Relation JPA

Les Types de Relations

@OneToMany / @ManyToOne

Cas d'usage : Un auteur a plusieurs livres, un livre a un seul auteur.

Configuration Unidirectionnelle (Rarement recommandée)

```
@Entity
public class Author {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @OneToMany
    @JoinColumn(name = "author_id") // Crée une FK dans Book
    private List<Book> books = new ArrayList<>();
}

@Entity
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    // Pas de référence à Author
}
```

Problème : Génère des UPDATE supplémentaires !

Configuration Bidirectionnelle (Recommandée)

```
@Entity
public class Author {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @OneToMany(
        mappedBy = "author",          // Référence au champ dans Book
        cascade = CascadeType.ALL,
        orphanRemoval = true,
        fetch = FetchType.LAZY         // Par défaut
    )
```

```
    private List<Book> books = new ArrayList<>();
    // Méthodes helper pour synchroniser les deux côtés
    public void addBook(Book book) {
        books.add(book);
        book.setAuthor(this);
    }
    public void removeBook(Book book) {
        books.remove(book);
        book.setAuthor(null);
    }
}

@Entity
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @ManyToOne(
        fetch = FetchType.LAZY,       // Recommandé
        optional = false              // NOT NULL en base
    )
    @JoinColumn(name = "author_id")   // Nom de la FK
    private Author author;
}
```

Règle d'or : Le côté @ManyToOne est TOUJOURS le propriétaire (owner) de la relation.