SPRING DATA JPA HANDSON

Hands-on 1: Introduction to HQL and JPQL

What is HQL?

- **HQL** stands for **Hibernate Query Language**.
- It is an **object-oriented** query language similar to SQL but works with entity objects and their properties, not database tables and columns.
- **Supports:** SELECT, UPDATE, DELETE, and also **INSERT** (which is *not* supported in JPQL).

What is JPQL?

- **JPQL** stands for **Java Persistence Query Language**.
- It is the official query language for **JPA (Java Persistence API)**.
- Also object-oriented and similar to SQL.
- **Supports:** SELECT, UPDATE, and DELETE.

Hands on 2: Get all permanent employees using HQL

Folder Structure

```
src
└── main
    ├── java
    │   └── com.example.ormlearn
    │       ├── OrmLearnApplication.java
    │       ├── entity
    │       │   ├── Department.java
    │       │   ├── Employee.java
    │       │   └── Skill.java
    │       ├── repository
    │       │   └── EmployeeRepository.java
    │       └── service
    │           ├── EmployeeService.java
    │           └── EmployeeServiceImpl.java
    └── resources
        ├── application.properties
        └── data.sql
```

application.properties

```
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driver-class-name=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=
spring.jpa.hibernate.ddl-auto=create
spring.jpa.show-sql=true
spring.h2.console.enabled=true
logging.level.org.hibernate.SQL=DEBUG
```

Entity Classes

Department.java

```
package com.example.ormlearn.entity;
```

```
import jakarta.persistence.*;
import java.util.List;

@Entity
public class Department {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    private String name;

    @OneToMany(mappedBy = "department")
    private List<Employee> employeeList;

    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public List<Employee> getEmployeeList() { return employeeList; }
    public void setEmployeeList(List<Employee> employeeList) { this.employeeList =
employeeList; }
}
```

Skill.java

```
package com.example.ormlearn.entity;

import jakarta.persistence.*;

@Entity
public class Skill {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    private String name;

    // Getters and Setters
```

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

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }
}
```

Employee.java

```
package com.example.ormlearn.entity;

import jakarta.persistence.*;
import java.util.Date;
import java.util.List;

@Entity
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    private String name;

    private boolean permanent;

    private double salary;

    @Temporal(TemporalType.DATE)
    private Date dateOfBirth;

    @ManyToOne
    @JoinColumn(name = "department_id")
    private Department department;

    @ManyToMany
    @JoinTable(
        name = "employee_skill",
        joinColumns = @JoinColumn(name = "employee_id"),
        inverseJoinColumns = @JoinColumn(name = "skill_id")
    )
    private List<Skill> skillList;
```

```
    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public boolean isPermanent() { return permanent; }
    public void setPermanent(boolean permanent) { this.permanent = permanent; }

    public double getSalary() { return salary; }
    public void setSalary(double salary) { this.salary = salary; }

    public Date getDateOfBirth() { return dateOfBirth; }
    public void setDateOfBirth(Date dateOfBirth) { this.dateOfBirth = dateOfBirth; }

    public Department getDepartment() { return department; }
    public void setDepartment(Department department) { this.department = department; }

    public List<Skill> getSkillList() { return skillList; }
    public void setSkillList(List<Skill> skillList) { this.skillList = skillList; }
}
```

Repository

EmployeeRepository.java

```
package com.example.ormlearn.repository;

import com.example.ormlearn.entity.Employee;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;

import java.util.List;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

    @Query("SELECT e FROM Employee e LEFT JOIN FETCH e.department d LEFT JOIN
FETCH e.skillList WHERE e.permanent = true")
    List<Employee> getAllPermanentEmployees();
}
```

Service Layer

EmployeeService.java

```
package com.example.ormlearn.service;

import com.example.ormlearn.entity.Employee;

import java.util.List;

public interface EmployeeService {
   List<Employee> getAllPermanentEmployees();
}
```

EmployeeServiceImpl.java

```
package com.example.ormlearn.service;

import com.example.ormlearn.entity.Employee;
import com.example.ormlearn.repository.EmployeeRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class EmployeeServiceImpl implements EmployeeService {

   @Autowired
   private EmployeeRepository employeeRepository;

   @Override
   public List<Employee> getAllPermanentEmployees() {
      return employeeRepository.getAllPermanentEmployees();
   }
}
```

Main Application & Test Method

OrmLearnApplication.java

