**Day4\_Assignments(Hibernate)**

Day 4: Object Relational Mapping and Hibernate - Database Integration for Claims and Policies

Task 1: Define Hibernate entity mappings for claim and policy data models.

Task 2: Develop Hibernate DAOs to handle CRUD operations for claims and policies.

Task 3: Write and test HQL and Criteria queries for advanced data retrieval and reporting.

**Solution**:

**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 https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-claims-policies</artifactId>

<version>5.4.5.Final</version>

<dependencies>

<!-- Hibernate Core -->

<dependency>

<groupId>org.hibernate.orm</groupId>

<artifactId>hibernate-core</artifactId>

<version>6.0.0.Final</version>

</dependency>

<!-- MySQL Connector -->

<dependency>

<groupId>mysql</groupId>

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

<version>8.0.32</version>

</dependency>

<!-- Jakarta Persistence API -->

<dependency>

<groupId>jakarta.persistence</groupId>

<artifactId>jakarta.persistence-api</artifactId>

<version>3.0.0</version>

</dependency>

<!-- JUnit 5 for Testing -->

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter</artifactId>

<version>5.9.3</version>

<scope>test</scope>

</dependency>

<!-- Hibernate Validator (optional, for validation) -->

<dependency>

<groupId>org.hibernate.validator</groupId>

<artifactId>hibernate-validator</artifactId>

<version>6.2.0.Final</version>

</dependency>

<!-- SLF4J for Logging -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-api</artifactId>

<version>1.7.36</version>

</dependency>

<!-- SLF4J Simple Binding (optional, for simple logging implementation) -->

<dependency>

<groupId>org.slf4j</groupId>

<artifactId>slf4j-simple</artifactId>

<version>1.7.36</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

**HibernateClaimsPoliciesExample Class**

package com.example;

import com.example.dao.ClaimDao;

import com.example.dao.PolicyDao;

import com.example.entity.Claim;

import com.example.entity.Policy;

import com.example.util.HibernateUtil;

import jakarta.persistence.\*;

import java.util.Date;

import java.util.List;

public class HibernateClaimsPoliciesExample {

public static void main(String[] args) {

PolicyDao policyDao = new PolicyDao();

ClaimDao claimDao = new ClaimDao();

// Create Policy

Policy policy = new Policy();

policy.setPolicyNumber("P123456");

policy.setPolicyType("Health");

policyDao.savePolicy(policy);

// Create Claim

Claim claim = new Claim();

claim.setClaimNumber("C123456");

claim.setClaimDate(new Date());

claim.setClaimAmount(1000.00);

claim.setPolicy(policy);

claimDao.saveClaim(claim);

// Read Policies

List<Policy> policies = policyDao.getPolicies();

policies.forEach(System.out::println);

// Read Claims

List<Claim> claims = claimDao.getClaims();

claims.forEach(System.out::println);

// Update Policy

policy.setPolicyType("Life");

policyDao.updatePolicy(policy);

// Delete Claim

claimDao.deleteClaim(claim.getId());

// Shutdown Hibernate

HibernateUtil.shutdown();

}

**ClaimDao Class**

package com.example.dao;

import com.example.entity.Claim;

import com.example.util.HibernateUtil;

import org.hibernate.Session;

import org.hibernate.Transaction;

import java.util.List;

public class ClaimDao {

public void saveClaim(Claim claim) {

Transaction transaction = null;

try (Session session = HibernateUtil.getSessionFactory().openSession()) {

transaction = session.beginTransaction();

session.save(claim);

transaction.commit();

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

}

}

public List<Claim> getClaims() {

try (Session session = HibernateUtil.getSessionFactory().openSession()) {

return session.createQuery("FROM Claim", Claim.class).list();

}

}

public void updateClaim(Claim claim) {

Transaction transaction = null;

try (Session session = HibernateUtil.getSessionFactory().openSession()) {

transaction = session.beginTransaction();

session.update(claim);

transaction.commit();

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

}

}

public void deleteClaim(Long claimId) {

Transaction transaction = null;

try (Session session = HibernateUtil.getSessionFactory().openSession()) {

transaction = session.beginTransaction();

Claim claim = session.get(Claim.class, claimId);

if (claim != null) {

session.delete(claim);

