**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](mailto:0000877515@senaimgaluno.com.br)

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

[1. OBJETIVO. 4](#_Toc108044743)

[2. MODELO CONCEITUAL - BRMODELO. 4](#_Toc108044744)

[3. MODELO LÓGICO – BRMODELO. 5](#_Toc108044745)

[4. MODELO LÓGICO CRIADO NO MySQL WORKBENCH. 6](#_Toc108044746)

[5. SCRIPT PARA CRIAÇÃO DO BANCO DE DADOS. 6](#_Toc108044747)

[6. COMANDOS DE INSERT INTO. 12](#_Toc108044748)

[7. COMANDOS. 14](#_Toc108044749)

[8. PROCEDIMENTOS. 16](#_Toc108044750)

[9. VIEWS. 20](#_Toc108044751)

[10. FUNCTIONS. 23](#_Toc108044752)

[11. Gráficos gerados pelo Power B.I 26](#_Toc108044753)

[12. Referências 29](#_Toc108044760)

Indicie de Imagens

[Figura 1 4](#_Toc108044808)

[Figura 2 5](#_Toc108044809)

[Figura 3 6](#_Toc108044810)

[Figura 4 13](#_Toc108044811)

[Figura 5 14](#_Toc108044812)

[Figura 6 26](file:///C:\Users\leogu\Downloads\relatório.docx#_Toc108044813)

[Figura 7 26](file:///C:\Users\leogu\Downloads\relatório.docx#_Toc108044814)

[Figura 8 27](file:///C:\Users\leogu\Downloads\relatório.docx#_Toc108044815)

[Figura 9 27](file:///C:\Users\leogu\Downloads\relatório.docx#_Toc108044816)

[Figura 10 28](file:///C:\Users\leogu\Downloads\relatório.docx#_Toc108044817)

[Figura 11 28](file:///C:\Users\leogu\Downloads\relatório.docx#_Toc108044818)

