--Criando o banco de dados

CREATE TABLE Clientes (

id serial PRIMARY KEY,

nome varchar(40)

);

CREATE TABLE Pedidos (

id serial PRIMARY KEY,

data date,

fk\_Clientes\_id int

);

CREATE TABLE Produtos (

id serial PRIMARY KEY,

descricao varchar(40),

preco decimal(6,2),

estoque decimal(6,2),

fk\_Categorias\_id int

);

CREATE TABLE ItensPedido (

Quantidade int,

ValorUnitario decimal(6,2),

fk\_Pedidos\_id int,

fk\_Produtos\_id int

);

CREATE TABLE Categorias (

id serial PRIMARY KEY,

descricao varchar(20)

);

ALTER TABLE Pedidos ADD CONSTRAINT FK\_Pedidos\_2

FOREIGN KEY (fk\_Clientes\_id)

REFERENCES Clientes (id)

ON DELETE CASCADE;

--Alterações necessárias para as functions

ALTER TABLE Pedidos ADD status varchar(1) not null DEFAULT 'A';

ALTER TABLE pedidos ADD parcelas int not null DEFAULT 1;

ALTER TABLE Produtos ADD CONSTRAINT FK\_Produtos\_2

FOREIGN KEY (fk\_Categorias\_id)

REFERENCES Categorias (id)

ON DELETE CASCADE;

ALTER TABLE ItensPedido ADD CONSTRAINT FK\_ItensPedido\_1

FOREIGN KEY (fk\_Pedidos\_id)

REFERENCES Pedidos (id)

ON DELETE CASCADE;

ALTER TABLE ItensPedido ADD CONSTRAINT FK\_ItensPedido\_2

FOREIGN KEY (fk\_Produtos\_id)

REFERENCES Produtos (id)

ON DELETE CASCADE;

--Atividade 1:

--Criação da tabela de log de produtos.

create table log\_produtos(

id serial primary key,

fk\_produtos\_id int,

descricao varchar(100),

estoque decimal(6,2),

valor decimal(6,2),

FOREIGN key (fk\_produtos\_id) references produtos (id)

);

-- Criando a função verifica\_log\_produtos

CREATE OR REPLACE FUNCTION verifica\_log\_produtos()

RETURNS TRIGGER AS $$

BEGIN

IF OLD.estoque <> NEW.estoque THEN

INSERT INTO log\_produtos(fk\_produtos\_id, descricao, estoque, valor)

VALUES (OLD.id, 'Estoque atualizado!', NEW.estoque, NEW.preco);

END IF;

IF OLD.preco <> NEW.preco THEN

INSERT INTO log\_produtos(fk\_produtos\_id, descricao, estoque, valor)

VALUES (OLD.id, 'Valor atualizado!', NEW.estoque, NEW.preco);

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

--Inserindo a trigger trigger\_verifica\_log\_produtos

CREATE or replace TRIGGER trigger\_verifica\_log\_produtos

before UPDATE ON produtos

FOR EACH ROW

EXECUTE FUNCTION verifica\_log\_produtos();

--Atividade 2:

--Criando a tabela log pedidos

create table log\_pedidos(

id serial primary key,

descricao varchar(100),

pedido int,

itemAdicionado int,

quantidade decimal(6,2)

);

-- Criação da log de adição

create or replace function verifica\_adicao\_itens()

returns trigger as $$

begin

if((select status from pedidos where id = new.fk\_pedidos\_id) = 'A' ) then

insert into log\_pedidos (descricao,pedido,itemAdicionado,quantidade)

values ('Item adicionado!', new.fk\_pedidos\_id, new.fk\_produtos\_id,new.quantidade);

else

raise 'O pedido não pode ser modificado!';

end if;

return new;

end;

$$ language plpgsql;

--Inserindo a trigger trigger\_verifica\_adicao\_itens

create trigger trigger\_verifica\_adicao\_itens

before insert on itenspedido

for each row

execute function verifica\_adicao\_itens();

