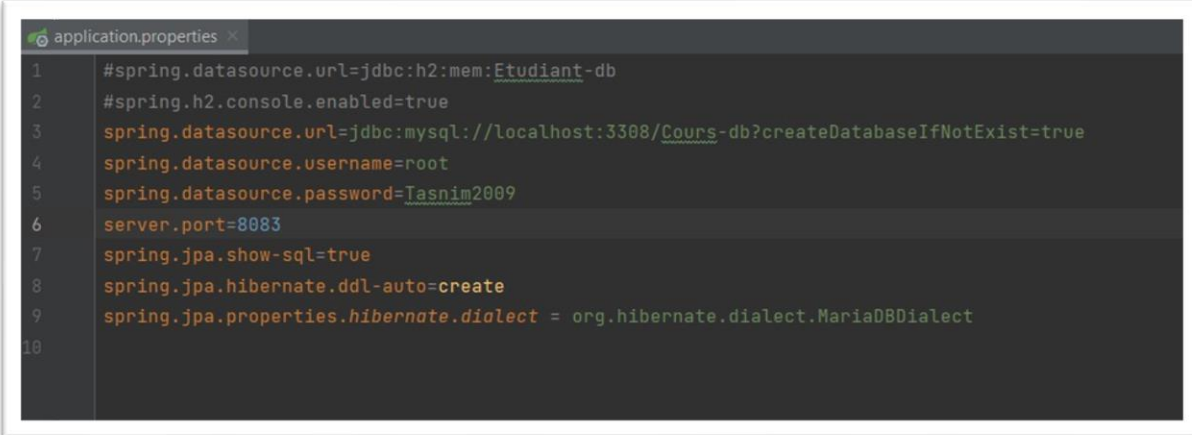


Compte Rendu : Spring Data JPA

Réalisé par : MOUATASSIM BILLAH ILYASS

```
Etudiant.java x Tp1Application.java x pom.xml (TP1) x EtudiantRepository.java x application.properties x
25 <dependencies>
26 <dependency>
27 <groupId>org.springframework.boot</groupId>
28 <artifactId>spring-boot-starter-data-jpa</artifactId>
29 </dependency>
30 <dependency>
31 <groupId>org.springframework.boot</groupId>
32 <artifactId>spring-boot-starter-web</artifactId>
33 </dependency>
34 <dependency>
35 <groupId>org.springframework.boot</groupId>
36 <artifactId>spring-boot-devtools</artifactId>
37 <scope>runtime</scope>
38 <optional>true</optional>
39 </dependency>
40 <dependency>
41 <groupId>com.h2database</groupId>
42 <artifactId>h2</artifactId>
43 <scope>runtime</scope>
44 </dependency>
45 <dependency>
46 <groupId>org.projectlombok</groupId>
47 <artifactId>lombok</artifactId>
48 <optional>true</optional>
49 </dependency>
50 <dependency>
51 <groupId>org.springframework.boot</groupId>
52 <artifactId>spring-boot-starter-test</artifactId>
53 <scope>test</scope>
```

```
<dependency>
  <groupId>mysql</groupId>
  <artifactId>mysql-connector-java</artifactId>
  <version>8.0.29</version>
</dependency>
```

