Тестовое задание
moyklass.com
2019-09-04

Тестовое задание на разработку отчета по занятиям

Краткое описание.

Требуется создать веб-сервер на базе KoaJS, который будет работать с данными по занятиям. Данные хранятся в СУБД PostgreSQL, дамп тестовых данных прилагается к тестовому заданию.
Предлагается сделать 2 задачи. Первая - запрос данных, вторая - манипуляция с данными.
Исполнителю предлагается сделать задачу, выбирая адекватные инструменты и общепринятые способы организации кода и API-интерфейсов, учитывая указанные в задании требования. Необходимо написать тесты для созданных методов.
При разработке учитывать, что данных может быть очень много (миллионы занятий).

Таблицы базы данных

Таблица **lessons**.
Содержит занятия. Поля: id, date (дата занятия), title (описание, тема занятия), status (статус занятия, 1-проведено, 0 - не проведено)
Таблица **teachers**
Содержит учителей. Поля: id, name (имя)
Таблица **students**
Содержит учеников. Поля: id, name (имя)
Таблица **lesson_teachers**
Связка учителей и занятий - кто какое занятие ведет. Может быть, что одно занятие ведет несколько учителей.
Таблица **lesson_students**
Связка учеников и занятий - кто на какое занятие записан. Есть дополнительное поле visit - посетил ученик занятие, или нет.

Общие требования

Язык: JavaScript
Веб-сервер: KoaJS
Версия NodeJS: 10 или выше
Работа с СУБД через knex или sequelizeJS
Используемый Content-Type при работе - application/json
Код должен быть выложен на GitHub, GitLab или BitBucket, система контроля версий - git.

Задача 1. Запрос данных.

Сделать корневой метод «/«, который осуществляет поиск по данным и возвращает массив объектов - занятий.
Метод принимает параметры фильтра. Все параметры не обязательные. Все параметры должны учитываться одновременно. С помощью параметров должна работать пагинация.

Параметры фильтра:

* **date**. Либо одна дата в формате YYYY-MM-DD, либо две в таком же формате через запятую (например, «2019-01-01,2019-09-01». Если указана одна дата, выбираются

занятия на эту дату. Если указаны 2 даты, то выбираются занятия за период, включая указанные даты.
* **status**. Статус занятия. принимается либо 0 (не проведено), либо 1 (проведено)
* **teacherIds**. id учителей через запятую. Выбираются все занятия, которые ведет хотя бы один из указанных учителей.
* **studentsCount**. количество записанных на занятия учеников. либо одно число (тогда выбирается занятие с точным числом записанных), либо 2 числа через запятую, тогда они рассматриваются как диапазон и выбираются занятия с количеством записанных, попадающих в диапазон включительно.
* **page**. Номер возвращаемой страницы. первая страница - 1.
* **lessonsPerPage**. Количество занятий на странице. По-умолчанию - 5 занятий.

В случае некорректных данных метод должен возвращать ошибку 400 с описанием ошибки (формат на выбор исполнителя).
В нормальном случае возвращается массив объектов-занятий. Каждый объект должен иметь вид:

```
{
  id : 9 // id занятия
  date: '2019-09-01' // Дата занятия
  title: 'Orange', // Тема занятия
  status: 1 // Статус занятия
  visitCount: 3, // Количество учеников, посетивших занятие (по полю visit)
  students: [ // Массив учеников, записанных на занятие
    { id: 1, // id ученика
    name: 'Ivan' // имя
    visit: true,
    }
  ],
  teachers: [ // Массив учителей, ведущих занятие
    { id: 1, // id учителя
    name: 'Tanya' // имя
    }
  ]
}
```

Задача 2. Создание занятий.

Сделать метод /lessons, который будет создавать одно или несколько занятий
Входные данные в виде json-объекта:

```
{
  teacherIds: [1,2], // id учителей, ведущих занятия
  title: 'Blue Ocean', // Тема занятия. Одинаковая на все создаваемые занятия
  days: [0,1,3,6], // Дни недели, по которым нужно создать занятия, где 0 - это
воскресенье
  firstDate: '2019-09-10', // Первая дата, от которой нужно создавать занятия
  lessonsCount: 9, // Количество занятий для создания
  lastDate: '2019-12-31', // Последняя дата, до которой нужно создавать занятия.
}
```

Параметры **lessonsCount** и **lastDate** взаимоисключающие, то есть должен использоваться только один из этих параметров.
Если указан **lessonsCount**, то нужно создавать занятия по указанным дням недели, начиная с **firstDate**, пока не создастся **lessonsCount** занятий.
Если указан **lastDate**, то нужно создавать занятия по указанным дням недели, начиная с **firstDate** и до даты **lastDate**.

Установить ограничение по количеству занятий - 300, и по дате - 1 год. Например, если мы указываем период 1 год и занятия каждый день, то должно создаться только 300 занятий.
Другой пример: Если мы указываем занятия только по понедельникам и количество 300, то занятия должны создастся только на год вперед (их будет около 50).
В случае некорректных данных метод должен возвращать ошибку 400 с описанием ошибки (формат на выбор исполнителя).
При успешном создании занятий должен возвращаться массив с id созданных занятий.
ы