--Criação do log de exclusão

create or replace function verifica\_exclusao\_itens()

returns trigger as $$

begin

if((select status from pedidos where id = new.fk\_pedidos\_id) = 'A' ) then

insert into log\_pedidos (descricao,pedido,itemAdicionado,quantidade)

values ('Item removido!', new.fk\_pedidos\_id, new.fk\_produtos\_id,new.quantidade);

else

raise 'O pedido não pode ser modificado!';

end if;

return new;

end;

$$ language plpgsql;

--Inserindo a trigger trigger\_verifica\_exclusao\_itens

create trigger trigger\_verifica\_exclusao\_itens

after delete on itenspedido

for each row

execute function verifica\_exclusao\_itens();

-- Cria a função de dar baixa no estoque

create or replace function verifica\_baixa\_estoque()

returns trigger

as $$

begin

if((select status from pedidos where id = new.id) = 'F' )then

UPDATE produtos

SET estoque = estoque - (

SELECT SUM(i.quantidade)

FROM itenspedido i

WHERE i.fk\_pedidos\_id = old.id AND i.fk\_produtos\_id = produtos.id)

WHERE id IN (

SELECT fk\_produtos\_id

FROM itenspedido

WHERE fk\_pedidos\_id = NEW.id

);

end if;

return new;

end;

$$ language plpgsql;

--Insere a trigger trigger\_verifica\_baixa\_estoque

create trigger trigger\_verifica\_baixa\_estoque

after update on pedidos

for each row

EXECUTE FUNCTION verifica\_baixa\_estoque();

--Atividade 3:

-- Cria a procedure cadastrar\_produtos()

create or replace procedure cadastrar\_produtos(p\_descricao varchar,p\_preco decimal,p\_estoque int,p\_categoria int)

as $$

begin

if ((select id from categorias where id = p\_categoria) is null) then

raise 'A categoria do produto que deseja cadastrar não existe!';

else

insert into produtos (descricao,preco,estoque,fk\_categorias\_id)

values (p\_descricao,p\_preco,p\_estoque,p\_categoria);

end if;

end;

$$language plpgsql;

--Atividade 4:

-- Cria da tabela parcelas

create table parcelas(

id serial primary key,

fk\_pedido\_id int not null,

forma\_pagamento varchar(9),

valor decimal(6,2) not null,

vencimento date not null

);

-- criação da função verifica\_parcelas()

CREATE OR REPLACE FUNCTION verifica\_parcelas() RETURNS TRIGGER AS $$

DECLARE

qnt\_parcelas INT;

data\_parcelas DATE;

valor\_parcela DECIMAL(6,2);

f\_forma\_pagamento VARCHAR(9);

BEGIN

IF ((SELECT status FROM pedidos WHERE id = NEW.id) = 'F') THEN

qnt\_parcelas := (SELECT parcelas FROM pedidos WHERE id = NEW.id);

data\_parcelas := (SELECT data FROM pedidos WHERE id = NEW.id);

valor\_parcela := (SELECT SUM(i.valorunitario\*i.quantidade) FROM itenspedido as i

WHERE fk\_pedidos\_id = NEW.id) / qnt\_parcelas;

FOR i IN 1..qnt\_parcelas LOOP

IF qnt\_parcelas > 1 THEN

f\_forma\_pagamento := 'parcelado';

ELSE

f\_forma\_pagamento := 'à vista';

END IF;

INSERT INTO parcelas (fk\_pedido\_id, forma\_pagamento, vencimento, valor)

VALUES (NEW.id, f\_forma\_pagamento, data\_parcelas + INTERVAL '1 month' \* i, valor\_parcela);

END LOOP;

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

-- inserindo trigger trigger\_verifica\_parcelas

create trigger trigger\_verifica\_parcelas

after update on pedidos

for each row

execute function verifica\_parcelas();