Departamento de Ciência da Computação
UFLA - Universidade Federal de Lavras
GCC214 - Introdução a Sistemas de Banco de Dados
Prof. Denilson Alves Pereira
Trabalho Prático – Etapa 1/3 – Modelo ER

Grupo 5 – Integrantes da turma 10A:

- Antônio Maurício Lanza de Sá e Melo Marques - 202020088
- David de Jesus Costa - 202020086
- Eduardo Dezena Gonçalves - 202020092
- Guilherme Grego Santos - 202020417
- Victor Gonçalves Lima - 202020775

Logística de Vendas

Descrição:

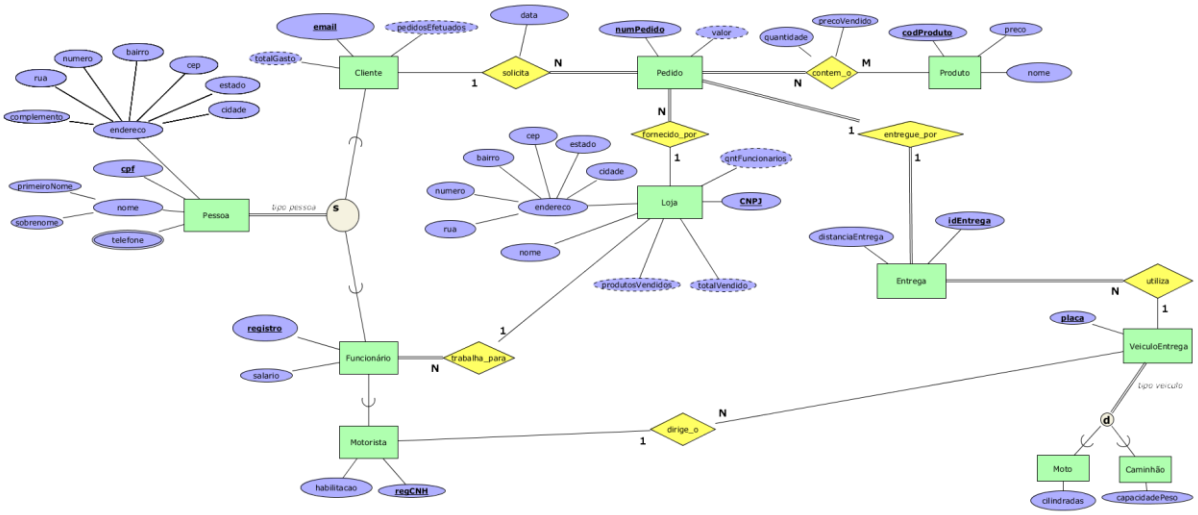
Levantamento de dados para uma empresa de tecnologia, denominada GroupFive, que oferece soluções de comércio eletrônico para pessoas físicas e jurídicas, sendo possível comprar, vender e enviar produtos.

A logística da empresa funciona com o cadastro de uma pessoa, que pode ser um cliente e/ou um funcionário de uma loja parceira, para uma pessoa é importante armazenar os atributos CPF, nome e endereço.

O cliente cadastrado pode solicitar um pedido de um produto fornecido por determinada loja, a quantidade do produto é estabelecida apenas quando o usuário faz o pedido, por isso o atributo quantidade é armazenado na relação entre pedido e produto.

Para o pedido, é importante saber o número do pedido; para a loja, o seu CNPJ; e para o produto, o seu código. Cada funcionário cadastrado está relacionado a somente uma loja, contudo uma loja parceira pode ter mais de um funcionário. Sempre que a GroupFive cadastrar um pedido, sua entrega vai ser controlada pela loja que o proveu. Cada entrega é referente a apenas um pedido, portanto é importante armazenar o ID da entrega. Um funcionário da empresa parceira pode pertencer à subclasse motorista, o qual será motorista de um veículo que será utilizado na entrega de um pedido. Na loja, o veículo utilizado na entrega é sempre um caminhão ou uma moto. É importante saber a placa do veículo para sua identificação na entrega do pedido.

Diagrama ER:



Dicionário de Dados:

Tipo Entidade	Pessoa		
Descrição	Conjunto de pessoas que apresentam funções específicas nas relações do levantamento.		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
cpf	Cadastro de pessoa física	Texto(11) Formato: ddd.ddd.ddd-dd	N
primeiroNome	Primeiro nome da pessoa	Texto(30)	N
sobrenome	Sobrenome da pessoa	Texto(70)	N
telefone	Telefones de contato da pessoa	Texto(11)	S
numero	Número da residência	Inteiro(4) positivo	N
rua	Nome da rua da residência	Texto(50)	N
bairro	Bairro da residência	Texto(30)	N
cidade	Cidade da residência	Texto(30)	N
estado	Sigla do estado da residência	Texto(2)	N
complemento	Informação adicional para encontrar a residência	Texto(30)	S
cep	Código de endereçamento postal	Texto(8) Formato: dd.ddd-ddd	N

Tipo Entidade	Cliente		
Descrição	Conjunto de clientes que fazem pedidos as lojas.		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
email	Endereço de e-mail para identificação do cliente	Texto(40)	N
totalGasto	Total gasto pelo cliente em seus pedidos, calculado a partir dos pedidos	Real(8,2) positivo	S
pedidosEfetuadas	Total de pedidos efetuados, calculado a partir dos pedidos	Inteiro(4) positivo	S

Tipo Entidade	Funcionário		
Descrição	Conjunto de funcionários que trabalham para uma lo a.		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
registro	Identificador do funcionário de uma loja	Texto(4) Formato: cddd	N
salario	Valor do salário de um funcionário	Real(6, 2)	N

Tipo Entidade	Motorista		
Descrição	Conjunto de motoristas que trabalham para uma loja e fazem a entrega de pedidos.		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
habilitacao	Categoria(s) habilita(s) para dirigir	Texto(2)	N
regCNH	Número de registro da Carteira Nacional de Habilitação	Texto(11) Formato: ddddddddddd	N

Tipo Entidade	Pedido
Descrição	Conjunto de pedidos que se relaciona com os produtos requisitados e suas quantidades.
Atributos	

Nome	Descrição	Domínio	Permite nulo? (S/N)
numPedido	Numero identificador do pedido	Inteiro(6) positivo	N
valor	Valor total do pedido, calculado a partir da soma dos preços de todos os produtos contidos no pedido	Real(7, 2) positivo	N

Tipo Entidade	Produto
Descrição	Conjunto de produtos que serão relacionados com os pedidos feitos por clientes.

