JPA HIBERNATE EXERCISE 3

1. **Create a class Address for Author with instance variables streetNumber, location, State.**

   //address entity

```
    package com.example.exercise.q1to2.entity;
import javax.persistence.Embeddable;
@Embeddable
public class Address {
   private String streetNumber;
   private String location;
   private String state;
   public String getStreetNumber() {
      return streetNumber;
   }
   public void setStreetNumber(String streetNumber) {
      this.streetNumber = streetNumber;
   }
   public String getLocation() {
      return location;
   }
   public void setLocation(String location) {
      this.location = location;
   }
   public String getState() {
      return state;
   }
   public void setState(String state) {
      this.state = state;
   }
}
```

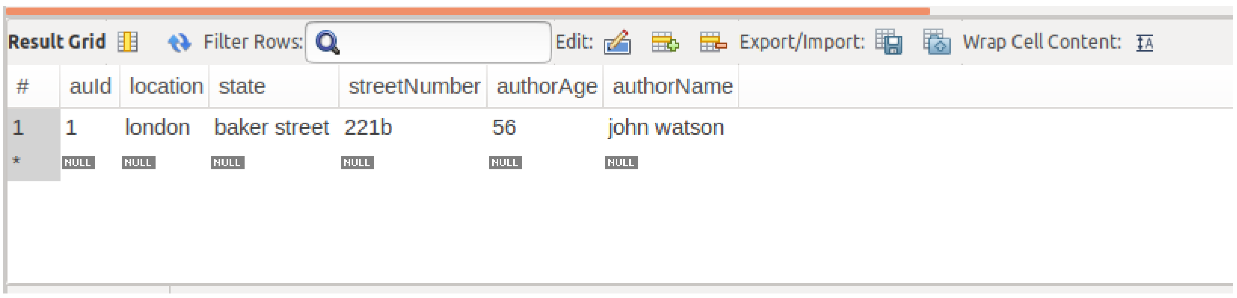
2. **Create instance variable of Address class inside Author class and save it as embedded object.**

   // AUTHOR ENTITY

```
    package com.example.exercise.q1to2.entity;
import javax.persistence.Embeddable;
@Embeddable
public class Address {
   private String streetNumber;
   private String location;
   private String state;
   public String getStreetNumber() {
      return streetNumber;
   }
   public void setStreetNumber(String streetNumber) {
      this.streetNumber = streetNumber;
   }
   public String getLocation() {
      return location;
   }
   public void setLocation(String location) {
      this.location = location;
   }
   public String getState() {
      return state;
   }
   public void setState(String state) {
      this.state = state;
   }
}
```

// creating author in Q1To4Tests test.java with address as embedded object

```
@SpringBootTest

class Q1To4Tests {
   @Autowired
   Author1to2Repository author1to2Repository;
   @Test
   void contextLoads() {
   }
   // Q1 to 4
   @Test // creating a new author with address as embedded object
   // Persist 3 subjects for each author.
   void testCreateAuthor1(){
      Author author=new Author();
      author.setAuthorName("mason");
      author.setAuthorAge(45);
      Address address=new Address();
      address.setStreetNumber("565");
      address.setState("york shire");
      address.setLocation("hobbiton");
      author.setAddress(address);
      author1to2Repository.save(author);
   }
```



#	auId	location	state	streetNumber	authorAge	authorName
1	1	london	baker street	221b	56	john watson
*	NULL	NULL	NULL	NULL	NULL	NULL

3. **Introduce a List of subjects for author.**
   // **SUBJECTS ENTITY**

```
      @Entity
public class Subjects {
    @Id
    private int subId;
    private String subjectName;
    @ManyToOne(cascade = CascadeType.ALL)
    @JoinColumn(name = "auId")
    private Author author;
    public int getSubId() {
        return subId;
    }
    public void setSubId(int subId) {
        this.subId = subId;
    }
    public String getSubjectName() {
        return subjectName;
    }
    public void setSubjectName(String subjectName) {
        this.subjectName = subjectName;
    }
    public Author getAuthor() {
        return author;
    }
    public void setAuthor(Author author) {
        this.author = author;
    }
}
```

