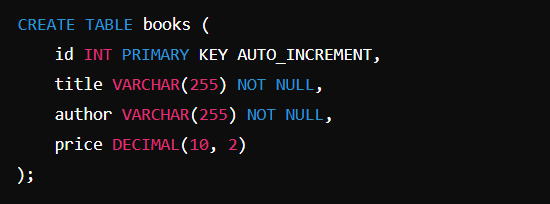
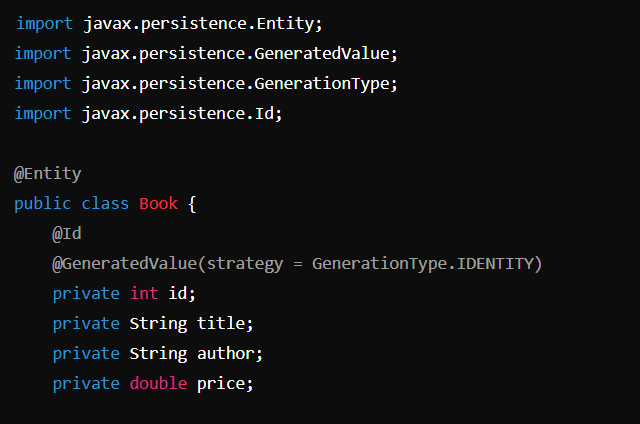
1. CREATE HIBERNATE CRUD OPERATIONS USING entity of your choice. Get the details from respective table using SQL. Define the necessary tables/entities to represent relevant information. Perform update and delete operation.





<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<!-- Database connection settings -->

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

<property name="hibernate.connection.url">jdbc:mysql://localhost:3306/Book</property>

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

<property name="hibernate.connection.password">mysql@123456789</property>

<!-- JDBC connection pool settings -->

<property name="hibernate.c3p0.min\_size">5</property>

<property name="hibernate.c3p0.max\_size">20</property>

<property name="hibernate.c3p0.timeout">300</property>

<property name="hibernate.c3p0.max\_statements">50</property>

<property name="hibernate.c3p0.idle\_test\_period">3000</property>

<!-- SQL dialect -->

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

<!-- Echo all executed SQL to stdout -->

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

<!-- Drop and re-create the database schema on startup -->

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

<!-- Specify our annotated class -->

<mapping class="com.example.Book"/>

</session-factory>

</hibernate-configuration>

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import org.hibernate.cfg.Configuration;

public class HibernateCRUD {

private static SessionFactory sessionFactory = new Configuration().configure().buildSessionFactory();

public static void createBook(String title, String author, double price) {

Session session = sessionFactory.openSession();

Transaction transaction = session.beginTransaction();

Book book = new Book();

book.setTitle(title);

book.setAuthor(author);

book.setPrice(price);

session.save(book);

transaction.commit();

session.close();

}

public static Book getBook(int id) {

Session session = sessionFactory.openSession();

Book book = session.get(Book.class, id);

session.close();

return book;

}

public static void updateBook(int id, String title, String author, double price) {

Session session = sessionFactory.openSession();

Transaction transaction = session.beginTransaction();

Book book = session.get(Book.class, id);

book.setTitle(title);

book.setAuthor(author);

book.setPrice(price);

session.update(book);

transaction.commit();

session.close();

}

public static void deleteBook(int id) {

Session session = sessionFactory.openSession();

Transaction transaction = session.beginTransaction();

Book book = session.get(Book.class, id);

session.delete(book);

transaction.commit();

session.close();

}

public static void main(String[] args) {

createBook("Hibernate in Action", "Christian Bauer", 45.99);

Book book = getBook(1);

System.out.println("Book Retrieved: " + book.getTitle());

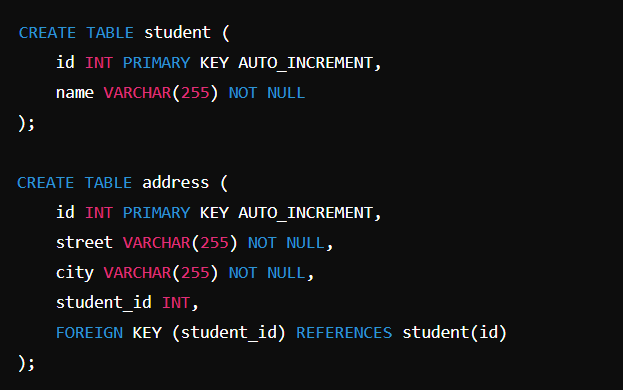
updateBook(1, "Hibernate in Action", "Christian Bauer and Gavin King", 49.99);

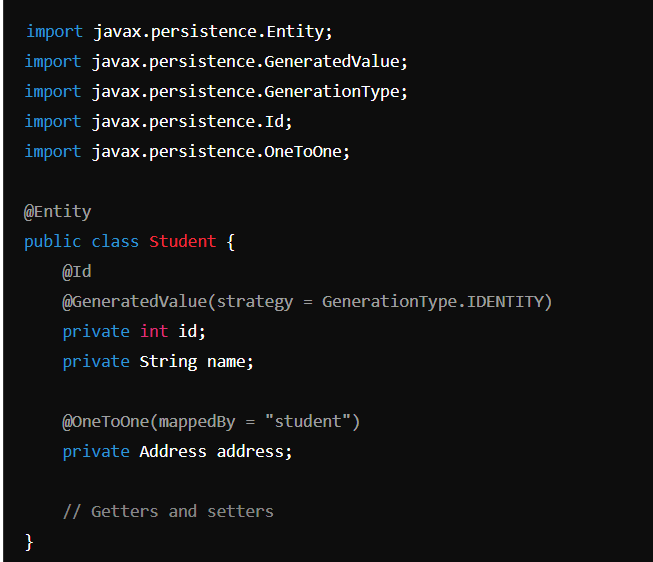
deleteBook(1);