Atributos

Nome	Descrição	Domínio	Permite nulo? (S/N)
nome	Nome do produto	Texto(30)	N
codProduto	Identificador do produto	Texto(14)	N
preco	Preço do produto	Real(6,2)	N

Tipo Entidade	Loja
Descrição	Conjunto de lojas responsáveis por fornecer pedidos aos clientes e controlar as entregas.

Atributos

Nome	Descrição	Domínio	Permite nulo? (S/N)
cnpj	Cadastro nacional de pessoa jurídica	Texto(14) Formato: dd.ddd.ddd/000ddd	N
nome	Nome da loja	Texto(50)	N
qntFuncionario	Quantidade de funcionários que uma loja tem contratada, valor obtido a partida da quantidade de vezes que a tabela funcionário faz referência a loja	Inteiro(4) positivo	N
produtosVendidos	Total de produtos vendidos pela loja, calculado a partir da soma da quantidade de produtos fornecidos para os pedidos que referenciam a loja	Inteiro(9) positivo	S
totalVendido	Valor total vendido pela loja, calculado a partir da soma dos valores dos pedidos que fazem referências a loja	Real(12,2) positivo	S
numero	Número da loja	Inteiro(4) positivo	N
rua	Nome da rua da loja	Texto(50)	N
bairro	Bairro da loja	Texto(30)	S
cidade	Cidade da loja	Texto(30)	N
estado	Sigla do estado da loja	Texto(2)	N

cep	Código de endereçamento postal	Texto(8) Formato: dd.ddd-ddd	N

Tipo Entidade	Entrega		
Descrição	Conjunto de pedidos já prontos para serem entregues.		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
codEntrega	Identificador da entrega	Texto(10) Formato: ddddddddd	N
distanciaEntrega	Distância em quilômetros da entrega do produto ao cliente	Real(4,1) positivo	N

Tipo Entidade	VeiculoEntrega		
Descrição	Conjunto de veículos utilizados para efetuar as entregas.		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
placa	Placa do veículo	Texto(7) Formato: ccccccc	N

Tipo Entidade	Moto		
Descrição	Conjunto de motos utilizadas para efetuar as entregas		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
cilindradas	Capacidade do motor da moto	Inteiro(4) positivo	N

Tipo Entidade	Caminhão

Descrição	Conjunto de caminhões utilizados para efetuar as entregas.		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
capacidadePeso	Capacidade para carregar produtos em tonelada	Real(2,1) positivo	N

Tipo Relacionamento	solicita		
Descrição	Indica que um cliente fez um pedido de um produto em uma determinada loja		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
data	Data da realização do pedido	Texto(8) Formato: dd/dd/dddd	N

Tipo Relacionamento	contem_o		
Descrição	Indica qual produto está alocado no pedido.		
Atributos			
Nome	**Descrição**	**Domínio**	**Permite nulo? (S/N)**
quantidade	Quantidade de produtos no pedido	Inteiro(5) positivo	N

Tipo Relacionamento	solicita
Descrição	Indica o pedido que o cliente solicitou.

Tipo Relacionamento	entregue_por
Descrição	Indica que o pedido é separado para entrega.

Tipo Relacionamento	utiliza
Descrição	Indica o meio utilizado para entrega.

Tipo Relacionamento	trabalha_para
Descrição	Indica a loja que o funcionário trabalha.

Tipo Relacionamento	dirige_o
Descrição	Indica o veículo que um motorista dirige.

Diagrama Relacional

Dicionário de Dados - Relacional

Tabela	Pessoa
Descrição	Conjunto de pessoas que apresentam funções específicas nas relações do levantamento

Atributos

Nome	Descrição
idPessoa	Código identificador de pessoa. Autoincrementável.
cpf	Cadastro de pessoa física.
primeiroNome	Primeiro nome de pessoa.
sobrenome	Sobrenome de pessoa.
cidade	Nome da cidade.
estado	Sigla do estado.
cep	Código de endereçamento postal.
bairro	Nome do bairro.
numero	Número da residência.
rua	Nome da rua.
complemento	Complemento da residência.

Tabela	Telefone

Descrição	Telefones de uma pessoa
Atributos	
Nome	**Descrição**
idPessoa	Referência ao id da pessoa que o telefone pertence.
fone	Número do telefone.

Tabela	Cliente
Descrição	Conjunto de pessoas que efetuam compras na GroupFive.
Atributos	
Nome	**Descrição**
idCliente	Referência ao id da pessoa que efetuou uma compra
email	E-mail do Cliente
totalGasto	Valor total gasto do cliente em compras
pedidosEfetuados	Número total de pedidos feitos pelo cliente

Tabela	Funcionario

Descrição	Conjunto de pessoas que trabalham em lojas cadastradas
Atributos	
Nome	**Descrição**
idFuncionario	Referência ao id da pessoa que trabalha em uma loja cadastrada
registro	Número de registro
salario	Salário mensal
idLoja	Referência ao id da loja que o funcionário trabalha

Tabela	Motorista
Descrição	Conjunto de funcionários que são motoristas
Atributos	
Nome	**Descrição**
idMotorista	Referência ao id do funcionário que trabalha como motorista
regCNH	Número de registro da carteira nacional de habilitação
habilitação	Tipo da cnh

Tabela	Loja

Descrição	Conjunto de lojas cadastradas
Atributos	
Nome	**Descrição**
idLoja	Código identificador de loja. Autoincrementável.
cnpj	Cadastro de pessoa jurídica.
nome	Nome da loja.
cep	Código de endereçamento postal.
rua	Nome da rua.
numero	Número da loja.
bairro	Nome do bairro.
cidade	Nome da cidade.
estado	Sigla do estado.
qntFuncionarios	Quantidade de funcionários.
totalVendidos	Valor total vendido pela loja.
produtosVendidos	Quantidade de produtos vendidos pela loja.

Tabela	VeiculoEntrega
Descrição	Conjunto de veículos utilizados para entrega

Atributos	
Nome	**Descrição**
idVeiculo	Código identificador de veículo. Autoincrementável.
placa	Placa do veículo.
idMotorista	Referência ao id do motorista que dirige o veículo.
tipoVeiculo	Tipo do veículo utilizado para a entrega.
cilindradas	Cilindradas da moto.
capacidadePeso	Peso máximo de carga que um caminhão suporta

Tabela	Produto
Descrição	Conjuntos de produtos
Atributos	
Nome	**Descrição**
codProduto	Código do produto.
preço	Preço do produto.
nome	Nome do produto.

