

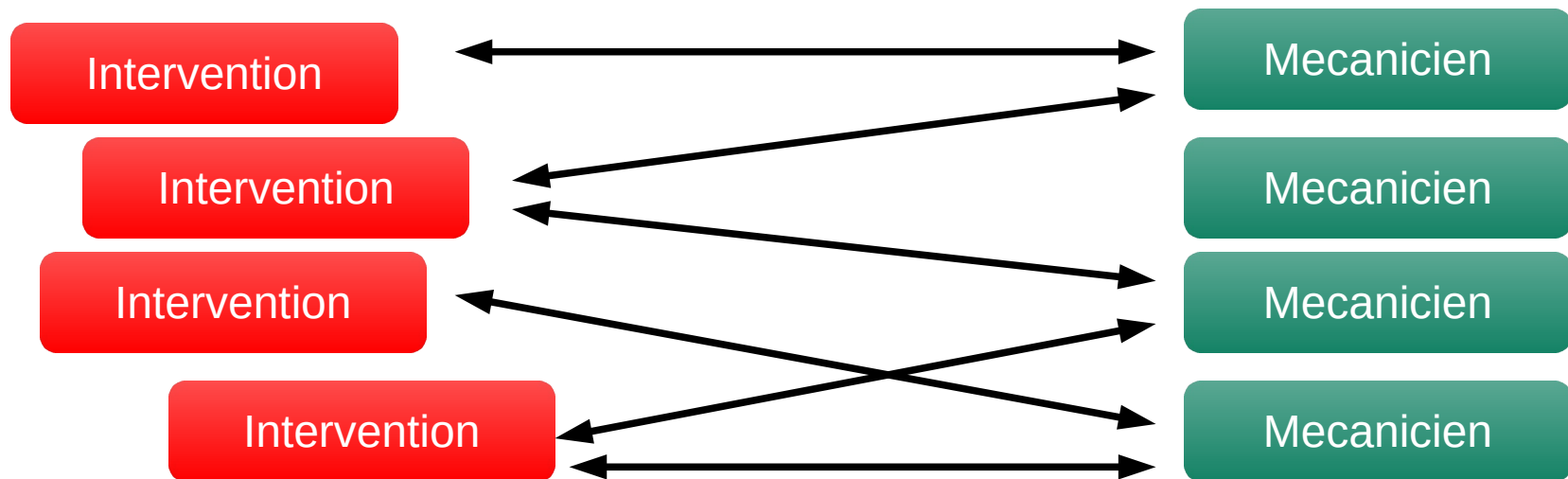


Hibernate relations

@ManyToMany

Vue d'ensemble

Une intervention peut impliquer plusieurs mécaniciens
Un mécanicien peut être affecté à plusieurs interventions



Si une intervention est supprimée nous ne souhaitons pas que les mécaniciens liés soient supprimés également (ne pas supprimer en cascade) et inversement, la suppression d'un mécanicien ne doit pas entraîner la suppression de ses interventions.

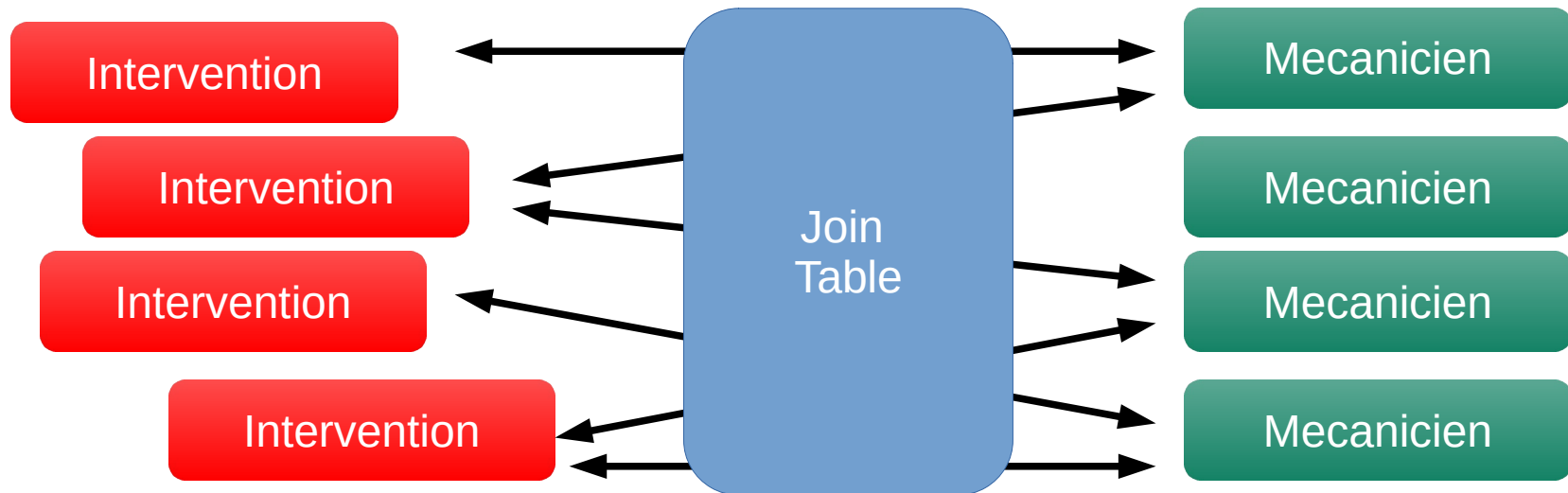
La création par contre sera en faite en cascade.

La relation Intervention-mecaniciens sera en lazy loading

Nous voudrions savoir quel mécanicien est sur quelle intervention et vice et versa.

Join Table

Pour cela c'est une table de jointure qui nous permettra de maintenir cette information