// **Updated Author Entity**

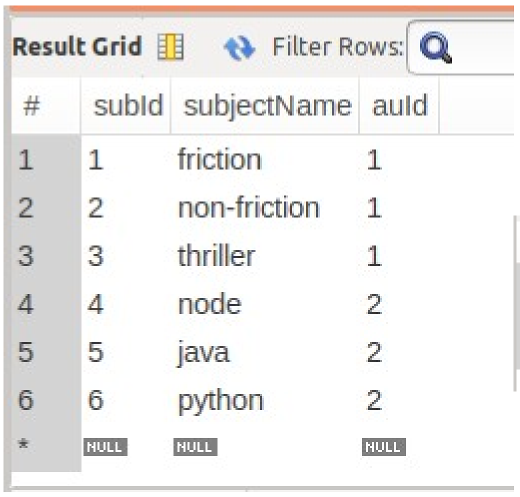
```
package com.example.exercise.q1to4.entity;

import javax.persistence.*;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
@Entity
public class Author {
   @Id
   @GeneratedValue(strategy = GenerationType.AUTO)
   private int auId;
   private String authorName;
   private int authorAge;
   @Embedded
   private Address address;
   @OneToMany(mappedBy = "author",cascade = CascadeType.ALL,fetch
= FetchType.EAGER)
   private Set<Subjects> subjectsList;
   public int getAuId() {
      return auId;
   }
   public void setAuId(int auId) {
      this.auId = auId;
   }
   public String getAuthorName() {
      return authorName;
   }
   public void setAuthorName(String authorName) {
      this.authorName = authorName;
   }
   public int getAuthorAge() {
      return authorAge;
   }
   public void setAuthorAge(int authorAge) {
      this.authorAge = authorAge;
   }
   public Address getAddress() {
      return address;
   }
   public void setAddress(Address address) {
      this.address = address;
   }
   public Set<Subjects> getSubjectsList() {
      return subjectsList;
   }
   public void setSubjectsList(Set<Subjects> subjectsList) {
      this.subjectsList = subjectsList;
   }
   public void addSubject(Subjects subject){
```

```java
        if(subjectsList==null){
            subjectsList= new HashSet<>();
        }
        subject.setAuthor(this);
        subjectsList.add(subject);
    }
}
```

4. Persist 3 subjects for each author.

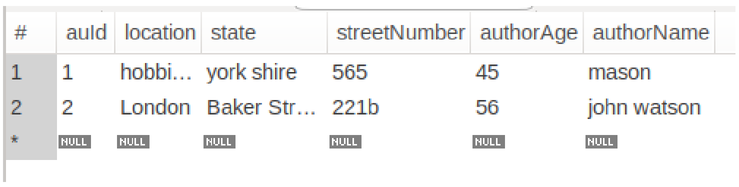
   // creating the author with 3 subjects in `Q1To4Tests` test

```java
  @SpringBootTest
class Q1To4Tests {
   @Autowired
   Author1to2Repository author1to2Repository;
   @Test
   void contextLoads() {
   }
   // Q1 to 4
   @Test // creating a new author with address as embedded object
   // Persist 3 subjects for each author.
   void testCreateAuthor1(){
      Author author=new Author();
      author.setAuthorName("mason");
      author.setAuthorAge(45);
      Address address=new Address();
      address.setStreetNumber("565");
      address.setState("york shire");
      address.setLocation("hobbiton");
      author.setAddress(address);
      Subjects sub1=new Subjects();
      sub1.setSubId(1);
      sub1.setSubjectName("friction");
      Subjects sub2=new Subjects();
      sub2.setSubId(2);
      sub2.setSubjectName("non-friction");
      Subjects sub3=new Subjects();
      sub3.setSubId(3);
      sub3.setSubjectName("thriller");
      author.addSubject(sub1);
      author.addSubject(sub2);
      author.addSubject(sub3);
      author1to2Repository.save(author);
   }
```



