Compte rendu du TP 4 : JPA

1. Les entités qu'on a utilisées :
   - La classe Patient :

```java
package com.example.TP4_JPA.entities;


import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import javax.persistence.*;
import java.util.Collection;
import java.util.Date;

@Entity
@Data @NoArgsConstructor @AllArgsConstructor
public class Patient {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    @Temporal(TemporalType.DATE)
    private Date datenaissance;
    private boolean malade;
    @OneToMany(mappedBy = "patient", fetch = FetchType.LAZY)
    private Collection<Rendez_vous> rendezVous;
}
```

   - la classe Médecin :

```java
package com.example.TP4_JPA.entities;

import com.fasterxml.jackson.annotation.JsonProperty;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import javax.persistence.*;
import java.util.Collection;

@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Medecin {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private String nom;
    private String email;
    private String specialite;
    @OneToMany(mappedBy = "medecin", fetch = FetchType.LAZY)
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private Collection<Rendez_vous> rendezVous;

}
```

- la classe Rendez-Vous:

```
package com.example.TP4_JPA.entities;

import com.fasterxml.jackson.annotation.JsonProperty;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import javax.persistence.*;
import java.util.Date;

@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Rendez_vous {
    @Id
    /* @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;*/
    private String id;
    private Date date;
    @Enumerated(EnumType.STRING)
    private StatusRDV status;
    @ManyToOne
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private Patient patient;
    @ManyToOne
    private Medecin medecin;
    @OneToOne(mappedBy = "rendezVous")
    private Consultation consultation;

}
```

- la classe Consultation:

```
package com.example.TP4_JPA.entities;

import com.fasterxml.jackson.annotation.JsonProperty;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

import javax.persistence.*;
import java.util.Date;

@Entity
@Data
@NoArgsConstructor
@AllArgsConstructor
public class Consultation {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    private Date dateConsultation;
    private String rapport;
    @OneToOne
    @JsonProperty(access = JsonProperty.Access.WRITE_ONLY)
    private Rendez_vous rendezVous;
}
```

- la classe StatusRDV:

```
package com.example.TP4_JPA.entities;

public enum StatusRDV {
    PENDING,
    CANCELED,
    DONE
}
```

2. Les repositories:
   - Pour Patient:

```
package com.example.TP4_JPA.repositories;

import com.example.TP4_JPA.entities.Patient;
import org.springframework.data.jpa.repository.JpaRepository;

public interface PatienRepository extends JpaRepository<Patient,Long>
{
    Patient findByNom(String name); // on suppose que c'est un nom
unique
}
```

   - Pour Medecin:

```
package com.example.TP4_JPA.repositories;

import com.example.TP4_JPA.entities.Medecin;
import com.example.TP4_JPA.entities.Patient;
import org.springframework.data.jpa.repository.JpaRepository;

public interface MedecinRepository extends JpaRepository<Medecin,
Long> {
    Medecin findByNom(String name); // on suppose que c'est un nom
unique
}
```

   - Pour Rendez-Vous:

```
package com.example.TP4_JPA.repositories;

import com.example.TP4_JPA.entities.Rendez_vous;
import org.springframework.data.jpa.repository.JpaRepository;

public interface RendezVousRepository extends
JpaRepository<Rendez_vous, Long> {
}
```

   - Pour Consultation:

```
package com.example.TP4_JPA.repositories;

import com.example.TP4_JPA.entities.Consultation;
import org.springframework.data.jpa.repository.JpaRepository;

public interface ConsultationRepository extends
```

```
JpaRepository<Consultation,Long> {
}
```

3. Les services:
   - Interface:

```
package com.example.TP4_JPA.service;

import com.example.TP4_JPA.entities.Consultation;
import com.example.TP4_JPA.entities.Medecin;
import com.example.TP4_JPA.entities.Patient;
import com.example.TP4_JPA.entities.Rendez_vous;

public interface IHospitalService {
    Patient savePatient(Patient patient);
    Medecin saveMedecin(Medecin medecin);
    Rendez_vous saveRDV(Rendez_vous rendezVous);
    Consultation saveConsultation(Consultation consultation);
}
```

