One to Many in Hibernate – Bidirectional

# SQL(ddl.sql)

**drop** **table** if **exists** emp;

**drop** **table** if **exists** department;

**CREATE** **TABLE** department (

department\_id BIGINT(20) **NOT** **NULL** AUTO\_INCREMENT,

dept\_name **VARCHAR**(50) **NOT** **NULL** ,

**PRIMARY** **KEY** (department\_id)

);

**CREATE** **TABLE** emp (

employee\_id BIGINT(10) **NOT** **NULL** AUTO\_INCREMENT,

firstname **VARCHAR**(50) **NULL** **DEFAULT** **NULL**,

lastname **VARCHAR**(50) **NULL** **DEFAULT** **NULL**,

department\_id BIGINT(20) **NULL** **DEFAULT** **NULL**,

**PRIMARY** **KEY** (employee\_id),

**FOREIGN** **KEY** (department\_id) **REFERENCES** department (department\_id)

);

**select** \* **from** emp;

**select** \* **from** department;

# Maven(pom.xml)

<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>one-to-many-bidirectional1</groupId>

<artifactId>one-to-many-bidirectional1</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>spring-hibernate</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<spring.version>4.1.2.RELEASE</spring.version>

<spring.security.version>3.2.3.RELEASE</spring.security.version>

</properties>

<dependencies>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>4.0</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-web</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-orm</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>5.1.30</version>

</dependency>

<dependency> <!-- C3P0 library -->

<groupId>com.mchange</groupId>

<artifactId>c3p0</artifactId>

<version>0.9.5</version>

</dependency>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>4.3.7.Final</version>

</dependency>

</dependencies>

</project>

# Spring-Hibernate Configuration (app-context.xml)

<beans xmlns=*"http://www.springframework.org/schema/beans"*

xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"* xmlns:context=*"http://www.springframework.org/schema/context"*

xmlns:aop=*"http://www.springframework.org/schema/aop"* xmlns:jee=*"http://www.springframework.org/schema/jee"*

xmlns:tx=*"http://www.springframework.org/schema/tx"* xmlns:jdbc=*"http://www.springframework.org/schema/jdbc"*

xmlns:osgi=*"http://www.springframework.org/schema/osgi"* xmlns:security=*"http://www.springframework.org/schema/security"*

xsi:schemaLocation=*"http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-3.0.xsd*

*http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.0.xsd*

*http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context-3.0.xsd*

*http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee-3.0.xsd*

*http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx-3.0.xsd*

*http://www.springframework.org/schema/jdbc http://www.springframework.org/schema/jdbc/spring-jdbc-3.0.xsd*

*http://www.springframework.org/schema/osgi http://www.springframework.org/schema/osgi/spring-osgi.xsd*

*http://www.springframework.org/schema/security http://www.springframework.org/schema/security/spring-security-3.0.3.xsd"*>

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

<bean id=*"dataSourceInternal"* class=*"com.mchange.v2.c3p0.ComboPooledDataSource"*

destroy-method=*"close"*>

<property name=*"driverClass"* value=*"com.mysql.jdbc.Driver"* />

<property name=*"jdbcUrl"* value=*"jdbc:mysql://localhost/test"* />

<property name=*"user"* value=*"deba"* />

<property name=*"password"* value=*"deba"* />

<!-- these are C3P0 properties -->

<property name=*"acquireIncrement"* value=*"5"* />

<property name=*"initialPoolSize"* value=*"5"* />

<property name=*"minPoolSize"* value=*"5"* />

<property name=*"maxPoolSize"* value=*"20"* />

</bean>

<!-- This is the lazy DataSource proxy that interacts with the target DataSource once a real statement is sent to the database. Users use this DataSource to set up their Hibernate session factory, which in turn forces the Hibernate second-level cache and also everything that interacts with that Hibernate session factory to use it. -->

<bean id=*"dataSource"* class=*"org.springframework.jdbc.datasource.LazyConnectionDataSourceProxy"*>

<property name=*"targetDataSource"*><ref bean=*"dataSourceInternal"* /></property>

</bean>

<bean id=*"hibSessionFactory"* class=*"org.springframework.orm.hibernate4.LocalSessionFactoryBean"*>

