Pro Java №38

Аннотация `@OneToMany`

- **Описание**: Связь "один ко многим". Одна сущность связана с несколькими объектами другой сущности.
- Таблица, связанная с этой сущностью, содержит внешние ключи, указывающие на родительскую таблицу.

Аннотация `@ManyToOne`

- **Описание**: Связь "многие к одному". Несколько объектов одной сущности связаны с одним объектом другой сущности.
- Внешний ключ в дочерней таблице ссылается на родительскую сущность.

Аннотация `@OneToOne`

- **Описание**: Связь "один к одному". Один объект одной сущности связан с одним объектом другой сущности.
- Эта связь часто используется для уникальных, связанных данных, таких как профиль пользователя.

Аннотация `@ManyToMany`

- **Описание**: Связь "многие ко многим". Несколько объектов одной сущности могут быть связаны с несколькими объектами другой сущности.
- Для таких связей обычно создаётся промежуточная таблица.

Основные параметры:

- `mappedBy`: Указывает, какая сторона связи является владельцем (для `@OneToMany` и `@ManyToMany`).
- `cascade`: Определяет каскадное поведение.
- `fetch`: Указывает, как данные загружаются – лениво (`FetchType.LAZY`) или сразу (`FetchType.EAGER`).

Эти аннотации позволяют легко работать с объектами и связями, избегая написания сложных SQL-запросов.

Аннотация `@RequiredArgsConstructor` – это аннотация из библиотеки Lombok, которая автоматически генерирует конструктор для класса. Она

добавляет конструктор для всех **нефинальных полей**, помеченных аннотацией `@NonNull`, а также для полей, которые являются `final`.

Вот основные отличия между односторонней и двунаправленной связями:

1. **Односторонняя связь**:
    - Одна сторона связи "знает" о другой стороне.
    - Работает только в одном направлении: от родительской сущности к дочерней или наоборот.
    - Используется для упрощённых и изолированных отношений между объектами.
2. **Двунаправленная связь**:
    - Обе стороны "знают" о существовании друг друга.
    - Работает в обоих направлениях: одна сущность может получить данные другой и наоборот.
    - Позволяет более сложное взаимодействие между связанными объектами, но требует тщательной настройки и поддержки согласованности данных.

Аннотации `@JsonManagedReference` и `@JsonBackReference` из библиотеки Jackson используются для предотвращения бесконечной рекурсии при сериализации/десериализации объектов с двунаправленными связями. Эти аннотации позволяют указать, какая сторона связи должна сериализоваться, а какая – нет, чтобы избежать зацикливания.

Проблема бесконечной рекурсии:

Когда два объекта ссылаются друг на друга (например, в двунаправленной связи), сериализация может зациклиться, так как Jackson будет пытаться сериализовать обе стороны связи бесконечно.

Роль аннотаций:

1. `@JsonManagedReference`:
    - Помечает "родительскую" сторону связи, которая будет сериализована.
    - Эта сторона будет включена в JSON-результат.
2. `@JsonBackReference`:
    - Помечает "дочернюю" или обратную сторону связи, которая **не будет** сериализована.
    - Вместо этого она пропускается, чтобы избежать бесконечной рекурсии.

Где использовать:

- `@JsonManagedReference` используется на стороне, где находится коллекция объектов или ссылка на связанные данные, которые вы хотите включить в JSON.
- `@JsonBackReference` используется на противоположной стороне связи, которая не должна включаться в JSON.

Аннотация `@Enumerated` в JPA используется для указания, как нужно сохранять значения перечислений (`enum`) в базе данных. Она применяется для полей, которые являются экземплярами перечислений, и позволяет задать, будет ли значение храниться в виде имени перечисления (текстом) или его порядкового индекса.

Аннотация `@Builder` из библиотеки Lombok используется для автоматической генерации шаблона построителя (Builder pattern) для класса. Это позволяет создавать объекты класса с помощью удобного и читаемого подхода, избегая конструкций с множеством аргументов или необходимости вызова множества сеттеров.