```
==========================pom.xml================================
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
   https://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>
   <parent>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-parent</artifactId>
      <version>2.3.0.RELEASE</version>
      <relativePath/> <!-- lookup parent from repository -->
   </parent>
   <groupId>com.ameya</groupId>
   <artifactId>004-UserInfo-api-JPA-Hibernate</artifactId>
   <version>0.0.1-SNAPSHOT</version>
   <name>004-UserInfo-api-JPA-Hibernate</name>
   <description>Demo project for Spring Boot</description>
   <properties>
      <java.version>1.8</java.version>
   </properties>
   <dependencies>
      <dependency>
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-starter-data-jpa</artifactId>
      </dependency>
      <dependency>
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-starter-web</artifactId>
      </dependency>

      <dependency>
         <groupId>mysql</groupId>
         <artifactId>mysql-connector-java</artifactId>
         <scope>runtime</scope>
      </dependency>
      <dependency>
         <groupId>org.springframework.boot</groupId>
         <artifactId>spring-boot-starter-test</artifactId>
         <scope>test</scope>
         <exclusions>
            <exclusion>
               <groupId>org.junit.vintage</groupId>
               <artifactId>junit-vintage-engine</artifactId>
            </exclusion>
         </exclusions>
      </dependency>
   </dependencies>
```

```

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
================================application.properties===========================
server.port=9001
logging.level.web=trace

# Database

db.url: jdbc:mysql://localhost:3306/sapientdb
db.driver: com.mysql.cj.jdbc.Driver
db.username: root
db.password: root

# Hibernate

hibernate.dialect: org.hibernate.dialect.MySQL5Dialect
hibernate.show_sql: true
hibernate.hbm2ddl.auto: update
entitymanager.pkgsScan: com

================================HibernateConfiguration.java==========================
package com.ameya.config;

import java.util.Properties;

import javax.sql.DataSource;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.jdbc.datasource.DriverManagerDataSource;
import org.springframework.orm.hibernate5.HibernateTransactionManager;
import org.springframework.orm.hibernate5.LocalSessionFactoryBean;
import org.springframework.transaction.annotation.EnableTransactionManagement;

@Configuration
@EnableTransactionManagement
```

```java
public class HibernateConfiguration {

    @Value("${db.driver}")
    private String DB_DRIVER;
    @Value("${db.password}")
    private String DB_PASSWORD;
    @Value("${db.url}")
    private String DB_URL;
    @Value("${db.username}")
    private String DB_USERNAME;
    @Value("${hibernate.dialect}")
    private String HIBERNATE_DIALECT;
    @Value("${hibernate.show_sql}")
    private String HIBERNATE_SHOW_SQL;
    @Value("${hibernate.hbm2ddl.auto}")
    private String HBM2DDL_AUTO;
    @Value("${entitymanager.pkgsScan}")
    private String PACKAGES_TO_SCAN;

    @Bean
    public DataSource dataSource() {
        DriverManagerDataSource dataSource=new DriverManagerDataSource();
        dataSource.setDriverClassName(DB_DRIVER);
        dataSource.setUrl(DB_URL);
        dataSource.setUsername(DB_USERNAME);
        dataSource.setPassword(DB_PASSWORD);
        return dataSource;
    }
    @Bean
    public LocalSessionFactoryBean sessionFactory() {
        LocalSessionFactoryBean sessionFactory=new LocalSessionFactoryBean();
        sessionFactory.setDataSource(dataSource());
        sessionFactory.setPackagesToScan(PACKAGES_TO_SCAN);
        Properties hibernateProps=new Properties();
        hibernateProps.put("hibernate.dialect", HIBERNATE_DIALECT);
        hibernateProps.put("hibernate.show_sql", HIBERNATE_SHOW_SQL);
        hibernateProps.put("hibernate.hbm2ddl.auto", HBM2DDL_AUTO);
        sessionFactory.setHibernateProperties(hibernateProps);
        return sessionFactory;
    }
    @Bean
    public HibernateTransactionManager transactionManager() {
        HibernateTransactionManager txManager=new HibernateTransactionManager();
