# Hibernate Tip: Avoid MultipleBagFetchException

## Question:

You explained that I should use a JOIN FETCH clause to initialize all associations of my entity that I will use in my use case. But as soon as I do that for more than 1 association, Hibernate throws a *MultipleBagFetchException*.

How can I JOIN FETCH multiple associations?

## Solution:

Hibernate throws a *MultipleBagFetchException* whenever you try to fetch multiple *Bag*s in a query.

> "org.hibernate.loader.MultipleBagFetchException: cannot simultaneously fetch multiple *Bag*s: [org.thoughts.on.java.model.Book.authors, org.thoughts.on.java.model.Book.reviews]"

As I explained in my post about the most efficient data type for to-many associations, a *Bag* is an unordered collection. Hibernate uses it if you model your association as a *List*.

```
@Entity
public class Book {

    // Don't do this!

    @ManyToMany
    @JoinTable(
        name="BookAuthor",
        joinColumns={@JoinColumn(name="bookId",
            referencedColumnName="id")},
        inverseJoinColumns={@JoinColumn(name="authorId",
            referencedColumnName="id")})
    private List authors = new ArrayList();

    @OneToMany(mappedBy = "book")
    private List reviews = new ArrayList();

    ...

}
```

You can avoid the *MultipleBagFetchException* by using a *Set* instead of a *List*.

```
@Entity
public class Book {

    @ManyToMany
    @JoinTable(
          name="BookAuthor",
          joinColumns={@JoinColumn(name="bookId",
                      referencedColumnName="id")},
          inverseJoinColumns={@JoinColumn(name="authorId",
                      referencedColumnName="id")})
    private Set authors = new HashSet();

    @OneToMany(mappedBy = "book")
    private Set reviews = new HashSet();

    ...

}
```

In contrast to the previously used *Bag*s, Hibernate fetches multiple *Set*s without throwing an exception.

```
Book b = em.createQuery("SELECT b "
                + "FROM Book b "
                    + "JOIN FETCH b.authors a "
                    + "JOIN FETCH b.reviews r "
                + "WHERE b.id = 1",
                Book.class).getSingleResult();
```

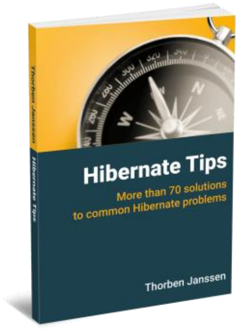
## Learn more

I explained the differences between a *Set* and a *Bag* in more details in How to Choose the Most Efficient Data Type for To-Many Associations – Bag vs. List vs. Set.

And if you want to learn more about Hibernate associations and how you can handle them efficiently, you should read the following posts:

- Ultimate Guide – Association Mappings with JPA and Hibernate
- How to map an association as a java.util.Map
- Best Practices for Many-To-One and One-To-Many Association Mappings

## Hibernate Tips Book



Get more recipes like this one in my book Hibernate Tips: More than 70 solutions to common Hibernate problems.

It gives you more than 70 ready-to-use recipes for topics like basic and advanced mappings, logging, Java 8 support, caching and statically and dynamically defined queries.