<property name=*"dataSource"* ref=*"dataSource"* />

<!--<property name="hibernateProperties"> <value> hibernate.show\_sql=true

</value> </property> -->

<property name=*"hibernateProperties"*>

<value>

hibernate.id.new\_generator\_mappings=true

<!-- hibernate.current\_session\_context\_class=thread -->

</value>

</property>

<property name=*"annotatedClasses"*>

<list>

<value>com.ddlab.rnd.orm.Department</value>

<value>com.ddlab.rnd.orm.Employee</value>

</list>

</property>

</bean>

</beans>

# Java Code

## Department.java

**package** com.ddlab.rnd.orm;

**import** java.util.List;

**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.JoinColumn;

**import** javax.persistence.OneToMany;

**import** javax.persistence.Table;

@Entity

@Table(name="DEPARTMENT")

**public** **class** Department {

@Id

@GeneratedValue(strategy=GenerationType.***IDENTITY***)

@Column(name="DEPARTMENT\_ID")

**private** Long departmentId;

@Column(name="DEPT\_NAME")

**private** String departmentName;

@OneToMany(cascade={CascadeType.***ALL***})

@JoinColumn(name="department\_id")

**private** List<Employee> employees;

**public** Long getDepartmentId() {

**return** departmentId;

}

**public** **void** setDepartmentId(Long departmentId) {

**this**.departmentId = departmentId;

}

**public** String getDepartmentName() {

**return** departmentName;

}

**public** **void** setDepartmentName(String departmentName) {

**this**.departmentName = departmentName;

}

**public** List<Employee> getEmployees() {

**return** employees;

}

**public** **void** setEmployees(List<Employee> employees) {

**this**.employees = employees;

}

}

## Employee.java

**package** com.ddlab.rnd.orm;

**import** javax.persistence.Column;

**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="EMP")

**public** **class** Employee {

@Id

@GeneratedValue(strategy=GenerationType.***IDENTITY***)

@Column(name="employee\_id")

**private** Long employeeId;

@Column(name="firstname")

**private** String firstname;

@Column(name="lastname")

**private** String lastname;

@ManyToOne

@JoinColumn(name="department\_id")

**private** Department department;

**public** Employee() {

}

**public** Employee(String firstname, String lastname) {

**this**.firstname = firstname;

**this**.lastname = lastname;

}

**public** Long getEmployeeId() {

**return** employeeId;

}

**public** **void** setEmployeeId(Long employeeId) {

**this**.employeeId = employeeId;

}

**public** String getFirstname() {

**return** firstname;

}

**public** **void** setFirstname(String firstname) {

**this**.firstname = firstname;

}

**public** String getLastname() {

**return** lastname;

}

**public** **void** setLastname(String lastname) {

**this**.lastname = lastname;

}

**public** Department getDepartment() {

**return** department;

}

**public** **void** setDepartment(Department department) {

**this**.department = department;

}

}

## Test Program(App.java)

**package** com.ddlab.rnd.orm;

**import** java.util.ArrayList;

**import** java.util.List;

**import** org.hibernate.Query;

**import** org.hibernate.Session;

**import** org.hibernate.SessionFactory;

**import** org.springframework.context.ApplicationContext;

**import** org.springframework.context.support.ClassPathXmlApplicationContext;

**public** **class** App

{

**public** **static** **void** showDepartment(Session session) {

**try** {

String hql = "FROM Department";

Query query = session.createQuery(hql);

List results = query.list();

**for**( **int** i = 0 ; i < results.size() ; i++) {

Department dept = (Department) results.get(i);

System.***out***.println(dept.getDepartmentName());

List<Employee> empList = dept.getEmployees();

**for**(Employee emp : empList) {

System.***out***.println(emp.getFirstname()+"----"+emp.getLastname()+"----"+emp.getEmployeeId());

}

}

} **catch** (Exception e) {

e.printStackTrace();

}

}

**public** **static** **void** saveDepartment(Session session) {

session.beginTransaction();

Department department = **new** Department();

department.setDepartmentName("Sales");

Employee emp1 = **new** Employee("Lilu", "Mishra");

Employee emp2 = **new** Employee("Bibhu", "Tripathy");

department.setEmployees(**new** ArrayList<Employee>());

department.getEmployees().add(emp1);

department.getEmployees().add(emp2);

session.save(department);

session.getTransaction().commit();

session.close();

}

**public** **static** **void** main( String[] args )

{

ApplicationContext context = **new** ClassPathXmlApplicationContext("app-context.xml");

SessionFactory sessionFactory = (SessionFactory)context.getBean("hibSessionFactory");

Session session = sessionFactory.openSession();

// saveDepartment(session);

*showDepartment*(session);

}

}