedor trabalha, formado por número, rua, cidade e bairro.
- nome: armazena o nome do fornecedor.
- telefone: armazena os telefones do fornecedor.

Entidade Estoque:

- Criada para armazenar as informações do estoque de produtos.

Atributo(s):

- idLote: armazena um único número identificador (id) para cada lote, incluindo lotes de mesmo produto, porém com validade diferente.
- produtosLote: armazena a quantidade de produtos em um único lote.
- dataFornecimento: armazena a data que o lote foi fornecido.

- dataValidade: armazena a data de validade dos produtos do lote.
- quantidadeLotes: armazena a quantidade de lotes de um produto que restam em estoque.

Entidade Compra:

- Criada para armazenar as informações das compras feitas pelo supermercado.

Atributo(s):

- codigo: armazena o código da compra.
- valor: armazena o valor da compra.
- data: armazena a data da compra.

Relacionamento(s):

- realiza: relaciona Funcionário com Venda. Cada funcionário pode realizar nenhuma ou várias vendas. Uma venda pode ser realizada por um e somente um funcionário.
- gerencia: relaciona Funcionário com Setor. Armazena a data de início e a data do fim da gerência. Um Funcionário pode gerenciar vários ou nenhum setor. Um Setor deve ser gerenciado por um ou vários Funcionário.
- paga: relaciona Cliente com Venda. Armazena o número do cheque do cliente. Um Cliente paga no cheque uma ou várias vendas. Uma Venda é paga por um e somente um cliente.
- paga: relaciona Cartão com Venda. Um cartão paga uma ou mais vendas .Uma Venda é paga por nenhum ou vários cartões.
- tem: relaciona Venda com Produto. Armazena a quantidade do produto na venda e o preço por unidade. Uma venda pode ter um ou mais produtos diferentes. Um produto pode estar em várias ou nenhuma venda.
- forma: relaciona Estoque e Produto. Um estoque é formado por um e somente um tipo de produto. Um mesmo tipo de produto forma um ou vários estoques.
- vende: relaciona Fornecedor com Produto. Um fornecedor vende vários produtos diferentes ou nenhum. Um mesmo tipo de produto é vendido por um ou vários fornecedores.
- possui: relaciona Compra com Fornecedor. Uma compra possui um ou vários fornecedores. Um fornecedor participa de várias ou nenhuma compra.
- administra: relaciona Funcionário com Compra. Um funcionário administra várias compras ou nenhuma compra. Uma compra é administrada por um e somente um funcionário.

- tem: relaciona compra com Produto. Armazena a quantidade dos produtos comprados e o preço por unidade. Uma compra deve ter pelo menos um produto. Um produto pode esta contido em nenhuma ou em várias compras.
- trabalha: relaciona Funcionário com Venda. Cada funcionário trabalha em um e somente um setor. Cada setor tem no mínimo um funcionário.

3. Modelo lógico

3.1 Esquema Entidade Relacionamento

1. Pessoa(CPF, nome, telefone)
2. Cliente(CPF, endereço)
3. Setor(codigo, nome)
4. Funcionario(matricula, CPF, dataadmissao, salario, codSetor)
5. GerenciaSetor(gerente, codSetor, dataInicio, dataFim)
6. Venda(codVenda, data, formaPagamento, valor, matFuncionario)
7. ClienteVenda(cpfCliente, codVenda, numCheque)
8. Compra(codigo, valor, data, matFuncionario)
9. Cartão(numero, bandeira, parcelas, titular)
10. VendaCartao(numeroCartao, codVenda)
11. Fornecedor(CNPJ, nome, numEndereco, rua, cidade, bairro)
12. TelefoneFornecedor(CNPJ, telefone)
13. CompraFornercedor(CNPJ, codCompra)
14. Produto(codBarras, valorUnid, descricao)
15. FornecedorProduto(CNPJ, codBarrasProduto)
16. CompraProduto(codCompra, codBarrasProduto, quantProduto, precoUnid)
17. VendaProduto(codVenda, codBarrasProduto, quantidade, precoUnid)
18. Estoque(idLote, produtosLote, dataFornecimento, dataValidade, quantidadeLotes, codBarrasProduto)

3.2 Refinamento do Esquema Entidade Relacionamento

Foi feito um refinamento no esquema Venda, movendo os atributos cpfCliente e numCheque, da entidade Cliente e do relacionamento paga, respectivamente, para uma nova tabela contendo também o codVenda, da entidade Venda, representada pelo esquema 7 da seção anterior.

3.3 Dicionario lógico de dados

Pessoa: Generalização da entidade Cliente e Funcionário. Guarda atributos comum a uma pessoa.				
Atributo	Descrição	Tipo	Domínio	Restrição
CPF	armazena o número do CPF de uma pessoa.	CHARACTER(11)	Caracteres que representam números.	* Chave primária
nome	armazena o nome de uma pessoa.	VARCHAR(100)	VARCHAR(100)	* Não nulo
telefone	armazena o número de telefone de uma pessoa.	VARCHAR(15)	Caracteres que representam números.	* Não nulo

Tabela 1 - Relação Pessoa

Cliente: Especialização da entidade Pessoa. Armazena informações de um cliente cadastrado.				
Atributo	Descrição	Tipo	Domínio	Restrição
CPF	armazena o número do CPF de uma pessoa.	CHARACTER(11)	Caracteres que representam números.	* Chave primária * Chave estrangeira referenciando CPF na tabela Pessoa
endereço	armazena o endereço de uma pessoa.	VARCHAR(100)	VARCHAR(100)	* Não nulo

Tabela 2 - Relação Cliente

Setor: armazenar informações relacionadas aos setores do supermercado.				
Atributo	Descrição	Tipo	Domínio	Restrição
nome	armazena o nome do setor	VARCHAR(50)	VARCHAR(50)	* Não nulo
codigo	armazena o código do setor	INT	Valores positivos	* Chave primária

Tabela 3 - Relação Setor

Funcionario: Especialização da entidade Pessoa. Armazena informações de um funcionário.				
Atributo	Descrição	Tipo	Domínio	Restrição
matricula	armazena a matrícula do funcionário	CHARACTER(6)	CHARACTER(6)	* Chave primária
CPF	armazena o número do CPF de um funcionário.	CHARACTER(11)	Caracteres que representam números.	* Chave estrangeira referenciando CPF na tabela Pessoa * UNIQUE
dataadmissao	armazena a data de admissão do funcionário	DATE	DATE	* Não Nulo
salario	armazena o salário do funcionário	REAL	Valores positivos	* Não nulo
codSetor	armazena o código do setor que o funcionário trabalha	INT	Valores positivos	* Não nulo * Chave estrangeira referenciando codigo na tabela Setor.

