SENAI – CENTRO DE TREINAMENTO DE TECNOLOGIA DA INFORMAÇÃO

CURSO TÉCNICO DE DESENVOLVIMENTO DE SISTEMAS

ANDRÉ SANTOS MARTINS DA SILVA
GABRIEL GOMES SOUSA
LEONARDO GUEDES GOMES JUNIOR
TARLEY JUNIO MOREIRA JESUS

HT-SIS-002-T-02

S.A BANCO DE DADOS – MYSQL
Loja de Lacticínios - ZéLáctea LTDA

BELO HORIZONTE
2022

SENAI - Centro de Treinamento da Tecnologia da Informação
André Santos Martins da Silva
Gabriel Gomes Sousa
Leonardo Guedes Gomes Junior
Tarley Junio Moreira Jesus

S.A BANCO DE DADOS – MYSQL

ZéLáctea – LTDA

Situação de Aprendizagem desenvolvida com intuito de estudo, prática e desenvolvimento das habilidades dos estudantes André, Gabriel, Leonardo e Tarley com o Banco de Dados desenvolvido no SGDB MySQL.

BELO HORIZONTE
2022

Indicie de Imagens

1. OBJETIVO.

A priori, tivemos a necessidade de desenvolver um banco de dados para a Loja de Laticínios ZéLáctea LTDA, que se encontrava tendo algumas tabelas de dados como requisições, tabelas essas quais eram:

- Clientes;
- Produtos;
- Vendedores;
- Vendas;
- Nota Fiscal;
- Feedback
- Informações Nutricionais dos Produtos;

Tendo então conhecimento sobre qual tipo de empresa lidariamos e sobre quais informações eram as principais a serem priorizadas, começamos então a realizar a criação do Banco de Dados pelo modelo conceitual, o qual utilizamos como software o brmodelo.

2. MODELO CONCEITUAL - BRMODELO.

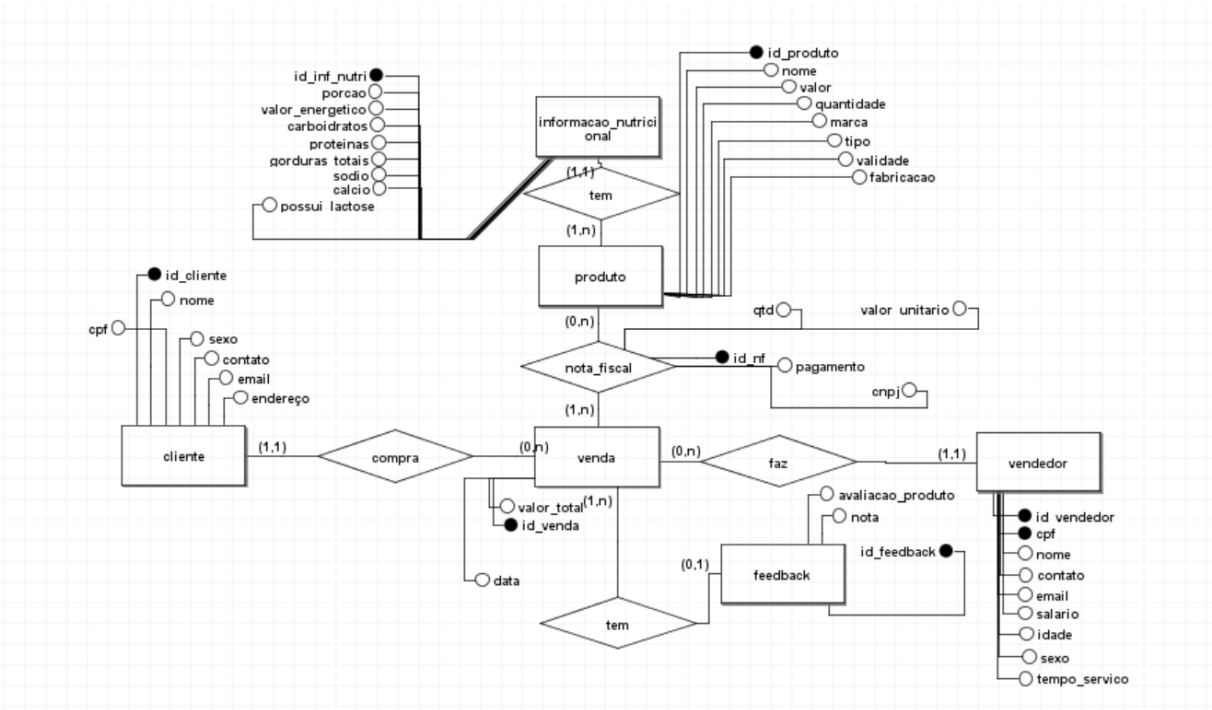

Figura 1

Pode-se notar que o Banco de Dados foi organizado tendo como centro dele a tabela de nota_fiscal e tendo como dois maiores opostos o feedback e a informação_nutricional, sendo essa as únicas duas tabelas que não dependem de nenhuma chave estrangeira, de nenhuma outra tabela. Sendo assim, vemos que a

tabela Produto, recebe a chave estrangeira da tabela informação_nutricional, e a tabela nota_fiscal recebe a chave estrangeira da tabela de produto e de venda, ligando então a venda, essa tabela recebe as chaves estrangeiras de cliente e vendedor, e feedback recebe a chave estrangeira de venda.

Agora com o raciocínio em mãos de como funcionará o Banco de Dados, mostra-se como venda se conecta com todo o “Setor Inferior” (Clientes, Feedback e Vendedor), Produto se conecta com Informacao_nutricional no “Setor Superior”, e a tabela de Nota_Fiscal realiza a junção destes 2 setores.

3. MODELO LÓGICO – BRMODELO.

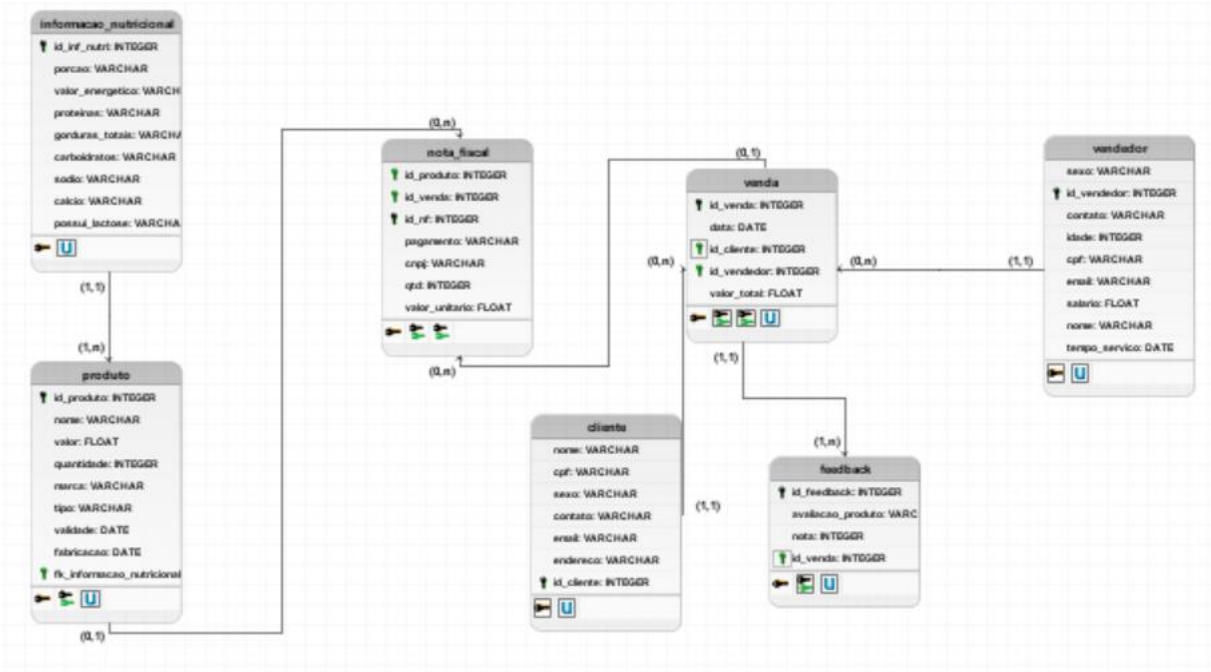

Figura 2

Agora com o modelo lógico do próprio brmodelo, podemos ter uma melhor noção de como irá funcionar o Banco de Dados, que agora foram definidos os parâmetros de cada tabela, com cada atributo sendo ele varchar, integer ou float, definidas também foram as chaves primárias, dados únicos e também as chaves estrangeiras que possibilitaram os futuros inner joins. O modelo que agora será refeito no SGBD Worckbench, será a base para o script que será importado para o MySQL e será executado pelo Xampp Php e irá mostrar como o banco de dados funciona por cmd.

