* Hibernate bidirectional one to many is created using @**ManyToOne** annotation on the owner side, the foreign column is added using @JoinColumn on owner side.
* **@OnetoMany** annotation is used on inverse side used with **mappedBy** attribute referencing owner side reference.
* **fetch=FetchType.LAZY**, cascade=**CascadeType.PERSIST** is used to for lazy loading and cascadeType.persist for cascade

|  |
| --- |
| Implementation Details |

package org.learnhibernatein5days.on.java.model;

import java.io.Serializable;

import java.util.HashSet;

import java.util.Set;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.OneToMany;

@Entity

public class User implements Serializable {

private static final long serialVersionUID = 5313493413859894403L;

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "id", updatable = false, nullable = false)

private long id;

@Column(nullable=false)

private String userId;

@Column(nullable=false, length=50)

private String firstName;

@Column(nullable=false, length=50)

private String lastName;

@Column(nullable=false, length=120,unique=true)

private String email;

@Column(nullable=false )

private Boolean emailVerificationStatus = false;

@OneToMany(mappedBy="userDetails", cascade=CascadeType.PERSIST,fetch=FetchType.LAZY)

private Set<Address> addresses;

public long getId() {

return id;

}

public User() {

this.addresses = new HashSet<Address>();

}

public void setId(long id) {

this.id = id;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

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 String getEmail() {

return email;

}

public void setEmail(String email) {

this.email = email;

}

public Boolean getEmailVerificationStatus() {

return emailVerificationStatus;

}

public void setEmailVerificationStatus(Boolean emailVerificationStatus) {

this.emailVerificationStatus = emailVerificationStatus;

}

public Set<Address> getAddresses() {

return addresses;

}

public void setAddresses(Set<Address> addresses) {

this.addresses = addresses;

}

//Constructors, getters and setters removed for brevity

public void addAddress(Address address) {

addresses.add(address);

address.setUserDetails(this);

}

public void removeAddress(Address address) {

addresses.remove(address);

address.setUserDetails(null);

}

@Override

public String toString() {

StringBuilder builder = new StringBuilder();

builder.append("User [id=");

builder.append(id);

builder.append(", userId=");

builder.append(userId);

builder.append(", firstName=");

builder.append(firstName);

builder.append(", lastName=");

builder.append(lastName);

builder.append(", email=");

builder.append(email);

builder.append(", emailVerificationStatus=");

builder.append(emailVerificationStatus);

builder.append("]");

return builder.toString();

}

}

package org.learnhibernatein5days.on.java.model;

import java.io.Serializable;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.ManyToOne;

@Entity

public class Address implements Serializable {

private static final long serialVersionUID = 7809200551672852690L;

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

@Column(name = "id", updatable = false, nullable = false)

private long id;

@Column(length=25, nullable=false)

private String city;

@Column(length=25, nullable=false)

private String country;

@Column(length=75, nullable=false)

private String streetName;

@Column(length=7, nullable=false)

private String postalCode;

@Column(length=10, nullable=false)

private String type;

@ManyToOne(fetch=FetchType.LAZY, cascade=CascadeType.PERSIST)

@JoinColumn(name="users\_id")

private User userDetails;

public long getId() {

return id;

}

public void setId(long id) {

this.id = id;

}

public String getCity() {

return city;

}

public void setCity(String city) {

this.city = city;

}

public String getCountry() {

return country;

}

public void setCountry(String country) {

this.country = country;

}

public String getStreetName() {

return streetName;

}

public void setStreetName(String streetName) {

this.streetName = streetName;

}

public String getPostalCode() {

return postalCode;

}

public void setPostalCode(String postalCode) {

this.postalCode = postalCode;

}

public String getType() {

return type;

}

public void setType(String type) {

this.type = type;

}

public User getUserDetails() {

return userDetails;

}

public void setUserDetails(User userDetails) {

this.userDetails = userDetails;

}

@Override

public String toString() {

StringBuilder builder = new StringBuilder();

builder.append("Address [id=");

builder.append(id);

builder.append(", city=");

builder.append(city);

builder.append(", country=");

builder.append(country);

builder.append(", streetName=");

builder.append(streetName);

builder.append(", postalCode=");

builder.append(postalCode);

builder.append(", type=");

builder.append(type);

builder.append("]");

return builder.toString();

}

}

log.info("... bidirectional One To Many ...");

// Add a new Review

EntityManager em = emf.createEntityManager();

em.getTransaction().begin();

User user = **new** User();

user.setUserId("nilmoksh2929");

user.setFirstName("Bhavesh");

user.setLastName("Patel");

user.setEmail("emailtobhavesh@gmail.com");

user.setEmailVerificationStatus(**false**);

Address address1 = **new** Address();

address1.setType("shipping");

address1.setCity("Milton");

address1.setCountry("Canada");

address1.setPostalCode("L9T3A3");

address1.setStreetName("861 Scott Blvd");

Address address2 = **new** Address();

address2.setType("billling");

address2.setCity("Brampton");

address2.setCountry("Canada");

address2.setPostalCode("L9T3A3");

address2.setStreetName("861 Scott Blvd");

address1.setUserDetails(user);

address2.setUserDetails(user);

user.addAddress(address1);

user.addAddress(address2);

em.persist(user);

em.getTransaction().commit();

em.close();

|  |
| --- |
| Query |

log.info("... bidirectionalOneToMany ...");

// Add a new Review

EntityManager em = emf.createEntityManager();

em.getTransaction().begin();

/\* Query one \*/

// Get User entity

User user = **null**;

user =em.find(User.**class**,1L);

System.***out***.println(user);

/\*\*\*\*\*\*\*\*\*\*\*\* identifying the N + 1 problem \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

System.***out***.println("/\*\*\*\*\*\*\*\*\*\*\*\* identifying the N + 1 problem \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/\r\n" + " ") ;

List<Address> addressListOne = em.createQuery( "select address from Address address").getResultList();

**for**(Address address\_1 : addressListOne )

{

**if**(address\_1 !=**null**)

{

System.***out***.println(address\_1.getStreetName() + " and user" + address\_1.getUserDetails().getFirstName());

}

}

/\*\*\*\*\*\*\*\*\*\*\*\* Resolving the N + 1 problem \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

System.***out***.println("/\*\*\*\*\*\*\*\*\*\*\*\* resolving the N + 1 problem \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/\r\n" + " ") ;

List<Address> addressList = em.createQuery( "select address from Address address left join fetch address.userDetails").getResultList();

**for**(Address address: addressList )

{

**if**(address !=**null**)

{

System.***out***.println(address.getStreetName() + " and user" + address.getUserDetails());

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\* Query One to Many End \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*88/

\*

\*/

System.***out***.println(" /\*\*\*\*\*\*\*\*\*\*\*\*\*\* Query One to Many End \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*88/\r\n" +

" \* ");

List l = em.createQuery( "SELECT u FROM User u LEFT JOIN Address a ON u.id = a.id WHERE u.firstName = 'Bhavesh'").getResultList();

**for** (Object o: l) {

System.***out***.println(((User)o).getFirstName());

}

TypedQuery<Object[]> query = em.createQuery("SELECT u.firstName,a.streetName FROM User u LEFT JOIN Address a ON a.id = a.id WHERE u.firstName = 'Bhavesh'", Object[].**class**);

List<Object[]> results = query.getResultList();

**for** (Object[] result : results)

{

System.***out***.println( "Title: " + result[0] + ", Title: " + result[1]);

}

em.getTransaction().commit();

em.close();

|  |
| --- |
| Pointer for Better Performance |