Tabela 4 - Relação Funcionario

GerenciaSetor: Armazena as informações do gerenciamento do setor.				
Atributo	Descrição	Tipo	Domínio	Restrição
gerente	armazena a matrícula do	CHARACTER(6)	CHARACTER(6)	* Chave estrangeira que referenciando

	funcionário			matricula na tabela Funcionario. * Chave primária
codSetor	armazena o código do setor que é gerenciado.	INT	Números inteiros positivos.	* Chave estrangeira que referenciando codigo na tabela Setor. * Chave primária
dataInicio	armazena a data de início do gerenciamento do setor.	DATE	DATE	* Não nulo.
dataFim	armazena a data do fim da gerencia do setor.	DATE	DATE	*Nenhuma.

Tabela 5 - Relação GerenciaSetor

Venda: Relação que armazena as informações de uma venda feita pelo supermercado.				
Atributo	Descrição	Tipo	Domínio	Restrição
codVenda	armazena o código da venda realizada.	INT	Números inteiros positivos.	* Chave primária
data	armazena a data que foi efetuada a venda	DATE	DATE	* Não nulo.
formaPagamento	armazena a forma de pagamento da venda.	VARCHAR(8)	O conjunto (“cartão”, “cheque”, “dinheiro”)	* Não nulo
valor	armazena o valor total da venda.	REAL	Números positivos.	* Não nulo
matFuncionario	armazena a matrícula do funcionário	CHARACTER(6)	CHARACTER(6)	* Chave estrangeira referenciando matricula na tabela Funcionario

	que efetuou a venda.			

Tabela 6 - Relação Venda

ClienteVenda: Relação que armazena as informações de uma venda paga no cheque.				
Atributo	Descrição	Tipo	Domínio	Restrição
codVenda	armazena o código da venda realizada.	INT	Números inteiros positivos.	* Chave estrangeira referenciando codVenda na tabela Venda.
cpfCliente	armazena o número do CPF de um cliente que pagou no cartão.	CHARACTER(11)	Caracteres que representam números.	* Chave estrangeira referenciando CPF na tabela Pessoa
numCheque	armazena o número do cheque.	CHARACTER(6)	Caracteres que representam números.	* Chave primária.

Tabela 7 - Relação ClienteVenda

Compra: Relação que armazena as informações das compras feitas pelo supermercado.				
Atributo	Descrição	Tipo	Domínio	Restrição
codigo	armazena o código da compra	INT	INT	* Chave primária
valor	armazena o valor da compra	REAL	números reais positivos	* Não nulo
data	armazena a data da compra	DATE	DATE	* Não nulo
matFuncionario	armazena a matrícula do funcionário	CHARACTER(6)	CHARACTER(6)	* Chave estrangeira que referenciando matricula na tabela

	encarregado pela compra			Funcionario.

Tabela 8 - Relação Compra

Cartão: Relação que armazena as informações do cartão de crédito utilizado em uma compra.				
Atributo	Descrição	Tipo	Domínio	Restrição
numero	armazena o número do cartão de crédito	VARCHAR(16)	Caracteres que representam números.	* Chave primária
bandeira	armazena a bandeira do cartão de crédito	VARCHAR(20)	VARCHAR(20)	* Não nulo
parcelas	armazena a quantidade de parcelas do valor total	INT	números inteiros positivos	* Não nulo
titular	armazena o nome do titular do cartão	VARCHAR(100)	VARCHAR(100)	* Não nulo

Tabela 9 - Relação Cartão

VendaCartao: Relação que armazena o número do cartão e o código das vendas pagas com cartão de crédito.				
Atributo	Descrição	Tipo	Domínio	Restrição
numeroCartao	armazena o número do cartão de crédito	VARCHAR(15)	Caracteres que representam números.	* Chave primária * Chave estrangeira referenciando numero na tabela Cartão.
codVenda	armazena o código da venda realizada.	INT	Números inteiros positivos.	* Chave primária * Chave estrangeira referenciando codVenda na tabela Venda.

Tabela 10 - Relação VendaCartao

Fornecedor: Relação que armazena as informações dos fornecedores de produtos.				
Atributo	Descrição	Tipo	Domínio	Restrição
CNPJ	armazena o CNPJ da empresa que o fornecedor trabalha	CHARACTER(18)	Caracteres que representam números	* Chave primária
nome	armazena o nome do fornecedor	VARCHAR(100)	VARCHAR(100)	* Não nulo
numEndereco	número do endereço do fornecedor	INT	Valores inteiros positivos	* Não nulo
rua	rua do endereço do fornecedor	VARCHAR(60)	VARCHAR(60)	* Não nulo
cidade	cidade do endereço do fornecedor	VARCHAR(35)	VARCHAR(35)	* Não nulo
bairro	bairro do endereço do fornecedor	VARCHAR(60)	VARCHAR(60)	* Não nulo

