Parte 3 – Implementação de Funções e Gatilhos

Alunos:
Henrique Sant'Anna de Faria
Wyllen Brito da Silva

1. Função para cadastro de um livro
2. Função para empréstimo de um livro
3. Gatilho para impedir o empréstimo de um livro emprestado
4. Gatilho para impedir o empréstimo de um livro a um usuário que não tiver livros atrasados, multas pendentes e se não for excedido o nº de livros emprestados que o usuário tem direito.
5. Função para verificar qual usuário está com um determinado livro
6. Gatilho para impedir o empréstimo de um livro da coleção reserva

```
--1. Função para cadastro de um livro

insert into livros values (1,'9780545010221','Livro 1','autor 1','editora
1','Reserva');
insert into exemplares values (1,1,1,'A-001')

select * from assistente
insert into assistente values (1,'Assistente 1');

select * from livros
select * from exemplares

create or replace function CadastraLivro(F_isbn varchar, F_titulo varchar
,F_autor varchar , F_editora varchar ,F_colecao varchar ,F_localizacao
varchar )
    returns void as
$$
declare
    Next_Id_Livro int default 0;
    Next_Id_Exemplar int default 0;
    Id_Assistente int default 0;
    vLinhas int default 0;
begin
    select max(id) from livros into Next_Id_Livro;
    if(Next_Id_Livro < 1) then
        raise exception 'Não há livros na tabela livros';
        return;
    end if;
    Next_Id_Livro := Next_Id_Livro + 1;
    insert into livros values
        (Next_Id_Livro,F_isbn,F_titulo,F_autor,F_editora,F_colecao);
    get diagnostics vLinhas = row_count;
```

```
    if(vLinhas < 1) then
        raise exception 'Erro ao inserir livro na tabela livros';
        return;
    end if;

    select max(id) from exemplares into Next_Id_Exemplar;
    if(Next_Id_Exemplar < 1) then
        raise exception 'Não há Exemplares na tabela Exemplares';
        return;
    end if;
    Next_Id_Exemplar := Next_Id_Exemplar + 1;

    select max(cod_a) from assistente into Id_Assistente;
    if(Id_Assistente < 1) then
        raise exception 'Não há assistentes na tabela Assistentes';
        return;
    end if;

    insert into exemplares values
(Next_Id_Exemplar,Next_Id_Livro,Id_Assistente,F_localizacao);
    get diagnostics vLinhas = row_count;

    if(vLinhas > 0) then
        raise notice 'SUCESSO: Livro e exemplar inserido com sucesso!';
            return;
        else
        raise notice 'ERRO: Exemplar não inserido';
            return;
    end if;

end;
$$
language plpgsql;


select CadastraLivro('12345678', 'Livro 2 teste func' ,'Autor 2 teste
func' , 'Editora 2 teste' ,'Premium Colec' ,'A-101' )
select * from livros
select * from exemplares




-------------------------------------------------
--2. Funcao para emprestimo de um livro


select * from usuarios
```

```
select * from exemplares
select * from emprestimo

insert into usuarios values (1,'Usuario 1','Endereco 1','telefone
1',1,NULL,'Aluno Grad','Integral')
insert into emprestimo values (1,1,1,'2023-05-01',null,'2023-05-15');

create or replace function Empresta_Livro(id_usuario int,id_exemplar int)
returns void as
$$
declare
    next_id_emprestimo int default 0;
    data_emprestimo date := CURRENT_DATE;
    data_devolucao date := date_trunc('day', data_emprestimo + interval
'15 day');
    vLinhas int default 0;
begin
    select max(id) from emprestimo into next_id_emprestimo;
    if(next_id_emprestimo < 1 ) then
        raise exception 'não há emprestimos na tabela emprestimos';
        return;
    end if;
    next_id_emprestimo := next_id_emprestimo + 1;
    insert into emprestimo values
(next_id_emprestimo,id_usuario,id_exemplar,data_emprestimo,null,data_devo
lucao);
    get diagnostics vLinhas = row_count;

    if(vLinhas > 0) then
        raise notice 'SUCESSO: Emprestimo feito!';
            return;
        else
        raise notice 'ERRO: Emprestimo não autorizado';
            return;
    end if;
end;
$$
language plpgsql;


select Empresta_Livro(1,1)
select * from emprestimo

select
select * from reservas
select * from emprestimo
```

```
-----------------------------------------------------
-- 3. Gatilho para impedir o empréstimo de um livro emprestado

create or replace function verificaLivro_emprestado() returns trigger as
$$
begin
     IF EXISTS (
        SELECT * FROM emprestimo
        WHERE id_exemplar = NEW.id_exemplar
     ) THEN
        RAISE EXCEPTION 'O livro já está emprestado';
    ELSE
        RETURN NEW;
    END IF;
end;
$$
language plpgsql;

select * from emprestimo
select * from exemplares
select * from usuarios

select Empresta_Livro(1,1)  --livro que já está emprestado. irá lançar o
gatilho de "livro já emprestado"

create trigger verificaLivro_emprestado before insert or update on
emprestimo
for each row execute procedure verificaLivro_emprestado();




-----------------------------------------------------
+++++++++++++++++++++++++++++++++++++++++++++++++++++
-- 4. Gatilho para impedir o empréstimo de um livro a um usuário que
tiver livros atrasados, multas pendentes e se não for excedido o nº de
livros
-- emprestados que o usuário tem direito. (3 livros)

select count(*)  from emprestimo
    where id_user = 1 and id_exemplar = 1

create or replace function emprestimo_usuario_valido()
returns trigger as $$
declare
    livros_atrasados int default 0;
    multas_pendentes float default 0.0;
    livros_emprestados_usuario int default 0;
```

```
begin
    select count(*) from emprestimo
    where id_user = new.id_user and data_dev < current_date into
livros_atrasados;

    select sum(multas)  from emprestimo
    where id_user = new.id_user and data_dev < current_date into
multas_pendentes;

    select count(*) from (select count(*)  from emprestimo
    where id_user = new.id_user group by(id_exemplar))
        as quant_livrosEmprestados into livros_emprestados_usuario;

    if livros_atrasados > 0 then
        raise exception 'usuário com livros atrasados não pode realizar
empréstimos';
    elsif multas_pendentes > 0 then
        raise exception 'usuário com multas pendentes não pode realizar
empréstimos';
    elsif livros_emprestados_usuario >= 3 then
        raise exception 'usuário já atingiu o limite máximo de
empréstimos (3 Livros por usuario)';
    else
        return new;
    end if;
end;
$$ language plpgsql;

create trigger emprestimo_usuario_valido
before insert or update on emprestimo
for each row execute procedure emprestimo_usuario_valido();

select Empresta_Livro(1,4)
select Empresta_Livro(1,5)
select * from emprestimo




insert into usuarios values (2,'Usuario 2','Endereco 2','telefone
2',1,NULL,'Aluno Grad','Integral')
select CadastraLivro('7864423', 'Livro 3 teste func' ,'Autor 3 teste
func' , 'Editora 3 teste' ,'Premium Colec' ,'A-105' )
select CadastraLivro('7864423', 'Livro 4 teste func' ,'Autor 3 teste
func' , 'Editora 3 teste' ,'Premium Colec' ,'A-105' )
select CadastraLivro('7864423', 'Livro 5 teste func' ,'Autor 3 teste
func' , 'Editora 3 teste' ,'Premium Colec' ,'A-105' )

select * from livros
```

```
select Empresta_Livro(2,3)



++++++++++++++++++++++++++++++++++++++++
--5. Função para verificar qual usuário está com um determinado livro

create or replace function procura_Usuario_Exemplar(F_id_exemplar int)
returns int as
$$
declare
    vId_usuario int default 0;
begin
    select distinct id_user from emprestimo where id_exemplar =
F_id_exemplar into vId_usuario;

    if(vId_usuario < 0) then
        raise exception 'Não há usuários com esse exemplar.';
        return 0;
    else
        return vId_usuario;
    end if;


end;
$$
language plpgsql;

select procura_Usuario_Exemplar(3)
select * from emprestimo



++++++++++++++++++++++++++++++++++++++++
----------------------------------------

---6. Gatilho para impedir o empréstimo de um livro da coleção reserva.

create or replace function verificaReserva() returns trigger as
$$
declare
    vColecao varchar;
begin
    if exists (
        select * from livros
        where id = new.id_exemplar and colecao = 'Reserva'
    ) then
        raise exception 'livro da coleção reserva não pode ser
emprestado';
```

```
    else
        return new;
    end if;
end;
$$
language plpgsql;

create trigger verificaReserva before insert or update on emprestimo
for each row execute procedure verificaReserva();

select CadastraLivro('7864423', 'Livro 6' ,'Autor 3 teste func' ,
'Editora 3 teste' ,'Reserva' ,'A-110' )
select CadastraLivro('7864423', 'Livro 7' ,'Autor 3 teste func' ,
'Editora 3 teste' ,'Emprestado' ,'A-110' )
select * from livros
select Empresta_Livro(2,6) -- vai impedir o emprestimo da colecao Reserva
select Empresta_Livro(2,7) -- vai permitir o emprestimo da colecao
Emprestado
```