HIBERNATE & JPA COURSE

JAVA AND HQL / JPQL QUERIES

By the expert: Eng. Ubaldo Acosta

Global Mentoring

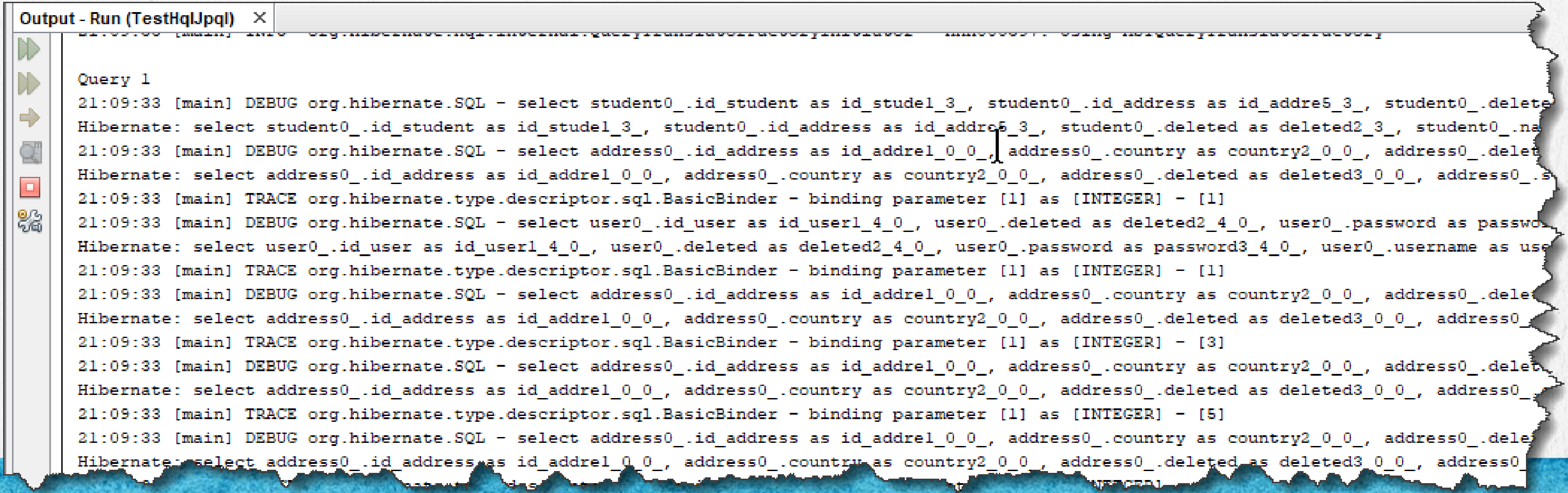
JAVA ONLINE COURSES UNIVERSITY

HIBERNATE & JPA COURSE

www.globalmentoring.com.mx

EXERCISE OBJECTIVE

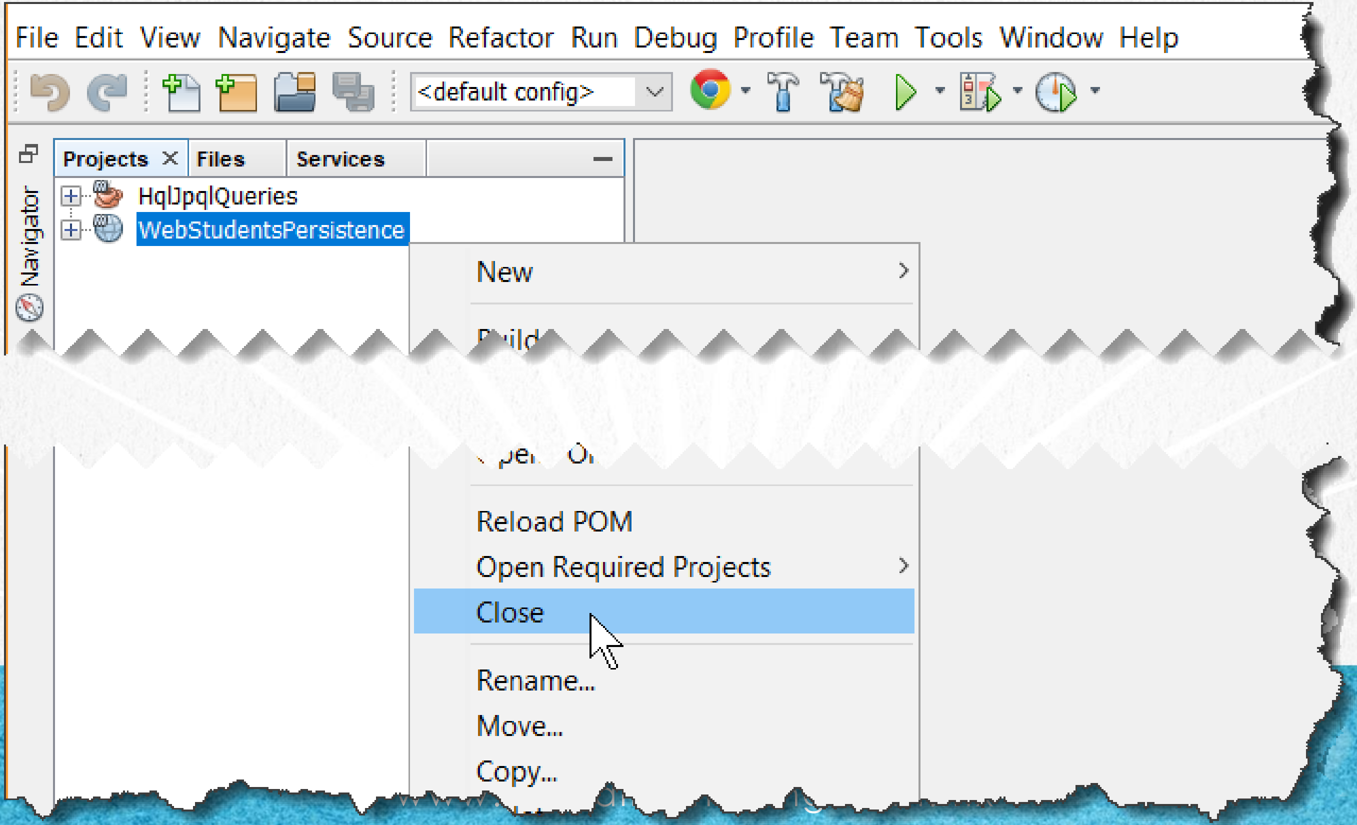
Create the necessary Java code to execute the HQL / JPQL queries. At the end we should see:



```
Query 1
21:09:33 [main] DEBUG org.hibernate.SQL - select student0_.id_student as id_stude1_3_, student0_.id_address as id_addre5_3_, student0_.delete
Hibernate: select student0_.id_student as id_stude1_3_, student0_.id_address as id_addre5_3_, student0_.deleted as deleted2_3_, student0_.na
21:09:33 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.delet
Hibernate: select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.s
21:09:33 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
21:09:33 [main] DEBUG org.hibernate.SQL - select user0_.id_user as id_user1_4_0_, user0_.deleted as deleted2_4_0_, user0_.password as passwo
Hibernate: select user0_.id_user as id_user1_4_0_, user0_.deleted as deleted2_4_0_, user0_.password as password3_4_0_, user0_.username as use
21:09:33 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
21:09:33 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.delet
Hibernate: select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_
21:09:33 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [3]
21:09:33 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.delet
Hibernate: select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_
21:09:33 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [5]
21:09:33 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.dele
Hibernate: select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_
```

