Grupo: Felipe Ferreira e Francisco Cavasan

Procedure inserção dados

1 ) Insere os valores dos números das salas

create or replace function adiciona\_sala()

returns integer

as $$

declare contador int = 1;

begin

while contador <= 15 loop

insert into sala values(contador);

contador = contador + 1;

end loop;

return 0;

end;

$$ language plpgsql;

select \* from adiciona\_sala()

2 ) Insere os valores da relação professor-turma-disciplina

create or replace function cria\_prof\_turma\_disc()

returns integer

as $$

declare contador int = 1;

begin

while contador <= 15 loop

insert into turma\_prof\_disc values(contador,contador,contador);

contador = contador + 1;

end loop;

return 0;

end;

$$ language plpgsql;

select \* from cria\_prof\_turma\_disc()

Visões

1) Retorna o nome, código e data de nascimento de cada aluno

CREATE VIEW v\_alunos

AS

SELECT ta.codaluno, al.nome, al.dt\_nascimento

FROM turma\_aluno ta INNER JOIN aluno al ON ta.codaluno=al.codaluno

group by ta.codturma, al.codaluno,ta.codaluno

2)Retorna o código, nome, sexo e nota de cada aluno

CREATE VIEW v\_alunos\_notas

AS

SELECT al.codaluno, al.nome,al.sexo,na.nota

FROM aluno al INNER JOIN nota\_aluno na ON al.codaluno=na.codaluno

order by al.codaluno

3) Retorna o código de todas as turmas que tiveram provas, juntamente com o código da prova e a data.

CREATE VIEW v\_alunos\_provas

AS

SELECT tr.codturma, pr.codprova, pr.dataavalicao

FROM turma tr INNER JOIN prova pr ON tr.codturma=pr.codturma

order by tr.codturma

-Procedure com cursor

1) Atualiza a tabela turma\_aluno com a data da matricula que está cadastrada na tabela matricula e recebe como parâmetros a data inicial e final dessa atualização.

CREATE OR REPLACE FUNCTION atualiza\_aluno(dtinicial date, dtfinal date)

RETURNS INTEGER

AS $$

DECLARE

cursordtmatri CURSOR FOR SELECT m.dt\_matricula, aluno.codaluno

FROM matricula m, aluno

WHERE m.dt\_matricula between dtinicial and dtfinal

and m.codaluno=aluno.codaluno;

datatemp date;

codaluno\_f int;

begin

open cursordtmatri;

fetch cursordtmatri into datatemp, codaluno\_f;

while found loop

update turma\_aluno set dt\_matricula = datatemp

where turma\_aluno.codaluno=codaluno\_f;

fetch cursordtmatri into datatemp, codaluno\_f;

end loop;

close cursordtmatri;

return 0;

end;

$$ language plpgsql;

Procedure com parâmetro

1. Retorna o código, nome e data de nascimento do aluno que nasceu entre das datas passadas como parâmetro.

create or replace function f\_aluno\_dtnascimento(dtinicial date, dtfinal date)

returns table (codaluno int, nome varchar(100), data date)

as $$

begin

return query select al.codaluno, al.nome, al.dt\_nascimento

from aluno al

where al.dt\_nascimento between dtinicial and dtfinal;

end;

$$ language plpgsql;

1. Retorna o código da turma e a quantidade de faltas de um determinado período.

create or replace function f\_quantidade\_faltas(dtinicial date, dtfinal date)

returns table (quantidade bigint, codturma int)

as $$

begin

return query select count(fa.codaluno), turma.codturma

from falta fa, turma

where fa.datafalta between dtinicial and dtfinal

and fa.codturma = turma.codturma

group by turma.codturma;

end;

$$ language plpgsql;

Consultas

1) Consulta que retorna os códigos das turmas e quantos alunos há em cada uma

SELECT t.codturma, t.numsala, COUNT(ta.codaluno)

FROM turma t, turma\_aluno ta

WHERE t.codturma=ta.codturma

GROUP BY t.codturma

2) Consulta que retorna os professores e qual disciplina cada um é responsável

select pro.codprofessor,pro.nome, dis.nome

from professor pro, turma\_prof\_disc tpd, disciplina dis

where tpd.coddisciplina = dis.coddisciplina

and tpd.codprofessor= pro.codprofessor

3) Retorna o código da turma com sua respectiva média de notas.

