Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»



**Домашнее задание №2 по дисциплине**

«Постреляционные базы данных»

**ИСПОЛНИТЕЛЬ:**

Березин И. С.

Группа ИУ5-23М

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

"\_\_"\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

Москва, 2019

**Задание для PostgeSQL:**

В среде СУБД PostgreSQL на основе предметной области, используя атрибуты и запросы (по варианту темы):

1. **Определить систему таблиц (поля, ключи, ограничения на уникальность, на значения, на связь таблиц, на NULL и значения по умолчанию) – 2 балла.**

*Таблица Clients (Клиенты):*

-- Table: dz.clients

-- DROP TABLE dz.clients;

CREATE TABLE dz.clients

(

id\_client serial NOT NULL,

fio character varying(255),

passport bigint NOT NULL,

age integer,

city character varying(255) DEFAULT 'Moscow'::character varying,

client\_address addresss,

contacts json,

CONSTRAINT clients\_pkey PRIMARY KEY (id\_client),

CONSTRAINT clients\_passport\_city\_client\_address\_key UNIQUE (passport, city, client\_address),

CONSTRAINT clients\_passport\_key UNIQUE (passport),

CONSTRAINT clients\_age\_check CHECK (age >= 18)

)

WITH (

OIDS=FALSE

);

ALTER TABLE dz.clients

OWNER TO postgres;

*Таблица Orders (Заказы):*

-- Table: dz.orders

-- DROP TABLE dz.orders;

CREATE TABLE dz.orders

(

id\_order serial NOT NULL,

client\_id integer,

summary real NOT NULL,

orderdate date,

clientcomment text DEFAULT 'good'::text,

CONSTRAINT orders\_pkey PRIMARY KEY (id\_order),

CONSTRAINT orders\_client\_id\_fkey FOREIGN KEY (client\_id)

REFERENCES dz.clients (id\_client) MATCH SIMPLE

ON UPDATE CASCADE ON DELETE SET NULL

)

WITH (

OIDS=FALSE

);

ALTER TABLE dz.orders

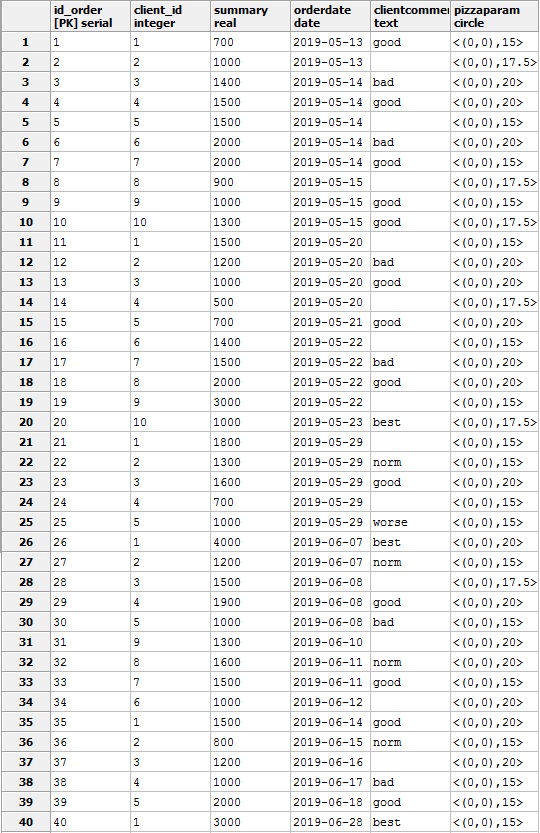
OWNER TO postgres;

Заполним таблицы данными:

1. Таблица Clients



1. Таблица Orders



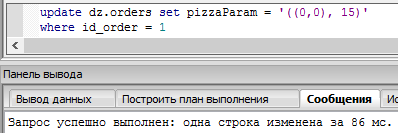
1. **Указать в таблицах атрибуты составных типов (геометрические типы и JSON) и продемонстрировать работу с ними, используя встроенные функции – 1 балл.**

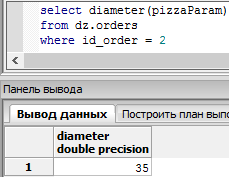
*Работа с геометрическим типом:*

Создание атрибута размер пиццы:

