Hibernate-1

What is ORM?

- ORM stands for Object-Relational Mapping (ORM) is a programming technique for converting data between relational databases and object oriented programming languages such as Java, C#, etc.
- An ORM system has the following advantages over plain JDBC:

-Let’s business code access objects rather than DB tables.

-Hides details of SQL queries from OO logic.

-Based on JDBC 'under the hood.’

-No need to deal with the database implementation.

-Fast development of application.

Java ORM Frameworks

- There are several persistent frameworks and ORM options in Java. A persistent framework is an ORM service that stores and retrieves objects into a relational database.

-Enterprise JavaBeans Entity Beans

-Java Data Objects

-Castor

-TopLink

-Spring DAO

-Hibernate, and many more

What is hibernate ?

- Hibernate is an Object-Relational Mapping(ORM) solution for JAVA. It is an open source persistent framework created by Gavin King in 2001. It is a powerful, high performance Object-Relational Persistence and Query service for any Java Application.
- Hibernate frameworks is a mediator through which java application can communicate with database. It is open source frameworks.

Why?

- In JDBC, if we open a database connection we need to write it in try and catch block, and if any exceptions occurred catch block will handle it, and finally used to close the connections.
- We must close the connection, or we may get a connections error message.
- Actually if we didn't close the connection in the finally block, then jdbcis not responsible to close that connection.
- In JDBC we need to write Sql commands in various places, after the program has created if the table structure is modified then the JDBC program doesn't work, again we need to modify and compile and re-deploy required, which is tedious.
- To overcome above drawbacks we should go for Hibernate framework.

Advantages of Hibernate-

- It is open source frameworks.
- Faster performance-

It uses cache concept hence the performance is fast.

- Database independent query-

It generates the database independent query.

- Automatic table creation-

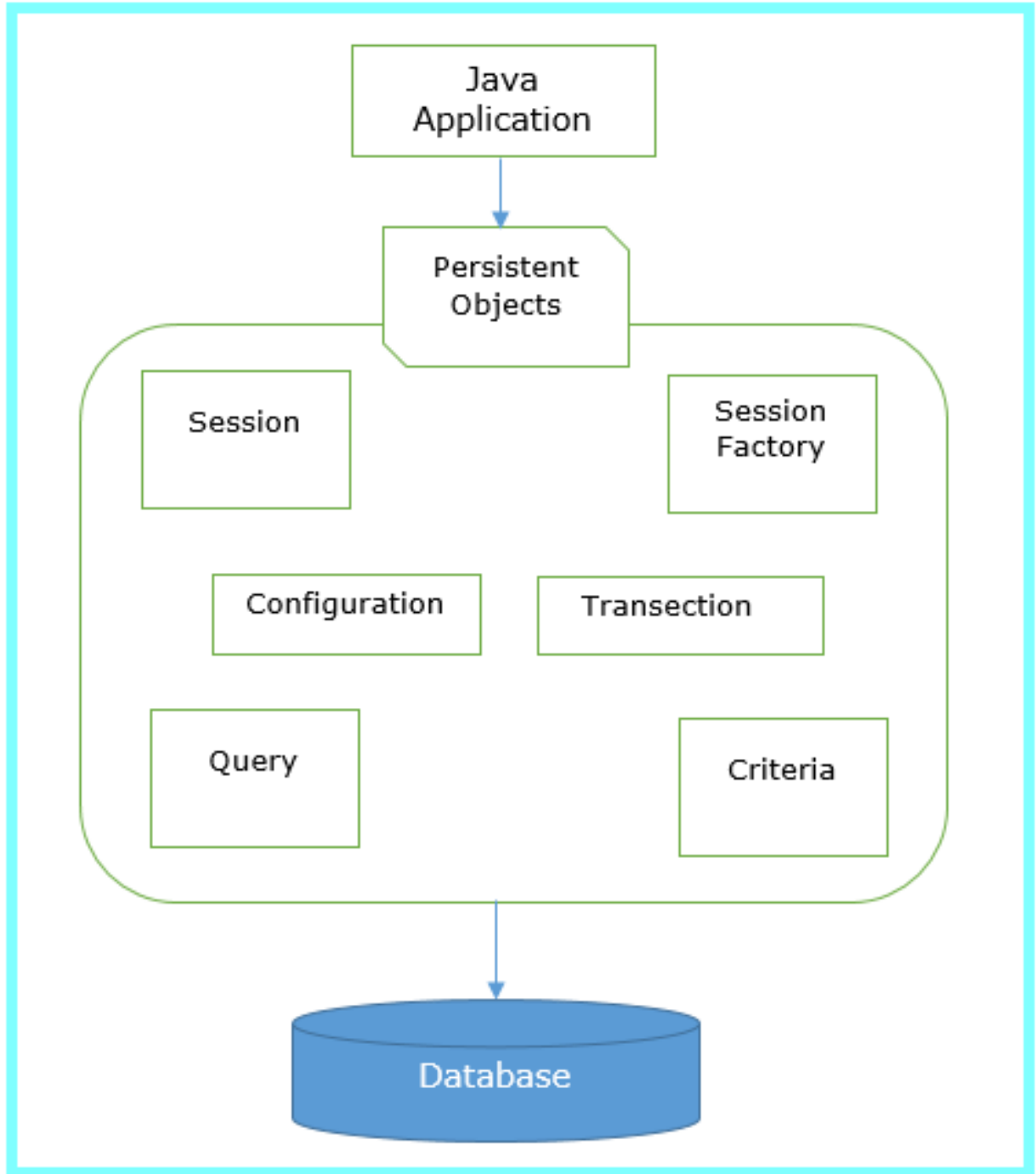
It has facility to create the database tables automatically. There is no need to create the database tables manually.

