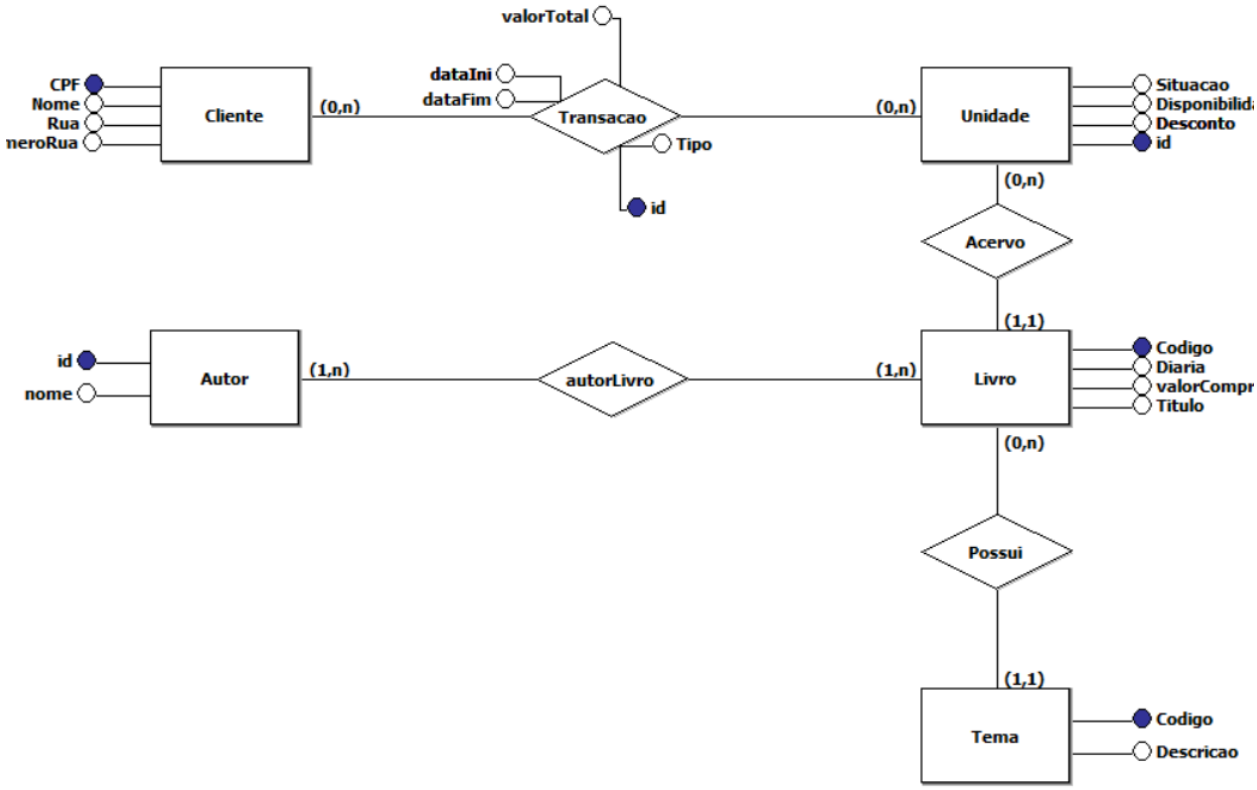
Trabalho de Laboratório de Banco de Dados

