In Spring Data JPA**, "batch fetching"** is a mechanism used to optimize the fetching of collections or related entities**. It can work with both lazy and eager loading, but it is more commonly used with lazy loading to improve performance.**

**Lazy Loading with Batch Fetching**

When entities are loaded lazily, associated collections or related entities are not fetched from the database until they are accessed for the first time. Batch fetching can optimize this process by reducing the number of database queries. Instead of fetching each related entity individually, batch fetching allows Hibernate to fetch multiple entities in a single query when they are accessed.

For example, consider the following entities:

**@Entity**

**public class Author {**

**@Id**

**private Long id;**

**@OneToMany(mappedBy = "author", fetch = FetchType.LAZY)**

**@BatchSize(size = 10)**

**private List<Book> books;**

**}**

**@Entity**

**public class Book {**

**@Id**

**private Long id;**

**@ManyToOne(fetch = FetchType.LAZY)**

**private Author author;**

**}**

In this example, when you access the **books** collection of an **Author**, Hibernate will fetch the books in batches of 10.

**Eager Loading with Batch Fetching**

When entities are loaded eagerly, associated collections or related entities are fetched from the database immediately with the main entity. Batch fetching can also be applied here to optimize the fetching process. However, since eager loading fetches everything at once, the benefit of batch fetching is less pronounced compared to lazy loading.

**Configuring Batch Fetching**

You can configure batch fetching using annotations or properties. Here's how you can do it with annotations:

**@OneToMany(mappedBy = "author", fetch = FetchType.LAZY)**

**@BatchSize(size = 10)**

**private List<Book> books;**

You can also configure it globally using properties in your **application.properties** file:

**spring.jpa.properties.hibernate.default\_batch\_fetch\_size=10**

**Conclusion**

Batch fetching works with both lazy and eager loading in Spring Data JPA. It is generally more beneficial with lazy loading, as it can significantly reduce the number of queries executed when accessing lazily loaded collections or associations. Eager loading already fetches related entities upfront, so the impact of batch fetching is less significant in that case.