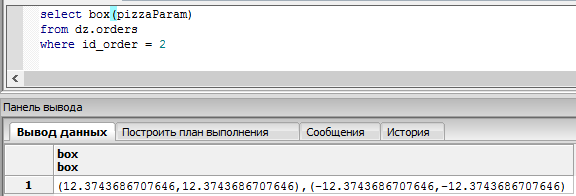
alter table dz.orders add column pizzaParam circle

Добавим значения:

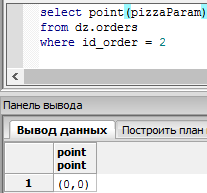


Найдем диаметр пиццы в заказе 2:  
  


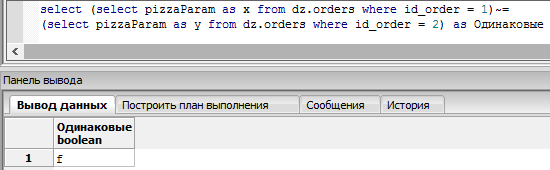
Какие будут размеры у пиццы в заказе под номером 2, если ее преобразовать из окружности в прямоугольник:



Найти центральную точку у пиццы в заказе под номером 2:



Одинаковы ли размеры пиццы в заказах под номерами 1 и 2 соответственно:

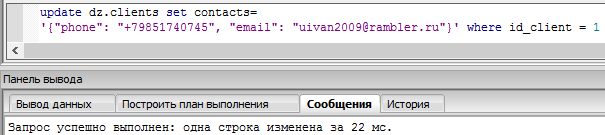


*Работа с json типом:*

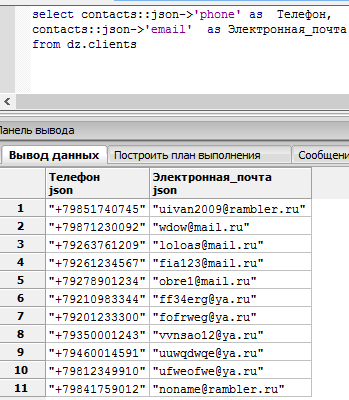
Добавить к таблице клиенты поле контакты типа json:

alter table dz.clients add column contacts json

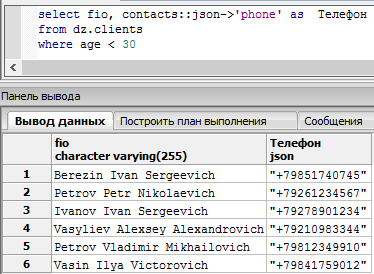
Добавим значения:



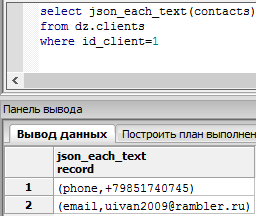
Вывести все контакты клиентов:



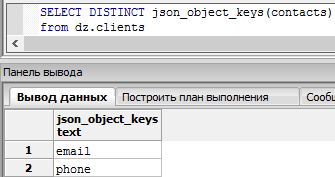
Вывести все номера клиентов, возраст которых меньше 30:



Развернем внешний объект json в набор пар ключ/значение (key/value). Возвращаемые значения имеют тип text:

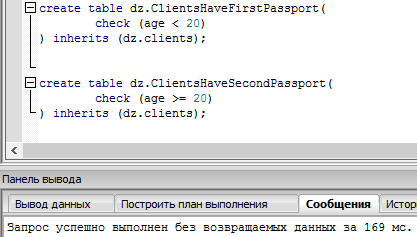


Возвращаем набор ключей во внешнем объекте json:

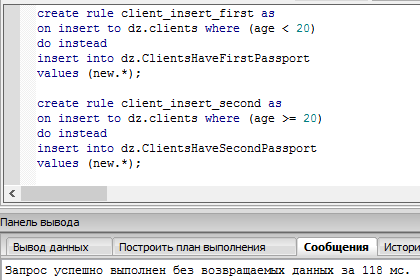


1. **Реализовать секционирование таблиц (две секции от базовой таблицы, наследование с ограничением и триггеры INSTEAD OF) и продемонстрировать CRUD – до 2 баллов.**