A screenshot of a code editor window titled 'application.properties'. The editor has a dark background with light-colored text. The code is a Spring application configuration file. It includes comments for H2 database settings and actual settings for a MySQL database. The settings include the database URL, username, password, server port, and JPA properties for SQL logging and dialect. Line numbers 1 through 10 are visible on the left side of the editor.

```
1 #spring.datasource.url=jdbc:h2:mem:Etudiant-db
2 #spring.h2.console.enabled=true
3 spring.datasource.url=jdbc:mysql://localhost:3308/Cours-db?createDatabaseIfNotExist=true
4 spring.datasource.username=root
5 spring.datasource.password=Tasnim2009
6 server.port=8083
7 spring.jpa.show-sql=true
8 spring.jpa.hibernate.ddl-auto=create
9 spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MariaDBDialect
10
```

```

Cours.java x
1 package ma.emsi.tpspringdata.entities;
2
3 import jakarta.persistence.*;
4 import lombok.AllArgsConstructor;
5 import lombok.Data;
6 import lombok.NoArgsConstructor;
7
8 import java.util.Collection;
9
10 10 usages
11 @Data @AllArgsConstructor @NoArgsConstructor
12 @Entity
13 public class Cours {
14     no usages
15     @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
16     private int id;
17     no usages
18     @Column(length = 25, nullable = false, unique = false)
19     private String title;
20     no usages
21     private String description;
22     no usages
23     private int timing;
24     no usages
25     @OneToOne
26     private Professeur professeur;
27     no usages
28     @OneToMany(mappedBy = "cours", fetch = FetchType.EAGER)
29     private Collection<Seance> seance;
30

```

```
Etudiant.java x
1 package ma.emsi.tpspringdata.entities;
2
3 import ...
  7 usages
13 @Entity @Table(name = "EMSI_STUDENTS")
14 @Data @NoArgsConstructor @AllArgsConstructor @ToString
15 public class Etudiant {
  no usages
16     @Id @GeneratedValue(strategy= GenerationType.IDENTITY)
17     private Integer id;
  no usages
18     @Column(name = "REGISTRATION_N", unique = true)
19     private String registrationNumber;
  no usages
20     @Column(name = "Name", length = 30, nullable = false)
21     private String fullName;
  no usages
22     @Temporal(TemporalType.DATE)
23     private Date birthay;
  no usages
24     private Boolean stillActive;
  no usages
25     @Temporal(TemporalType.TIMESTAMP) @CreationTimestamp
26     private Date lastConnection;
  no usages
27     @ManyToMany(mappedBy = "etudiants", fetch = FetchType.EAGER)
28     private Collection<Seance> seances=new ArrayList<>();
29 }
30
```

```

Professeur.java x
5  import lombok.Data;
6  import lombok.NoArgsConstructor;
7  import org.hibernate.annotations.CreationTimestamp;
8  import org.springframework.boot.autoconfigure.web.WebProperties;
9
10 import java.util.Date;
11
12 9 usages
13 @Data @AllArgsConstructor @NoArgsConstructor
14 @Entity
15 public class Professeur {
16     no usages
17     @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
18     private int id;
19     no usages
20     @Column(length = 30, nullable = false)
21     private String fullName;
22     no usages
23     @Temporal(TemporalType.DATE) @CreationTimestamp
24     private Date assignementDate;
25     no usages
26     @OneToOne(mappedBy = "professeur")
27     private Cours cours;
28 }

```

```
Seance.java x
1 package ma.emsi.tpspringdata.entities;
2
3 import ...
4
5 8 usages
6
7 @Entity
8 @Data @AllArgsConstructor @NoArgsConstructor
9
10 public class Seance {
11     no usages
12     @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
13     private int id;
14     no usages
15     @Temporal(TemporalType.DATE)
16     private Date date;
17     no usages
18     @Temporal(TemporalType.TIME)
19     private Date start_time;
20     no usages
21     @Temporal(TemporalType.TIME)
22     private Date end_time;
23     no usages
24     @ManyToOne
25     private Cours cours;
26     no usages
27     @ManyToMany(fetch = FetchType.EAGER)
28     @ToString.Exclude
29     @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
30     private Collection<Etudiant> etudiants = new ArrayList<>();
31 }
```

```

1 package ma.emsi.tpspringdata.repositories;
2
3 import ma.emsi.tpspringdata.entities.Cours;
4 import org.springframework.data.jpa.repository.JpaRepository;
5
6 2 pages
7 public interface CoursRepository extends JpaRepository<Cours,Integer> {
8 }