4. MODELO LÓGICO CRIADO NO MySQL WORKBENCH.

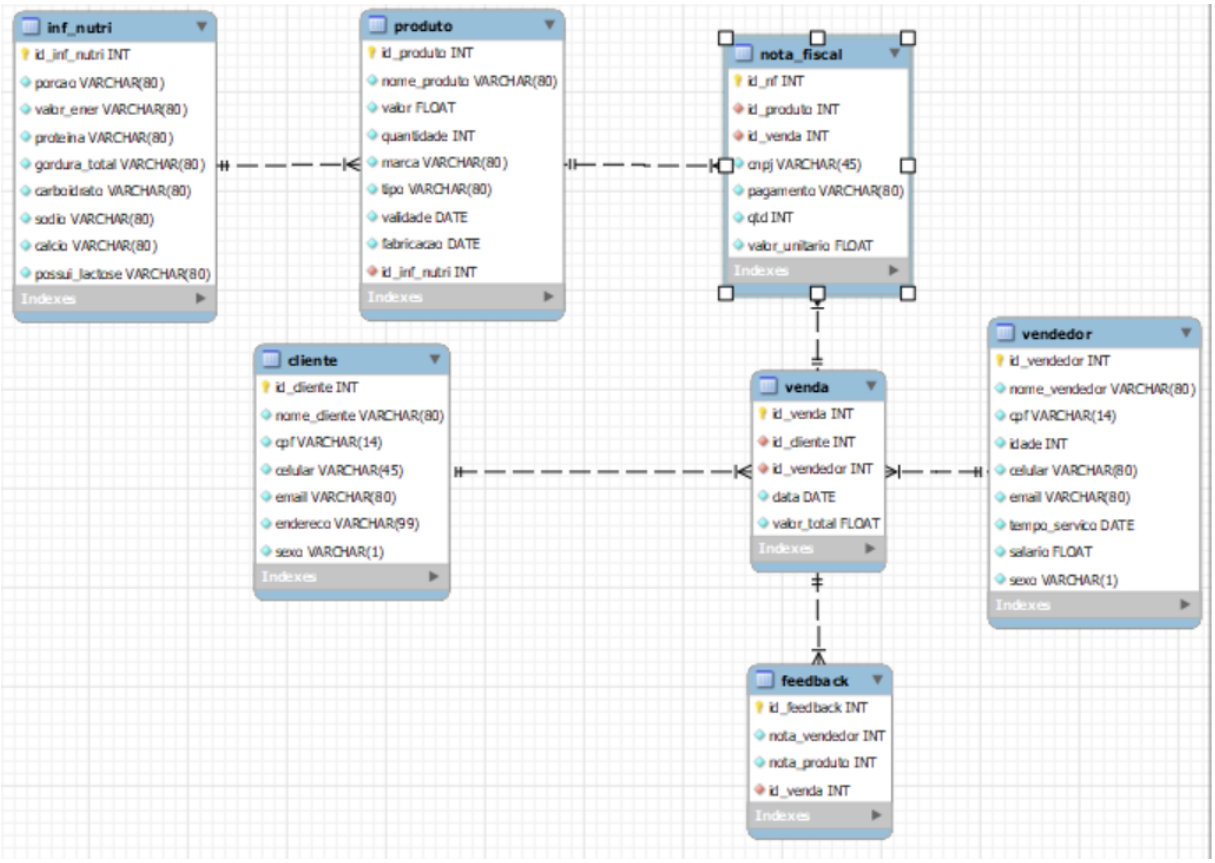

Figura 3

Modelo Lógico enfim produzido pelo MySQL, que é a base do script, todos dados são correspondentes aos dados criados no Modelo Lógico do brmodelo e no modelo conceitual. Podemos ver aqui claramente as chaves estrangeiras nas tabelas, as classificações como INT, varchar e float, também vemos as cardinalidades entre as tabelas.