Tabela 11 - Relação Fornecedor

TelefoneFornecedor: Relação que armazena os números de telefone dos fornecedores.				
Atributo	Descrição	Tipo	Domínio	Restrição
CNPJ	armazena o CNPJ da empresa que o fornecedor trabalha	CHARACTER(18)	Caracteres que representam números	* Chave primária * Chave estrangeira referenciando CNPJ da relação Fornecedor
telefone	armazena o número do telefone do fornecedor	VARCHAR(15)	Caracteres que representam números	*Chave primária

Tabela 12 - Relação TelefoneFornecedor

CompraForncedor: Relação que armazena as compras feitas aos fornecedores pelo supermercado.				
Atributo	Descrição	Tipo	Domínio	Restrição
CNPJ	armazena o CNPJ da empresa que o fornecedor trabalha	CHARACTER(18)	Caracteres que representam números	* Chave primária * Chave estrangeira referenciando CNPJ da relação Fornecedor
codCompra	armazena o código da compra	INT	INT	* Chave primária. * Chave estrangeira que referenciando codigo na relação Compra

Tabela 13 - Relação CompraFornecedor

Produto: Relação que armazena as informações dos produtos do supermercado.				
Atributo	Descrição	Tipo	Domínio	Restrição
codBarras	armazena o código de barras do produto.	VARCHAR(13)	Caracteres que representam números	* Chave primária.
valorUnid	armazena o valor do produto por unidade.	REAL	Números reais positivos.	* Não nulo.
descricao	armazena uma breve descrição do produto	VARCHAR(80)	VARCHAR(80)	* Não nulo.

Tabela 14 - Relação Produto

FornecedorProduto: Relação que armazena os produtos vendidos ao supermercado pelos fornecedores.				
Atributo	Descrição	Tipo	Domínio	Restrição
CNPJ	armazena o CNPJ da empresa que o	CHARACTER(18)	Caracteres que representam números	* Chave primária * Chave estrangeira referenciando CNPJ da relação

	fornecedor trabalha			Fornecedor
CodBarrasProduto	armazena o código de barras do produto vendido.	VARCHAR(13)	Números inteiros positivos	* Chave primária. * Chave estrangeira referenciando codBarras na tabela Produto.

Tabela 15 - Relação FornecedorProduto

CompraProduto: Relação que armazena o código de barras dos produtos comprados pelo supermercado e respectivamente, os códigos das compras.				
Atributo	Descrição	Tipo	Domínio	Restrição
codCompra	armazena o código da compra feita pelo supermercado	INT	Inteiros positivos	* Chave primária * Chave estrangeira referenciando codigo da relação Compra
CodBarrasProduto	armazena o código de barras do produto vendido.	VARCHAR(13)	Números inteiros positivos	* Chave primária. * Chave estrangeira referenciando codBarras da relação Produto.

Tabela 16 - Relação CompraProduto

VendaProduto: Relação que armazena as informações dos produtos que foram vendidos.				
Atributo	Descrição	Tipo	Domínio	Restrição
codVenda	armazena o código da venda realizada.	INT	Números inteiros positivos.	* Chave primária * Chave estrangeira que referenciando codVenda na tabela Venda.
codBarrasProduto	armazena o código de barras do produto vendido.	VARCHAR(13)	Caracteres que representam números	* Chave primária. * Chave estrangeira referenciando codBarras na tabela Produto.
quantidade	armazena a	INT	Números	* Não nulo.

	quantidade do produto na venda.		inteiros positivos	

Tabela 17 - Relação VendaProduto

Estoque: Relação que armazena as informações do estoque do supermercado.				
Atributo	Descrição	Tipo	Domínio	Restrição
idLote	armazena o id(identificador) de cada lote.	INT	Números inteiros positivos	* Chave primária
produtosLote	armazena a quantidade de produtos por lote.	INT	Números inteiros positivos	* Não nulo.
dataFornecimento	armazena a data que o lote foi fornecido ao estoque	DATE	DATE	* Não nulo.
dataValidade	armazena a data de validade dos produtos do lote.	DATE	DATE	* Não nulo.
quantidadeLotes	armazena a quantidade de lotes do mesmo produto que resta no estoque.	INT	Números inteiros positivos.	* Não nulo.
codBarrasProduto	armazena o código de barras do produto vendido.	VARCHAR(13)	Caracteres que representam números	* Chave estrangeira referenciando codBarras na tabela Produto.

Tabela 18 - Relação Estoque

4 Modelo físico

4.1 scripts SQL para criação e povoamento das tabelas

Criando a relação Pessoa

```
CREATE TABLE Pessoa (
CPF CHARACTER(11),
nome VARCHAR(100) NOT NULL,
telefone VARCHAR(15) NOT NULL,
PRIMARY KEY (CPF)
);
```

Criando a relação Cliente

```
CREATE TABLE Cliente (
CPF CHARACTER(11),
endereco VARCHAR(100) NOT NULL,
PRIMARY KEY (CPF),
FOREIGN KEY (CPF) REFERENCES Pessoa (CPF)
);
```

Criando a relação Setor

```
CREATE TABLE Setor (
nome VARCHAR(50) NOT NULL,
codigo INT,
PRIMARY KEY (codigo),
CHECK (codigo>0)
);
```

Criando a relação Funcionario

```
CREATE TABLE Funcionario(
matricula CHARACTER(6),
CPF CHARACTER(11) UNIQUE,
dataAdmissao DATE NOT NULL,
salario REAL NOT NULL,
codSetor INT NOT NULL,
PRIMARY KEY (matricula),
FOREIGN KEY (CPF) REFERENCES Pessoa(CPF),
FOREIGN KEY (codSetor) REFERENCES Setor(codigo),
```

```
CHECK(salario>0)
);
```

Criando a relação GerenciaSetor

```
CREATE TABLE GerenciaSetor(
gerente CHARACTER(6),
codSetor INT,
dataInicio DATE NOT NULL,
dataFim DATE,
PRIMARY KEY (gerente, codSetor),
FOREIGN KEY (gerente) REFERENCES Funcionario (matricula),
FOREIGN KEY (codSetor) REFERENCES Setor (codigo)
);
```

