**Каскадирование в Hibernate**

Для описания механизма каскадирования в **Hibernate** рассмотрми пример:

Создадим новый объект класса **Person** и создадим новй товар(**item**) для этого человека. Этот товар принадлежит человеку **Person**. Теперь настроим обратную связь, укажем, что у человека в списке товаров лежит товар. Дальше сохраняем человека.



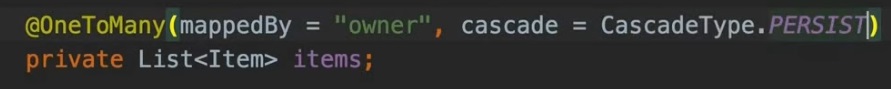
Теперь после сохранения человека посмотрим какой **sql** был сгенерирован: мы видим, что у нас был сгенерирован только **sql** который сохраняет человека, но связанный с ним товар не сохранился в базу данных.



**Теперь возникает вопрос как сделать так, чтобы Hibernate автоматически сохранял дочернюю сущность в том сулчае если мы сохраняем родительскую сущность.**

Для этого есть несколько вариантов(с помощью метода ***save()*** и с помощью метода ***persist()***):

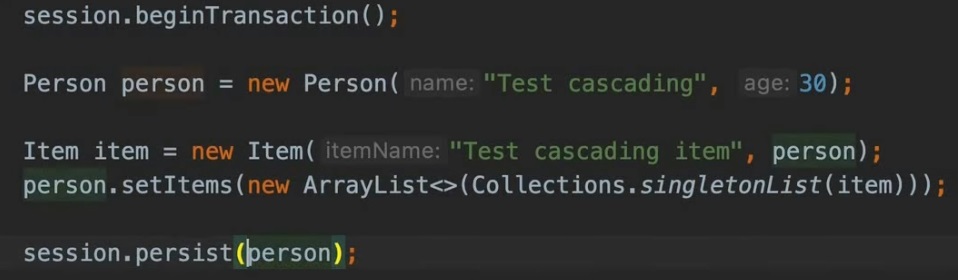
**1.** Перейти в родительскую сущность(обычно каскадирирование настраивается именно в родительской сущности) и в аннотации **@OneToMany** нужно указать дополнительный аргумент **cascade**. В этом аргументе мы указываем те опперации которые будут каскадироваться.

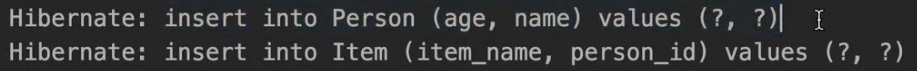


В данном примере выбранно значение аргумента **PERSIST**. Теперь при сохранении человека у нас так же будут сохраняться все связанные с ним товары, тоесть будеть производится каскадирование.

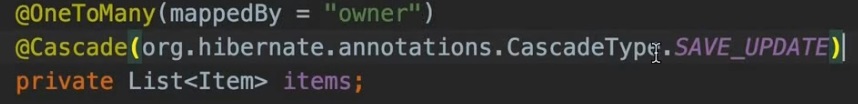
Теперь чтобы всё правильно сохранилось нужно поменять метод ***save()*** на метод ***persist()***.

Тоесть этот тип каскадирования **PERSIST** будет работать только с методом **persist()**.

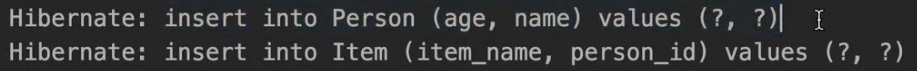
  
  
  
  
  
  
  
  
Теперь при вызове метода ***persist()*** у нас генерирнуются две **sql** команды:



**2.** Для того, чтобы настроить каскадирования для метода ***save()*** нужно использовать аннотацию **@Cascade** и в аругментах указать тип операции которые будут каскадироваться:



Теперь при вызове метода ***save()*** у нас генерирнуются так же две **sql** команды:



Разница между методом ***save()*** и методом ***persist():***

**JPA** – стандартная спецификация для **java**, которая предназначена для того, чтобы сохранять объеты в базы данных. А **Hibernate** это всего лишь один из **JPA** библиотек для **java**. Есть еще много других библиотек которые занимаются сохранением **java** объектов в базу данных. Все эти библиотеки подчиняются единой спецификации — спецификации **JPA**.

Метод ***save()***

* Этот метод есть только в библиотеке **Hibernate**(есть только у Hibernate, нет у других JPA библиотек).
* Возвращает значение первичного ключа для добавления сущности. Этот метод возрващает **Serialuzible**.
* Гарантирует, что значение первичного ключа будет определено сразу после вызова метода.



Метод ***persist()***

* Этот метод есть в библиотеке JPA(Java Persistance API) поэтому любой JPA провайдер будет иметь этот метод.
* Ничего не возвращает.
* Не гарантирует, что значение первичного ключа будет определено сразу после вызова метода.



В целом оба метода очень похожи и большенстве случаев нет разницы, какой из них использовать.

**Некоторые операции для аннотации @Cascade:**

**1. DETACH** - если родительскую сущность уберем из **Persistance** контекста, то связанные с этим объектом сущности тоже будут убраны из **Persistance** контекста.



**2. MERGE**- если возвращаем объект в **Persistance** контекст, то все связанные с этим объектом другие объекты так же будут возвращены в **Persistance** контекст.

  
  
**3. REFRESH**- возникает тогда когда мы на сессии вызываем метод ***refresh()* -** он обращается к базе данных и в наш кэше который есть в **hibernate** обновляет значения объекта и значения всех объектов которые с ним связанны. Н]ъужно для того, чтобы наш **java** сущности были идентичны данным из таблицы базы данных.  
  
  
**4. ALL** – будут выбраны все операции каскадирования. НО здесь нужно быть аккуратным так как не всегда нужно что все опперации каскадировались.  