5. SCRIPT PARA CRIAÇÃO DO BANCO DE DADOS.

A seguir mostrará o script exportado do Modelo Lógico anterior que fará a criação do banco de dados no Xampp Php:

```
-- Schema ZELACTEA

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `ZELACTEA` DEFAULT CHARACTER SET utf8
;
```

```
USE `ZELACTEA` ;


-- -----------------------------------------------------
-- Table `ZELACTEA`.`inf_nutri`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ZELACTEA`.`inf_nutri` (
  `id_inf_nutri` INT NOT NULL AUTO_INCREMENT,
  `porcao` VARCHAR(80) NOT NULL,
  `valor_ener` VARCHAR(80) NOT NULL,
  `proteina` VARCHAR(80) NOT NULL,
  `gordura_total` VARCHAR(80) NOT NULL,
  `carboidrato` VARCHAR(80) NOT NULL,
  `sodio` VARCHAR(80) NOT NULL,
  `calcio` VARCHAR(80) NOT NULL,
  `possui_lactose` VARCHAR(80) NOT NULL,
  PRIMARY KEY (`id_inf_nutri`),
  UNIQUE INDEX `id_inf_nutri_UNIQUE` (`id_inf_nutri` ASC) )
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `ZELACTEA`.`produto`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ZELACTEA`.`produto` (
  `id_produto` INT NOT NULL AUTO_INCREMENT,
  `nome_produto` VARCHAR(80) NOT NULL,
  `valor` FLOAT NOT NULL,
```

```
  `quantidade` INT NOT NULL,
  `marca` VARCHAR(80) NOT NULL,
  `tipo` VARCHAR(80) NOT NULL,
  `validade` DATE NOT NULL,
  `fabricacao` DATE NOT NULL,
  `id_inf_nutri` INT NOT NULL,
  PRIMARY KEY (`id_produto`),
  UNIQUE INDEX `id_produto_UNIQUE` (`id_produto` ASC) ,
  INDEX `fk_produto_inf_nutri1_idx` (`id_inf_nutri` ASC) ,
  CONSTRAINT `fk_produto_inf_nutri1`
    FOREIGN KEY (`id_inf_nutri`)
    REFERENCES `ZELACTEA`.`inf_nutri` (`id_inf_nutri`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `ZELACTEA`.`cliente`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ZELACTEA`.`cliente` (
  `id_cliente` INT NOT NULL AUTO_INCREMENT,
  `nome_cliente` VARCHAR(80) NOT NULL,
  `cpf` VARCHAR(14) NOT NULL,
  `celular` VARCHAR(45) NOT NULL,
  `email` VARCHAR(80) NOT NULL,
  `endereco` VARCHAR(99) NOT NULL,
```

```
  `sexo` VARCHAR(1) NOT NULL,
  PRIMARY KEY (`id_cliente`),
  UNIQUE INDEX `id_cliente_UNIQUE` (`id_cliente` ASC) ,
  UNIQUE INDEX `cpf_UNIQUE` (`cpf` ASC) )
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `ZELACTEA`.`vendedor`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ZELACTEA`.`vendedor` (
  `id_vendedor` INT NOT NULL AUTO_INCREMENT,
  `nome_vendedor` VARCHAR(80) NOT NULL,
  `cpf` VARCHAR(14) NOT NULL,
  `idade` INT NOT NULL,
  `celular` VARCHAR(80) NOT NULL,
  `email` VARCHAR(80) NOT NULL,
  `tempo_servico` DATE NOT NULL,
  `salario` FLOAT NOT NULL,
  `sexo` VARCHAR(1) NOT NULL,
  PRIMARY KEY (`id_vendedor`),
  UNIQUE INDEX `id_vendedor_UNIQUE` (`id_vendedor` ASC) ,
  UNIQUE INDEX `cpf_UNIQUE` (`cpf` ASC)
ENGINE = InnoDB;


-- -----------------------------------------------------
```

```
-- Table `ZELACTEA`.`venda`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ZELACTEA`.`venda` (
  `id_venda` INT NOT NULL AUTO_INCREMENT,
  `id_cliente` INT NOT NULL,
  `id_vendedor` INT NOT NULL,
  `data` DATE NOT NULL,
  `valor_total` FLOAT NOT NULL,
  PRIMARY KEY (`id_venda`),
  UNIQUE INDEX `id_venda_UNIQUE` (`id_venda` ASC) ,
  INDEX `fk_venda_cliente_idx` (`id_cliente` ASC) ,
  INDEX `fk_venda_vendedor1_idx` (`id_vendedor` ASC) ,
  CONSTRAINT `fk_venda_cliente`
    FOREIGN KEY (`id_cliente`)
    REFERENCES `ZELACTEA`.`cliente` (`id_cliente`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_venda_vendedor1`
    FOREIGN KEY (`id_vendedor`)
    REFERENCES `ZELACTEA`.`vendedor` (`id_vendedor`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `ZELACTEA`.`feedback`
```

```
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ZELACTEA`.`feedback` (
  `id_feedback` INT NOT NULL AUTO_INCREMENT,
  `nota_vendedor` INT NOT NULL,
  `nota_produto` INT NOT NULL,
  `id_venda` INT NOT NULL,
  PRIMARY KEY (`id_feedback`),
  UNIQUE INDEX `id_feedback_UNIQUE` (`id_feedback` ASC) ,
  INDEX `fk_feedback_venda1_idx` (`id_venda` ASC) ,
  CONSTRAINT `fk_feedback_venda1`
    FOREIGN KEY (`id_venda`)
    REFERENCES `ZELACTEA`.`venda` (`id_venda`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;



-- -----------------------------------------------------
-- Table `ZELACTEA`.`nota_fiscal`
-- -----------------------------------------------------
CREATE TABLE IF NOT EXISTS `ZELACTEA`.`nota_fiscal` (
  `id_nf` INT NOT NULL AUTO_INCREMENT,
  `id_produto` INT NOT NULL,
  `id_venda` INT NOT NULL,
  `cnpj` VARCHAR(45) NOT NULL,
  `pagamento` VARCHAR(80) NOT NULL,
  `qtd` INT NOT NULL,
```

```
  `valor_unitario` FLOAT NOT NULL,
  INDEX `fk_produto_has_venda_venda1_idx` (`id_venda` ASC) ,
  INDEX `fk_produto_has_venda_produto1_idx` (`id_produto` ASC) ,
  PRIMARY KEY (`id_nf`),
  UNIQUE INDEX `id_nf_UNIQUE` (`id_nf` ASC) ,
  CONSTRAINT `fk_produto_has_venda_produto1`
    FOREIGN KEY (`id_produto`)
    REFERENCES `ZELACTEA`.`produto` (`id_produto`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_produto_has_venda_venda1`
    FOREIGN KEY (`id_venda`)
    REFERENCES `ZELACTEA`.`venda` (`id_venda`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

6. COMANDOS DE INSERT INTO.

Para começarmos a encher a tabela, precisamos preencher ela em uma determinada ordem, já que algumas tabelas dependem de outras para serem preenchidas (as famosas chaves estrangeiras), então vamos para a ordem:

1. inf_nutri
2. produto
3. feedback
4. cliente
5. vendedor
6. venda
7. nota_fiscal

Para o preenchimento das tabelas foram necessários o uso de dados fictícios. As informações sobre os vendedores e clientes foram retiradas do site https://www.4devs.com.br/ gerador_de_pessoas acessado em 01/07/2022

Para o preenchimento das tabelas de produtos e inf_nutri foram retirados os dados do site https://www.italac.com.br/ acessado em 01/07/2022.

Segue abaixo alguns prints sobre o processo de criação e preenchimento de nossas tabelas:

```
insert into vendedor (nome_vendedor, cpf, idade, celular, email, tempo_servico, salario, sexo) values
            ('Letícia Giovanna Tatiane Santos', '951.499.179-66', 36, '(31) 99179-7004', 'leticia-santos81@hotmail.com', '2020-07-08', 200
            ('Fábio Ruan Luan Caldeira', '475.119.745-20', 35, '(31) 99358-1470', 'fabioruancaldeira@yahho.com.br', '2019-05-11', 1900, 'M
            ('Luan César Bernardo Pereira', '246.689.221-96', 26, '(31) 98833-6148', 'luan.cesar.pereira@liv.com', '2021-11-03', 2100, 'M'
            ('Bryan Iago Joaquim das Neves', '977.414.193-82', 57, '(31) 99918-9175', 'bryaniagodasneves@ufscar.br', '2012-02-09', 5000, '
            ('Fátima Rita Marcela Almeida', '793.079.396-67', 50, '(31) 98752-8876', 'fatima_rita_almeida@onvale.com', '2017-05-06', 3000,
            ('Giovanna Bárbara Mendes', '902.419.094-06', 39, '(31) 98195-0209', 'giovanna_mendes@msds.com.br', '2017-06-10', 2500, 'M'),
            ('Rodrigo Erick Barros', '378.825.782-23', 47, '(31) 98725-6535', 'rodrigo.erick.barros@ahlstrom.com', '2016-04-26', 3000, 'M'
            ('Milena Jennifer da Costa', '146.506.123-14', 24, '(31) 98500-7955', 'milena-dacosta78@solpro.biz', '2022-02-05', 1500, 'F'),
            ('Larissa Valentina Regina Corte Real', '898.059.687-15', 34, '(31) 98425-8290', 'larissa_cortereal@meteorus.com.br', '2018-06
            ('Alícia Elza Moreira', '321.996.367-63', 26, '(31) 99437-2316', 'aliciaelzamoreira@temp.com.br', '2020-05-15', 1700, 'F'),
            ('Bernardo Lorenzo Renato Martins', '612.187.526-70', 35, '(31) 99965-5065', 'bernardo.lorenzo.martins@mciimoveis.com.br', '20
            ('Marcela Eliane Fernanda Figueiredo', '101.819.714-10', 27, '(31) 99252-7959', 'marcela.eliane.figueiredo@ibest.com.br', '201
            ('Renata Amanda Freitas', '491.936.768-67', 29, '(31) 98348-7502', 'renata.amanda.freitas@fosj.unesp.br', '2018-07-12', 2000,
            ('Lúcia Marina Rebeca Carvalho', '104.446.474-73', 43, '(31) 99377-8514', 'luciamarinacarvalho@bn.com.br', '2016-04-15', 3000,
            ('Henrique Ruan Nunes', '753.013.523-69', 39, '(31) 98461-5348', 'henriqueruannunes@efetivaseguros.com.br', '2013-08-15', 3500

select buscar_vendedores(13) ;

insert into cliente (nome_cliente, cpf, celular, email, endereco, sexo) values
                    ('Renata Vanessa Jéssica Dias','292.012.226-61','(31) 98763-1502','renata-dias96@sociedadeweb.com.br','Rua Guaxupé,510
                    ('Henry Yago Theo da Luz','978.493.026-99','(31) 98127-9341','henry_yago_daluz@fundasa.com.br','Rua dos Fidelis, 510 -
                    ('Yuri Marcos Vinicius Bryan Baptista','617.472.836-65','(33) 99352-0237','yurimarcosbaptista@csjsistemas.com.br','Rua
                    ('Alícia Antonella Melo','440.127.526-85','(37) 98535-3744','alicia_melo@franciscofilho.adv.br','Rua das Rosas, 454 - D
                    ('Raimunda Márcia Gabriela Galvão','475.851.636-73','(38) 99756-4396','raimundamarciagalvao@peopleside.com.br','Rua Equ
```

Figura 4

Acima estão alguns insert into que popularam nosso Banco de Dados sobre vendedor e cliente, respectivamente.