        txManager.setSessionFactory(sessionFactory().getObject());
        return txManager;
    }

}
```

```
========================================Application.java============================
package com.ameya;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.orm.jpa.HibernateJpaAutoConfiguration;

@SpringBootApplication(exclude = HibernateJpaAutoConfiguration.class)
public class Application {

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

}
===================================UserInfo.java======================================
package com.ameya.models;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.Table;

import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

@Entity
@Table(name="aj_userinfo")
@JsonIgnoreProperties({"hibernateLazyInitializer","handler"})
public class UserInfo {

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private int id;
    @Column(name="fullname")
    private String fullName;
    @Column(name="country")
    private String country;
    public UserInfo() {

    }
    public UserInfo(int id, String fullName, String country) {
        super();
        this.id = id;
        this.fullName = fullName;
        this.country = country;
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getFullName() {
        return fullName;
    }
    public void setFullName(String fullName) {
        this.fullName = fullName;
    }
    public String getCountry() {
        return country;
    }
    public void setCountry(String country) {
        this.country = country;
    }
    @Override
    public String toString() {
        return "UserInfo [id=" + id + ", fullName=" + fullName + ", country=" +
        country + "]";
    }

}
============================================UserInfoDAO.java=============
=======================
package com.ameya.daos;

import java.util.List;

import com.ameya.models.UserInfo;

public interface UserInfoDAO {
    void addUser(UserInfo userInfo);
    List<UserInfo>getAllUserInfo();
    UserInfo findById(int id);
    UserInfo findByIdQuery(int id);
    UserInfo update(UserInfo userInfo,int id);
    void delete(int id);

}
========================================UserInfoDAOImpl.java==========
==================
package com.ameya.daos.impl;

```

```java
import java.util.List;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.query.Query;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

import com.ameya.daos.UserInfoDAO;
import com.ameya.models.UserInfo;

@Repository
public class UserInfoDAOImpl implements UserInfoDAO {

    @Autowired
    private SessionFactory sessionFactory;

    @Override
    public void addUser(UserInfo userInfo) {
        Session session=sessionFactory.getCurrentSession();
        session.save(userInfo);

    }

    @Override
    public List<UserInfo> getAllUserInfo() {
        Session session=sessionFactory.getCurrentSession();
        List<UserInfo> list=session.createQuery("from UserInfo").list();
        return list;
    }

    @Override
    public UserInfo findById(int id) {
        Session session=sessionFactory.getCurrentSession();
        UserInfo userInfo=(UserInfo)session.get(UserInfo.class, id);
        return userInfo;
    }

    @Override
    public UserInfo findByIdQuery(int id) {
        Session session=sessionFactory.getCurrentSession();
        Query<UserInfo> query=session.createQuery("from UserInfo where id = :id");
        query.setParameter("id", id);
        List<UserInfo> users=query.getResultList();
        Query<Integer> cntQuery=session.createQuery("select count(id) from
        UserInfo");
        List<Integer> cntList=cntQuery.getResultList();
        System.out.println("COUNT => "+cntList.get(0));
```

```
            return users.get(0);
    }

    @Override
    public UserInfo update(UserInfo userInfo, int id) {
            Session session=sessionFactory.getCurrentSession();
            UserInfo existingUserInfo=(UserInfo)session.load(UserInfo.class, id);
            existingUserInfo.setFullName(userInfo.getFullName());
            existingUserInfo.setCountry(userInfo.getCountry());
            return existingUserInfo;
    }

    @Override
    public void delete(int id) {