Tabela	Pedido

Descrição	Dados de um pedido
Atributos	
Nome	**Descrição**
numPedido	Número do pedido.
idCliente	Referência ao id do cliente que efetuou o pedido.
idLoja	Referência ao id da loja que forneceu os produtos.
codEntrega	Código da entrega do pedido.
idVeiculo	Referência ao id do veículo utilizado para entrega.
valor	Valor total do pedido.
data	Data em que o pedido foi efetuado.
distanciaEntrega	Distância que o veículo percorrerá para completar a entrega.

Tabela	Produtos-Pedido

Descrição	Conjuntos de produtos pedidos no pedido
Atributos	
Nome	**Descrição**
numPedido	Referência ao número do pedido que os produtos pertence.
codProduto	Referência ao código do produto que o pedido contém.
quantidade	Quantidade de produtos.
preçoVendido	Preço final dos Produtos.

Etapa 3 – Implementação em SQL

(a) Criação de todas as tabelas e de todas as restrições de integridade. Todas as restrições de chave (PRIMARY KEY) e de integridade referencial (FOREIGN KEY) devem ser criadas.

```
-- Criação e utilização do esquema --

CREATE SCHEMA LogisticaVendas;

USE LogisticaVendas;

-- Criação das tabelas e restrições --

CREATE TABLE Pessoa (
    idPessoa INT NOT NULL AUTO_INCREMENT,
    cpf CHAR(11) NOT NULL,
    primeiroNome VARCHAR(30) NOT NULL,
    sobrenome VARCHAR(70) NOT NULL,
    cidade VARCHAR(30) NOT NULL,
    estado CHAR(2) NOT NULL,
    cep CHAR(8) NOT NULL,
    bairro VARCHAR(30) NULL,
    rua VARCHAR(50) NOT NULL,
    numero INT UNSIGNED NOT NULL,
    complemento VARCHAR(30) NULL,
    PRIMARY KEY (idPessoa),
    UNIQUE INDEX cpf_UNIQUE (cpf ASC) VISIBLE
);

CREATE TABLE Telefone (
    idPessoa INT NOT NULL,
    fone VARCHAR(11) NOT NULL,
    PRIMARY KEY (idPessoa, fone),
    CONSTRAINT fk_Pessoa
            FOREIGN KEY (idPessoa)
            REFERENCES Pessoa (idPessoa)
            ON DELETE CASCADE
            ON UPDATE CASCADE
);

CREATE TABLE Cliente (
    idCliente INT NOT NULL,
    email VARCHAR(40) NOT NULL,
    pedidosEfetuados INT UNSIGNED NOT NULL DEFAULT 0,
    totalGasto DECIMAL(8,2) NOT NULL DEFAULT 0.00,
    PRIMARY KEY (idCliente),
    UNIQUE INDEX email_UNIQUE (email ASC) VISIBLE,
    CONSTRAINT fk_ClientePessoa
            FOREIGN KEY (idCliente)
            REFERENCES Pessoa (idPessoa)
            ON DELETE CASCADE
            ON UPDATE CASCADE
);

CREATE TABLE Loja (
    idLoja INT NOT NULL,
```

```
    cnpj CHAR(14) NOT NULL,
    nome VARCHAR(50) NOT NULL,
    cep CHAR(8) NOT NULL,
    rua VARCHAR(50) NOT NULL,
    numero INT UNSIGNED NOT NULL,
    bairro VARCHAR(30) NULL,
    cidade VARCHAR(30) NOT NULL,
    estado CHAR(2) NOT NULL,
    qtdFuncionarios INT NULL DEFAULT 0,
    totalVendido DECIMAL(12,2) NULL DEFAULT 0.00,
    produtosVendidos INT NULL DEFAULT 0,
    PRIMARY KEY (idLoja),
    UNIQUE INDEX cnpj_UNIQUE (cnpj ASC) VISIBLE
);

CREATE TABLE Funcionario (
    idFuncionario INT NOT NULL,
    regFuncionario VARCHAR(4) NOT NULL,
    salario DECIMAL(6,2) NOT NULL,
    idLoja INT NOT NULL,
    PRIMARY KEY (idFuncionario),
    UNIQUE INDEX registro_UNIQUE (regFuncionario ASC) VISIBLE,
    CONSTRAINT idFuncionario
        FOREIGN KEY (idFuncionario)
        REFERENCES Pessoa (idPessoa)
        ON DELETE RESTRICT
        ON UPDATE RESTRICT,
    CONSTRAINT idLoja
        FOREIGN KEY (idLoja)
        REFERENCES Loja (idLoja)
        ON DELETE RESTRICT
        ON UPDATE RESTRICT
);

CREATE TABLE Motorista (
    idMotorista INT NOT NULL,
    regCNH CHAR(11) NOT NULL,
    habilitacao VARCHAR(2) NOT NULL,
    PRIMARY KEY (idMotorista),
    UNIQUE INDEX regCNH_UNIQUE (regCNH ASC) VISIBLE,
    CONSTRAINT idMotorista
        FOREIGN KEY (idMotorista)
        REFERENCES Funcionario (idFuncionario)
        ON DELETE RESTRICT
        ON UPDATE RESTRICT
);

CREATE TABLE VeiculoEntrega (
    idVeiculo INT NOT NULL,
```

```
    idMotorista INT NOT NULL,
    placa VARCHAR(7) NOT NULL,
    tipoVeiculo VARCHAR(15) NOT NULL,
    cilindradas INT NULL,
    capacidadePeso DECIMAL(2,1) NULL,
    PRIMARY KEY (idVeiculo),
    UNIQUE INDEX placa_UNIQUE (placa ASC) VISIBLE,
    CONSTRAINT fk_VeiculoMotorista
        FOREIGN KEY (idMotorista)
        REFERENCES Motorista (idMotorista)
        ON DELETE RESTRICT
        ON UPDATE RESTRICT
);

CREATE TABLE Pedido (
    numPedido INT NOT NULL,
    idCliente INT NOT NULL,
    idLoja INT NOT NULL,
    codEntrega VARCHAR(11) NOT NULL,
    idVeiculo INT NOT NULL,
    valor DECIMAL(7,2) NOT NULL DEFAULT 0.00,
    data DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    distanciaEntrega DECIMAL(4,1) NOT NULL,
    PRIMARY KEY (numPedido),
    UNIQUE INDEX codEntrega_UNIQUE (codEntrega ASC) VISIBLE,
    CONSTRAINT fk_PedidoCliente
        FOREIGN KEY (idCliente)
        REFERENCES Cliente (idCliente)
        ON DELETE RESTRICT
        ON UPDATE RESTRICT,
    CONSTRAINT fk_PedidoLoja
        FOREIGN KEY (idLoja)
        REFERENCES Loja (idLoja)
        ON DELETE RESTRICT
        ON UPDATE RESTRICT,
    CONSTRAINT fk_PedidoVeiculo
        FOREIGN KEY (idVeiculo)
        REFERENCES VeiculoEntrega (idVeiculo)
        ON DELETE RESTRICT
        ON UPDATE RESTRICT
);

CREATE TABLE Produto (
    codProduto VARCHAR(14) NOT NULL,
    preco DECIMAL(6,2) NOT NULL,
    nome VARCHAR(30) NOT NULL,
    UNIQUE INDEX codProduto_UNIQUE (codProduto ASC) VISIBLE,
    PRIMARY KEY (codProduto)
);




CREATE TABLE ProdutosPedido (
    numPedido INT NOT NULL,
```

```
    codProduto VARCHAR(14) NOT NULL,
    quantidade INT UNSIGNED NOT NULL,
    precoVendido DECIMAL(6,2) NOT NULL,
    PRIMARY KEY (codProduto, numPedido),
    CONSTRAINT numPedido
            FOREIGN KEY (numPedido)
            REFERENCES Pedido (numPedido)
            ON DELETE RESTRICT
            ON UPDATE RESTRICT,
    CONSTRAINT codProduto
            FOREIGN KEY (codProduto)
            REFERENCES Produto (codProduto)
            ON DELETE RESTRICT
            ON UPDATE RESTRICT
);

-- Exemplo de inserção de CPF Repetido (UNIQUE) --

INSERT INTO Pessoa VALUES (11, "11111111111", "Ademir", "Guia","Rio de Janeiro", "RJ", "11111111","Botafogo", "Rua 11", 11, NULL);

INSERT INTO Pessoa VALUES (11, "11111111111", "Ademir", "Guia","Rio de Janeiro", "RJ", "11111111","Botafogo", "Rua 11", 11, NULL);

DELETE FROM PESSOA
WHERE idPessoa = 11;

COMMIT;
```