```
package com.example.ormlearn;
```

```
import com.example.ormlearn.entity.Employee;
import com.example.ormlearn.service.EmployeeService;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

import java.util.List;
import java.util.stream.Collectors;

@SpringBootApplication
public class OrmLearnApplication implements CommandLineRunner {

    private static final Logger LOGGER =
LoggerFactory.getLogger(OrmLearnApplication.class);

    @Autowired
    private EmployeeService employeeService;

    public static void main(String[] args) {
        SpringApplication.run(OrmLearnApplication.class, args);
    }

    @Override
    public void run(String... args) {
        testGetAllPermanentEmployees();
    }

    public void testGetAllPermanentEmployees() {
        LOGGER.info("Start");
        List<Employee> employees = employeeService.getAllPermanentEmployees();
        LOGGER.debug("Permanent Employees: {}", employees);
        employees.forEach(e -> {
            LOGGER.debug("Employee: {}", e.getName());
            LOGGER.debug("Department: {}", e.getDepartment().getName());
            LOGGER.debug("Skills: {}", e.getSkillList().stream().map(skill ->
skill.getName()).collect(Collectors.toList()));
        });
        LOGGER.info("End");
    }
}
```

data.sql (in resources)

```
INSERT INTO department (id, name) VALUES (1, 'HR'), (2, 'Engineering');

INSERT INTO skill (id, name) VALUES (1, 'Java'), (2, 'Spring'), (3, 'SQL');

INSERT INTO employee (id, name, permanent, salary, date_of_birth, department_id)
VALUES (1, 'Alice', true, 50000, '1990-01-01', 2),
    (2, 'Bob', false, 40000, '1992-02-02', 1),
    (3, 'Charlie', true, 60000, '1988-03-03', 2);

INSERT INTO employee_skill (employee_id, skill_id) VALUES (1, 1), (1, 2), (3, 3);
```

Output

```
INFO  Start
DEBUG Permanent Employees: [Employee@1234, Employee@5678]
DEBUG Employee: Alice
DEBUG Department: Engineering
DEBUG Skills: [Java, Spring]
DEBUG Employee: Charlie
DEBUG Department: Engineering
DEBUG Skills: [SQL]
INFO  End
```

Hands on 3: Fetch quiz attempt details using HQL

Folder Structure

```
src
└── main
    ├── java
    │   └── com.example.ormlearn
    │       ├── OrmLearnApplication.java
    │       ├── entity
    │       │   ├── User.java
    │       │   ├── Attempt.java
    │       │   ├── Question.java
    │       │   ├── Option.java
    │       │   ├── AttemptQuestion.java
    │       │   └── AttemptOption.java
    │       ├── repository
    │       │   └── AttemptRepository.java
    │       └── service
    │           ├── AttemptService.java
    │           └── AttemptServiceImpl.java
    └── resources
        ├── application.properties
        └── data.sql
```

User.java

```
package com.example.ormlearn.entity;

import jakarta.persistence.*;
import java.util.List;

@Entity
public class User {
   @Id
   private int id;

   private String username;

   @OneToMany(mappedBy = "user", cascade = CascadeType.ALL)
   private List<Attempt> attempts;
```

```
    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getUsername() { return username; }
    public void setUsername(String username) { this.username = username; }

    public List<Attempt> getAttempts() { return attempts; }
    public void setAttempts(List<Attempt> attempts) { this.attempts = attempts; }
}
```

Attempt.java

```
package com.example.ormlearn.entity;

import jakarta.persistence.*;
import java.util.Date;
import java.util.List;

@Entity
public class Attempt {
    @Id
    private int id;

    @ManyToOne
    @JoinColumn(name = "user_id")
    private User user;

    private Date attemptedDate;

    @OneToMany(mappedBy = "attempt", cascade = CascadeType.ALL)
    private List<AttemptQuestion> attemptQuestions;

    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public User getUser() { return user; }
    public void setUser(User user) { this.user = user; }

    public Date getAttemptedDate() { return attemptedDate; }
    public void setAttemptedDate(Date attemptedDate) { this.attemptedDate = attemptedDate;
}
```

```
    public List<AttemptQuestion> getAttemptQuestions() { return attemptQuestions; }
    public void setAttemptQuestions(List<AttemptQuestion> attemptQuestions) {
this.attemptQuestions = attemptQuestions; }
}
```

Question.java