```
insert into feedback (avaliacao_produto, nota) values ('Ótimo', 10),
('Bom', 8),
('Bom', 7),
('Ruim', 3),
('Péssimo', 1),
('Ok', 5),
('Bom', 7),
('Ótimo', 10),
('Ótimo', 9),
('Bom', 8),
('Ruim', 4),
('Ruim', 3),
('Ruim', 2),
('Péssimo', 1),
('Ótimo', 10),
('Bom', 7),
('Ruim', 4),
('Ok', 6),
('Bom', 8),
('Ótimo', 9),
('Ótimo', 10),
('Ótimo', 9),
('Bom', 8),
('Ok', 6),
```

Figura 5

Acima vemos o insert into para a tabela feedback.

Para sairmos da monótona rotina de prints sobre os insert into, partiremos para os comandos diferenciais, úteis para a Loja de Laticínios ZéLáctea LTDA e os mais utilizados.

7. COMANDOS.

Principais Comandos:

- select sum(qtd) as Total_de_Produtos_Vendidos from nota_fiscal;
- select v.data, c.nome_cliente, ve.nome_vendedor, p.nome_produto, p.valor, n.qtd
  from venda v inner join cliente c on v.id_cliente = c.id_cliente
  inner join vendedor ve on v.id_vendedor = ve.id_vendedor
  inner join feedback f on f.id_venda = v.id_venda
  inner join nota_fiscal n on n.id_venda = v.id_venda
  inner join produto p on n.id_produto = p.id_produto
  inner join inf_nutri i on p.id_inf_nutri = i.id_inf_nutri
  where data between '2022-03-11' and '2022-07-01';
- select * from dias_de_servico;
- select * from feedback_avaliacao;

- select * from status_produto where nome_produto like '%Leite%';
- select * from top_3;
- select * from top_10ve;
- select * from top_5c;
- call buscar_p(13);
- call comissao_por_mes(6,6);
- select nome_cliente as Nome, email as Email, celular as Celular, endereco as Endereço
  from cliente where nome_cliente like 'R%';
- select nome_cliente as Nome, email as Email, celular as Celular, endereco as Endereço
  from cliente where nome_cliente like 'S%';
- select nome_cliente as Nome, email as Email, celular as Celular, endereco as Endereço
  from cliente where celular like '(31)%';
- select nome_cliente as Nome, email as Email, celular as Celular, endereco as Endereço
  from cliente where endereco like '%Belo Horizonte%';
- select id_cliente, nome_cliente as Nome, email as Email, celular as Celular, endereco as Endereço
  from cliente where endereco not like '%Belo Horizonte%';
- select c.nome_cliente as Cliente, c.sexo as 'Sexo Cliente', ve.nome_vendedor as Vendedor, ve.sexo as 'Sexo Vendedor', f.nota_vendedor as 'Nota Vendedor', f.nota_produto as 'Nota Produto'
  from cliente c inner join venda v on v.id_cliente = c.id_cliente
  inner join vendedor ve on ve.id_vendedor = v.id_vendedor
  inner join feedback f on f.id_venda = v.id_venda;

Comandos mais úteis para a Loja:

- select v.data, c.nome_cliente, ve.nome_vendedor, p.nome_produto, p.valor, n.qtd
  from venda v inner join cliente c on v.id_cliente = c.id_cliente
  inner join vendedor ve on v.id_vendedor = ve.id_vendedor
  inner join feedback f on f.id_venda = v.id_venda
  inner join nota_fiscal n on n.id_venda = v.id_venda
  inner join produto p on n.id_produto = p.id_produto
  inner join inf_nutri i on p.id_inf_nutri = i.id_inf_nutri
  where data between '2022-03-11' and '2022-07-01';
- select * from dias_de_servico;
- select * from feedback_avaliacao;
- select * from top_10ve;

- call buscar_p(13);

Comandos mais utilizados durante a criação do Banco de Dados:

DROP DATABASE;

8. PROCEDIMENTOS.

- CREATE DEFINER=`root`@`localhost` PROCEDURE `buscar_p`(IN `a` INT)
  BEGIN
  select buscar_produtos(a) as Produto, buscar_produtos2(a) as 'Preço do Produto';
  END

- CREATE DEFINER=`root`@`localhost` PROCEDURE `caro_e_barato`()
  BEGIN
  select nome_produto as 'Prouto Mais Caro', valor as Valor from produto where valor = (select max(valor) from produto);
  select nome_produto as 'Prouto Mais Barato', valor as Valor from produto where valor = (select min(valor) from produto);

  END

- CREATE DEFINER=`root`@`localhost` PROCEDURE `cliente_mascfem`()
  BEGIN
  select count(sexo) as 'Clientes Homens' from cliente where sexo = 'M';
  select count(sexo) as 'Clientes Mulheres' from cliente where sexo = 'F';

  END

- CREATE DEFINER=`root`@`localhost` PROCEDURE `comissao_por_mes`(IN `cod` INT, IN `mes` INT)
  BEGIN
  declare totalv float;

  select sum(valor_total) into totalv from venda where id_vendedor = cod and month(data) = mes;

  select ve.nome_vendedor,
  case
  when totalv > 500 then round(totalv * 0.15, 2)
  when totalv > 250 then round(totalv * 0.10, 2)
  when totalv > 100 then round(totalv * 0.05, 2)

```
        else 'Sem comissão'
          end as Comissão_do_mês
    from vendedor as ve inner join venda as v on v.id_vendedor = ve.id_vendedor
    where ve.id_vendedor = cod and month(v.data) = mes limit 1;
    END
```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `compras_cliente`(IN `id` INT)
  BEGIN
  select c.id_cliente as id, c.nome_cliente as cliente, ve.nome_vendedor as vendedor,
  p.nome_produto as produto, n.qtd as quantidade, v.data, v.valor_total as total from inf_nutri i
  inner join produto p on i.id_inf_nutri = p.id_inf_nutri
  inner join nota_fiscal n on p.id_produto = n.id_produto
  inner join venda v on n.id_venda = v.id_venda
  inner join cliente c on v.id_cliente = c.id_cliente
  inner join vendedor ve on v.id_vendedor = ve.id_vendedor
  inner join feedback f on v.id_venda = f.id_venda
  where c.id_cliente = id;
  END
  ```
- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `contar_mascfem`()
  BEGIN
  select count(sexo) as Homens from vendedor where sexo = 'M';
  select count(sexo) as Mulheres from vendedor where sexo = 'F';