(b) Exemplos de ALTER TABLE (pelo menos 3 exemplos, envolvendo alterações diversas) e DROP TABLE.

```
USE LogisticaVendas;

CREATE TABLE ClienteBloqueado (
idCliente INT NOT NULL,
   totalPedidos INT NOT NULL
);

ALTER TABLE ClienteBloqueado ADD COLUMN cpf CHAR(11) NULL;

ALTER TABLE ClienteBloqueado DROP COLUMN totalPedidos CASCADE;

ALTER TABLE ClienteBloqueado ADD CONSTRAINT fk_BloqueadoCliente FOREIGN KEY (idCliente) REFERENCES Cliente (idCliente);

DROP TABLE ClienteBloqueado;

SELECT *
FROM ClienteBloqueado;

COMMIT;
```

(c) Exemplo de uma inserção de dados em cada uma das tabelas. Para testar o trabalho, recomenda-se inserir dezenas de registros em cada tabela.

```
USE LogisticaVendas;

-- Exemplo de inserção de dados na tabela Pessoa --

INSERT INTO Pessoa VALUES (1, "12121212121", "David", "Costa", "Carmo da Cachoeira","MG", "37225000", "Bom Retiro", "Lauro Rezende de Vilela", 201, NULL);

INSERT INTO Pessoa VALUES (2, "12365289753", "Daphne", "Vilela", "Carmo da Cachoeira","MG", "63100545", "Centro", "Avenida JK", 201, "Próximo ao shopping");

INSERT INTO Pessoa VALUES (3, "78219401001", "Guilherme", "Grego", "Divinópolis", "MG", "68553015", "Serrinha", "Rua Primavera", 501, NULL);

INSERT INTO Pessoa VALUES (4, "28923867055", "Mário", "Couto", "São Gonçalo", "MG", "30112971", "Funcionários", "Avenida Getúlio Vargas", 631, NULL);

INSERT INTO Pessoa VALUES (5, "10101010100", "Antonio", "Lanza", "Lavras", "MG", "72241564", "Avenida Lagoa dos Ipês","Aquenta Sol", "61", NULL);

INSERT INTO Pessoa VALUES (6, "10101010122", "Joaozinho", "Silva", "Lavras", "MG", "65906564", "Rua Qualquer", "Bairro Qualquer", "782", "Complemento qualquer");

INSERT INTO Pessoa VALUES (7, "12345678900", "Victor", "Lima", "Campo Belo", "MG", "41219895", "Panorama", "Pedro Neves", 75, NULL);

INSERT INTO Pessoa VALUES (8, "12547895673", "Dudu", "Mata", "Campo Belo", "MG", "58400227", "Árvores", "Adel", 130, "Casa");

INSERT INTO Pessoa VALUES (9, "47845698725", "Kaka", "Gon", "Campo Belo", "MG", "76913807", "Palmeiras", "Barros", 35, "Prédio");

INSERT INTO Pessoa VALUES (10, "15983264598", "Carla", "Miranda", "Belo Horizonte", "MG", "41412270", "Alto do Peró", "Avenida Pinheiros", 412, "Ao lado do mercado Mart Minas");

INSERT INTO Pessoa VALUES (11, "11111111111", "Ademir", "Guia","Rio de Janeiro", "RJ", "11111111","Botafogo", "Rua 11", 11, NULL);

INSERT INTO Pessoa VALUES (12, "22222222222", "Fausto", "Silva","Porto Ferreira", "SP", "22222222","Jardim Dalva", "Rua 12", 22, "Perto do cemitério");

-- Exemplo de inserção de dados na tabela Telefone --

INSERT INTO Telefone VALUES (1,"2239282181");
INSERT INTO Telefone VALUES (1,"8192323090");
INSERT INTO Telefone VALUES (2,"6132482381");
```

```
INSERT INTO Telefone VALUES (3, "35921313098");
INSERT INTO Telefone VALUES (3, "35111111111");
INSERT INTO Telefone VALUES (4, "35222222222");
INSERT INTO Telefone VALUES (4, "35333333333");
INSERT INTO Telefone VALUES (5, "35444444444");
INSERT INTO Telefone VALUES (6, "35555555555");
INSERT INTO Telefone VALUES (6, "35666666666");
INSERT INTO Telefone VALUES (6, "35777777777");
INSERT INTO Telefone VALUES (7, "35888888888");
INSERT INTO Telefone VALUES (8, "35999999999");
INSERT INTO Telefone VALUES (9, "11111111111");
INSERT INTO Telefone VALUES (10, "11222222222");
INSERT INTO Telefone VALUES (10, "11333333333");
INSERT INTO Telefone VALUES (10, "11444444444");
INSERT INTO Telefone VALUES (11, "11555555555");
INSERT INTO Telefone VALUES (11, "11666666666");
INSERT INTO Telefone VALUES (12, "11777777777");

-- Exemplo de inserção de dados na tabela Cliente --

INSERT INTO Cliente (idCliente, email) VALUES (1,"david@email.com");
INSERT INTO Cliente (idCliente, email) VALUES (2, "emailTeste@email.com");
INSERT INTO Cliente (idCliente, email) VALUES (3, "dudu@email.com");
INSERT INTO Cliente (idCliente, email) VALUES (4, "cleinte98@email.com");
INSERT INTO Cliente (idCliente, email) VALUES (5, "fausto123@email.com");
INSERT INTO Cliente (idCliente, email) VALUES (6, "nescau_radical@email.com");

-- Exemplo de inserção de dados na tabela Loja --