            Session session=sessionFactory.getCurrentSession();
            UserInfo userInfo=findById(id);
            session.delete(userInfo);

    }

}

/*
 Query<Emp> query=session.createQuery("from Emp");//here persistent class name is
 Emp
List list=query.list();

Query<Emp> query=session.createQuery("from Emp");
query.setFirstResult(5);
query.setMaxResult(10);
List list=query.list();//will return the records from 5 to 10th number

Query<Integer> q=session.createQuery("update User set name=:n where id=:i");
q.setParameter("n","Ameya Joshi");
q.setParameter("i",111);

int status=q.executeUpdate();

Query<Integer> query=session.createQuery("delete from Emp where id=100");
query.executeUpdate();

Query<Integer> q=session.createQuery("select sum(salary) from Emp");
List<Integer> list=q.list();

Query<Integer> q=session.createQuery("select max(salary) from Emp");

Query<Integer> q=session.createQuery("select min(salary) from Emp");

```

```
Query<Integer> q=session.createQuery("select count(id) from Emp");
List<Integer> cntList=q.getResultList();
System.out.println("COUNT =====> "+cntList.get(0));

 */
=============================================UserInfoService.java=========
======================
package com.ameya.services;

import java.util.List;

import com.ameya.models.UserInfo;

public interface UserInfoService {
    void createUser(UserInfo userInfo);
    List<UserInfo> findAll();
    UserInfo findById(int id);
    UserInfo findByIdQuery(int id);
    UserInfo updateUserInfo(UserInfo userInfo,int id);
    void deleteById(int id);
}
============================================UserInfoServiceImpl.java=======
===================
package com.ameya.services.impl;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import com.ameya.daos.UserInfoDAO;
import com.ameya.models.UserInfo;
import com.ameya.services.UserInfoService;

@Service
@Transactional
public class UserInfoServiceImpl implements UserInfoService {

    @Autowired
    private UserInfoDAO userInfoDao;

    @Override
    public void createUser(UserInfo userInfo) {
        userInfoDao.addUser(userInfo);

    }

```

```java
    @Override
    public List<UserInfo> findAll() {
        return userInfoDao.getAllUserInfo();
    }

    @Override
    public UserInfo findById(int id) {
        return userInfoDao.findById(id);
    }

    @Override
    public UserInfo findByIdQuery(int id) {
        return userInfoDao.findByIdQuery(id);
    }

    @Override
    public UserInfo updateUserInfo(UserInfo userInfo, int id) {
        return userInfoDao.update(userInfo, id);
    }

    @Override
    public void deleteById(int id) {
        userInfoDao.delete(id);

    }

}
===================================UserInfoController.java=============================
package com.ameya.controllers;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpStatus;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.util.UriComponentsBuilder;

import com.ameya.models.UserInfo;
import com.ameya.services.UserInfoService;

@RestController
@RequestMapping("/userinfo")
public class UserInfoController {
    @Autowired
    private UserInfoService userInfoService;

    @GetMapping(path="/{id}",produces=MediaType.APPLICATION_JSON_VALUE)
    public ResponseEntity<UserInfo> getUserById(@PathVariable("id") int id){
        UserInfo userInfo=userInfoService.findById(id);
        if(userInfo==null) {
            return new ResponseEntity<UserInfo>(HttpStatus.NOT_FOUND);
        }
        return new ResponseEntity<UserInfo>(userInfo,HttpStatus.OK);
    }


    @GetMapping(path="/byquery/{id}",produces=MediaType.APPLICATION_JSON_VAL
    UE)
    public ResponseEntity<UserInfo> getUserByIdQuery(@PathVariable("id") int id){
        UserInfo userInfo=userInfoService.findById(id);
        if(userInfo==null) {
            return new ResponseEntity<UserInfo>(HttpStatus.NOT_FOUND);
        }
        return new ResponseEntity<UserInfo>(userInfo,HttpStatus.OK);
    }

    @PostMapping(path="/create",headers="Accept=application/json")
    public ResponseEntity<Void> createUser(@RequestBody UserInfo userInfo,
            UriComponentsBuilder builder){
        userInfoService.createUser(userInfo);
