```
CREATE TABLE Categoria (
    categoria_id INT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL
);
CREATE TABLE Produto (
    produto_id INT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    descricao TEXT,
    peso DECIMAL(10, 2),
    data_validade DATE,
    categoria_id INT,
    FOREIGN KEY (categoria_id) REFERENCES Categoria(categoria_id)
);
CREATE TABLE Fornecedor (
    fornecedor_id INT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    cnpj VARCHAR(20) UNIQUE,
    endereco TEXT,
    telefone VARCHAR(20),
    email VARCHAR(100)
);
CREATE TABLE ProdutoFornecedor (
    produto_id INT,
    fornecedor_id INT,
    PRIMARY KEY (produto_id, fornecedor_id),
    FOREIGN KEY (produto_id) REFERENCES Produto(produto_id),
    FOREIGN KEY (fornecedor_id) REFERENCES Fornecedor(fornecedor_id)
);
CREATE TABLE LocalArmazenagem (
```

```
    local_id INT PRIMARY KEY,
    nome VARCHAR(100) NOT NULL,
    tipo VARCHAR(50),
    capacidade_maxima INT,
    descricao TEXT
);
CREATE TABLE Estoque (
    estoque_id INT PRIMARY KEY,
    produto_id INT,
    local_id INT,
    quantidade INT NOT NULL,
    data_ultima_movimentacao DATE,
    FOREIGN KEY (produto_id) REFERENCES Produto(produto_id),
    FOREIGN KEY (local_id) REFERENCES LocalArmazenagem(local_id)
);
CREATE TABLE Inventario (
    inventario_id INT PRIMARY KEY,
    produto_id INT,
    local_id INT,
    data DATE NOT NULL,
    quantidade_sistema INT,
    quantidade_fisica INT,
    observacoes TEXT,
    FOREIGN KEY (produto_id) REFERENCES Produto(produto_id),
    FOREIGN KEY (local_id) REFERENCES LocalArmazenagem(local_id)
);


INSERT INTO Categoria (categoria_id, nome) VALUES
```

(1, 'Alimentos'),

(2, 'Eletrônicos'),

(3, 'Limpeza'),

(4, 'Bebidas'),

(5, 'Papelaria'),

(6, 'Higiene Pessoal');

INSERT INTO Produto (produto_id, nome, descricao, peso, data_validade, categoria_id) VALUES

(101, 'Arroz 5kg', 'Arroz branco tipo 1', 5.0, '2025-12-31', 1),

(102, 'Notebook Dell', 'Notebook 15 polegadas', 2.2, NULL, 2),

(103, 'Detergente', 'Detergente neutro 500ml', 0.5, '2026-01-01', 3),

(104, 'Refrigerante Cola', 'Garrafa 2L', 2.0, '2025-11-15', 4),

(105, 'Caderno 100 folhas', 'Capa dura', 0.4, NULL, 5),

(106, 'Feijão Preto 1kg', 'Feijão tipo 1', 1.0, '2026-01-10', 1),

(107, 'Macarrão Espaguete', 'Macarrão tipo espaguete 500g', 0.5, '2026-03-01', 1),

(108, 'Queijo Mussarela', 'Queijo fatiado 1kg', 1.0, '2025-08-20', 1),

(109, 'Smartphone X', 'Celular com 128GB', 0.3, NULL, 2),

(110, 'TV 50 polegadas', 'Smart TV 4K', 10.0, NULL, 2),

(111, 'Shampoo 300ml', 'Shampoo anticaspa', 0.3, '2026-01-01', 6),

(112, 'Sabonete Neutro', 'Sabonete 90g', 0.1, '2026-01-01', 6),

(113, 'Arroz Integral 1kg', 'Arroz integral tipo 1', 1.0, '2026-02-01', 1),

(114, 'Bolacha Recheada', 'Sabor chocolate 150g', 0.15, '2025-12-10', 1),

(115, 'Tablet Android', 'Tablet 10 polegadas', 0.7, NULL, 2);

INSERT INTO Fornecedor (fornecedor_id, nome, cnpj, endereco, telefone, email) VALUES

(1, 'Fornecedor Alimentos LTDA', '11.111.111/0001-11', 'Rua A, 100', '8599001122', 'alimentos@forn.com'),

(2, 'Tech Eletrônicos', '22.222.222/0001-22', 'Av. B, 200', '8599002233', 'eletronicos@tech.com'),

(3, 'Limpa Tudo S.A.', '33.333.333/0001-33', 'Rua C, 300', '8599003344', 'contato@limpatudo.com'),

(4, 'Bebidas Brasil', '44.444.444/0001-44', 'Av. D, 400', '8599004455', 'bebidas@brasil.com'),

(5, 'Escola Fácil', '55.555.555/0001-55', 'Rua E, 500', '8599005566', 'vendas@escolafacil.com'),

(6, 'Delícias do Sul', '66.666.666/0001-66', 'Rua F, 600', '8599006677', 'contato@deliciasdosul.com'),

(7, 'Eletronic World', '77.777.777/0001-77', 'Av. G, 700', '8599007788', 'suporte@eletronicworld.com');

INSERT INTO ProdutoFornecedor (produto_id, fornecedor_id) VALUES

(101, 1),

(102, 2),

(103, 3),

(104, 4),

(105, 5),

(106, 1), (106, 6),

(107, 1), (107, 6),

(108, 1), (108, 6),

(113, 1), (113, 6),

(114, 1), (114, 6),

(109, 2), (109, 7),

(110, 2), (110, 7),

(111, 6),

(112, 6),

(115, 2), (115, 7);

INSERT INTO LocalArmazenagem (local_id, nome, tipo, capacidade_maxima, descricao) VALUES