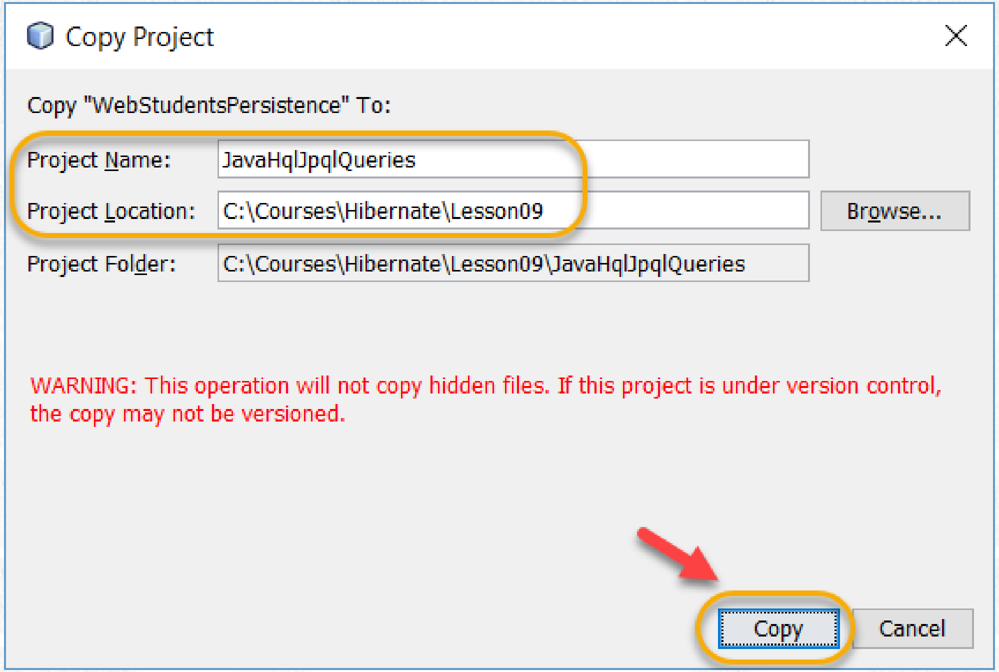
1. COPY THE PROJECT

Copy the Project WebStudentsPersistence:



1. COPY THE PROJECT

Create the project JavaHqlJpqlQueries:



Copy Project

Copy "WebStudentsPersistence" To:

Project Name: JavaHqlJpqlQueries

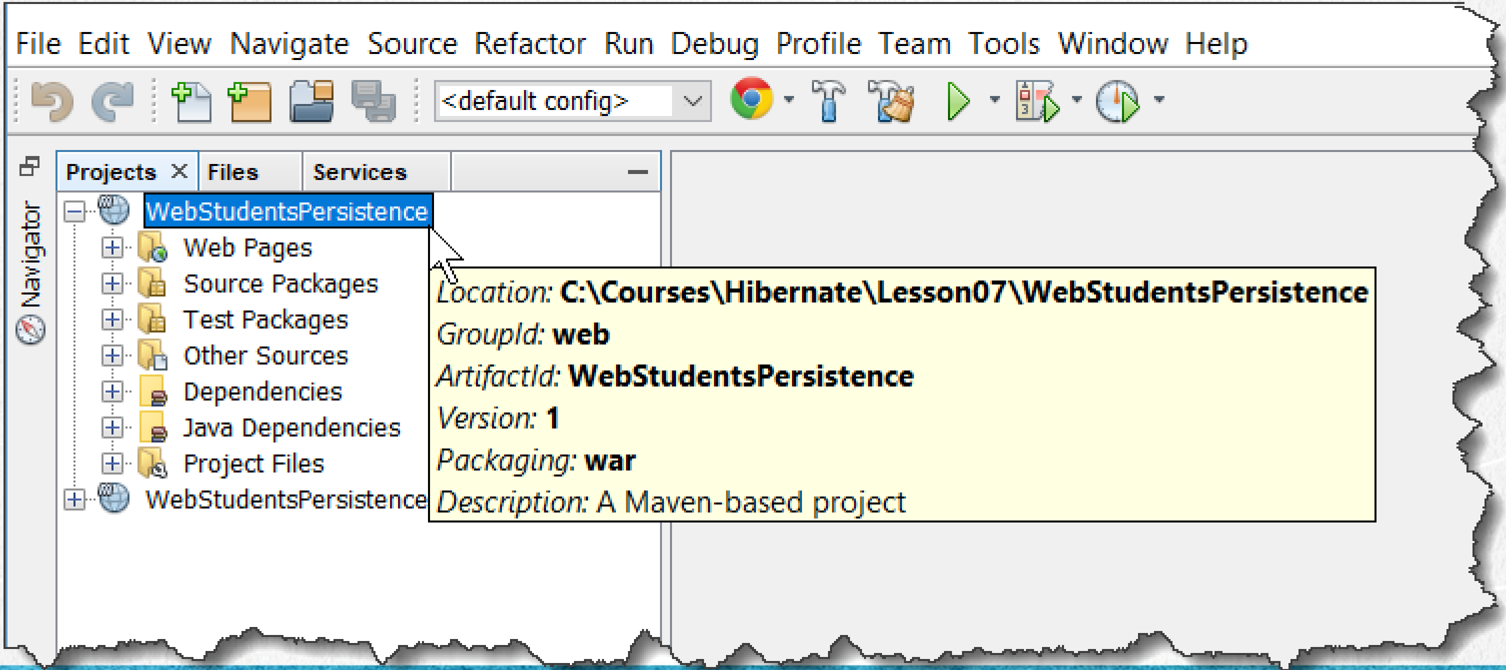
Project Location: C:\Courses\Hibernate\Lesson09

