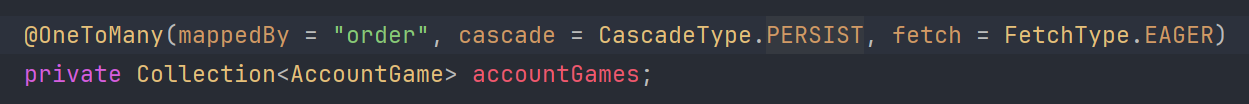
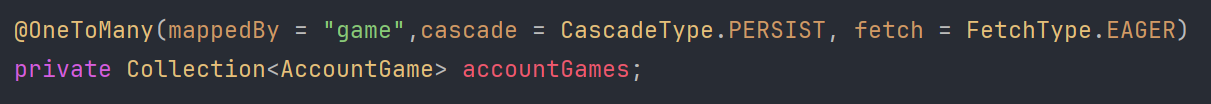
Một lỗi của CascadeType ảnh hưởng đến 3 bảng dưới

Ảnh có chứa văn bản, Phông chữ, số, biểu đồ

Mô tả được tạo tự động



Nếu orders quan hệ account\_game 1 nhìu và sử dụng CascadeType.All thay vì CascadeType.PERSIST



Tương tự nếu game quan hệ account\_game 1 nhìu và sử dụng CascadeType.All thay vì CascadeType.PERSIST

Thì hypernate sẽ hiểu lầm rằng account\_game là bảng trung gian và nối bảng orders với game thành 1 nhìu luôn gây dư thừa dữ liệu.

**NÊU SỬ DỤNG PERSIST THAY VÌ ALL**

**Ngoài ra**

**# neu muon thay schema nay bằng 1 schema moi thi phai doi update thanh create**

**spring.jpa.hibernate.ddl-auto = update**

| **Cascade Type** | **Hành vi** | **Khi nào nên sử dụng** |
| --- | --- | --- |

|  |  |  |
| --- | --- | --- |
| **PERSIST** | **Lưu tự động các thực thể con chưa tồn tại** | **Khi thực thể con phụ thuộc vào thực thể cha** |

|  |  |  |
| --- | --- | --- |
| **MERGE** | **Cập nhật tự động các thực thể con** | **Khi muốn đồng bộ các thay đổi của cha và con** |

|  |  |  |
| --- | --- | --- |
| **REMOVE** | **Xóa tự động các thực thể con** | **Khi thực thể con không có ý nghĩa độc lập** |

|  |  |  |
| --- | --- | --- |
| **REFRESH** | **Làm mới tự động các thực thể con** | **Khi cần đảm bảo trạng thái của cha và con khớp với cơ sở dữ liệu** |

|  |  |  |
| --- | --- | --- |
| **DETACH** | **Tách tự động các thực thể con khỏi ngữ cảnh** | **Khi không muốn cập nhật các thay đổi của thực thể cha và con** |

|  |  |  |
| --- | --- | --- |
| **ALL** | **Áp dụng tất cả các hành vi trên** | **Khi muốn tất cả các thao tác lan truyền** |

**Add 1 danh sách**

public class Main {

public static void main(String[] args) {

// Tạo danh sách ban đầu

List<Float> number = new ArrayList<>();

// Tạo danh sách các phần tử cần thêm

List<Float> additionalNumbers = new ArrayList<>();

additionalNumbers.add(3.3f);

additionalNumbers.add(4.4f);

additionalNumbers.add(5.5f);

// Thêm toàn bộ phần tử của danh sách mới vào danh sách ban đầu

number.addAll(additionalNumbers);

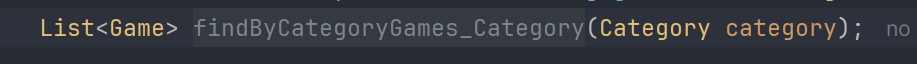
// In danh sách sau khi thêm

System.out.println("Danh sách sau khi thêm: " + number);

}

}

Giải thích dòng code này trong file gameRepository



Dòng code này là một **phương thức tùy chỉnh trong repository** Spring Data JPA. Nó được viết theo cú pháp **method naming convention** của Spring Data để tự động tạo truy vấn SQL mà không cần viết thủ công.