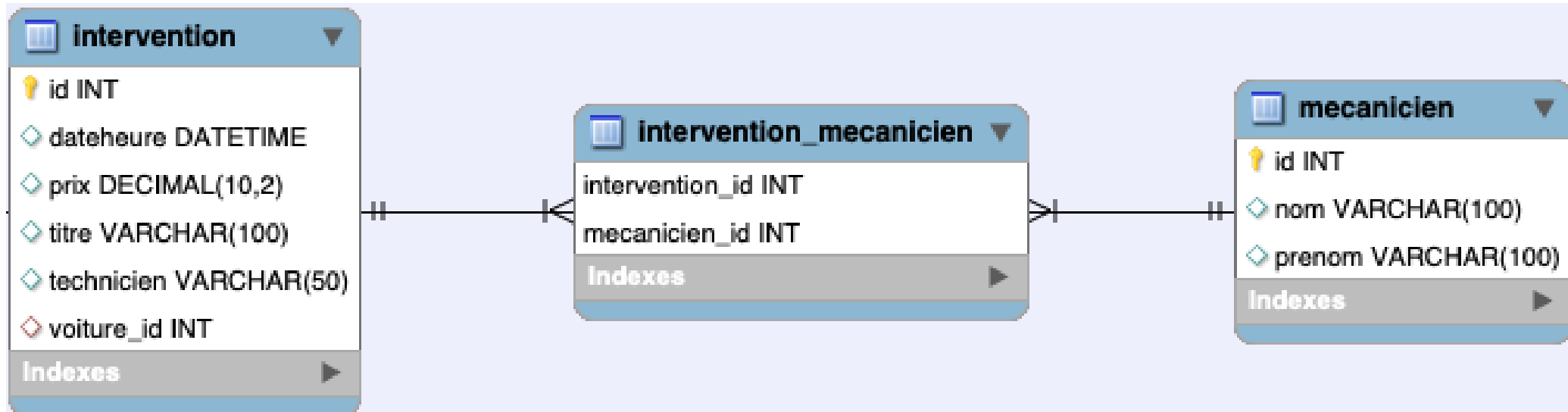
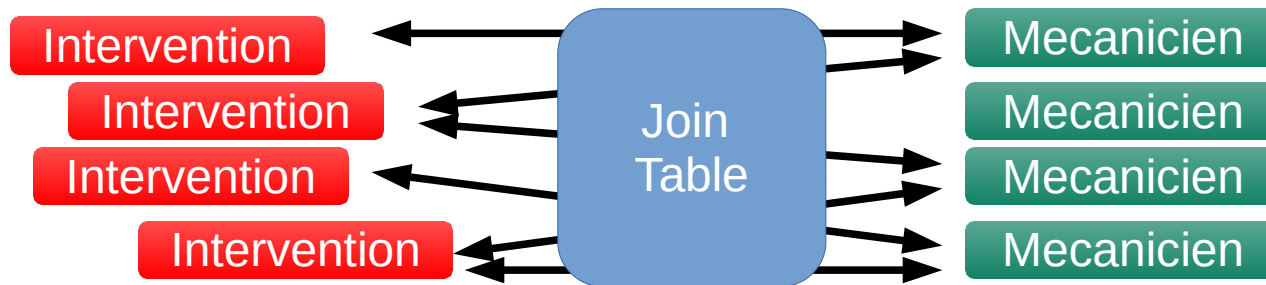


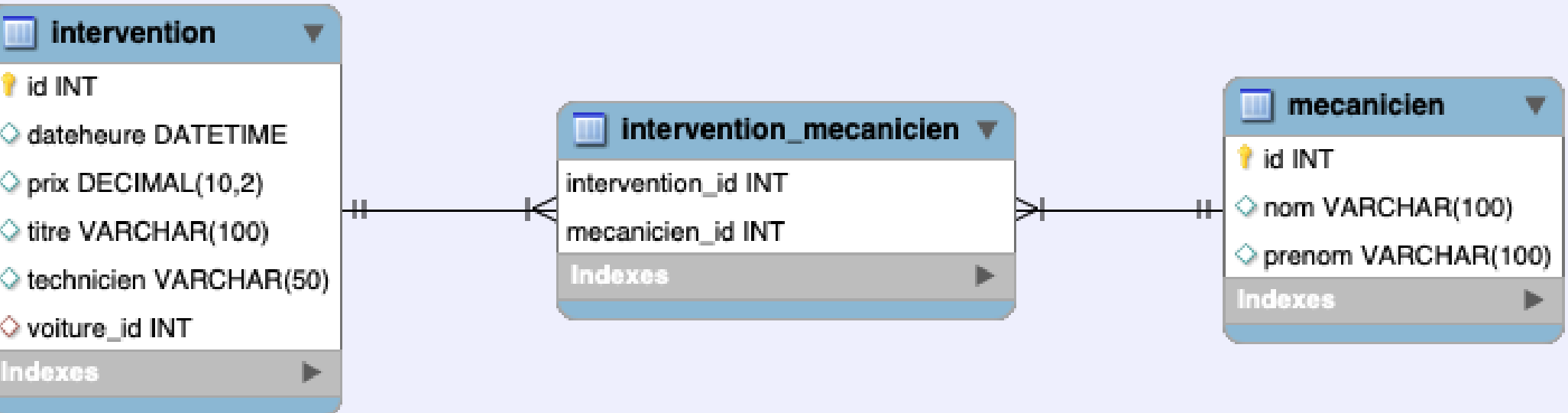
Définition :

Il s'agit d'une table qui fournit une correspondance entre deux tables.

Elle possède des clés étrangères vers chacune des tables jointes pour définir la relation.

Schéma de données





id	dateheure	prix	titre	technicien	voiture_id
1	2020-10-10 08:00:00	80.50	Petite Révision	A. Didonk	NULL
NULL	NULL	NULL	NULL	NULL	NULL

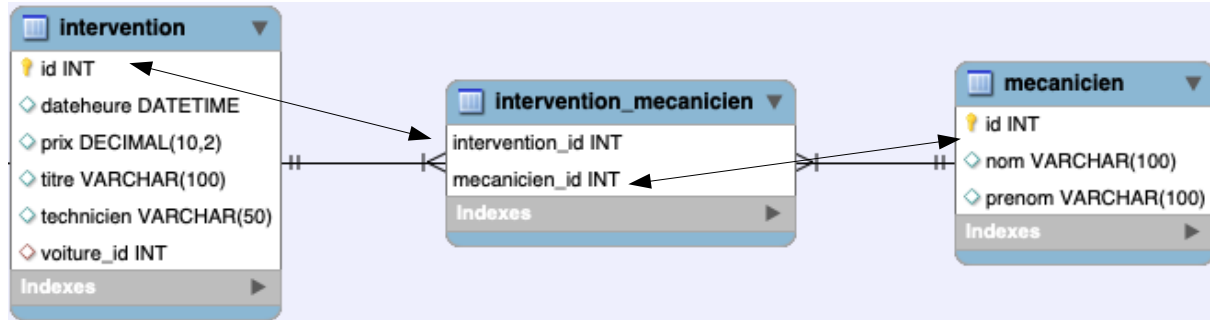
intervention_id	mecanicien_id
1	1
1	2
1	3
NULL	NULL

id	nom	prenom
1	Vincent	David
2	léponge	Bob
3	LeGrandGourou	Skipy
NULL	NULL	NULL

Notre programme

- Créer un nouveau projet java hibernate
- Préparer la bdd
- Maj de la classe Intervention.java
- Créer la classe Mechanicien.java
- Créer une methode main()

Create db sql



-- Table structure for table `mecanicien`

```
DROP TABLE IF EXISTS `mecanicien`;
CREATE TABLE `mecanicien` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `nom` varchar (100) DEFAULT NULL,
  `prenom` varchar (100) DEFAULT NULL,
```

```
PRIMARY KEY (`id`)
```

```
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=latin1;
```

