```
==========================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/nineleapsdb
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
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;

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());
```

```
288             return existingUserInfo;
289         }
290 
291         @Override
292         public void delete(int id) {
293             Session session=sessionFactory.getCurrentSession();
294             UserInfo userInfo=findById(id);
295             session.delete(userInfo);
296 
297         }
298 
299     }
300 
301     /*
302      Query<Emp> query=session.createQuery("from Emp");//here persistent class name is
       Emp
303     List list=query.list();
304 
305     Query<Emp> query=session.createQuery("from Emp");
306     query.setFirstResult(5);
307     query.setMaxResult(10);
308     List list=query.list();//will return the records from 5 to 10th number
309 
310     Query<Integer> q=session.createQuery("update User set name=:n where id=:i");
311     q.setParameter("n","Udit Kumar");
312     q.setParameter("i",111);
313 
314     int status=q.executeUpdate();
315 
316     Query<Integer> query=session.createQuery("delete from Emp where id=100");
317     query.executeUpdate();
318 
319     Query<Integer> q=session.createQuery("select sum(salary) from Emp");
320     List<Integer> list=q.list();
321 
322     Query<Integer> q=session.createQuery("select max(salary) from Emp");
323 
324     Query<Integer> q=session.createQuery("select min(salary) from Emp");
325 
326     Query<Integer> q=session.createQuery("select count(id) from Emp");
327     List<Integer> cntList=q.getResultList();
328     System.out.println("COUNT =====> "+cntList.get(0));
329 
330      */
331     ==========================================UserInfoService.java==================
       ===========
332     package com.ameya.services;
333 
334     import java.util.List;
335 
336     import com.ameya.models.UserInfo;
337 
338     public interface UserInfoService {
339         void createUser(UserInfo userInfo);
340         List<UserInfo> findAll();
341         UserInfo findById(int id);
342         UserInfo findByIdQuery(int id);
343         UserInfo updateUserInfo(UserInfo userInfo,int id);
344         void deleteById(int id);
345     }
346     ==========================================UserInfoServiceImpl.java================
       =======
347     package com.ameya.services.impl;
348 
349     import java.util.List;
350 
351     import org.springframework.beans.factory.annotation.Autowired;
352     import org.springframework.stereotype.Service;
353     import org.springframework.transaction.annotation.Transactional;
354 
355     import com.ameya.daos.UserInfoDAO;
356     import com.ameya.models.UserInfo;
357     import com.ameya.services.UserInfoService;
```

```
358
359 @Service
360 @Transactional
361 public class UserInfoServiceImpl implements UserInfoService {
362
363     @Autowired
364     private UserInfoDAO userInfoDao;
365
366     @Override
367     public void createUser(UserInfo userInfo) {
368         userInfoDao.addUser(userInfo);
369
370     }
371
372     @Override
373     public List<UserInfo> findAll() {
374         return userInfoDao.getAllUserInfo();
375     }
376
377     @Override
378     public UserInfo findById(int id) {
379         return userInfoDao.findById(id);
380     }
381
382     @Override
383     public UserInfo findByIdQuery(int id) {
384         return userInfoDao.findByIdQuery(id);
385     }
386
387     @Override
388     public UserInfo updateUserInfo(UserInfo userInfo, int id) {
389         return userInfoDao.update(userInfo, id);
390     }
391
392     @Override
393     public void deleteById(int id) {
394         userInfoDao.delete(id);
395
396     }
397
398 }
399 ==================================UserInfoController.java============================
    =
400 package com.ameya.controllers;
401
402 import java.util.List;
403
404 import org.springframework.beans.factory.annotation.Autowired;
405 import org.springframework.http.HttpHeaders;
406 import org.springframework.http.HttpStatus;
407 import org.springframework.http.MediaType;
408 import org.springframework.http.ResponseEntity;
409 import org.springframework.web.bind.annotation.DeleteMapping;
410 import org.springframework.web.bind.annotation.GetMapping;
411 import org.springframework.web.bind.annotation.PathVariable;
412 import org.springframework.web.bind.annotation.PostMapping;
413 import org.springframework.web.bind.annotation.PutMapping;
414 import org.springframework.web.bind.annotation.RequestBody;
415 import org.springframework.web.bind.annotation.RequestMapping;
416 import org.springframework.web.bind.annotation.RestController;
417 import org.springframework.web.util.UriComponentsBuilder;
418
419 import com.ameya.models.UserInfo;
420 import com.ameya.services.UserInfoService;
421
422 @RestController
423 @RequestMapping("/userinfo")
424 public class UserInfoController {
425     @Autowired
426     private UserInfoService userInfoService;
427
428     @GetMapping(path="/{id}",produces=MediaType.APPLICATION_JSON_VALUE)
429     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_VALUE)
    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.get
        Id()).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": "Sanjay Joshi",
    "country": "Germany"
}
http://localhost:9001/userinfo/delete/4
http://localhost:9001/userinfo/update/3
http://localhost:9001/userinfo/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")
    @PrimaryKeyJoinColumn
    private int id;
    @Column(name="name")
    private String name;
    @OneToOne(targetEntity = Address.class, cascade = CascadeType.ALL)
    private Address address;
    public Employee() {

    }
    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;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToOne;
import javax.persistence.Table;

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

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

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