Project Folder: C:\Courses\Hibernate\Lesson09\JavaHqlJpqlQueries

WARNING: This operation will not copy hidden files. If this project is under version control, the copy may not be versioned.

2. CLOSE THE PROJECT

We close the project that we no longer use:



Location: **C:\Courses\Hibernate\Lesson07\WebStudentsPersistence**

GroupId: **web**

ArtifactId: **WebStudentsPersistence**

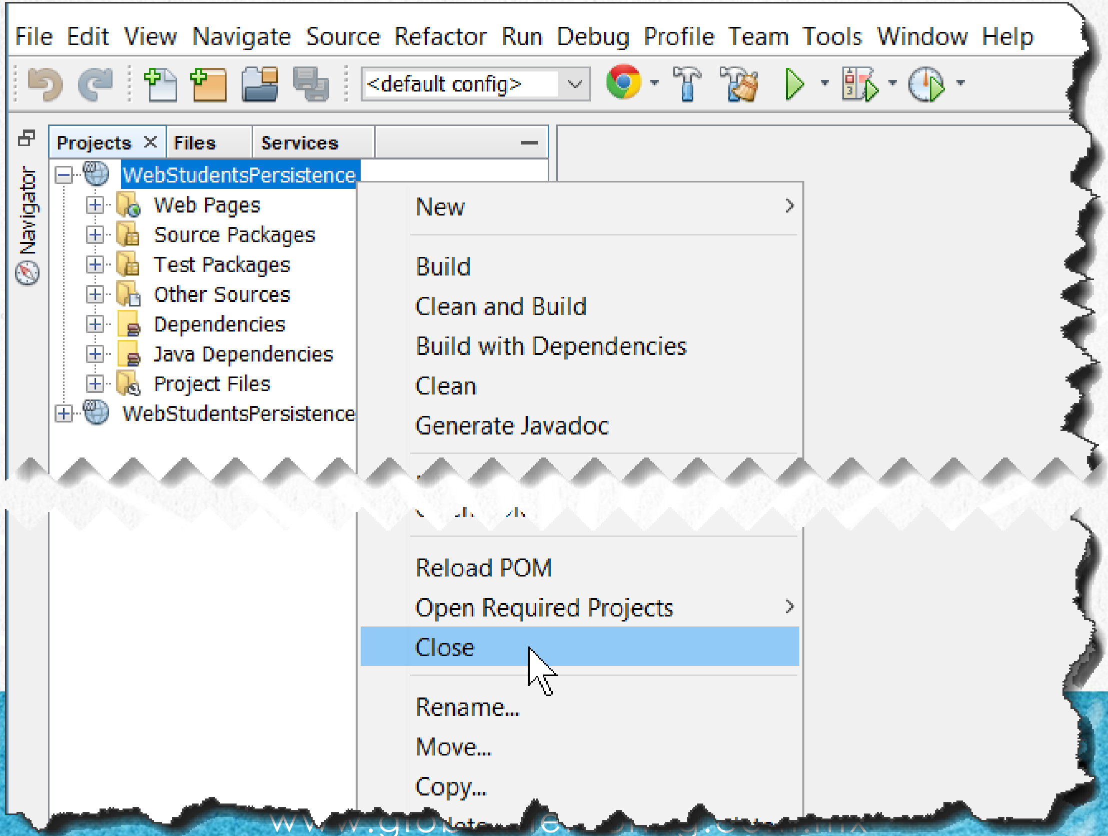
Version: **1**