- Simplifies the complex join-

It is easy to fetch the data from multiple tables in hibernate framework.

Hibernate Architecture

- Hibernate architecture includes the many persistent objects such as session, session factory, transection factory, connection factory and transection etc. there are different types of layer in hibernate such as
- Java application layer
- Hibernate frameworks layer
- Database layer


Java Application
Persistent Objects
Session
Session Factory
Configuration
Transection
Query
Criteria
Database

Key points of Hibernate architecture-

1.SessionFactory-

-It is factory of session. It holds the second level cache. SessionFactory interface provides the factory method to get the object of session.

2.Session-

-It is the factory of transection, query and criteria. It holds first level cache. It provides the method to insert, update and delete the objects. It also provides the factory method for transection, query and criteria, etc.

3.Transection-
-It is the interface that provides the method for transection management.

4.Connection provider-
-It is the factory of JDBC connection. Which driver is used to connect to database.

5.TransectionFactory-
-It is the factory of transection.

- Note- Every hibernate program we must need two files for mapping.

-Hibernate.properties and XML mapping

First Hibernate Application

- 1. Create java project (ok)
- 2. Add jar for hibernate (ok)
- 3. Create the persistent class or POJO class (ok)
- 4. Create the mapping file for persistent class (ok)
- 5. Create configuration file (ok)
- 6. Create the class that store persistent objects (ok)
- 7. Run application. (ok)

What is pojo class?

- Plain Old Java Objects, also known as POJO refer to java objects that are not bordered by any kind of restriction (like any special convention for access modifiers, need to implement Serializable interfaces, etc). They are also known as free objects. The POJO class in java does not need any kind of special classpath.
- POJO acts as a pure data structure that has getter and setter methods.

A POJO should not :

- A pojo class should not extend prespecified classes.
- A pojo class should not implement prespecified interfaces.
- A pojo class should not contain prespecified annotations.

Following are some important properties of the POJO class in java :

- The POJO class in java should always be public.
- The POJO class in java should always have a public default constructor.
- The POJO class in java may contain the arguments constructor.
- The objects of the POJO class in java should contain public Getters and Setters so that the other Java programs can access object values.
- The access modifiers for the objects in the POJO class of java could be public, private, or protected.
- The instance variables for the objects in the POJO class of java should be private for better security.

