FAQ: @JoinColumn ... where does it find the column?

@JoinColum ... where does it find the column?

Question

In the Course class,we have OneToMany relation with reviews with join column course_id.

But in course table we do not have column course_id.

Ideally when we say @JoinColumn a new column should be created in course table ... isn't it?

How does @JoinColum know where to find the join column?

Answer

The JoinColumn is actually fairly complex and it goes through a number of advanced steps to find the desired column.

This info below is from the documentation

Source:
http://docs.oracle.com/javaee/7/api/javax/persistence/JoinColumn.html#name--

---

The table in which it is found depends upon the context.

- If the join is for a OneToOne or ManyToOne mapping using a foreign key mapping strategy, the foreign key column is in the table of the source entity or embeddable.

- If the join is for a unidirectional OneToMany mapping using a foreign key mapping strategy, the foreign key is in the table of the target entity.

- If the join is for a ManyToMany mapping or for a OneToOne or bidirectional ManyToOne/OneToMany mapping using a join table, the foreign key is in a join table.

- If the join is for an element collection, the foreign key is in a collection table.

--

So as you can see, it depends on the context.

In our training video, we are using @OneToMany uni-directional (course has one-to-many reviews).

As a result, the join column / foreign key column is in the target entity. In this case, the target entity is the Review class. So, you will find the join column "course_id" in the "review" table.