Packaging: **war**

Description: A Maven-based project

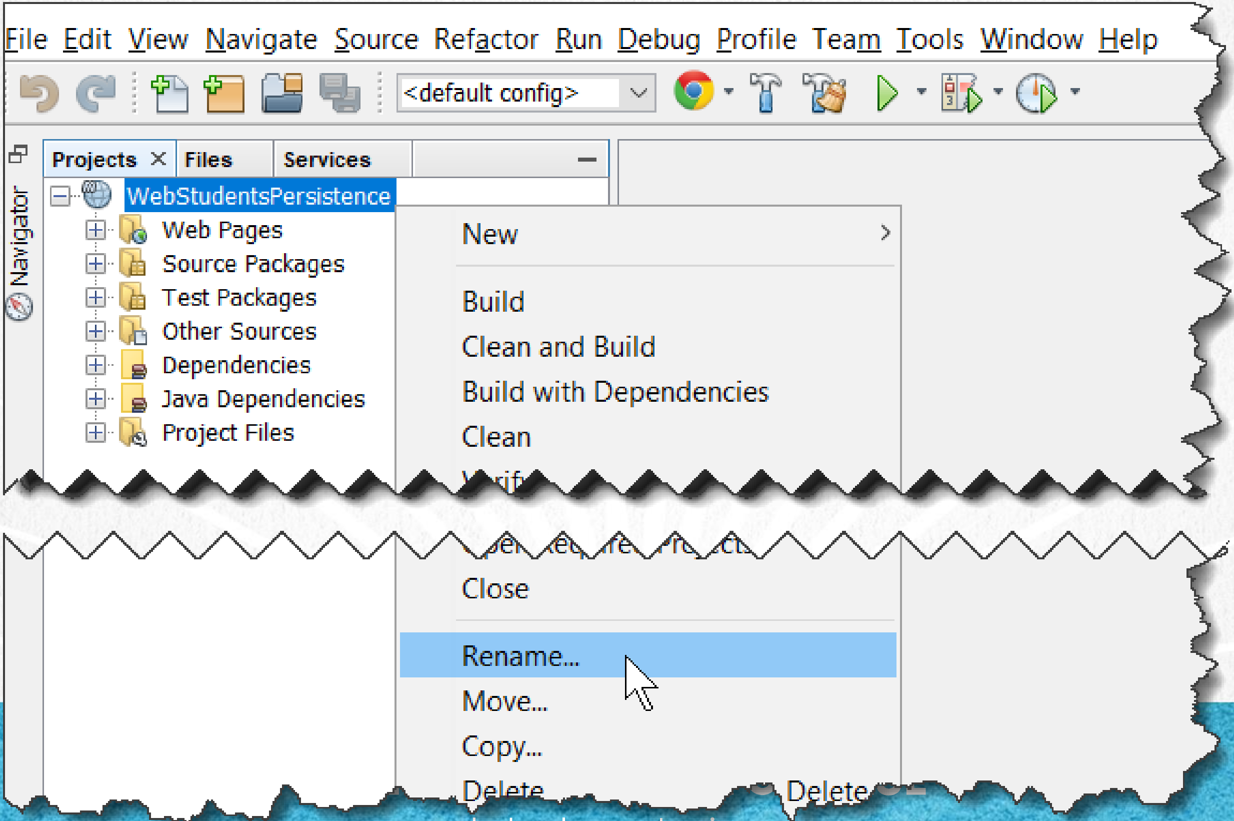
2. CLOSE THE PROJECT

We close the project that we no longer use:



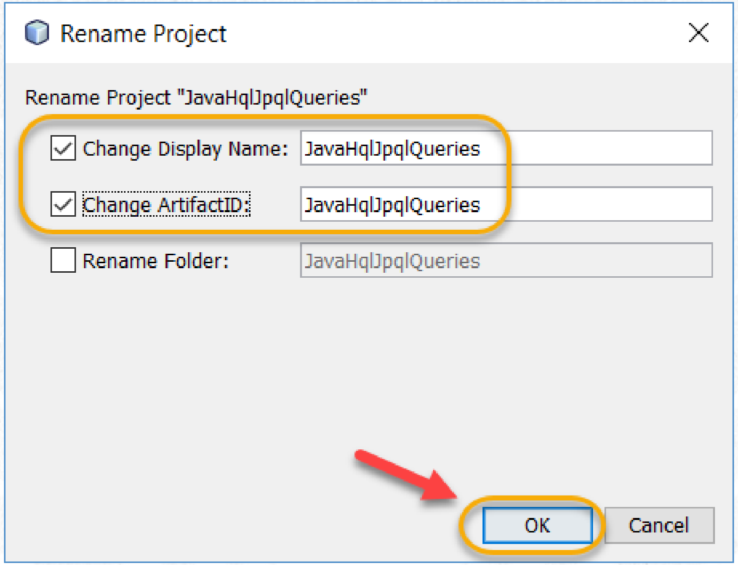
3. RENAME THE PROJECT

Rename the project:



3. RENAME THE PROJECT

Rename the project:



Rename Project

Rename Project "JavaHqlJpqlQueries"

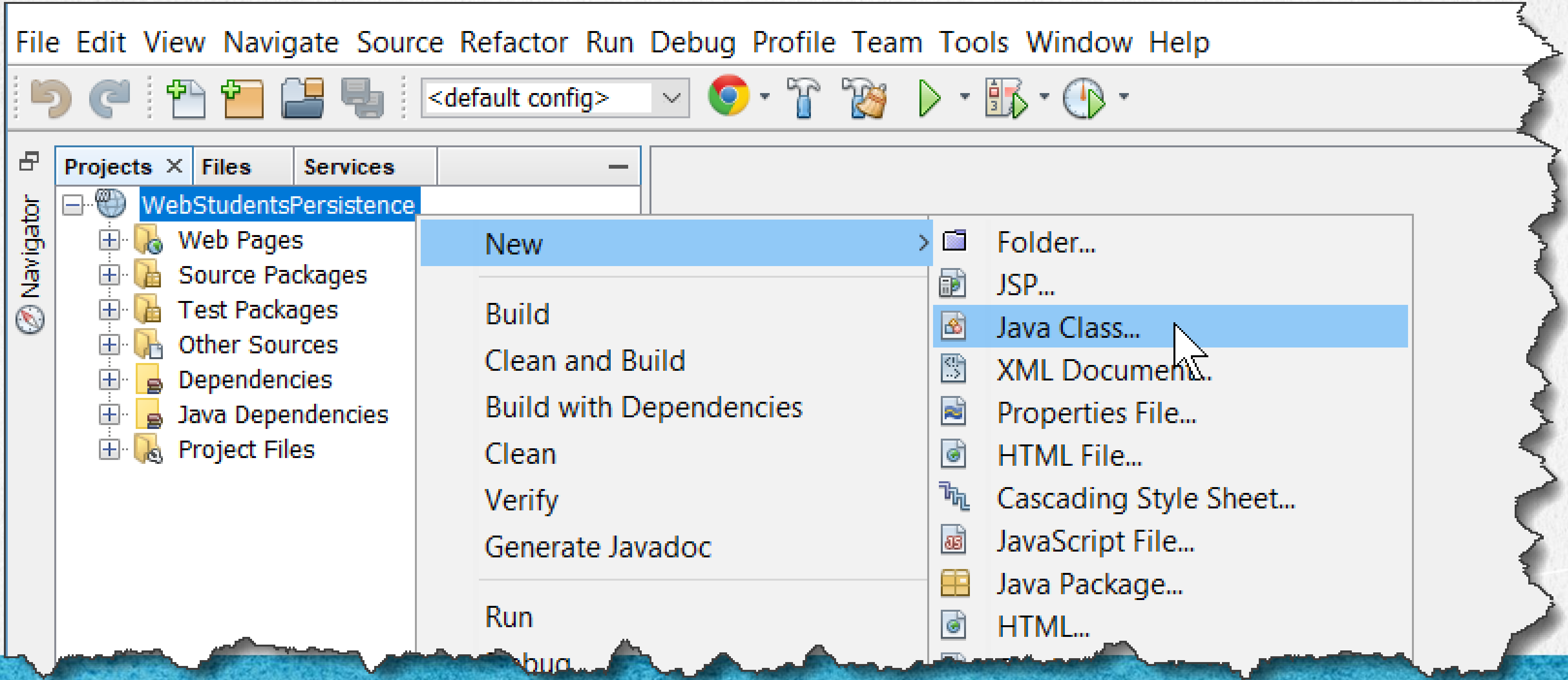
☑ Change Display Name: JavaHqlJpqlQueries