   - Classe :

```
package com.example.TP4_JPA.service;

import com.example.TP4_JPA.entities.Consultation;
import com.example.TP4_JPA.entities.Medecin;
import com.example.TP4_JPA.entities.Patient;
import com.example.TP4_JPA.entities.Rendez_vous;
import com.example.TP4_JPA.repositories.ConsultationRepository;
import com.example.TP4_JPA.repositories.MedecinRepository;
import com.example.TP4_JPA.repositories.PatienRepository;
import com.example.TP4_JPA.repositories.RendezVousRepository;
import lombok.AllArgsConstructor;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import javax.transaction.Transactional;
import java.util.UUID;

@Service
@Transactional @AllArgsConstructor
public class HospitalServiceImpl implements IHospitalService{
    private PatienRepository patientRepository;
    private MedecinRepository medecinRepository;
    private RendezVousRepository rendezVousRepository;
    private ConsultationRepository consultationRepository;

    @Override
    public Patient savePatient(Patient patient) {
        return patientRepository.save(patient);
    }

    @Override
    public Medecin saveMedecin(Medecin medecin) {
        return medecinRepository.save(medecin);
    }

    @Override
    public Rendez_vous saveRDV(Rendez_vous rendezVous) {
        rendezVous.setId(UUID.randomUUID().toString());
```

```
        return rendezVousRepository.save(rendezVous);
    }

    @Override
    public Consultation saveConsultation(Consultation
consultation) {
        return consultationRepository.save(consultation);
    }
}
```

4. Service Web :

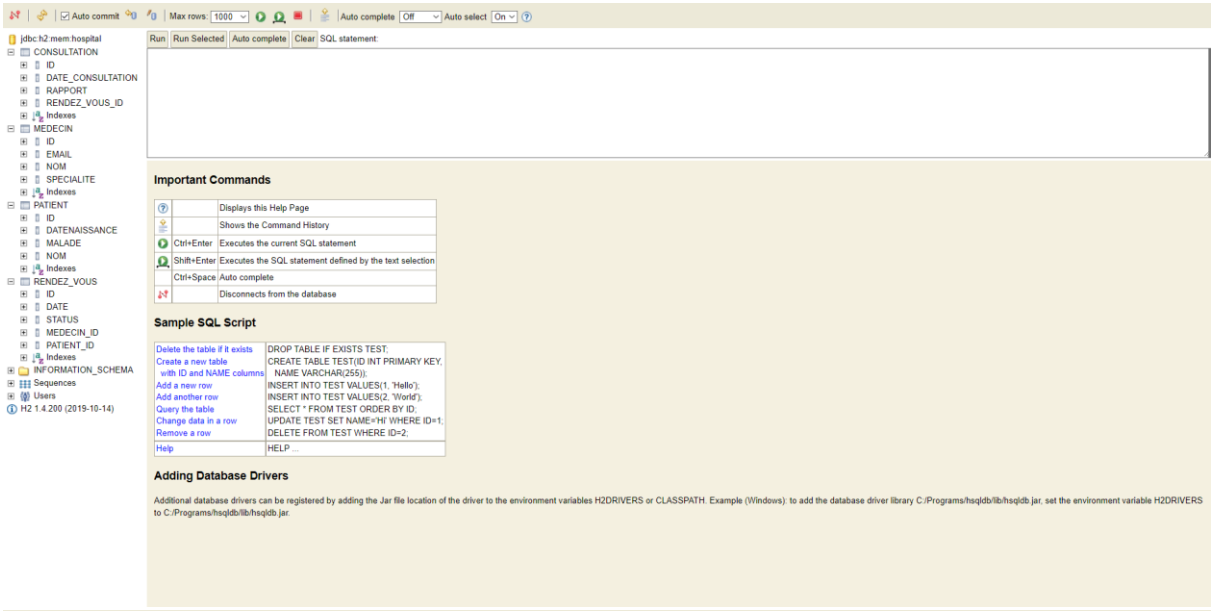
```
package com.example.TP4_JPA.web;

import com.example.TP4_JPA.entities.Patient;
import com.example.TP4_JPA.repositories.PatienRepository;
import com.sun.xml.bind.annotation.XmlIsSet;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

import java.util.List;

@RestController
public class PatientRestController {
    @Autowired
    private PatienRepository patienRepository;
    @GetMapping("/patient")
    public List<Patient> patientsList(){
        return patienRepository.findAll();
    }
}
```