  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `euro_e_dolar`()
  BEGIN
  select nome_produto, format(valor, 2) as 'Real', round(valor/5.53, 2) as Euro, round(valor/5.23, 2) as Dólar from produto;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `mostrar_cliente`(IN `id` INT)
  BEGIN
  select * from cliente where id_cliente = id;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `mostrar_produto`(IN `id` INT)
  BEGIN
  ```
  ```

```
select * from produto where id_produto = id;
END
```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `mostrar_vendedor`(IN `id` INT)
  BEGIN
  select * from vendedor where id_vendedor = id;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `ordem_avaliacao`()
  BEGIN
  select p.nome_produto, round(avg(f.nota_produto), 2) as nota_media from inf_nutri i
  inner join produto p on i.id_inf_nutri = p.id_inf_nutri
  inner join nota_fiscal n on p.id_produto = n.id_produto
  inner join venda v on n.id_venda = v.id_venda
  inner join cliente c on v.id_cliente = c.id_cliente
  inner join vendedor ve on v.id_vendedor = ve.id_vendedor
  inner join feedback f on v.id_venda = f.id_venda
  group by f.id_venda order by nota_media desc;
  END
  ```
- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `qtd_cliente`()
  BEGIN
  select count(id_cliente) as 'Quantidade clientes' from cliente;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `qtd_estoque`()
  BEGIN
  select nome_produto as Nome, quantidade as Quantidade from produto;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `qtd_vendedor`()
  BEGIN
  select count(id_vendedor) as 'Quantidade vendedores' from vendedor;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `sexo`()
  BEGIN
  select nome_vendedor as Vendedor, email, sexo from vendedor where sexo = 'M'
  order by nome_vendedor asc;
  ```

```
select nome_vendedor as Vendedora, email, sexo from vendedor where sexo = 'F'
order by nome_vendedor asc;

END
```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `sexo_cliente`()
  BEGIN
  select nome_cliente as Homem, email, sexo from cliente where sexo = 'M' order by nome_cliente asc;
  select nome_cliente as Mulher, email, sexo from cliente where sexo = 'F' order by nome_cliente asc;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `soma_estoque`()
  BEGIN
  declare total int;
  select sum(quantidade) into total from produto;
  select total;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `vendas_vendedor`(IN `id` INT)
  BEGIN
  select ve.id_vendedor as id, ve.nome_vendedor as vendedor, c.nome_cliente as cliente,
  p.nome_produto as produto, n.qtd as quantidade, v.data, v.valor_total as total from inf_nutri i
  inner join produto p on i.id_inf_nutri = p.id_inf_nutri
  inner join nota_fiscal n on p.id_produto = n.id_produto
  inner join venda v on n.id_venda = v.id_venda
  inner join cliente c on v.id_cliente = c.id_cliente
  inner join vendedor ve on v.id_vendedor = ve.id_vendedor
  inner join feedback f on v.id_venda = f.id_venda
  where ve.id_vendedor = id;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `vendas_vendedor`(IN `id` INT)
  BEGIN
  select ve.id_vendedor as ID, ve.nome_vendedor as Vendedor, c.nome_cliente as Cliente,
  ```

```
p.nome_produto as 'Produto Vendido', n.qtd as Quantidade, v.data as Data,
v.valor_total as Total from inf_nutri i
inner join produto p on i.id_inf_nutri = p.id_inf_nutri
inner join nota_fiscal n on p.id_produto = n.id_produto
inner join venda v on n.id_venda = v.id_venda
inner join cliente c on v.id_cliente = c.id_cliente
inner join vendedor ve on v.id_vendedor = ve.id_vendedor
inner join feedback f on v.id_venda = f.id_venda
where ve.id_vendedor = id;
END
```

- ```
  CREATE DEFINER=`root`@`localhost` PROCEDURE `validade`()
  BEGIN
  select nome_produto as Produto, validade as Validade, marca as Marca,
          case
                  when datediff(validade, now()) <= 7 then 'Vender com urgência'
        when datediff(validade, now()) <= 15 then 'Ficar atento ao prazo'
        else 'Mais de uma quinzena até o vecimento'
          end as 'Tempo até vencer'
   from produto order by id_produto;
  END
  ```

9. VIEWS.

- ```
  CREATE
     ALGORITHM = UNDEFINED
     DEFINER = `root`@`localhost`
     SQL SECURITY DEFINER
  VIEW `zelactea`.`dias_de_servico` AS
     SELECT
        `zelactea`.`vendedor`.`nome_vendedor` AS `nome_vendedor`,
        TO_DAYS(CURRENT_TIMESTAMP())                                    -
  TO_DAYS(`zelactea`.`vendedor`.`tempo_servico`) AS `dias_de_servico`
     FROM
        `zelactea`.`vendedor`
  ```
- ```
  CREATE
     ALGORITHM = UNDEFINED
     DEFINER = `root`@`localhost`
     SQL SECURITY DEFINER
  VIEW `zelactea`.`feedback_avaliacao` AS
     SELECT DISTINCT
        `f`.`id_feedback` AS `ID`,
        `p`.`nome_produto` AS `Produto`,
  ```
  ```

```
        `f`.`nota_produto` AS `Nota_Produto`,
        `f`.`nota_vendedor` AS `Nota_Vendedor`,
        CASE
          WHEN (`f`.`nota_produto` + `f`.`nota_vendedor`) / 2 <= 1 THEN
    'Péssimo'
          WHEN
            (f.nota_produto + f.nota_vendedor) / 2 >= 2
              AND (f.nota_produto + f.nota_vendedor) / 2 <= 4
          THEN
            'Ruim'
          WHEN
            (f.nota_produto + f.nota_vendedor) / 2 >= 5
              AND (f.nota_produto + f.nota_vendedor) / 2 <= 6
          THEN
            'Ok'
          WHEN
            (f.nota_produto + f.nota_vendedor) / 2 >= 7
              AND (f.nota_produto + f.nota_vendedor) / 2 <= 8
          THEN
            'Bom'
          WHEN
            (f.nota_produto + f.nota_vendedor) / 2 >= 9
              AND (f.nota_produto + f.nota_vendedor) / 2 <= 10
          THEN
            'Ótimo'
        END AS avaliacao
      FROM
        ((((((zelactea.inf_nutri i
        JOIN zelactea.produto p ON (`i`.id_inf_nutri = `p`.id_inf_nutri))
        JOIN zelactea.nota_fiscal n ON (`p`.id_produto = `n`.id_produto))
        JOIN zelactea.venda v ON (`n`.id_venda = `v`.id_venda))
        JOIN zelactea.cliente c ON (`v`.id_cliente = `c`.id_cliente))
        JOIN zelactea.vendedor ve ON (`v`.id_vendedor = ve.id_vendedor))
        JOIN zelactea.feedback f ON (`v`.id_venda = `f`.id_venda))
```

- ```
  CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
  VIEW `zelactea`.`media_precos` AS
    SELECT
      FORMAT(SUM(`zelactea`.`produto`.`valor`)
  COUNT(`zelactea`.`produto`.`id_produto`),
  ```
  ```

```
        2,
        'de_DE') AS `media_precos`
    FROM
      `zelactea`.`produto`
```

- ```
  CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
  VIEW `zelactea`.`status_produto` AS
    SELECT
      `zelactea`.`produto`.`nome_produto` AS `nome_produto`,
      `zelactea`.`produto`.`quantidade` AS `quantidade`,
      CASE
        WHEN `zelactea`.`produto`.`quantidade` >= 50 THEN 'Estoque cheio'
        WHEN `zelactea`.`produto`.`quantidade` >= 15 THEN 'Estoque aceitável'
        WHEN zelactea.produto.quantidade >= 5 THEN 'Alerta de reposição de estoque'
        WHEN zelactea.produto.quantidade < 5 THEN 'Repor estoque urgente'
      END AS status_estoque
    FROM
      zelactea.produto
  ```

- ```
  CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
    SQL SECURITY DEFINER
  VIEW `zelactea`.`top_10ve` AS
    SELECT
      `ve`.`id_vendedor` AS `id_vendedor`,
      `ve`.`nome_vendedor` AS `nome_vendedor`,
      COUNT(0) AS `numero_de_vendas`
    FROM
      (`zelactea`.`vendedor` `ve`
      JOIN `zelactea`.`venda` `v` ON (`v`.`id_vendedor` = `ve`.`id_vendedor`))
    GROUP BY `v`.`id_vendedor`
    ORDER BY COUNT(0) DESC
    LIMIT 10
  ```

- ```
  CREATE
    ALGORITHM = UNDEFINED
    DEFINER = `root`@`localhost`
  ```
  ```

```
    SQL SECURITY DEFINER
VIEW `zelactea`.`top_3` AS
    SELECT
        `n`.`id_produto` AS `ID`,
        `p`.`nome_produto` AS `Nome`,
        SUM(`n`.`qtd`) AS `Total Vendido`
    FROM
        (`zelactea`.`nota_fiscal` `n`
        JOIN `zelactea`.`produto` `p` ON (`n`.`id_produto` = `p`.`id_produto`))
    GROUP BY `n`.`id_produto`
    ORDER BY SUM(`n`.`qtd`) DESC
    LIMIT 3
```

- ```
  CREATE
      ALGORITHM = UNDEFINED
      DEFINER = `root`@`localhost`
      SQL SECURITY DEFINER
  VIEW `zelactea`.`top_5c` AS
      SELECT
          `c`.`id_cliente` AS `ID`,
          `c`.`nome_cliente` AS `'Nome Cliente'`,
          COUNT(0) AS `numero_de_compras`
      FROM
          (`zelactea`.`cliente` `c`
          JOIN `zelactea`.`venda` `v` ON (`v`.`id_cliente` = `c`.`id_cliente`))
      GROUP BY `c`.`id_cliente`
      ORDER BY COUNT(0) DESC
      LIMIT 5
  ```