        HttpHeaders headers=new HttpHeaders();

        headers.setLocation(builder.path("/userinfo/{id}").buildAndExpand(userInfo.getId
        ()).toUri());
        return new ResponseEntity<Void>(headers,HttpStatus.CREATED);
    }

    @GetMapping("/getall")
    public List<UserInfo> getAllUserInfo(){
        return userInfoService.findAll();
    }

    @PutMapping(path="/update/{id}")
    public ResponseEntity<String> updateUserInfo(@RequestBody UserInfo
    currentUserInfo,@PathVariable("id") int id){
        UserInfo userInfo=userInfoService.findById(id);
        if(userInfo==null) {
            return  new ResponseEntity<String>("User Not
            Found",HttpStatus.NOT_FOUND);
        }
        userInfoService.updateUserInfo(currentUserInfo, id);
        return  new ResponseEntity<String>("User Updated",HttpStatus.OK);
    }

    @DeleteMapping(path="/delete/{id}")
    public ResponseEntity<String> deleteUserInfo(@PathVariable("id") int id){
        UserInfo userInfo=userInfoService.findById(id);
        if(userInfo==null) {
            return  new ResponseEntity<String>("User Not
            Found",HttpStatus.NOT_FOUND);
        }
        userInfoService.deleteById(id);
        return  new ResponseEntity<String>("User Deleted",HttpStatus.NO_CONTENT);
    }
}
=====================================================


http://localhost:9001/userinfo/create
{
    "fullName": "Ameya Joshi",
    "country": "Germany"
}
http://localhost:9001/userinfo/delete/4
http://localhost:9001/userinfo/update/3
http://localhost:9001/userinfo/3
http://localhost:9001/userinfo/byquery/3
http://localhost:9001/userinfo/getall

+++++++++++++++++++++++++++++++++++++++++++++RelationShips
++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++++++++

=====================================================Employee.java===========
================
package com.ameya.models;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToOne;
import javax.persistence.PrimaryKeyJoinColumn;
import javax.persistence.Table;

@Entity
@Table(name="aj_employee")
public class Employee {

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name="id")
    private int id;
    @Column(name="name")
    private String name;
    @OneToOne(targetEntity=Address.class, cascade = CascadeType.ALL)
    private Address address;
    public Employee() {
        super();
        // TODO Auto-generated constructor stub
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public Address getAddress() {
        //address.setEmpId(getId());
        return address;
    }
    public void setAddress(Address address) {
        this.address = address;
    }

}

========================================Address.java=====================
=============
package com.ameya.models;

import javax.persistence.Entity;
```

```java
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToOne;
import javax.persistence.PrimaryKeyJoinColumn;
import javax.persistence.Table;

import org.hibernate.annotations.GenericGenerator;
import org.hibernate.annotations.Parameter;

@Entity
@Table(name="aj_address")
public class Address {

    @Id
    @GeneratedValue(generator="gen")
    @GenericGenerator(name="gen", strategy="foreign",
    parameters=@Parameter(name="property",value="employee"))
    private int id;

    @OneToOne(targetEntity = Employee.class)
    @PrimaryKeyJoinColumn
    private Employee employee;
    private String address;
    private String country;
    public Address() {
        super();
        // TODO Auto-generated constructor stub
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }


    public Employee getEmployee() {
        return employee;
    }
    public void setEmployee(Employee employee) {
        this.employee = employee;
    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
    public String getCountry() {
        return country;
    }
    public void setCountry(String country) {
        this.country = country;
    }

}
===================================== + Tables Created as below + 
=========================================================
CREATE TABLE `aj_employee` (
    `id` INT(11) NOT NULL,
    `name` VARCHAR(255) NULL DEFAULT NULL COLLATE 'latin1_swedish_ci',