Result Grid | Filter Rows:

#	subId	subjectName	auId
1	1	friction	1
2	2	non-friction	1
3	3	thriller	1
4	4	node	2
5	5	java	2
6	6	python	2
*	NULL	NULL	NULL



#	auId	location	state	streetNumber	authorAge	authorName
1	1	hobbi...	york shire	565	45	mason
2	2	London	Baker Str...	221b	56	john watson
*	NULL	NULL	NULL	NULL	NULL	NULL

5. Create an Entity book with an instance variable bookName.

// **BOOK ENTITY**

```
     package com.example.exercise.q1to6.entity;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToOne;
@Entity
public class Book {
    @Id
    private int bId;
    private String BookName;
    private String PublisherName;
    @OneToOne
    @JoinColumn(name = "auId")
    Author author;
    public int getbId() {
        return bId;
    }
    public void setbId(int bId) {
        this.bId = bId;
    }
    public String getBookName() {
        return BookName;
    }
    public void setBookName(String bookName) {
        BookName = bookName;
    }
    public String getPublisherName() {
        return PublisherName;
    }
    public void setPublisherName(String publisherName) {
        PublisherName = publisherName;
    }
    public Author getAuthor() {
        return author;
    }
    public void setAuthor(Author author) {
        this.author = author;
    }
}
```

6. Implement One to One mapping between Author and Book.

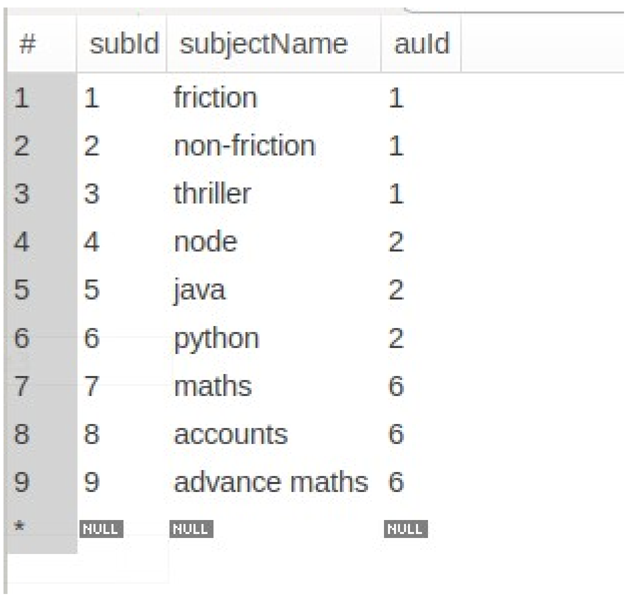
// UPDATED AUTHOR

```
package com.example.exercise.q1to6.entity;
import javax.persistence.*;
import java.util.HashSet;
import java.util.Set;
@Entity
public class Author {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private int auId;
    private String authorName;
    private int authorAge;
    @OneToOne(mappedBy = "author",cascade = CascadeType.ALL,fetch =
FetchType.EAGER)
    private Book book;
    @Embedded
    private Address address;
    @OneToMany(mappedBy = "author",cascade = CascadeType.ALL,fetch
= FetchType.EAGER)
    private Set<Subjects> subjectsList;
    public int getAuId() {
        return auId;
    }
    public void setAuId(int auId) {
        this.auId = auId;
    }
    public String getAuthorName() {
        return authorName;
    }
    public void setAuthorName(String authorName) {
        this.authorName = authorName;
    }
    public int getAuthorAge() {
        return authorAge;
    }
    public void setAuthorAge(int authorAge) {
        this.authorAge = authorAge;
    }
    public Address getAddress() {
        return address;
    }
    public void setAddress(Address address) {
        this.address = address;
    }
    public Set<Subjects> getSubjectsList() {
        return subjectsList;
    }
    public void setSubjectsList(Set<Subjects> subjectsList) {
        this.subjectsList = subjectsList;
    }
    public Book getBook() {
        return book;
    }
    public void setBook(Book book) {
        this.book = book;
    }
    public void addSubject(Subjects subject){
        if(subjectsList==null){
            subjectsList= new HashSet<>();
        }
        subject.setAuthor(this);
        subjectsList.add(subject);
    }
    public void addBook(Book book){
        this.setBook(book);
        book.setAuthor(this);
    }
}
```