```

```

EtudiantRepository.java
1 package ma.emsi.tpspringdata.repositories;
2
3 import ma.emsi.tpspringdata.entities.Etudiant;
4 import org.springframework.data.jpa.repository.JpaRepository;
5
6 2 pages
7 public interface EtudiantRepository extends JpaRepository<Etudiant,Integer> {
8 }
9

```

```

ProfesseurRepository.java
1 package ma.emsi.tpspringdata.repositories;
2
3
4 import ma.emsi.tpspringdata.entities.Professeur;
5 import org.springframework.data.jpa.repository.JpaRepository;
6
7 2 pages
8 public interface ProfesseurRepository extends JpaRepository<Professeur,Integer> {
9 }

```



```
SeanceRepository.java X
1 package ma.emsi.tpspringdata.repositories;
2
3 import ma.emsi.tpspringdata.entities.Seance;
4 import org.springframework.data.jpa.repository.JpaRepository;
5
6 public interface SeanceRepository extends JpaRepository<Seance,Integer> {
7 }
8
```

```
TpSpringDataApplication.java X
1 package ma.emsi.tpspringdata;
2
3 import ...
4
5 @SpringBootApplication
6 public class TpSpringDataApplication {
7
8     public static void main(String[] args) {
9
10         SpringApplication.run(TpSpringDataApplication.class, args);
11     }
12
13     @Bean
14     CommandLineRunner start(CoursRepository coursRepository, ProfesseurRepository professeurRepository, SeanceRepository seanceRep
15     return args -> {
16         Stream.of(...values: "Akram Lkihal", "Ahmed Zidany", "Yasser Hmimou")
17             .forEach(fullName->{
18                 Etudiant etudiant= new Etudiant();
19                 etudiant.setRegistrationNumber(fullName );
20                 etudiant.setFullName(fullName);
21                 etudiant.setBirthay(new Date());
22                 etudiant.setLastConnection(new Date());
23                 etudiant.setStillActive(Math.random()>0.5?true:false);
24                 etudiant.setSeances(null);
25                 etudiantRepository.save(etudiant);
26             });
27     }
28 }
```

```

Stream.of("Yassine Barami", "Ilyass Mouatassim", "Abdelmoughit Aitchihab")
    .forEach(fullName->{
        Professeur professeur= new Professeur();
        professeur.setFullName(fullName);
        professeur.setAssignementDate(new Date());
        professeur.setCours(null);
        professeurRepository.save(professeur);

    });

Stream.of("1", "2", "3")
    .forEach(Seance->{
        Seance seance= new Seance();
        seance.setDate(new Date());
        seance.setStart_time(new Date());
        seance.setEnd_time(new Date());
        seance.setCours(null);
        seance.setEtudiants(null);
        seanceRepository.save(seance);
    });

```

```

Stream.of( ...values: "math", "physique", "informatique")
    .forEach(cour->{
        Cours cours = new Cours();
        cours.setTitle(cour);
        cours.setDescription(Math.random()>0.5?"intéressant":"long");
        cours.setTiming(5);
        cours.setProfesseur(null);
        cours.setSeance(null);
        coursRepository.save(cours);
    });

Cours cours1= coursRepository.findById(1).orElse( other: null);
Professeur professeur1= professeurRepository.findById(1).orElse( other: null);
cours1.setProfesseur(professeur1);
coursRepository.save(cours1);

Cours cours2= coursRepository.findById(2).orElse( other: null);
Professeur professeur3= professeurRepository.findById(3).orElse( other: null);
cours2.setProfesseur(professeur3);
coursRepository.save(cours2);

Cours cours3= coursRepository.findById(3).orElse( other: null);
Professeur professeur2= professeurRepository.findById(2).orElse( other: null);
cours3.setProfesseur(professeur2);
coursRepository.save(cours3);