Criando a relação Venda

```
CREATE TABLE Venda (
codVenda INT,
data DATE NOT NULL,
formaPagamento VARCHAR(8) NOT NULL,
valor REAL NOT NULL,
matFuncionario CHARACTER(6),
PRIMARY KEY(codVenda),
FOREIGN KEY (matFuncionario) REFERENCES Funcionario (matricula),
CHECK (codVenda>0),
CHECK (valor>0)
);
```

Criando a relação ClienteVenda

```
CREATE TABLE ClienteVenda (
codVenda INT,
cpfCliente CHARACTER(11),
numCheque VARCHAR(10),
FOREIGN KEY (codVenda) REFERENCES Venda (codVenda),
FOREIGN KEY (cpfCliente) REFERENCES Pessoa (CPF),
PRIMARY KEY (numCheque)
);
```

Criando a relação Compra

```
CREATE TABLE Compra (
codigo INT,
valor REAL NOT NULL,
data DATE NOT NULL,
matFuncionario CHARACTER(6),
CHECK (valor>0),
PRIMARY KEY (codigo),
FOREIGN KEY (matFuncionario) REFERENCES Funcionario(matricula)
);
```

Criando a relação Cartão

```
CREATE TABLE Cartão (
numero VARCHAR(16),
bandeira VARCHAR(20) NOT NULL,
parcelas INT NOT NULL,
titular VARCHAR(100) NOT NULL,
CHECK (parcelas>0),
PRIMARY KEY (numero)
);
```

Criando a relação VendaCartao

```
CREATE TABLE VendaCartao (
numeroCartao VARCHAR(16),
codVenda INT,
PRIMARY KEY (numeroCartao, codVenda),
FOREIGN KEY (numeroCartao) REFERENCES Cartão(numero),
FOREIGN KEY (codVenda) REFERENCES Venda(codVenda)
);
```

Criando a relação Fornecedor

```
CREATE TABLE Fornecedor(
CNPJ CHARACTER(18),
nome VARCHAR(100) NOT NULL,
numEndereco INT NOT NULL,
rua VARCHAR(60) NOT NULL,
cidade VARCHAR(35) NOT NULL,
bairro VARCHAR(60) NOT NULL,
CONSTRAINT pk_fornecedor PRIMARY KEY (CNPJ)
```

```
);
```

Criando a relação TelefoneFornecedor

```
CREATE TABLE TelefoneFornecedor(
CNPJ CHARACTER(18),
telefone VARCHAR(15),
CONSTRAINT pk_telefoneFornecedor PRIMARY KEY (CNPJ,telefone),
CONSTRAINT fk_cnpjForncedor FOREIGN KEY (CNPJ) REFERENCES Fornecedor(CNPJ)
);
```

Criando a relação CompraFornecedor

```
CREATE TABLE CompraFornecedor(
CNPJ CHARACTER(18),
codCompra INT,
CONSTRAINT pk_compraForncedor PRIMARY KEY (CNPJ, codCompra),
CONSTRAINT fk_compraFornecedorForFornecedor FOREIGN KEY (CNPJ) REFERENCES
Fornecedor(CNPJ),
CONSTRAINT fk_compraFornecedorForCompra FOREIGN KEY (codCompra) REFERENCES
Compra(codigo)
);
```

Criando a relação Produto

```
CREATE TABLE Produto(
codBarras VARCHAR(13),
valorUnid REAL NOT NULL,
descricao VARCHAR(80) NOT NULL,
CONSTRAINT pk_produto PRIMARY KEY (codBarras)
);
```

Criando a relação ForneccedorProduto

```
CREATE TABLE FornecedorProduto(
CNPJ CHARACTER(18),
codBarrasProduto VARCHAR(13),
CONSTRAINT pk_fornecedorProduto PRIMARY KEY (CNPJ, codBarrasProduto),
CONSTRAINT fk_fornecedorProdutoForFornecedor FOREIGN KEY (CNPJ) REFERENCES
Fornecedor(CNPJ),
CONSTRAINT fk_fornecedorProdutoForProduto FOREIGN KEY (codBarrasProduto)
REFERENCES Produto(codBarras)
```

```
);
```

Criando a relação CompraProduto

```
CREATE TABLE CompraProduto(
codCompra INT,
codBarrasProduto VARCHAR(13),
CONSTRAINT pk_compraProduto PRIMARY KEY (codCompra, codBarrasProduto),
CONSTRAINT fk_compraProdutoForCompra FOREIGN KEY (codCompra) REFERENCES
Compra(codigo),
CONSTRAINT fk_compraProdutoForProduto FOREIGN KEY (codBarrasProduto)
REFERENCES Produto(codBarras)
);
```

Criando a relação VendaProduto

```
CREATE TABLE VendaProduto(
codVenda INT,
codBarrasProduto VARCHAR(13),
quantidade INT NOT NULL,
CONSTRAINT pk_vendaProduto PRIMARY KEY (codVenda, codBarrasProduto),
CONSTRAINT fk_vendaProdutoForVenda FOREIGN KEY (codVenda) REFERENCES
Venda(codVenda),
CONSTRAINT fk_vendaProdutoForProduto FOREIGN KEY (codBarrasProduto)
REFERENCES Produto(codBarras),
CONSTRAINT quantidadePositiva CHECK (quantidade > 0)
);
```

Criando a relação Estoque

```
CREATE TABLE Estoque(
idLote INT,
produtosLote INT NOT NULL,
dataFornecimento DATE NOT NULL,
dataValidade DATE NOT NULL,
quantidadeLotes INT NOT NULL,
codBarrasProduto VARCHAR(13),
CONSTRAINT pk_estoque PRIMARY KEY (idLote),
CONSTRAINT fk_estoqueForProduto FOREIGN KEY (codBarrasProduto) REFERENCES
Produto(codBarras),
CONSTRAINT quantMinimaProdutosLote CHECK(produtosLote > 0),
CONSTRAINT quantMinimaLotes CHECK(quantidadeLotes >= 0)
```

```
);
```