# 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.

# MODELO CONCEITUAL - BRMODELO.

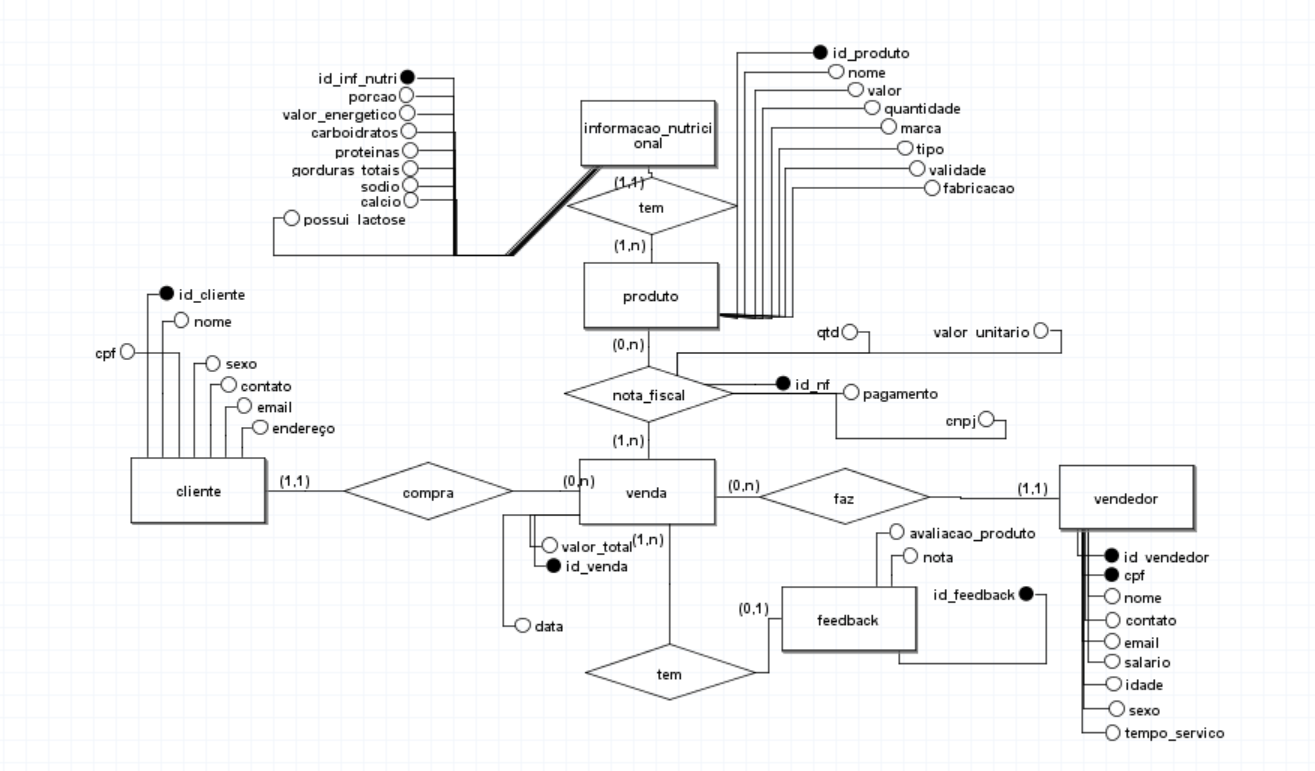


Figura 1

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.

# MODELO LÓGICO – BRMODELO.

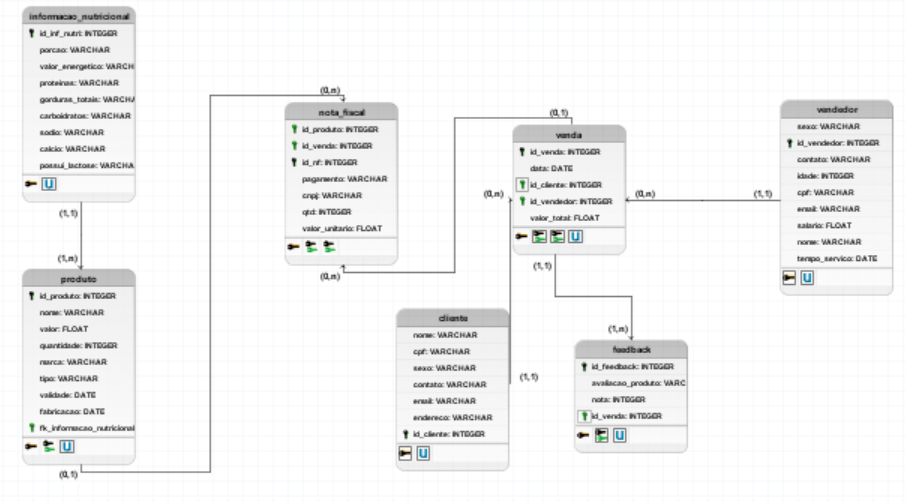


Figura 2

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.

