**Задачи на тему Spring Boot Data + REST + Swagger UI**

**Технологии и фреймворки, используемые в работе:**

* **Spring Boot Data JPA**<https://spring.io/guides/gs/accessing-data-jpa/>  
  <https://spring.io/guides/gs/accessing-data-rest/>  
  <https://www.baeldung.com/jpa-entities>  
  <https://www.baeldung.com/the-persistence-layer-with-spring-data-jpa>  
  <https://www.baeldung.com/the-persistence-layer-with-spring-and-jpa>  
  <https://www.baeldung.com/jpa-mapping-single-entity-to-multiple-tables>
* **Spring Boot REST**<https://spring.io/guides/gs/rest-service/>  
  <https://www.baeldung.com/spring-mvc-controller-custom-http-status-code>  
  <https://www.baeldung.com/spring-response-status>
* **Swagger UI**<https://www.baeldung.com/swagger-2-documentation-for-spring-rest-api>
* **Jackson**<https://www.baeldung.com/jackson>
* **PostgreSQL**<https://www.postgresql.org/>

**Предлагаемая архитектура приложения**

Spring Boot application

HTTP Request

Service layer

REST Controller

Repository layer

HTTP Response

PostgreSQLK

…/swagger-ui.html

Swagger UI

**Заданы следующие таблицы**

I. ПОКУПАТЕЛЬ

|  |  |  |  |
| --- | --- | --- | --- |
| ИДЕНТИФИКАТОР | ФАМИЛИЯ | РАЙОН ПРОЖИВАНИЯ | СКИДКА, % |

**II. МАГАЗИН**

|  |  |  |  |
| --- | --- | --- | --- |
| ИДЕНТИФИКАТОР | НАЗВАНИЕ | РАЙОН РАЗМЕЩЕНИЯ | КОММИСИОННЫЕ % |

**III. КНИГИ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ИДЕНТИФИКАТОР | НАЗВАНИЕ | СТОИМОСТЬ, РУБ. | СКЛАД | КОЛ-ВО |

**IV. ПОКУПКА**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| НОМЕР ЗАКАЗА | ДАТА | ПРОДАВЕЦ | ПОКУПАТЕЛЬ | КНИГА | КОЛ-ВО | СУММА, РУБ |

**Таблицы I,II,III связаны с таблицей IV следующим образом:**

ПРОДАВЕЦ (FK) 🡪 МАГАЗИН.ИДЕНТИФИКАТОР(PK)

ПОКУПАТЕЛЬ(FK)🡪ПОКУПАТЕЛЬ.ИДЕНТИФИКАТОР(PK)

КНИГА(FK)🡪КНИГИ.ИДЕНТИФИКАТОР(PK)

Создать и заполнить указанные таблицы, используя одну из СУБД. Например, PostgreSQL. Тестовые данные для таблиц придумать самостоятельно. Тип данных выбрать самостоятельно.

Используя Spring Boot фреймворк реализовать Java приложение, предоставляющее следующий REST API:

**1. Реализовать базовые CRUD операции. Для каждой из таблиц необходимо обеспечить возможность**

a)удаления по id (DELETE method)

b)частичного обновления по id(PATCH method)

c)добавления(POST method)

d)вывода всех записей(GET method)

e)запрос объекта по id(GET method)

f)полной перезаписи по id (PUT method)

**2. Создать запросы для вывода:**

a)всех различных названий и стоимостей книг;

b)всех различных районов, в которых проживают покупатели;

c)всех различных месяцев, когда производились покупки.

**3. Создать запросы для получения информации о:**

* 1. фамилиях и размере скидки всех покупателей, проживающих в Нижегородском районе;
  2. названиях магазинов Сормовского или Советского районов;
  3. Названиях и стоимости книг, в которых встречается слово Windows, или стоящих более 20000 руб. Вывод результатов организовать по названию и убыванию цены книг.

1. **Для каждой покупки вывести следующие данные:**
   1. фамилию покупателя и название магазина, где производилась покупка;
   2. дату, фамилию покупателя, скидку, название и количество купленных книг.
2. **Определить:**
   1. номер заказа, фамилию покупателя и дату для покупок, в которых было продано книг на сумму не меньшую чем 60000 руб.
   2. покупки, сделанные покупателем в своем районе не ранее марта месяца. Вывести фамилию покупателя, район, дату. Произвести сортировку;
   3. магазины, расположенные в любом районе, кроме Автозаводского, где покупали книги те, у кого скидка от 10 до 15 %;
   4. данные по покупке книг (название, район складирования, количество), приобретенных в районе складирования и содержащихся в запасе более 10 штук. Включить данные о стоимости и отсортировать по возрастанию.