Povoando a relação Pessoa

```
INSERT INTO Pessoa VALUES('80143817000', 'Alice', '83998350911');
INSERT INTO Pessoa VALUES('57126386060', 'Sophia', '83987021883');
INSERT INTO Pessoa VALUES('69305311032', 'Helena', '83996339656');
INSERT INTO Pessoa VALUES('56059472060', 'Valentina', '83984411400');
INSERT INTO Pessoa VALUES('58486545064', 'Miguel', '83988650159');
INSERT INTO Pessoa VALUES('43942001055', 'Arthur', '83985734990');
INSERT INTO Pessoa VALUES('10427597072', 'Bernardo', '83985459432');
INSERT INTO Pessoa VALUES('99415616059', 'Heitor', '83992971315');
INSERT INTO Pessoa VALUES('43995365000', 'Júlia', '83986753859');
INSERT INTO Pessoa VALUES('94480025057', 'Pedro', '83995609748');
INSERT INTO Pessoa VALUES('12143817000', 'João', '83999511952');
```

Povoando a relação Cliente

```
INSERT INTO Cliente VALUES('80143817000', 'RUA ZÉ DAS COUVES, 132');
INSERT INTO Cliente VALUES('94480025057', 'RUA ZÉ DAS COUVES, 321');
INSERT INTO Cliente VALUES('57126386060', 'RUA ZÉ DAS COUVES, 221');
INSERT INTO Cliente VALUES('43995365000', 'RUA PROFESSOR ALENCAR, 789');
INSERT INTO Cliente VALUES('99415616059', 'RUA PROFESSOR ALENCAR, 453');
INSERT INTO Cliente VALUES('69305311032', 'RUA PROFESSOR ALENCAR, 467');
```

Povoando a relação Setor

```
INSERT INTO Setor VALUES ('atendimento', 1);
INSERT INTO Setor VALUES ('financeiro', 2);
INSERT INTO Setor VALUES ('gerência', 3);
INSERT INTO Setor VALUES ('almoxarifado', 4);
INSERT INTO Setor VALUES ('marketing', 5);
```

Povoando a relação Funcionário

```
INSERT INTO Funcionario VALUES('1111-1', '56059472060', '2017-01-03',
1000.00, 1);
INSERT INTO Funcionario VALUES('1111-2', '58486545064', '2018-05-05',
1500.00, 1);
INSERT INTO Funcionario VALUES('1111-3', '43942001055', '2014-05-05',
4043.00, 3);
```

```
INSERT INTO Funcionario VALUES('1111-4', '10427597072', '2015-04-23',
3050.00, 5);
INSERT INTO Funcionario VALUES('1111-5', '80143817000', '2013-03-29',
3050.00, 2);
INSERT INTO Funcionario VALUES('1111-6', '12143817000', '2015-03-29',
2000.00, 4);
```

Povoando a relação GerenciaSetor

```
INSERT INTO GerenciaSetor VALUES('1111-1', 1, '2017-02-03', '2018-04-05');
INSERT INTO GerenciaSetor VALUES('1111-2', 1, '2018-06-29', null);
INSERT INTO GerenciaSetor VALUES('1111-3', 3, '2014-06-05', null);
INSERT INTO GerenciaSetor VALUES('1111-4', 5, '2015-05-23', null);
INSERT INTO GerenciaSetor VALUES('1111-5', 2, '2013-03-29', null);
INSERT INTO GerenciaSetor VALUES('1111-6', 4, '2015-03-29', null);
```

Povoando a relação Venda

```
INSERT INTO Venda VALUES(1, '2018-09-7', 'cartão', 197.91, '1111-2');
INSERT INTO Venda VALUES(2, '2018-09-7', 'cheque', 393.3, '1111-2');
INSERT INTO Venda VALUES(3, '2018-09-7', 'cartão', 26.95, '1111-1');
INSERT INTO Venda VALUES(4, '2018-09-7', 'dinheiro', 41.4, '1111-1');
INSERT INTO Venda VALUES(5, '2018-09-7', 'cheque', 107.8, '1111-1');
INSERT INTO Venda VALUES(6, '2018-09-7', 'cartão', 29.4, '1111-2');
INSERT INTO Venda VALUES(7, '2018-09-10', 'cartão', 48.93, '1111-2');
INSERT INTO Venda VALUES(8, '2018-09-10', 'cheque', 285.87, '1111-2');
INSERT INTO Venda VALUES(9, '2018-09-10', 'cartão', 6.99, '1111-1');
INSERT INTO Venda VALUES(10, '2018-09-10', 'cheque', 175.00, '1111-1');
INSERT INTO Venda VALUES(11, '2018-09-10', 'cheque', 529.2, '1111-1');
```

Povoando a relação ClienteVenda

```
INSERT INTO ClienteVenda VALUES(5, '80143817000', '000545');
INSERT INTO ClienteVenda VALUES(2, '94480025057', '003251');
INSERT INTO ClienteVenda VALUES(8, '43995365000', '055388');
INSERT INTO ClienteVenda VALUES(10, '99415616059', '012756');
INSERT INTO ClienteVenda VALUES(11, '69305311032', '000989');
```

Povoando a relação Compra

```
INSERT INTO Compra VALUES(5, 5022.00, '2018-09-06', '1111-6');
INSERT INTO Compra VALUES(4, 3050.00, '2018-09-05', '1111-6');
INSERT INTO Compra VALUES(3, 6770.00, '2018-09-04', '1111-6');
INSERT INTO Compra VALUES(2, 7500.00, '2018-09-03', '1111-6');
INSERT INTO Compra VALUES(1, 3312.00, '2018-09-02', '1111-6');
```

