JUNIOR NGALEU

Tu les utilises avec Spring Boot
mais sais-tu vraiment la différence ?

JPA, SPRING DATA JPA, HIBERNATE

Comprendre simplement avec des exemples

JUNIOR NGALEU

1. JPA (JAVA PERSISTENCE API)

JPA définit comment mapper des objets Java à une base de données.

Elle introduit des concepts comme :

- @Entity → pour représenter une table
- @Id → clé primaire
- relations (@OneToMany, @ManyToOne)

```java
@Entity
public class User {

    @Id
    @GeneratedValue
    private Long id;

    private String name;
}
```

Avantages :

1. Code propre et structuré
2. Indépendant d’une implémentation (portable)
3. Manipulation via objets plutôt que SQL
4. Facilite la maintenabilité

Java

JUNIOR NGALEU

2. SPRING DATA JPA

Spring Data JPA n’est : pas un ORM, pas une implémentation JPA, pas un moteur SQL

C’est un outil Spring qui te fait gagner du temps :

- crée les méthodes findByName(), findByEmail(), etc.
- évite d’écrire l’EntityManager
- génère automatiquement les repositories
- fournit JpaRepository

```
// Repository (Spring Data JPA)
public interface UserRepository extends JpaRepository<User, Long> {}

// Service
User user = userRepository.findById(1L).orElse(null);
```

Avantages :

1. Très peu de code à écrire
2. Requêtes automatiques (findByName(), findByEmail(), findAll(), findById(id), etc.)
3. CRUD, pagination et tri déjà prêts
4. Intégration parfaite avec Spring Boot

Java

3. HIBERNATE

Hibernate : c'est un ORM complet

- celui qui exécute réellement les requêtes SQL
- celui qui gère le cache, les sessions, le flush
- celui qui convertit les objets en SQL

```
SELECT * FROM user WHERE id = 1;
```

Avantages :

1. Génère automatiquement le SQL
2. Gère le mapping objet ↔ base de données
3. Optimisations puissantes (cache, lazy loading)
4. Supporte des relations complexes

Bonne Dégustation

JUNIOR NGALEU