JPQL:

STATIC JPQL QUERIES:

READ SINGLE DATA STATIC:

**package** com.te.JpaWithHibernateDemoJpql;

**import** javax.persistence.EntityManager;

**import** javax.persistence.EntityManagerFactory;

**import** javax.persistence.Persistence;

**import** javax.persistence.Query;

**import** com.te.JpaWithHibernateDemo.bean.Movies;

**public** **class** ReadDataUsingJpql {

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

EntityManagerFactory entityManagerFactory = **null**;

EntityManager entityManager = **null**;

**try** {

entityManagerFactory= Persistence.*createEntityManagerFactory*("moviesData");

entityManager= entityManagerFactory.createEntityManager();

String select = "from Movies where id= 100";

Query query = entityManager.createQuery(select);

Movies movie= (Movies) query.getSingleResult();

System.***out***.println(movie.getId());

System.***out***.println(movie.getName());

System.***out***.println(movie.getGenre());

System.***out***.println(movie.getRatings());

} **catch** (Exception e) {

e.printStackTrace();

}

**finally** {

**if**(entityManagerFactory!=**null**) {

entityManagerFactory.close();

}

**if**(entityManager!=**null**) {

entityManager.close();

}

}

}

}

**READ ALL JPQL STATIC:**

package com.tyss.hibernate.jpqlqueries.common;

import java.util.List;

import javax.persistence.EntityManager;

import javax.persistence.EntityManagerFactory;

import javax.persistence.Persistence;

import javax.persistence.Query;

import com.tyss.JpaWithHibernate.bean.Movies;

public class ReadJpqlQueries {

@SuppressWarnings("unchecked")

public static void main(String[] args) {

EntityManagerFactory entityManagerFactory = null;

EntityManager entityManager = null;

try {

entityManagerFactory = Persistence.createEntityManagerFactory("moviesData");

entityManager = entityManagerFactory.createEntityManager();

String select = "from Movies";

Query query = entityManager.createQuery(select);

List<Movies> list = query.getResultList();

for (Movies movie : list) {

System.out.println(movie);

}

} catch (Exception e) {

e.printStackTrace();

} finally {

if (entityManagerFactory != null) {

entityManagerFactory.close();

}

if (entityManager != null) {

entityManager.close();

}

}

}

}

**Update jpql static:**

package com.te.JpaWithHibernateDemoJpql;

import javax.persistence.EntityManager;

import javax.persistence.EntityManagerFactory;

import javax.persistence.EntityTransaction;

import javax.persistence.Persistence;

import javax.persistence.Query;

public class UpdateDataUsingJpql {

public static void main(String[] args) {

EntityManagerFactory entityManagerFactory = null;

EntityManager entityManager = null;

EntityTransaction entityTransaction = null;

try {

entityManagerFactory= Persistence.createEntityManagerFactory("moviesData");

entityManager= entityManagerFactory.createEntityManager();

entityTransaction= entityManager.getTransaction();

entityTransaction.begin();

String select = "update Movies set name = 'Kgf' where id= 200";

Query query = entityManager.createQuery(select);

int result = query.executeUpdate();

entityTransaction.commit();

} catch (Exception e) {

if(entityTransaction!=null) {

entityTransaction.rollback();

}

e.printStackTrace();

}

finally {

if(entityManagerFactory!=null) {

entityManagerFactory.close();

}

if(entityManager!=null) {

entityManager.close();

}

}

}

}

**DELETE DATA STATIC JPQL:**

**package** com.te.JpaWithHibernateDemoJpql;

**import** javax.persistence.EntityManager;

**import** javax.persistence.EntityManagerFactory;

**import** javax.persistence.EntityTransaction;

**import** javax.persistence.Persistence;

**import** javax.persistence.Query;

**public** **class** DeleteDataUsingJpql {

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

EntityManagerFactory entityManagerFactory = **null**;

EntityManager entityManager = **null**;

EntityTransaction entityTransaction = **null**;

**try** {

entityManagerFactory = Persistence.*createEntityManagerFactory*("moviesData");

entityManager = entityManagerFactory.createEntityManager();

entityTransaction = entityManager.getTransaction();

entityTransaction.begin();

String delete = "delete from Movies where id = 100";

Query query = entityManager.createQuery(delete);

**int** result = query.executeUpdate();

entityTransaction.commit();

} **catch** (Exception e) {

**if**(entityTransaction!=**null**) {

entityTransaction.rollback();

}

e.printStackTrace();

}

**finally** {

**if**(entityManagerFactory!=**null**) {

entityManagerFactory.close();

}

**if**(entityManager!=**null**) {

entityManager.close();

}

}

}

}

**READ ALL RECORDS:**

**package** com.te.JpaWithHibernateDemoJpql;

**import** javax.persistence.EntityManager;

**import** javax.persistence.EntityManagerFactory;

**import** javax.persistence.Persistence;

**import** javax.persistence.Query;

**import** com.te.JpaWithHibernateDemo.bean.Movies;

**public** **class** CountAllRecords {

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

EntityManagerFactory entityManagerFactory = **null**;

EntityManager entityManager = **null**;

**try** {

entityManagerFactory = Persistence.*createEntityManagerFactory*("moviesData");

entityManager = entityManagerFactory.createEntityManager();

String select = " Select count(\*) from Movies";

Query query = entityManager.createQuery(select);

Long movie = (Long) query.getSingleResult();

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

} **catch** (Exception e) {

e.printStackTrace();

