Contents

- Custom Queries with JPQL (Java Persistence Query Language)
- Relationships in JPA
- Minor Project (Online Library Management System)
- Project Flowchart

Custom Queries with JPQL (Java Persistence Query Language) :

The JPQL (Java Persistence Query Language) is an object-oriented query language which is used to perform database operations on persistent entities. Instead of a database table, JPQL uses entity object model to operate the SQL queries. Here, the role of JPA is to transform JPQL into SQL. Thus, it provides an easy platform for developers to handle SQL tasks.

JPQL is an extension of Entity JavaBeans Query Language (EJBQL), adding the following important features to it: -

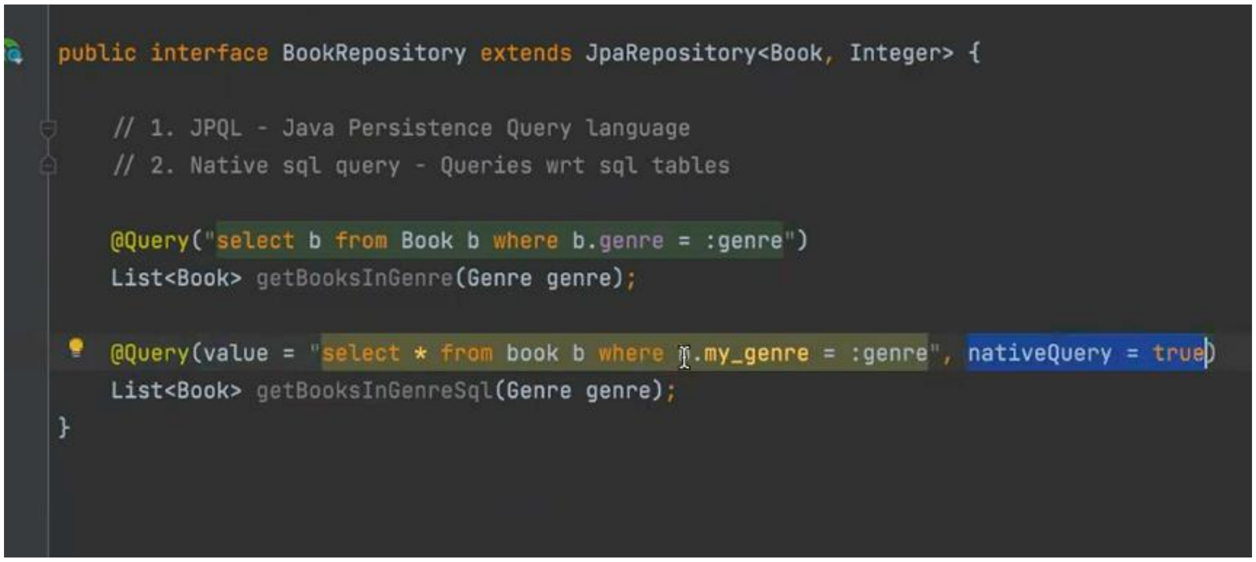
- It can perform join operations.
- It can update and delete data in a bulk.
- It can perform aggregate function with sorting and grouping clauses.
- Single and multiple value result types.

If there is any issue in the JPQL query ,we will get an error when the application server starts.

References :

https://www.geeksforgeeks.org/sql-join-cartesian-join-self-join/?ref=lbp

https://www.geeksforgeeks.org/sql-join-set-1-inner-left-right-and-full-joins/



```java
public interface BookRepository extends JpaRepository<Book, Integer> {

    // 1. JPQL - Java Persistence Query language
    // 2. Native sql query - Queries wrt sql tables

    @Query("select b from Book b where b.genre = :genre")
    List<Book> getBooksInGenre(Genre genre);

    @Query(value = "select * from book b where b.my_genre = :genre", nativeQuery = true)
    List<Book> getBooksInGenreSql(Genre genre);
}
```

Let Us start with our Minor Project (Guess from image , what we are going to build !)

Q. What is Data Modeling ?

0. Flowchart of the project

1. Decide your entities

Book

Student

Admin

Author

Request

Transaction

2. Relationships between Entities

Multiple copies of the same book will have different ids.

One book can have only one author NOT multiple authors.

Students can not raise a request to return a book he does not have .

Student can not raise an issue request if he has the max limit of books already issued i.e. 3.

One Request can have only one book,one admin , one student

For every ADMIN APPROVED request there will be a transaction created after being processed by admin . Requests can be approved or denied but transactions will be created only if Request is APPROVED by admin. In the rejection case , we just update the request status and add an admin comment in the request object .We will calculate the fine in Transaction .

3. Functionalities for each Entity

Let us now start coding in INTELLIJ

-dependencies

-application.properties

-create packages acc to mvc

--What is Enum ?

--What is @Enumerated ?

--What is foreign key ? Can it be null ?

https://www.geeksforgeeks.org/foreign-key-constraint-in-sql/

https://www.geeksforgeeks.org/how-to-create-a-table-with-a-foreign-key-in-sql/

--Let's discuss which table in the relationship should have the foreign key and which should have the back reference?

(IMP : class with back referencing is called referenced class ,class with foreign key is called referencing class.)

