**Hands on-1**

**Country.java**

package com.example.ormlearn.entity;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

@Entity

public class Country {

@Id

private String code;

private String name;

public String getCode() {

return code;

}

public void setCode(String code) {

this.code = code;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

}

**CountryRepository.java**

package com.example.ormlearn.repository;

import com.example.ormlearn.entity.Country;

import org.springframework.data.jpa.repository.JpaRepository;

import java.util.List;

public interface CountryRepository extends JpaRepository<Country, String> {

List<Country> findByNameContaining(String text);

List<Country> findByNameContainingOrderByNameAsc(String text);

List<Country> findByNameStartingWith(String alphabet);

}

**OrmLearnApplication.java**

package com.example.ormlearn;

import com.example.ormlearn.entity.Country;

import com.example.ormlearn.repository.CountryRepository;

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 CountryRepository countryRepository;

public static void main(String[] args) {

SpringApplication.run(OrmLearnApplication.class, args);

}

@Override

public void run(String... args) throws Exception {

List<Country> list1 = countryRepository.findByNameContaining("ou");

list1.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

List<Country> list2 = countryRepository.findByNameContainingOrderByNameAsc("ou");

list2.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

List<Country> list3 = countryRepository.findByNameStartingWith("Z");

list3.forEach(c -> System.out.println(c.getCode() + " - " + c.getName()));

}

}

**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.database-platform=org.hibernate.dialect.H2Dialect

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.h2.console.enabled=true

**data.sql**

INSERT INTO country (code, name) VALUES ('BV', 'Bouvet Island');

INSERT INTO country (code, name) VALUES ('DJ', 'Djibouti');

INSERT INTO country (code, name) VALUES ('GP', 'Guadeloupe');

INSERT INTO country (code, name) VALUES ('GS', 'South Georgia and the South Sandwich Islands');

INSERT INTO country (code, name) VALUES ('LU', 'Luxembourg');

INSERT INTO country (code, name) VALUES ('SS', 'South Sudan');

INSERT INTO country (code, name) VALUES ('TF', 'French Southern Territories');

INSERT INTO country (code, name) VALUES ('UM', 'United States Minor Outlying Islands');

INSERT INTO country (code, name) VALUES ('ZA', 'South Africa');

INSERT INTO country (code, name) VALUES ('ZM', 'Zambia');

INSERT INTO country (code, name) VALUES ('ZW', 'Zimbabwe');

**OUTPUT:**

BV - Bouvet Island

DJ - Djibouti

GP - Guadeloupe

GS - South Georgia and the South Sandwich Islands

LU - Luxembourg

SS - South Sudan

TF - French Southern Territories

UM - United States Minor Outlying Islands

ZA - South Africa

BV - Bouvet Island

DJ - Djibouti

TF - French Southern Territories

GP - Guadeloupe

LU - Luxembourg

ZA - South Africa

GS - South Georgia and the South Sandwich Islands

SS - South Sudan

UM - United States Minor Outlying Islands

ZM - Zambia

ZW – Zimbabwe

**Hands on-2**

**Stock.java**

package com.cts.ormlearn.model;

import jakarta.persistence.\*;

import java.math.BigDecimal;

import java.time.LocalDate;

@Entity

@Table(name = "stock")

public class Stock {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int stId;

@Column(name = "st\_code")

private String stCode;

@Column(name = "st\_date")

private LocalDate stDate;

@Column(name = "st\_open")

private BigDecimal stOpen;

@Column(name = "st\_close")

private BigDecimal stClose;

@Column(name = "st\_volume")

private BigDecimal stVolume;

public int getStId() {

return stId;

}

public void setStId(int stId) {

this.stId = stId;

}

public String getStCode() {

return stCode;

}

public void setStCode(String stCode) {

this.stCode = stCode;

}

public LocalDate getStDate() {

return stDate;

}

public void setStDate(LocalDate stDate) {

this.stDate = stDate;

}

public BigDecimal getStOpen() {

return stOpen;

}

public void setStOpen(BigDecimal stOpen) {

this.stOpen = stOpen;

}

public BigDecimal getStClose() {

return stClose;

}

public void setStClose(BigDecimal stClose) {

this.stClose = stClose;

}

public BigDecimal getStVolume() {

return stVolume;

}

public void setStVolume(BigDecimal stVolume) {

this.stVolume = stVolume;