10. FUNCTIONS.

- ```
  CREATE DEFINER=`root`@`localhost` FUNCTION `buscar_clientes`(`id` INT)
  RETURNS varchar(80) CHARSET utf8
  BEGIN
  declare resposta varchar(80);
  declare qtd int(11);
  select count(*) into qtd from cliente
  where id = id_cliente;
  if qtd = 1 then
  select nome_cliente into resposta from cliente
  where id_cliente = id;
  else
  set resposta = 'Cliente não existe nesta base de dados.';
  ```

```
end if;
RETURN (resposta);
END
```

- ```
  CREATE DEFINER=`root`@`localhost` FUNCTION `buscar_produtos`(`id` INT) RETURNS varchar(80) CHARSET utf8
  BEGIN
  declare resposta varchar(80);
  declare qtd int(11);
  select count(*) into qtd from produto
  where id = id_produto;
  if qtd = 1 then
  select nome_produto into resposta from produto
  where id_produto = id;
  else
  set resposta = 'Produto não existe nesta base de dados.';
  end if;
  RETURN (resposta);
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` FUNCTION `buscar_produtos2`(`id` INT) RETURNS varchar(80) CHARSET utf8
  BEGIN
  declare resposta varchar(80);
  declare qtd int(11);
  select count(*) into qtd from produto
  where id = id_produto;
  if qtd = 1 then
  select valor into resposta from produto
  where id_produto = id;
  else
  set resposta = 'Produto não existe nesta base de dados.';
  end if;
  RETURN round((resposta), 2);
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` FUNCTION `buscar_vendedores`(`id` INT) RETURNS varchar(80) CHARSET utf8
  BEGIN
  declare resposta varchar(80);
  declare qtd int(11);
  select count(*) into qtd from vendedor
  where id = id_vendedor;
  ```
  ```

```
if qtd = 1 then
select nome_vendedor into resposta from vendedor
where id_vendedor = id;
else
set resposta = 'Vendedor não existe nesta base de dados.';
end if;
RETURN (resposta);
END
```

- ```
  CREATE DEFINER=`root`@`localhost` FUNCTION `saldo_mes`(mes int, lucro float) RETURNS float
  BEGIN
  declare valortotal_venda float;
  declare valortotal_salario float;

  select sum(valor_total) into valortotal_venda from venda where month(data) = mes;
  select sum(salario) into valortotal_salario from vendedor;

  set lucro = valortotal_venda - valortotal_salario;

  return lucro;
  END
  ```

- ```
  CREATE DEFINER=`root`@`localhost` FUNCTION `lucro_total`(lucro float) RETURNS int(11)
  BEGIN

  declare valortotal_venda float;
  declare valortotal_salario float;
  declare x date;

  select sum(valor_total) into valortotal_venda from venda;
  select sum(salario) into valortotal_salario from vendedor;

  select data into x from venda order by id_venda desc limit 1;

  set lucro = valortotal_venda - valortotal_salario * (month(x) - 2);