}

transaction.commit();

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

}

}

}

}

**PolicyDao Class**

package com.example.dao;

import com.example.entity.Policy;

import com.example.util.HibernateUtil;

import org.hibernate.Session;

import org.hibernate.Transaction;

import java.util.List;

public class PolicyDao {

public void savePolicy(Policy policy) {

Transaction transaction = null;

try (Session session = HibernateUtil.getSessionFactory().openSession()) {

transaction = session.beginTransaction();

session.save(policy);

transaction.commit();

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

}

}

public List<Policy> getPolicies() {

try (Session session = HibernateUtil.getSessionFactory().openSession()) {

return session.createQuery("FROM Policy", Policy.class).list();

}

}

public void updatePolicy(Policy policy) {

Transaction transaction = null;

try (Session session = HibernateUtil.getSessionFactory().openSession()) {

transaction = session.beginTransaction();

session.update(policy);

transaction.commit();

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

}

}

public void deletePolicy(Long policyId) {

Transaction transaction = null;

try (Session session = HibernateUtil.getSessionFactory().openSession()) {

transaction = session.beginTransaction();

Policy policy = session.get(Policy.class, policyId);

if (policy != null) {

session.delete(policy);

}

transaction.commit();

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

}

}

}

**Claim Class**

package com.example.entity;

import jakarta.persistence.\*;

import java.util.Date;

@Entity

@Table(name = "claim")

public class Claim {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(name = "claim\_number", unique = true, nullable = false)

private String claimNumber;

@Column(name = "claim\_date")

@Temporal(TemporalType.DATE)

private Date claimDate;

@Column(name = "claim\_amount")

private Double claimAmount;

@ManyToOne

@JoinColumn(name = "policy\_id", nullable = false)

private Policy policy;

// Constructors

public Claim() {}

public Claim(String claimNumber, Date claimDate, Double claimAmount, Policy policy) {

this.claimNumber = claimNumber;

this.claimDate = claimDate;

this.claimAmount = claimAmount;

this.policy = policy;

}

// Getters and Setters

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getClaimNumber() {

return claimNumber;

}

public void setClaimNumber(String claimNumber) {

this.claimNumber = claimNumber;

}

public Date getClaimDate() {

return claimDate;

}

public void setClaimDate(Date claimDate) {

this.claimDate = claimDate;

}

public Double getClaimAmount() {

return claimAmount;

}

public void setClaimAmount(Double claimAmount) {

this.claimAmount = claimAmount;

}

public Policy getPolicy() {

return policy;

}

public void setPolicy(Policy policy) {

this.policy = policy;

}

@Override

public String toString() {

return "Claim [id=" + id + ", claimNumber=" + claimNumber + ", claimDate=" + claimDate + ", claimAmount=" + claimAmount + ", policy=" + policy + "]";

}

}

**Policy Class**

package com.example.entity;

import jakarta.persistence.\*

@Entity

@Table(name = "policy")

public class Policy {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long i;

@Column(name = "policy\_number", unique = true, nullable = false)

private String policyNumber;

@Column(name = "policy\_type")

private String policyType;

// Constructors

public Policy() {}

public Policy(String policyNumber, String policyType) {

this.policyNumber = policyNumber;

this.policyType = policyType;

}

// Getters and Setters

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getPolicyNumber() {

return policyNumber;

}

public void setPolicyNumber(String policyNumber) {

this.policyNumber = policyNumber;

}

public String getPolicyType() {

return policyType;

}

public void setPolicyType(String policyType) {

this.policyType = policyType;

}

@Override

public String toString() {

return "Policy [id=" + id + ", policyNumber=" + policyNumber + ", policyType=" + policyType + "]";

}

}

**HibernateUtil Class**

package com.example.util;

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 HibernateUtil {

private static StandardServiceRegistry registry;

private static SessionFactory sessionFactory;

public static SessionFactory getSessionFactory() {

if (sessionFactory == null) {

try {

registry = new StandardServiceRegistryBuilder().configure().build();

MetadataSources sources = new MetadataSources(registry);

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

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

} catch (Exception e) {

e.printStackTrace();

if (registry != null) {

StandardServiceRegistryBuilder.destroy(registry);

}

}

}

return sessionFactory;