    `address_id` INT(11) NULL DEFAULT NULL,
    PRIMARY KEY (`id`) USING BTREE,
    INDEX `FKey4gc58msqeklskis35o7f81w` (`address_id`) USING BTREE,
    CONSTRAINT `FKey4gc58msqeklskis35o7f81w` FOREIGN KEY (`address_id`) 
    REFERENCES `sapientdb`.`aj_address` (`id`) ON UPDATE RESTRICT ON 
    DELETE RESTRICT
)
COLLATE='latin1_swedish_ci'
ENGINE=InnoDB
;


CREATE TABLE `aj_address` (
    `id` INT(11) NOT NULL,
    `address` VARCHAR(255) NULL DEFAULT NULL COLLATE 'latin1_swedish_ci',
    `country` VARCHAR(255) NULL DEFAULT NULL COLLATE 'latin1_swedish_ci',
    PRIMARY KEY (`id`) USING BTREE
)
COLLATE='latin1_swedish_ci'
ENGINE=InnoDB
;
==========================================================================
====================================================

=================================================Group.java===============
==
package com.ameya.models;

import java.util.List;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToMany;
import javax.persistence.OrderColumn;
import javax.persistence.Table;

@Entity
@Table(name="aj_group")
public class Group {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private int id;
    private String name;
    @OneToMany(cascade = CascadeType.ALL,fetch = FetchType.LAZY)
    @OrderColumn(name="listIdx")
    @JoinColumn(name="gr_id")
    private List<Story> stories;
    public Group() {}
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public List<Story> getStories() {
        return stories;
    }
    public void setStories(List<Story> stories) {
        this.stories = stories;
    }
}
==============================Story.java============================
package com.ameya.models;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.Table;
```

```

@Entity
@Table(name="aj_story")
public class Story {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private int id;
    private String info;
    @ManyToOne
    @JoinColumn(name="gr_id")
    private Group group;
    public Story() {}
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getInfo() {
        return info;
    }
    public void setInfo(String info) {
        this.info = info;
    }
    public Group getGroup() {
        return group;
    }
    public void setGroup(Group group) {
        this.group = group;
    }
}
============================================+ Tables created as below +======================================
CREATE TABLE `aj_group` (
    `id` INT(11) NOT NULL,
    `name` VARCHAR(255) NULL DEFAULT NULL COLLATE 'latin1_swedish_ci',
    PRIMARY KEY (`id`) USING BTREE
)
COLLATE='latin1_swedish_ci'
ENGINE=InnoDB
;

CREATE TABLE `aj_story` (
    `id` INT(11) NOT NULL,
    `info` VARCHAR(255) NULL DEFAULT NULL COLLATE 'latin1_swedish_ci',
    `gr_id` INT(11) NULL DEFAULT NULL,
    `listIdx` INT(11) NULL DEFAULT NULL,
    PRIMARY KEY (`id`) USING BTREE,
    INDEX `FKmjkud1gmd4b6d02xnskmx2xwc` (`gr_id`) USING BTREE,
    CONSTRAINT `FKmjkud1gmd4b6d02xnskmx2xwc` FOREIGN KEY (`gr_id`) REFERENCES `sapientdb`.`aj_group` (`id`) ON UPDATE RESTRICT ON DELETE RESTRICT
)
COLLATE='latin1_swedish_ci'
ENGINE=InnoDB
;
===========================================================================================================
=======================================Author.java===============================
package com.ameya.models;

import java.util.Set;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.Table;

@Entity
@Table(name="aj_authors")
public class Author {
    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    private int id;
    private String authorName;
    @ManyToMany(cascade=CascadeType.ALL)
    @JoinTable(
            name="aj_author_book",
            joinColumns= {@JoinColumn(name="author_id")},
            inverseJoinColumns= {@JoinColumn(name="book_id")}
            )
    private Set<Book> books;
    public Author() {}
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getAuthorName() {
        return authorName;
    }
    public void setAuthorName(String authorName) {
        this.authorName = authorName;
    }
    public Set<Book> getBooks() {
        return books;
    }
    public void setBooks(Set<Book> books) {
        this.books = books;
    }
}