-- Table structure for table `intervention_mecanicien`

```
DROP TABLE IF EXISTS `intervention_mecanicien`;
CREATE TABLE `intervention_mecanicien` (
  `intervention_id` INT NOT NULL ,
  `mecanicien_id` INT NOT NULL ,
```

```
PRIMARY KEY (`intervention_id`,`mecanicien_id`),
KEY `FK_MECHANICIEN_idx` (`mecanicien_id`),
```

```
CONSTRAINT `FK_INTERVENTION` FOREIGN KEY (`intervention_id`) REFERENCES `intervention` (`id`)
ON DELETE NO ACTION ON UPDATE NO ACTION ,
```

```
CONSTRAINT `FK_MECHANICIEN` FOREIGN KEY (`mecanicien_id`) REFERENCES `mecanicien` (`id`)
ON DELETE NO ACTION ON UPDATE NO ACTION
```

```
) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=latin1;
```

```
SET FOREIGN_KEY_CHECKS = 1;
```

Maj Intervention.java

```
@Entity
@Table(name="intervention")
public class Intervention {
    . . .

    @Column
    private String technicien;

    @ManyToOne(cascade= {CascadeType.PERSIST, CascadeType.MERGE,
                        CascadeType.DETACH, CascadeType.REFRESH})
    @JoinColumn(name="voiture_id")
    private Voiture voiture;

    @OneToMany(fetch=FetchType.LAZY, cascade=CascadeType.ALL)
    @JoinColumn(name="intervention_id")
    private List<Operation> operations ;

    @ManyToMany(fetch=FetchType.LAZY, cascade= {CascadeType.PERSIST,
        CascadeType.MERGE, CascadeType.DETACH, CascadeType.REFRESH })
    @JoinTable(
        name="intervention_mecanicien",
        joinColumns=@JoinColumn(name="intervention_id"),
        inverseJoinColumns=@JoinColumn(name="mecanicien_id")
    )
    private List<Mecanicien> mecaniciens;

    // add : méthode utile pour ajouter des mecaniciens
    // getters & setters ...
    // constructors. . .
```


Maj Intervention.java

```
@Entity
@Table(name="intervention")
public class Intervention {
    . . .

    @Column
    private String technicien;

    @ManyToOne(cascade= {CascadeType.PERSIST, CascadeType.MERGE,
                        CascadeType.DETACH, CascadeType.REFRESH})
    @JoinColumn(name="voiture_id")
    private Voiture voiture;

    @OneToMany(fetch=FetchType.LAZY, cascade=CascadeType.ALL)
    @JoinColumn(name="intervention_id")
    private List<Operation> operations ;

    @ManyToMany(fetch=FetchType.LAZY, cascade= {CascadeType.PERSIST,
        CascadeType.MERGE, CascadeType.DETACH, CascadeType.REFRESH})
    @JoinTable(
        name="intervention_mecanicien",
        joinColumns=@JoinColumn(name="intervention_id"),
        inverseJoinColumns=@JoinColumn(name="mecanicien_id")
    )
    private List<Mecanicien> mecaniciens;

    // add : méthode utile pour ajouter des mecaniciens
    // getters & setters ...
    // constructors. . .
}
```

Intervention_mecanicien ▼

intervention_id INT

mecanicien_id INT

Indexes ▶

Maj Intervention.java

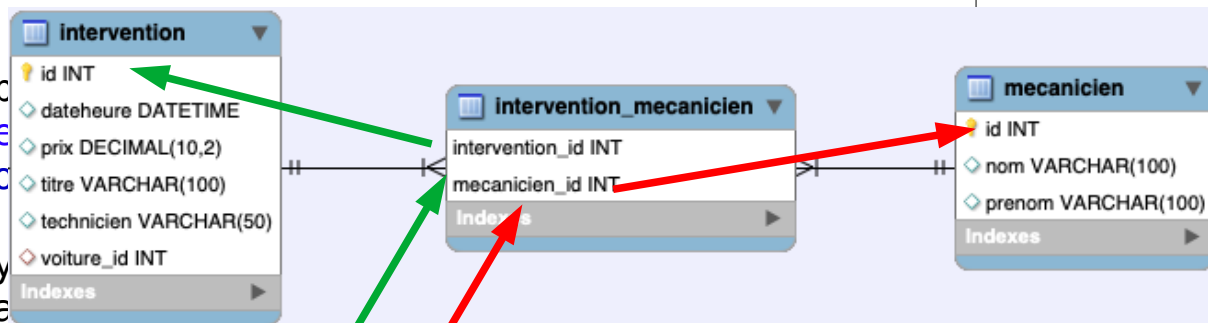
```
@Entity
@Table(name="intervention")
public class Intervention {
    . . .

    @Column
    private String technicien;

    @ManyToOne(cascade= {CascadeType.PERSIST, CascadeType.MERGE,
                        CascadeType.DETACH, CascadeType.REFRESH})
    @JoinColumn(name="voiture_id")
    private Voiture voiture;
```

```
@OneToMany(fetch=FetchType.LAZY)
@JoinColumn(name="intervention_id")
private List<Operation> operations;

@ManyToMany(fetch=FetchType.LAZY,
             cascade={CascadeType.PERSIST, CascadeType.MERGE,
                     CascadeType.DETACH, CascadeType.REFRESH})
@JoinTable(
    name="intervention_mecanicien",
    joinColumns=@JoinColumn(name="intervention_id"),
    inverseJoinColumns=@JoinColumn(name="mecanicien_id")
)
private List<Mecanicien> mecaniciens;
```



```
// add : méthode utile pour ajouter des mecaniciens
// getters & setters ...
// constructors. . .
```

Mecanicien.java

@Entity

public class Mecanicien {

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

@Column(name="id")

private int id;

@Column

private String nom;

@Column

private String prenom;

```
@ManyToMany(fetch=FetchType.LAZY, cascade= {CascadeType.PERSIST,
CascadeType.MERGE, CascadeType.DETACH, CascadeType.REFRESH })
@JoinTable(
    name="intervention_mecanicien",
    joinColumns=@JoinColumn(name="mecanicien_id"),
    inverseJoinColumns=@JoinColumn(name="intervention_id")
)
private List<Intervention> interventions;
```

//getters() @ Setters() . . .

