Compte rendu du TP3 :

La classe patient qui va être transformer en une table dans la base de données

```java
@Data // generer getters et setters ,lombok qui fait ça
@NoArgsConstructor //constructeur sans parametres
@AllArgsConstructor // constructeur avec parametres
@Entity //pour creer une entité JPA
public class patient {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Column(name = "NOM",length = 50)// dans la base de données ,la column correspondante à cette attribut
    private String nom;
    @Temporal(TemporalType.DATE)// dans la base de données garder que la date(dd/mm/yyyy)
    private Date dateNaissance;
    private Boolean malade;
    private int score;

}
```

L'interface patientRepository qui contient les méthodes de récupération, de suppression des enregistrements de la table

```java
public interface patientRepository extends JpaRepository<patient,Long> {
    List<patient> findByMalade(Boolean m);
    //retourner les la liste des patients where malade=true dans la page x
    Page<patient> findByMalade(Boolean m,Pageable pageable);

    //retourner les la liste des patients where malade=true and score < valeur x
    List<patient> findByMaladeAndScoreLessThan(Boolean m,int score);

    //retourner les la liste des patients where DateNaissance between d1 and d2
    List<patient> findByDateNaissanceBetween(Date d1,Date d2);

    @Query("select p from patient p where p.dateNaissance between  :x and :y")
    List<patient> chercherPatients(@Param("x") Date d1, @Param("y")Date d2);
}
```

Le fichier application.properties qui contient les informations pour la connexion à la base de données

```
spring.datasource.url=jdbc:h2:mem:patient-db
spring.h2.console.enabled=true
server.port=8082
spring.jpa.show-sql=true
```

La classe Main

```java
@SpringBootApplication
public class Tp3Application implements CommandLineRunner {

    @Autowired
    private patientRepository patientRepository;

    public static void main(String[] args) {

        SpringApplication.run(Tp3Application.class, args);

    }

    @Override
    public void run(String... args) throws Exception {

        //la methode save() est héritée de JPA
        for(int i=0;i<50;i++){
            patientRepository.save(new patient( id: null, nom: "salma",new Date(),Math.random()>1?true:false,(int)Math.random()));

        }
```

La méthode qui permet de récupérer tous les patients

```java
List<patient> patients=patientRepository.findAll();
patients.forEach(patient ->{
    System.out.println("----------------------");
    System.out.println(patient.getNom());
    System.out.println(patient.getMalade());
});
```

La méthode qui permet de récupérer les cinq premiers patients

```java
//retourner les 5 premiers patients
Page<patient> patientsPage=patientRepository.findAll(PageRequest.of( page: 0, size: 5));
patientsPage.forEach(patient ->{
    System.out.println("----------------------");
    System.out.println(patient.getNom());
    System.out.println(patient.getMalade());
});
```

Pour afficher le nombre de page et le nombre d'éléments de la table

```java
System.out.println("nombre de pages:"+patientsPage.getTotalPages());
System.out.println("nombre d'éléments:"+patientsPage.getTotalElements());
System.out.println("numéro de page courante:"+patientsPage.getNumber());
```

La méthode qui permet de récupérer les patients malades

```
List<patient> patientsByMalade=patientRepository.findByMalade( m: true);
patientsByMalade.forEach(patient ->{
    System.out.println("----------------------");
    System.out.println(patient.getNom());
    System.out.println(patient.getMalade());
});

Page<patient> patientsByMaladePage=patientRepository.findByMalade( m: true,PageRequest.of( page: 1, size: 5));
patientsByMaladePage.forEach(patient ->{
    System.out.println("----------------------");
    System.out.println(patient.getNom());
    System.out.println(patient.getMalade());
});
```

Affichage

```
salma
false
Hibernate: select patient0_.id as id1_0_, patient0_.date_naissance as date_nai2_0_, patient0_.malade as malade3_0_, patient0_.nom as nom4_0_
Hibernate: select count(patient0_.id) as col_0_0_ from patient patient0_
----------------------
salma
false
----------------------
salma
false
----------------------
salma
false
----------------------
salma
false
----------------------
salma
false
----------------------
salma
false
nombre de pages:10
nombre d'éléments:50
numéro de page courante:0
```

```
SELECT * FROM PATIENT
```

SELECT * FROM PATIENT;

ID	DATE_NAISSANCE	MALADE	NOM	SCORE
1	2022-03-15	FALSE	salma	0
2	2022-03-15	FALSE	salma	0
3	2022-03-15	FALSE	salma	0
4	2022-03-15	FALSE	salma	0
5	2022-03-15	FALSE	salma	0
6	2022-03-15	FALSE	salma	0
7	2022-03-15	FALSE	salma	0
8	2022-03-15	FALSE	salma	0
9	2022-03-15	FALSE	salma	0
10	2022-03-15	FALSE	salma	0
11	2022-03-15	FALSE	salma	0
12	2022-03-15	FALSE	salma	0
13	2022-03-15	FALSE	salma	0
14	2022-03-15	FALSE	salma	0
15	2022-03-15	FALSE	salma	0
16	2022-03-15	FALSE	salma	0
17	2022-03-15	FALSE	salma	0