select sum(nota\_aluno.nota)/count(turma\_aluno.codaluno) as media, turma\_aluno.codturma

from nota\_aluno, turma\_aluno, prova

where nota\_aluno.codprova = prova.codprova

and prova.codturma = turma\_aluno.codturma

group by turma\_aluno.codturma

4) Retorna as turmas e seus números de faltas do período especificado.

select t.codturma, fqf.quantidade

from turma t, f\_quantidade\_faltas('01/01/2010', '31/12/2020') fqf

where fqf.codturma=t.codturma

group by t.codturma, fqf.quantidade

5) Retorna quantos alunos nasceram em algum ano e em que turma estão.

select fad.codaluno, fad.nome,fad.data, ta.codturma

from turma\_aluno ta, f\_aluno\_dtnascimento('01/01/2000', '31/12/2001') fad

where ta.codaluno=fad.codaluno

6) Retorna o aluno mais velho da escola.

select va.nome, va.codaluno, ta.codturma,va.dt\_nascimento

from v\_alunos va, turma\_aluno ta

where va.dt\_nascimento <= all(select va.dt\_nascimento from v\_alunos va)

and ta.codaluno=va.codaluno

7) Retorna quantidade de alunas em cada turma.

select turma.codturma, count(aluno.codaluno) as quantidade

from turma, turma\_aluno ta, aluno

where ta.codturma=turma.codturma

and ta.codaluno = aluno.codaluno

and aluno.sexo='F'

group by turma.codturma

8 ) Retorna o aluno com maior média da escola.

select van.codaluno, van.nome, sum(van.nota)/count(van.codaluno) as media

from v\_alunos\_notas van left join turma\_aluno ta on van.codaluno=ta.codaluno

group by van.codaluno, van.nome, ta.codaluno

having sum(van.nota)/count(van.codaluno) >= all

(select sum(van.nota)/count(van.codaluno) as media

from v\_alunos\_notas van left join turma\_aluno ta on van.codaluno=ta.codaluno

group by van.codaluno, van.nome, ta.codaluno)

9) Retorna as turmas que tiveram ao menos uma prova e quantas provas tiveram/

select turma.codturma, count(turma.codturma) as numprovas

from prova left join turma on prova.codturma=turma.codturma

group by turma.codturma, prova.codturma

having turma.codturma=prova.codturma

and count(prova.codturma) >= 1

10) Retornas as turmas que não tem nenhuma prova registrada.

select turma.codturma

from turma

where turma.codturma not in(select vap.codturma from v\_alunos\_provas vap)

Trigger

1. Valida se a nota é de uma prova que foi aplicada na turma do respetivo aluno.

create or replace function f\_valida\_prova()

returns trigger

as $$

declare

codaluno\_f int;

begin

select turma\_aluno.codaluno into codaluno\_f

from turma\_aluno

where new.codaluno=turma\_aluno.codaluno;

if (select prova.codturma from prova where prova.codprova=new.codprova) != (select ta.codturma from turma\_aluno ta where ta.codaluno=codaluno\_f)

then Raise exception 'Aluno não pertence a turma na qual a prova foi aplicada.';

end if;

return null;

end;

$$ language plpgsql;

create trigger t\_valida\_prova

before insert or update on nota\_aluno

for each row

execute procedure f\_valida\_prova()

1. Valida se a falta está sendo registrada na turma certa para o respectivo aluno

create or replace function f\_valida\_falta()

returns trigger

as $$

declare

codaluno\_f int;

begin

select new.codaluno into codaluno\_f;

if(select distinct ta.codturma from turma\_aluno ta where ta.codturma=new.codturma) != (select distinct ta.codturma from turma\_aluno ta where ta.codaluno=codaluno\_f)

then raise exception 'Aluno não está nessa turma';

end if;

return null;

end;

$$ language plpgsql;

create trigger t\_valida\_falta

after insert or update on falta

for each row

execute procedure f\_valida\_falta()

1. Impede que o nome de um aluno seja deixado em branco ou deletado

create or replace function f\_impede\_delete()

returns trigger

as $$

begin

if TG\_OP='DELETE' then

Raise exception 'Não é possível deletar o nome do aluno';

end if;

if new.nome='' or new.nome=null then

Raise exception 'Não é possível deixar o nome do aluno em branco';

end if;

return null;

end;

$$ language plpgsql;

create trigger t\_impede\_delete

after delete or update on aluno

for each row

execute procedure f\_impede\_delete()