ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Н.В Путилова |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

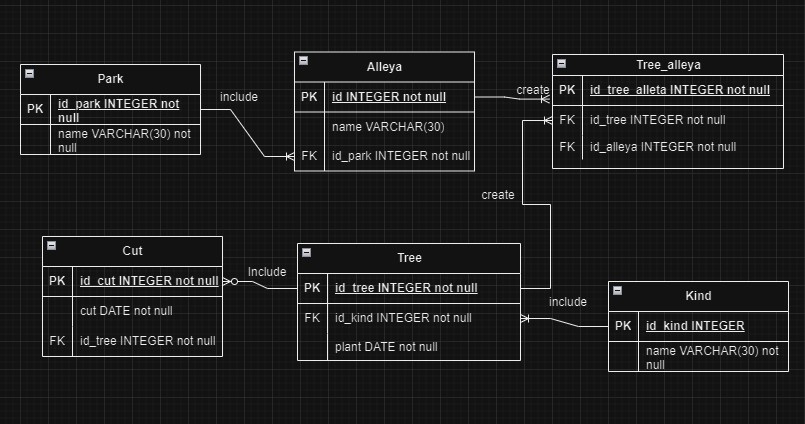
|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №7  Разработка SQL запросов: запросы с подзапросами |
| **по дисциплине: Проектирование баз данных** |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4134к |  |  |  | Костяков Н.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург

2023



**Процедура вставки с пополнением**

create OR REPLACE procedure ins\_tree(ins\_kind varchar(128), ins\_plant date, ins\_alleya varchar(128), ins\_park varchar(128))

LANGUAGE plpgsql

AS $$

BEGIN declare id\_all\_new int;

declare id\_park\_new int;

begin

IF exists (select \* from park where park.name=ins\_park)

THEN select park.id\_park into id\_park\_new from park where park.name = ins\_park;

else begin

select id\_park\_new = (nullif(max(park.id\_park), 0)+1) from park ;

insert into park(id\_park, name\_park) values (id\_park\_new, ins\_park);

end;

end if;

begin

IF exists(select \* from alleya where alleya.name = ins\_alleya)

THEN select alleya.id\_alleya into id\_all\_new from alleya where alleya.name = ins\_alleya;

else begin

select id\_all\_new = (nullif(max(alleya.id\_alleya), 0)+1) from alleya ;

insert into alleya(id\_alleya, name\_alleya, id\_park) values (id\_alleya\_new, ins\_alleya, is\_park\_new);

END;

end if;

declare id\_kind\_new int;

begin

if exists(select \* from kind where kind.name = ins\_kind)

then select kind.id\_kind into id\_kind\_new from kind where kind.name = ins\_kind;

else begin

select id\_kind\_new = (nullif(max(kind.id\_kind), 0)+1) from kind ;

insert into kind(id\_kind, name) values (id\_kind\_new, ins\_kind);

end;

end if;

declare id\_t\_a int = (nullif(max(tree\_alleya.id\_tree\_alleya), 0)+1) from tree\_alleya;

declare id\_tree\_new int = (nullif(max(tree.id\_tree), 0)+1) from tree;

begin

insert into tree values (id\_tree\_new, id\_kind\_new, ins\_plant);

insert into tree\_alleya(id\_tree\_alleya ,id\_tree, id\_alleya) values (id\_t\_a,id\_tree\_new, id\_all\_new);

end;

end;

end;

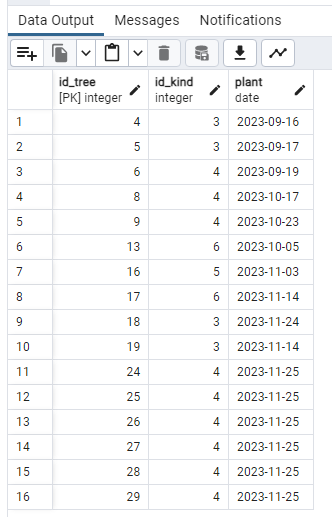
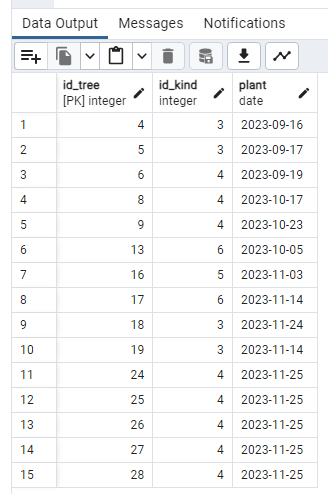
end;

end

$$;

call ins\_tree ( 'Дуб'::varchar(128), '2023-11-25'::date, 'Быстрая'::varchar(128), 'Космический'::varchar(128));