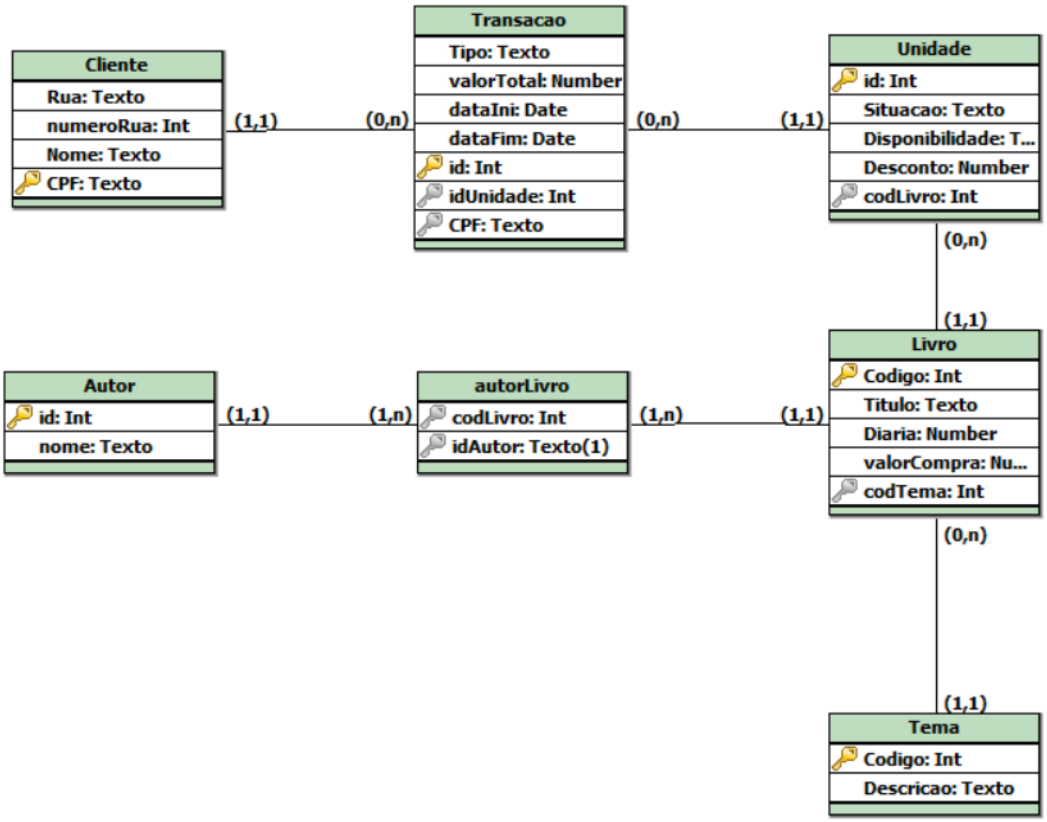
Grupo: Jean Bonadeo Dal Santo, Felipe Marostega Fagundes, Guilherme Silveira Machado

Tema: Editora de livros(venda e empréstimo)

1-
Modelo Conceitual



Modelo Lógico



Modelo Físico

```
CREATE TABLE Autor (
nome VARCHAR2(40),
id INT PRIMARY KEY
);

CREATE TABLE Tema (
descricao VARCHAR(40),
codigo INT PRIMARY KEY
);

CREATE TABLE Livro (
codigo INT PRIMARY KEY,
titulo VARCHAR2(40),
valorCompra NUMBER,
diaria NUMBER,
codTema INT,
FOREIGN KEY(codTema) REFERENCES Tema (codigo)
);
```

```
CREATE TABLE Unidade (
Id INT PRIMARY KEY,
disponibilidade VARCHAR2(15),
situacao VARCHAR2(10),
desconto NUMBER(10,2) default 0,
codLivro INT,
FOREIGN KEY(codLivro) REFERENCES Livro (codigo)
);

CREATE TABLE Cliente (
Nome VARCHAR2(50),
CPF VARCHAR(20) PRIMARY KEY,
Rua VARCHAR(50),
NumeroRua INT
);

CREATE TABLE AutorLivro (
codLivro INT,
idAutor INT,
FOREIGN KEY(codLivro) REFERENCES Livro (codigo),
FOREIGN KEY(idAutor) REFERENCES Autor (id)
);

CREATE TABLE Transacao (
id INT PRIMARY KEY,
tipo VARCHAR2(25),
dataIni DATE DEFAULT (CURRENT_DATE),
dataFim DATE,
valorTotal NUMBER,
IdUnidade INT,
CPF VARCHAR(20),
FOREIGN KEY (CPF) REFERENCES Cliente (CPF),
FOREIGN KEY(IdUnidade) REFERENCES Unidade (Id)
);
```