//Constructors . . .

intervention_mecanicien ▼
intervention_id INT
mecanicien_id INT
Indexes ▶

JoinTable

- Dans Mecanicien, Join Table demande à hibernate :

Regarde la colonne `mecanicien_id` dans la table `intervention_mecanicien`

pour le retour (inverse, de l'autre côté de la relation) regarde la colonne `Intervention_id` dans la table `intervention_mecanicien`

utilise cette information pour trouver des relations entre `mecanicien` et `intervention`

main()

```
// créer un objet Intervention
Intervention i = new Intervention("Petite Révision", 80.5, "A. Didonk",
LocalDateTime.of(2020, Month.OCTOBER, 10, 10, 00, 00) );

// save l'intervention
session.save(i);

//afficher l'intervention sauvegardée
System.out.println("intervention sauvegardée : "+i);

// créer les mécaniciens
Mecanicien m1 = new Mecanicien( "Vincent", "David");
Mecanicien m2 = new Mecanicien("léponge", "Bob");
Mecanicien m3 = new Mecanicien("LeGrandGourou", "SkiPy" );

// ajouter les mecaniciens à l'intervention
i.add(m1);
i.add(m2);
i.add(m3);

// save les mécaniciens
System.out.println("\n sauvegarde des mécaniciens ...");
session.save(m1);
session.save(m2);
session.save(m3);
System.out.println("\n mécaniciens sauvegardés : "+i.getMecaniciens());

//affiche les mecaniciens associés
System.out.println("mecaniciens : "+i.getMecaniciens());
```

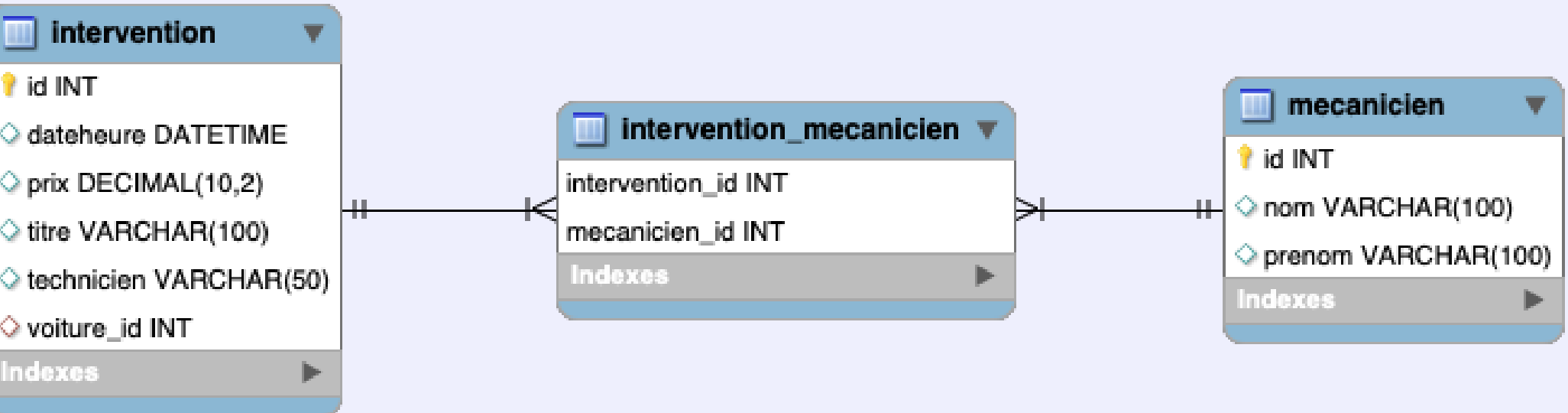
Affichage console

```
août 05, 2020 6:03:10 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildcreator
INFO: HHH10001003: Autocommit mode: false
août 05, 2020 6:03:10 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH000115: Hibernate connection pool size: 1 (min=1)
août 05, 2020 6:03:10 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: insert into intervention (dateheure, prix, technicien, titre, voiture_id) values (?, ?, ?, ?, ?)
intervention sauvegardée : Intervention [id=1, dateheure=2020-10-10T10:00, titre=Petite Révision, prix=80.5, technicien=A. Didonk

sauvegarde des mécaniciens ...
Hibernate: insert into Mecanicien (nom, prenom) values (?, ?)
Hibernate: insert into Mecanicien (nom, prenom) values (?, ?)
Hibernate: insert into Mecanicien (nom, prenom) values (?, ?)

mécaniciens sauvegardés : [Mecanicien [id=1, nom=Vincent, prenom=David, interventions=null], Mecanicien [id=2, nom=léponge, prenom=Bob, interventions=null]]
mécaniciens : [Mecanicien [id=1, nom=Vincent, prenom=David, interventions=null], Mecanicien [id=2, nom=léponge, prenom=Bob, interventions=null]]
Hibernate: insert into intervention_mecanicien (intervention_id, mecanicien_id) values (?, ?)
Hibernate: insert into intervention_mecanicien (intervention_id, mecanicien_id) values (?, ?)
Hibernate: insert into intervention_mecanicien (intervention_id, mecanicien_id) values (?, ?)
Terminé !
août 05, 2020 6:03:11 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PoolState stop
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/bdd_08_many_to_many?useSSL=false&serverTimezone=UTC]
```

Insert dans la table de
jointure



Result Grid							
Filter Rows: <input type="text" value="Search"/>							
	id	dateheure	prix	titre	technicien	voiture_id	
▶	1	2020-10-10 08:00:00	80.50	Petite Révision	A. Didonk	NULL	
	NULL	NULL	NULL	NULL	NULL	NULL	

	intervention_id	mecanicien_id	
▶	1	1	
	1	2	
	1	3	
	NULL	NULL	

	id	nom	prenom	
▶	1	Vincent	David	
	2	léponge	Bob	
	3	LeGrandGourou	Skipy	
	NULL	NULL	NULL	



Ajouter plus d'interventions à un mecanicien

Nouvelle classe exécutable :

```
public class AddInterventionsToOneMecanicienDemo{

    public static void main(String[] args) {

        // create session factory
        SessionFactory factory = new Configuration().configure("hibernate.cfg.xml")
                                                    .addAnnotatedClass(Voiture.class)
                                                    .addAnnotatedClass(Moteur.class)
                                                    .addAnnotatedClass(Intervention.class)
                                                    .addAnnotatedClass(Operation.class)
                                                    .addAnnotatedClass(Mecanicien.class)
                                                    .buildSessionFactory();

        // déclarer session
        Session session = null;
        try {

            // récupérer une session & ouvrir une transaction
            session = factory.getCurrentSession();
            session.beginTransaction();

            //commit transaction
            session.getTransaction().commit();

            session.close();
            System.out.println("_____Terminé !");

        }
        finally {
            //régler le pb de connection leak
            session.close();
            factory.close();
        }
    }
}
```


Commentaires

```
try {  
  
    // récupérer une session & ouvrir une transaction  
    session = factory.getCurrentSession();  
    session.beginTransaction();  
  
    // get un mecanicien  
  
    // créer plus d'interventions  
  
    // ajouter le mecanicien à ces interventions  
  
    // enregistrer les interventions  
  
  
    //commit transaction  
    session.getTransaction().commit();  
  
    session.close();  
    System.out.println("_____Terminé !");  
  
}  
finally {  
  
    session.close();  
    factory.close();  
}
```

	id	nom	prenom	
▶	1	Vincent	David	
	2	léponge	Bob	
	3	LeGrandGourou	Skipy	
	NULL	NULL	NULL	