```
package com.example.ormlearn.entity;

import jakarta.persistence.*;
import java.util.List;

@Entity
public class Question {
    @Id
    private int id;

    private String content;

    @OneToMany(mappedBy = "question", cascade = CascadeType.ALL)
    private List<Option> options;

    private double score;

    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getContent() { return content; }
    public void setContent(String content) { this.content = content; }

    public List<Option> getOptions() { return options; }
    public void setOptions(List<Option> options) { this.options = options; }

    public double getScore() { return score; }
    public void setScore(double score) { this.score = score; }
}
```

Option.java

```
package com.example.ormlearn.entity;
```

```
import jakarta.persistence.*;

@Entity
public class Option {
    @Id
    private int id;

    private String content;

    private boolean correct;

    @ManyToOne
    @JoinColumn(name = "question_id")
    private Question question;

    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getContent() { return content; }
    public void setContent(String content) { this.content = content; }

    public boolean isCorrect() { return correct; }
    public void setCorrect(boolean correct) { this.correct = correct; }

    public Question getQuestion() { return question; }
    public void setQuestion(Question question) { this.question = question; }
}
```

AttemptQuestion.java

```
package com.example.ormlearn.entity;

import jakarta.persistence.*;
import java.util.List;

@Entity
public class AttemptQuestion {
    @Id
    private int id;

    @ManyToOne
```

```
    @JoinColumn(name = "attempt_id")
    private Attempt attempt;

    @ManyToOne
    @JoinColumn(name = "question_id")
    private Question question;

    @OneToMany(mappedBy = "attemptQuestion", cascade = CascadeType.ALL)
    private List<AttemptOption> attemptOptions;

    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public Attempt getAttempt() { return attempt; }
    public void setAttempt(Attempt attempt) { this.attempt = attempt; }

    public Question getQuestion() { return question; }
    public void setQuestion(Question question) { this.question = question; }

    public List<AttemptOption> getAttemptOptions() { return attemptOptions; }
    public void setAttemptOptions(List<AttemptOption> attemptOptions) {
this.attemptOptions = attemptOptions; }
}
```

AttemptOption.java

```
package com.example.ormlearn.entity;

import jakarta.persistence.*;

@Entity
public class AttemptOption {
    @Id
    private int id;

    @ManyToOne
    @JoinColumn(name = "attempt_question_id")
    private AttemptQuestion attemptQuestion;

    @ManyToOne
    @JoinColumn(name = "option_id")
    private Option option;
```

```
    private boolean selected;

    // Getters and Setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public AttemptQuestion getAttemptQuestion() { return attemptQuestion; }
    public void setAttemptQuestion(AttemptQuestion attemptQuestion) { this.attemptQuestion
= attemptQuestion; }

    public Option getOption() { return option; }
    public void setOption(Option option) { this.option = option; }

    public boolean isSelected() { return selected; }
    public void setSelected(boolean selected) { this.selected = selected; }
}
```

Repository

AttemptRepository.java

```
package com.example.ormlearn.repository;

import com.example.ormlearn.entity.Attempt;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;

public interface AttemptRepository extends JpaRepository<Attempt, Integer> {

    @Query("SELECT a FROM Attempt a " +
           "LEFT JOIN FETCH a.user u " +
           "LEFT JOIN FETCH a.attemptQuestions aq " +
           "LEFT JOIN FETCH aq.question q " +
           "LEFT JOIN FETCH q.options o " +
           "LEFT JOIN FETCH aq.attemptOptions ao " +
           "LEFT JOIN FETCH ao.option opt " +
           "WHERE a.id = :attemptId AND u.id = :userId")

    Attempt getAttempt(int userId, int attemptId);

}
```

Service Layer

AttemptService.java

```
package com.example.ormlearn.service;
import com.example.ormlearn.entity.Attempt;
public interface AttemptService {
   Attempt getAttempt(int userId, int attemptId);
}
```

AttemptServiceImpl.java

```
package com.example.ormlearn.service;

import com.example.ormlearn.entity.Attempt;
import com.example.ormlearn.repository.AttemptRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class AttemptServiceImpl implements AttemptService {

   @Autowired
   private AttemptRepository attemptRepository;