2- Carga de dados e duas consultas (envolvendo pelo menos 3 tabelas). Uma delas com uso de Having

INSERT INTO Autor(nome, id) VALUES('Mario Quintana', 1);
INSERT INTO Autor(nome, id) VALUES('Policarpo Quaresma', 2);
INSERT INTO Autor(nome, id) VALUES('Machado de Assis', 3);
INSERT INTO Autor(nome, id) VALUES('Mario Sergio Cortella', 4);
INSERT INTO Autor(nome, id) VALUES('Pero Vaz de Caminha', 5);
INSERT INTO Autor(nome, id) VALUES('Paulo Freire', 6);

INSERT INTO Tema(descricao, codigo) VALUES('Romance', 1);
INSERT INTO Tema(descricao, codigo) VALUES('Ação', 2);
INSERT INTO Tema(descricao, codigo) VALUES('Terror', 3);
INSERT INTO Tema(descricao, codigo) VALUES('Drama', 4);
INSERT INTO Tema(descricao, codigo) VALUES('Suspense', 5);
INSERT INTO Tema(descricao, codigo) VALUES('Ficçao', 6);

INSERT INTO Livro(codigo, titulo, valorCompra, diaria, codTema) VALUES(1, 'A Bruxa', 60, 2, 3);
INSERT INTO Livro(codigo, titulo, valorCompra, diaria, codTema) VALUES(2, 'Fight Club', 45.9, 1.2, 2);
INSERT INTO Livro(codigo, titulo, valorCompra, diaria, codTema) VALUES(3, 'De Volta pro Futuro', 49.9, 0.8,6);
INSERT INTO Livro(codigo, titulo, valorCompra, diaria, codTema) VALUES(4, 'O Hobbit', 50, 1.5, 6);
INSERT INTO Livro(codigo, titulo, valorCompra, diaria, codTema) VALUES(5, 'Hush', 19.9, 0.5, 3);
INSERT INTO Livro(codigo, titulo, valorCompra, diaria, codTema) VALUES(6, 'Como Treinar Seu Dragao', 24.5, 1.0, 6);
INSERT INTO Livro(codigo, titulo, valorCompra, diaria, codTema) VALUES(7, 'Interestelar', 99.5, 2.5, 4);
INSERT INTO Livro(codigo, titulo, valorCompra, diaria, codTema) VALUES(8, 'Marley e Eu', 30.5, 1.25, 4);

INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(1,'Indisponivel','Usado',1);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(2,'Indisponivel','Usado',1);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro) VALUES(3,'Disponivel','Novo',1);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro) VALUES(4,'Disponivel','Novo',2);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(5,'Indisponivel','Usado',2);

```
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro) VALUES(6,'Disponivel','Novo',2);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro) VALUES(7,'Disponivel','Novo',2);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro) VALUES(8,'Disponivel','Novo',3);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro) VALUES(9,'Disponivel','Novo',3);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(10,'Disponivel','Novo',3);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(11,'Indisponivel','Usado',4);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(12,'Disponivel','Novo',4);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(13,'Disponivel','Novo',4);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(14,'Disponivel','Novo',5);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(15,'Disponivel','Novo',5);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(16,'Disponivel','Novo',5);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(17,'Disponivel','Novo',6);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(18,'Disponivel','Novo',6);
INSERT INTO Unidade(id, disponibilidade, situacao, codLivro)
VALUES(19,'Disponivel','Novo',6);

INSERT INTO Cliente(CPF, Rua, NumeroRua, Nome) VALUES('234.543.464-06', 'Paissandu',
32, 'Zeca Fernandes');
INSERT INTO Cliente(CPF, Rua, NumeroRua, Nome) VALUES('343.353.756-76', 'Teixeira
Soares', 653, 'Paula Antunes');
INSERT INTO Cliente(CPF, Rua, NumeroRua, Nome) VALUES('643.623.764-43', 'Brigadeiro
Faria Lima', 332, 'Humberto Carlos Andrade');
INSERT INTO Cliente(CPF, Rua, NumeroRua, Nome) VALUES('453.667.353-64', 'Tiradentes',
782, 'Sergio Augusto');
INSERT INTO Cliente(CPF, Rua, NumeroRua, Nome) VALUES('674.765.234-27', 'Dom Pedro',
112, 'Ines da Silva');
INSERT INTO Cliente(CPF, Rua, NumeroRua, Nome) VALUES('454.753.654-68', 'Princesa
Isabel', 232, 'Marcio Cesar');
INSERT INTO Cliente(CPF, Rua, NumeroRua, Nome) VALUES('743.232.876-87', 'Augusta',
532, 'Arthur Paulo');

INSERT INTO AutorLivro(codLivro, idAutor) VALUES(1, 1);
INSERT INTO AutorLivro(codLivro, idAutor) VALUES(2, 2);
INSERT INTO AutorLivro(codLivro, idAutor) VALUES(3, 3);
INSERT INTO AutorLivro(codLivro, idAutor) VALUES(4, 4);
```

```
INSERT INTO AutorLivro(codLivro, idAutor) VALUES(5, 5);
INSERT INTO AutorLivro(codLivro, idAutor) VALUES(6, 6);
INSERT INTO AutorLivro(codLivro, idAutor) VALUES(7, 1);
INSERT INTO AutorLivro(codLivro, idAutor) VALUES(8, 2);

INSERT INTO Transacao(id, tipo, dataFim, valorTotal, IdUnidade, CPF) VALUES(1, 'Aluguel',
'30/06/2024', 50, 1, '234.543.464-06');
INSERT INTO Transacao(id, tipo, valorTotal, IdUnidade, CPF) VALUES(2, 'Venda', 45.9, 2,
'343.353.756-76');
INSERT INTO Transacao(id, tipo, valorTotal,  IdUnidade, CPF) VALUES(3, 'Aluguel', 65, 5,
'453.667.353-64');
INSERT INTO Transacao(id, tipo, valorTotal, IdUnidade, CPF) VALUES(4, 'Venda', 50, 11,
'674.765.234-27');


--SELECT USANDO HAVING, ESSE SELECT É PARA MOSTRAR OS LIVROS QUE
POSSUEM MAIS DE 2 UNIDADES NO NOSSO ESTOQUE;
SELECT livro.titulo, count(unidade.id)
FROM Livro
INNER JOIN unidade on livro.codigo = unidade.codLivro
GROUP BY livro.titulo
HAVING count(unidade.id) > 2;

--SELECT PARA MOSTRAR TODAS AS UNIDADES DOS LIVROS DO AUTOR 'MARIO
QUINTANA';
SELECT unidade.id, livro.titulo, unidade.situacao
FROM unidade
INNER JOIN livro on unidade.codlivro = livro.codigo
INNER JOIN autorlivro on livro.codigo = autorlivro.codlivro
INNER JOIN autor on autorlivro.idautor = autor.id
WHERE autor.nome = 'Mario Quintana';
```

3. Criação de uma Função (fazer a descrição e propor a solução)

```
--FUNÇÃO COM O OBJETIVO DE MOSTRAR A QUANTIDADE TOTAL DE LIVROS
DISPONÍVEIS PARA COMPRA/ALUGUEL EM NOSSO ESTOQUE;
CREATE or replace function f_quantLivrosdisp
   return VARCHAR is
   varCountLivros VARCHAR(10);
   begin
      SELECT count(unidade.id) INTO varCountLivros FROM unidade
      WHERE  unidade.disponibilidade = 'Disponivel'
      GROUP BY unidade.disponibilidade;
```

```
    RETURN varCountLivros;
  END f_quantLivrosdisp;

 --select f_quantLivrosdisp() from DUAL;
```