Povoando a relação Cartão

```
INSERT INTO Cartão VALUES('5278415299452338', 'MasterCard', 3, 'FELIPE NOAH
RIBEIRO');
INSERT INTO Cartão VALUES('5176533150137062', 'MasterCard', 4, 'YURI C E
MORAES');
INSERT INTO Cartão VALUES('5101417723962281', 'MasterCard', 3, 'MIGUEL
LEANDRO L MENDES');
INSERT INTO Cartão VALUES('4485078431133976', 'Visa', 2, 'LUCIA L M
RODRIGUES');
INSERT INTO Cartão VALUES('4539161021923976', 'Visa', 5, 'ISABELA ELIANE T
CARVALHO');
```

Povoando a relação VendaCartao

```
INSERT INTO VendaCartao VALUES('5278415299452338', 1);
INSERT INTO VendaCartao VALUES('5176533150137062', 3);
INSERT INTO VendaCartao VALUES('5101417723962281', 6);
INSERT INTO VendaCartao VALUES('4485078431133976', 7);
INSERT INTO VendaCartao VALUES('4539161021923976', 9);
```

Povoando a relação Fornecedor

```
INSERT INTO Fornecedor VALUES('14.218.835/0001-27', 'Johnson & Johnson
Comercio E Distribuicao Ltda.', '233', 'Nova Brunswick', ' Nova Jersey', '
Nova Jersey');
INSERT INTO Fornecedor VALUES('13.481.309/0192-13', 'RN COMÉRCIO
S.A.','132','Praçã central', 'Mossoró','Praçã central RN');
INSERT INTO Fornecedor VALUES('11.242.434/0231-34', 'M Dias Branco
Indústria e Comércio de Alimentos', '242','Av. Padre Cicero', 'Juazeiro do
Norte', 'Centro');
```

```
INSERT INTO Fornecedor VALUES('23.142.552/0023-12', 'Econômico
Supermercado','193','R. Abel Sobreira', 'Juazeiro do Norte', 'Santa
Tereza');
INSERT INTO Fornecedor VALUES('24.513.234/0153-23', 'Dw
Acessórios','374','Padre Nestor Sampaio', 'Juazeiro do Norte', 'Lagoa
Seca');
```

Povoando a relação TelefoneFornecedor

```
INSERT INTO TelefoneFornecedor VALUES('14.218.835/0001-27', '(88)3242-
6352');
INSERT INTO TelefoneFornecedor VALUES('14.218.835/0001-27', '(88)2425-
1242');
INSERT INTO TelefoneFornecedor VALUES('13.481.309/0192-13', '(88)7969-
2374');
INSERT INTO TelefoneFornecedor VALUES('11.242.434/0231-34', '(88)5274-
2583');
INSERT INTO TelefoneFornecedor VALUES('24.513.234/0153-23', '(88)2343-
7390');
INSERT INTO TelefoneFornecedor VALUES('24.513.234/0153-23','(88)2673-
7557');
```

Povoando a relação CompraFornecedor

```
INSERT INTO CompraFornecedor VALUES('14.218.835/0001-27', '4');
INSERT INTO CompraFornecedor VALUES('13.481.309/0192-13', '5');
INSERT INTO CompraFornecedor VALUES('11.242.434/0231-34', '2');
INSERT INTO CompraFornecedor VALUES('23.142.552/0023-12', '3');
INSERT INTO CompraFornecedor VALUES('24.513.234/0153-23', '1');
```

Povoando a relação Produto

```
INSERT INTO Produto VALUES('0242-1342', 6.99, 'SHAMPOO Infaltil - Johnson &
Johnson');
INSERT INTO Produto VALUES('0284-2365', 4.90, 'SABÃO DE COCO - RN');
INSERT INTO Produto VALUES('0364-3740', 6.90, 'ARROZ INTEGRAL - M Dias
Comércio');
INSERT INTO Produto VALUES('3470-4743', 5.39, 'MACARRÃO - Econômico
Supermercado');
INSERT INTO Produto VALUES('0374-3874', 21.99, 'BATOM MATE SOUL KISS - Dw
Acessórios');
INSERT INTO Produto VALUES('3729-4629', 25.00, 'BIFE - Friboi');
```

Povoando a relação FornecedorProduto

```
INSERT INTO FornecedorProduto VALUES('14.218.835/0001-27', '0242-1342');
INSERT INTO FornecedorProduto VALUES('13.481.309/0192-13', '0284-2365');
INSERT INTO FornecedorProduto VALUES('11.242.434/0231-34', '0364-3740');
INSERT INTO FornecedorProduto VALUES('23.142.552/0023-12', '3470-4743');
INSERT INTO FornecedorProduto VALUES('24.513.234/0153-23', '0374-3874');
```

Povoando a relação CompraProduto

```
INSERT INTO CompraProduto VALUES (5,'0284-2365');
INSERT INTO CompraProduto VALUES (4,'0242-1342');
INSERT INTO CompraProduto VALUES (3,'3470-4743');
INSERT INTO CompraProduto VALUES (2,'0364-3740');
INSERT INTO CompraProduto VALUES (1,'0374-3874');
```

Povoando a relação VendaProduto

```
INSERT INTO VendaProduto VALUES (1,'0374-3874',9);
INSERT INTO VendaProduto VALUES (2,'0364-3740',57);
INSERT INTO VendaProduto VALUES (3,'3470-4743',5);
INSERT INTO VendaProduto VALUES (4,'0364-3740',6);
INSERT INTO VendaProduto VALUES (5,'3470-4743',20);
INSERT INTO VendaProduto VALUES (6,'0284-2365',6);
INSERT INTO VendaProduto VALUES (7,'0242-1342',7);
INSERT INTO VendaProduto VALUES (8,'0374-3874',13);
INSERT INTO VendaProduto VALUES (9,'0242-1342',1);
INSERT INTO VendaProduto VALUES (10,'3729-4629',7);
INSERT INTO VendaProduto VALUES (11,'0284-2365',108);
```

