**@JoinColumn could be used on both sides of the relationship.** The question was about using @JoinColumn on the @OneToMany side (rare case). And the point here is in *physical information* location.

According to [documentation](http://docs.jboss.org/hibernate/annotations/3.5/reference/en/html_single/):

Since **many to one are** (almost) always the **owner side** of a bidirectional relationship in the JPA spec, the one to many association is annotated by @OneToMany(mappedBy=...)

@Entity

public class Troop {

@OneToMany(mappedBy="troop")

public Set<Soldier> getSoldiers() {

...

}

@Entity

public class Soldier {

@ManyToOne

@JoinColumn(name="troop\_fk")

public Troop getTroop() {

...

}

Troop has a bidirectional one to many relationship with Soldier through the troop property. You don't have to (must not) define any physical mapping in the mappedBy side.

To map a bidirectional one to many, with the **one-to-many side as the owning side**, you have to remove the mappedBy element and set the many to one @JoinColumn as insertable and updatable to false. This solution is not optimized and will produce some additional UPDATE statements.

@Entity

public class Troop {

@OneToMany

@JoinColumn(name="troop\_fk") //we need to duplicate the physical information

public Set<Soldier> getSoldiers() {

...

}

@Entity

public class Soldier {

@ManyToOne

@JoinColumn(name="troop\_fk", insertable=false, updatable=false)

public Troop getTroop() {

...

}