====================================Book.java========================
package com.ameya.models;

import java.util.Set;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.Table;

@Entity
@Table(name="aj_books")
public class Book {
    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private int id;
    private String bookName;
    @ManyToMany(cascade = CascadeType.ALL)
    @JoinTable(
            name="aj_author_book",
            joinColumns = {@JoinColumn(name="book_id")},
            inverseJoinColumns = {@JoinColumn(name="author_id")}
            )
    private Set<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 Set<Author> getAuthors() {
        return authors;
    }
    public void setAuthors(Set<Author> authors) {
        this.authors = authors;
    }
}
==========================================+ Tables created as below
+==========================================
CREATE TABLE `aj_authors` (
    `id` INT(11) NOT NULL,
    `authorName` VARCHAR(255) NULL DEFAULT NULL COLLATE 'latin1_swedish_ci',
    PRIMARY KEY (`id`) USING BTREE
)
COLLATE='latin1_swedish_ci'
ENGINE=InnoDB
;

CREATE TABLE `aj_books` (
    `id` INT(11) NOT NULL,
    `bookName` VARCHAR(255) NULL DEFAULT NULL COLLATE 'latin1_swedish_ci',
    PRIMARY KEY (`id`) USING BTREE
)
COLLATE='latin1_swedish_ci'
ENGINE=InnoDB
;

CREATE TABLE `aj_author_book` (
    `book_id` INT(11) NOT NULL,
    `author_id` INT(11) NOT NULL,
    PRIMARY KEY (`author_id`, `book_id`) USING BTREE,
    INDEX `FK2yv8pkohjaj7gijby3mnvup2p` (`book_id`) USING BTREE,
    CONSTRAINT `FK2yv8pkohjaj7gijby3mnvup2p` FOREIGN KEY (`book_id`)
    REFERENCES `sapientdb`.`aj_books` (`id`) ON UPDATE RESTRICT ON DELETE
    RESTRICT,
    CONSTRAINT `FKafyj8djplfr8j5g2wkolllcge` FOREIGN KEY (`author_id`)
    REFERENCES `sapientdb`.`aj_authors` (`id`) ON UPDATE RESTRICT ON
    DELETE RESTRICT
)
COLLATE='latin1_swedish_ci'
ENGINE=InnoDB
;
```

```
=====================================RelationShipDAOImpl.java==========
=====================
package com.ameya.daos.impl;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.Transaction;
import org.hibernate.query.Query;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

import com.ameya.models.Address;
import com.ameya.models.Author;
import com.ameya.models.Book;
import com.ameya.models.Employee;
import com.ameya.models.Group;
import com.ameya.models.Story;

@Repository
public class RelationShipDAOImpl {

    @Autowired
    private SessionFactory sessionFactory;
    public void testOneToOne() {

        Session sess=sessionFactory.openSession();
        Transaction tr=sess.beginTransaction();
        Employee emp=null;
        Address a1=null;
        emp=new Employee();
        emp.setName("Ameya Joshi");
        a1=new Address();
        a1.setAddress("Kothrud, Pune");
        a1.setCountry("India");
        emp.setAddress(a1);
        a1.setEmployee(emp);
        sess.save(emp);
        sess.flush();
        Query<Employee> query=sess.createQuery("from Employee");
        List<Employee> list=query.list();
        for(Employee e : list) {
            System.out.println(e.getId()+" : "+e.getName()+" :
                "+e.getAddress().getAddress()+" : "+e.getAddress().getCountry());
        }
        Query<Address> query1=sess.createQuery("from Address");
        List<Address> list1=query1.list();
        for(Address e : list1) {
            System.out.println(e.getAddress()+" : "+e.getCountry()+" :
            "+e.getEmployee().getId()+" : "+e.getEmployee().getName());
        }
        tr.commit();
        sess.close();
    }
    public void testOneToMany() {
        Session sess=sessionFactory.openSession();
        Transaction tr=sess.beginTransaction();
        Group group=new Group();
        group.setName("SPORTS");
        ArrayList<Story> stories=new ArrayList<>();

        Story s1=new Story();
        s1.setInfo("The Allegations- Life Of a Player");
        stories.add(s1);
        Story s2=new Story();
        s2.setInfo("The Cancer Of match Fixing");
        stories.add(s2);
        Story s3=new Story();
        s3.setInfo("The Master Blaster - Sachin");
        stories.add(s3);

        group.setStories(stories);

        Serializable id=sess.save(group);