INSERT INTO Loja (idLoja, cnpj, nome, cep, rua, numero, bairro, cidade, estado)
VALUES (1, "23456", "Mercenaria Azul", "62011177", "Rua da Aleatoria", 127, "Bairro Final", "São Paulo", "SP");

INSERT INTO Loja (idLoja, cnpj, nome, cep, rua, numero, bairro, cidade, estado)
VALUES (2, "12345678998765", "Papelaria X", "62011140", "Rua Domingos Olímpio", 35, "Centro", "Lavras", "MG");

INSERT INTO Loja (idLoja, cnpj, nome, cep, rua, numero, bairro, cidade, estado)
VALUES (3, "45256255221114", "Sapataria V", "04302021", "Rua Paracatu", 47, "Parque Imperial", "São Paulo", "SP");

INSERT INTO Loja (idLoja, cnpj, nome, cep, rua, numero, bairro, cidade, estado)
VALUES (4,"73182454000102","Orient","32143290","Rua B","17","Colorado","Contagem","MG");

INSERT INTO Loja (idLoja, cnpj, nome, cep, rua, numero, bairro, cidade, estado)
VALUES (5,"37243428000175", "Fluxo de Dados", "79050300","Rua Geraldo Castelo", "99", "Jardim Paulista", "Campo Grande", "MS");



-- Exemplo de inserção de dados na tabela Funcionario --

INSERT INTO Funcionario VALUES (1, "1010", 1000.00, 1);
INSERT INTO Funcionario VALUES (6, "2020", 1760.50, 2);
INSERT INTO Funcionario VALUES (7, "1020", 1000.00, 2);
```

```
INSERT INTO Funcionario VALUES (8, "5252", 2100.00, 2);
INSERT INTO Funcionario VALUES (9, "0707", 5600.00, 3);
INSERT INTO Funcionario VALUES (10, "1111", 800.00, 4);
INSERT INTO Funcionario VALUES (11, "2222", 1700.00, 5);

-- Exemplo de inserção de dados na tabela Motorista --

INSERT INTO Motorista VALUES (7, "91239293023", "B");
INSERT INTO Motorista VALUES (9, "47895632587", "D");

-- Exemplo de inserção de dados na tabela VeiculoEntrega --

INSERT INTO VeiculoEntrega VALUES (1, 7, "BCA2925", "Caminhão", NULL, 4.5);
INSERT INTO VeiculoEntrega VALUES (2, 9, "ABC2525", "Moto", 160, NULL);
INSERT INTO VeiculoEntrega VALUES (3, 7, "ZSA2755", "Caminhão", NULL, 3.7);

-- Exemplo de inserção de dados na tabela Produto --

INSERT INTO Produto VALUES(1, 25.00, "Par de Luvas");
INSERT INTO Produto VALUES(2, 4000.00, "Notebook");
INSERT INTO Produto VALUES(3, 1200.50, "Cama de Solteiro");
INSERT INTO Produto VALUES(4, 9000.00, "Galaxy S22");
INSERT INTO Produto VALUES(5, 1.99, "Bala");
INSERT INTO Produto VALUES(6, 9.99, "Chocolate");
INSERT INTO Produto VALUES(7, 12.50, "Chinelo");
INSERT INTO Produto VALUES(8, 111.99, "Sapato");
INSERT INTO Produto VALUES(9, 69.99, "Mouse");
INSERT INTO Produto VALUES(10, 50.00, "Película Iphone 8");
INSERT INTO Produto VALUES(11, 75.00, "Teclado Logitech");
INSERT INTO Produto VALUES(12, 9999.50 , "Macbook M1 PRO");
INSERT INTO Produto VALUES(13, 199.00, "Fone sem fio Xiaomi");
INSERT INTO Produto VALUES(14, 256.95, "Filtro de barro");
INSERT INTO Produto VALUES(15, 15.00, "Garrafa");
INSERT INTO Produto VALUES(16, 1.00, "Chiclete");
INSERT INTO Produto VALUES(17, 359.00, "Tênis adidas All black");
INSERT INTO Produto VALUES(18, 898.25, "CookTop");

-- Exemplo de inserção de dados na tabela Pedido --

INSERT INTO Pedido (numPedido, idCliente, idLoja, codEntrega, idVeiculo, distanciaEntrega)
VALUES (1, 2, 1, "45454477789", 1, 35.0);
INSERT INTO Pedido (numPedido, idCliente, idLoja, codEntrega, idVeiculo, distanciaEntrega)
VALUES (2, 3, 5, "09122930239", 2, 75.0);
INSERT INTO Pedido (numPedido, idCliente, idLoja, codEntrega, idVeiculo, distanciaEntrega)
VALUES (3, 5, 3, "12932332301", 1, 25.0);
INSERT INTO Pedido (numPedido, idCliente, idLoja, codEntrega, idVeiculo, distanciaEntrega)
VALUES (4, 3, 1, "54533443413", 2, 8.0);



-- Exemplo de inserção de dados na tabela Loja --

INSERT INTO ProdutosPedido (numPedido, codProduto, quantidade, precoVendido)
VALUES(1,2,1,4000.00);
INSERT INTO ProdutosPedido (numPedido, codProduto, quantidade, precoVendido)
```

```
VALUES(2,15,5,15.00);
INSERT INTO ProdutosPedido (numPedido, codProduto, quantidade, precoVendido)
VALUES(3,10,3,50.00);
INSERT INTO ProdutosPedido (numPedido, codProduto, quantidade, precoVendido)
VALUES(1, 10, 20, 100.00);
INSERT INTO ProdutosPedido (numPedido, codProduto, quantidade, precoVendido)
VALUES(4, 18, 2, 850.00);


COMMIT;
```