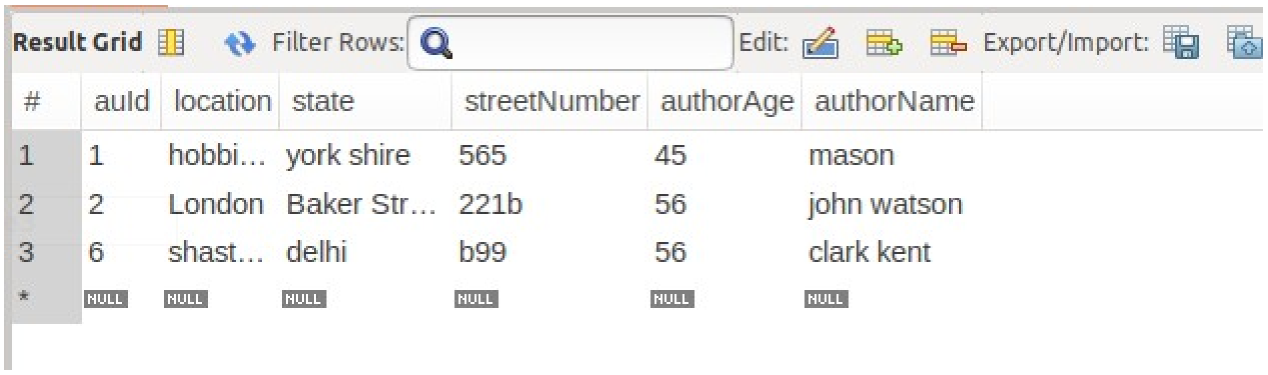
// creating the author

```
    @Test
void contextLoads() {
}
// Q1 to 4
@Test // creating a new author with address as embedded object
// Persist 3 subjects for each author.
void testCreateAuthor1(){
  Author author=new Author();
  author.setAuthorName("mason");
  author.setAuthorAge(45);
  // embedding address
  Address address=new Address();
  address.setStreetNumber("565");
  address.setState("york shire");
  address.setLocation("hobbiton");
  author.setAddress(address);
  Subjects sub1=new Subjects();
  sub1.setSubId(1);
  sub1.setSubjectName("friction");
  Subjects sub2=new Subjects();
```

```
    sub2.setSubId(2);
    sub2.setSubjectName("non-friction");
    Subjects sub3=new Subjects();
    sub3.setSubId(3);
    sub3.setSubjectName("thriller");
    // adding subject to the list
    author.addSubject(sub1);
    author.addSubject(sub2);
    author.addSubject(sub3);
    // adding book to author
    //adding book to the author
    Book b1=new Book();
    b1.setbId(12);
    b1.setBookName("rd sharma");
    b1.setPublisherName("sharma publications");
    author.addBook(b1);
    author1To6Repository.save(author);
}
```



#	subId	subjectName	auId
1	1	friction	1
2	2	non-friction	1
3	3	thriller	1
4	4	node	2
5	5	java	2
6	6	python	2
7	7	maths	6
8	8	accounts	6
9	9	advance maths	6
*	NULL	NULL	NULL



#	auId	location	state	streetNumber	authorAge	authorName
1	1	hobbi...	york shire	565	45	mason
2	2	London	Baker Str...	221b	56	john watson
3	6	shast...	delhi	b99	56	clark kent
*	NULL	NULL	NULL	NULL	NULL	NULL

7. Implement One to Many Mapping between Author and Book(Unidirectional, BiDirectional and without additional table) and implement cascade save.