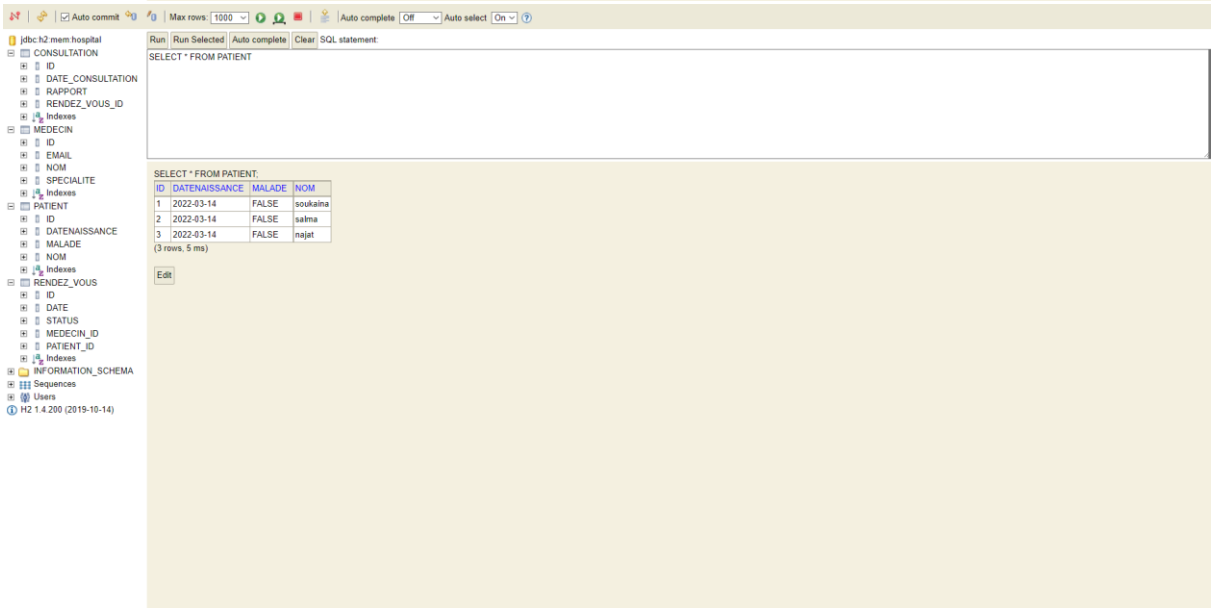
5. Affichage :
   - Le Main :

