Here,

The Dependency 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
http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.samples</groupId>
  <artifactId>s01springcoredi</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <name>s01springcoredi</name>
  <!-- FIXME change it to the project's website -->
  <url>http://www.example.com</url>

  <properties>
                <java-version>1.6</java-version>

<org.springframework-version>4.0.3.RELEASE</org.springframework-version>
                <org.aspectj-version>1.7.4</org.aspectj-version>
                <org.slf4j-version>1.7.5</org.slf4j-version>
                <hibernate.version>4.3.5.Final</hibernate.version>
        </properties>
        <dependencies>
                <!-- Spring -->
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-context</artifactId>
                        <version>${org.springframework-version}</version>
                        <exclusions>
                                <!-- Exclude Commons Logging in favor of SLF4j -->
                                <exclusion>
                                        <groupId>commons-logging</groupId>
                                        <artifactId>commons-logging</artifactId>
                                 </exclusion>
                        </exclusions>
                </dependency>
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-webmvc</artifactId>
                        <version>${org.springframework-version}</version>
```

```xml
</dependency>
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-tx</artifactId>
        <version>${org.springframework-version}</version>
</dependency>

<!-- Hibernate -->
<dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-core</artifactId>
        <version>${hibernate.version}</version>
</dependency>
<dependency>
        <groupId>org.hibernate</groupId>
        <artifactId>hibernate-entitymanager</artifactId>
        <version>${hibernate.version}</version>
</dependency>

<!-- Apache Commons DBCP -->
<dependency>
        <groupId>commons-dbcp</groupId>
        <artifactId>commons-dbcp</artifactId>
        <version>1.4</version>
</dependency>
<!-- Spring ORM -->
<dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-orm</artifactId>
        <version>${org.springframework-version}</version>
</dependency>

<!-- AspectJ -->
<dependency>
        <groupId>org.aspectj</groupId>
        <artifactId>aspectjrt</artifactId>
        <version>${org.aspectj-version}</version>
</dependency>

<!-- Logging -->
<dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-api</artifactId>
        <version>${org.slf4j-version}</version>
</dependency>
<dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>jcl-over-slf4j</artifactId>
        <version>${org.slf4j-version}</version>
        <scope>runtime</scope>
```

```
</dependency>
<dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-log4j12</artifactId>
        <version>${org.slf4j-version}</version>
        <scope>runtime</scope>
</dependency>
<dependency>
        <groupId>log4j</groupId>
        <artifactId>log4j</artifactId>
        <version>1.2.15</version>
        <exclusions>
                <exclusion>
                        <groupId>javax.mail</groupId>
                        <artifactId>mail</artifactId>
                </exclusion>
                <exclusion>
                        <groupId>javax.jms</groupId>
                        <artifactId>jms</artifactId>
                </exclusion>
                <exclusion>
                        <groupId>com.sun.jdmk</groupId>
                        <artifactId>jmxtools</artifactId>
                </exclusion>
                <exclusion>
                        <groupId>com.sun.jmx</groupId>
                        <artifactId>jmxri</artifactId>
                </exclusion>
        </exclusions>
        <scope>runtime</scope>
</dependency>

<!-- @Inject -->
<dependency>
        <groupId>javax.inject</groupId>
        <artifactId>javax.inject</artifactId>
        <version>1</version>
</dependency>

<!-- Servlet -->
<dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>servlet-api</artifactId>
        <version>2.5</version>
        <scope>provided</scope>
</dependency>
<dependency>
        <groupId>javax.servlet.jsp</groupId>
        <artifactId>jsp-api</artifactId>
        <version>2.1</version>
```

```
                        <scope>provided</scope>
                </dependency>
                <dependency>
                        <groupId>javax.servlet</groupId>
                        <artifactId>jstl</artifactId>
                        <version>1.2</version>
                </dependency>

                <!-- Test -->
                <dependency>
                        <groupId>junit</groupId>
                        <artifactId>junit</artifactId>
                        <version>4.7</version>
                        <scope>test</scope>
                </dependency>