Povoando a relação Estoque

```
INSERT INTO Estoque VALUES (1, 30, '2018-01-1', '2020-9-4', 3, '0242-
1342');
INSERT INTO Estoque VALUES (2, 15, '2018-3-4', '2021-4-2', 4, '0284-2365');
INSERT INTO Estoque VALUES (3, 60, '2017-11-11', '2019-12-12', 2, '0364-
3740');
INSERT INTO Estoque VALUES (4, 25, '2018-5-6', '2020-1-2', 5, '3470-4743');
```

```
INSERT INTO Estoque VALUES (5, 50, '2018-2-23', '2021-2-5', 2, '0374-
3874');
INSERT INTO Estoque VALUES (6, 4, '2017-5-28', '2029-6-3', 4, '3729-4629');
INSERT INTO Estoque VALUES (7, 4, '2017-1-21', '2031-5-7', 6, '3729-4629');
```

4.2 Criando índices

```
CREATE INDEX pessoaindice ON Pessoa(CPF);
CREATE INDEX clienteindice ON Cliente(CPF);
CREATE INDEX setorindice ON Setor(codigo);
CREATE INDEX funcionarioindice ON Funcionario(matricula);
CREATE INDEX gerenciasetorindice ON GerenciaSetor(gerente);
CREATE INDEX vendaindice ON Venda(codVenda);
CREATE INDEX clientevendaindice ON ClienteVenda(cpfCliente);
CREATE INDEX compraindice ON Compra(codigo);
CREATE INDEX cartaoindice ON Cartão(numero);
CREATE INDEX vendacartaoindice ON VendaCartao(numeroCartao);
CREATE INDEX fornecedorindice ON Fornecedor(CNPJ);
CREATE INDEX telefonefornecedorindice ON TelefoneFornecedor(telefone);
CREATE INDEX comprafornecedorindice ON CompraFornecedor(codCompra);
CREATE INDEX produtoindice ON Produto(codBarras);
CREATE INDEX fornecedorprodutoindice ON
FornecedorProduto(codBarrasProduto);
CREATE INDEX compraprodutoindice ON CompraProduto(codCompra);
CREATE INDEX vendaprodutoindice ON VendaProduto(codVenda);
CREATE INDEX estoqueindice ON Estoque(idLote);
```

4.3 Criando visões

1. Esta visão auxilia na recuperação dos produtos mais vendidos pelo supermercado ela retorna o código de barras, o nome do produto, a quantidade de produtos vendidos, a quantidade de vendas onde o produto estava presente e a quantidade do produto em estoque.

```
CREATE VIEW ProdutoInfo as

select codBarrasProduto, descricao, produtosVendidos, quantVendas,
quantProdutoEstoque

from(

   (select codbarrasproduto, sum(quantidadelotes * produtoslote) as
quantProdutoEstoque

   from estoque

   group by codbarrasproduto) tab1


   natural full join


   (select vp.codbarrasproduto, p.descricao, sum(quantidade) as
produtosVendidos, count(*) as quantVendas

   from vendaproduto vp join produto p on codbarrasproduto = codbarras

   group by vp.codbarrasproduto, p.descricao) tab2

   ) tab3

order by quantVendas desc, produtosVendidos desc, descricao
```

2. Esta visão retorna os nomes dos fornecedores e o nome dos respectivos produtos que os mesmo fornecem ao supermercado.

```
CREATE VIEW FornecedorProdutoNomes as

select nome, descricao

from (fornecedor f natural join fornecedorProduto fp), produto p

where p.codbarras = fp.codbarrasproduto
```

3. Esta visão retorna o nome do fornecedor e o telefone.

```
create view TelefoneFonecedorView as
select f.nome, tf.telefone
from fornecedor f natural join telefonefornecedor tf
order by f.nome
```

4.4 scripts SQL para consultas

1. Agrupar as vendas entre as datas 01/09/2018 e 15/09/2018.

```
select *
from venda
where data > '2018-09-01' and data < '2018-09-15'
```

2. Recuperar o código de barras, o nome e a quantidade, de todos os produtos em estoque, ordenada de forma crescente pela quantidade.

```
select qp.codbarras, qp.descricao, sum(QuantidadeProdutos) as quantidade
from(
select p.descricao, p.codbarras,quantidadelotes * produtoslote as
QuantidadeProdutos
from produto p join estoque e on p.codbarras = e.codBarrasProduto) as qp
group by qp.codbarras, qp.descricao
order by quantidade
```

3. Recuperar os produtos que tem sua data de validade entre 01/01/2019 e 25/10/2020 ordenados pela data como primeiro critério de ordenação e depois pela descrição do produto.

```
select descricao, datavalidade
from estoque e join produto p on e.codbarrasproduto = p.codbarras
where datavalidade >= '2019-01-01' and datavalidade <= '2020-10-25'
order by datavalidade, p.descricao asc
```

4. Recuperar a descrição e a data de fornecimento dos produtos que foram comprados para estoque e ordena-los pela data de fornecimento.

```
select p.descricao, e.datafornecimento
from produto p, estoque e
where p.codbarras = e.codbarrasproduto
order by e.datafornecimento
```

5. Recuperar os telefones do fornecedor Dw Acessórios.

```
select tf.telefone
from telefonefornecedor tf, fornecedor f
where tf.cnpj = f.cnpj and f.nome = 'Dw Acessórios'
```

6. Recuperar o nome e a matrícula do funcionário que ficou responsável pela compra do código 3 além do valor total da compra.

```
select p.nome, f.matricula, c.valor as valorCompra
from compra c, funcionario f, pessoa p
where c.matfuncionario = f.matricula and f.cpf = p.cpf and codigo = 3
```