1. Unidirectional

// Author.java

```
package com.example.mapping.one2many.unidirectional;
import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;
@Entity
@Table(name = "author")
public class Author {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "author_id")
    private int aId;
    @Column(name = "author_name")
    private String authorName;
    @Column(name = "author_age")
    private int authorAge;
    @OneToMany(cascade = CascadeType.ALL,fetch = FetchType.EAGER)
//    @JoinColumn(name = "a_id")
    private List<Book> bookList;
    public Author(){}
    public Author(String authorName, int authorAge, List<Book>
bookList) {
        this.authorName = authorName;
        this.authorAge = authorAge;
        this.bookList = bookList;
    }
    public int getaId() {
        return aId;
    }
    public void setaId(int aId) {
        this.aId = aId;
    }
    public String getAuthorName() {
        return authorName;
    }
    public void setAuthorName(String authorName) {
        this.authorName = authorName;
    }
    public int getAuthorAge() {
        return authorAge;
    }
    public void setAuthorAge(int authorAge) {
        this.authorAge = authorAge;
    }
    public List<Book> getBookList() {
        return bookList;
    }
    public void setBookList(List<Book> bookList) {
        this.bookList = bookList;
    }
}
```

// **Book.java**

```
package com.example.mapping.one2many.unidirectional;
import javax.persistence.*;
@Entity
@Table(name = "book")
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "book_id")
    private int id;
    @Column(name = "book_name")
    private String bookName;
    public Book(){}
    public Book(String bookName) {
        this.bookName = bookName;
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getBookName() {
        return bookName;
    }
    public void setBookName(String bookName) {
        this.bookName = bookName;
    }
}
```

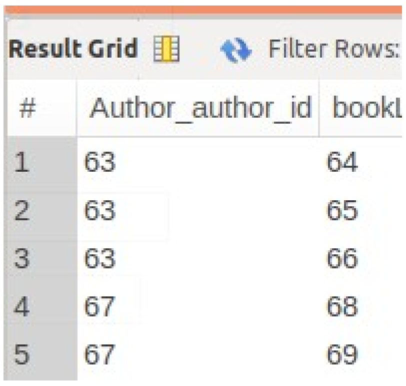
// **AuthorService.java**

```
package com.example.mapping.one2many.unidirectional;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.util.Arrays;
import java.util.List;
@Service
public class AuthorService {
    @Autowired
    UniRepository uniRepository;
    public void addAuthor(){
        List<Book> bookList= Arrays.asList(new Book("bam"),new
Book("kabam"),
                new Book("sazam"));
        Author author=new Author("yani",56,bookList);
        uniRepository.save(author);
        List<Book> bookList1= Arrays.asList(new Book("teenage"),new
Book("wasteland"),
                new Book("this is"));
        Author author1=new Author("baba o riley",76,bookList1);
        uniRepository.save(author1);
    }
}
```

// **UniRepository.java**

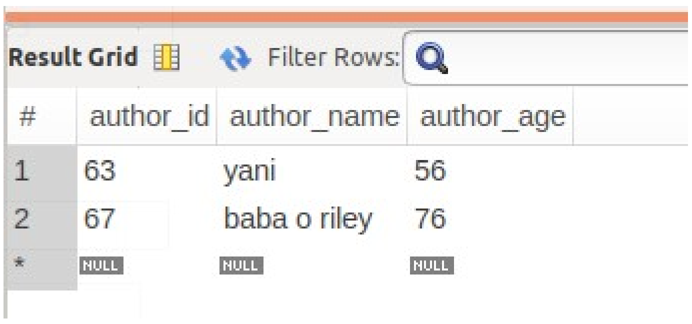
```
     package com.example.mapping.one2many.unidirectional;
import org.springframework.data.repository.CrudRepository;
public interface UniRepository extends
CrudRepository<Author,Integer> {
}
```

```
     //package com.example.mapping.one2many.unidirectional;
import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;
@SpringBootApplication
public class UnidirectionalApplication {
   //RUNNING THE SCRIPT RUN FOLOWING COMMANDS IN  DATABASE
   //BEFORE
   //SET foreign_key_checks = 0;
   //drop table author;
   //drop table book;
   //SET foreign_key_checks = 1;
   public static void main(String[] args) {
     ApplicationContext
applicationContext=SpringApplication.run(UnidirectionalApplication
.class, args);
     AuthorService
as=applicationContext.getBean(AuthorService.class);
     as.addAuthor();
   }
}
```



