这是Users.java部分

package Hibernate;

import java.util.HashSet;

import java.util.Set;

/\*\*

\* Users entity. @author MyEclipse Persistence Tools

\*/

public class Users implements java.io.Serializable {

// Fields

private String username;

private String pwd;

private String teleNo;

private Integer age;

private String sex;

private String account;

private Integer grade;

private Integer warnings;

// Constructors

/\*\* default constructor \*/

public Users() {

}

/\*\* full constructor \*/

public Users(String username, String pwd, String teleNo, Integer age,

String sex, String account, Integer grade, Integer warnings) {

// TODO Auto-generated constructor stub

this.username = username;

this.pwd = pwd;

this.teleNo = teleNo;

this.age = age;

this.sex = sex;

this.account = account;

this.grade = grade;

this.warnings = warnings;

}

// Property accessors

private Integer id;

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getUsername() {

return username;

}

public void setUsername(String username) {

this.username = username;

}

public String getPwd() {

return pwd;

}

public void setPwd(String pwd) {

this.pwd = pwd;

}

public String getTeleNo() {

return teleNo;

}

public void setTeleNo(String teleNo) {

this.teleNo = teleNo;

}

public Integer getAge() {

return age;

}

public void setAge(Integer age) {

this.age = age;

}

public String getSex() {

return sex;

}

public void setSex(String sex) {

this.sex = sex;

}

public String getAccount() {

return account;

}

public void setAccount(String account) {

this.account = account;

}

public Integer getGrade() {

return grade;

}

public void setGrade(Integer grade) {

this.grade = grade;

}

public Integer getWarnings() {

return warnings;

}

public void setWarnings(Integer warnings) {

this.warnings = warnings;

}

}

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

<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<!--

Mapping file autogenerated by MyEclipse Persistence Tools

-->

<hibernate-mapping>

<class name="Hibernate.Users" table="users" catalog="myparking">

<id name="id" type="java.lang.Integer">

<column name="id" />

<generator class="identity" />

</id>

<property name="username" type="java.lang.String">

<column name="username" length="10" />

</property>

<property name="pwd" type="java.lang.String">

<column name="pwd" length="10" />

</property>

<property name="teleNo" type="java.lang.String">

<column name="teleNo" length="15" />

</property>

<property name="age" type="java.lang.Integer">

<column name="age" />

</property>

<property name="sex" type="java.lang.String">

<column name="sex" length="1" />

</property>

<property name="account" type="java.lang.String">

<column name="account" length="11" />

</property>

<property name="grade" type="java.lang.Integer">

<column name="grade" />

</property>

<property name="warnings" type="java.lang.Integer">

<column name="warnings" />

</property>

</class>

</hibernate-mapping>

UsersBO.java部分

package Hibernate;

import java.io.Serializable;

import java.math.BigDecimal;

import java.math.BigInteger;

import java.util.Calendar;

import java.util.Collection;

import java.util.Date;

import java.util.Iterator;

import java.util.Locale;

import java.util.Map;

import java.util.Set;

import org.hibernate.CacheMode;

import org.hibernate.FlushMode;

import org.hibernate.HibernateException;

import org.hibernate.LockMode;

import org.hibernate.LockOptions;

import org.hibernate.Query;

import org.hibernate.ScrollMode;

import org.hibernate.ScrollableResults;

import org.hibernate.Session;

import org.hibernate.Transaction;

import org.hibernate.transform.ResultTransformer;

import org.hibernate.type.Type;

import java.util.List;

public class UsersBO {

public void addUsers(String username, String pwd, String teleNo, Integer age,

String sex, String account, Integer grade,Integer warnings){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Users users= new Users(username, pwd, teleNo, age,sex, account, grade,warnings);

session.save(users);

tx.commit();

}

public Users getUsers(Integer id){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Users user = (Users)session.get(Users.class, id);

tx.commit();

return user;

}

public boolean IsUserExist(Integer id){

Users user = getUsers(id);

if(user==null)return false;

else return true;

}

public boolean checkUser(Integer id, String pwd){

Users user=getUsers(id);

if(user.getPwd().equals(pwd))return true;

else return false;

}

public List<Users> getUsers(){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Query query = session.createQuery("from Users");

List<Users> list = query.list();

tx.commit();

return list;

}

public void updateUsersByUser(Users users, String username, String pwd, String teleNo, Integer age,