**Phân Tích Cú Pháp**

List<Game> findByCategoryGames\_Category(Category category);

**1. List<Game>**

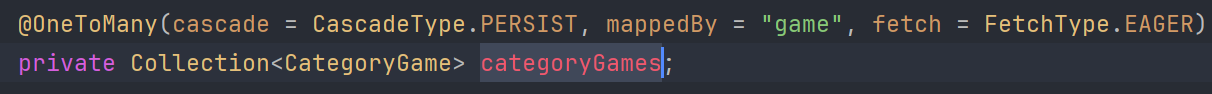
* **Loại trả về**: Một danh sách các đối tượng Game.
* Dữ liệu được trả về là danh sách các game khớp với điều kiện tìm kiếm.

**2. findBy**

* **Bắt đầu tên phương thức**: Quy ước của Spring Data JPA để tạo truy vấn.
* **findBy** biểu thị rằng phương thức này sẽ tìm kiếm các đối tượng dựa trên điều kiện.

**3. CategoryGames\_Category**

* **Đường dẫn thuộc tính**: Thể hiện quan hệ giữa các bảng.
  + CategoryGames: Là trường trong Game (quan hệ OneToMany) kết nối với bảng CategoryGame.



* + \_Category: Chỉ ra rằng điều kiện tìm kiếm nằm trên thuộc tính Category trong CategoryGame.

Ảnh có chứa văn bản, Phông chữ, ảnh chụp màn hình

Mô tả được tạo tự động

**4. Category category**

* **Tham số**: Một đối tượng Category được truyền vào để làm điều kiện tìm kiếm.
* Spring Data JPA sẽ tự động ánh xạ đối tượng Category sang cột liên quan trong bảng.

**Cách Hoạt Động**

Phương thức này yêu cầu JPA thực hiện truy vấn để tìm tất cả các game liên kết với một thể loại (Category) thông qua bảng trung gian CategoryGame.

**Tương Đương SQL**

Giả sử bạn có các bảng:

* game: Lưu thông tin game.
* category: Lưu thông tin thể loại.
* category\_game: Bảng trung gian liên kết game và category.

SQL tương đương:

SELECT g.\*

FROM game g

JOIN category\_game cg ON g.id = cg.game\_id

WHERE category\_game.category\_id = :categoryId;

**Tóm Lược**

* Phương thức findByCategoryGames\_Category giúp tự động tìm tất cả các Game liên quan đến một Category thông qua bảng trung gian CategoryGame.
* Spring Data JPA sẽ tự động ánh xạ quan hệ và tạo truy vấn SQL tương ứng dựa trên tên phương thức và quan hệ giữa các entity.

Có nghĩa là List<Game> findByCategoryGames\_Category(Category category);

**Nó sẽ tìm all cả game có categoryId ở bảng categoryGame bằng với categoryId ở bảng category**

Câu query truy vấn account game đã bán theo gameId sắp xếm giảm dần theo số lượng game đã bán.

*Chọn cột game\_id từ bảng account\_game, đếm game\_id trong account\_game và đặt tên là total\_game từ bảng account\_game, điều kiện là account\_game.status là 1 (1 là trạng thái những tài khoản đã bán) nhóm lại thoe game\_id trong account\_game, sắp xếp giảm dần theo số lượng account đã bán theo account\_game.game\_id*

