ГУАП

КАФЕДРА № 43

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

старший преподаватель		М. В. Величко
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ

Лабораторная работа №10
Объектно-реляционные базы данных. Манипуляция данными и пользовательские операторы

по курсу: «Проектирование баз данных»

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. №	Z1431		М.Д..Быстров
		подпись, дата	инициалы, фамилия

Санкт-Петербург 2024

ЗАДАНИЕ

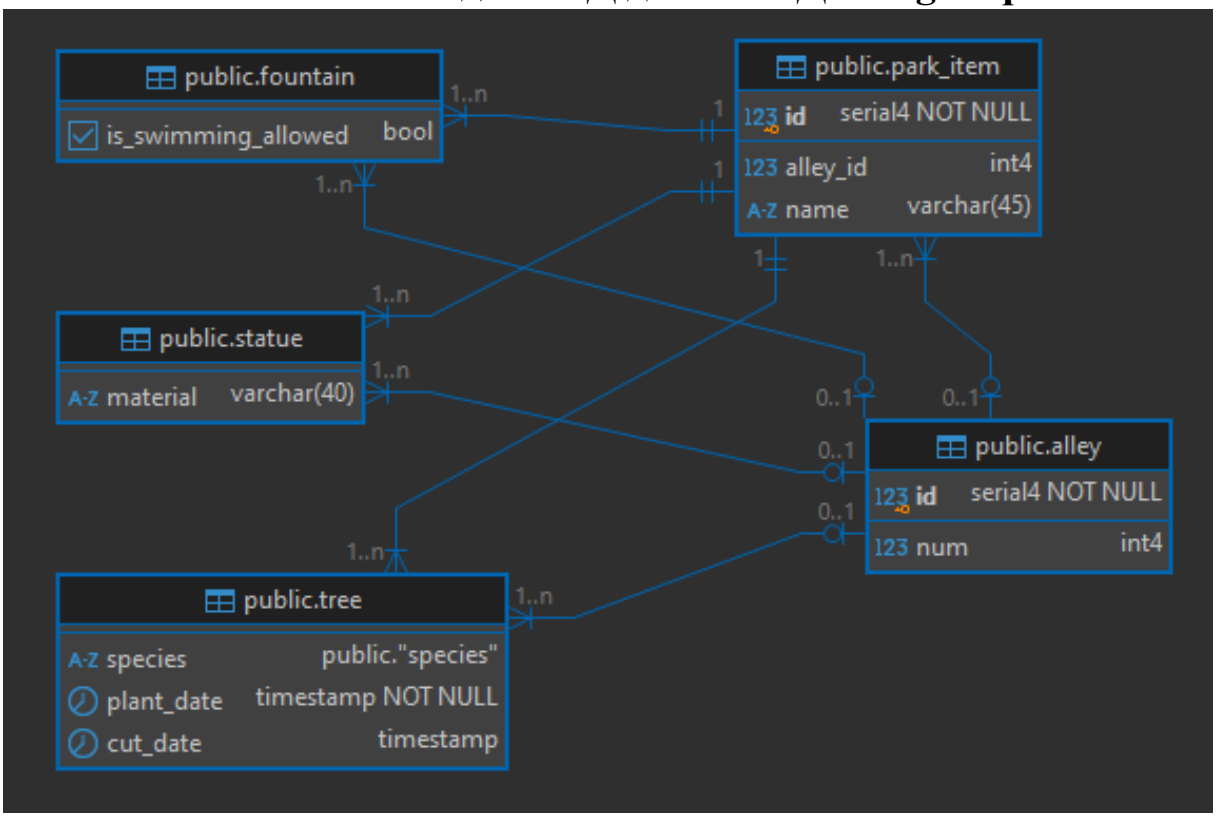
1. Выполнить вставку тестовых данных в таблицы, созданные в ходе выполнения лабораторной работы 9.
2. Сделать запросы выборки с условием к таблицам предку и потомку, только потомку и только предку.
3. Придумать и создать пользовательский оператор для своей предметной области
4. Придумать и создать пользовательскую агрегатную функцию для своей предметной области

Вариант 4:

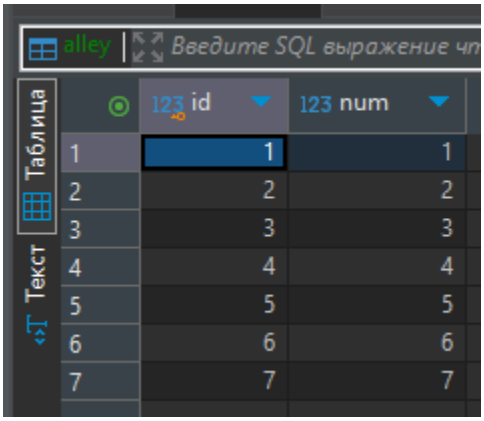
парк: статуи, фонтаны, деревья ,породы, дата высадки, дата обрезки, расположение, аллеи

а. аллеи, на которых встречаются разные виды кленов (клен в названии)
б. аллеи, на которых есть и статуи и фонтаны
в. дерево, которое было посажено позже всех
г. порода, деревьев которой больше всего
д. аллея, на которой нет фонтанов

Физическая модель БД для СУБД Postgresql



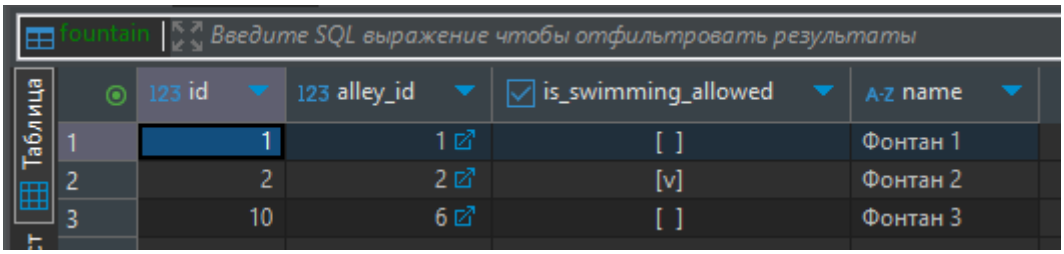
Наборы данных



alley | Введите SQL выражение чт

Таблица | Текст

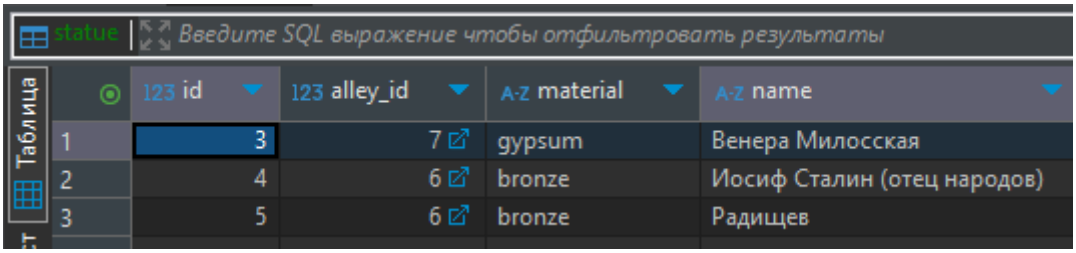
	id	num
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6
7	7	7



fountain | Введите SQL выражение чтобы отфильтровать результаты

Таблица

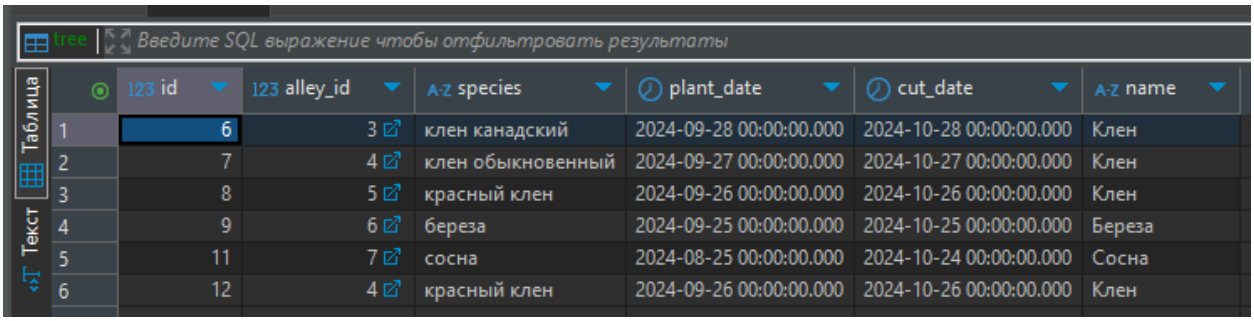
	id	alley_id	is_swimming_allowed	name
1	1	1	[ ]	Фонтан 1
2	2	2	[v]	Фонтан 2
3	10	6	[ ]	Фонтан 3



statue | Введите SQL выражение чтобы отфильтровать результаты

Таблица

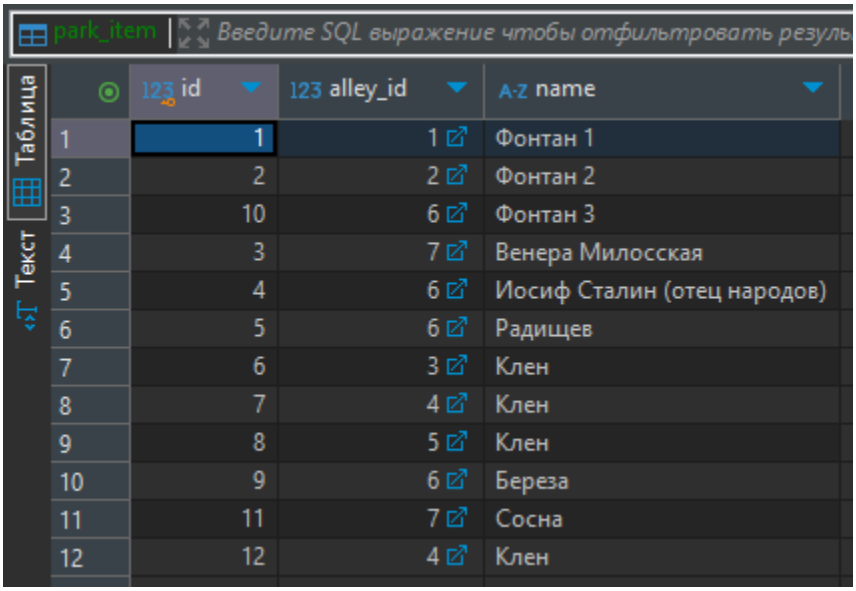
	id	alley_id	material	name
1	3	7	gypsum	Венера Милосская
2	4	6	bronze	Иосиф Сталин (отец народов)
3	5	6	bronze	Радищев



tree | Введите SQL выражение чтобы отфильтровать результаты

Таблица | Текст

	id	alley_id	species	plant_date	cut_date	name
1	6	3	клен канадский	2024-09-28 00:00:00.000	2024-10-28 00:00:00.000	Клен
2	7	4	клен обыкновенный	2024-09-27 00:00:00.000	2024-10-27 00:00:00.000	Клен
3	8	5	красный клен	2024-09-26 00:00:00.000	2024-10-26 00:00:00.000	Клен
4	9	6	береза	2024-09-25 00:00:00.000	2024-10-25 00:00:00.000	Береза
5	11	7	сосна	2024-08-25 00:00:00.000	2024-10-24 00:00:00.000	Сосна
6	12	4	красный клен	2024-09-26 00:00:00.000	2024-10-26 00:00:00.000	Клен



	id	alley_id	name
1	1	1	Фонтан 1
2	2	2	Фонтан 2
3	10	6	Фонтан 3
4	3	7	Венера Милосская
5	4	6	Иосиф Сталин (отец народов)
6	5	6	Радищев
7	6	3	Клен
8	7	4	Клен
9	8	5	Клен
10	9	6	Береза
11	11	7	Сосна
12	12	4	Клен

Текст запросов

```
--а.  аллеи, на которых встречаются разные виды кленов (клен в названии)
-- к наследнику
select a.*,
count(t.id) as tree_count, -- кол-во деревьев
unique_by_similar(t."species") as species_count -- кол-во пород
from alley a join tree t on t.alley_id = a.id
where lower(t."name") like '%клен%' -- в имени содержится клен
and exists(
      select t2.id from tree t2
      where t2.alley_id = a.id
      and t2."species" ~ t."species"
      and t2.id != t.id) -- на аллее есть другие клены (подобные деревья)
group by a.id
having unique_by_similar(t."species") = 1; -- в выборку одной группы попали строго
подобные породы



--б.  аллеи, на которых есть и статуи и фонтаны
-- к главной таблице и наследнику
SELECT a.* FROM alley a
where (
-- счет уникальных номеров таблиц для элементов на аллее
      SELECT count(distinct i.tableoid) FROM park_item i
      left JOIN fountain f ON f.id = i.id -- фонтаны
      left join statue s on s.id = i.id -- статуи
      where i.alley_id = a.id -- на аллее
```

```
      and (f.id is not null or s.id is not null) -- нашлась либо статуя либо фонтан
) >= 2; -- как минимум два типа - нашлось и то и другое

--в.  дерево, которое было посажено позже всех
-- к наследнику
select t.* from tree t
where t.plant_date =
      (select max(t2.plant_date) from tree t2);

--г.  порода, деревьев которой больше всего
-- к наследнику
select t."species", count(t.id) as cnt from tree t
group by t."species"
order by cnt desc limit 1;


--д.  аллея, на которой нет фонтанов
-- к главной таблице
      select a.* from alley a -- все аллеи
except
      --исключая аллеи на которых есть фонтаны
      select distinct a2.* from alley a2
      join park_item pi2
      on pi2.alley_id = a2.id
      and pi2.tableoid = 'fountain'::regclass::oid
order by id;
```

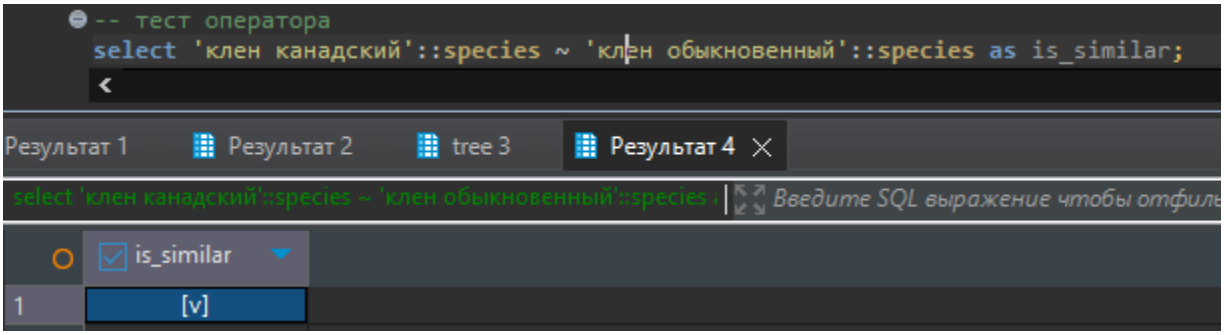
Код операторов

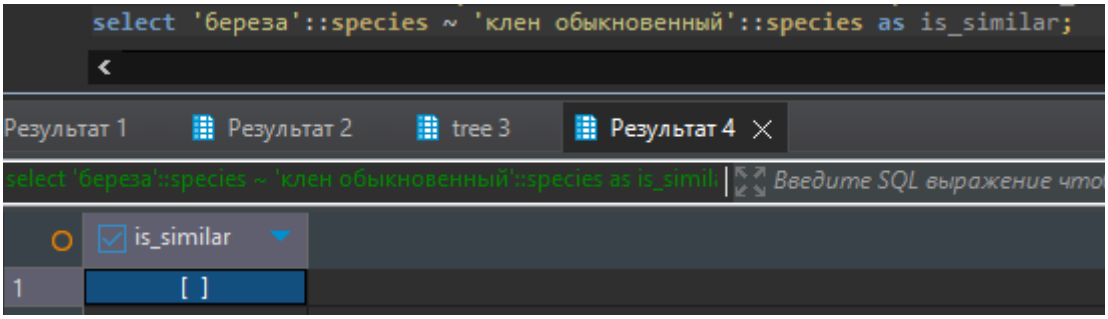
```
CREATE OR REPLACE FUNCTION public.specia_is_similar(first species, second species)
 RETURNS boolean
 LANGUAGE plpgsql
AS $function$
      BEGIN

            if first = second then
                  return true;
            end if;

            if (first in ('клен канадский', 'клен обыкновенный', 'красный клен')
            and second in ('клен канадский', 'клен обыкновенный', 'красный клен'))
            or (first in ('береза', 'гималайская береза')
            and second in ('береза', 'гималайская береза'))  then

                  return true;
```

```
            end if;

            return false;
      END;
$function$
;


-- создание оператора
create OPERATOR ~ (
      leftarg = species,
      rightarg = species,
      procedure = specia_is_similar
);


CREATE OR REPLACE FUNCTION public.species_acc(arr species[], next species)
 RETURNS species[]
 LANGUAGE plpgsql
AS $function$
      declare
            ret species[];
      BEGIN
            ret = array_append(arr, next);

            return ret;
      END;
$function$
;


CREATE OR REPLACE FUNCTION public.count_unique_similar("values" species[])
 RETURNS integer
 LANGUAGE plpgsql
AS $function$
      declare
            distinct_values species[];
            current_specia species;
            inner_current_specia species;

            specs RECORD;
            spec RECORD;

            similar_count int4;
            group_count int4;
            i int4;
      BEGIN

            group_count = 0;

            -- уникальные значения в массиве
            for spec in select distinct * from unnest("values") as val loop

                  distinct_values = array_append(
                        distinct_values,
```

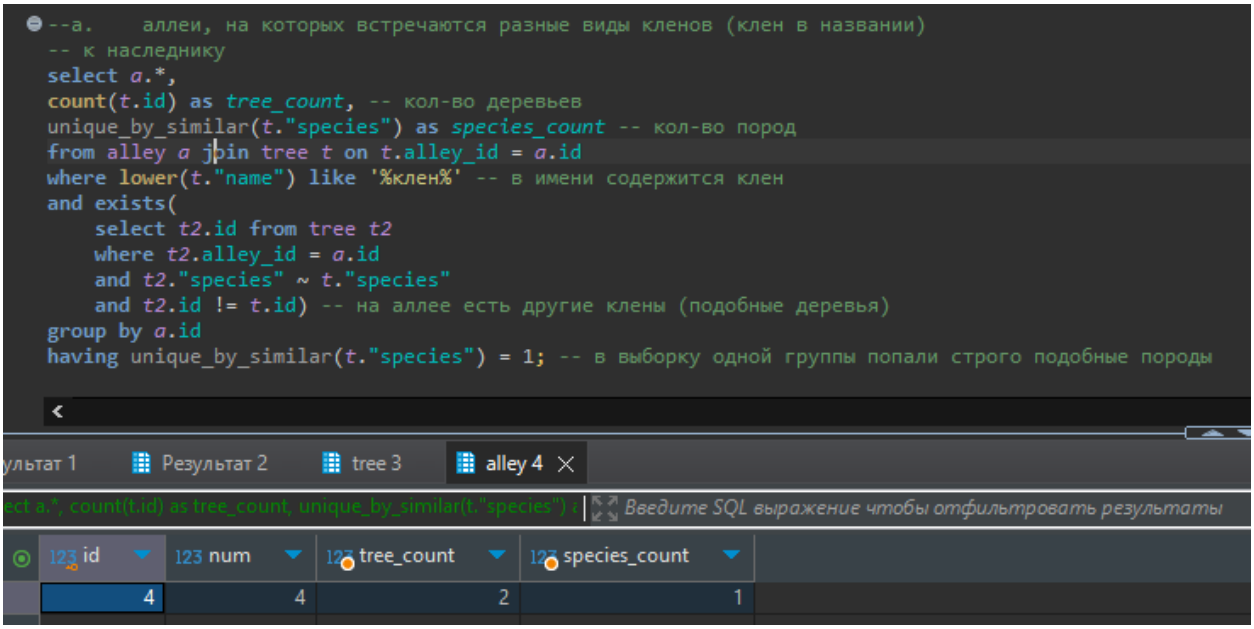
```
                        spec.val::species);

            end loop;

            group_count = 0;

            -- удаляем поэлементно с расчетом кол-ва подобных
            while array_length(distinct_values, 1) > 0 loop

                  current_specia = distinct_values[1];

                  group_count = group_count + 1;

                  i = 1;

                  while i <= array_length(distinct_values, 1) loop

                        inner_current_specia = distinct_values[i];



                        -- если вид подобен текущему - удаляем
                        if  inner_current_specia ~ current_specia then

                              distinct_values = array_remove(distinct_values,
inner_current_specia);

                        else

                              i = i + 1;

                        end if;

                  end loop;

            end loop;



            return group_count;



      END;
$function$
;

-- создание агрегатной функции
CREATE AGGREGATE unique_by_similar(species) (
    SFUNC = species_acc, -- функция, собирающая массив
    STYPE = species[], -- тип данных состояния
    FINALFUNC = count_unique_similar, -- финализируюшая функция
    INITCOND = "{}"); -- начальный пустой массив
```

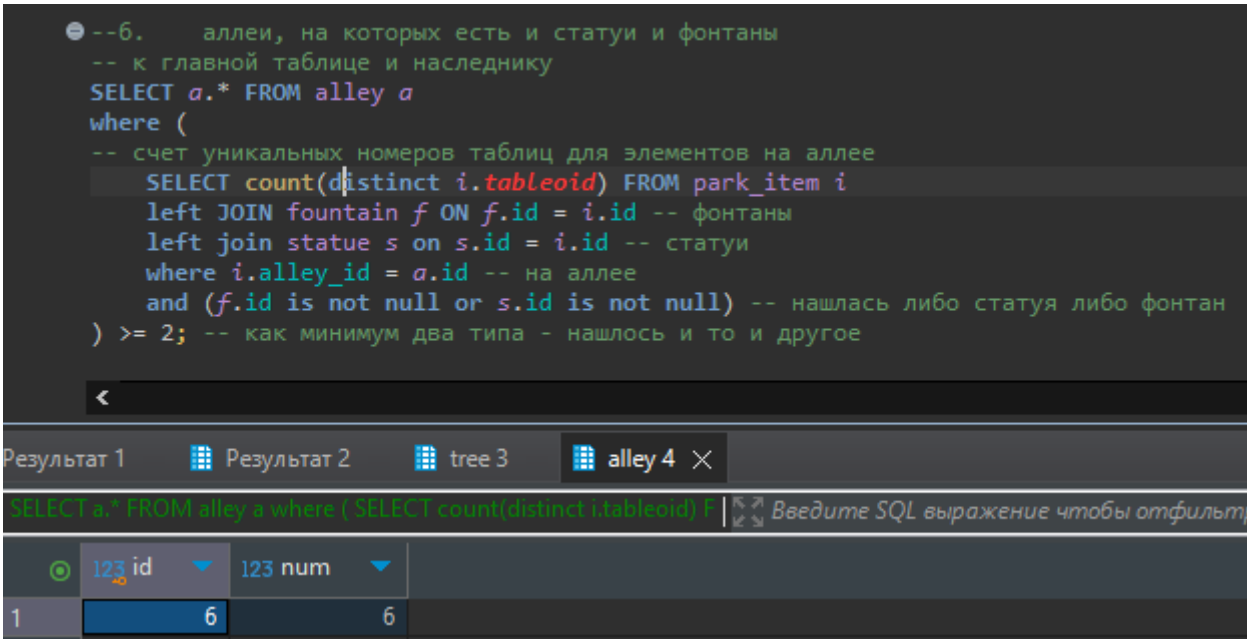
Пример выполнения операторов





Наборы данных, возвращаемых запросами





```sql
--в.    дерево, которое было посажено позже всех
-- к наследнику
select t.* from tree t
where t.plant_date =
    (select max(t2.plant_date) from tree t2);
```