  return lucro;
  END
  ```

11. Gráficos gerados pelo Power B.I

1. Sexos Cliente e Vendedores

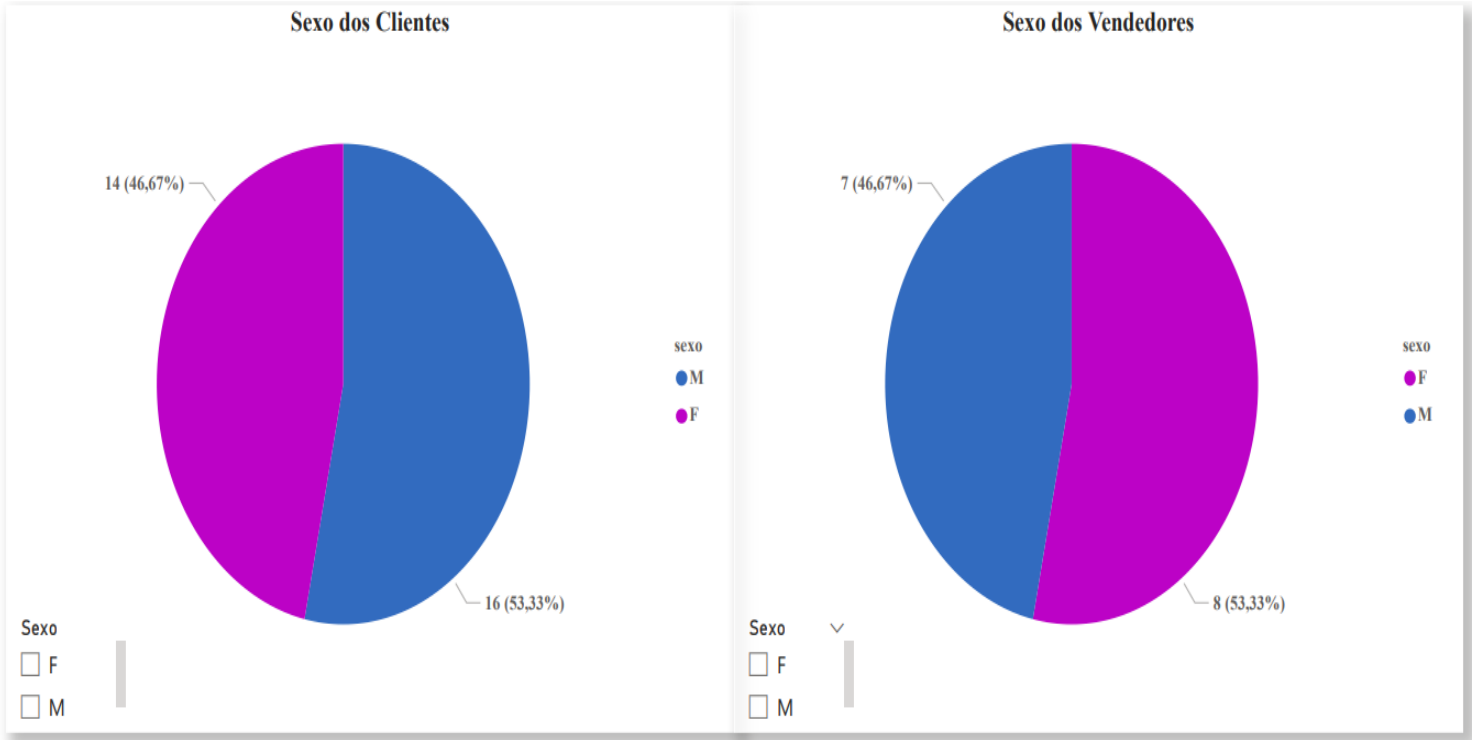

Figura 6

2. Idade dos Vendedores

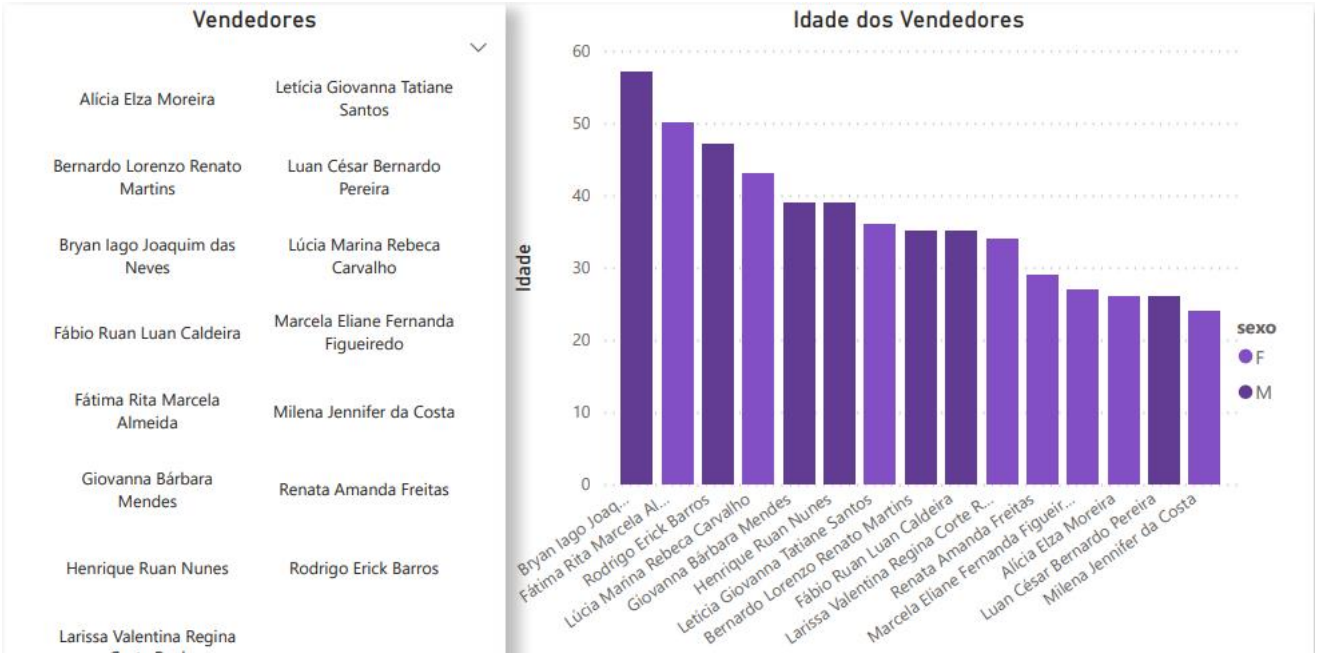

Figura 7

3. Informações dos Produtos

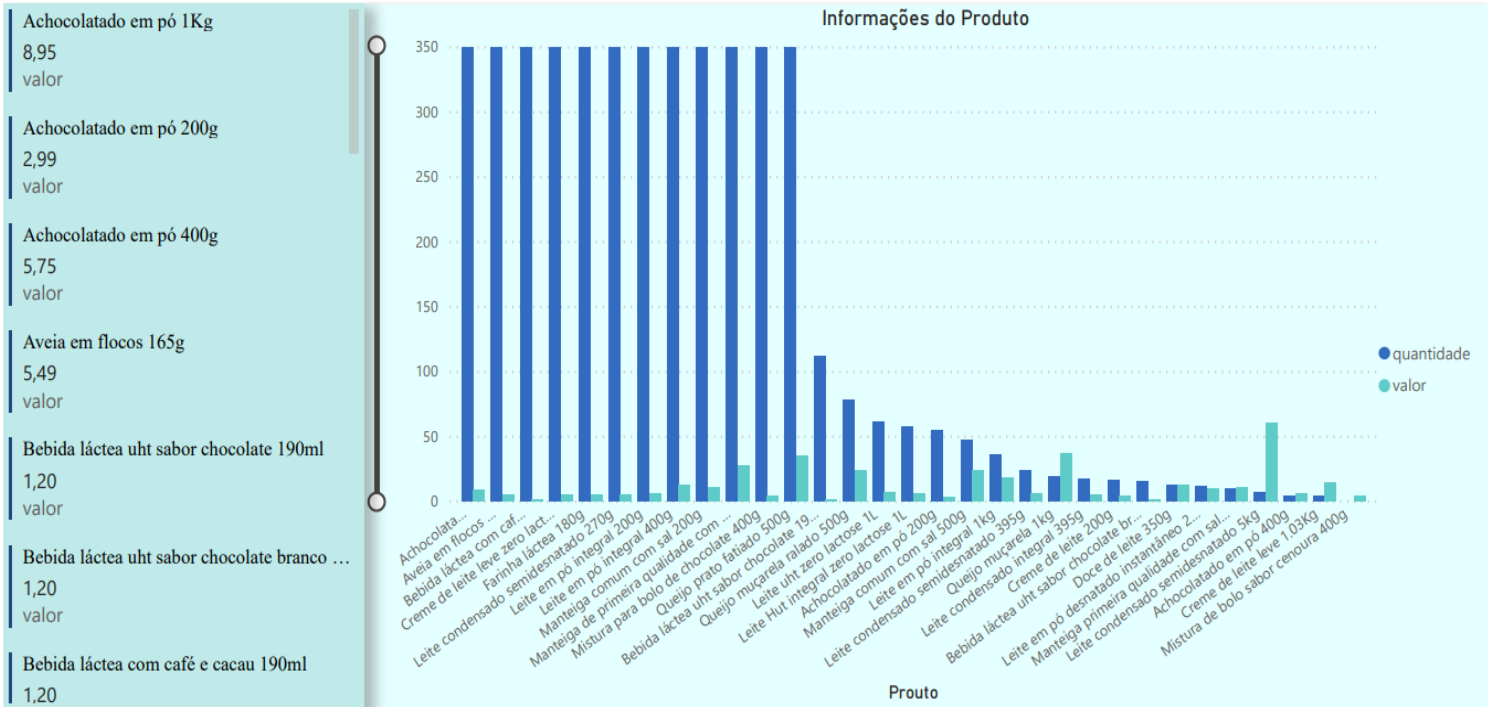

Figura 8

4. Notas das Vendas e dos Vendedores

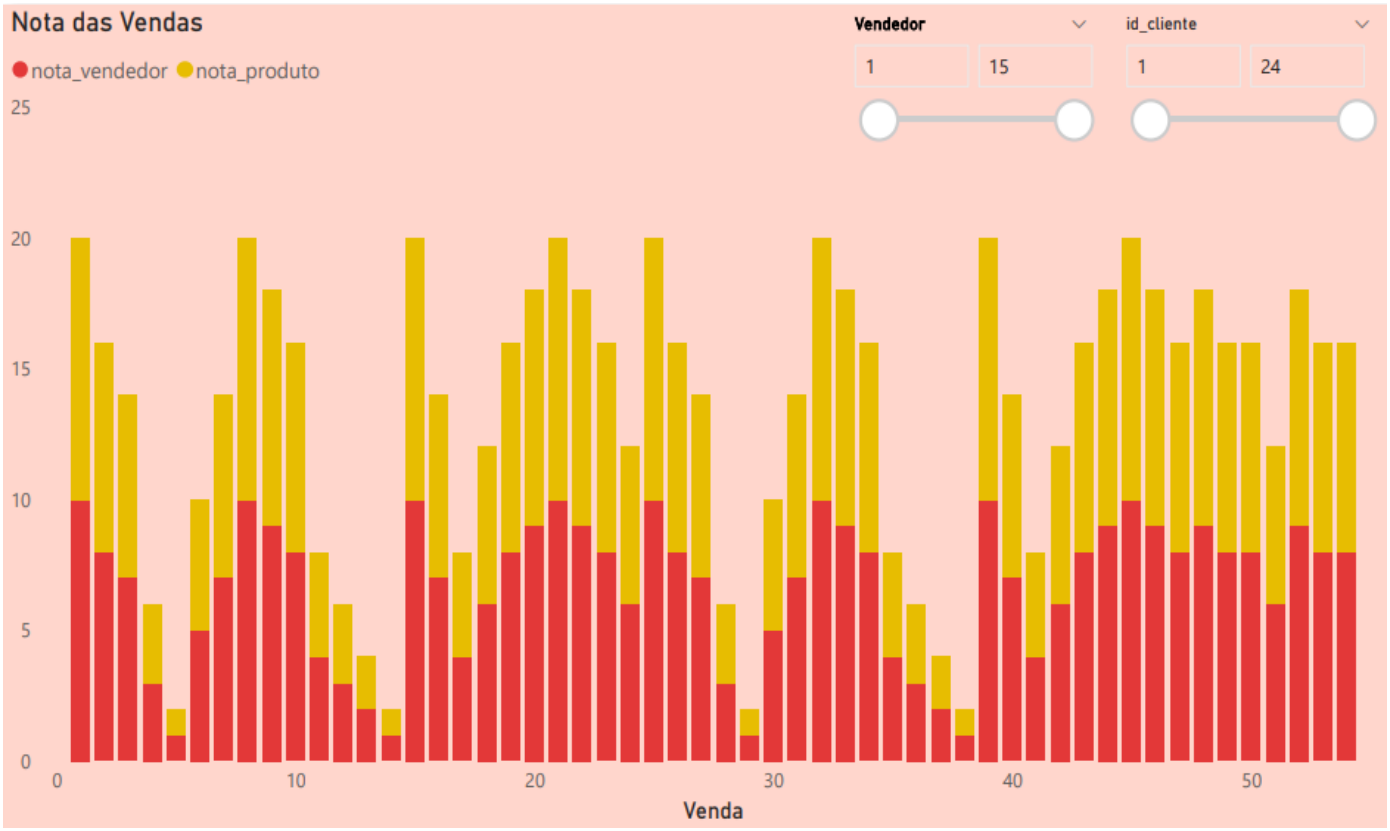

Figura 9

5. Valor Total das Vendas

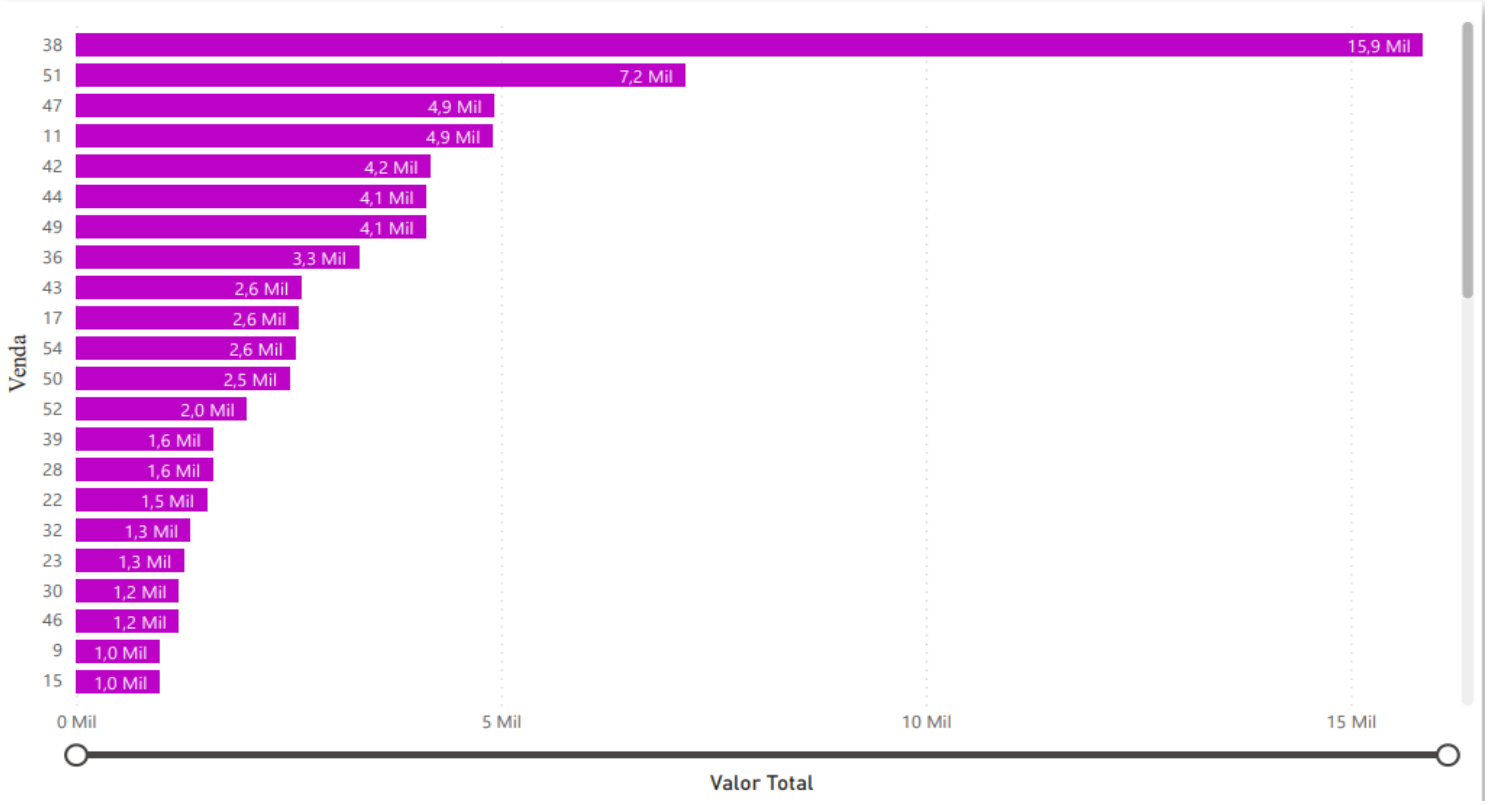

Figura 10