*Две секции от базовой таблицы Clients, наследование с ограничением:*



*Создаем правила для вставки в таблицы-наследники с помощью триггера INSTEAD OF:*



*Добавим записи и проверим корректность вышенаписанных правил для вставки:*

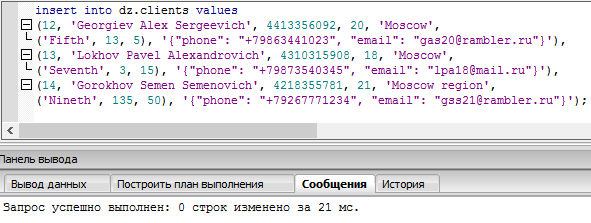


Таблица Clients:

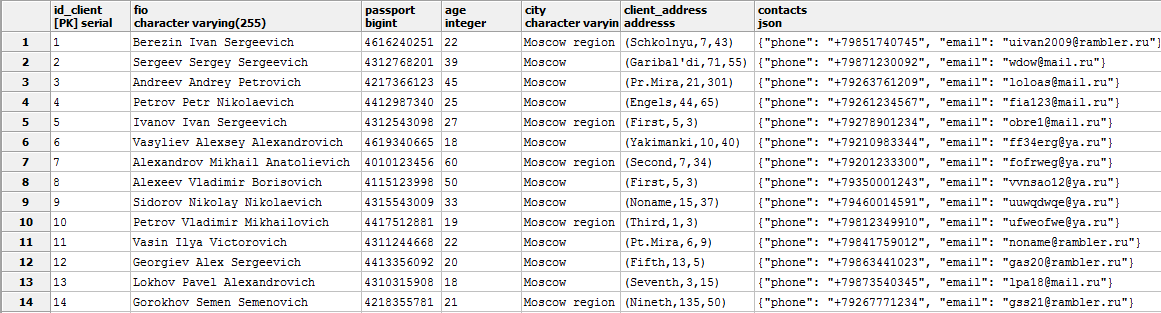


Таблица ClientsHaveFirstPassport:

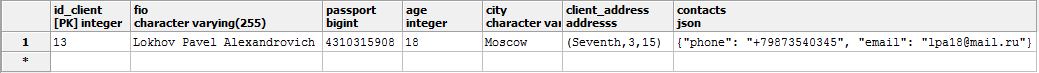
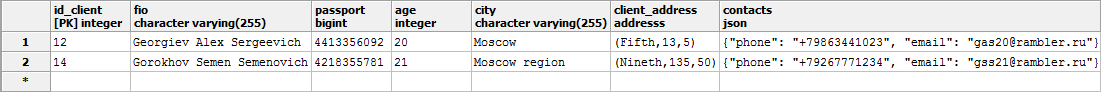
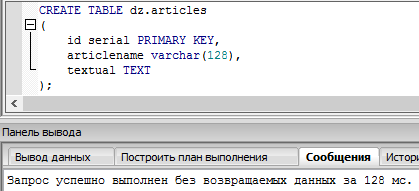


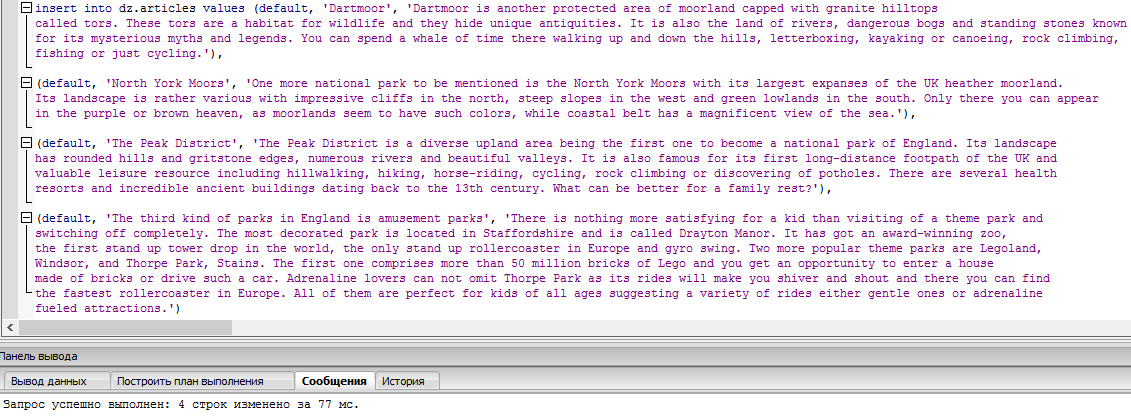
Таблица ClientsHaveSecondPassport:

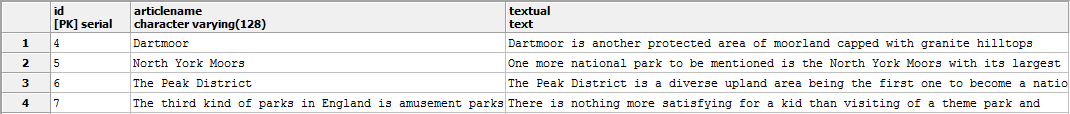


1. **Продемонстрировать полнотекстовый поиск, используя – до 3 баллов:**

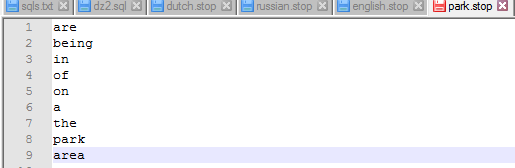
*Создадим таблицу, содержащую информацию о статьях, и наполним данными:*

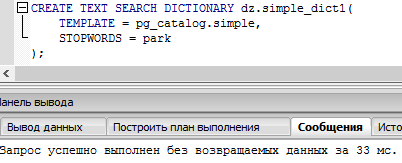


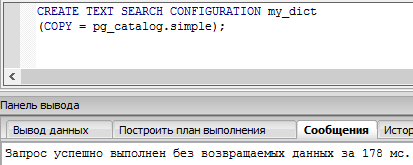


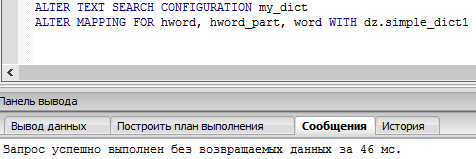


*Создадим свой словарь с файлом со стоп-словами:*



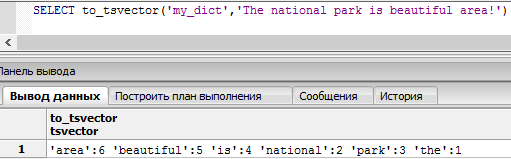






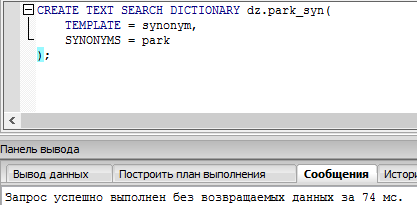
*Проверим работу словаря (пусто – слово уже имеется, появляется введенное слово – добавление его в наш словарь:*