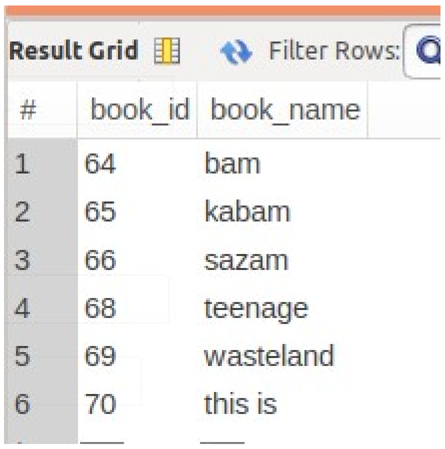
Result Grid Filter Rows:

#	Author_author_id	book
1	63	64
2	63	65
3	63	66
4	67	68
5	67	69



Result Grid Filter Rows:

#	author_id	author_name	author_age
1	63	yani	56
2	67	baba o riley	76
*	NULL	NULL	NULL



Result Grid Filter Rows:

#	book_id	book_name
1	64	bam
2	65	kabam
3	66	sazam
4	68	teenage
5	69	wasteland
6	70	this is

2. Bidirectional without additional table

// **Author.java**

```
package com.example.mapping.one2many.bidirectional;
import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;
@Entity
@Table(name = "author")
public class Author {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   @Column(name = "author_id")
   private int aId;
   @Column(name = "author_name")
   private String authorName;
   @Column(name = "author_age")
   private int authorAge;
   @OneToMany(mappedBy = "author",cascade = CascadeType.ALL,fetch
= FetchType.EAGER)
   private List<Book> bookList;
   public int getaId() {
      return aId;
   }
   public void setaId(int aId) {
      this.aId = aId;
   }
   public String getAuthorName() {
      return authorName;
   }
   public void setAuthorName(String authorName) {
      this.authorName = authorName;
   }
   public int getAuthorAge() {
      return authorAge;
   }
   public void setAuthorAge(int authorAge) {
      this.authorAge = authorAge;
   }
   public List<Book> getBookList() {
      return bookList;
   }
   public void setBookList(List<Book> bookList) {
      this.bookList = bookList;
   }
   public void addBook(Book book){
    if(bookList==null){
       bookList=new ArrayList<Book>();
    }
```

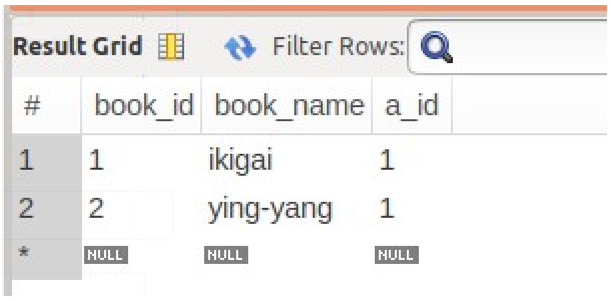
```
        book.setAuthor(this);
        bookList.add(book);
    }
}
```

