Java9

Hibernate 5

**Database**

Create database Test;

Use Test;

CREATE TABLE `worker` (   
`worker\_id` bigint(10) NOT NULL auto\_increment,   
`firstname` varchar(50) default NULL,   
`lastname` varchar(50) default NULL,   
`birth\_date` date NOT NULL,   
`cell\_phone` varchar(15) NOT NULL,   
PRIMARY KEY (`worker\_id`)   
) ENGINE=InnoDB DEFAULT CHARSET=latin1   
  
  
CREATE TABLE `workerdetail` (   
`worker\_id` bigint(20) NOT NULL auto\_increment,   
`street` varchar(50) default NULL,   
`city` varchar(50) default NULL,   
`state` varchar(50) default NULL,   
`country` varchar(50) default NULL,   
PRIMARY KEY (`worker\_id`),   
CONSTRAINT `FK\_workerdetail` FOREIGN KEY (`worker\_id`) REFERENCES `worker` (`worker\_id`)   
) ENGINE=InnoDB DEFAULT CHARSET=latin1

**Maven dependencies**

<dependencies>

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.2.6.Final</version>

</dependency>

<dependency>

<groupId>mysql</groupId>

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

<version>5.1.10</version>

</dependency>

</dependencies>

**Steps to create project**

Create java project

Convert to maven

Run class with main method

Make sure you added **--add-modules java.xml.bind** vm arguments

**Hibernate.cfg.xml**

<?xml version='1.0' encoding='utf-8'?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<!-- Database connection settings -->

<property name="connection.driver\_class">com.mysql.jdbc.Driver</property>

<property name="connection.url">jdbc:mysql://localhost:3306/Test</property>

<property name="connection.username">root</property>

<property name="connection.password">root</property>

<property name="connection.pool\_size">1</property>

<property name="dialect">org.hibernate.dialect.MySQLDialect</property>

<property name="current\_session\_context\_class">thread</property>

<property name="cache.provider\_class">org.hibernate.cache.NoCacheProvider</property>

<property name="show\_sql">true</property>

<property name="hbm2ddl.auto">validate</property>

<mapping class="com.worldclock.WorkerDetail"/>

<mapping class="com.worldclock.Worker"/>

</session-factory>

</hibernate-configuration>

Primary Entity class

package com.worldclock;

import java.util.Date;

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.Table;

@Entity

@Table(name = "worker")

public class Worker {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "worker\_id")

private Long workerId;

@Column(name = "firstname")

private String firstname;

@Column(name = "lastname")

private String lastname;

@Column(name = "birth\_date")

private Date birthDate;

@Column(name = "cell\_phone")

private String cellphone;

@OneToOne(mappedBy = "worker", cascade = CascadeType.ALL)

private WorkerDetail workerDetail;

public Worker() {

}

public Worker(String firstname, String lastname, Date birthdate,

String phone) {

this.firstname = firstname;

this.lastname = lastname;

this.birthDate = birthdate;

this.cellphone = phone;

}

public Long getWorkerId() {

return workerId;

}

public void setWorkerId(Long workerId) {

this.workerId = workerId;

}

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 Date getBirthDate() {

return birthDate;

}

public void setBirthDate(Date birthDate) {

this.birthDate = birthDate;

}

public String getCellphone() {

return cellphone;

}

public void setCellphone(String cellphone) {

this.cellphone = cellphone;

}

public WorkerDetail getWorkerDetail() {

return workerDetail;

}

public void setWorkerDetail(WorkerDetail workerDetail) {

this.workerDetail = workerDetail;

}

}

**Mapped Entity class**

package com.worldclock;

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;

import org.hibernate.annotations.GenericGenerator;

import org.hibernate.annotations.Parameter;

@Entity

@Table(name="workerdetail")

public class WorkerDetail {

@Id

@Column(name="worker\_id", unique=true, nullable=false)

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long workerId;

@Column(name="street")

private String street;

@Column(name="city")

private String city;

@Column(name="state")

private String state;

@Column(name="country")

private String country;

@OneToOne

@PrimaryKeyJoinColumn

private Worker worker;

public WorkerDetail() {

}

public WorkerDetail(String street, String city, String state, String country) {

this.street = street;

this.city = city;

this.state = state;

this.country = country;

}

public Long getWorkerId() {

return workerId;

}

public void setWorkerId(Long workerId) {

this.workerId = workerId;

}

public String getStreet() {

return street;

}

public void setStreet(String street) {

this.street = street;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

public String getState() {

return state;

}

public void setState(String state) {

this.state = state;

}

public String getCountry() {

return country;

}

public void setCountry(String country) {

this.country = country;

}

public Worker getWorker() {

return worker;

}

public void setWorker(Worker worker) {

this.worker = worker;

}

}

**Main class**

package com.worldclock;

import java.util.Date;

import java.text.ParseException;

import java.text.SimpleDateFormat;

import java.util.List;

import javax.persistence.TypedQuery;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.boot.Metadata;

import org.hibernate.boot.MetadataSources;

import org.hibernate.boot.registry.StandardServiceRegistry;

import org.hibernate.boot.registry.StandardServiceRegistryBuilder;

public class ManageWorker {

private static SessionFactory sf;

private static StandardServiceRegistry standardServiceRegistry;

@SuppressWarnings("unchecked")

public static void main(String[] args) {

try {

standardServiceRegistry = new StandardServiceRegistryBuilder()

.configure()

.build();

// Create MetadataSources

MetadataSources metadataSources = new MetadataSources(standardServiceRegistry);

// Create Metadata

Metadata metadata = metadataSources.getMetadataBuilder().build();

// Create SessionFactory

sf = metadata.getSessionFactoryBuilder().build();

} catch (Throwable ex) {

System.err.println("Failed to create sessionFactory object." + ex);

throw new ExceptionInInitializerError(ex);

}

System.out.println("Example : Hibernate One to One Mapping using Annotation ");

Session session = sf.openSession();

session.beginTransaction();

WorkerDetail workerDetail = new WorkerDetail("Richmond Town", "Bangalore",

"Karnataka", "India");

//For passing Date of birth as String

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

Date dob=null;

try {

dob = sdf.parse("1987-05-21");

} catch (ParseException e) {

e.printStackTrace();

}

Worker worker = new Worker("ram", "prased",dob,

"919595959595");

worker.setWorkerDetail(workerDetail);

workerDetail.setWorker(worker);

session.save(worker);

TypedQuery<Worker> query = session.createQuery("FROM Worker");

List<Worker> workerList = query.getResultList();

//List<Worker> workerList = session.createQuery("from Worker").list();

for (Worker work1 : workerList) {

System.out.println(work1.getFirstname() + " , "

+ work1.getLastname() + ", "

+ work1.getWorkerDetail().getState());

}

session.getTransaction().commit();

session.close();

}

}