--How to read relationship annotations in the Entity classes ? Many(Java class)ToOne(object property of the class)

- A Request and a Transaction are mapped OneToOne . Which one should be forign key ?

-What is an ER Model ?

Note : ER Model is used to model the logical view of the system from data perspective which consists of these components.

https://www.geeksforgeeks.org/er-diagram-of-a-company

https://www.geeksforgeeks.org/introduction-of-er-model/

-Post Data Modeling is completed , we can see the ER Model for our project in DBeaver. (Dot has foreign key of crystal.) , alternatively we can use dbdiagram.io

==

Data Modeling is complete , let's focus on developing API's now .

TASK : Let's start creating Controllers , Service and Repository class for functionality of students creating a request and assign an admin based on least request count .

NOTE : Students should not be able to return a book they have not issued .

Q. Now if I don't have the book with bookId as 40 and also don't have a student with studentId as 1 . What happens if I try to create a request from Postman using these bookId and studentId.What will happen ?

--Imp thing is to understand that first Validation checks (400) are happening and then foreign key validation will take place(500) .

TASK : Let's create controllers ,Service and Repository class for functionality of creating and fetching a Book and Student .

Note : We would not have a controller for Author as Author will be onboarded along with the book . So we will be creating an author at the time of creating the book .

Before creating an author , we have to check if it already exists in db.

NOTE :

1. If the author is not created, we need to create the author

2. Fetch the authorId and attach it in the book object

3. Save the book object

--Why do we need to (CRUD)Requests and Responses classes ?

--How to make an API request through terminal ?

-- When we fetch the details of Admin 1 , why does the json response go into an infinite loop and give a Null pointer exception in the logs ?

HINT : JsonIgnoreProperties("admin") put over requestList in the Admin class.

TASK : Now do JsonIgnoreProperties for all Entity classes to stop the echo.

HomeWork

TASK : Request to Transaction creation

TASK : Complete the logic for calculateFine function in transaction service class .

TASK : Can we do something to accommodate these 4 tasks in one api :

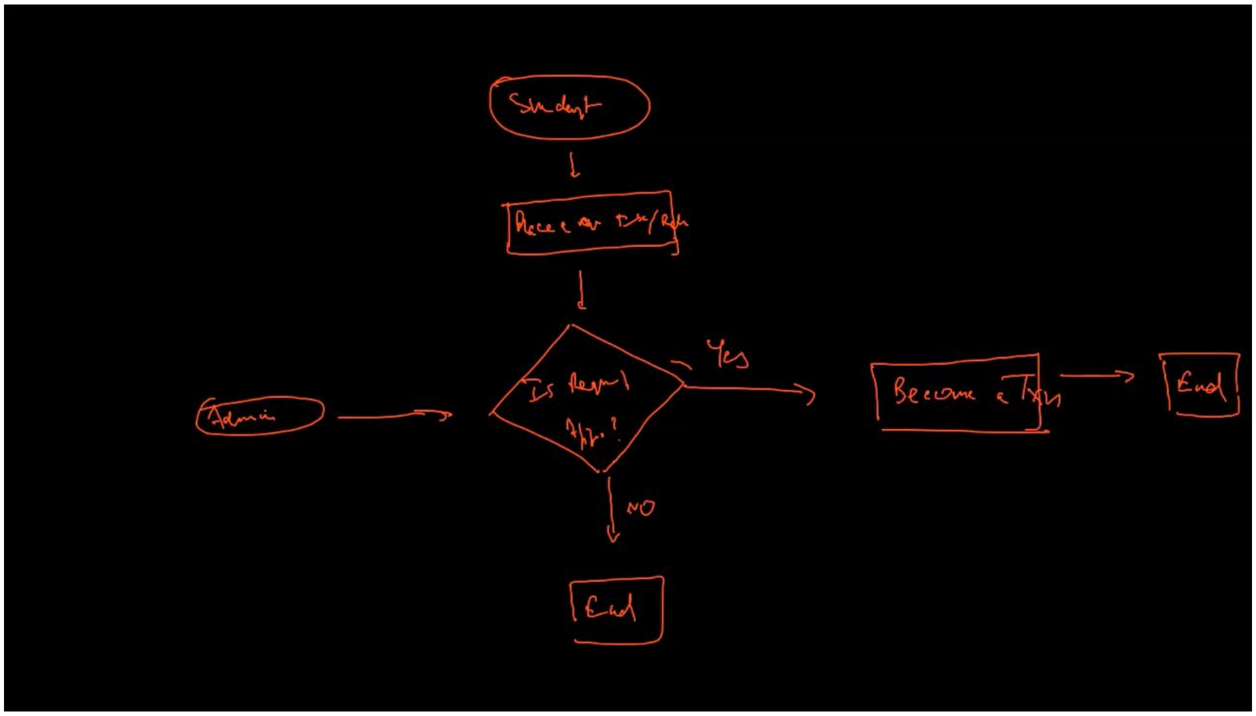
1 : Create an API to get all books of a particular genre

2 : Create an API to get all books written by a particular author

3 : Create an API to get all available books which are not yet issued to anyone .

4 : Create an API to get all books with a given book id .

COMPLETE FLOWCHART :


Student
Place a new Tx/Reqn
Is Request Appr?
Yes
Become a Txn
End
Admin
NO
End

LIVE
Java
Backend
Development