1. Web-сервис(CRUD)

Написать сервис, который будет слушать входящие запросы по HTTP, преобразовывать их в запрос к соответствующей функции Postgres (по схеме трансляции, приведённой ниже), выполнять запрос и возвращать ответ клиенту.

Как плюс: ограничить максимальное количество одновременных коннектов к БД.

Настройки соединения с сервером Postgres читать из ENV среды:

\* port - (int) порт, на котором слушать запросы

\* endpoint - (string) название API

\* host - (string) hostname, где установлен Postgres

\* user - (string) имя пользователя Postgres

\* password - (string) пароль пользователя Postgres

\* schema - (string) схема в Postgres

Трансляция запроса в вызов Postgres функции

Формат запроса к сервису:

HTTP\_METHOD server:port/endpoint/vversion[/object/id...]]/destination/[id] , где

HTTP\_METHOD - одно из: GET, POST, PUT, DELETE

server - сервер, где запущен веб-сервис

port - порт

endpoint - значение из config-файла

version - номер версии API, число

/object/id - необязательный повторяющийся параметр, определяющий путь в иерархии объектов

/destination/ - конечный объект

id - id конечного объекта. Обязателен для методов PUT, DELETE, не указывается для POST. Для GET -- если указан, то возвращает элемент с данным id, если не указан, возвращает полный список элементов.

Правила трансляции

запрос в Postgres = select \* from схема.[object1[\_object2]...]\_destination\_method( [id1[, id2]... ,] id[, params])

В зависимости от HTTP метода к имени функции добавляется cуффикс method:

для GET - get

для POST - ins

для PUT - upd

для DELETE - del

Для POST и PUT методов в теле запроса принимается JSON, который передаётся в Postgres в качестве параметра params.

Все методы должны возвращать результат работы соответствующей Postgres функции с ContentType = 'application/json'

create schema if not exists test;

create sequence if not exists test.seq\_users;

create sequence if not exists test.seq\_comments;

create table if not exists test.users

(

id int not null default nextval('test.seq\_users'::regclass),

name varchar not null,

email varchar not null,

constraint "PK\_users" primary key (id),

constraint "UQ\_users\_email" unique (email),

constraint "CHK\_users\_email" check (email like '%@%')

);

create table if not exists test.comments

(

id int not null default nextval('test.seq\_comments'::regclass),

id\_user int not null,

txt varchar not null,

constraint "PK\_comments" primary key (id)

);

2. Задачи

2.2 На вход функция получает a = []int{-1,2,22,213123123,0,1} и исходное значение. Необходимо вернуть набор комбинаций из 3-ёх чисел

которые в сумме дадут исходное значение. Комбинации должны быть уникальными(1,0,2) == (2,1,0)