ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

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

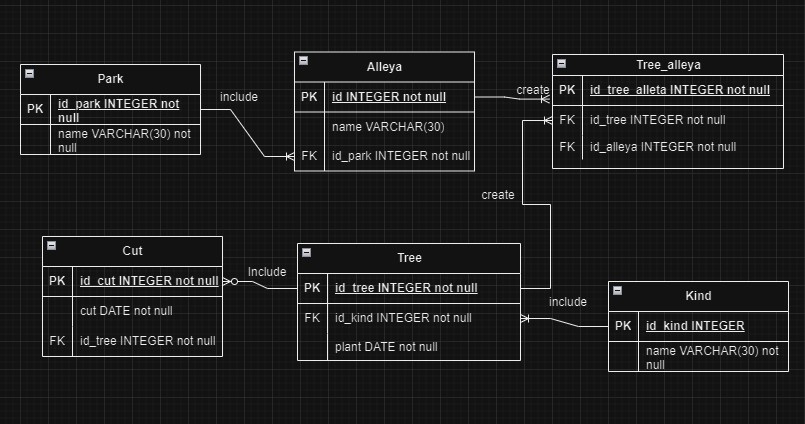
|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №7  Хранимые процедуры |
| **по дисциплине: Проектирование баз данных** |
|  |
|  |

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 $$

DECLARE

    id\_all\_new INT;

    id\_park\_new INT;

    id\_kind\_new INT;

    id\_t\_a INT;

    id\_tree\_new INT;

BEGIN

    -- Блок с парками

    BEGIN

        IF EXISTS (SELECT 1 FROM park WHERE park.name = ins\_park) THEN

            SELECT park.id\_park INTO id\_park\_new FROM park WHERE park.name = ins\_park;

        ELSE

            id\_park\_new := (NULLIF((SELECT MAX(id\_park) FROM park), 0) + 1);

            INSERT INTO park(id\_park, name) VALUES (id\_park\_new, ins\_park);

        END IF;

    END;

    -- Блок с аллеями

    BEGIN

        IF EXISTS (SELECT 1 FROM alleya WHERE alleya.name = ins\_alleya) THEN

            SELECT alleya.id\_alleya INTO id\_all\_new FROM alleya WHERE alleya.name = ins\_alleya;

        ELSE

            id\_all\_new := (NULLIF((SELECT MAX(id\_alleya) FROM alleya), 0) + 1);

            INSERT INTO alleya(id\_alleya, name, id\_park) VALUES (id\_all\_new, ins\_alleya, id\_park\_new);

        END IF;

    END;

    -- Блок с видами деревьев

    BEGIN

        IF EXISTS (SELECT 1 FROM kind WHERE kind.name = ins\_kind) THEN

            SELECT kind.id\_kind INTO id\_kind\_new FROM kind WHERE kind.name = ins\_kind;

        ELSE

            id\_kind\_new := (NULLIF((SELECT MAX(id\_kind) FROM kind), 0) + 1);

            INSERT INTO kind(id\_kind, name) VALUES (id\_kind\_new, ins\_kind);

        END IF;

    END;

    -- Блок с деревьями и аллеями

    id\_t\_a := (SELECT COALESCE(MAX(id\_tree\_alleya), 0) + 1 FROM tree\_alleya);

    id\_tree\_new := (SELECT COALESCE(MAX(id\_tree), 0) + 1 FROM tree);

    INSERT INTO tree(id\_tree, id\_kind, plant) VALUES (id\_tree\_new, id\_kind\_new, ins\_plant);