--FUNÇÃO PARA MOSTRAR A QUANTIDADE DE UNIDADES DISPONÍVEIS A PARTIR DO NOME DE ALGUM LIVRO;

```
CREATE or replace FUNCTION f_achaLivro(varNome VARCHAR)
  return VARCHAR is
  varlivro Number; erro1 EXCEPTION;
  begin
    SELECT count(livro.titulo) into varlivro
    FROM livro
    INNER JOIN unidade on  livro.codigo = unidade.codlivro
    WHERE livro.titulo = varNome and unidade.disponibilidade = 'Disponivel';

    IF (varlivro = 0) THEN
       raise erro1;
    END IF;
    RETURN 'livros Disponiveis: '||varlivro;
    EXCEPTION
      WHEN erro1 THEN
       raise_application_error(-20898,'Livro nao encontrado');

  END f_achaLivro;

  --select f_achaLivro('A  irmandade') from Dual;
  --select f_achaLivro('Como Treinar Seu Dragao') from Dual;
```

4. Criação de um Procedure que faça uso de uma função (fazer a descrição e propor a solução)

```
--FUNÇÃO USADA NA PROCEDURE, ESSA FUNÇÃO RETORNA O VALOR TOTAL QUE O CLIENTE IRA PAGAR PARA ALUGAR X LIVRO EM Y DIAS E CONTA TAMBÉM O DESCONTO CASO O LIVRO FOR USADO;
CREATE or replace FUNCTION f_calculapreco(vardataini date, vardataFim date, variduni int)
    return number is
    varValor number;
    varsit varchar(5);
    begin
        SELECT situacao INTO varsit FROM unidade WHERE id = variduni;
        SELECT diaria INTO varValor FROM livro INNER JOIN unidade on livro.codigo = unidade.codlivro WHERE unidade.id = variduni;
        varValor:= varValor * (vardataFim - vardataini);
        if(varsit = 'Usado') then
            varValor := varValor - (varValor * 0.1);
        end if;
        return varValor;
    END f_calculapreco;



--PROCEDURE PARA REALIZAR O INSERT DE ALGUMA TRANSACAO, FAZENDO TODOS OS TESTES NECESSÁRIOS(TIPO DE TRANSACAO, PRECO A PAGAR, E ATUALIZAÇÃO DA SITUAÇÃO DA UNIDADE EM NOSSO ESTOQUE;
CREATE or replace PROCEDURE p_inseretransacao(varcod int, varDataIni date, varDataFim date, varTipo varchar, varIdUnidade int, varCpf varchar) IS
   varpreco number;
   vardisp varchar(15);
    varsit varchar(5);
    varDesconto number(10,2);

   BEGIN
    SELECT disponibilidade INTO vardisp FROM unidade WHERE id = varIdUnidade;
    SELECT situacao INTO varsit FROM unidade WHERE id = varidunidade;
    SELECT desconto into varDesconto from unidade where id = varidunidade;
    if(vardisp = 'indisponivel' and varTipo <> 'Devolucao')then
        DBMS_OUTPUT.PUT_LINE('!!!Livro Indisponível!!!');
    else
         if(varTipo = 'Compra') then
            SELECT livro.valorcompra INTO varpreco FROM livro INNER JOIN unidade on livro.codigo = unidade.codlivro WHERE unidade.id = varidunidade;
```

```
        if(varsit = 'Usado') then
            varpreco := varpreco - vardesconto;
        elsif(varsit = 'Doacao') then
             varpreco := 0.0;
        end if;
       elsif(varTipo = 'Aluguel') then
          varpreco:= f_calculapreco(varDataIni, varDataFim, varIdUnidade);
       else
        varpreco:= 0;
     end if;

      INSERT INTO transacao(id, dataini,tipo,datafim,valortotal,idunidade,cpf)
      VALUES(varcod,varDataIni,varTipo, varDataFim,varpreco,varidUnidade,varCpf);

     if(varTipo = 'Aluguel') then
      UPDATE UNIDADE
      SET disponibilidade = 'Indisponivel', situacao = 'Usado'
      WHERE id = varIdUnidade;
    elsif(varTipo = 'Compra') then
     UPDATE UNIDADE
      SET disponibilidade = 'Indisponivel', situacao = 'Vendido'
      WHERE id = varIdUnidade;
     end if;
  end if;
  EXCEPTION
     WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE(SQLCODE);
      DBMS_OUTPUT.PUT_LINE(SQLERRM);
      RAISE;

  END  p_inseretransacao;




--exec p_inseretransacao(63, '07/05/2024', '07/05/2024', 'Compra',17, '743.232.876-87');
--exec p_inseretransacao(30, '07/04/2024', '07/05/2024', 'Aluguel',6, '743.232.876-87');
--exec p_inseretransacao(32, '07/04/2024', '07/05/2024', 'Devolucao',6, '743.232.876-87');
```