        sess.flush();
        Group g=(Group) sess.load(Group.class, id);
        System.out.println("GROUP ID :: "+g.getId()+" GROUP NAME ::
        "+g.getName());
        List<Story> groupStories=g.getStories();
        System.out.println("STORIES :: ");
        for(Story story : groupStories) {
            System.out.println(story.getId()+" : "+story.getInfo());
        }
        tr.commit();
        sess.close();
    }
    public void testManyToOne() {
        Session sess=sessionFactory.openSession();
        Story story =(Story)sess.load(Story.class, 5);
        System.out.println(story.getId()+" : "+story.getInfo());
        Group group=story.getGroup();
```

```java
        System.out.println(group.getId()+" : "+group.getName());
        sess.close();
    }
    public void testManyToMany() {
        Session sess=sessionFactory.openSession();
        Transaction tr=sess.beginTransaction();

        Author a1=new Author();
        a1.setAuthorName("Sachin");

        Book b1=new Book();
        b1.setBookName("My First Pakistan Tour");
        Book b2=new Book();
        b2.setBookName("My Cricket - My Life");
        Book b3=new Book();
        b3.setBookName("The Unforgettable Don - Don Bradman");
        Book b4=new Book();
        b4.setBookName("WC-2011 The Fantastic Experience");
        Set<Book> books=new HashSet<>();
        books.add(b1);
        books.add(b2);
        books.add(b3);
        books.add(b4);

        a1.setBooks(books);

        Serializable authorId=sess.save(a1);
        sess.flush();

        Book b5=new Book();
        b5.setBookName("The Wonderful Experiences From Cricketing Life");

        Set<Author> authors=new HashSet<>();
        Author a2=new Author();
        a2.setAuthorName("Viru");
        Author a3=new Author();
        a3.setAuthorName("Zaheer");
        Author a4=new Author();
        a4.setAuthorName("Sunny G");
        authors.add(a2);
        authors.add(a3);
        authors.add(a4);
        authors.add(a1);
        b5.setAuthors(authors);

        Serializable bookId=sess.save(b5);

        Author a=(Author)sess.get(Author.class, authorId);
        System.out.println(a.getId()+" : "+a.getAuthorName());
        Set<Book> bookSet=a.getBooks();
        for(Book b : bookSet) {
            System.out.println(b.getId()+" : "+b.getBookName());
        }
        System.out.println("+++++++++++++++++++++");

        Book bk=(Book)sess.get(Book.class,bookId);
        System.out.println(bk.getId()+" : "+bk.getBookName());
        Set<Author> authorSet=bk.getAuthors();
        for(Author at : authorSet) {
            System.out.println(at.getId()+" : "+at.getAuthorName());
        }

        tr.commit();
        sess.close();
    }
}
==========================================RelationShipServiceImpl.java=========================
package com.ameya.services.impl;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.ameya.daos.impl.RelationShipDAOImpl;

@Service
public class RelationShipServiceImpl {

    @Autowired
    private RelationShipDAOImpl dao;

    public void testOneToOne() {
        dao.testOneToOne();
    }
    public void testOneToMany() {
        dao.testOneToMany();
    }
    public void testManyToOne() {
        dao.testManyToOne();
    }
    public void testManyToMany() {
        dao.testManyToMany();
    }
}
=====================================RelationShipController.java===========================
package com.ameya.controllers;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.ameya.services.impl.RelationShipServiceImpl;

@RestController
@RequestMapping("/relations")
public class RelationShipController {

    @Autowired
    private RelationShipServiceImpl service;
    @GetMapping(path="/onetoone")
    public void testOneToOne() {
        service.testOneToOne();
    }
    @GetMapping(path="/onetomany")
    public void testOneToMany() {
        service.testOneToMany();
    }
    @GetMapping(path="/manytoone")
    public void testManyToOne() {
        service.testManyToOne();
    }
    @GetMapping(path="/manytomany")
    public void testManyToMany() {
        service.testManyToMany();
    }
}
```

==

http://localhost:9001/relations/manytomany