String sex, String account){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

users.setUsername(username);

users.setPwd(pwd);

users.setTeleNo(teleNo);

users.setAge(age);

users.setSex(sex);

users.setAccount(account);

session.saveOrUpdate(users);

tx.commit();

}

public void updateUsersBySuper(Users users,Integer grade){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

users.setGrade(grade);

session.saveOrUpdate(users);

tx.commit();

}

public void deleteUsers(Integer id){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Users user = (Users)session.get(Users.class, id);

session.delete(user);

tx.commit();

}

}

Parkspace.java

package Hibernate;

import java.sql.Timestamp;

/\*\*

\* Parkspace entity. @author MyEclipse Persistence Tools

\*/

public class Parkspace implements java.io.Serializable {

// Fields

private Integer pos;

private String isEmpty;

private String carnum;

private String isReserved;

private Timestamp reserveStart;

private Timestamp reserveEnd;

// Constructors

/\*\* default constructor \*/

public Parkspace() {

}

/\*\* minimal constructor \*/

public Parkspace(Integer pos, Timestamp reserveStart, Timestamp reserveEnd) {

this.pos = pos;

this.reserveStart = reserveStart;

this.reserveEnd = reserveEnd;

}

/\*\* full constructor \*/

public Parkspace(Integer pos, String isEmpty, String carnum,

String isReserved, Timestamp reserveStart, Timestamp reserveEnd) {

this.pos = pos;

this.isEmpty = isEmpty;

this.carnum = carnum;

this.isReserved = isReserved;

this.reserveStart = reserveStart;

this.reserveEnd = reserveEnd;

}

// Property accessors

public Integer getPos() {

return this.pos;

}

public void setPos(Integer pos) {

this.pos = pos;

}

public String getIsEmpty() {

return this.isEmpty;

}

public void setIsEmpty(String isEmpty) {

this.isEmpty = isEmpty;

}

public String getCarnum() {

return this.carnum;

}

public void setCarnum(String carnum) {

this.carnum = carnum;

}

public String getIsReserved() {

return this.isReserved;

}

public void setIsReserved(String isReserved) {

this.isReserved = isReserved;

}

public Timestamp getReserveStart() {

return this.reserveStart;

}

public void setReserveStart(Timestamp reserveStart) {

this.reserveStart = reserveStart;

}

public Timestamp getReserveEnd() {

return this.reserveEnd;

}

public void setReserveEnd(Timestamp reserveEnd) {

this.reserveEnd = reserveEnd;

}

}

Parkspace.hbm.xml

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

<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<!--

Mapping file autogenerated by MyEclipse Persistence Tools

-->

<hibernate-mapping>

<class name="Hibernate.Parkspace" table="parkspace" catalog="myparking">

<id name="pos" type="java.lang.Integer">

<column name="pos" />

<generator class="assigned" />

</id>

<property name="isEmpty" type="java.lang.String">

<column name="is\_empty" length="1" />

</property>

<property name="carnum" type="java.lang.String">

<column name="carnum" length="7" />

</property>

<property name="isReserved" type="java.lang.String">

<column name="is\_reserved" length="1" />

</property>

<property name="reserveStart" type="java.sql.Timestamp">

<column name="reserve\_start" length="19" not-null="false" />

</property>

<property name="reserveEnd" type="java.sql.Timestamp">

<column name="reserve\_end" length="19" not-null="false" />

</property>

</class>

</hibernate-mapping>

ParkspaceBO.java

package Hibernate;

import org.hibernate.Query;

import org.hibernate.Session;

import org.hibernate.Transaction;

import java.sql.Timestamp;

import java.util.List;

public class ParkspaceBO {

public void addParkspace(Integer pos, String isEmpty, String carnum,

String isReserved, Timestamp reserveStart, Timestamp reserveEnd){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Parkspace Parkspace= new Parkspace(pos, "0", null, "0", null,null);

session.save(Parkspace);

tx.commit();

}

public void addParkspace(Parkspace parkspace){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

session.save(parkspace);

tx.commit();

}

public Parkspace getParkspace(String size){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

String s = "";

if(size.indexOf("A")!=-1){

s += "from Parkspace where pos between 1 and 100 and ";

}

else if(size.indexOf("B")!=-1){

s += "from Parkspace where pos between 101 and 200 and ";

}

else if(size.indexOf("C")!=-1){

s += "from Parkspace where pos between 201 and 300 and ";

}

else{

s += "from Parkspace where pos between 301 and 400 and ";

}

s+="is\_empty=1";