5. Criação de um Procedure que faça uso de um Cursor (fazer a descrição e propor a solução)

```
   --PROCEDURE UTILIZANDO CURSOR PARA MOSTRAR OS DADOS DOS LIVROS
DISPONIVEIS(TITULO, AUTOR, TEMA, VALOR PARA COMPRA E QUANTIDADE DE
UNIDADES DISPONIVEIS);
   CREATE or replace PROCEDURE p_CursorMostraLivros IS
     Cursor c1 IS

     SELECT livro.titulo, autor.nome, tema.descricao, livro.valorcompra,
count(unidade.disponibilidade) as Unidades
     FROM livro
     INNER JOIN unidade on livro.codigo = unidade.codlivro
     INNER JOIN tema on livro.codtema = tema.codigo
     INNER JOIN autorlivro on livro.codigo = autorlivro.codlivro
     INNER JOIN autor on autor.id = autorlivro.idautor
     WHERE unidade.disponibilidade = 'Disponivel'
     GROUP BY livro.titulo, tema.descricao, livro.valorcompra, autor.nome
     ORDER BY count(unidade.disponibilidade) DESC;

     registro c1%rowtype;
   BEGIN
     OPEN c1;
     DBMS_OUTPUT.PUT_LINE('LIVROS DISPONÍVEIS:');
     LOOP
     FETCH c1 INTO registro;
     EXIT WHEN c1%NOTFOUND;
       DBMS_OUTPUT.PUT_LINE('LIVRO: ' || registro.titulo || ', AUTOR: ' || registro.nome || ',
TEMA: ' || registro.descricao ||
       ', VALOR COMPRA: ' || registro.valorcompra || ' UNIDADES: ' || registro.unidades );
     END LOOP;
     DBMS_OUTPUT.PUT_LINE('Esses são os livros disponiveis!');
     CLOSE c1;

   END p_CursorMostraLivros;


   exec p_CursorMostraLivros;
```

6. Análise o contexto da aplicação e programe uma regra de negócio em um Trigger (fazer a descrição e propor a solução)

-- TRIGGER COM O PROPÓSITO DE ATUALIZAR O VALOR DE DESCONTO CONFORME A UNIDADE É ALUGADA, AUMENTANDO SUCESSIVAMENTE O VALOR DO DESCONTO;

```
CREATE or replace TRIGGER tg_updateDesconto
  AFTER insert on transacao
  FOR EACH ROW
  WHEN (NEW.tipo = 'Aluguel')
  DECLARE
  varValorComp NUMBER(10,2);
     varDesconto number(10,2);
  BEGIN
    select livro.valorcompra into varvalorcomp from livro inner join unidade on livro.codigo =
unidade.codlivro where unidade.id = :NEW.idunidade;
    SELECT desconto into varDesconto from unidade where id = :NEW.idunidade;
    IF (varDesconto >= varvalorcomp) then
      update unidade set desconto = 0, situacao = 'Doacao' where id = :NEW.idunidade;
    elsif (varDesconto = 0) then
      update unidade set desconto = 1 where id = :NEW.idunidade;
    else
      UPDATE unidade SET desconto =  desconto + (varDesconto * 0.05)  WHERE id =
:NEW.idunidade;
    end if;

END tg_updateDesconto;


select * from transacao
select * from livro
select * from unidade



CREATE or replace TRIGGER tg_setDevolucao
  AFTER insert on transacao
  FOR EACH ROW
  WHEN (NEW.tipo = 'Devolucao')

  BEGIN
    UPDATE unidade SET disponibilidade =  'Disponivel' WHERE id = :NEW.idunidade;

 END tg_setDevolucao;
```