// **Book.java**

```
package com.example.mapping.one2many.bidirectional;
import com.example.mapping.one2many.bidirectional.Author;
import javax.persistence.*;
@Entity
@Table(name = "book")
public class Book {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   @Column(name = "book_id")
   private int id;
   @Column(name = "book_name")
   private String bookName;
   @ManyToOne(cascade = CascadeType.ALL)
   @JoinColumn(name = "a_id")
   private Author author;
   public int getId() {
      return id;
   }
   public void setId(int id) {
      this.id = id;
   }
   public String getBookName() {
      return bookName;
   }
   public void setBookName(String bookName) {
      this.bookName = bookName;
   }
   public Author getAuthor() {
      return author;
   }
   public void setAuthor(Author author) {
      this.author = author;
   }
}
```

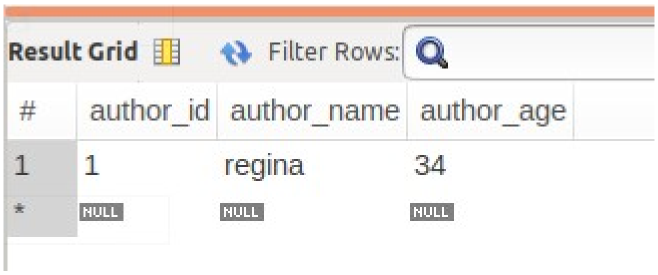
// **AuthorService.java**

```
package com.example.mapping.one2many.bidirectional;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.SpringApplication;
import org.springframework.stereotype.Service;
@Service
public class AuthorService {
    @Autowired
    BiRepository biRepository;
    public void createAuthor(){
        Author author = new Author();
        author.setAuthorName("regina");
        author.setAuthorAge(34);
        Book book1 = new Book();
        book1.setBookName("ikigai");
        author.addBook(book1);
        Book book2 = new Book();
        book2.setBookName("ying-yang");
        author.addBook(book2);
        biRepository.save(author);
    }
}
```



Result Grid Filter Rows:

#	book_id	book_name	a_id
1	1	ikigai	1
2	2	ying-yang	1
*	NULL	NULL	NULL



Result Grid Filter Rows:

#	author_id	author_name	author_age
1	1	regina	34
*	NULL	NULL	NULL

8. Implement Many to Many Mapping between Author and Book.

// Author.java

```
package com.example.mapping.many2many;
import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;
@Entity
@Table(name = "author")
public class Author {
    @Id
    @Column(name = "author_id")
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int aId;
    @Column(name = "author_name")
    private String authorName;
    @Column(name = "author_age")
    private int authorAge;
```

```
  @ManyToMany(cascade = CascadeType.ALL)
   @JoinTable(name = "author_booklist",
          joinColumns = @JoinColumn(name =
"a_id",referencedColumnName = "author_id"),
          inverseJoinColumns = @JoinColumn(name =
"b_id",referencedColumnName = "book_id"))
   private List<Book> bookList;
   public Author() {
   }
   public int getaId() {
      return aId;
   }
   public void setaId(int aId) {
      this.aId = aId;
   }
   public String getAuthorName() {
      return authorName;
   }
   public void setAuthorName(String authorName) {
      this.authorName = authorName;
   }
   public int getAuthorAge() {
      return authorAge;
   }
   public void setAuthorAge(int authorAge) {
      this.authorAge = authorAge;
   }
   public List<Book> getBookList() {
      return bookList;
   }
   public void setBookList(List<Book> bookList) {
      this.bookList = bookList;
   }
   public void addBook(Book book){
      if(bookList==null){
         bookList=new ArrayList<>();
      }
      bookList.add(book);
      book.getAuthors().add(this);
   }
}
```