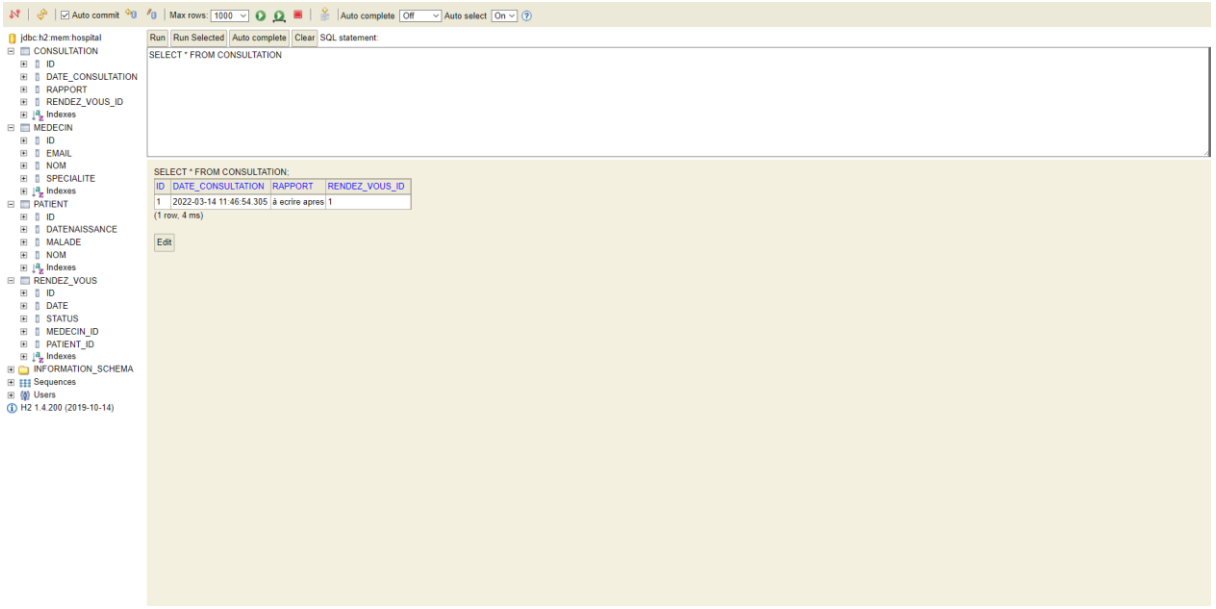
```
package com.example.TP4_JPA;

import com.example.TP4_JPA.entities.*;
import com.example.TP4_JPA.repositories.ConsultationRepository;
import com.example.TP4_JPA.repositories.MedecinRepository;
import com.example.TP4_JPA.repositories.PatienRepository;
import com.example.TP4_JPA.repositories.RendezVousRepository;
import com.example.TP4_JPA.service.IHospitalService;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

import java.util.Date;
import java.util.stream.Stream;

@SpringBootApplication
public class Tp4JpaApplication {

   public static void main(String[] args) {
      SpringApplication.run(Tp4JpaApplication.class, args);
   }
```

