Base de Dados

Parte 4

Grupo 30 - Turno BD817957L02

Docente - Ricardo Eugénio Proença Rodrigues

Nome/Número	% de trabalho realizado	Total de horas
Duarte Faria 79856	50%	9
Vasco Morganho 81920	50%	9

Restrições de Integridade

a)

```
drop trigger if exists verifica_coordenador_trigger on Solicita;
drop function if exists verifica_coordenador();

create or replace function verifica_coordenador() returns trigger as $$
    declare local1 varchar (255);
    declare local2 varchar (255);
    declare vid integer;
begin

    if new.idCoordenador not in (select idCoordenador from Coordenador) then
        raise exception 'O coordenador % nao existe.', new.idCoordenador;
    end if;

    if new.idCoordenador not in (select idCoordenador from Audita) then
        raise exception 'O coordenador % nunca auditou.', new.idCoordenador;
    end if;

    select numCamara into vid
    from Video
    where Video.numCamara = new.numCamara;

    if vid is null then raise exception 'Esta camara nao tem videos';
    end if;

    select moradaLocal into local1
    from EventoEmergencia inner join Audita on EventoEmergencia.numProcessoSocorro = Audita.numProcessoSocorro
    where Audita.idCoordenador = new.idCoordenador;

    if local1 is null then raise exception 'O coordenador nao auditou.';
    end if;

    select moradaLocal into local2
    from Vigia
    where Vigia.numCamara = new.numCamara;

    if local2 is null then raise exception 'Nao existe camara nesse local.';
    end if;

    if local1 <> local2 then raise exception 'O coordenador nao auditou nenhum accionamento cujo evento ocorreu
                                              num local vigiado pela camara solicitada';
    end if;
    return new;

    drop table if exists local1;
    drop table if exists local2;
    drop table if exists vid;

End;
$$ Language plpgsql;

create trigger verifica_coordenador_trigger before insert on Solicita
for each row execute procedure verifica_coordenador();
```

b)

```
drop trigger if exists verifica_meio_trigger on Alocado;
drop function if exists verifica_meio();

create or replace function verifica_meio() returns trigger as $$
    declare novoMeio integer;
    declare novoProcesso integer;
    declare mE integer;
begin
    if new.numMeio not in (select numMeio from Acciona) then
        raise exception 'O meio % nao foi accionado.', new.numMeio;
    end if;

    if new.numProcessoSocorro not in (select numProcessoSocorro from Acciona) then
        raise exception 'O processo de socorro % nao foi accionado.', new.numProcessoSocorro;
    end if;

    select numMeio into mE
    from Acciona
    where Acciona.numMeio = new.numMeio AND Acciona.nomeEntidade = new.nomeEntidade;

    if mE is null then raise exception 'numMeio e nomeEntidade inconsistentes.';
    end if;

    select numMeio, numProcessoSocorro into novoProcesso
    from Acciona
    where Acciona.numMeio = new.numMeio and Acciona.numProcessoSocorro = new.numProcessoSocorro;

    if novoProcesso is null then raise exception 'Este processo nao accionou este meio.';
    end if;

    return new;

    drop table if exists novoMeio;
    drop table if exists novoProcesso;
    drop table if exists mE;

End;
$$ Language plpgsql;

create trigger verifica_meio_trigger before insert on Alocado
for each row execute procedure verifica_meio();
```

Índices

1-

a) Fazer um índice composto do tipo hash para o numCamara e moradaLocal na tabela Vigia e um índice do tipo hash do numCamara da tabela Video. Escolhemos o hash em vez do b+tree porque índices do tipo hash são mais apropriados para verificação de igualdades enquanto que índices do tipo b+tree são mais eficazes para ordenações..

b) CREATE INDEX vigia_idx ON Vigia(numCamera,moradaLocal) USING hash;
CREATE INDEX video_idx ON Video(numCamera) USING hash;

2-

a) Fazer um índice do tipo hash para o numProcessoSocorro na tabela Transporta, um índice do tipo hash para o numProcessoSocorro na tabela EventoEmergencia e um índice composto do tipo hash para o numTelefone e instanteChamada na tabela EventoEmergencia

b) CREATE INDEX transporta_idx ON Transporta(numProcessoSocorro) USING hash;
CREATE INDEX numP_idx ON EventoEmergencia(numProcessoSocorro) USING hash;
CREATE INDEX evento_idx ON EventoEmergencia(numTelefone, instanteChamada) USING hash;

Modelo Multidimencional

```
DROP TABLE IF EXISTS d_evento CASCADE;

CREATE TABLE if not exists d_evento(
numTelefone varchar(255) not null,
instanteChamada timestamp not null,
idEvento Serial,
primary key(idEvento),
foreign key (numTelefone, instanteChamada) references EventoEmergencia(numTelefone, instanteChamada) ON DELETE CASCADE ON UPDATE CASCADE);

insert into d_evento
SELECT numTelefone, instanteChamada
from EventoEmergencia;

-----------------------------------------------------------------------------------------------
DROP TABLE IF EXISTS d_meio CASCADE;

CREATE TABLE if not exists d_meio(
numMeio integer not null,
nomeEntidade varchar(255) not null,
tipo varchar(255) not null,
idMeio Serial,
primary key(idMeio),
foreign key(numMeio, nomeEntidade) references Meio(numMeio, nomeEntidade) ON DELETE CASCADE ON UPDATE CASCADE);

insert into d_meio
    select numMeio, nomeEntidade, 'Apoio'
    from meioApoio NATURAL JOIN Meio UNION
    select numMeio, nomeEntidade, 'Socorro'
    from meioSocorro NATURAL JOIN Meio UNION
    select numMeio, nomeEntidade, 'Combate'
    from meioCombate NATURAL JOIN Meio UNION
    select numMeio, nomeEntidade, 'Nao expecificado'
    from (select numMeio, nomeEntidade
            from Meio except
            (select numMeio, nomeEntidade
                from meioApoio NATURAL JOIN Meio UNION
            select numMeio, nomeEntidade
                from meioCombate NATURAL JOIN Meio UNION
            select numMeio, nomeEntidade
                from meioSocorro NATURAL JOIN Meio)) as t1;
```

```
DROP TABLE IF EXISTS d_tempo CASCADE;

CREATE TABLE if not exists d_tempo(
dia integer not null,
mes integer not null,
ano integer not null,
idTempo Serial,
primary key(idTempo));

insert into d_tempo
    select date_part('day', instanteChamada), date_part('month', instanteChamada), date_part('year', instanteChamada)
    from d_evento;



drop table if exists factos CASCADE;

CREATE TABLE if not exists factos(
idEvento integer,
idMeio integer,
idTempo integer,
idFactos Serial,
primary key(idFactos),
foreign key(idEvento) references d_evento(idEvento) ON DELETE CASCADE ON UPDATE CASCADE,
foreign key(idMeio) references d_meio(idMeio) ON DELETE CASCADE ON UPDATE CASCADE,
foreign key(idTempo) references d_tempo(idTempo) ON DELETE CASCADE ON UPDATE CASCADE);

insert into factos
    select idEvento, idMeio, idEvento
    from d_evento natural join d_meio natural join Acciona natural join EventoEmergencia UNION
    select idEvento, null, idEvento
    from d_evento except (select idEvento, null, idEvento
                          from d_evento natural join d_meio natural join Acciona natural join EventoEmergencia) UNION
    select null, idMeio, null
    from d_meio except (select null, idMeio, null
                        from d_evento natural join d_meio natural join Acciona natural join EventoEmergencia)
    ;
```

Data Analytics

```
select tipo, ano, mes, count(idMeio) as nMeios
        from factos natural join d_meio natural join d_tempo
        where factos.idEvento = 15
group by tipo, ano, mes
UNION
select tipo, ano, null, count(idMeio) as nMeios
        from factos natural join d_meio natural join d_tempo
        where factos.idEvento = 15
group by tipo, ano
UNION
select tipo, null, null, count(idMeio) as nMeios
        from factos natural join d_meio natural join d_tempo
        where factos.idEvento = 15
group by tipo
UNION
select null, null, null, count(idMeio) as nMeios
        from factos natural join d_meio natural join d_tempo
        where factos.idEvento = 15
order by tipo, ano, mes;
```