Trabalho 6 Relatório

SCC-541 Laboratório de Bases de Dados

Leonardo Gonçalves Chahud - 5266649 Murilo Franchi - 9790760 Rafael Dantas - 12563686

Prof. Dr Caetano Traina Jr. PAE: Igor Alberte R. Eleutério

Exercício 1

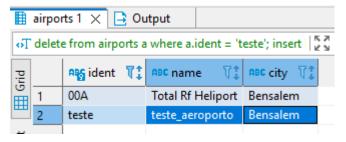
Na tabela **AIRPORTS**, há vários Aeroportos cujo atributo **City** apresenta um nome que não aparece na tabela **GEOCITIES15K** no atributo **name**. A equipe responsável pelo projeto decidiu que, em relação aos dados já persistidos, nada irá ser feito. No entanto, para novos aeroportos inseridos na base, será obrigatório vinculá-los a uma cidade. Sua equipe foi designada para desenvolver uma trigger que avalia se o atributo **City** de um aeroporto sendo inserido (ou modificado) na tabela **AIRPORTS** não encontra correspondência com o atributo **Name** da tabela **GEOCITIES15K**. Quando isso acontecer, a operação não deve ser concluída e uma exceção deve ser lançada.

- (a) Nome da função: VerificaAeroporto
- (b) Nome do trigger : TR_Airports
- (c) Mensagem da exceção: 'Cidade não encontrada! Operação cancelada.'

Faça ao menos um teste para inserção e um para atualização, e mostre-os no relatório junto do resultado.

```
create or replace trigger TR_airports
  before insert or update on airports
  for each row
  execute function VerificaAeroporto();
```

```
--teste
delete from airports a
     where a.ident = 'teste';
insert into airports (ident, name, city) values
      ('teste', 'teste_aeroporto', 'teste_cidade');
--retorna 'Cidade não encontrada! Operação cancelada.'
--e a operacao e cancelada, pois nao existe
--cidade chama teste_cidade
--na tabela geocities15k
insert into airports (ident, name, city) values
      ('teste', 'teste_aeroporto', 'Bensalem');
--operacao e sucedida, pois existe uma cidade
--chamada Bensalem
update airports
     set city = 'teste'
     where ident = '00A';
--retorna 'Cidade não encontrada! Operação cancelada.'
--e a operacao e cancelada, pois nao existe
--cidade chamada teste
--na tabela geocitites15k
select a.ident, a.name, a.city
     from airports a
     where a.ident = 'teste' or a.ident = '00A';
```



Exercício 2

Execute o SEGUINTE script e faça o que se pede.

```
CREATE TABLE Results_Status (
StatusID INTEGER PRIMARY KEY,
Contagem INTEGER,
FOREIGN KEY (StatusID) REFERENCES Status(StatusID)
);
INSERT INTO Results_Status
SELECT S.StatusId , COUNT (*)
FROM Status S
JOIN Results R ON R.StatusID = S.StatusID
GROUP BY S.StatusId , S.Status;
```

RESPONDA: O que esse script faz?

Cria uma tabela com a contagem de cada tipo de status gerado em cada corrida, através da junção entre as tabelas de results e status.

Depois, crie uma única trigger para as questões (a), (b) e (c) e outra para a questão (d). A trigger das três primeiras questões deve se chamar **TR_ResultsStatus**, e a função associada **AtualizaContagem**; A trigger da questão (d) deve se chamar **TR_Results** e a função relacionada **VerificaStatus**:

- (a) Ao inserir novas tuplas na tabela **Results**, incremente as quantidades inseridas nos respectivos **status** na tabela **Results_Status**. Além disso, mostre na tela a mensagem 'StatusID: <status modificado>, Contagem: <nova contagem>.'
- (b) Ao remover tuplas da tabela **Results**, diminua as quantidades removidas nos respectivos **status** da tabela **Results_Status**. Além disso, mostre na tela a mensagem 'StatusID: <status modificado>, Contagem: <nova contagem>.'
- (c) Ao atualizar tuplas da tabela **Results** (especificamente o atributo **statusid**), altere, na tabela **Results_Status** as quantidades removidas nas respectivas escuderias. Além disso, mostre na tela a mensagem
- 'StatusID Anterior: <status anterior>, Contagem: <nova contagem>' e
 'StatusID Atual: <status atual>, Contagem: <nova contagem>,
 em que a primeira mensagem é o **Status** que teve a quantidade subtraída, e a segunda se
 refere à contagem que teve a quantidade aumentada.
- (d) Antes de inserir novas tuplas na tabela **Results** ou atualizar o **Statusid** de alguma delas, verifique se o **Statusid** fica negativo. Caso fique, levante uma exceção com a mensagem 'StatusID Negativo! Operação cancelada.' e não execute a operação.

Execute comandos para testar cada um dos casos e apresente os testes e resultados no relatório.

A,B,C):

```
create or replace function AtualizaContagem()
    returns trigger as $$
   declare cont int;
   begin
      if TG OP = 'INSERT' then
             update results_status
                   set Contagem = Contagem + 1
                   where statusId = new.StatusID;
            select into cont rs.contagem
                  from results status rs
                  where rs.statusid = new.statusid;
            raise notice 'StatusID: %, Contagem: %', new.StatusID, cont;
      end if;
      if TG_OP = 'DELETE' then
             update results_status rs
                   set Contagem = Contagem - 1
                   where statusId = old.StatusID;
             select into cont rs.contagem
                  from results status rs
                  where rs.statusid = old.statusid;
             raise notice 'StatusID: %, Contagem: %', old.StatusID,
cont;
      end if;
      if TG OP = 'UPDATE' then
            update results_status rs
                  set Contagem = Contagem + 1
                  where statusId = new.StatusID;
            select into cont rs.contagem
                  from results_status rs
                  where rs.statusid = new.statusid;
            raise notice 'StatusID atual: %, Contagem: %', new.StatusID,
cont;
            update results_status rs
                  set Contagem = Contagem - 1
                  where statusId = old.StatusID;
            select into cont rs.contagem
                  from results_status rs
                  where rs.statusid = old.statusid;
            raise notice 'StatusID anterior: %, Contagem: %',
old.StatusID, cont;
      end if;
      return null;
   end;
   $$ language plpgsql;
```

```
create or replace trigger TR_ResultsStatus
  after insert or delete or update on results
  for each row
  execute function AtualizaContagem();
```

```
-- Testes
-- Inicial
select * from results_status order by statusid

-- Inserindo resultado

insert into results (resultid, statusid) values (25906, 1);
insert into results (resultid, statusid) values (259067, 1);

select * from results_status order by statusid;

-- Atualizando resultado
update results
    set statusid = 2
    where resultid = 25906;
select * from results_status order by statusid;

-- Deletando resultado
delete from results
    where resultid = 25906 or resultid = 259067;
```

d)

```
CREATE OR REPLACE FUNCTION VerificaStatus()
RETURNS TRIGGER AS $$
BEGIN

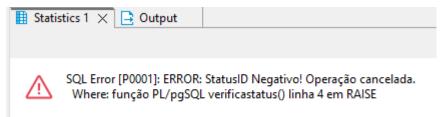
IF NEW.statusid < 0 THEN

RAISE EXCEPTION 'StatusID Negativo! Operação cancelada.';
END IF;

RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER TR_Results
BEFORE INSERT OR UPDATE ON Results
FOR EACH ROW
EXECUTE FUNCTION VerificaStatus();
```

```
--teste
insert into results (resultid, statusid) values (259060, -150);
```



Error position: