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 №9   
Объектно-реляционные базы данных. Проектирование и создание

по курсу: «Проектирование баз данных»

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. № Z1431 М.Д..Быстров

подпись, дата инициалы, фамилия

Санкт-Петербург 2024

# ЗАДАНИЕ

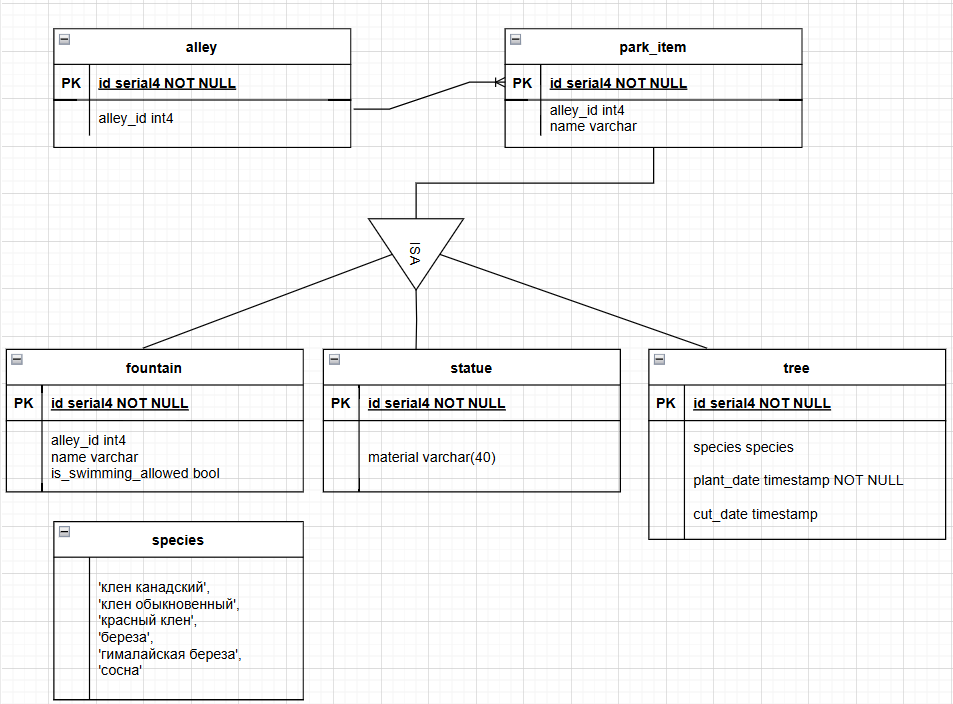
Спроектировать физическую модель базы данных, находящуюся в третьей нормальной форме и включающей наследование и хотя бы один пользовательский тип в соответствии с заданным вариантом. Написать соответствующий скрипт создания базы данных

**Вариант** 4:

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

1. аллеи, на которых встречаются разные виды кленов (клен в названии)
2. аллеи, на которых есть и статуи и фонтаны
3. дерево, которое было посажено позже всех
4. порода, деревьев которой больше всего
5. аллея, на которой нет фонтанов

**Физическая модель БД для СУБД Postgresql**



**Скрипт создания базы данных**

**CREATE** **SCHEMA** public;

**ALTER** **SCHEMA** public **OWNER** **TO** pg\_database\_owner;

**COMMENT** **ON** **SCHEMA** public **IS** 'standard public schema';

**CREATE** **TYPE** public.species **AS** **ENUM** (

'клен канадский',

'клен обыкновенный',

'красный клен',

'береза',

'гималайская береза',

'сосна'

);

**ALTER** **TYPE** public.species **OWNER** **TO** postgres;

**SET** default\_tablespace = '';

**SET** default\_table\_access\_method = heap;

**CREATE** **TABLE** public.alley (

id **integer** **NOT** **NULL**,

num **integer**

);

**ALTER** **TABLE** public.alley **OWNER** **TO** postgres;

**CREATE** **SEQUENCE** public.alley\_id\_seq

**AS** **integer**

**START** **WITH** 1

**INCREMENT** **BY** 1

**NO** **MINVALUE**

**NO** **MAXVALUE**

**CACHE** 1;

**ALTER** **SEQUENCE** public.alley\_id\_seq **OWNER** **TO** postgres;

**ALTER** **SEQUENCE** public.alley\_id\_seq **OWNED** **BY** public.alley.id;

**CREATE** **TABLE** public.park\_item (

id **integer** **NOT** **NULL**,

alley\_id **integer**

);

**ALTER** **TABLE** public.park\_item **OWNER** **TO** postgres;

**COMMENT** **ON** **TABLE** public.park\_item **IS** 'Главная таблица элементов парка';

**COMMENT** **ON** **COLUMN** public.park\_item.alley\_id **IS** 'id аллеи';

**CREATE** **TABLE** public.fountain (

is\_swimming\_allowed **boolean**

)

**INHERITS** (public.park\_item);

**ALTER** **TABLE** public.fountain **OWNER** **TO** postgres;

**CREATE** **SEQUENCE** public.park\_item\_id\_seq

**AS** **integer**

**START** **WITH** 1

**INCREMENT** **BY** 1

**NO** **MINVALUE**

**NO** **MAXVALUE**

**CACHE** 1;

**ALTER** **SEQUENCE** public.park\_item\_id\_seq **OWNER** **TO** postgres;

**ALTER** **SEQUENCE** public.park\_item\_id\_seq **OWNED** **BY** public.park\_item.id;

**CREATE** **TABLE** public.statue (

material **character** **varying**(40)

)

**INHERITS** (public.park\_item);

**ALTER** **TABLE** public.statue **OWNER** **TO** postgres;

**CREATE** **TABLE** public.tree (

species public.species,

plant\_date **timestamp** ***without*** **time** **zone** **NOT** **NULL**,

cut\_date **timestamp** ***without*** **time** **zone**

)

**INHERITS** (public.park\_item);

**ALTER** **TABLE** public.tree **OWNER** **TO** postgres;

**ALTER** **TABLE** **ONLY** public.alley **ALTER** **COLUMN** id **SET** **DEFAULT** **nextval**('public.alley\_id\_seq'::**regclass**);

**ALTER** **TABLE** **ONLY** public.fountain **ALTER** **COLUMN** id **SET** **DEFAULT** **nextval**('public.park\_item\_id\_seq'::**regclass**);

**ALTER** **TABLE** ***ONLY*** public.park\_item **ALTER** **COLUMN** id **SET** **DEFAULT** **nextval**('public.park\_item\_id\_seq'::**regclass**);

**ALTER** **TABLE** ***ONLY*** public.statue **ALTER** **COLUMN** id **SET** **DEFAULT** **nextval**('public.park\_item\_id\_seq'::**regclass**);

**ALTER** **TABLE** ***ONLY*** public.tree **ALTER** **COLUMN** id **SET** **DEFAULT** **nextval**('public.park\_item\_id\_seq'::**regclass**);

**SELECT** pg\_catalog.**setval**('public.alley\_id\_seq', 1, **false**);

**SELECT** pg\_catalog.**setval**('public.park\_item\_id\_seq', 1, **false**);

**ALTER** **TABLE** **ONLY** public.alley

**ADD** **CONSTRAINT** alley\_pk **PRIMARY** **KEY** (id);

**ALTER** **TABLE** **ONLY** public.park\_item

**ADD** **CONSTRAINT** park\_item\_pk **PRIMARY** **KEY** (id);

**ALTER** **TABLE** ONLY public.fountain

**ADD** **CONSTRAINT** fountain\_unique **UNIQUE** (id);

**ALTER** **TABLE** ONLY public.statue

**ADD** **CONSTRAINT** statue\_unique **UNIQUE** (id);

**ALTER** **TABLE** ONLY public.tree

**ADD** **CONSTRAINT** tree\_unique **UNIQUE** (id);

**ALTER** **TABLE** ***ONLY*** public.fountain

**ADD** **CONSTRAINT** fountain\_alley\_fk **FOREIGN** **KEY** (alley\_id) **REFERENCES** public.alley(id) **ON** **UPDATE** **RESTRICT** **ON** **DELETE** **RESTRICT**;

**ALTER** **TABLE** ***ONLY*** public.park\_item

**ADD** **CONSTRAINT** park\_item\_alley\_fk **FOREIGN** **KEY** (alley\_id) **REFERENCES** public.alley(id) **ON** **UPDATE** **RESTRICT** **ON** **DELETE** **RESTRICT**;

**ALTER** **TABLE** ***ONLY*** public.statue

**ADD** **CONSTRAINT** statue\_alley\_fk **FOREIGN** **KEY** (alley\_id) **REFERENCES** public.alley(id) **ON** **UPDATE** **RESTRICT** **ON** **DELETE** **RESTRICT**;

**ALTER** **TABLE** ***ONLY*** public.tree

**ADD** **CONSTRAINT** tree\_alley\_fk **FOREIGN** **KEY** (alley\_id) **REFERENCES** public.alley(id) **ON** **UPDATE** **RESTRICT** **ON** **DELETE** **RESTRICT**;

**CREATE** **OR** **REPLACE** **FUNCTION** public.park\_item\_check\_id()

RETURNS **trigger**

LANGUAGE plpgsql

**AS** **$function$**

**DECLARE**

is\_found **boolean**;

**BEGIN**

**select** **exists**(**select** \* **from** park\_item i

**where** i.id = **NEW**.id)

**into** is\_found;

**if** (is\_found = **true**) **then**

**RAISE** **EXCEPTION** 'duplicate ids';

**end** **if**;

**return** **new**;

**END**;

**$function$**

;

**CREATE** **TRIGGER** check\_id **before** **insert** **or** **update** **ON** public.park\_item **FOR** **EACH** **ROW** **EXECUTE** **FUNCTION** public.park\_item\_check\_id();

**CREATE** **TRIGGER** check\_id **before** **insert** **or** **update** **ON** public.fountain **FOR** **EACH** **ROW** **EXECUTE** **FUNCTION** public.park\_item\_check\_id();

**CREATE** **TRIGGER** check\_id **before** **insert** **or** **update** **ON** public.tree **FOR** **EACH** **ROW** **EXECUTE** **FUNCTION** public.park\_item\_check\_id();

**CREATE** **TRIGGER** check\_id **before** **insert** **or** **update** **ON** public.statue **FOR** **EACH** **ROW** **EXECUTE** **FUNCTION** public.park\_item\_check\_id();

**REVOKE** **USAGE** **ON** **SCHEMA** public **FROM** PUBLIC;

**Вывод**

В ходе выполнения 9 лабораторной работы была спроектирована и создана реляционная база данных в СУБД Postgresql. В базе данных присутствует созданный пользовательский тип (перечисление) и использовано наследование при создании таблиц.

В отчете представлен скрипт по созданию базы данных. Версия СУБД Postgresql – 17.