AddInterventionsToOneMecanicienDemo.java

```
public class AddInterventionsToOneMecanicienDemo{

    public static void main(String[] args) {

        // récupérer une session & ouvrir une transaction
        session = factory.getCurrentSession();
        session.beginTransaction();

        // get le mecanicien S. LeGrandGourou
        Mecanicien skipy = session.get(Mecanicien.class, 3);
        System.out.println("Le mecanicien récupéré est :"+skipy);
        System.out.println("Interventions liées : "+skipy.getInterventions());

        // créer plus d'interventions
        Intervention i1= new Intervention("Revision des 130 000km", 205.10,
            "J. Denovan",LocalDateTime.of(2020, Month.SEPTEMBER,18,10,30,00));
        Intervention i2= new Intervention("Courroie de distribution", 860.70,
            "J. Denovan",LocalDateTime.of(2020, Month.SEPTEMBER,18,14,00,00));

        // ajouter le mecanicien à ces interventions
        i1.add(skipy);
        i2.add(skipy);

        // enregistrer les interventions
        System.out.println("\n Enregistrement des Interventions .. ");
        session.save(i1);
        session.save(i2);

        //commit transaction
        session.getTransaction().commit();
        session.close();
    }
}
```

Affichage en console

```
INFO: HHH000115: Hibernate connection pool size: 1 (min=1)
août 05, 2020 10:02:50 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: select mecanicien0_.id as id1_2_0_, mecanicien0_.nom as nom2_2_0_, mecanicien0_.prenom as prenom3_2_0_
Hibernate: select interventi0_.mecanicien_id as mecanici1_1_0_, interventi0_.intervention_id as interven2_1_0_,
Le mecanicien récupéré est :Mecanicien [id=3, nom=LeGrandGourou, prenom=Skipy, interventions=[ Intervention [id=
]]
Interventions liées : [ Intervention [id=1, dateheure=2020-10-10T10:00, titre=Petite Révision, prix=80.5, techni
]

Enregistrement des Interventions ..
Hibernate: insert into intervention (dateheure, prix, technicien, titre, voiture_id) values (?, ?, ?, ?, ?)
Hibernate: insert into intervention (dateheure, prix, technicien, titre, voiture_id) values (?, ?, ?, ?, ?)
Hibernate: insert into intervention_mecanicien (intervention_id, mecanicien_id) values (?, ?)
Hibernate: insert into intervention_mecanicien (intervention_id, mecanicien_id) values (?, ?)
Terminé !
août 05, 2020 10:02:51 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$Pool
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/bdd_08_many_to_many?useSSL=false&ser
```

Ajout des relations
dans la table de jointure



id	dateheure	prix	titre	technicien	voiture_id
1	2020-10-10 08:00:00	80.50	Petite Révision	A. Didonk	NULL
2	2020-09-18 08:30:00	205.10	Revision des...	J. Denovan	NULL
3	2020-09-18 12:00:00	860.70	Courroie de d...	J. Denovan	NULL
NULL	NULL	NULL	NULL	NULL	NULL

intervention_id	mecanicien_id
1	1
1	2
1	3
2	3
3	3
NULL	NULL

Nouveau Use Case : lire les interventions d'un mécanicien

Nouvelle classe exécutable :

```
public class GetInterventionsDeSkipyDemo{

    public static void main(String[] args) {

        . . .
        try {

            // récupérer une session & ouvrir une transaction
            session = factory.getCurrentSession();
            session.beginTransaction();

            // get le mécanicien S. LeGrandGourou
            Mecanicien skipy = session.get(Mecanicien.class, 3);
            System.out.println("Le mécanicien récupéré est :"+skipy);
            System.out.println("Interventions liées : "+skipy.getInterventions());

            //commit transaction
            session.getTransaction().commit();
            . . .
        }
    }
}
```

ExE

Affichage console

Selectionner cette
loooooongue requête .

```
août 05, 2020 10:14:59 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: select mecanicien0_.id as id1_2_0_, mecanicien0_.nom as nom2_2_0_, mecanicien0_.prenom as prenc
Hibernate: select interventi0_.mecanicien_id as mecanici1_1_0_, interventi0_.intervention_id as interven2_
Le mecanicien récupéré est :Mecanicien [id=3, nom=LeGrandGourou, prenom=Skipy, interventions=[ Interventic
, Intervention [id=2, dateheure=2020-09-18T10:30, titre=Revision des 130 000km, prix=205.1, technicien=J.
, Intervention [id=3, dateheure=2020-09-18T14:00, titre=Courroie de distribution, prix=860.7, technicien=
]]
Interventions liées : [ Intervention [id=1, dateheure=2020-10-10T10:00, titre=Petite Révision, prix=80.5,
, Intervention [id=2, dateheure=2020-09-18T10:30, titre=Revision des 130 000km, prix=205.1, technicien=J.
, Intervention [id=3, dateheure=2020-09-18T14:00, titre=Courroie de distribution, prix=860.7, technicien=
]
Terminé !
août 05, 2020 10:15:00 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderIn
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost
```

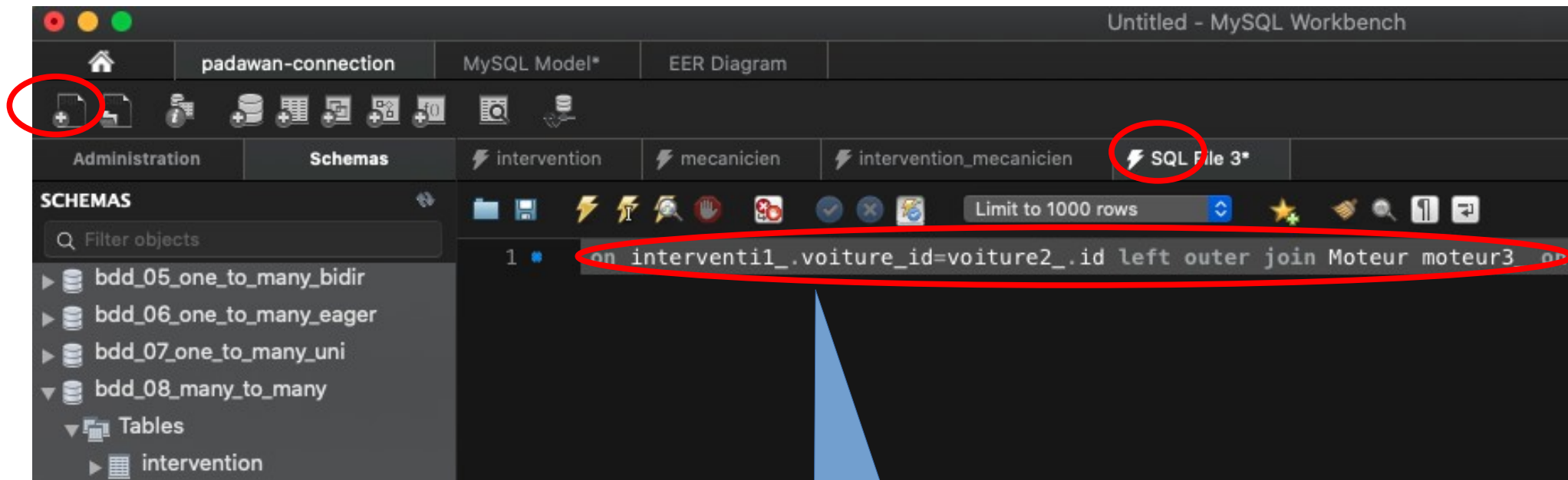
Les interventions insérées sont
toutes présentes