7. Recuperar todos os funcionários que trabalham no setor de atendimento

```
select f.matricula, f.cpf, f.dataadmissao, f.salario
from setor s, funcionario f, pessoa p
where f.codsetor = s.codigo and f.cpf = p.cpf and s.nome = 'atendimento'
```

8. Recuperar o nome e a matricula do único funcionário que gerencia o setor de atendimento.

```
select f.matricula, p.nome
from ((funcionario f join pessoa p on p.cpf = f.cpf) join gerenciasetor gc
on f.matricula = gc.gerente) join setor s on gc.codsetor = s.codigo
where s.nome = 'atendimento' and datafim is null
```

9. Recuperar o salário e o nome de cada funcionário.

```
select p.nome, f.salario
from funcionario f join pessoa p on f.cpf = p.cpf
```

10. Recuperar o valor da soma de todas as vendas pagas no cartão entre 07/09/2018 e 11/09/2018.

```
select sum(valor) as total
from venda
where formapagamento = 'cartão' and data >= '2018-09-07' and data < '2018-
09-11'
```

11. Recuperar o valor e a descrição do produto 0364-3740.

```
select valorunid, descricao
from produto
```

```
where codbarras = '0364-3740'
```

12. Recuperar o nome e a matrícula de todos os funcionários que não gerenciam nenhum setor atualmente mas já foram gerentes.

```
select nome, matricula
from(
select *
from funcionario f
where not exists
(
select *
from gerenciasetor gs
where f.matricula = gs.gerente and datafim is null
)) as tab, pessoa p
where tab.cpf = p.cpf
```

13. Recuperar o nome do setor e a quantidade de funcionários que estão alocados em cada setor.

```
select s.nome, count(*) as qtdfuncionarios
from funcionario f join setor s on f.codsetor = s.codigo
group by s.codigo, s.nome
```

14. Recuperar os produtos que começam com a letra B.

```
select *
from produto
where descricao LIKE 'B%'
```

15. Recuperar o CPF, nome, salário e data de admissão de todos os funcionários que se chamam Bernardo.

```
select p.cpf, p.nome, f.salario, f.dataadmissao
from funcionario f join pessoa p on f.cpf = p.cpf
where nome LIKE 'Bernardo%'
```

16. Recuperar o CNPJ dos fornecedores que vendem “MACARRÃO - Econômico Supermercado” mas não vendem “BATOM MATE SOUL KISS - Dw Acessórios”.

```
(select fp.cnpj
from fornecedorproduto fp, produto p
where fp.codbarrasproduto = p.codbarras and p.descricao = 'MACARRÃO -
Econômico Supermercado'
```

```
except

select fp.cnpj
from fornecedorproduto fp, produto p
where fp.codbarrasproduto = p.codbarras and p.descricao = 'BATOM MATE SOUL
KISS - Dw Acessórios')
```

17. Recuperar o código de barras, descrição e a quantidade de todos os produtos vendidos entre 01/09/2018 e 30/09/2018, filtrando os produtos com mais de 30 unidades vendidas.

```
select codbarras, descricao, sum(quantidade) quantProdutos
from
(
select *
from venda v, vendaProduto vp, produto p
where v.codvenda = vp.codvenda and p.codbarras = vp.codbarrasproduto and
v.data > '2018-09-01' and v.data < '2018-09-30'
) v
group by codbarras, descricao
having sum(quantidade)>30
order by sum(quantidade) desc
```

18. Recuperar todos os produtos que não estão em estoque mas que estão na tabela produtos.

```
select *
from produto p
where not exists(
select *
from estoque e
where e.codbarrasproduto = p.codbarras
)
```

19. Recupere o código da venda que teve o valor total maior que 100 e que não foi paga nem no cartão nem no cheque.

```
(select codVenda
from venda v
where valor > 100
```

```
intersect
select codVenda
from venda v
where formapagamento = 'dinheiro')
```

4.5 Criando gatilhos

1. Este gatilho garante que não será alocado novos produtos com data de validade anterior ou igual a data atual.

```
create function verificadata()
    returns trigger as $$
    declare

    begin
        if new.datavalidade<=now() then return null;
            else return new;
        end if;
    end $$ language PLPGSQL;

create trigger verificaDataValidadeProduto before
insert or update on estoque
for each row
execute procedure verificadata();
```

2. Este gatilho garante que um funcionário nunca será alocado em um setor que não existe.

```
create function verificaCodigoSetor()
returns trigger as $$
declare
    codSetorMaior Funcionario.codSetor%type;
begin
    select into codSetorMaior max(codSetor) from Funcionario;
    if new.codSetor>codSetorMaior then return null;
        else return new;
    end if;

end
$$ language plpgsql;
```

```
create trigger verificaCodigoSetorFuncionario before
insert or update on Funcionario
for each row
execute procedure verificaCodigoSetor();
```

4.6 Criando procedimentos armazenados.

1. Este procedimento ajuda no controle financeiro do supermercado, retornando o valor total vendido por um funcionário.

```
create function vendaTotalFuncionario(CHARACTER(6))
returns real as $$
declare
    matFunc alias for $1;
    total real:=0;
begin
    select into total sum(valor)
    from venda
    where matfuncionario = matFunc
    group by matfuncionario;
    return total;
end
$$ language plpgsql;
```

2. Este procedimento ajuda no controle da gerencia do supermercado, informando quantos gerentes passaram por um setor informado.

```
create function funcionariosGerentes(integer)
returns integer as $$
declare
    codigosetor alias for $1;
    quantidade integer:=0;
begin
    select into quantidade count(gerente)
    from gerenciasetor g
    where g.codsetor=codsetor and datafim is not null
    group by g.codsetor;
    return quantidade + 1;
end
$$ language plpgsql;
```