```
==========================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>
```

```
47
48      <build>
49          <plugins>
50              <plugin>
51                  <groupId>org.springframework.boot</groupId>
52                  <artifactId>spring-boot-maven-plugin</artifactId>
53              </plugin>
54          </plugins>
55      </build>
56
57  </project>
58  ================================application.properties=====================
    =====
59  server.port=9001
60  logging.level.web=trace
61
62  # Database
63
64  db.url: jdbc:mysql://localhost:3306/sapientdb
65  db.driver: com.mysql.cj.jdbc.Driver
66  db.username: root
67  db.password: root
68
69  # Hibernate
70
71  hibernate.dialect: org.hibernate.dialect.MySQL5Dialect
72  hibernate.show_sql: true
73  hibernate.hbm2ddl.auto: update
74  entitymanager.pkgsScan: com
75
76  ================================HibernateConfiguration.java===============
    ==========
77  package com.ameya.config;
78
79  import java.util.Properties;
80
81  import javax.sql.DataSource;
82
83  import org.springframework.beans.factory.annotation.Value;
84  import org.springframework.context.annotation.Bean;
85  import org.springframework.context.annotation.Configuration;
86  import org.springframework.jdbc.datasource.DriverManagerDataSource;
87  import org.springframework.orm.hibernate5.HibernateTransactionManager;
88  import org.springframework.orm.hibernate5.LocalSessionFactoryBean;
89  import org.springframework.transaction.annotation.EnableTransactionManagement;
90
91  @Configuration
92  @EnableTransactionManagement
```

```
93  public class HibernateConfiguration {
94
95      @Value("${db.driver}")
96      private String DB_DRIVER;
97      @Value("${db.password}")
98      private String DB_PASSWORD;
99      @Value("${db.url}")
100     private String DB_URL;
101     @Value("${db.username}")
102     private String DB_USERNAME;
103     @Value("${hibernate.dialect}")
104     private String HIBERNATE_DIALECT;
105     @Value("${hibernate.show_sql}")
106     private String HIBERNATE_SHOW_SQL;
107     @Value("${hibernate.hbm2ddl.auto}")
108     private String HBM2DDL_AUTO;
109     @Value("${entitymanager.pkgsScan}")
110     private String PACKAGES_TO_SCAN;
111
112     @Bean
113     public DataSource dataSource() {
114         DriverManagerDataSource dataSource=new DriverManagerDataSource();
115         dataSource.setDriverClassName(DB_DRIVER);
116         dataSource.setUrl(DB_URL);
117         dataSource.setUsername(DB_USERNAME);
118         dataSource.setPassword(DB_PASSWORD);
119         return dataSource;
120     }
121     @Bean
122     public LocalSessionFactoryBean sessionFactory() {
123         LocalSessionFactoryBean sessionFactory=new LocalSessionFactoryBean();
124         sessionFactory.setDataSource(dataSource());
125         sessionFactory.setPackagesToScan(PACKAGES_TO_SCAN);
126         Properties hibernateProps=new Properties();
127         hibernateProps.put("hibernate.dialect", HIBERNATE_DIALECT);
128         hibernateProps.put("hibernate.show_sql", HIBERNATE_SHOW_SQL);
129         hibernateProps.put("hibernate.hbm2ddl.auto", HBM2DDL_AUTO);
130         sessionFactory.setHibernateProperties(hibernateProps);
131         return sessionFactory;
132     }
133     @Bean
134     public HibernateTransactionManager transactionManager() {
135         HibernateTransactionManager txManager=new HibernateTransactionManager();
136         txManager.setSessionFactory(sessionFactory().getObject());
137         return txManager;
138     }
139
140 }
```

```
==========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;
```

```
187         this.country = country;
188     }
189     public int getId() {
190         return id;
191     }
192     public void setId(int id) {
193         this.id = id;
194     }
195     public String getFullName() {
196         return fullName;
197     }
198     public void setFullName(String fullName) {
199         this.fullName = fullName;
200     }
201     public String getCountry() {
202         return country;
203     }
204     public void setCountry(String country) {
205         this.country = country;
206     }
207     @Override
208     public String toString() {
209         return "UserInfo [id=" + id + ", fullName=" + fullName + ", country=" +
            country + "]";
210     }
211 
212 }
213 ==========================================UserInfoDAO.java=============
    ========================
214 package com.ameya.daos;
215 
216 import java.util.List;
217 
218 import com.ameya.models.UserInfo;
219 
220 public interface UserInfoDAO {
221     void addUser(UserInfo userInfo);
222     List<UserInfo>getAllUserInfo();
223     UserInfo findById(int id);
224     UserInfo findByIdQuery(int id);
225     UserInfo update(UserInfo userInfo,int id);
226     void delete(int id);
227 
228 }
229 ======================================UserInfoDAOImpl.java==========
    ==================
230 package com.ameya.daos.impl;
231 
```

```
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));
```

```
279         return users.get(0);
280     }
281 
282     @Override
283     public UserInfo update(UserInfo userInfo, int id) {
284         Session session=sessionFactory.getCurrentSession();
285         UserInfo existingUserInfo=(UserInfo)session.load(UserInfo.class, id);
286         existingUserInfo.setFullName(userInfo.getFullName());
287         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","Ameya Joshi");
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 
```

```
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);

    }

```

```
    @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_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.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": "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/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")
    @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;
```

```
552 import javax.persistence.GeneratedValue;
553 import javax.persistence.GenerationType;
554 import javax.persistence.Id;
555 import javax.persistence.OneToOne;
556 import javax.persistence.Table;
557
558 import org.hibernate.annotations.GenericGenerator;
559 import org.hibernate.annotations.Parameter;
560
561 @Entity
562 @Table(name="aj_address")
563 public class Address {
564
565     @Id
566     @GeneratedValue(strategy=GenerationType.AUTO)
567     private int id;
568     @GeneratedValue(generator="gen")
569     @GenericGenerator(name="gen", strategy="foreign",
570     parameters=@Parameter(name="property",value="employee"))
571     private int empId;
572     private String address;
573     private String country;
574     @OneToOne(targetEntity = Employee.class)
575     private Employee employee;
576     public Address() {
577
578     }
579     public int getId() {
580         return id;
581     }
582     public void setId(int id) {
583         this.id = id;
584     }
585     public int getEmpId() {
586         return empId;
587     }
588     public void setEmpId(int empId) {
589         this.empId = empId;
590     }
591     public String getAddress() {
592         return address;
593     }
594     public void setAddress(String address) {
595         this.address = address;
596     }
597     public String getCountry() {
598         return country;
599     }
```

```
    public void setCountry(String country) {
        this.country = country;
    }
    public Employee getEmployee() {
        return employee;
    }
    public void setEmployee(Employee employee) {
        this.employee = employee;
    }

}
=======================================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.Employee;

@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());
        }
        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();
    }
}
====================================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 {
```

```
693
694     @Autowired
695     private RelationShipServiceImpl service;
696     @GetMapping(path="/onetoone")
697     public void testOneToOne() {
698         service.testOneToOne();
699     }
700 }
```