What is XML file?

- XML stands for eXtensible Markup Language
- XML is a markup language much like HTML
- XML was designed to store and transport data
- XML was designed to be self-descriptive

Element of mapping file-

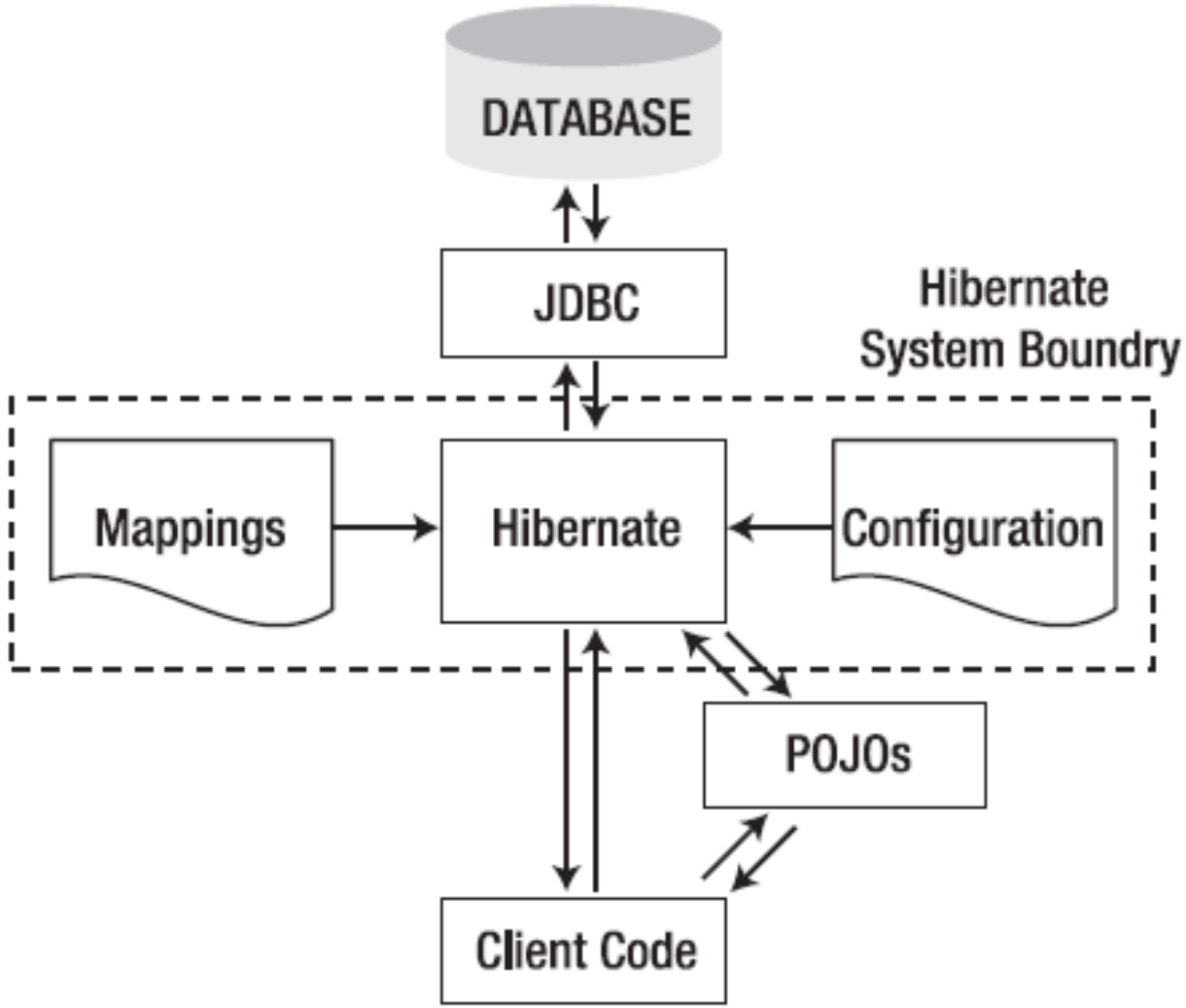
1.hibernate-mapping: It is the root element.