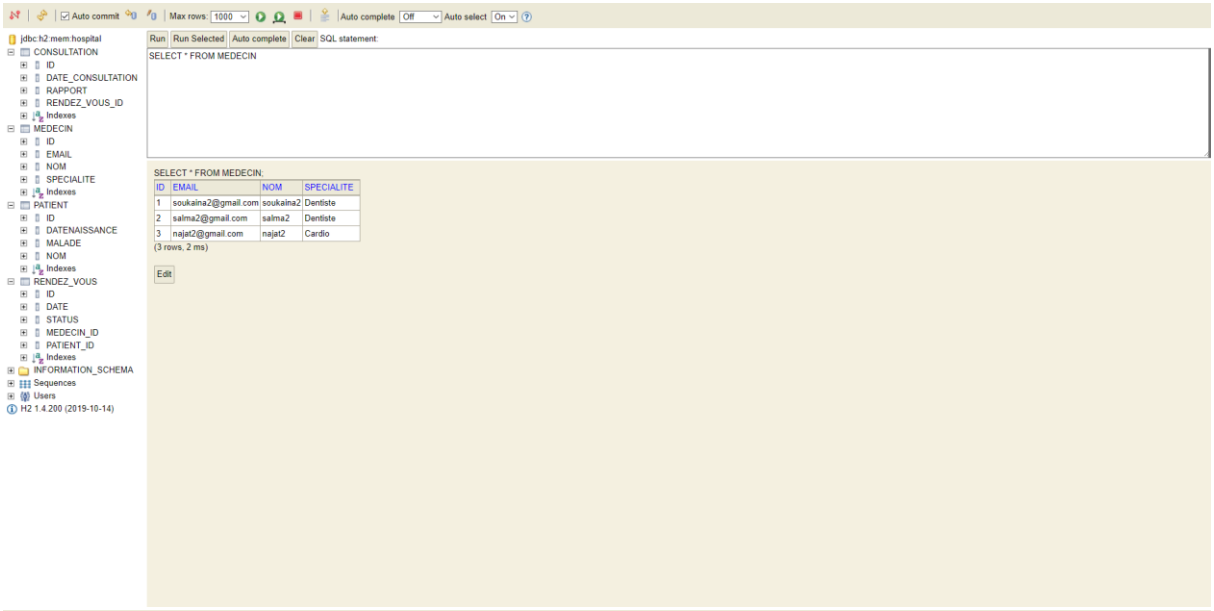
```
   @Bean // pour démarrer cette méthode au démarage
   CommandLineRunner start(IHospitalService hospitalService,
PatienRepository patienRepository, MedecinRepository
medecinRepository, RendezVousRepository rendezVousRepository){
      return args ->{
         Stream.of("soukaina","salma","najat").forEach(name-> {
                  hospitalService.savePatient(new Patient(null,
name, new Date(), false, null));
               });
         /*
         Stream.of("soukaina","salma","najat").forEach(name-> {
            Patient patient=new Patient();
            patient.setNom(name);
            patient.setDateNaissance(new Date());
            patient.setMalade(false);
            patientRepository.save(patient);
         });
          */

         Stream.of("soukaina2","salma2","najat2").forEach(name-> {
            Medecin medecin=new Medecin();
            medecin.setNom(name);
            medecin.setEmail(name+"@gmail.com");

medecin.setSpecialite(Math.random()>0.5?"Cardio":"Dentiste");
            hospitalService.saveMedecin(medecin);
         });

         Patient
patient=patienRepository.findById(1L).orElse(null);
         Patient patient1=patienRepository.findByNom("soukaina");

         Medecin medecin =
medecinRepository.findByNom("soukaina2");

         Rendez_vous rendezVous = new Rendez_vous();
         rendezVous.setDate(new Date());
         rendezVous.setStatus(StatusRDV.PENDING);
         rendezVous.setMedecin(medecin);
         rendezVous.setPatient(patient);
         hospitalService.saveRDV(rendezVous);
         Rendez_vous savedRDV =
hospitalService.saveRDV(rendezVous);
         System.out.println(savedRDV.getId());

         //Rendez_vous rendezVous2 =
rendezVousRepository.findById(1L).orElse(null);
         Rendez_vous rendezVous2 =
rendezVousRepository.findAll().get(0);
         Consultation consultation = new Consultation();
         consultation.setDateConsultation(new Date());
         consultation.setRendezVous(rendezVous2);
         consultation.setRapport("à ecrire apres");
         hospitalService.saveConsultation(consultation);
      };
   }

}
```

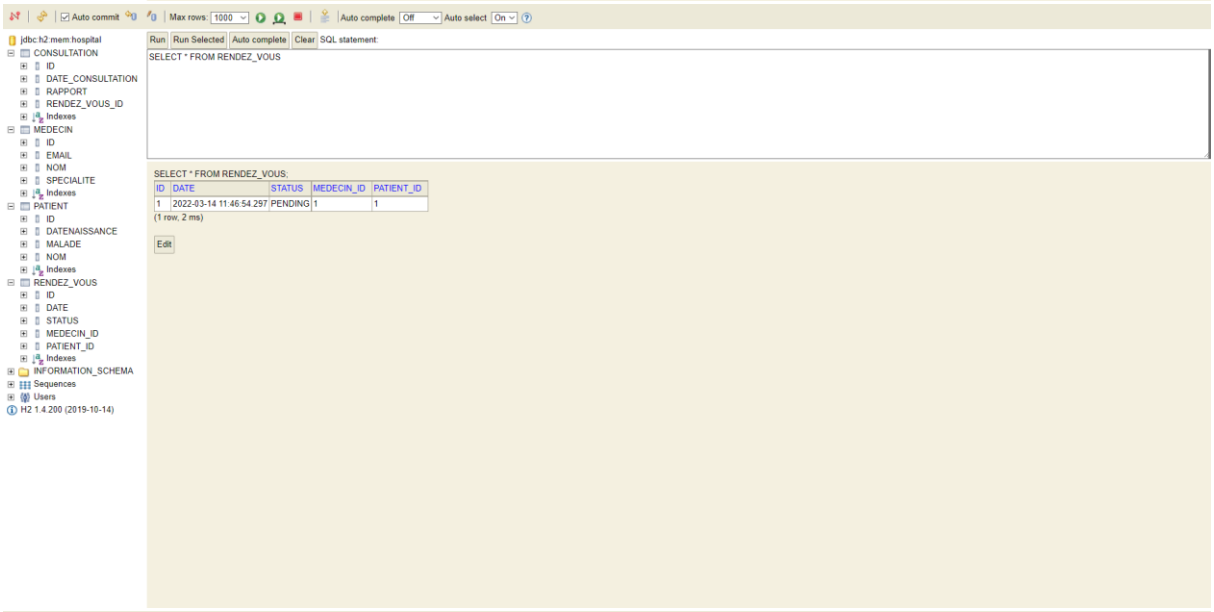
- Exécution :