} **finally** {

**if** (entityManagerFactory != **null**) {

entityManagerFactory.close();

}

**if** (entityManager != **null**) {

entityManager.close();

}

}

}

}

**READ ONLY MOVIE NAMES:**

**package** com.te.JpaWithHibernateDemoJpql;

**import** java.util.List;

**import** javax.persistence.EntityManager;

**import** javax.persistence.EntityManagerFactory;

**import** javax.persistence.Persistence;

**import** javax.persistence.Query;

**public** **class** ReadOnlyMovieNames {

@SuppressWarnings("unchecked")

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

EntityManagerFactory entityManagerFactory = **null**;

EntityManager entityManager = **null**;

**try** {

entityManagerFactory = Persistence.*createEntityManagerFactory*("moviesData");

entityManager = entityManagerFactory.createEntityManager();

String select = "select name from Movies";

Query query = entityManager.createQuery(select);

List<String> list = query.getResultList();

**for** (String movie : list) {

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

}

} **catch** (Exception e) {

e.printStackTrace();

} **finally** {

**if** (entityManagerFactory != **null**) {

entityManagerFactory.close();

}

**if** (entityManager != **null**) {

entityManager.close();

}

}

}

}

DYNAMIC JPQL QUERIES:

UPDATE DYNAMIC:

**package** com.te.JpaWithHibernateDemo.dynamicQueries;

**import** javax.persistence.EntityManager;

**import** javax.persistence.EntityManagerFactory;

**import** javax.persistence.EntityTransaction;

**import** javax.persistence.Persistence;

**import** javax.persistence.Query;

**public** **class** UpdateDataDynamic {

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

EntityManagerFactory entityManagerFactory = **null**;

EntityManager entityManager = **null**;

EntityTransaction entityTransaction = **null**;

**try** {

entityManagerFactory = Persistence.*createEntityManagerFactory*("moviesData");

entityManager = entityManagerFactory.createEntityManager();

entityTransaction = entityManager.getTransaction();

entityTransaction.begin();

String update = "update Movies set name=:name, ratings=:ratings where id=:id";

Query query = entityManager.createQuery(update);

query.setParameter("name", args[0]);

query.setParameter("ratings", Double.*parseDouble*(args[1]));

query.setParameter("id", Integer.*parseInt*(args[2]));

**int** result = query.executeUpdate();

System.***out***.println(result + " no. of rows affected");

entityTransaction.commit();

} **catch** (Exception e) {

**if** (entityTransaction != **null**) {

entityTransaction.rollback();

}

e.printStackTrace();

} **finally** {

**if** (entityManagerFactory != **null**) {

entityManagerFactory.close();

}

**if** (entityManager != **null**) {

entityManager.close();

}

}

}

}

**READ DYNAMIC:**

package com.te.JpaWithHibernateDemo.dynamicQueries;

import javax.persistence.EntityManager;

import javax.persistence.EntityManagerFactory;

import javax.persistence.EntityTransaction;

import javax.persistence.Persistence;

import javax.persistence.Query;

import com.te.JpaWithHibernateDemo.bean.Movies;

public class ReadDataDynamic {

public static void main(String[] args) {

EntityManagerFactory entityManagerFactory = null;

EntityManager entityManager = null;

EntityTransaction transaction = null;

try {

entityManagerFactory= Persistence.createEntityManagerFactory("moviesData");

entityManager= entityManagerFactory.createEntityManager();

transaction = entityManager.getTransaction();

transaction.begin();

String read= "from Movies where id= :id";

Query query = entityManager.createQuery(read);

query.setParameter("id", Integer.parseInt(args[0]));

Movies movies = (Movies) query.getSingleResult();

System.out.println("Movie ID= "+ movies.getId());

System.out.println("Movie Name= "+movies.getName());

System.out.println("Movie Reviews= "+movies.getGenre());

System.out.println("Movie Rating="+movies.getRatings());

transaction.commit();

} catch (Exception e) {

if(transaction!=null) {

transaction.rollback();

}

e.printStackTrace();

}

finally {

if(entityManagerFactory!=null) {

entityManagerFactory.close();

}

if(entityManager!=null) {

entityManager.close();

}

}

}

}

-----------------------------------------

**DELETE DATA DYNAMIC:**

package com.tyss.hibernate.dynamicqueries.common;

import javax.persistence.EntityManager;

import javax.persistence.EntityManagerFactory;

import javax.persistence.EntityTransaction;

import javax.persistence.Persistence;

import javax.persistence.Query;

public class DeleteDataDynamic {

public static void main(String[] data) {

EntityManagerFactory entityManagerFactory = null;

EntityManager entityManager = null;

EntityTransaction entityTransaction = null;

try {

entityManagerFactory = Persistence.createEntityManagerFactory("moviesData");

entityManager = entityManagerFactory.createEntityManager();

entityTransaction = entityManager.getTransaction();

entityTransaction.begin();

String delete = "delete from Movies where id=:id";

Query query = entityManager.createQuery(delete);

query.setParameter("id", Integer.parseInt(data[0]));

int result = query.executeUpdate();

System.out.println(result + " no. of rows deleted");

entityTransaction.commit();

} catch (Exception e) {

if (entityTransaction != null) {

entityTransaction.rollback();

}

e.printStackTrace();

} finally {

if (entityManagerFactory != null) {

entityManagerFactory.close();

}

if (entityManager != null) {

entityManager.close();

}

}

}

}