**SELECT account\_game.game\_id,**

**COUNT(account\_game.game\_id) as total\_game**

**FROM account\_game**

**WHERE account\_game.status = 1**

**GROUP BY account\_game.game\_id**

**ORDER BY total\_game DESC;**

**Câu query trên nhưng làm việc với entiry**

** AccountGame là Entity trong Java:**

** ag.game.gameId:**

* **Tham chiếu đến trường gameId thông qua quan hệ với Entity Game.**

** COUNT(AccountGame.game.gameId):**

* **Đếm số lần gameId xuất hiện trong các bản ghi có status = 1.**

** GROUP BY AccountGame.game.gameId:**

* **Gom nhóm kết quả theo gameId.**

** ORDER BY totalGame DESC:**

* **Sắp xếp theo số lượng đã bán (totalGame) giảm dần.**

** List<Object[]>:**

* **Kết quả của truy vấn sẽ là danh sách các mảng đối tượng (Object[]), trong đó:**
  + **Object[0]: Là gameId.**
  + **Object[1]: Là totalGame.**

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động**

Status mình xài kiểu boolean trong database nó là 1 tương ứng với true

Ở trên query bị lỗi vì spring jpa yêu cầu phải có alias từ những thực thể được truy xuất

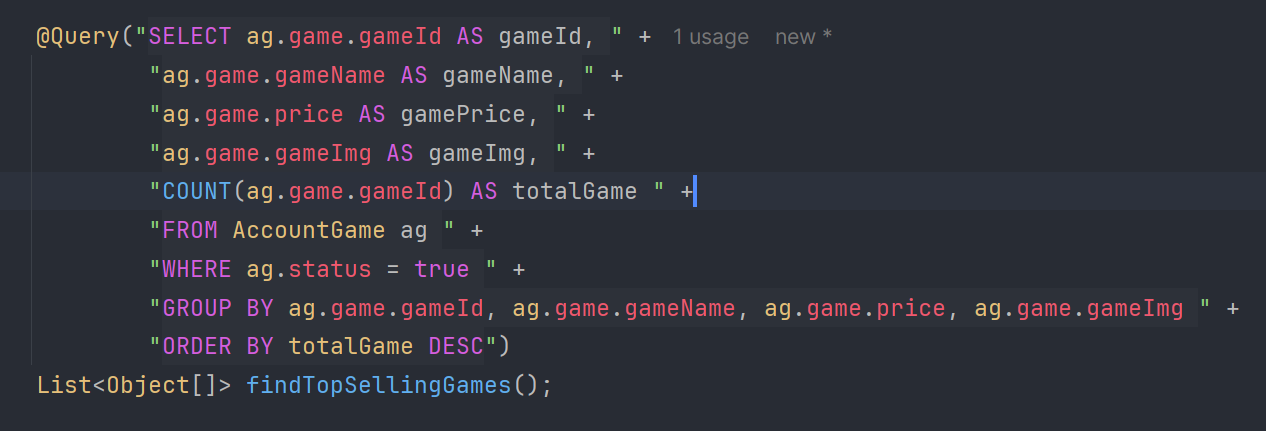
Còn lại những thứ khác vì vẫn đúng

**Bạn đang gọi đầy đủ AccountGame thay vì sử dụng alias. JPQL yêu cầu phải sử dụng alias nếu muốn thao tác với các trường. Ở dưới câu query đúng**

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động**

Đây là câu query phiên bản cuối ở trên chỉ select gameId nhưng ở dưới mình muốn select thêm 3 thuộc tính nữa vẫn count theo gameId



Nhưng thay vì sử dụng object thì mình dùng GameSaleDto để nhận dữ liệu

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động

 **Constructor Expression:**

* Phần new com.webgame.webgame.dto.GameSalesDTO(...) trong query cho phép JPA gọi trực tiếp constructor của DTO và truyền kết quả query vào.
* Các trường trong SELECT phải trùng với tham số trong constructor của DTO.

 **GROUP BY và COUNT:**

* GROUP BY bảo đảm kết quả được nhóm theo gameId, gameName, gamePrice, và gameImg.
* COUNT(ag.game.gameId) đếm số lượng tài khoản bán được cho từng game.

 **Trả về danh sách DTO:**

* Thay vì danh sách các mảng đối tượng (List<Object[]>), kết quả trả về sẽ là danh sách các đối tượng GameSalesDTO.