   @Override
   public Attempt getAttempt(int userId, int attemptId) {
      return attemptRepository.getAttempt(userId, attemptId);
   }
}
```

Main Application Test Method

OrmLearnApplication.java

```
package com.example.ormlearn;

import com.example.ormlearn.entity.Attempt;
import com.example.ormlearn.entity.AttemptOption;
import com.example.ormlearn.entity.AttemptQuestion;
import com.example.ormlearn.entity.Option;
import com.example.ormlearn.service.AttemptService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
@SpringBootApplication
public class OrmLearnApplication implements CommandLineRunner {

    @Autowired
    private AttemptService attemptService;

    public static void main(String[] args) {
        SpringApplication.run(OrmLearnApplication.class, args);
    }

    @Override
    public void run(String... args) {
        testGetAttemptDetails();
    }

    private void testGetAttemptDetails() {
        int userId = 1;
        int attemptId = 1;

        Attempt attempt = attemptService.getAttempt(userId, attemptId);

        System.out.println("Username: " + attempt.getUser().getUsername());
        System.out.println("Attempted Date: " + attempt.getAttemptedDate());

        for (AttemptQuestion aq : attempt.getAttemptQuestions()) {
            System.out.println("\n" + aq.getQuestion().getContent());
            for (Option opt : aq.getQuestion().getOptions()) {
                boolean selected = aq.getAttemptOptions().stream()
                        .anyMatch(ao -> ao.getOption().getId() == opt.getId() && ao.isSelected());
                System.out.printf("%2d) %-12s %-4.1f %s%n",
                        opt.getId(), opt.getContent(),
                        opt.isCorrect() ? aq.getQuestion().getScore() : 0.0,
                        selected);
            }
        }
    }
}
```

application.properties

```
spring.datasource.url=jdbc:mysql://localhost:3306/ormlearn
```

spring.datasource.username=root
spring.datasource.password=yourpassword
spring.jpa.hibernate.ddl-auto=none
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect

Output

Username: laura
Attempted Date: 2025-07-06

What is the extension of the hyper text markup language file?
1) .xhtm 0.0 false
2) .ht 0.0 false
3) .html 1.0 true
4) .htmx 0.0 false

What is the maximum level of heading tag can be used in a HTML page?
5) 5 0.0 false
6) 3 0.0 true
7) 4 0.0 false
8) 6 1.0 false

The HTML document itself begins with <html> and ends </html>. State True or False
9) false 0.0 false
10) true 1.0 true

Choose the right option to store text value in a variable
11) 'John' 0.5 true
12) John 0.0 false
13) "John" 0.5 false
14) /John/ 0.0 false

Hands on 4: Get average salary using HQL

Folder Structure

```
src
└── main
    ├── java
    │   └── com.example.ormlearn
    │       ├── OrmLearnApplication.java          # Main application file
    │       ├── entity
    │       │   ├── Employee.java                 # Employee entity
    │       │   └── Department.java               # Department entity
    │       ├── repository
    │       │   └── EmployeeRepository.java       # Repository interface with HQL queries
    │       └── service
    │           ├── EmployeeService.java          # Service interface
    │           └── EmployeeServiceImpl.java      # Service implementation
    └── resources
        ├── application.properties                # DB configuration
        └── data.sql                              # Sample insert statements (optional)
```

EmployeeRepository.java

```
package com.example.ormlearn.repository;

import com.example.ormlearn.entity.Employee;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

    // Average salary of all employees
    @Query("SELECT AVG(e.salary) FROM Employee e")
    double getAverageSalary();

    // Average salary of employees in a specific department
    @Query("SELECT AVG(e.salary) FROM Employee e WHERE e.department.id = :id")
    double getAverageSalary(@Param("id") int id);
}
```

EmployeeService.java

```
package com.example.ormlearn.service;

public interface EmployeeService {
   double getAverageSalary();
   double getAverageSalary(int departmentId);
}
```

EmployeeServiceImpl.java

```
package com.example.ormlearn.service;

import com.example.ormlearn.repository.EmployeeRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class EmployeeServiceImpl implements EmployeeService {

   @Autowired
   private EmployeeRepository employeeRepository;

   @Override
   public double getAverageSalary() {
      return employeeRepository.getAverageSalary();
   }

   @Override
   public double getAverageSalary(int departmentId) {
      return employeeRepository.getAverageSalary(departmentId);
   }
}
```

OrmLearnApplication.java