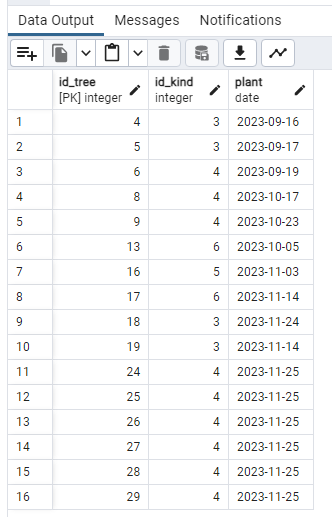
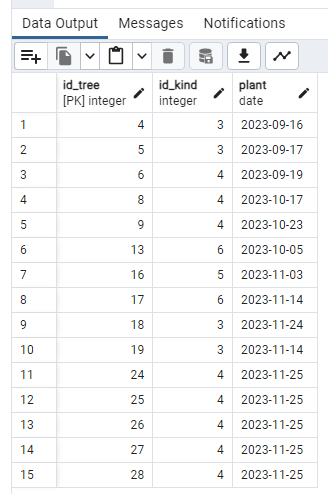
    INSERT INTO tree\_alleya(id\_tree\_alleya, id\_tree, id\_alleya) VALUES (id\_t\_a, id\_tree\_new, id\_all\_new);

END;

$$;

CALL ins\_tree('Сакура', '2023-12-11', 'Вылет', 'Аэро2');

Слева – было. справа - стало



--удаление с очисткой справочника

CREATE OR REPLACE PROCEDURE del\_tree(id\_t INT)

LANGUAGE plpgsql

AS $$

DECLARE

    id\_all\_new INT;

BEGIN

    id\_all\_new := (SELECT id\_alleya FROM tree\_alleya WHERE id\_tree = id\_t);

    BEGIN

        DELETE FROM tree\_alleya WHERE id\_tree = id\_t;

    END;

    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;

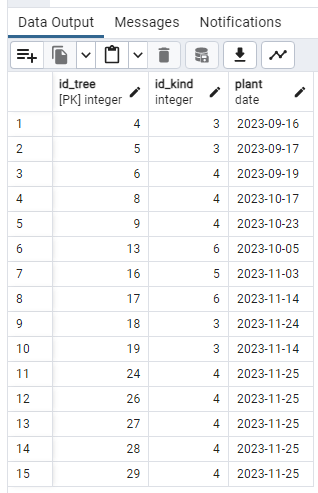
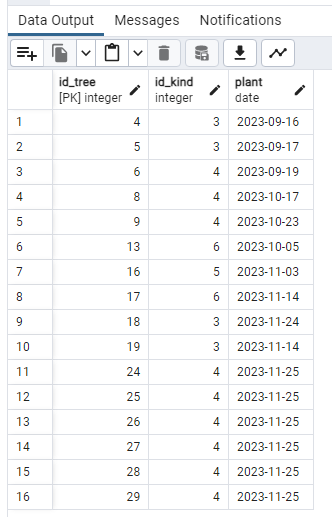
    END;

    DELETE FROM tree WHERE id\_tree = id\_t;

END;

$$;

call del\_tree(16);



--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

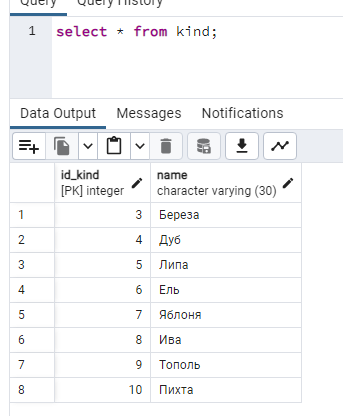
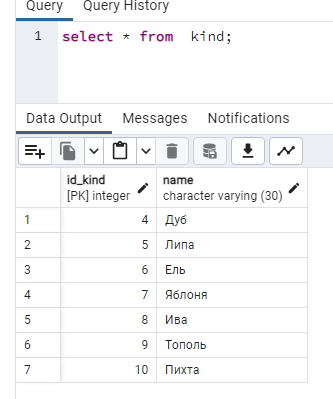
        (SELECT tree.id\_tree FROM tree WHERE tree.id\_kind = id\_to\_del);

    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 $$

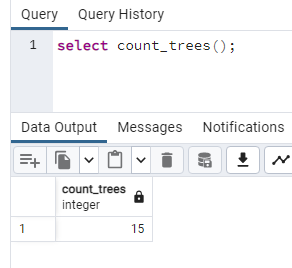
BEGIN

    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();