☑ Change ArtifactID: JavaHqlJpqlQueries

□ Rename Folder: JavaHqlJpqlQueries

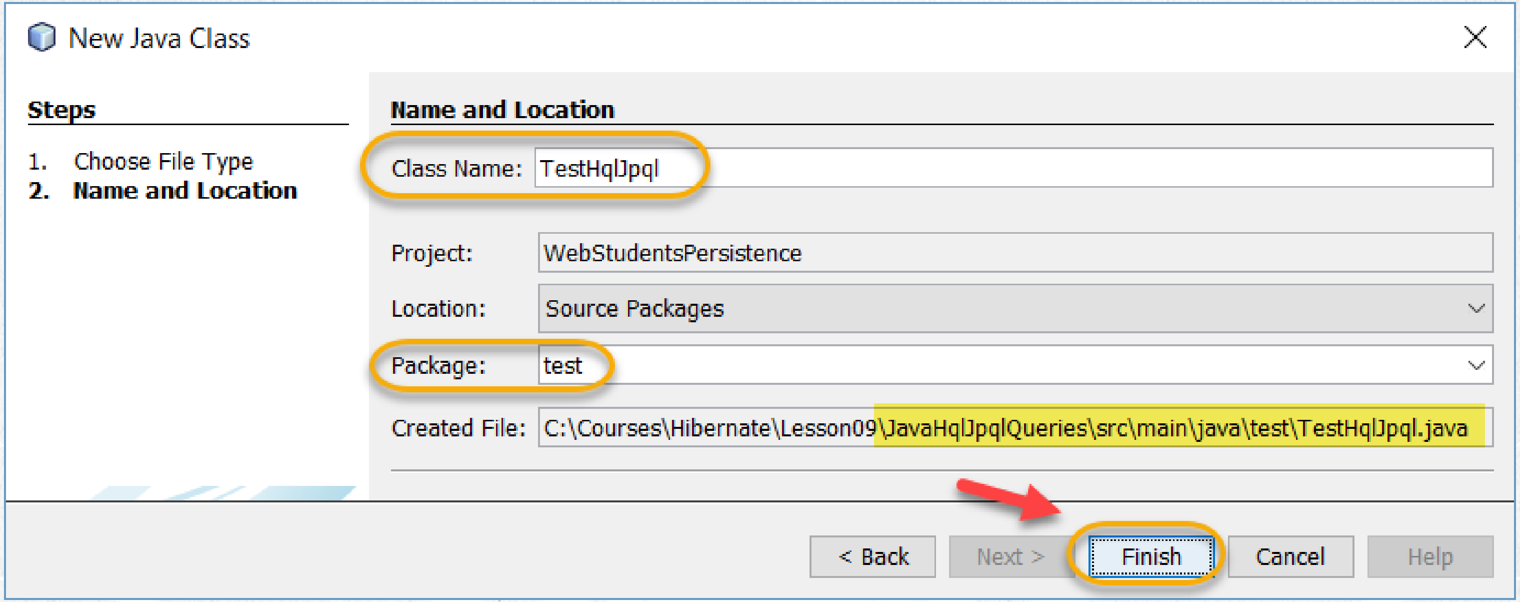
4. CREATE A NEW CLASS

We create a Java class:



4. CREATE A NEW CLASS

We create a Java class:



New Java Class

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name: TestHqlJpql

Project: WebStudentsPersistence

Location: Source Packages

Package: test

Created File: C:\Courses\Hibernate\Lesson09\JavaHqlJpqlQueries\src\main\java\test\TestHqlJpql.java

5. MODIFY THE FILE

TestHqlJpql.java:

Click to download

```
package test;

import java.util.*;
import javax.persistence.*;
import model.*;

public class TestHqlJpql {

    public static void main(String[] args) {
        EntityManagerFactory fabrica = Persistence.createEntityManagerFactory("HibernateJpaPU");
        EntityManager em = fabrica.createEntityManager();

        //Auxiliary variables to execute the queries
        String queryString = null;
        Query query = null;
        List<Student> students = null;
        Student student = null;
        Iterator iter = null;
        Object[] tuple = null;
```

5. MODIFY THE FILE

TestHqlJpql.java:

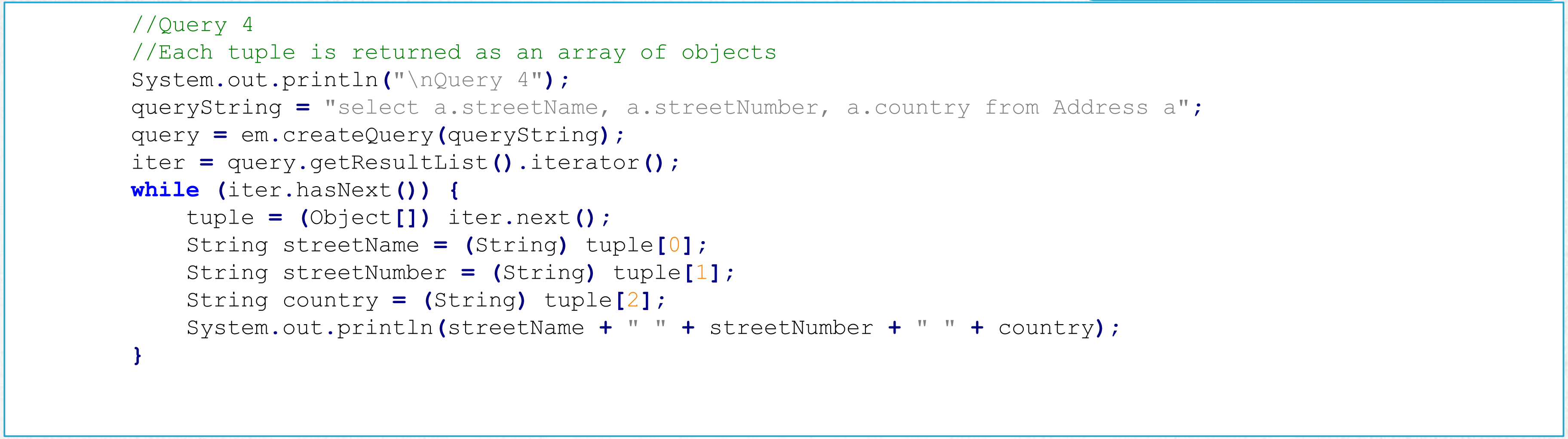
Click to download

```
        //Query 1
        System.out.println("\nQuery 1");
        queryString = "from Student a";
        query = em.createQuery(queryString);
        students = query.getResultList();
        //We print to all Student-type objects
        for (Student s : students) {
            System.out.println(s);
        }

        //Query2
        System.out.println("\nQuery 2");
        queryString = "from Student s where s.idStudent = 2";
        query = em.createQuery(queryString);
        student = (Student) query.getSingleResult();
        System.out.println(student);

        //Query 3
        System.out.println("\nQuery 3");
        queryString = "from Student s where s.name = 'Charly'";
        query = em.createQuery(queryString);
        students = query.getResultList();
        for (Student s : students) {
            System.out.println(s);
        }
```

5. MODIFY THE FILE

TestHqlJpql.java:

Click to download



```
//Query 4
//Each tuple is returned as an array of objects
System.out.println("\nQuery 4");
queryString = "select a.streetName, a.streetNumber, a.country from Address a";
query = em.createQuery(queryString);
iter = query.getResultList().iterator();
while (iter.hasNext()) {
    tuple = (Object[]) iter.next();
    String streetName = (String) tuple[0];
    String streetNumber = (String) tuple[1];
    String country = (String) tuple[2];
    System.out.println(streetName + " " + streetNumber + " " + country);
}
```

5. MODIFY THE FILE

TestHqlJpql.java:

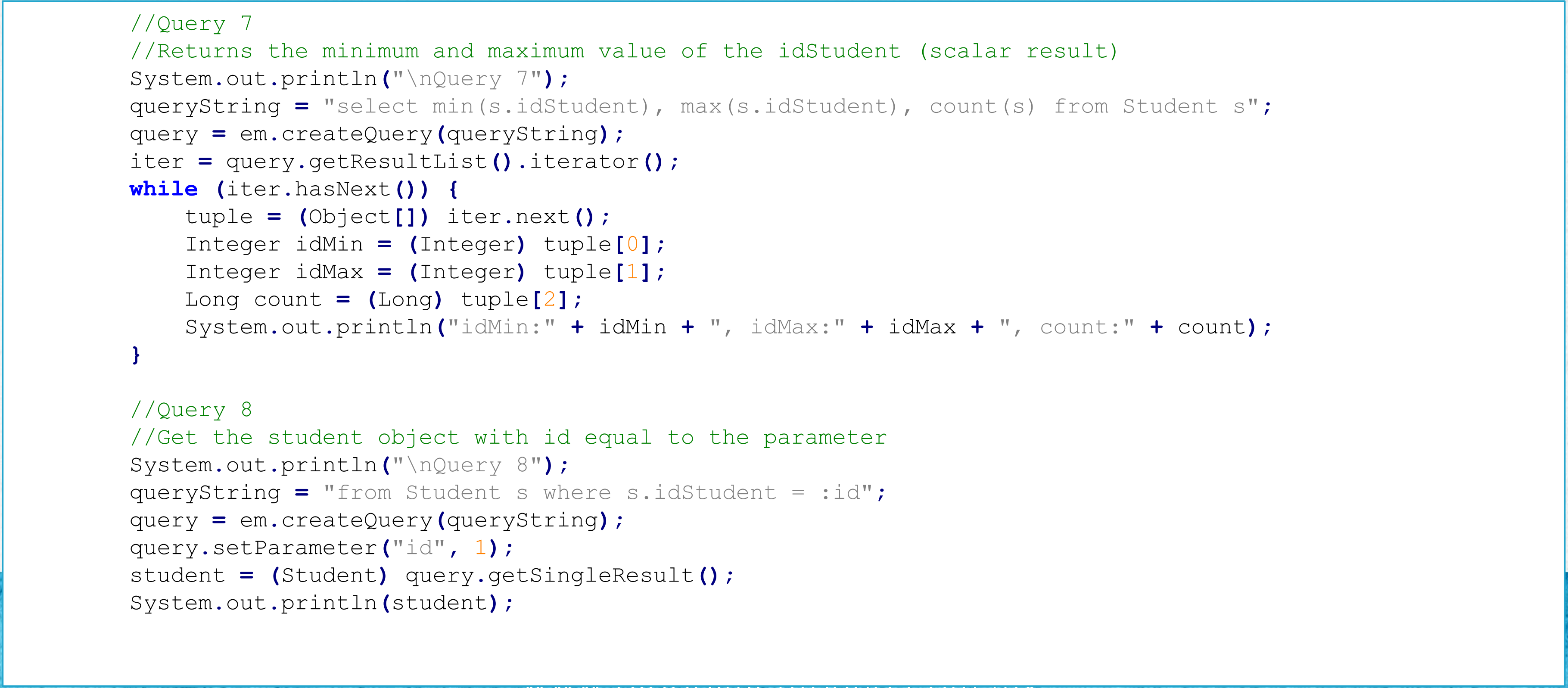
Click to download

```
//Query 5
//Each tuple is returned as an array of objects
System.out.println("\nQuery 5");
queryString = "select s, s.idStudent from Student s";
query = em.createQuery(queryString);
iter = query.getResultList().iterator();
while (iter.hasNext()) {
    tuple = (Object[]) iter.next();
    student = (Student) tuple[0];
    Integer idStudent = (Integer) tuple[1];
    System.out.println("idStudent:" + idStudent);
    System.out.println("Student Object:" + student);
}

//Query 6
System.out.println("\nQuery 6");
queryString = "select new Student(s.idStudent) from Student s";
query = em.createQuery(queryString);
students = query.getResultList();
for (Student s : students) {
    System.out.println(s);
}
```