(d) Exemplos de modificação de dados em 5 tabelas.

```
USE LogisticaVendas;

UPDATE Loja
SET nome = "Loja Oriental"
WHERE idloja = 4;

UPDATE Produto P
SET preco = "23.35"
WHERE P.nome = "Par de luvas";

UPDATE Pedido P
SET valor = valor - 10
WHERE idCliente IN (
  SELECT idCliente
  FROM Cliente C
  WHERE C.email = "fausto123@email.com"
);

UPDATE Telefone
SET fone = fone + 1
WHERE idPessoa > 5;

UPDATE motorista
SET habilitacao = "AB"
WHERE idMotorista = 7;

COMMIT;
```

(e) Exemplos de exclusão de dados em 5 tabelas.

```
/*(e) Exemplos de exclusão de dados em 5 tabelas. */
```

```
USE LogisticaVendas;

/* 1 - Deleta a pessoas que são de Minas Gerais com DDD 81*/
DELETE Pessoa
FROM Telefone NATURAL JOIN Pessoa
WHERE estado = 'MG' AND fone LIKE '81%';

/*2 - Deleta veículos que são diferentes dos tipo caminhão*/
DELETE FROM VeiculoEntrega
WHERE tipoVeiculo != 'caminhão';

/*3 - Deleta Produtos pedidos com valores entre 1 e 100 reais */
DELETE ProdutosPedido
FROM ProdutosPedido NATURAL JOIN Produto
WHERE preco BETWEEN 1 AND 100;

/*4 - Deleta cliente com email david@email.com*/
DELETE FROM Cliente
WHERE email LIKE 'david@email.com';

/*5 Deleta telefones pessoas do estado de MG com DDD diferente de 35*/
DELETE FROM Telefone
WHERE fone NOT LIKE '35%' AND EXISTS(
   SELECT *
   FROM Pessoa
   WHERE estado = 'MG');

COMMIT;
```

(f) Exemplos de, pelo menos, 12 consultas.

```
USE LogisticaVendas;
```

```
/* 1. Recupera o nome completo, o CPF e o salario de pessoas que são funcionários de uma loja
     da cidade de Lavras em ordem decrescente de salário */
SELECT CONCAT(P.primeiroNome, " ", P.sobrenome) AS nome, P.cpf, F.salario
FROM Pessoa P, Funcionario F, Loja L
WHERE P.idPessoa = F.idFuncionario AND F.idloja = L.idLoja AND L.cidade = "Lavras"
ORDER BY salario DESC, P.primeiroNome ASC, P.sobrenome ASC;

/* 2. Recupera o nome completo, o CPF e o email de clientes que não fizeram nenhum pedido
     em ordem alfabética de nomes.*/
SELECT CONCAT(P.primeiroNome, " ", P.sobrenome) AS nome, P.cpf, C.email
FROM Cliente C LEFT OUTER JOIN Pedido Pd ON C.idCliente = Pd.idCliente, Pessoa P
WHERE P.idPessoa = C.idCliente AND Pd.idCliente IS NULL;

/* 3. Recupera o nome completo, o registro da CNH e o número de veículos que um motorista
trabalha */
SELECT CONCAT(P.primeiroNome, " ", P.sobrenome) AS nome, M.regCNH, COUNT(*) AS
numeroVeiculos
FROM Pessoa P, Funcionario F, Motorista M NATURAL JOIN VeiculoEntrega V
WHERE idPessoa = idFuncionario AND idFuncionario = idMotorista
GROUP BY M.idMotorista;

/*4. Recupera o nome, o valor e quantidade de vezes que um produto aparece nos pedidos.
 Apenas para produtos que estão contido em mais de um pedido */
SELECT nome, preco, COUNT(*) qtdePedida
FROM produto NATURAL JOIN produtospedido
GROUP BY codProduto
HAVING qtdePedida > 1;

/*5.Recupera o nome, endereço e a identificação da loja em que trabalha o
funcionário que tem a letra A no fim do primeiro nome */
SELECT CONCAT(P.primeiroNome, " ", P.sobrenome) AS nome, rua, numero, bairro, cep, cidade,
estado, idLoja
FROM pessoa P JOIN funcionario F ON P.idPessoa = F.idFuncionario
WHERE P.primeiroNome LIKE '%a';

/*6. Recupera o numero e o valor do pedido, onde a distancia da entrega
 está entre 5 e 25 km ou o valor do pedido seja maior que 5000*/
SELECT numPedido, valor
FROM pedido
WHERE distanciaEntrega BETWEEN 5 and 30
UNION
SELECT numPedido, valor
FROM pedido
WHERE valor > 5000;

/*7. Seleciona a identificação do cliente e da loja que pediram o produto de codigo 2 */
SELECT idCliente, idLoja
FROM Pedido NATURAL JOIN ProdutosPedido
WHERE codProduto = 2;



/*8. Seleciona a identificação, nome, sobrenome e telefones
     dos funcionarios cujo DDD do telefone é 35 */
SELECT P.idPessoa, primeiroNome, sobrenome, fone
From Telefone AS T, Pessoa AS P, Funcionario AS F
```

```
WHERE fone LIKE '35%' AND F.idFuncionario = P.idPessoa AND T.idPessoa = P.idPessoa;

/*9. Seleciona a identificação da loja, soma os salarios dos funcionarios
 e conta a quantidade de funcionarios, em que a soma dos salários é maior que 1000
 e ordenando em ordem crescente */
SELECT idLoja, COUNT(*) AS qtdeFuncs, SUM(salario) AS SomaSalarios
FROM Funcionario GROUP BY idLoja
HAVING SomaSalarios > 1000 ORDER BY SomaSalarios ASC;

/*10 Recupera a quantidade e o nome do produto que teve maior quantidade pedida de uma só vez*/
SELECT nome AS nome_do_produto, quantidade AS maior_quantidade_pedida
FROM ProdutosPedido NATURAL JOIN Produto
WHERE quantidade IN (
SELECT MAX(quantidade)
FROM ProdutosPedido);

/*11 Recupera a média salarial de cada estado com salario e nome em ordem crescente */
SELECT nome AS NomeLoja, AVG(salario) AS MediaSalarial, estado AS Estado
FROM Loja NATURAL JOIN Funcionario
GROUP BY estado, nome, salario
ORDER BY nome ASC, salario ASC;

/*12 Recupera o nome completo e o telefone de pessoas que são de Minas Gerais e possuem telefone com DDD diferente de 35 com o nome completo em ordem crescente*/
SELECT DiSTINCT CONCAT(primeiroNome, " ", sobrenome) AS NomeCompleto, fone AS Telefone
FROM Telefone NATURAL JOIN Pessoa
WHERE fone NOT LIKE '35%' AND EXISTS(
SELECT *
FROM Pessoa
WHERE estado = 'MG')
ORDER BY nomeCompleto ASC;

/*13 Recupera o nome, o estado e o total vendido de lojas em Minas Gerais ou lojas que venderam mais que 500 */
SELECT L.nome, L.estado, L.totalVendido
FROM Loja L
WHERE estado = "MG" OR totalVendido > 500;

/*14 Recupera o salario e o id da loja de funcionarios que não são motoristas */
SELECT DISTINCT F.salario, F.idLoja
FROM Funcionario F JOIN Motorista M
WHERE F.idFuncionario NOT IN(SELECT M.idMotorista
                                                FROM Motorista M);

/*15 Recupera o código e nome de produtos distintos que foram pedidos */
SELECT DISTINCT P.codProduto, P.nome
FROM Produto P JOIN ProdutosPedido PP
WHERE P.codProduto = PP.codProduto;



/*16 Recupera o id das lojas que possuem (EXIST) um fucionário com sobrenome "Silva" */
SELECT F.idLoja
FROM Funcionario F
WHERE EXISTS (SELECT P.*
```

```
            FROM Pessoa P
        WHERE P.idPessoa = F.idFuncionario AND P.sobrenome = "Silva");

/*17 Recupera todos os produtos que são mais caros que "Par de Luvas" */
SELECT P.nome, P.preco
FROM Produto P
WHERE P.preco > ALL(SELECT P.preco
                    FROM Produto P
                    WHERE P.nome = "Par de Luvas")
ORDER BY P.preco ASC;

/*18 Recupera o id e o salario dos funcionarios que ganham mais do que algum funcionário da loja
2*/
SELECT F.idFuncionario, F.salario
FROM Funcionario F
WHERE F.salario > SOME(SELECT F.salario
                                FROM Funcionario F
            WHERE F.idLoja = 2)
ORDER BY F.salario ASC;

COMMIT;
```