?

```
select interventi0_.mecanicien_id as mecanici1_1_0_,
interventi0_.intervention_id as interven2_1_0_, interventi1_.id as
id1_0_1_, interventi1_.dateheure as dateheur2_0_1_, interventi1_.prix as
prix3_0_1_, interventi1_.technicien as technici4_0_1_, interventi1_.titre
as titre5_0_1_, interventi1_.voiture_id as voiture_6_0_1_, voiture2_.id
as id1_5_2_, voiture2_.immatriculation as immatric2_5_2_,
voiture2_.modele as modele3_5_2_, voiture2_.moteur_id as moteur_i4_5_2_,
moteur3_.id as id1_3_3_, moteur3_.carburant as carburan2_3_3_,
moteur3_.cylindree as cylindre3_3_3_, moteur3_.puissance as
puissanc4_3_3_ from intervention_mecanicien interventi0_ inner join
intervention interventi1_ on interventi0_.intervention_id=interventi1_.id
left outer join voiture voiture2_ on interventi1_.voiture_id=voiture2_.id
left outer join Moteur moteur3_ on voiture2_.moteur_id=moteur3_.id where
interventi0_.mecanicien_id=?
```



Exécutez cette requête dans MySQL Workbench





Exploiter l'affichage console

Retour à la ligne
Avant la clause "where"

```
1 on intervent11.voiture_id=voiture2.id left outer join Moteur moteur3_ on voiture2.moteur_id=moteur3.id where interventio.mecanicien_id=?
```



Modifier l'id

```
1 select interventio.mecanicien_id as mecanici1_1_0_, interventio.int
2 where interventio.mecanicien_id=3
```




On récupère les interventions de notre mécanicien

mecanici1_1_0_	interven2_1_0_	id1_0_1_	dateheur2_0_1_	prix3_0_1_	technici4_0_1_	titre5_0_1_	voiture_6_0_1_	id1_5_2_	immatric2_5_2_	modele3_5_2_
3	1	1	2020-10-10 08:00:00	80.50	A. Didonk	Petite Révision	NULL	NULL	NULL	NULL
3	2	2	2020-09-18 08:30:00	205.10	J. Denovan	Revision des 130 000km	NULL	NULL	NULL	NULL
3	3	3	2020-09-18 12:00:00	860.70	J. Denovan	Courroie de distribution	NULL	NULL	NULL	NULL

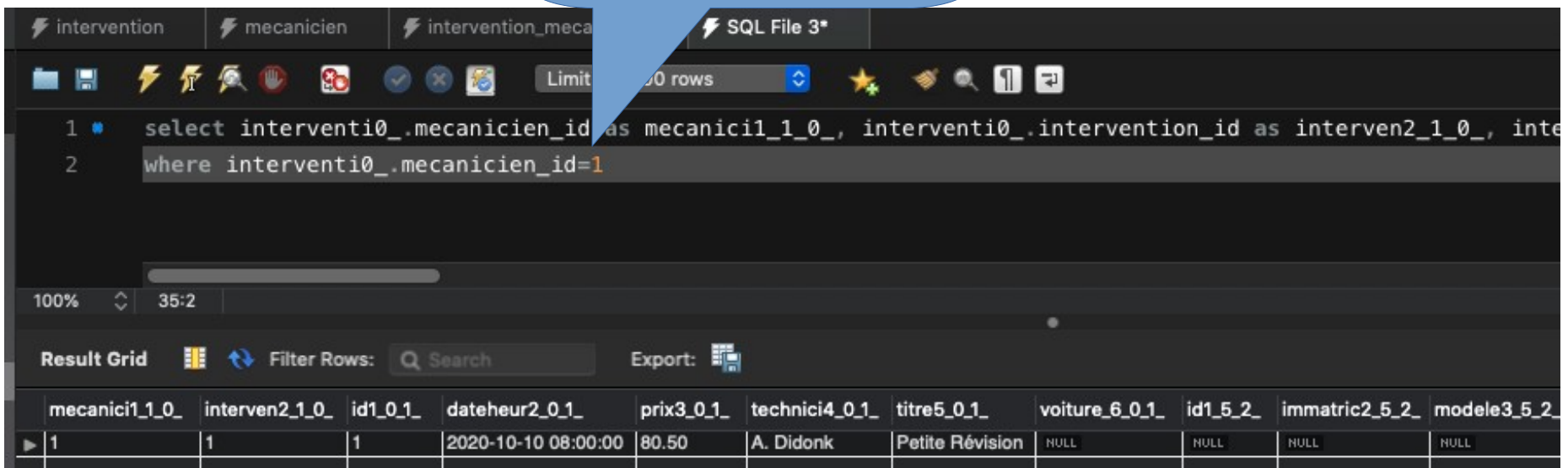
Nouveau Use Case : supprimer une Intervention , et confirmer que les mécaniciens existent encore

Nouvelle classe exécutable :

```
public class DeletetInterventionPetiteRevisionDemo{  
  
    public static void main(String[] args) {  
  
        ...  
        try {  
  
            // récupérer une session & ouvrir une transaction  
            session = factory.getCurrentSession();  
            session.beginTransaction();  
  
            // get l'intervention "Petite Révision"  
  
            // delete l'intervention  
  
            //commit transaction  
            session.getTransaction().commit();  
            . . .  
        }  
    }  
}
```

Je veux les interventions du mécanicien V.David dont l'id =1

Je réexecute la
précédente requête
avec l'i =1



The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and search. The SQL editor contains the following query:

```
1 select interventi0.mecanicien_id as mecanici1_1_0_, interventi0.intervention_id as interven2_1_0_, inte
2 where interventi0.mecanicien_id=1
```

Below the editor, the 'Result Grid' is displayed with a search bar and an 'Export' button. The results table has the following columns and data:

mecanici1_1_0_	interven2_1_0_	id1_0_1_	dateheur2_0_1_	prix3_0_1_	technici4_0_1_	titre5_0_1_	voiture_6_0_1_	id1_5_2_	immatric2_5_2_	modele3_5_2_
1	1	1	2020-10-10 08:00:00	80.50	A. Didonk	Petite Révision	NULL	NULL	NULL	NULL

Conclusion : le mécanicien id=1 (D.Vincent) n'est intervenu que
sur la Petite Revision (intervention id= 1 aussi)