5. MODIFY THE FILE

TestHqlJpql.java:

Click to download



```java
//Query 7
//Returns the minimum and maximum value of the idStudent (scalar result)
System.out.println("\nQuery 7");
queryString = "select min(s.idStudent), max(s.idStudent), count(s) from Student s";
query = em.createQuery(queryString);
iter = query.getResultList().iterator();
while (iter.hasNext()) {
    tuple = (Object[]) iter.next();
    Integer idMin = (Integer) tuple[0];
    Integer idMax = (Integer) tuple[1];
    Long count = (Long) tuple[2];
    System.out.println("idMin:" + idMin + ", idMax:" + idMax + ", count:" + count);
}

//Query 8
//Get the student object with id equal to the parameter
System.out.println("\nQuery 8");
queryString = "from Student s where s.idStudent = :id";
query = em.createQuery(queryString);
query.setParameter("id", 1);
student = (Student) query.getSingleResult();
System.out.println(student);
```

5. MODIFY THE FILE

TestHqlJpql.java:

Click to download

```
//Query 9
//Get students that contain a letter a, regardless of masuculas or lowercase
System.out.println("\nQuery 9");
String likeString = "%A%"; //like is used in the query
queryString = "from Student s where upper(s.name) like upper(:param1)";
query = em.createQuery(queryString);
query.setParameter("param1", likeString);
students = query.getResultList();
for (Student s : students) {
    System.out.println(s);
}

//Query 10
System.out.println("\nQuery 10");
queryString = "from Student s join s.address a";
query = em.createQuery(queryString);
iter = query.getResultList().iterator();
while (iter.hasNext()) {
    tuple = (Object[]) iter.next();
    student = (Student) tuple[0];
    Address address = (Address) tuple[1];
    System.out.println();
    System.out.println("Student:" + student);
    System.out.println("Address:" + address);
}
```

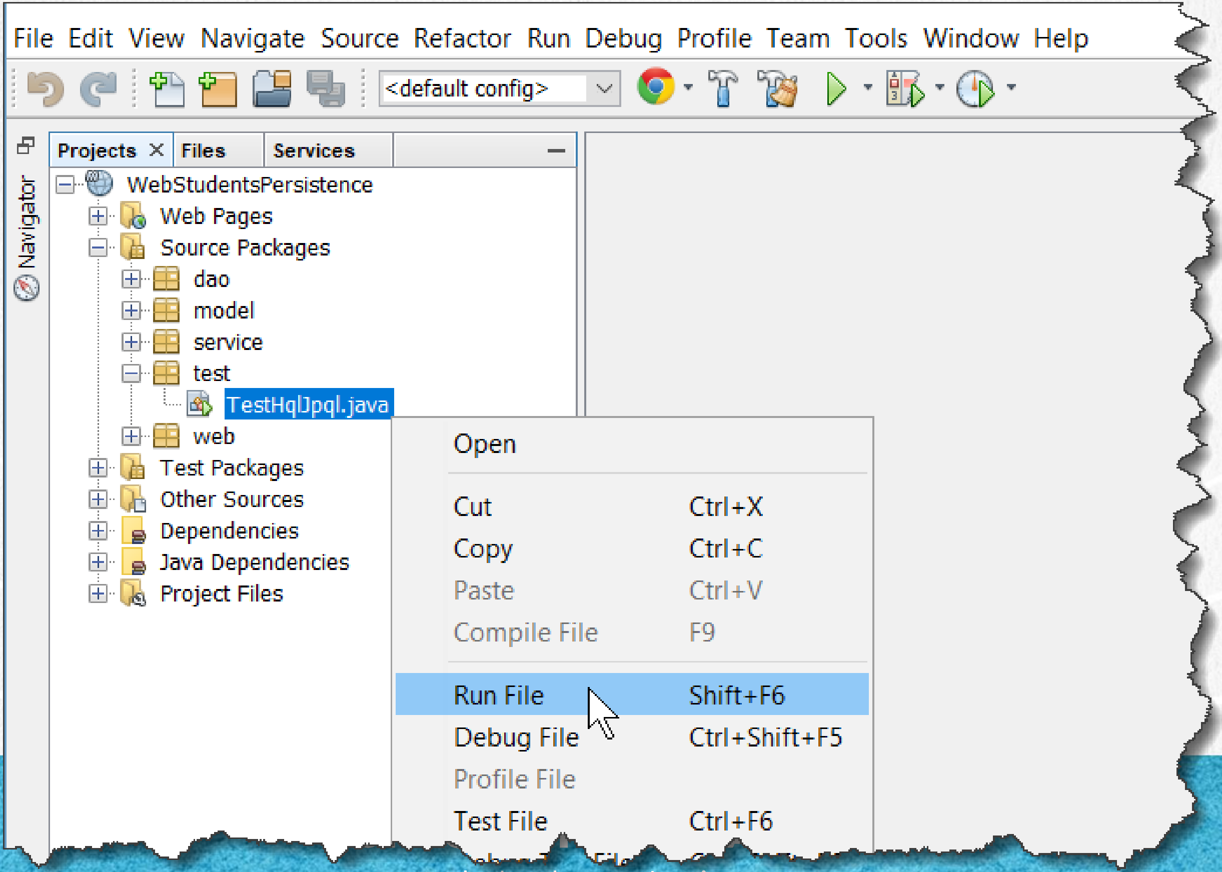
5. MODIFY THE FILE

TestHqlJpql.java:

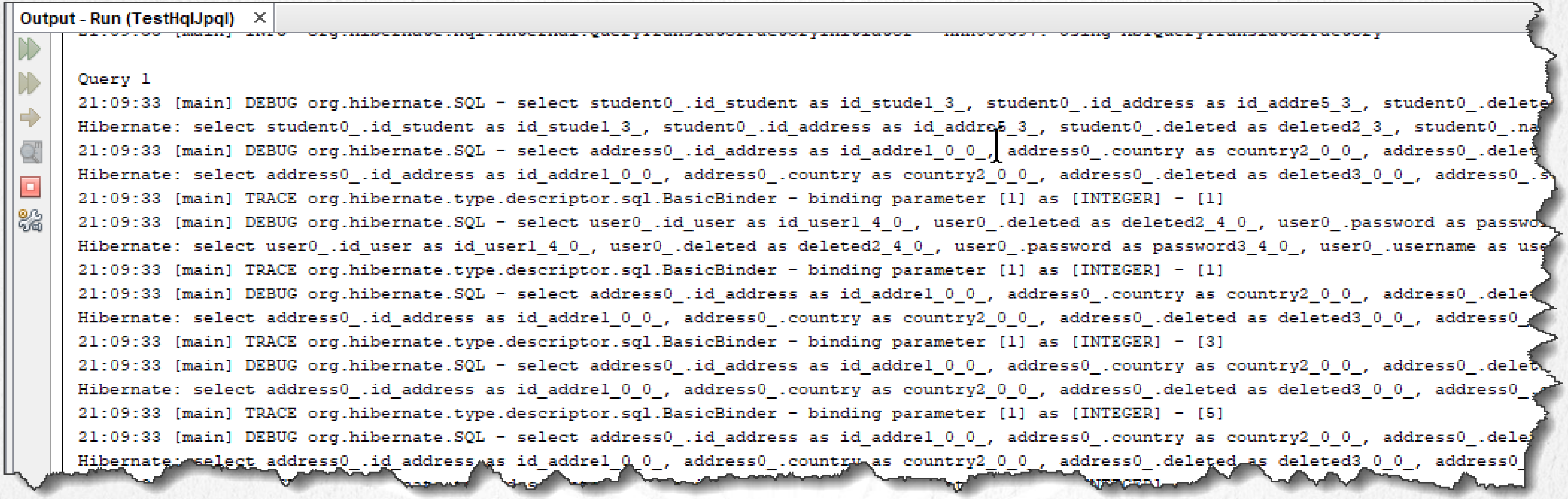
Click to download

```
        //Query 11
        System.out.println("\nQuery 11");
        queryString = "from Student s join fetch s.address a";
        query = em.createQuery(queryString);
        students = query.getResultList();
        for (Student s : students) {
            System.out.println();
            System.out.println("Student:" + s);
            System.out.println("Address:" + s.getAddress());
        }
    }
}
```

6. EXECUTE THE TEST CLASS



6. EXECUTE THE TEST CLASS



```
Query 1
21:09:33 [main] DEBUG org.hibernate.SQL - select student0_.id_student as id_stude1_3_, student0_.id_address as id_addre5_3_, student0_.delete
Hibernate: select student0_.id_student as id_stude1_3_, student0_.id_address as id_addre5_3_, student0_.deleted as deleted2_3_, student0_.na
21:09:33 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.delet
Hibernate: select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_.s
21:09:33 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
21:09:33 [main] DEBUG org.hibernate.SQL - select user0_.id_user as id_user1_4_0_, user0_.deleted as deleted2_4_0_, user0_.password as passwo
Hibernate: select user0_.id_user as id_user1_4_0_, user0_.deleted as deleted2_4_0_, user0_.password as password3_4_0_, user0_.username as use
21:09:33 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [1]
21:09:33 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.delet
Hibernate: select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_
21:09:33 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [3]
21:09:33 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.delet
Hibernate: select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_
21:09:33 [main] TRACE org.hibernate.type.descriptor.sql.BasicBinder - binding parameter [1] as [INTEGER] - [5]
21:09:33 [main] DEBUG org.hibernate.SQL - select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.dele
Hibernate: select address0_.id_address as id_addre1_0_0_, address0_.country as country2_0_0_, address0_.deleted as deleted3_0_0_, address0_
```

7. EXTRA TASKS

It is left as an exercise to execute each of the queries, and review the standard output to be able to observe in more detail what is happening in each of the queries, you can modify the log4j2.xml file to see more information in the console, for this they can modify the file according to the following content and change to debug level instead of info in the root:

```
<Root level="debug">
    <AppenderRef ref="Console" />
 </Root>
```

EXERCISE CONCLUSION

With this exercise we have reviewed the HQL / JPQL queries created in the previous project, but executed from the Java code.

We review from simple queries, with parameters, ordering, groupings, lazy loading and eager loading, among several other examples, so we already have more clear not only how to check our queries with the help of the JPQL console, but also how to execute these queries from Java code or from a Web application.

Global Mentoring

ONLINE COURSE

HIBERNATE & JPA

By: Eng. Ubaldo Acosta

Global Mentoring

JAVA ONLINE COURSES UNIVERSITY

HIBERNATE & JPA COURSE

www.globalmentoring.com.mx