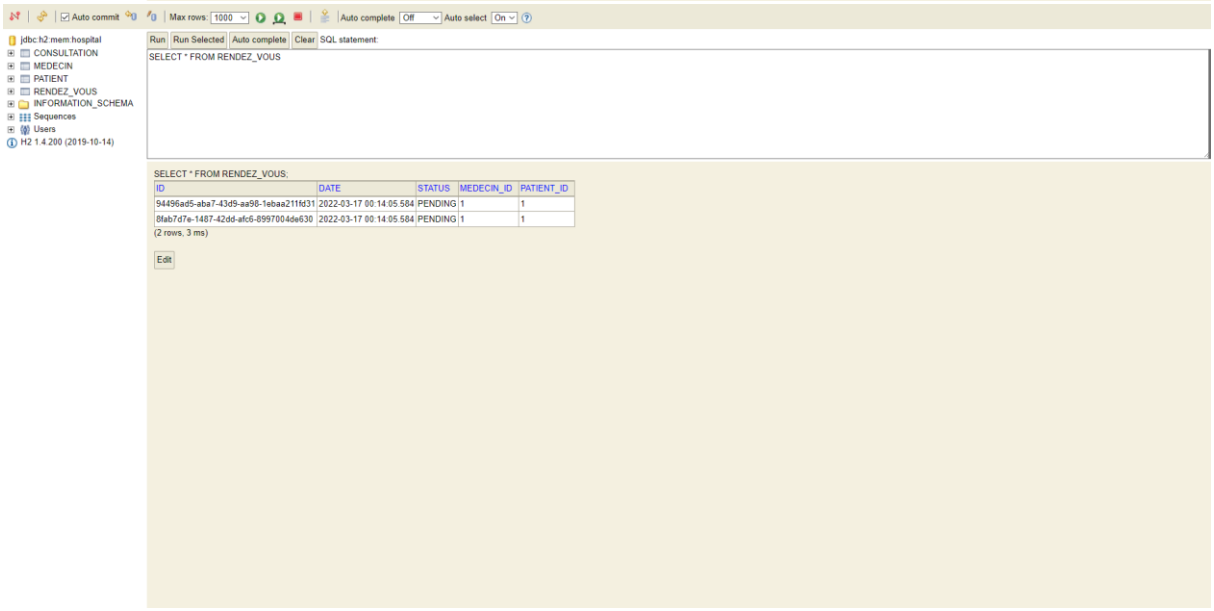

Auto commit
Max rows: 1000
Auto complete Off
Auto select On
jdbc:h2:mem:hospital
CONSULTATION
ID
DATE_CONSULTATION
RAPPORT
RENDEZ_VOUS_ID
Indexes
MEDECIN
ID
EMAIL
NOM
SPECIALITE
Indexes
PATIENT
ID
DATENAISSANCE
MALADE
NOM
Indexes
RENDEZ_VOUS
ID
DATE
STATUS
MEDECIN_ID
PATIENT_ID
Indexes
INFORMATION_SCHEMA
Sequences
Users
H2 1.4.200 (2019-10-14)
Run
Run Selected
Auto complete
Clear
SQL statement:
Important Commands
Displays this Help Page
Shows the Command History
Ctrl+Enter
Executes the current SQL statement
Shift+Enter
Executes the SQL statement defined by the text selection
Ctrl+Space
Auto complete
Disconnects from the database
Sample SQL Script
Delete the table if it exists
DROP TABLE IF EXISTS TEST;
Create a new table with ID and NAME columns
CREATE TABLE TEST(ID INT PRIMARY KEY, NAME VARCHAR(255));
Add a new row
INSERT INTO TEST VALUES(1, 'Hello');
Add another row
INSERT INTO TEST VALUES(2, 'World');
Query the table
SELECT * FROM TEST ORDER BY ID;
Change data in a row
UPDATE TEST SET NAME='Hi' WHERE ID=1;
Remove a row
DELETE FROM TEST WHERE ID=2;
Help
HELP ...
Adding Database Drivers
Additional database drivers can be registered by adding the Jar file location of the driver to the environment variables H2DRIVERS or CLASSPATH. Example (Windows): to add the database driver library C:/Programs/hsqldb/lib/hsqldb.jar, set the environment variable H2DRIVERS to C:/Programs/hsqldb/lib/hsqldb.jar.


Auto commit
Max rows: 1000
Auto complete Off
Auto select On
jdbc:h2:mem:hospital
CONSULTATION
ID
DATE_CONSULTATION
RAPPORT
RENDEZ_VOUS_ID
Indexes
MEDECIN
ID
EMAIL
NOM
SPECIALITE
Indexes
PATIENT
ID
DATENAISSANCE
MALADE
NOM
Indexes
RENDEZ_VOUS