DeleteInterventionPetiteRevisionDemo.java

id	dateheure	prix	titre	tech
1	2020-10-10 08:00:00	80.50	Petite Révision	A. D
2	2020-09-18 08:30:00	205.10	Revision des...	J. D
3	2020-09-18 12:00:00	860.70	Courroie de d...	J. D
NULL	NULL	NULL	NULL	NULL

```
...
// récupérer une session & ouvrir une transaction
session = factory.getCurrentSession();
session.beginTransaction();

// get l'intervention "Petite Révision"
int id= 1;
Intervention petiteRevision=
session.get(Intervention.class,1);

// delete l'intervention
session.delete(petiteRevision);

//commit transaction
session.getTransaction().commit();
...
```

EXE

Vérification du résultat

```
août 05, 2020 10:58:26 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: select interventi0_.id as id1_0_0_, interventi0_.dateheure as dateheur2_0_0_, inter
Hibernate: select operations0_.intervention_id as interven5_4_0_, operations0_.id as id1_4_0_,
Hibernate: delete from intervention_mecanicien where intervention_id=?
Hibernate: delete from intervention where id=?
_____Terminé !
août 05, 2020 10:58:27 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnecti
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/bdd_08 many to man
```

La petite révision n'est plus en bdd

▶	2	2020-09-18 08:30:00	205.10	Revision des 130 000km	J. Denovan	NULL	
	3	2020-09-18 12:00:00	860.70	Courroie de distribution	J. Denovan	NULL	
	NULL	NULL	NULL	NULL	NULL	NULL	

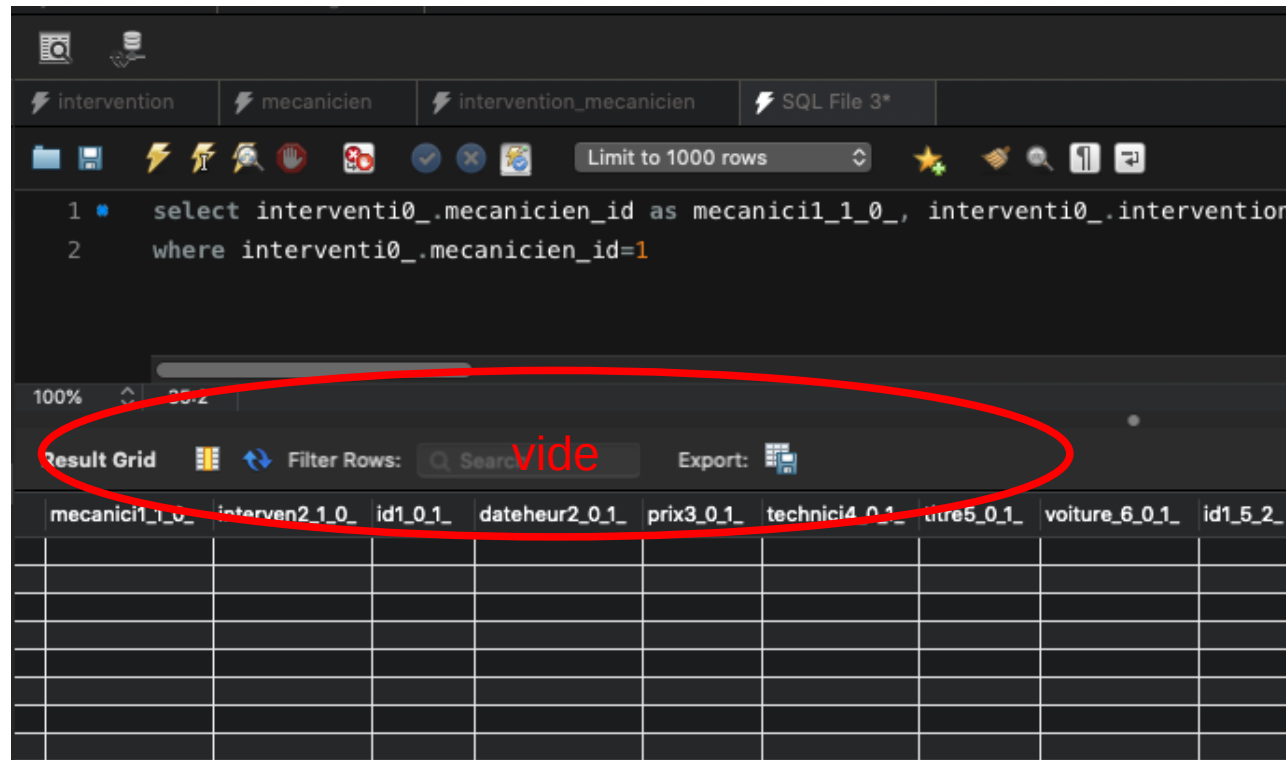
	intervention_id	mecanicien_id
▶	2	3
	3	3
	NULL	NULL

Plus de relation entre l'intervention n°1 et des mécaniciens n'existe dans la Joion table (table d'association)

	id	nom	prenom
▶	1	Vincent	David
	2	léponge	Bob
	3	LeGrandGourou	Skipy
	NULL	NULL	NULL

Le mécanicien n° 1 n'a pas été supprimé

Réexécuter la requête dans mySQL Worbench pour consulter les Interventions du mécanicien n°1



GetInterventionsDeDavidVincentDemo.java


```
...
// récupérer une session & ouvrir une transaction
session = factory.getCurrentSession();
session.beginTransaction();

// get le mecanicien
Mecanicien m = session.get(Mecanicien.class, 1);
System.out.println("Le mecanicien récupéré est :"+m);
System.out.println("Interventions liées : "+m.getInterventions());

//commit transaction
session.getTransaction().commit();

...
```

id	nom	prenom
1	Vincent	David
2	léponge	Bob
3	LeGrandGourou	Skipy
NULL	NULL	NULL



```
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: select mecanicien0_.id as id1_2_0_, mecanicien0_.nom as n
Hibernate: select interventi0_.mecanicien_id as mecanici1_1_0_, inte
Le mecanicien récupéré est :Mecanicien [id=1, nom=Vincent, prenom=Da
Interventions liées : []
_____Terminé !
août 05, 2020 11:17:15 PM org.hibernate.engine.jdbc.connections.inte
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localho
```

Dernier Use Case : supprimer un mecanicien , sans supprimer ses interventions.