6. Saldo Mensal

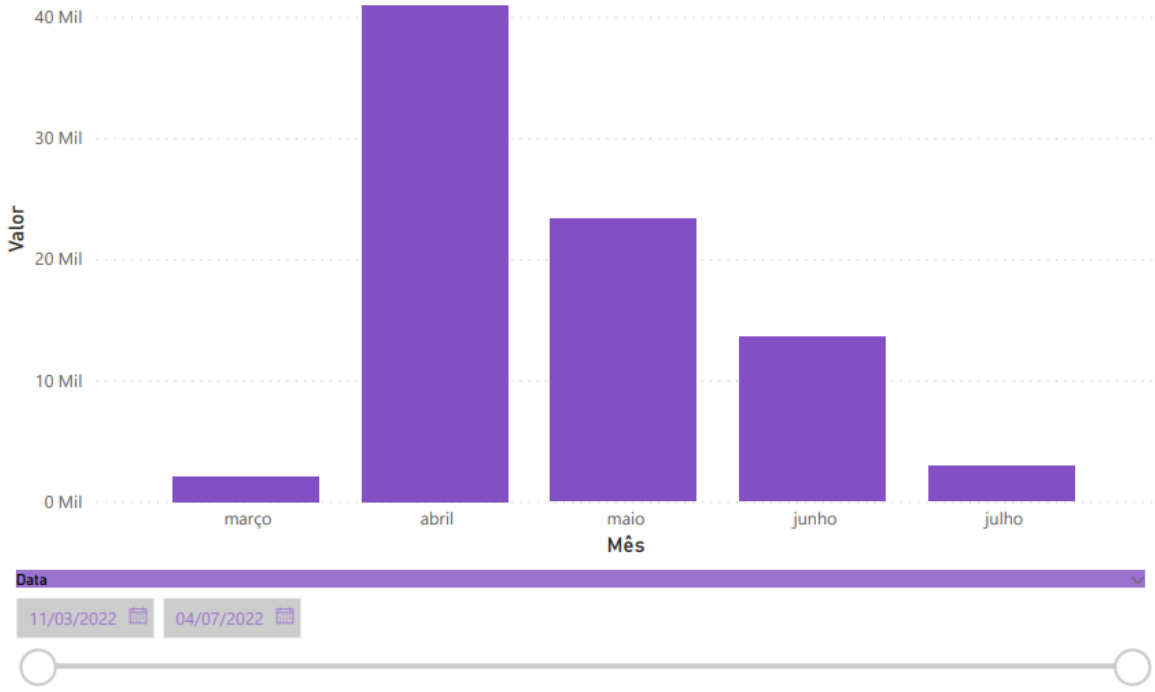

Figura 11

12. Referências

https://www.italac.com.br/ acessado em 01/07/2022;

https://www.4devs.com.br/ gerador_de_pessoas acessado em 01/07/2022;

brmodelo.jar utilizado para fazer o modelo conceitual e o primeiro modelo lógico;

MySQL Workbench foi utilizado para fazer o segundo modelo lógico e o script;

Comandos desenvolvidos por André, Gabriel, Leonardo e Tarley;

Gráficos criados pela ferramenta do Power B.I Desktop.