(1, 'Prateleira A1', 'Prateleira', 100, 'Local superior'),

(2, 'Estante B2', 'Estante', 200, 'Estante central'),

(3, 'Palete C3', 'Palete', 50, 'Área baixa'),

(4, 'Corredor D4', 'Corredor', 300, 'Área de fácil acesso'),

(5, 'Freezer E5', 'Refrigerado', 30, 'Armazenamento de bebidas'),

(6, 'Depósito F6', 'Depósito', 500, 'Armazenamento geral adicional');

INSERT INTO Estoque (estoque_id, produto_id, local_id, quantidade, data_ultima_movimentacao) VALUES

(1, 101, 1, 50, '2025-07-01'),

(2, 102, 2, 10, '2025-07-05'),

(3, 103, 3, 80, '2025-07-10'),

(4, 104, 5, 20, '2025-07-15'),

(5, 105, 2, 60, '2025-07-16'),

(6, 106, 1, 40, '2025-07-20'),

(7, 107, 1, 60, '2025-07-20'),

(8, 108, 5, 25, '2025-07-20'),

(9, 109, 2, 15, '2025-07-20'),

(10, 110, 4, 8, '2025-07-20'),

(11, 111, 3, 70, '2025-07-20'),

(12, 112, 3, 90, '2025-07-20'),

(13, 113, 6, 30, '2025-07-20'),

(14, 114, 6, 45, '2025-07-20'),

(15, 115, 4, 12, '2025-07-20');

INSERT INTO Inventario (inventario_id, produto_id, local_id, data, quantidade_sistema, quantidade_fisica, observacoes) VALUES

(1, 101, 1, '2025-07-15', 50, 48, 'Diferença de 2 unidades'),

(2, 102, 2, '2025-07-15', 10, 10, 'Estoque OK'),

(3, 103, 3, '2025-07-15', 80, 80, 'Sem divergência'),

(4, 104, 5, '2025-07-15', 20, 19, '1 unidade danificada'),

(5, 105, 2, '2025-07-15', 60, 62, 'Erro de contagem anterior'),

(6, 106, 1, '2025-07-25', 40, 40, 'OK'),

(7, 107, 1, '2025-07-25', 60, 59, '1 unidade faltando'),

(8, 108, 5, '2025-07-25', 25, 25, 'OK'),

(9, 109, 2, '2025-07-25', 15, 15, 'OK'),

(10, 110, 4, '2025-07-25', 8, 8, 'OK'),

(11, 111, 3, '2025-07-25', 70, 70, 'OK'),

(12, 112, 3, '2025-07-25', 90, 89, '1 unidade faltando'),

(13, 113, 6, '2025-07-25', 30, 30, 'OK'),

(14, 114, 6, '2025-07-25', 45, 46, '1 unidade a mais'),

(15, 115, 4, '2025-07-25', 12, 12, 'OK');

-- Função que remove referências do produto em tabelas associadas

CREATE OR REPLACE FUNCTION remover_referencias_produto() RETURNS TRIGGER AS $$ BEGIN -- Remove entradas de ProdutoFornecedor DELETE FROM ProdutoFornecedor WHERE produto_id = OLD.produto_id;

```
-- Remove entradas de Estoque
DELETE FROM Estoque WHERE produto_id = OLD.produto_id;

-- Remove entradas de Inventario
DELETE FROM Inventario WHERE produto_id = OLD.produto_id;

RETURN OLD;
```

END; $$ LANGUAGE plpgsql;

-- Trigger que chama a função antes de excluir um produto

CREATE TRIGGER trg_remover_referencias_produto BEFORE DELETE ON Produto FOR EACH ROW EXECUTE FUNCTION remover_referencias_produto();

```
CREATE OR REPLACE FUNCTION verificar_capacidade_local() RETURNS
TRIGGER AS $$ DECLARE capacidade_max INT; total_existente INT;
BEGIN -- Obtém a capacidade máxima do local SELECT
l.capacidade_maxima INTO capacidade_max FROM LocalArmazenagem l
WHERE l.local_id = NEW.local_id;

-- Soma a quantidade já existente nesse local (excluindo o atual
se for UPDATE)
IF TG_OP = 'UPDATE' THEN
    SELECT COALESCE(SUM(quantidade), 0)
    INTO total_existente
    FROM Estoque
    WHERE local_id = NEW.local_id AND estoque_id <>
OLD.estoque_id;
ELSE
    SELECT COALESCE(SUM(quantidade), 0)
    INTO total_existente
    FROM Estoque
    WHERE local_id = NEW.local_id;
END IF;

-- Verifica se o novo total excede a capacidade
IF total_existente + NEW.quantidade > capacidade_max THEN
    RAISE EXCEPTION 'Capacidade excedida no local de armazenagem
(máximo: %, atual: %, tentativa: %)',
        capacidade_max, total_existente, NEW.quantidade;
END IF;

RETURN NEW;


END; $$ LANGUAGE plpgsql;

CREATE TRIGGER trg_verificar_capacidade BEFORE INSERT OR UPDATE
ON Estoque FOR EACH ROW EXECUTE FUNCTION
verificar_capacidade_local();



CREATE OR REPLACE FUNCTION atualizar_quantidade_sistema()
RETURNS TRIGGER AS $$ BEGIN -- Atualiza a quantidade_sistema no
Inventario com base na nova quantidade de Estoque UPDATE
Inventario SET quantidade_sistema = NEW.quantidade WHERE
produto_id = NEW.produto_id AND local_id = NEW.local_id;
```

```
RETURN NEW;


END; $$ LANGUAGE plpgsql;

CREATE TRIGGER trg_atualizar_quantidade_sistema AFTER INSERT OR
UPDATE ON Estoque FOR EACH ROW EXECUTE FUNCTION
atualizar_quantidade_sistema();
```