```
package com.example.ormlearn;

import com.example.ormlearn.service.EmployeeService;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class OrmLearnApplication implements CommandLineRunner {

    @Autowired
    private EmployeeService employeeService;

    public static void main(String[] args) {
        SpringApplication.run(OrmLearnApplication.class, args);
    }

    @Override
    public void run(String... args) {
        testGetAverageSalary();
        testGetAverageSalaryByDepartment();
    }

    public void testGetAverageSalary() {
        double avg = employeeService.getAverageSalary();
        System.out.println("Average salary of all employees: " + avg);
    }

    public void testGetAverageSalaryByDepartment() {
        int deptId = 2;
        double avg = employeeService.getAverageSalary(deptId);
        System.out.println("Average salary in Department " + deptId + ": " + avg);
    }
}
```

Output

```
Average salary of all employees: 55000.0
Average salary in Department 2: 60000.0
```

Hands on 5: Get all employees using Native Query

Folder Structure

```
src
└── main
    ├── java
    │   └── com.example.ormlearn
    │       ├── OrmLearnApplication.java
    │       ├── entity
    │       │   └── Employee.java
    │       ├── repository
    │       │   └── EmployeeRepository.java
    │       └── service
    │           ├── EmployeeService.java
    │           └── EmployeeServiceImpl.java
    └── resources
        ├── application.properties
        └── data.sql (optional test data)
```

EmployeeRepository.java

```
package com.example.ormlearn.repository;

import com.example.ormlearn.entity.Employee;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;

import java.util.List;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

    @Query(value = "SELECT * FROM employee", nativeQuery = true)
    List<Employee> getAllEmployeesNative();
}
```

EmployeeService.java

```
package com.example.ormlearn.service;
```

```
import com.example.ormlearn.entity.Employee;
import java.util.List;

public interface EmployeeService {
   List<Employee> getAllEmployeesNative();
}
```

EmployeeServiceImpl.java

```
package com.example.ormlearn.service;

import com.example.ormlearn.entity.Employee;
import com.example.ormlearn.repository.EmployeeRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class EmployeeServiceImpl implements EmployeeService {

   @Autowired
   private EmployeeRepository employeeRepository;

   @Override
   public List<Employee> getAllEmployeesNative() {
      return employeeRepository.getAllEmployeesNative();
   }
}
```

OrmLearnApplication.java

```
package com.example.ormlearn;

import com.example.ormlearn.entity.Employee;
import com.example.ormlearn.service.EmployeeService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
```

```
import java.util.List;

@SpringBootApplication
public class OrmLearnApplication implements CommandLineRunner {

    @Autowired
    private EmployeeService employeeService;

    public static void main(String[] args) {
        SpringApplication.run(OrmLearnApplication.class, args);
    }

    @Override
    public void run(String... args) {
        testGetAllEmployeesNative();
    }

    public void testGetAllEmployeesNative() {
        List<Employee> employees = employeeService.getAllEmployeesNative();
        employees.forEach(e -> System.out.println(e.getId() + " - " + e.getName()));
    }
}
```

Output

```
1 - Alice
2 - Bob
3 - Charlie
…
```

Hands on 6: Criteria Query

Folder Structure

```
src
└── main
    ├── java
    │   └── com.example.ormlearn
    │       ├── OrmLearnApplication.java          # Main class (includes test method)
    │       ├── entity
    │       │   └── Product.java                  # Product entity with fields and getters/setters
    │       ├── service
    │       │   ├── ProductService.java           # Interface for product-related logic
    │       │   └── ProductServiceImpl.java       # Implementation using Criteria API
    └── resources
        ├── application.properties                # DB config (MySQL or H2)
        └── data.sql                              # Sample product data for testing (optional)
```

Product.java

```
@Entity
public class Product {
   @Id
   private int id;

   private String name;
   private int ram;
   private String os;
   private double weight;
   private double cpuSpeed;
   private int hddSize;
   private String cpu;