(g) Exemplos de criação de 3 visões (Views).

```
USE LogisticaVendas;
```

```
/*  Visão que armazena para cada produto a quantidade de pedidos realizados,
    e soma o valor total pedido */
CREATE VIEW PedidosDeProdutos AS
SELECT codProduto, COUNT(numPedido) AS qtdePedida, SUM(preco) AS valorTotalPedido
FROM ProdutosPedido NATURAL JOIN Produto
GROUP BY codProduto;

SELECT codProduto, qtdePedida FROM PedidosDeProdutos ORDER BY qtdePedida DESC;

DROP VIEW PedidosDeProdutos;


/* Visao que armazena os dados do endereco de uma pessoa em apenas uma coluna */
CREATE VIEW enderecoPessoa AS
SELECT idPessoa, CONCAT(rua,', ', numero, ', ',bairro,', ', cidade, ', ', estado, ', ',cep) AS endereco,
complemento
FROM Pessoa;

SELECT endereco, complemento FROM enderecoPessoa WHERE idPessoa = 8;
SELECT idPessoa, endereco FROM enderecoPessoa WHERE endereco LIKE '%MG%';

DROP VIEW enderecoPessoa;



/* Visao que armazena os telefones de uma pessoa em apenas uma coluna */
CREATE VIEW telefonesPessoa (idPessoa, telefones)AS
SELECT idPessoa, GROUP_CONCAT(fone) AS fones
FROM Telefone
GROUP BY idPessoa;

SELECT telefones FROM telefonesPessoa WHERE idPessoa = 1;

DROP VIEW telefonesPessoa;


/* Visao que armazena os dados principais de um funcionario */
CREATE VIEW dadosFuncionario (idPessoa, nome, cpf, endereco, telefones, salario, lojaTrabalho
) AS
SELECT P.idPessoa, CONCAT(P.primeiroNome, ' ', P.sobrenome), P.cpf, E.endereco, T.telefones,
F.salario, F.idLoja
FROM Pessoa AS P, enderecoPessoa AS E, Funcionario AS F, telefonesPessoa AS T
WHERE P.idPessoa = F.idFuncionario AND P.idPessoa = E.idPessoa AND T.idPessoa =
P.idPessoa;

SELECT * FROM dadosFuncionario;
SELECT idPessoa, nome, endereco, telefones FROM dadosFuncionario WHERE lojaTrabalho = 1;

DROP VIEW dadosFuncionario;

COMMIT;
```

(h) Exemplos de criação de usuários (pelo menos 2), concessão (GRANT) e revocação (REVOKE) de permissão de acesso.

```
USE LogisticaVendas;

/* Criação de um usuário que tem permissão de selecionar dados dos veículos utilizados para entrega e
   executar um store producere. */

CREATE USER 'guilherme'@'localhost' IDENTIFIED BY '0000';

GRANT SELECT ON LogisticaVendas.veiculoentrega TO 'guilherme'@'localhost';

REVOKE SELECT ON LogisticaVendas.veiculoentrega FROM 'guilherme'@'localhost';

GRANT EXECUTE ON PROCEDURE LogisticaVendas.InfosDaLoja TO 'guilherme'@'localhost';

DROP USER 'guilherme'@'localhost';


/* Criação de um usuário que tem permissão de selecionar dados de todas as tabelas */

CREATE USER 'teste'@'localhost' IDENTIFIED BY '1111';

GRANT SELECT ON LogisticaVendas.* TO 'teste'@'localhost';

DROP USER 'teste'@'localhost';

COMMIT;
```