  </dependencies>

  <build>
    <pluginManagement><!-- lock down plugins versions to avoid using Maven defaults
(may be moved to parent pom) -->
      <plugins>
        <!-- clean lifecycle, see
https://maven.apache.org/ref/current/maven-core/lifecycles.html#clean_Lifecycle -->
        <plugin>
          <artifactId>maven-clean-plugin</artifactId>
          <version>3.1.0</version>
        </plugin>
        <!-- default lifecycle, jar packaging: see
https://maven.apache.org/ref/current/maven-core/default-bindings.html#Plugin_bindin
gs_for_jar_packaging -->
        <plugin>
          <artifactId>maven-resources-plugin</artifactId>
          <version>3.0.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-compiler-plugin</artifactId>
          <version>3.8.0</version>
        </plugin>
        <plugin>
          <artifactId>maven-surefire-plugin</artifactId>
          <version>2.22.1</version>
        </plugin>
        <plugin>
          <artifactId>maven-jar-plugin</artifactId>
          <version>3.0.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-install-plugin</artifactId>
          <version>2.5.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-deploy-plugin</artifactId>
          <version>2.8.2</version>
        </plugin>
        <!-- site lifecycle, see
https://maven.apache.org/ref/current/maven-core/lifecycles.html#site_Lifecycle -->
        <plugin>
          <artifactId>maven-site-plugin</artifactId>
          <version>3.7.1</version>
        </plugin>
        <plugin>
          <artifactId>maven-project-info-reports-plugin</artifactId>
          <version>3.0.0</version>
        </plugin>
      </plugins>
    </pluginManagement>
  </build>
</project>
```

Object model creation--->

```
package com.journaldev.spring.model;

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


@Entity
@Table(name="PERSON")
public class Person {

        @Id
        @Column(name="id")
        @GeneratedValue(strategy=GenerationType.IDENTITY)
        private int id;

        private String name;

        private String country;

        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 String getCountry() {
		return country;
	}

	public void setCountry(String country) {
		this.country = country;
	}

	@Override
	public String toString(){
		return "id="+id+", name="+name+", country="+country;
	}
}
```

DAO ---->

```
package com.journaldev.spring.dao;

import java.util.List;

import com.journaldev.spring.model.Person;

public interface PersonDAO {

	public void addPerson(Person p);
	public void updatePerson(Person p);
	public List<Person> listPersons();
	public Person getPersonById(int id);
	public void removePerson(int id);
}
```

implementation--->

```
package com.journaldev.spring.dao;

import java.util.List;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.stereotype.Repository;

import com.journaldev.spring.model.Person;

@Repository
public class PersonDAOImpl implements PersonDAO {

        private static final Logger logger =
LoggerFactory.getLogger(PersonDAOImpl.class);

        private SessionFactory sessionFactory;

        public void setSessionFactory(SessionFactory sf){
                this.sessionFactory = sf;
        }

        @Override
        public void addPerson(Person p) {
                Session session = this.sessionFactory.getCurrentSession();
                session.persist(p);
                logger.info("Person saved successfully, Person Details="+p);
        }

        @Override
        public void updatePerson(Person p) {
                Session session = this.sessionFactory.getCurrentSession();
                session.update(p);
                logger.info("Person updated successfully, Person Details="+p);
        }

        @SuppressWarnings("unchecked")
        @Override
        public List<Person> listPersons() {
                Session session = this.sessionFactory.getCurrentSession();
                List<Person> personsList = session.createQuery("from
Person").list();
                for(Person p : personsList){
                        logger.info("Person List::"+p);
                }
                return personsList;
        }

        @Override
        public Person getPersonById(int id) {
                Session session = this.sessionFactory.getCurrentSession();

                Person p = (Person) session.load(Person.class, new Integer(id));
                logger.info("Person loaded successfully, Person details="+p);
                return p;
        }