Query query = session.createQuery(s);

List<Parkspace> list = query.list();

try

{

tx.commit();

} catch (Exception e)

{

if (null != tx)

tx.rollback();

e.printStackTrace();

}

return list.get(0);

}

public void updateParkspace(Parkspace parkspace,String carnum, String isempty){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

parkspace.setCarnum(carnum);

parkspace.setIsEmpty(isempty);

parkspace.setReserveStart(parkspace.getReserveEnd());

session.saveOrUpdate(parkspace);

tx.commit();

}

public void updateParkspace(Parkspace parkspace, String isempty){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

parkspace.setCarnum(null);

parkspace.setIsEmpty(isempty);

session.saveOrUpdate(parkspace);

tx.commit();

}

public void updateParkspace(Parkspace parkspace, String carnum,String isempty, String isreserve,Timestamp start,Timestamp end){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

parkspace.setIsEmpty(isempty);

parkspace.setCarnum(carnum);

parkspace.setIsReserved(isreserve);

parkspace.setReserveStart(start);

parkspace.setReserveEnd(end);

session.saveOrUpdate(parkspace);

tx.commit();

}

public Parkspace getParkspace(Integer pos){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Parkspace user = (Parkspace)session.get(Parkspace.class, pos);

tx.commit();

return user;

}

public boolean IsParkspaceAvailable(Integer pos){

Parkspace parkspace = getParkspace(pos);

return Integer.parseInt(parkspace.getIsEmpty())==1?true:false;

}

public boolean IsParkspaceReserved(Integer pos){

Parkspace parkspace = getParkspace(pos);

return Integer.parseInt(parkspace.getIsReserved())==1?true:false;

}

public List<Parkspace> getParkspace(){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Query query = session.createQuery("from Parkspace");

List<Parkspace> list = query.list();

tx.commit();

return list;

}

public List<Parkspace> getparkspace(String carnum){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

String a = "from Parkspace where carnum ="+carnum+" and is\_reserved = 1";

Query query = session.createQuery(a);

List<Parkspace> list = query.list();

tx.commit();

return list;

}

public void reserve(Parkspace cars,Timestamp start,Timestamp end){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

cars.setReserveStart(start);

cars.setReserveEnd(end);

session.saveOrUpdate(cars);

tx.commit();

}

public void deleteParkspace(Integer id){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Parkspace Parkspace = getParkspace(id);

session.delete(Parkspace);

tx.commit();

}

}

Employers.java

package Hibernate;

import java.sql.Timestamp;

/\*\*

\* Employers entity. @author MyEclipse Persistence Tools

\*/

public class Employers implements java.io.Serializable {

// Fields

private Integer employId;

private String name;

private String pwd;

private Integer age;

private String sex;

private String grade;

private String isWork;

private Timestamp lastUsed;

// Constructors

/\*\* default constructor \*/

public Employers() {

}

/\*\* minimal constructor \*/

public Employers(Timestamp lastUsed) {

this.lastUsed = lastUsed;

}

/\*\* full constructor \*/

public Employers(String name, String pwd, Integer age, String sex,

String grade, String isWork, Timestamp lastUsed) {

this.name = name;

this.pwd = pwd;

this.age = age;

this.sex = sex;

this.grade = grade;

this.isWork = isWork;

this.lastUsed = lastUsed;

}

// Property accessors

public Integer getEmployId() {

return this.employId;

}

public void setEmployId(Integer employId) {

this.employId = employId;

}

public String getName() {

return this.name;

}

public void setName(String name) {

this.name = name;

}

public String getPwd() {

return this.pwd;

}

public void setPwd(String pwd) {

this.pwd = pwd;

}

public Integer getAge() {

return this.age;

}

public void setAge(Integer age) {

this.age = age;

}

public String getSex() {

return this.sex;

}

public void setSex(String sex) {

this.sex = sex;

}

public String getGrade() {

return this.grade;

}

public void setGrade(String grade) {

this.grade = grade;

}

public String getIsWork() {

return this.isWork;

}

public void setIsWork(String isWork) {

this.isWork = isWork;

}

public Timestamp getLastUsed() {

return this.lastUsed;

}

public void setLastUsed(Timestamp lastUsed) {

this.lastUsed = lastUsed;