```

```

Seance seance1= seanceRepository.findById(1).orElse( other: null);
seance1.setCours(cours1);
seanceRepository.save(seance1);

Etudiant etudiant1= etudiantRepository.findById(1).orElse( other: null);
if(etudiant1.getSeances()!=null) {
    etudiant1.getSeances().add(seance1);
    seance1.getEtudiants().add(etudiant1);
    etudiantRepository.save(etudiant1);
    seanceRepository.save(seance1);
}

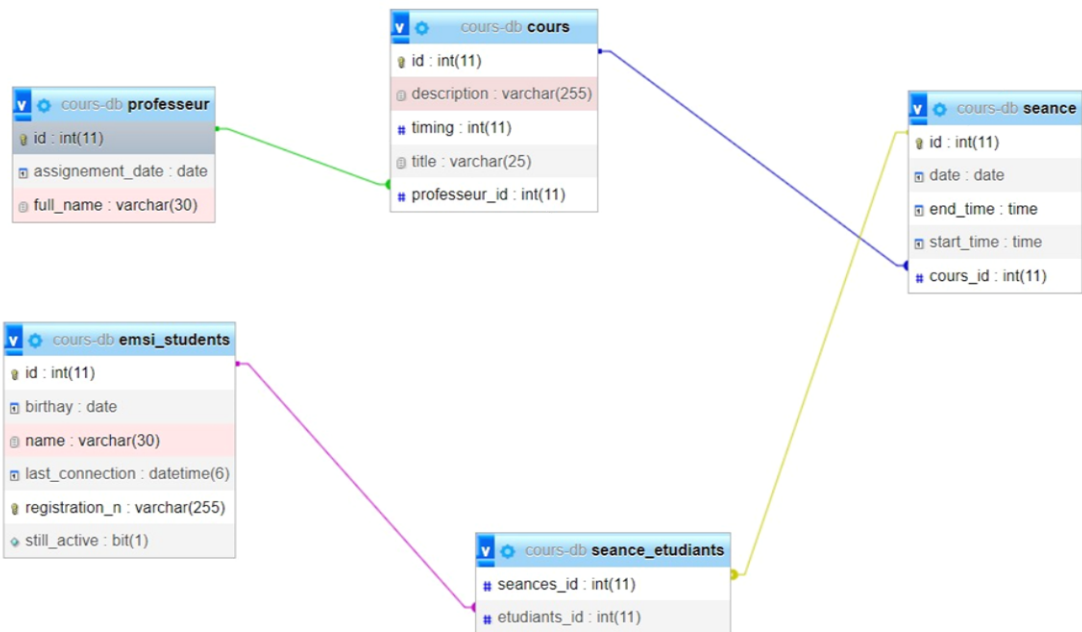
etudiant1.getSeances().forEach(s->{
    System.out.println("SEANCE=>" + s.toString());
});

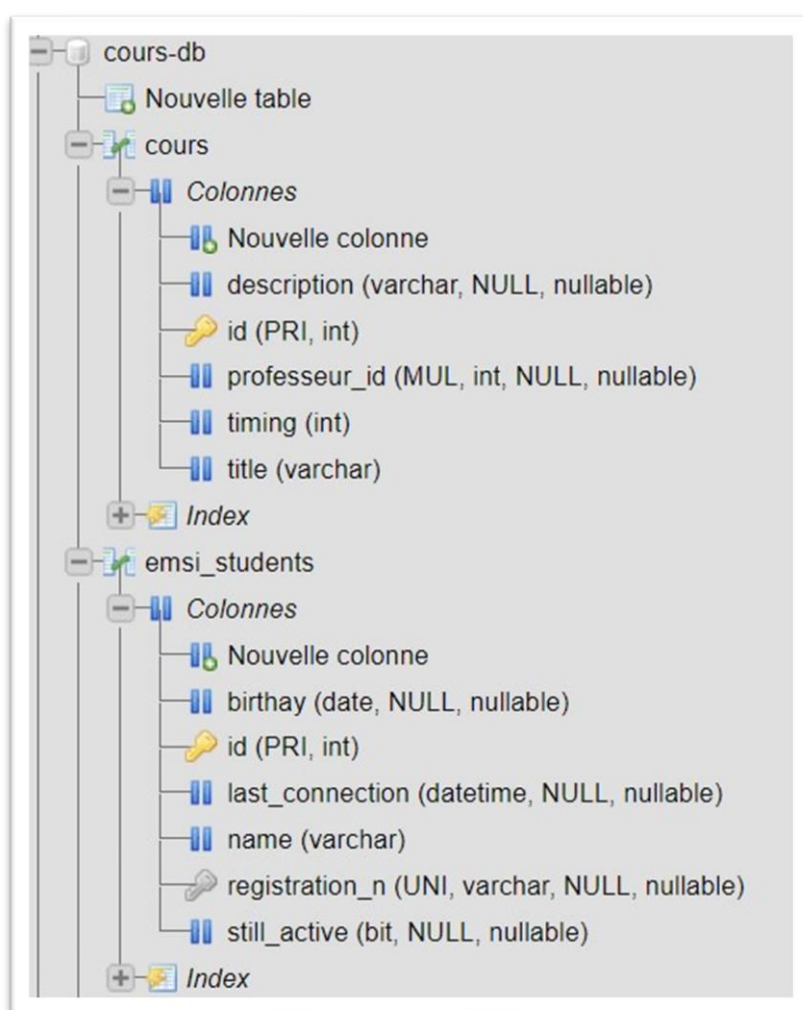
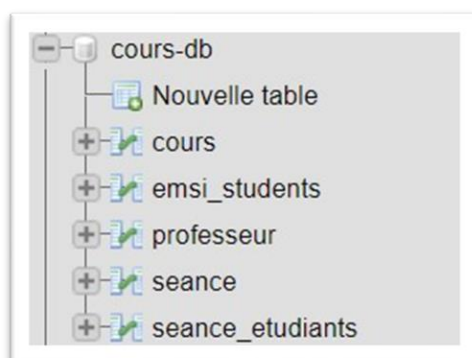
```