        @Override
        public void removePerson(int id) {
                Session session = this.sessionFactory.getCurrentSession();
                Person p = (Person) session.load(Person.class, new Integer(id));
                if(null != p){
                        session.delete(p);
                }
                logger.info("Person deleted successfully, person details="+p);
        }

}
```

Service Class ---->

```
package com.journaldev.spring.service;

import java.util.List;

import com.journaldev.spring.model.Person;

public interface PersonService {

        public void addPerson(Person p);
        public void updatePerson(Person p);
        public List<Person> listPersons();
        public Person getPersonById(int id);
        public void removePerson(int id);

}

package com.journaldev.spring.service;

import java.util.List;

import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;
import com.journaldev.spring.dao.PersonDAO;
import com.journaldev.spring.model.Person;

@Service
public class PersonServiceImpl implements PersonService {

        private PersonDAO personDAO;

        public void setPersonDAO(PersonDAO personDAO) {
                this.personDAO = personDAO;
        }

        @Override
        @Transactional
        public void addPerson(Person p) {
                this.personDAO.addPerson(p);
        }

        @Override
        @Transactional
        public void updatePerson(Person p) {
                this.personDAO.updatePerson(p);
        }

        @Override
        @Transactional
        public List<Person> listPersons() {
                return this.personDAO.listPersons();
        }

        @Override
        @Transactional
        public Person getPersonById(int id) {
                return this.personDAO.getPersonById(id);
        }

        @Override
        @Transactional
        public void removePerson(int id) {
                this.personDAO.removePerson(id);
        }

}
```

Controller Class---->

```
package com.journaldev.spring;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

import com.journaldev.spring.model.Person;
import com.journaldev.spring.service.PersonService;

@Controller
public class PersonController {

        private PersonService personService;

        @Autowired(required=true)
        @Qualifier(value="personService")
        public void setPersonService(PersonService ps){
                this.personService = ps;
        }

        @RequestMapping(value = "/persons", method = RequestMethod.GET)
        public String listPersons(Model model) {
                model.addAttribute("person", new Person());
                model.addAttribute("listPersons",
this.personService.listPersons());
                return "person";
        }

        //For add and update person both
        @RequestMapping(value= "/person/add", method = RequestMethod.POST)
        public String addPerson(@ModelAttribute("person") Person p){

                if(p.getId() == 0){
                        //new person, add it
                        this.personService.addPerson(p);
                }else{
                        //existing person, call update
                        this.personService.updatePerson(p);
                }

                return "redirect:/persons";

        }
        @RequestMapping("/remove/{id}")
    public String removePerson(@PathVariable("id") int id){

        this.personService.removePerson(id);
        return "redirect:/persons";
    }

    @RequestMapping("/edit/{id}")
    public String editPerson(@PathVariable("id") int id, Model model){
        model.addAttribute("person", this.personService.getPersonById(id));
        model.addAttribute("listPersons", this.personService.listPersons());
        return "person";
    }

}
```

beans--->

```
<beans:bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
                destroy-method="close">
                <beans:property name="driverClassName"
value="com.mysql.jdbc.Driver" />
                <beans:property name="url"
                        value="jdbc:mysql://localhost:3306/mydb" />
                <beans:property name="username" value="root" />
                <beans:property name="password" value="Veda@1202189" />
        </beans:bean>


<beans:bean id="personDAO" class="com.journaldev.spring.dao.PersonDAOImpl">
                <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
        </beans:bean>
        <beans:bean id="personService"
class="com.journaldev.spring.service.PersonServiceImpl">
                <beans:property name="personDAO" ref="personDAO"></beans:property>
        </beans:bean>
        <context:component-scan base-package="com.journaldev.spring" />

        <tx:annotation-driven transaction-manager="transactionManager"/>

        <beans:bean id="transactionManager"
class="org.springframework.orm.hibernate4.HibernateTransactionManager">
                <beans:property name="sessionFactory"
ref="hibernate4AnnotatedSessionFactory" />
        </beans:bean>
```