}

}

Employers.hbm.xml

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

<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<!--

Mapping file autogenerated by MyEclipse Persistence Tools

-->

<hibernate-mapping>

<class name="Hibernate.Employers" table="employers" catalog="myparking">

<id name="employId" type="java.lang.Integer">

<column name="employ\_id" />

<generator class="identity" />

</id>

<property name="name" type="java.lang.String">

<column name="name" length="10" />

</property>

<property name="pwd" type="java.lang.String">

<column name="pwd" length="10" />

</property>

<property name="age" type="java.lang.Integer">

<column name="age" />

</property>

<property name="sex" type="java.lang.String">

<column name="sex" length="1" />

</property>

<property name="grade" type="java.lang.String">

<column name="grade" length="1" />

</property>

<property name="isWork" type="java.lang.String">

<column name="is\_work" length="1" />

</property>

<property name="lastUsed" type="java.sql.Timestamp">

<column name="last\_used" length="19" not-null="true" />

</property>

</class>

</hibernate-mapping>

EmployersBO.java

package Hibernate;

import org.hibernate.Query;

import org.hibernate.Session;

import org.hibernate.Transaction;

import java.sql.Timestamp;

import java.util.List;

public class EmployersBO {

public void addEmployers(String name, String pwd, Integer age, String sex,

String grade, String isWork,Timestamp lastUsed){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Employers Employers= new Employers(name, pwd, age, sex, grade, "1", new Timestamp(System.currentTimeMillis()));

session.save(Employers);

tx.commit();

}

public Employers getEmployers(Integer id){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Employers user = (Employers)session.get(Employers.class, id);

tx.commit();

return user;

}

public boolean IsEmployerExist(Integer id){

Employers employers = getEmployers(id);

if(employers==null)return false;

else return true;

}

public boolean checkEmployer(Integer id, String pwd){

Employers employers=getEmployers(id);

if(employers.getPwd().equals(pwd))return true;

else return false;

}

public List<Employers> getEmployers(){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Query query = session.createQuery("from Employers");

List<Employers> list = query.list();

tx.commit();

return list;

}

public void updateEmployersByEmployee(Employers employers,String name, String pwd, Integer age, String sex

){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

employers.setName(name);

employers.setPwd(pwd);

employers.setAge(age);

employers.setSex(sex);

session.saveOrUpdate(employers);

tx.commit();

}

public void updateEmployersBySuper(Employers employers,String grade,String isWork){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

employers.setIsWork(isWork);

employers.setGrade(grade);

session.saveOrUpdate(employers);

tx.commit();

}

public void deleteEmployers(Integer id){

Session session = (Session) HibernateSessionFactory.getSessionFactory().getCurrentSession();

Transaction tx = session.beginTransaction();

Employers Employers = (Employers)session.get(Employers.class, id);

session.delete(Employers);

tx.commit();

}

}

Charge.java

package Hibernate;

import java.sql.Timestamp;

/\*\*

\* Charge entity. @author MyEclipse Persistence Tools

\*/

public class Charge implements java.io.Serializable {

// Fields

private Integer cardId;

private Users users;

private String cardType;

private Double cost;

private Timestamp starttime;

private Timestamp endtime;

private Double balance;

// Constructors

/\*\* default constructor \*/

public Charge() {

}

/\*\* full constructor \*/

public Charge(Users users, String cardType, Double cost,

Timestamp starttime, Timestamp endtime, Double balance) {

this.users = users;

this.cardType = cardType;

this.cost = cost;

this.starttime = starttime;

this.endtime = endtime;

this.balance = balance;

}

// Property accessors

public Charge(Users users, String cardType, Timestamp starttime) {

// TODO Auto-generated constructor stub

this.users = users;

this.cardType=cardType;

this.starttime = starttime;

}

public Integer getCardId() {

return this.cardId;

}

public void setCardId(Integer cardId) {

this.cardId = cardId;

}

public Integer getId(){

return this.users.getId();

}

public Users getUsers() {

return this.users;

}

public void setUsers(Users users) {

this.users = users;

}

public String getCardType() {

return this.cardType;

}

public void setCardType(String cardType) {

this.cardType = cardType;

}

public Double getCost() {

return this.cost;

}

public void setCost(Double cost) {

this.cost = cost;

}

public Timestamp getStarttime() {

return this.starttime;

}

public void setStarttime(Timestamp starttime) {

this.starttime = starttime;

}

public Timestamp getEndtime() {

return this.endtime;

}

public void setEndtime(Timestamp endtime) {

this.endtime = endtime;

}

public Double getBalance() {

return this.balance;

}

public void setBalance(Double balance) {

this.balance = balance;

}

}