}

public static void shutdown() {

if (registry != null) {

StandardServiceRegistryBuilder.destroy(registry);

}

}

}

**Hibernate.cfg.xml**

<!DOCTYPE hibernate-configuration PUBLIC

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

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

<hibernate-configuration>

<session-factory>

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

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/wipro?useSSL=false</property>

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

<property name="hibernate.connection.password">rps@123</property>

<property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>

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

<property name="hibernate.hbm2ddl.auto">update</property>

<mapping class="com.example.entity.Claim"/>

<mapping class="com.example.entity.Policy"/>

</session-factory>

</hibernate-configuration>

**ClaimDaoTest Class**

package com.example.dao;

import com.example.entity.Claim;

import com.example.entity.Policy;

import com.example.util.HibernateUtil;

import org.hibernate.Session;

import org.junit.jupiter.api.\*;

import java.util.Date;

import java.util.List;

import static org.junit.jupiter.api.Assertions.\*;

@TestInstance(TestInstance.Lifecycle.PER\_CLASS)

public class ClaimDaoTest {

private ClaimDao claimDao;

private PolicyDao policyDao;

private Policy testPolicy;

@BeforeAll

public void setup() {

claimDao = new ClaimDao();

policyDao = new PolicyDao();

// Setup a Policy for the Claim

testPolicy = new Policy();

testPolicy.setPolicyNumber("P123456");

testPolicy.setPolicyType("Health");

policyDao.savePolicy(testPolicy);

}

@Test

public void testSaveClaim() {

Claim claim = new Claim();

claim.setClaimNumber("C123456");

claim.setClaimDate(new Date());

claim.setClaimAmount(1000.00);

claim.setPolicy(testPolicy);

claimDao.saveClaim(claim);

// Verify the claim is saved

List<Claim> claims = claimDao.getClaims();

assertEquals(1, claims.size());

assertEquals("C123456", claims.get(0).getClaimNumber());

}

@Test

public void testGetClaims() {

List<Claim> claims = claimDao.getClaims();

assertFalse(claims.isEmpty());

}

@Test

public void testUpdateClaim() {

List<Claim> claims = claimDao.getClaims();

Claim claim = claims.get(0);

claim.setClaimAmount(2000.00);

claimDao.updateClaim(claim);

// Verify the update

Claim updatedClaim = claimDao.getClaims().get(0);

assertEquals(2000.00, updatedClaim.getClaimAmount());

}

@Test

public void testDeleteClaim() {

List<Claim> claims = claimDao.getClaims();

Claim claim = claims.get(0);

claimDao.deleteClaim(claim.getId());

// Verify the claim is deleted

List<Claim> remainingClaims = claimDao.getClaims();

assertTrue(remainingClaims.isEmpty());

}

@AfterAll

public void tearDown() {

policyDao.deletePolicy(testPolicy.getId());

HibernateUtil.shutdown();

}

}

**PolicyDaoTest Class**

package com.example.dao;

import com.example.entity.Policy;

import com.example.util.HibernateUtil;

import org.junit.jupiter.api.\*;

import java.util.List;

import static org.junit.jupiter.api.Assertions.\*;

@TestInstance(TestInstance.Lifecycle.PER\_CLASS)

public class PolicyDaoTest {

private PolicyDao policyDao;

@BeforeAll

public void setup() {

policyDao = new PolicyDao();

}

@Test

public void testSavePolicy() {

Policy policy = new Policy();

policy.setPolicyNumber("P654321");

policy.setPolicyType("Auto");

policyDao.savePolicy(policy);

// Verify the policy is saved

List<Policy> policies = policyDao.getPolicies();

assertEquals(1, policies.size());

assertEquals("P654321", policies.get(0).getPolicyNumber());

}

@Test

public void testGetPolicies() {

List<Policy> policies = policyDao.getPolicies();

assertFalse(policies.isEmpty());

}

@Test

public void testUpdatePolicy() {

List<Policy> policies = policyDao.getPolicies();

Policy policy = policies.get(0);

policy.setPolicyType("Home");

policyDao.updatePolicy(policy);

// Verify the update

Policy updatedPolicy = policyDao.getPolicies().get(0);

assertEquals("Home", updatedPolicy.getPolicyType());

}

@Test

public void testDeletePolicy() {

List<Policy> policies = policyDao.getPolicies();