ультат 1 | Результат 2 | tree 3 | tree 4

ct t.* from tree t where t.plant_date = (select max(t2.plant_date) | Введите SQL выражение чтобы отфильтровать резу

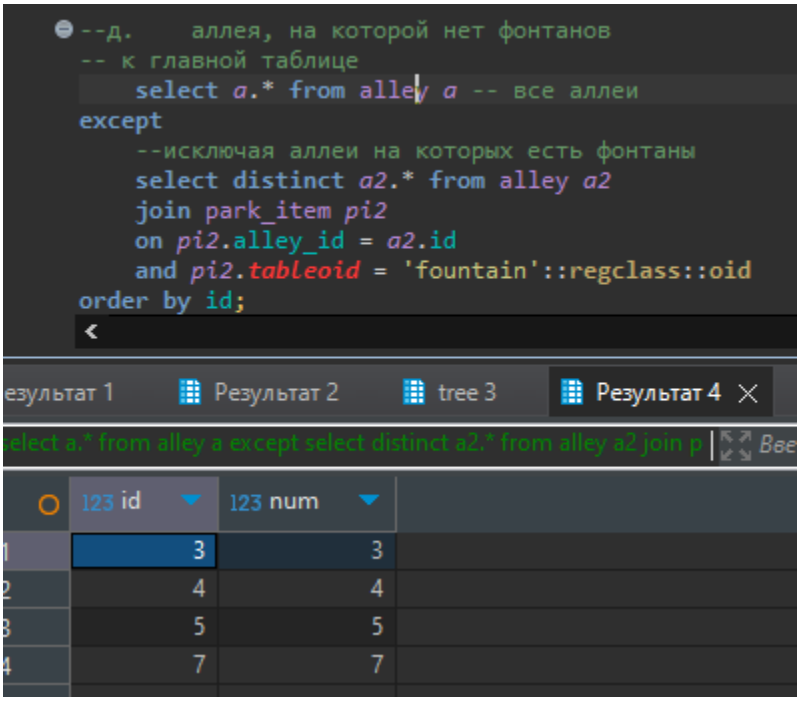
123 id	123 alley_id	A-Z species	plant_date	cut_date	A-Z name
6	3	клен канадский	2024-09-28 00:00:00.000	2024-10-28 00:00:00.000	Клен

```sql
--г.    порода, деревьев которой больше всего
-- к наследнику
select t."species", count(t.id) as cnt from tree t
group by t."species"
order by cnt desc limit 1;
```

ультат 1 | Результат 2 | tree 3 | tree 4

ct t."species", count(t.id) as cnt from tree t group by t."species" | Введ

A-Z species	123 cnt
красный клен	2


--д.   аллея, на которой нет фонтанов
-- к главной таблице
select a.* from alley a -- все аллеи
except
--исключая аллеи на которых есть фонтаны
select distinct a2.* from alley a2
join park_item pi2
on pi2.alley_id = a2.id
and pi2.tableoid = 'fountain'::regclass::oid
order by id;
езультат 1
Результат 2
tree 3
Результат 4
elect a.* from alley a except select distinct a2.* from alley a2 join p
123 id
123 num
3 3
4 4
5 5
7 7

Вывод

В ходе выполнения 10 лабораторной работы были созданы:

1. Пользовательский оператор
2. Агрегатная функция

В СУБД PostgreSQL.

Написаны запросы с использованием созданных оператора и функции. Среди запросов есть запросы к наследнику, к наследнику и родителю, и только к родителю в иерархии наследования реляционных таблиц.