Слева – было. справа - стало



--удаление с очисткой справочника

create or replace procedure del\_tree(id\_t int)

language plpgsql

as $$

begin

begin

delete from tree\_alleya where id\_t = tree\_alleya.id\_tree;

end;

declare id\_all\_new int = (select id\_alleya from tree\_alleya where id\_tree = id\_t);

begin

if not exists (select \* from tree\_alleya where id\_alleya=id\_all\_new)

then delete from alleya where id\_alleya = id\_all\_new;

end if;

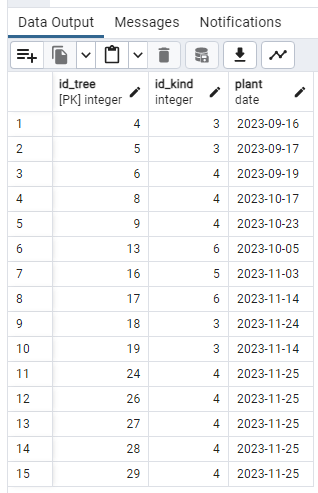
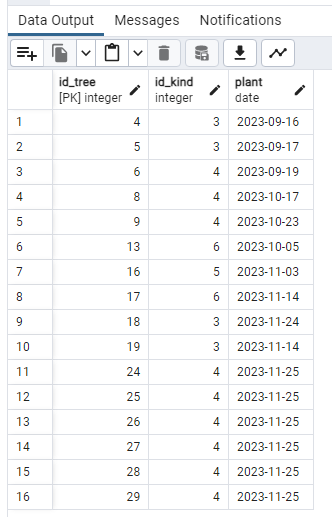
delete from tree where id\_tree = id\_t;

end;

end

$$;

-- call del\_tree(10);



--Каскадное удаление

create or replace procedure del\_kind\_cascade(id\_to\_del int)

language plpgsql

as $$

begin

delete from tree\_alleya where tree\_alleya.id\_tree in

(select tree.id\_tree from tree where id\_kind = id\_to\_del);

delete from cut where cut.id\_tree in

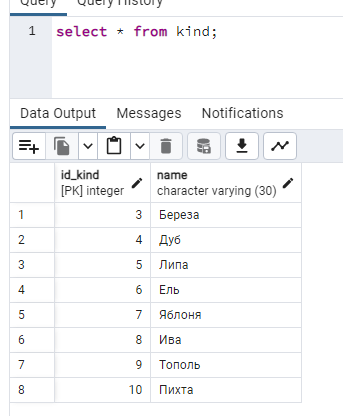
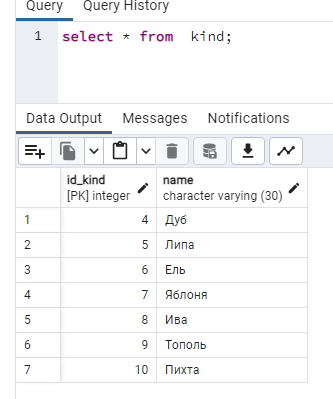
(select tree.id\_tree from tree where tree.id\_kind=id\_to\_del);

delete from tree where tree.id\_kind = id\_to\_del;

delete from kind where kind.id\_kind = id\_to\_del;

end;

$$

--Процедура вычисления и возврат значения агрегатной функции

create or replace function count\_trees() returns integer

language plpgsql

as $$

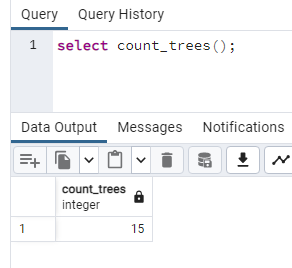
begin

return nullif(count(id\_tree), 0) from tree;

end

$$;

select count\_trees();



--Формирование статистики во временной таблице

create or replace function stats()

returns table (id\_park integer, name\_park varchar(128),

alleya\_cnt integer, tree\_cnt integer,

kind\_cnt integer)

language plpgsql

as $$

begin

CREATE TEMPORARY TABLE stat(

id\_park integer, name\_park varchar(128),

alleya\_cnt integer, tree\_cnt integer,

kind\_cnt integer

);

insert into stat(id\_park, name\_park, alleya\_cnt, tree\_cnt, kind\_cnt)

select park.id\_park,

park.name,

count(DISTINCT alleya.name),

count( tree\_alleya.id\_tree ),

count(DISTINCT kind.name)

FROM park

join alleya on alleya.id\_park = park.id\_park

join tree\_alleya on tree\_alleya.id\_alleya = alleya.id\_alleya

join tree on tree.id\_tree = tree\_alleya.id\_tree

join kind on kind.id\_kind = tree.id\_kind

GROUP BY park.id\_park;

return query select stat.id\_park, stat.name\_park

,stat.alleya\_cnt , stat.tree\_cnt, stat.kind\_cnt FROM stat;

drop table stat;

end

$$;

select \* from stats();