ID
DATE
STATUS
MEDECIN_ID
PATIENT_ID
Indexes
INFORMATION_SCHEMA
Sequences
Users
H2 1.4.200 (2019-10-14)
Run
Run Selected
Auto complete
Clear
SQL statement:
SELECT * FROM PATIENT
SELECT * FROM PATIENT;
ID DATENAISSANCE MALADE NOM
1 2022-03-14 FALSE soukaina
2 2022-03-14 FALSE salma
3 2022-03-14 FALSE najat
(3 rows, 5 ms)
Edit


Auto commit
Max rows: 1000
Auto complete Off
Auto select On
jdbc:h2:mem:hospital
CONSULTATION
ID
DATE_CONSULTATION
RAPPORT
RENDEZ_VOUS_ID
Indexes
MEDECIN
ID
EMAIL
NOM
SPECIALITE
Indexes
PATIENT
ID
DATENAISSANCE
MALADE
NOM
Indexes
RENDEZ_VOUS
ID
DATE
STATUS
MEDECIN_ID
PATIENT_ID
Indexes
INFORMATION_SCHEMA
Sequences
Users
H2 1.4.200 (2019-10-14)
Run
Run Selected
Auto complete
Clear
SQL statement:
SELECT * FROM CONSULTATION
SELECT * FROM CONSULTATION;
ID DATE_CONSULTATION RAPPORT RENDEZ_VOUS_ID
1 2022-03-14 11:46:54.305 à ecrire apres 1
(1 row, 4 ms)
Edit







- Affichage en Web :

```
[{"id":1,"nom":"soukaina","datenaissance":"2022-03-17","malade":false,"rendezVous":[{"id":"a6cefeb8-2d28-483c-baf5-1199c4c4b43f","date":"2022-03-16T23:18:07.456+00:00","status":"PENDING","medecin":
{"id":1,"nom":"soukaina2","email":"soukaina2@gmail.com","specialite":"Dentiste"},"consultation":{"id":1,"dateConsultation":"2022-03-16T23:18:07.530+00:00","rapport":"à ecrire apres"}},{"id":"df9cbfc6-0292-46fc-bf5f-
d5375926lc51","date":"2022-03-16T23:18:07.456+00:00","status":"PENDING","medecin":{"id":1,"nom":"soukaina2","email":"soukaina2@gmail.com","specialite":"Dentiste"},"consultation":null}]},{"id":2,"nom":"salma","datenaissance":"2022-03-
17","malade":false,"rendezVous":[]},{"id":3,"nom":"najat","datenaissance":"2022-03-17","malade":false,"rendezVous":[]}]
```