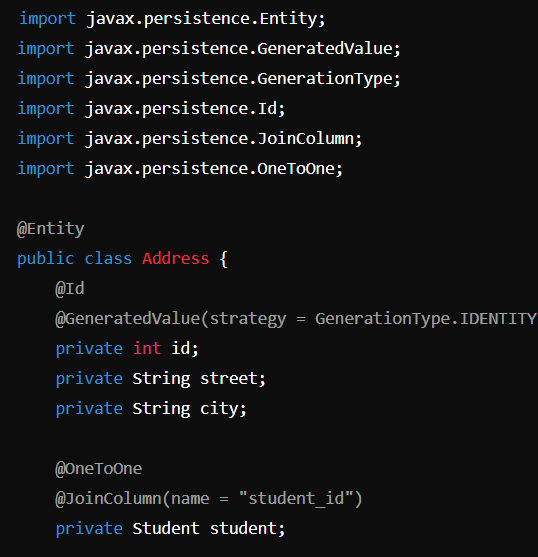
}

}

1. You are working on a Java application to manage information about students and their respective addresses. Implement a one-to-one association between the Student and Address entities using Hibernate.





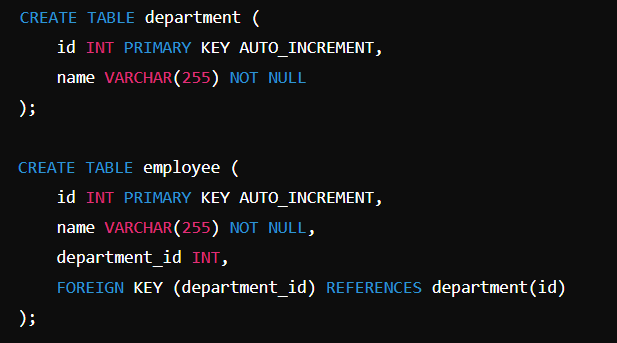


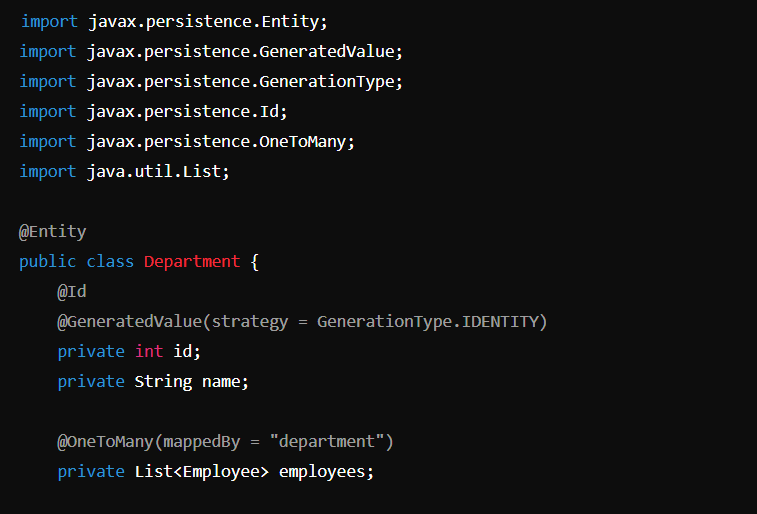
Hibernate Configuration (hibernate.cfg.xml)

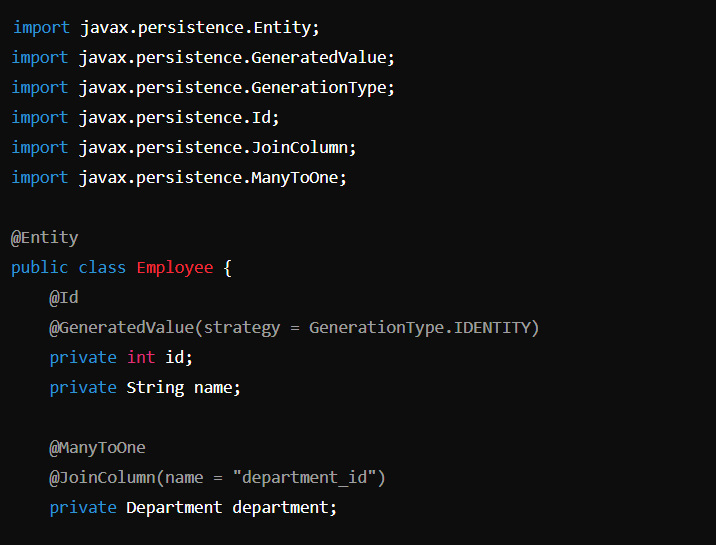
<mapping class="com.example.Student"/>

<mapping class="com.example.Address"/>

1. You are working on a Java application to manage information about employees and their respective departments. Implement a one-to-many association between the Employee and Department entities using Hibernate.







Hibernate Configuration (hibernate.cfg.xml)

<mapping class="com.example.Department"/>

<mapping class="com.example.Employee"/>

public static void main(String[] args) {

Department department = new Department();

department.setName("Engineering");

Employee employee1 = new Employee();

employee1.setName("John Doe");

employee1.setDepartment(department);

Employee employee2 = new Employee();

employee2.setName("Jane Doe");

employee2.setDepartment(department);

// Save department and employees

Session session = sessionFactory.openSession();

Transaction transaction = session.beginTransaction();

session.save(department);

session.save(employee1);

session.save(employee2);

transaction.commit();

session.close();

}