Nouvelle classe exécutable :

```
public class DeleteSkippyMecanicienDemo{

    public static void main(String[] args) {
        .
        .
        .
        try {

            // récupérer une session & ouvrir une transaction
            session = factory.getCurrentSession();
            session.beginTransaction();

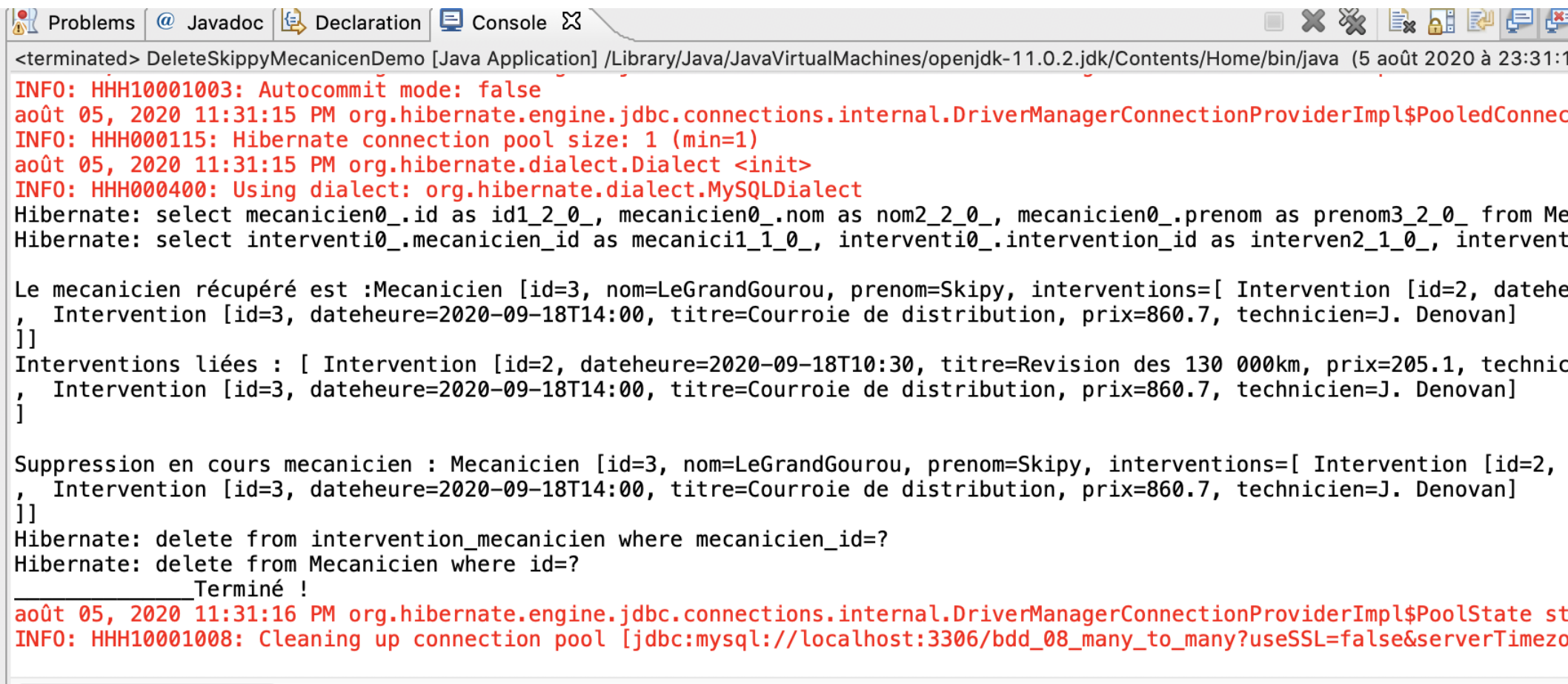
            // get le mecanicien S. LeGrandGourou (son id =3)
            int id=3;
            Mecanicien m = session.get(Mecanicien.class, id);
            System.out.println("\nLe mecanicien récupéré est :"+m);
            System.out.println("Interventions liées : "+m.getInterventions());

            //delete Mecanicien
            System.out.println("\nSuppression en cours mecanicien : "+m);
            session.delete(m);

            //commit transaction
            session.getTransaction().commit();
            .
            .
            .
        }
    }
}
```

EXE

Affichage Console



The screenshot shows an IDE window with a tab labeled 'Console'. The console output is as follows:

```
<terminated> DeleteSkippyMecanicienDemo [Java Application] /Library/Java/JavaVirtualMachines/openjdk-11.0.2.jdk/Contents/Home/bin/java (5 août 2020 à 23:31:15)
INFO: HHH10001003: Autocommit mode: false
août 05, 2020 11:31:15 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnection
INFO: HHH000115: Hibernate connection pool size: 1 (min=1)
août 05, 2020 11:31:15 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: select mecanicien0_.id as id1_2_0_, mecanicien0_.nom as nom2_2_0_, mecanicien0_.prenom as prenom3_2_0_ from Mecanicien0_
Hibernate: select interventi0_.mecanicien_id as mecanici1_1_0_, interventi0_.intervention_id as interven2_1_0_, interventi0_.dateheure as dateheure3_1_0_ from Intervention0_ where mecanicien_id=3
Le mecanicien récupéré est :Mecanicien [id=3, nom=LeGrandGourou, prenom=Skipy, interventions=[ Intervention [id=2, dateheure=2020-09-18T10:30, titre=Revision des 130 000km, prix=205.1, technicien=J. Denovan], Intervention [id=3, dateheure=2020-09-18T14:00, titre=Courroie de distribution, prix=860.7, technicien=J. Denovan]
]]
Interventions liées : [ Intervention [id=2, dateheure=2020-09-18T10:30, titre=Revision des 130 000km, prix=205.1, technicien=J. Denovan], Intervention [id=3, dateheure=2020-09-18T14:00, titre=Courroie de distribution, prix=860.7, technicien=J. Denovan]
]
Suppression en cours mecanicien : Mecanicien [id=3, nom=LeGrandGourou, prenom=Skipy, interventions=[ Intervention [id=2, dateheure=2020-09-18T10:30, titre=Revision des 130 000km, prix=205.1, technicien=J. Denovan], Intervention [id=3, dateheure=2020-09-18T14:00, titre=Courroie de distribution, prix=860.7, technicien=J. Denovan]
]]
Hibernate: delete from intervention_mecanicien where mecanicien_id=?
Hibernate: delete from Mecanicien where id=?
Terminé !
août 05, 2020 11:31:16 PM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PoolState
INFO: HHH10001008: Cleaning up connection pool [jdbc:mysql://localhost:3306/bdd_08_many_to_many?useSSL=false&serverTimezone=UTC]
```


	id	nom	prenom	
▶	1	Vincent	David	
	2	léponge	Bob	
	NULL	NULL	NULL	

LeGrandGourou n'existe plus en bdd

	id	dateheure	prix	titre	technicien	voiture_id	
▶	2	2020-09-18 08:30:00	205.10	Revision des 130 000km	J. Denovan	NULL	
	3	2020-09-18 12:00:00	860.70	Courroie de distribution	J. Denovan	NULL	
	NULL	NULL	NULL	NULL	NULL	NULL	

Les interventions du mecanicien existent toujours dans la bdd

	intervention_id	mechanicien_id	
▶	NULL	NULL	

La table d'association est mise à jour