Policy policy = policies.get(0);

policyDao.deletePolicy(policy.getId());

// Verify the policy is deleted

List<Policy> remainingPolicies = policyDao.getPolicies();

assertTrue(remainingPolicies.isEmpty());

}

@AfterAll

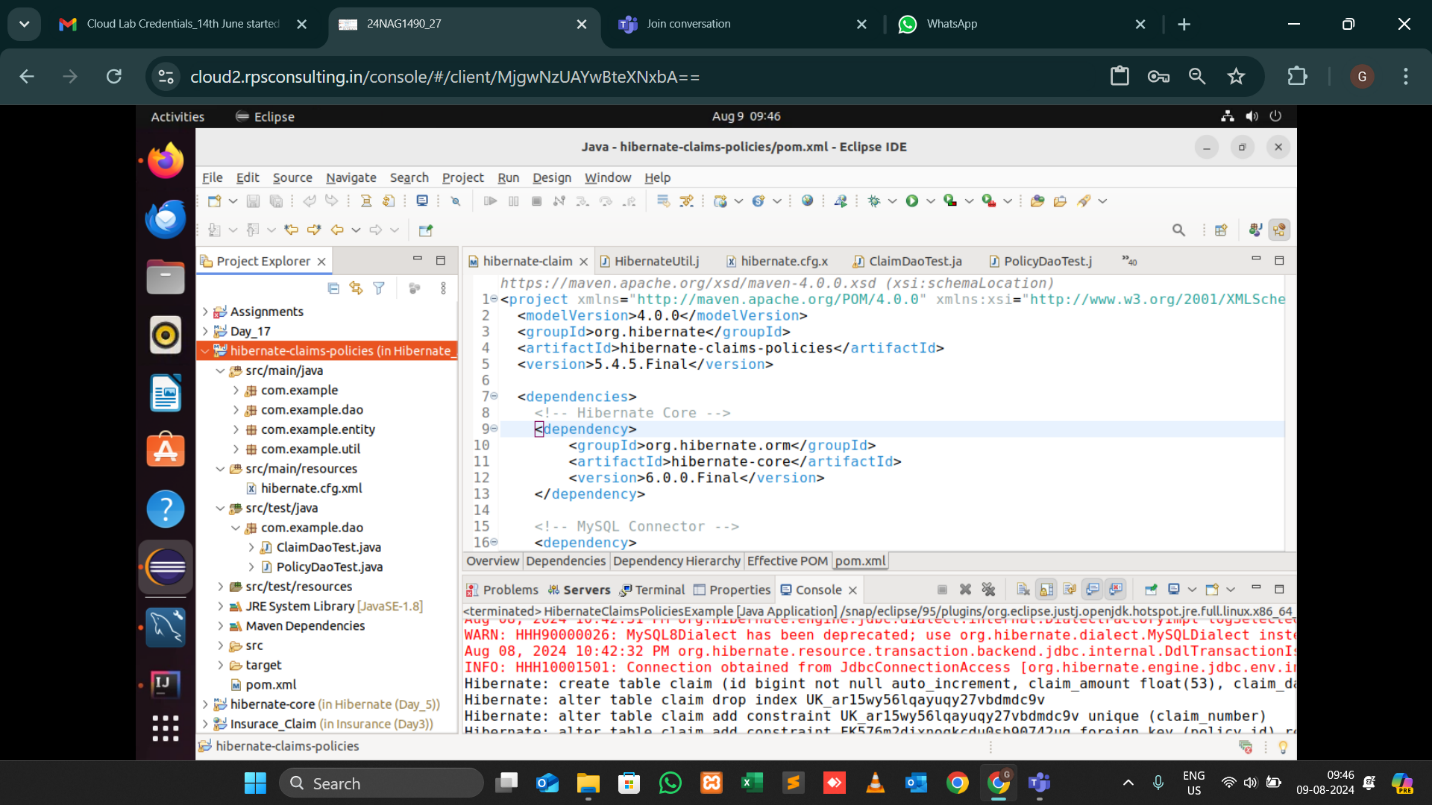
public void tearDown() {

HibernateUtil.shutdown();

}

}

**Folder Structure:**



Output:

Table has been Created with the name “claim”