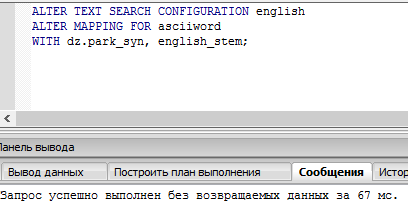
|  |  |
| --- | --- |
|  |  |

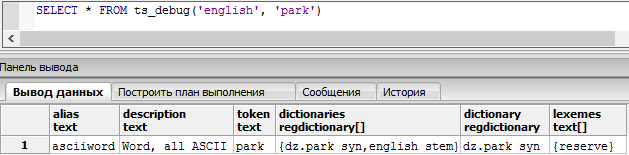


*Создадим словарь синонимов:*



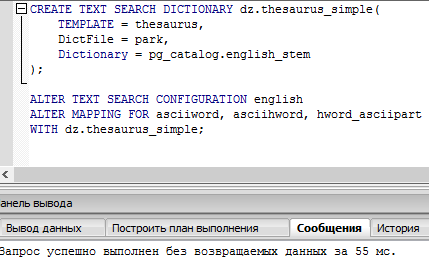


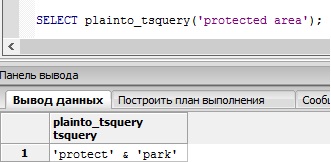




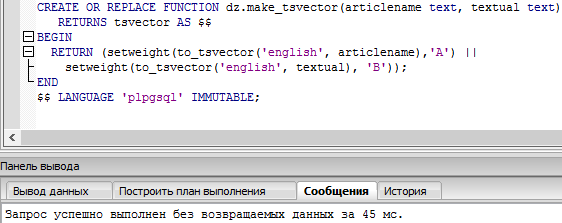
*Создадим тезариус:*





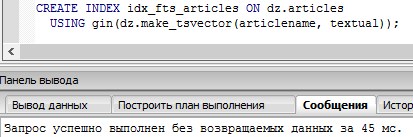
******

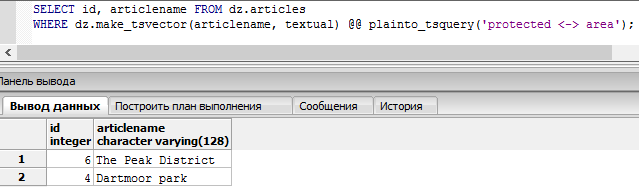
*Чтобы избежать дублирования кода, создадим следующую хранимую процедуру на plpgsql:*



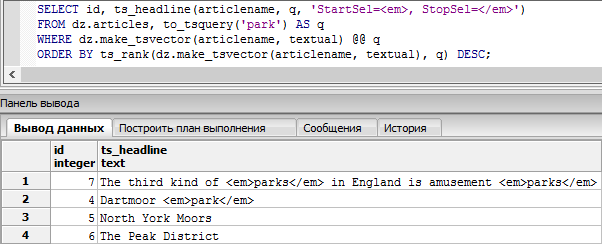
Данная процедура принимает заголовок и содержимое статьи и возвращает соответствующий ей tsvector. Здесь используется процедура setweight, с тем, чтобы придать заголовкам статьи больший вес, чем содержимому. Веса потребуются нам чуть позже для ранжирования статей. Представленная процедура была объявлена, как immutable, чтобы ее можно было использовать при построении индекса.

*Построим индекс и попробуем поискать с его помощью статьи:*





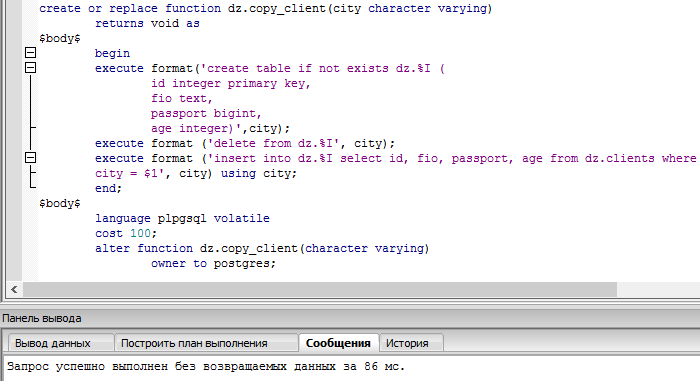
*Ранжирование и подсветка:*

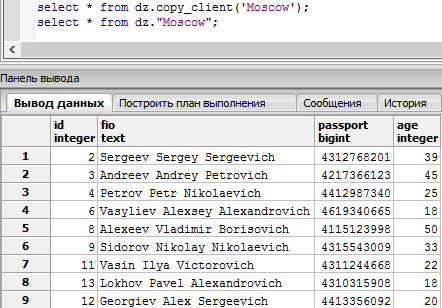


1. **Написать хранимые функции и применить при их описании (до 2 баллов):**

* **условия и динамический запрос с вариацией его команд - 1 балл,**
* **циклы и работу с курсором – 1 балл.**

*Создадим таблицу (имя таблицы – название города) и заполнить id, ФИО, паспорт и возраст клиента, проживающих в городе, например, в Москве.*





*Вывести все заказы из конкретного города:*