2.Class: It defines the mapping of a POJO class to a database table.

3.Id: It defines the unique key attribute or primary key of the table.

4.generator: It is the sub element of the id element. It is used to automatically generate the id.

5.property: It is used to define the mapping of a POJO class property to database table column


DATABASE
JDBC
Hibernate
System Boundry
Mappings
Hibernate
Configuration
POJOs
Client Code

HibernateDemo/src/com/velocity/pojo/Student.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

*Student.java | student.hbm.xml | hibernate.cfg.xml | Test.java

```java
package com.velocity.pojo;

public class Student {
    private int id;
    private String firstName;
    private String lastName;

    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    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;
    }

}
```

Writable | Smart Insert | 26 : 5 : 497

Start

Type here to search

ENG 21:46 26-09-2022

HibernateDemo/src/student.hbm.xml - Eclipse IDE

File Edit Source Source Navigate Search Project Run Window Help

*Student.java | student.hbm.xml | hibernate.cfg.xml | Test.java

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE hibernate-mapping PUBLIC
3 "-//Hibernate/Hibernate Mapping DTD//EN"
4 "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">
5 <hibernate-mapping>
6 <class name ="com.velocity.pojo.Student" table ="stud">
7 <id name ="id">
8 <generator class="identity"/>
9 </id>
10 <property name="firstName" column="first"></property>
11 <property name="lastName" column="second"></property>
12 </class>
13 </hibernate-mapping>
```

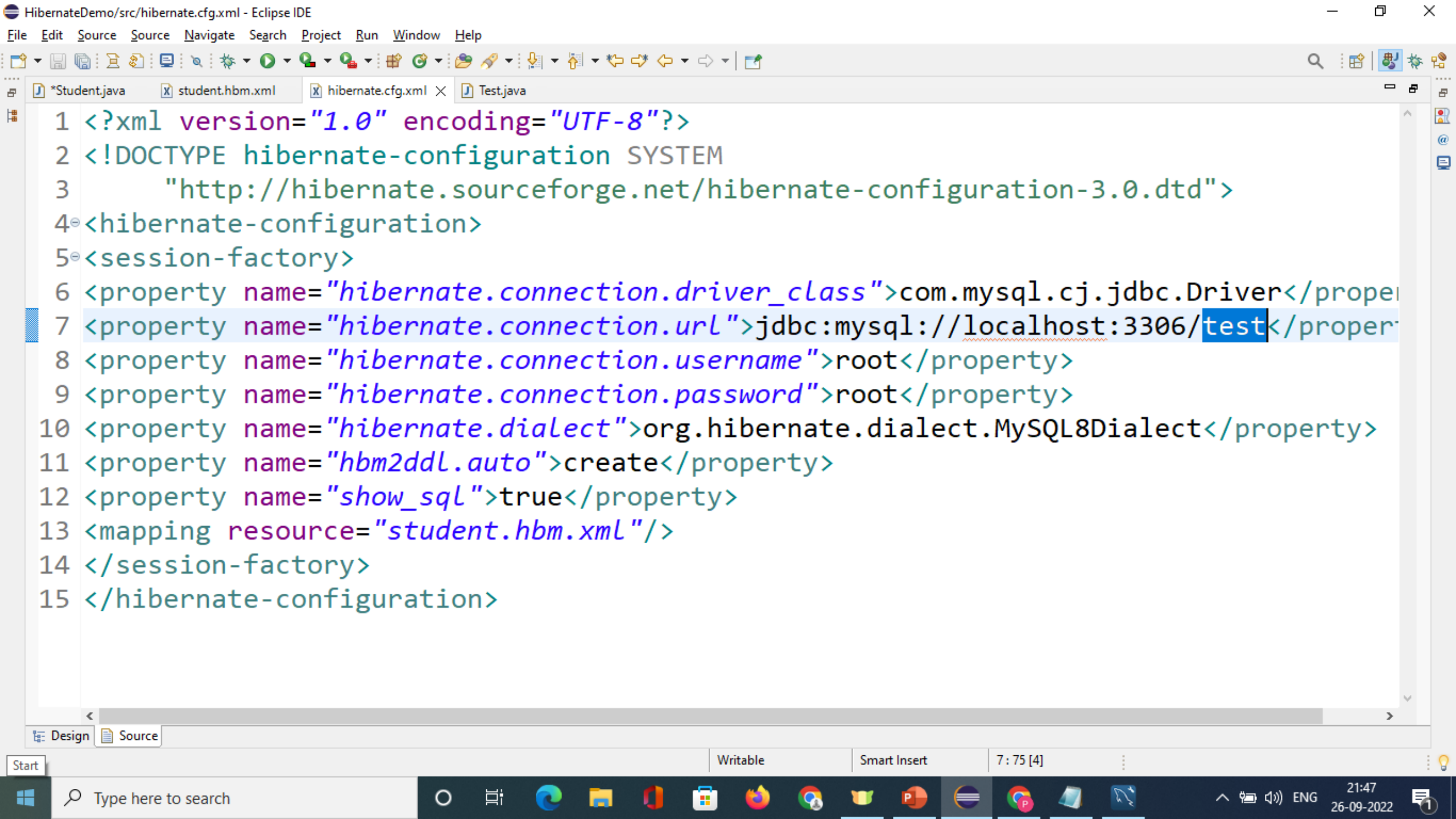
Design | Source

ernate-mapping/class/property/column | Writable | Smart Insert | 11 : 41 [6]

Start

Type here to search

ENG 21:46 26-09-2022


HibernateDemo/src/hibernate.cfg.xml - Eclipse IDE
File Edit Source Source Navigate Search Project Run Window Help
*Student.java
student.hbm.xml
hibernate.cfg.xml
Test.java
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE hibernate-configuration SYSTEM