   // Getters and Setters
  public int getId() {
      return id;
   }

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

```
public String getName() {
    return name;
}

public void setName(String name) {
    this.name = name;
}

public int getRam() {
    return ram;
}

public void setRam(int ram) {
    this.ram = ram;
}

public String getOs() {
    return os;
}

public void setOs(String os) {
    this.os = os;
}

public double getWeight() {
    return weight;
}

public void setWeight(double weight) {
    this.weight = weight;
}

public double getCpuSpeed() {
    return cpuSpeed;
}

public void setCpuSpeed(double cpuSpeed) {
    this.cpuSpeed = cpuSpeed;
}

public int getHddSize() {
    return hddSize;
}
```

```
    public void setHddSize(int hddSize) {
        this.hddSize = hddSize;
    }

    public String getCpu() {
        return cpu;
    }

    public void setCpu(String cpu) {
        this.cpu = cpu;
    }

}
```

ProductService.java

```
package com.example.ormlearn.service;

import com.example.ormlearn.entity.Product;
import java.util.List;
import java.util.Map;

public interface ProductService {
    List<Product> searchProducts(Map<String, Object> filters);
}
```

ProductServiceImpl.java

```
package com.example.ormlearn.service;

import com.example.ormlearn.entity.Product;
import jakarta.persistence.EntityManager;
import jakarta.persistence.PersistenceContext;
import jakarta.persistence.criteria.*;
import org.springframework.stereotype.Service;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;

@Service
```

```
public class ProductServiceImpl implements ProductService {

    @PersistenceContext
    private EntityManager em;

    @Override
    public List<Product> searchProducts(Map<String, Object> filters) {
        CriteriaBuilder cb = em.getCriteriaBuilder();
        CriteriaQuery<Product> cq = cb.createQuery(Product.class);
        Root<Product> product = cq.from(Product.class);

        List<Predicate> predicates = new ArrayList<>();

        if (filters.containsKey("ram")) {
            predicates.add(cb.greaterThanOrEqualTo(product.get("ram"), (Integer)
filters.get("ram")));
        }

        if (filters.containsKey("os")) {
            predicates.add(cb.equal(product.get("os"), filters.get("os")));
        }

        if (filters.containsKey("weight")) {
            predicates.add(cb.lessThanOrEqualTo(product.get("weight"), (Double)
filters.get("weight")));
        }

        if (filters.containsKey("cpuSpeed")) {
            predicates.add(cb.greaterThanOrEqualTo(product.get("cpuSpeed"), (Double)
filters.get("cpuSpeed")));
        }

        if (filters.containsKey("hddSize")) {
            predicates.add(cb.equal(product.get("hddSize"), (Integer) filters.get("hddSize")));
        }

        if (filters.containsKey("cpu")) {
            predicates.add(cb.equal(product.get("cpu"), filters.get("cpu")));
        }

        cq.where(cb.and(predicates.toArray(new Predicate[0])));

        return em.createQuery(cq).getResultList();
    }
```

}

OrmLearnApplication.java

```
package com.example.ormlearn;

import com.example.ormlearn.entity.Product;
import com.example.ormlearn.service.ProductService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

import java.util.HashMap;
import java.util.List;
import java.util.Map;

@SpringBootApplication
public class OrmLearnApplication implements CommandLineRunner {

    @Autowired
    private ProductService productService;

    public static void main(String[] args) {
        SpringApplication.run(OrmLearnApplication.class, args);
    }

    @Override
    public void run(String... args) throws Exception {
        testProductSearchWithFilters();
    }

    public void testProductSearchWithFilters() {
        Map<String, Object> filters = new HashMap<>();
```

```
        filters.put("ram", 8);
        filters.put("os", "Windows");
        filters.put("weight", 2.5);

        List<Product> results = productService.searchProducts(filters);
        System.out.println("=== Filtered Products ===");
        results.forEach(p ->
            System.out.println(
                p.getName() +
                " | RAM: " + p.getRam() + "GB" +
                " | OS: " + p.getOs() +
                " | Weight: " + p.getWeight() + "kg" +
                " | CPU: " + p.getCpu() +
                " | HDD: " + p.getHddSize() + "GB" +
                " | CPU Speed: " + p.getCpuSpeed() + "GHz"
            )
        );
    }
}
```

Output

=== Filtered Products ===

HP Pavilion | RAM: 8GB | OS: Windows | Weight: 2.2kg | CPU: i5 | HDD: 512GB | CPU Speed: 2.4GHz

Dell Inspiron | RAM: 16GB | OS: Windows | Weight: 2.3kg | CPU: i7 | HDD: 1024GB | CPU Speed: 3.0GHz