# MODELO LÓGICO CRIADO NO MySQL WORKBENCH.

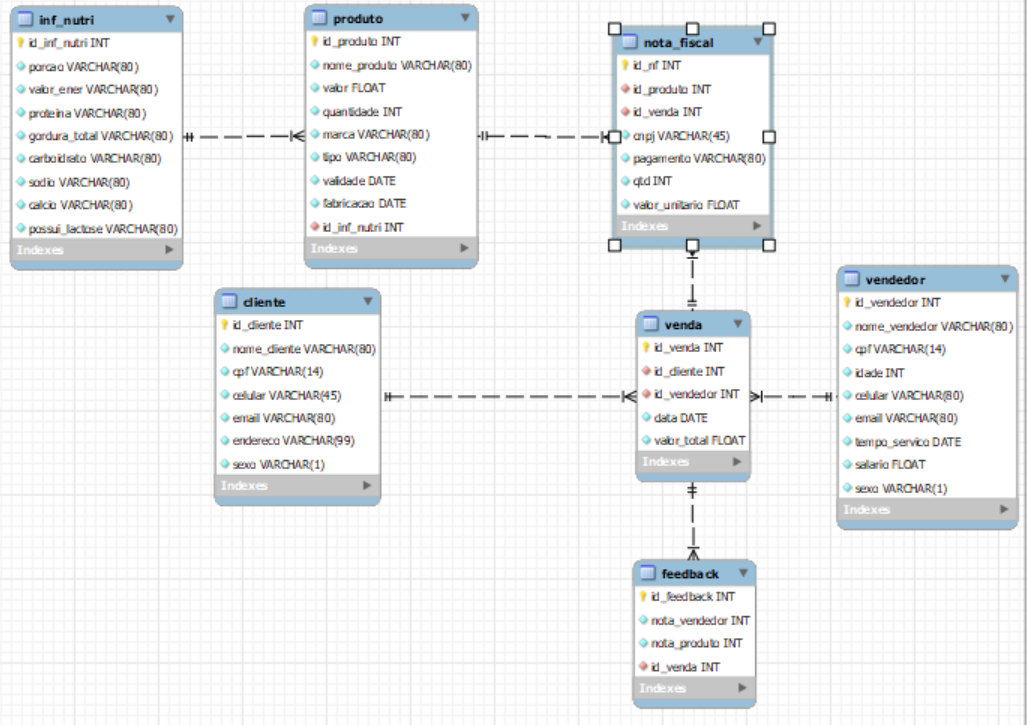


Figura 3

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.

# 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;

# 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](https://www.4devs.com.br/%20gerador_de_pessoas%20acessado%20em%2001/07/2022)

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:

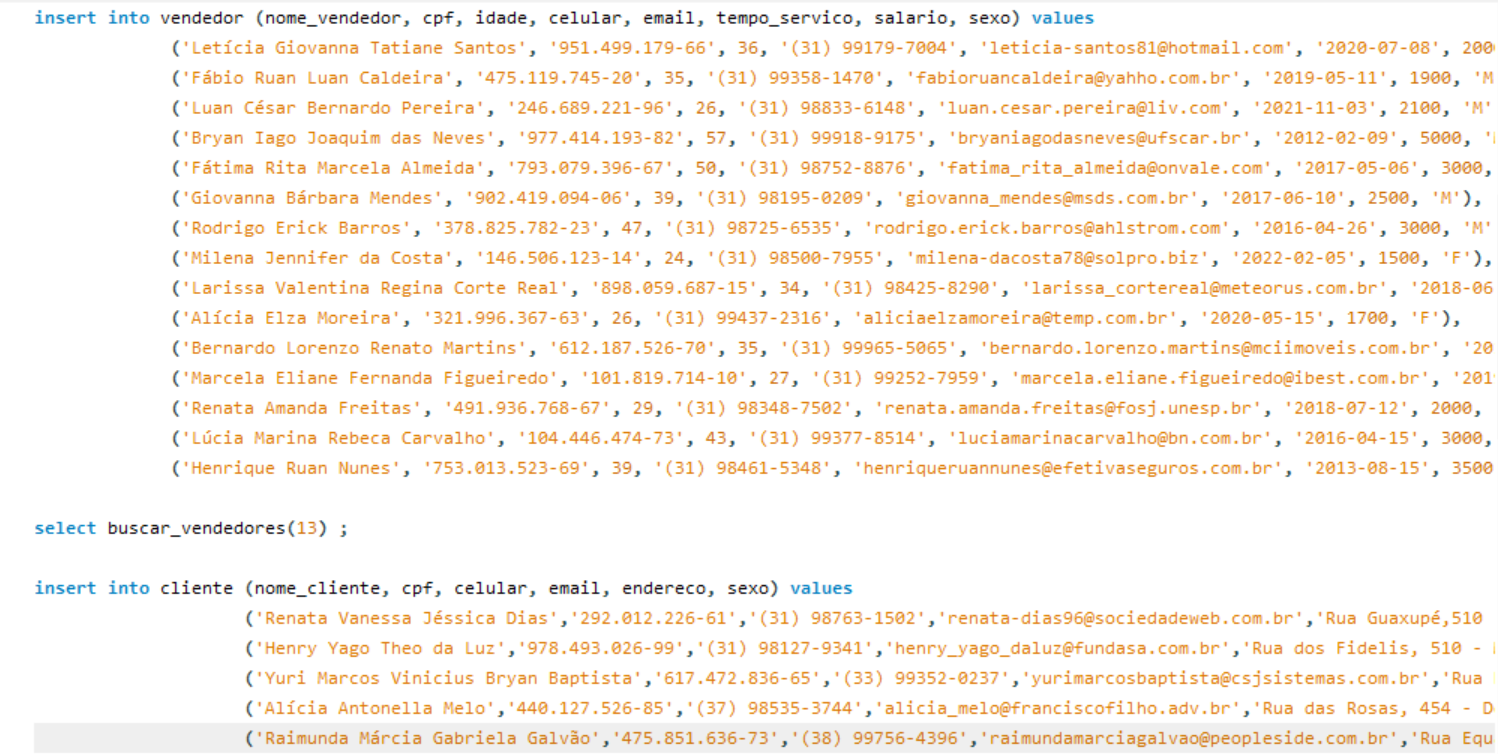


Figura 4

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

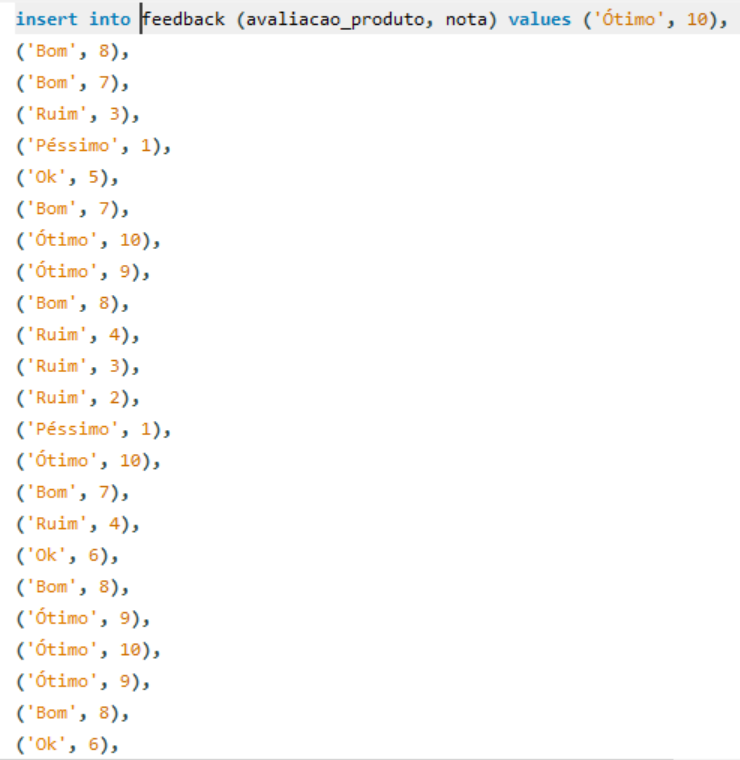


Figura 5

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.

# 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;**

# 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

# 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

# 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

# Gráficos gerados pelo Power B.I

## Sexos Cliente e Vendedores

Figura 6

## Idade dos Vendedores

Figura 7

## Informações dos Produtos

Figura 8

## Notas das Vendas e dos Vendedores

Figura 9

## Valor Total das Vendas

Figura 10

## Saldo Mensal

Figura 11

# Referências

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

[https://www.4devs.com.br/ gerador\_de\_pessoas acessado em 01/07/2022](https://www.4devs.com.br/%20gerador_de_pessoas%20acessado%20em%2001/07/2022);

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.