3 "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
4 <hibernate-configuration>
5 <session-factory>
6 <property name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</prope
7 <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/test</proper
8 <property name="hibernate.connection.username">root</property>
9 <property name="hibernate.connection.password">root</property>
10 <property name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>
11 <property name="hbm2ddl.auto">create</property>
12 <property name="show_sql">true</property>
13 <mapping resource="student.hbm.xml"/>
14 </session-factory>
15 </hibernate-configuration>
Design
Source
Start
Writable
Smart Insert
7 : 75 [4]
Type here to search
ENG
21:47
26-09-2022

HibernateDemo/src/com/velocity/pojo/Test.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

*Student.java | student.hbm.xml | hibernate.cfg.xml | Test.java

```
 8 public class Test {
 9
10     public static void main(String[] args) {
11         //Step-1
12         Configuration configuration = new Configuration();
13         configuration.configure("hibernate.cfg.xml");
14
15         //Step-2
16         SessionFactory sessionFactory = configuration.buildSessionFactory();
17         //Step-3
18         Session session = sessionFactory.openSession();
19         //Step-4
20         Transaction transaction = session.beginTransaction();
21         //Step-5
22         Student student = new Student();
23         student.setFirstName("Pramod");
24         student.setLastName("patil");
25         //Step-6
26         session.save(student);
27         transaction.commit();
28         session.close();
29         System.out.println("Record saved successfully.");
30
31     }
```

Writable | Smart Insert | 25 : 17 : 704

Start

Type here to search

ENG 21:47 26-09-2022

Configurations of xml files

- <!DOCTYPE hibernate-mapping PUBLIC
- "-//Hibernate/Hibernate Mapping DTD//EN"
- "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

- <!DOCTYPE hibernate-configuration SYSTEM
- "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

Thank You