```
    private int empId;
    private String address;
    private String country;
    @OneToOne(targetEntity = Employee.class)
    private Employee employee;
    public Address() {

    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public int getEmpId() {
        return empId;
    }
    public void setEmpId(int empId) {
        this.empId = empId;
    }
    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;
    }
    public Employee getEmployee() {
        return employee;
    }
    public void setEmployee(Employee employee) {
        this.employee = employee;
    }

}
==================================================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;
    }

}
==================================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() {
```

```
789
790         }
791         public int getId() {
792             return id;
793         }
794         public void setId(int id) {
795             this.id = id;
796         }
797         public String getBookName() {
798             return bookName;
799         }
800         public void setBookName(String bookName) {
801             this.bookName = bookName;
802         }
803         public Set<Author> getAuthors() {
804             return authors;
805         }
806         public void setAuthors(Set<Author> authors) {
807             this.authors = authors;
808         }
809
810     }
811     ========================================RelationShipDAOImpl.java=====================
    ==========
812     package com.ameya.daos.impl;
813
814     import java.io.Serializable;
815     import java.util.ArrayList;
816     import java.util.HashSet;
817     import java.util.List;
818     import java.util.Set;
819
820     import org.hibernate.Session;
821     import org.hibernate.SessionFactory;
822     import org.hibernate.Transaction;
823     import org.hibernate.query.Query;
824     import org.springframework.beans.factory.annotation.Autowired;
825     import org.springframework.stereotype.Repository;
826
827     import com.ameya.models.Address;
828     import com.ameya.models.Author;
829     import com.ameya.models.Book;
830     import com.ameya.models.Employee;
831     import com.ameya.models.Group;
832     import com.ameya.models.Story;
833
834     @Repository
835     public class RelationShipDAOImpl {
836
837         @Autowired
838         private SessionFactory sessionFactory;
839         public void testOneToOne() {
840             Session sess=sessionFactory.openSession();
841             Transaction tr=sess.beginTransaction();
842             Employee emp=null;
843             Address a1=null;
844
845             emp=new Employee();
846             emp.setName("Ameya Joshi");
847             a1=new Address();
848             a1.setAddress("Kothrud, Pune");
849             a1.setCountry("India");
850
851             emp.setAddress(a1);
852             a1.setEmployee(emp);
853
854             sess.save(emp);
855             sess.flush();
856             Query<Employee> query=sess.createQuery("from Employee");
857             List<Employee> list=query.list();
858             for(Employee e : list) {
859                 System.out.println(e.getId()+" : "+e.getName()+" : "+
860             e.getAddress().getAddress()+" : "+e.getAddress().getCountry());
```

```
861         }
862         tr.commit();
863         sess.close();
864     }
865     public void testOneToMany() {
866         Session sess=sessionFactory.openSession();
867         Transaction tr=sess.beginTransaction();
868         Group group=new Group();
869         group.setName("SPORTS");
870         ArrayList<Story> stories=new ArrayList<>();
871 
872         Story s1=new Story();
873         s1.setInfo("The Allegations- Life Of a Player");
874         stories.add(s1);
875         Story s2=new Story();
876         s2.setInfo("The Cancer Of match Fixing");
877         stories.add(s2);
878         Story s3=new Story();
879         s3.setInfo("The Master Blaster - Sachin");
880         stories.add(s3);
881 
882         group.setStories(stories);
883 
884         Serializable id=sess.save(group);
885         sess.flush();
886         Group g=(Group) sess.load(Group.class, id);
887         System.out.println("GROUP ID :: "+g.getId()+" GROUP NAME :: "+g.getName());
888         List<Story> groupStories=g.getStories();
889         System.out.println("STORIES :: ");
890         for(Story story : groupStories) {
891             System.out.println(story.getId()+" : "+story.getInfo());
892         }
893         tr.commit();
894         sess.close();
895     }
896     public void testManyToOne() {
897         Session sess=sessionFactory.openSession();
898         Story story =(Story)sess.load(Story.class, 5);
899         System.out.println(story.getId()+" : "+story.getInfo());
900         Group group=story.getGroup();
901         System.out.println(group.getId()+" : "+group.getName());
902         sess.close();
903     }
904     public void testManyToMany() {
905         Session sess=sessionFactory.openSession();
906         Transaction tr=sess.beginTransaction();
907 
908         Author a1=new Author();
909         a1.setAuthorName("Sachin");
910 
911         Book b1=new Book();
912         b1.setBookName("My First Pakistan Tour");
913         Book b2=new Book();
914         b2.setBookName("My Cricket - My Life");
915         Book b3=new Book();
916         b3.setBookName("The Unforgettable Don - Don Bradman");
917         Book b4=new Book();
918         b4.setBookName("WC-2011 The Fantastic Experience");
919         Set<Book> books=new HashSet<>();
920         books.add(b1);
921         books.add(b2);
922         books.add(b3);
923         books.add(b4);
924 
925         a1.setBooks(books);
926 
927         Serializable authorId=sess.save(a1);
928         sess.flush();
929 
930         Book b5=new Book();
931         b5.setBookName("The Wonderful Experiences From Cricketing Life");
932 
933         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