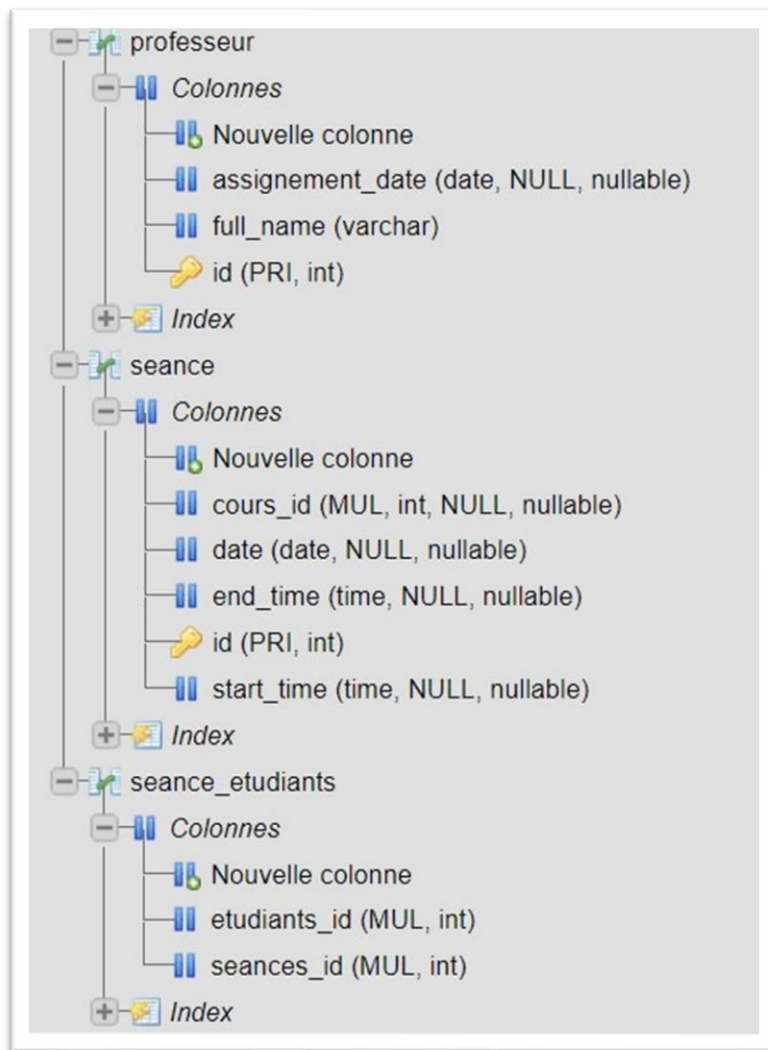
```

Hibernate: alter table cours drop foreign key FK3vnhktsmivcvmzqjsorbu63u
Hibernate: alter table seance drop foreign key FKrrc1k4hpxsmh2hau4gl5mjgod
Hibernate: alter table seance_etudiants drop foreign key FKl9vlgstleaoui4y809lkde3t
Hibernate: alter table seance_etudiants drop foreign key FKgs3nftvelfvhrx4grl040bbi
Hibernate: drop table if exists cours
Hibernate: drop table if exists emsi_students
Hibernate: drop table if exists professeur
Hibernate: drop table if exists seance
Hibernate: drop table if exists seance_etudiants
Hibernate: create table cours (id integer not null auto_increment, description varchar(255), timing integer not null, title varchar(25) not null, professeur_id integer, primary
Hibernate: create table emsi_students (id integer not null auto_increment, birthay date, name varchar(30) not null, last_connection datetime(6), registration_n varchar(255), sti
Hibernate: create table professeur (id integer not null auto_increment, assignement_date date, full_name varchar(30) not null, primary key (id)) engine=InnoDB
Hibernate: create table seance (id integer not null auto_increment, date date, end_time time, start_time time, cours_id integer, primary key (id)) engine=InnoDB
Hibernate: create table seance_etudiants (seances_id integer not null, etudiants_id integer not null) engine=InnoDB
Hibernate: alter table emsi_students add constraint UK_a3vjtvlbahy45i3mbr04y7hsu unique (registration_n)

