



Amina Dahmouni

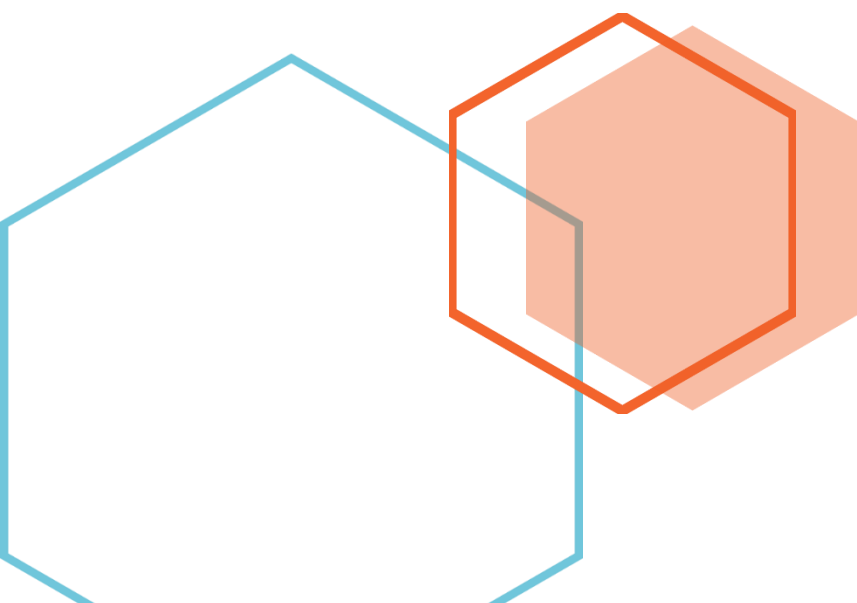
Classe : II-BDCC2

# [Compte Rendu]

---

Spring Boot - ORM avec Spring Data JPA  
Patient

Architecture JEE et Middlewares  
L'Ecole Normale Supérieure de l'Enseignement  
Technique de Mohammedia (ENSET)



Travailler dans les deux univers que sont l'orienté objet et la base de données relationnelle peut être lourd et consommateur en temps dans le monde de l'entreprise d'aujourd'hui.

L'utilisation pour la persistance d'un mapping O/R permet de proposer un niveau d'abstraction plus élevé que la simple utilisation de JDBC : ce mapping permet d'assurer la transformation d'objets vers la base de données et vice versa que cela soit pour des lectures ou des mises à jour (création, modification ou suppression).

L'API Java Persistence repose sur des entités qui sont de simples POJOs annotés et sur un gestionnaire de ces entités (EntityManager) qui propose des fonctionnalités pour les manipuler (ajout, modification suppression, recherche). Ce gestionnaire est responsable de la gestion de l'état des entités et de leur persistance dans la base de données.

**JPA** est une API qui possède plusieurs implémentations parmi lesquelles Hibernate, TopLink..

**Spring Data** est un module de spring qui permet d'offrir une implémentation générique des opérations de gestion des données.

**Hibernate** est un outil de mapping objet/relationnel pour le monde Java. Le terme mapping objet/relationnel (ORM) décrit la technique consistant à faire le lien entre la représentation objet des données et sa représentation relationnelle basée sur un schéma SQL.

## Classe Patient :

```
package ma.enset.jpap.entities;
import ...
@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class Patient {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Column(length = 50)
    private String nom;
    @Temporal(TemporalType.DATE)
    private Date dateNaissance;
    private boolean malade;
    private int score;
}
```

## L'interface PatientRepository :

```
package ma.enset.jpap.repositories;
import ...
public interface PatientRepository extends JpaRepository<Patient,Long> {
    //on définit la méthode sans l'implémenter cela va être fait par spring

    //Chercher la liste des patient selon l'attribut malade
    public List<Patient> findByMalade(boolean m);

    //Retourne une page
    public Page<Patient> findByMalade(boolean m, Pageable pageable);

    List<Patient> findByMaladeAndScoreLessThan(boolean m,int score);
    List<Patient> findByMaladeIsTrueAndScoreLessThan(boolean m,int score);
    List<Patient> findByDateNaissanceBetweenAndMaladeIsTrueOrNomLike(Date d1, Date d2,String mc);

    @Query("select p from Patient p where p.nom like :x and p.score<:y")
    List<Patient> chercherPatients(@Param("x") String nom,@Param("y") int scoreMin);

    @Query("select p from Patient p where p.dateNaissance between :x and :y or p.nom like :z")
    List<Patient> chercherPatients1(@Param("x") Date d1,@Param("y") Date d2,@Param("z") String nom);
}
```























































## Le fichier application.properties :

```
#spring.datasource.url=jdbc:h2:mem:patient-db
#pour avoir une interface web de base de données
#spring.h2.console.enabled=true
# on lui demande d'aller créer une base des données
spring.datasource.url=jdbc:mysql://localhost:3306/DBP?createDatabaseIfNotExist=true
spring.datasource.username=root
spring.datasource.password=
#Avec update chaque fois on démarre il insère mais h2 est une memory il perdre les données
spring.jpa.hibernate.ddl-auto=update
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MariaDBDialect
server.port=8082
#pour afficher les requêtes SQL
spring.jpa.show-sql=true
```