(i) Exemplos de 3 procedimentos/funções, com e sem parâmetros, de entrada e de saída, contendo alguns comandos tais como IF, CASE WHEN, WHILE, declaração de variáveis e funções prontas.

```
USE LogisticaVendas;

/* Retorna o nome de uma determinada loja passada como parametro
   e a quantidade de produtos vendidos por ela */
DELIMITER //
CREATE PROCEDURE InfosDaLoja(IN pIdLoja DECIMAL(2))
BEGIN
   SELECT nome, produtosVendidos
   FROM Loja
   WHERE idLoja = pIdLoja;
END //
DELIMITER ;

CALL InfosDaLoja(1);
CALL InfosDaLoja(2);
CALL InfosDaLoja(3);
CALL InfosDaLoja(4);
CALL InfosDaLoja(5);

DROP PROCEDURE InfosDaloja;


/* Retorna a quantidade de numero de telefone cadastrado de uma pessoa, dado seu nome */
DELIMITER //
CREATE PROCEDURE NroTelDePessoa(IN pPrimeiroNome varchar(30), IN pSobrenome
varchar(70), OUT pNumTel DECIMAL(4))
BEGIN
   SELECT Count(*) INTO pNumTel
   FROM Pessoa NATURAL JOIN Telefone
   WHERE primeiroNome = pPrimeiroNome AND sobrenome = pSobrenome;
END //
DELIMITER ;

CALL NroTelDePessoa('guilherme', 'grego', @numTel);
SELECT @numTel AS qtdTelefone;

DROP PROCEDURE NroTelDePessoa;




/* Retorna o nome e o preço do produto mais caro. */

DELIMITER //
```

```
CREATE PROCEDURE ProdutoMaisCaro (OUT nomeSaida VARCHAR(30), OUT resultado
DECIMAL(6,2))
BEGIN
     DECLARE valor DECIMAL(6,2) DEFAULT 0;
   DECLARE i INT DEFAULT 0;

     DECLARE PrecoComp DECIMAL(6,2) DEFAULT 0;
   DECLARE nomeComp VARCHAR(30);

   DECLARE meuCursor CURSOR FOR SELECT preco, nome FROM produto;
   DECLARE CONTINUE HANDLER FOR NOT FOUND SET i = 1;

   OPEN meuCursor;

   WHILE(i != 1) DO
              FETCH meuCursor INTO PrecoComp, nomeComp;
              IF PrecoComp > valor THEN
                        SET valor = PrecoComp;
         SET nomeSaida = nomeComp;
              END IF;
   END WHILE;

   SET resultado = valor;

   CLOSE meuCursor;

END //
DELIMITER ;

CALL ProdutoMaisCaro(@nomeA, @resultado1);
SELECT  @nomeA AS nome, @resultado1 AS Preço;

DROP PROCEDURE ProdutoMaisCaro;

COMMIT;
```

(j) Exemplos de 3 triggers, um para cada evento (inserção, alteração e exclusão). Inclua exemplos de como disparar os triggers.

```
USE LogisticaVendas;

-- Para atualizar o valor de funcionários trabalhando para uma loja. --

DELIMITER //
CREATE TRIGGER adicionarFuncionario
AFTER INSERT ON Funcionario
FOR EACH ROW
BEGIN
    UPDATE Loja L
    SET L.qtdFuncionarios = L.qtdFuncionarios + 1
    WHERE L.idLoja = NEW.idLoja;
END //
DELIMITER ;

DELIMITER //
CREATE TRIGGER removerFuncionario
AFTER DELETE ON Funcionario
FOR EACH ROW
BEGIN
    UPDATE Loja L
    SET L.qtdFuncionarios = L.qtdFuncionarios - 1
    WHERE L.idLoja = OLD.idLoja;
END //
DELIMITER ;

-- Para atualizar o valor total do pedido de acordo com os produtos adicionados. --

DELIMITER //
CREATE TRIGGER valorPedido
AFTER INSERT ON ProdutosPedido
FOR EACH ROW
BEGIN
    UPDATE Pedido P
    SET P.valor = P.valor + NEW.quantidade * New.precoVendido
    WHERE P.numPedido = NEW.numPedido;
END //
DELIMITER ;

-- Para atualizar o valor total e a quantidade de itens vendidos por uma loja. --

DELIMITER //
CREATE TRIGGER totalVendidoLoja
AFTER INSERT ON ProdutosPedido
FOR EACH ROW
BEGIN
    UPDATE Loja L, Pedido P
    SET L.totalVendido = L.totalVendido + NEW.quantidade * New.precoVendido
    WHERE P.numPedido = NEW.numPedido AND L.idLoja = P.idLoja;
END //
DELIMITER ;

DELIMITER //
CREATE TRIGGER produtosVendidosLoja
AFTER INSERT ON ProdutosPedido
```

```
FOR EACH ROW
BEGIN
    UPDATE Loja L, Pedido P
    SET L.produtosVendidos = L.produtosVendidos + NEW.quantidade
    WHERE P.numPedido = NEW.numPedido AND L.idLoja = P.idLoja;
END //
DELIMITER ;
```

-- Para atualizar o valor total gasto e a quantidade de pedidos feitos por um cliente. --

```
DELIMITER //
CREATE TRIGGER totalGastoCliente
AFTER INSERT ON ProdutosPedido
FOR EACH ROW
BEGIN
    UPDATE Cliente C, Pedido P
    SET C.totalGasto = C.totalGasto + NEW.quantidade * NEW.precoVendido
    WHERE P.numPedido = NEW.numPedido AND C.idCliente = P.idCliente;
END //
DELIMITER ;

DELIMITER //
CREATE TRIGGER totalPedidosCliente
AFTER INSERT ON Pedido
FOR EACH ROW
BEGIN
    UPDATE Cliente C
    SET C.pedidosEfetuados = C.pedidosEfetuados + 1
    WHERE C.idCliente = NEW.idCliente;
END //
DELIMITER ;
```

-- Exemplo de trigger que adiciona funcionarios que recebem um aumento de salário numa tabela própria. --

```
CREATE TABLE FuncionarioPromovido (
```

```
    idFuncionario INT NOT NULL,
  CONSTRAINT fk_idFuncionario
            FOREIGN KEY (idFuncionario)
            REFERENCES Funcionario (idFuncionario)
            ON DELETE RESTRICT
            ON UPDATE RESTRICT
);

DELIMITER //
CREATE TRIGGER verificarPromocao
AFTER UPDATE ON Funcionario
FOR EACH ROW
BEGIN
 IF NEW.salario > OLD.salario THEN
   INSERT INTO FuncionarioPromovido VALUE (NEW.idFuncionario);
 END IF;
END //
DELIMITER ;

COMMIT;
```