```
























				id	description	timing	title	professeur_id
<input type="checkbox"/>	Éditer	Copier	Supprimer	1	intéressant	5	math	1
<input type="checkbox"/>	Éditer	Copier	Supprimer	2	intéressant	5	physique	3
<input type="checkbox"/>	Éditer	Copier	Supprimer	3	long	5	informatique	2

<div><div></div><div></div><div></div></div>			id	birthay	name	last_connection	registration_n	still_active	
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	1	2023-04-16	Akram Lkihal	2023-04-16 12:36:41.000000	Akram Lkihal	1
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	2	2023-04-16	Ahmed Zidany	2023-04-16 12:36:41.000000	Ahmed Zidany	0
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	3	2023-04-16	Yasser Hmimou	2023-04-16 12:36:41.000000	Yasser Hmimou	0

<div><div>←T→</div><div></div></div>				id	assignement_date	full_name
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	1	2023-04-16	Yassine Barami
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	2	2023-04-16	Ilyass Mouatassim
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	3	2023-04-16	Abdelmoughit Aitchihab

<div><div>←</div><div>T</div><div>→</div></div>					id	date	end_time	start_time	cours_id
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer		1	2023-04-16	12:36:41	12:36:41	1
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer		2	2023-04-16	12:36:41	12:36:41	NULL
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer		3	2023-04-16	12:36:41	12:36:41	NULL

seances_id	etudiants_id
1	1

SEANCE=>Seance(id=1, date=2023-04-16, start_time=12:36:41, end_time=12:36:41, cours=Cours(id=1, title=math, description=intéressant, timing=5, professeur=Professeur(id=1, fullN