## Application Spring Boot :

```
package ma.enset.jpaap;
import ...
@SpringBootApplication
public class JpaApApplication implements CommandLineRunner {
    /** on a crée l'interface mais on a pas crée l'implémentation
     * car spring dispose déjà d'une implémentation générique
     * et il l'injecte
     */
    @Autowired
    private PatientRepository patientRepository;
    public static void main(String[] args) {
        SpringApplication.run(JpaApApplication.class, args);
    }
    @Override
    //Faire des tests au démarrage de l'application
    public void run(String... args) throws Exception {
        for (int i = 0; i < 100; i++) {
            patientRepository.save(new Patient( id: null, nom: "Amina", new Date(),
                Math.random() > 0.5 ? true : false, (int)(Math.random() * 1000)));
        }
    }
}
```

#	Nom	Type	Interclassement	Attributs	Null	valeur par défaut	Commentaires	Extra
<input type="checkbox"/> 1	id 🗝️	bigint(20)			Non	Aucun(e)		AUTO_INCREMENT
<input type="checkbox"/> 2	date_naissance	date			Oui	NULL		
<input type="checkbox"/> 3	malade	bit(1)			Non	Aucun(e)		
<input type="checkbox"/> 4	nom	varchar(50)	utf8mb4_general_ci		Oui	NULL		
<input type="checkbox"/> 5	score	int(11)			Non	Aucun(e)		

←T→				id	date_naissance	malade	nom	score
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	1	2022-03-13	0	Amina	629
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	2	2022-03-13	1	Amina	862
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	3	2022-03-13	0	Amina	151
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	4	2022-03-13	0	Amina	715
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	5	2022-03-13	1	Amina	163
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	6	2022-03-13	0	Amina	903
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	7	2022-03-13	1	Amina	220
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	8	2022-03-13	0	Amina	628
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	9	2022-03-13	1	Amina	165
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	10	2022-03-13	1	Amina	522
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	11	2022-03-13	0	Amina	487
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	12	2022-03-13	0	Amina	447
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	13	2022-03-13	1	Amina	227
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	14	2022-03-13	0	Amina	790
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	15	2022-03-13	1	Amina	360
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	16	2022-03-13	1	Amina	834
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	17	2022-03-13	0	Amina	586
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	18	2022-03-13	1	Amina	341

### La pagination➔

```
Page<Patient> patients = patientRepository.findAll(PageRequest.of( page: 0, size: 5));
System.out.println("Total des pages: "+patients.getTotalPages());
System.out.println("Total des éléments : "+patients.getTotalElements());
System.out.println("Numéro Page : "+patients.getNumber());
```

```
Total des pages: 20
Total des éléments : 100
Numéro Page : 0
```

### Afficher le contenu du première page➡

```
List<Patient> content = patients.getContent();
content.forEach(p -> {
    System.out.println("#####");
    System.out.println(p.getId());
    System.out.println(p.getNom());
    System.out.println(p.getDateNaissance());
    System.out.println(p.getScore());
    System.out.println(p.isMalade());
});
System.out.println("*****");
```

```
#####
1
Amina
2022-03-13
629
false
#####
2
Amina
2022-03-13
862
true
#####
3
Amina
2022-03-13
151
false
#####
```

```
#####

4
Amina
2022-03-13
715
false
#####

5
Amina
2022-03-13
163
true
*****
```

Liste des patients qui sont malades➡

```
//Liste des patients qui sont malades
Page<Patient> byMalade = patientRepository.findByMalade(m: true, PageRequest.of(page: 0, size: 4));
byMalade.forEach(p -> {
    System.out.println("#####");
    System.out.println(p.getId());
    System.out.println(p.getNom());
    System.out.println(p.getDateNaissance());
    System.out.println(p.getScore());
    System.out.println(p.isMalade());
});
System.out.println("*****");
```

```
#####  
2  
Amina  
2022-03-13  
862  
true  
#####  
5  
Amina  
2022-03-13  
163  
true  
#####  
7  
Amina  
2022-03-13  
220  
true  
#####
```

```
#####  
9  
Amina  
2022-03-13  
165  
true  
*****
```



## Chercher par Id ➡

```
//s'il existe il va l'afficher sinon il va afficher un message d'erreur
Patient patient1=patientRepository.findById(1L).orElseThrow(()->new RuntimeException("patient not found"));
//s'il existe il va l'afficher sinon il va afficher une exception
Patient patient2=patientRepository.findById(1L).get();
//s'il existe il va l'afficher sinon il va afficher null
Patient patient3=patientRepository.findById(1L).orElse( other: null);
if (patient3!=null){
    System.out.println(patient3.getNom());
    System.out.println(patient3.isMalade());
}
```