// BOOKS.JAVA

```
package com.example.mapping.many2many;
import javax.persistence.*;
import java.util.ArrayList;
import java.util.List;
```

```
@Entity
@Table(name = "book")
public class Book {
   @Id
   @GeneratedValue(strategy = GenerationType.IDENTITY)
   @Column(name = "book_id")
   private int id;
   @Column(name = "book_name")
   private String bookName;
   @ManyToMany(mappedBy = "bookList")
   private List<Author> authors;
   public Book() {
   }
   public int getId() {
      return id;
   }
   public void setId(int id) {
      this.id = id;
   }
   public String getBookName() {
      return bookName;
   }
   public void setBookName(String bookName) {
      this.bookName = bookName;
   }
   public List<Author> getAuthors() {
      return authors;
   }
   public void setAuthors(List<Author> authors) {
      this.authors = authors;
   }
}
```

// ManyRepository.java

```
package com.example.mapping.many2many;

import org.springframework.data.repository.CrudRepository;
import org.springframework.stereotype.Repository;
@Repository
public interface ManyRepository extends
CrudRepository<Author,Integer> {
}
```

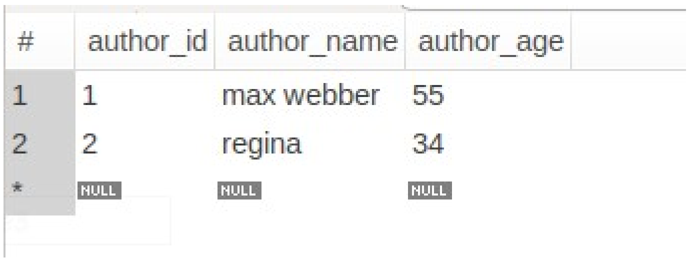
// AuthorService.java

```java
package com.example.mapping.many2many;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.example.mapping.many2many.Author;
import java.util.*;
@Service
public class AuthorService {
   @Autowired
   ManyRepository manyRepository;
   public void insertAuthor(){
      Book b1=new Book();
      b1.setBookName("alpha");
      Book b2=new Book();
      b2.setBookName("beta");
      Book b3=new Book();
      b3.setBookName("omega");
      Author author=new Author();
      author.setAuthorName("hoax");
      author.setAuthorAge(23);
      author.setBookList(Arrays.asList(b1,b3));
      Author author1=new Author();
      author1.setAuthorName("xavier");
      author1.setAuthorAge(23);
      author1.setBookList(Arrays.asList(b2,b3));
      manyRepository.saveAll(Arrays.asList(author,author1));
   }
}
```

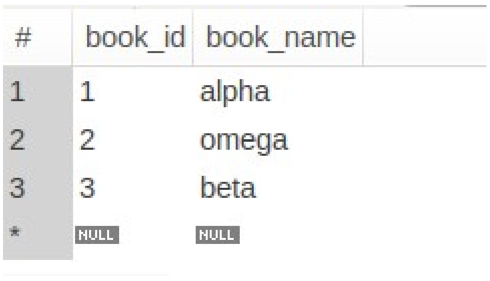
// Many2ManyApplication.java

```java
package com.example.mapping.many2many;
import org.springframework.boot.SpringApplication;
import
org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.ApplicationContext;
@SpringBootApplication
public class Many2ManyApplication {
  public static void main(String[] args) {
     ApplicationContext
applicationContext=SpringApplication.run(Many2ManyApplication.clas
s, args);
     AuthorService
as=applicationContext.getBean(AuthorService.class);
```

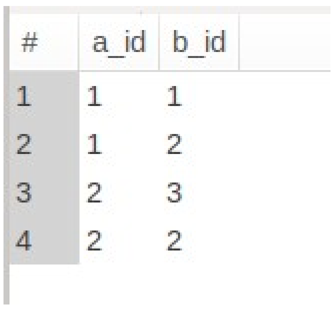
```
            as.insertAuthor();
        }
    }
```



#	author_id	author_name	author_age
1	1	max webber	55
2	2	regina	34
*	NULL	NULL	NULL



#	book_id	book_name
1	1	alpha
2	2	omega
3	3	beta
*	NULL	NULL



#	a_id	b_id
1	1	1
2	1	2
3	2	3
4	2	2