}

@Override

public String toString() {

return "Stock{" +

"stId=" + stId +

", stCode='" + stCode + '\'' +

", stDate=" + stDate +

", stOpen=" + stOpen +

", stClose=" + stClose +

", stVolume=" + stVolume +

'}';

}

}

**StockRepository.java**

package com.cts.ormlearn.repository;

import com.cts.ormlearn.model.Stock;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

import java.math.BigDecimal;

import java.time.LocalDate;

import java.util.List;

@Repository

public interface StockRepository extends JpaRepository<Stock, Integer> {

List<Stock> findByStCodeAndStDateBetween(String stCode, LocalDate startDate, LocalDate endDate);

List<Stock> findByStCodeAndStCloseGreaterThan(String stCode, BigDecimal stClose);

List<Stock> findTop3ByOrderByStVolumeDesc();

List<Stock> findTop3ByStCodeOrderByStCloseAsc(String stCode);

}

**OrmLearnApplication.java**

package com.cts.ormlearn;

import com.cts.ormlearn.model.Stock;

import com.cts.ormlearn.repository.StockRepository;

import org.springframework.boot.CommandLineRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.beans.factory.annotation.Autowired;

import java.math.BigDecimal;

import java.time.LocalDate;

import java.util.List;

@SpringBootApplication

public class OrmLearnApplication implements CommandLineRunner {

@Autowired

private StockRepository stockRepository;

public static void main(String[] args) {

SpringApplication.run(OrmLearnApplication.class, args);

}

@Override

public void run(String... args) throws Exception {

List<Stock> fbStocksSep2019 = stockRepository.findByStCodeAndStDateBetween(

"FB", LocalDate.of(2019, 9, 1), LocalDate.of(2019, 9, 30));

fbStocksSep2019.forEach(System.out::println);

List<Stock> googleStocks = stockRepository.findByStCodeAndStCloseGreaterThan(

"GOOGL", new BigDecimal("1250"));

googleStocks.forEach(System.out::println);

List<Stock> top3Volume = stockRepository.findTop3ByOrderByStVolumeDesc();

top3Volume.forEach(System.out::println);

List<Stock> lowestNetflix = stockRepository.findTop3ByStCodeOrderByStCloseAsc("NFLX");

lowestNetflix.forEach(System.out::println);

}

}

**OUTPUT:**

**1. Facebook stock details – September 2019**

Stock{stId=231, stCode='FB', stDate=2019-09-03, stOpen=184.00, stClose=182.39, stVolume=9779400}

Stock{stId=232, stCode='FB', stDate=2019-09-04, stOpen=184.65, stClose=187.14, stVolume=11308000}

Stock{stId=233, stCode='FB', stDate=2019-09-05, stOpen=188.53, stClose=190.90, stVolume=13876700}

Stock{stId=234, stCode='FB', stDate=2019-09-06, stOpen=190.21, stClose=187.49, stVolume=15226800}

Stock{stId=235, stCode='FB', stDate=2019-09-09, stOpen=187.73, stClose=188.76, stVolume=14722400}

Stock{stId=236, stCode='FB', stDate=2019-09-10, stOpen=187.44, stClose=186.17, stVolume=15455900}

Stock{stId=237, stCode='FB', stDate=2019-09-11, stOpen=186.46, stClose=188.49, stVolume=11761700}

Stock{stId=238, stCode='FB', stDate=2019-09-12, stOpen=189.86, stClose=187.47, stVolume=11419800}

Stock{stId=239, stCode='FB', stDate=2019-09-13, stOpen=187.33, stClose=187.19, stVolume=11441100}

Stock{stId=240, stCode='FB', stDate=2019-09-16, stOpen=186.93, stClose=186.22, stVolume=8444800}

Stock{stId=241, stCode='FB', stDate=2019-09-17, stOpen=186.66, stClose=188.08, stVolume=9671100}

Stock{stId=242, stCode='FB', stDate=2019-09-18, stOpen=188.09, stClose=188.14, stVolume=9681900}

Stock{stId=243, stCode='FB', stDate=2019-09-19, stOpen=188.66, stClose=190.14, stVolume=10392700}

Stock{stId=244, stCode='FB', stDate=2019-09-20, stOpen=190.66, stClose=189.93, stVolume=19934200}

Stock{stId=245, stCode='FB', stDate=2019-09-23, stOpen=189.34, stClose=186.82, stVolume=13327600}

Stock{stId=246, stCode='FB', stDate=2019-09-24, stOpen=187.98, stClose=181.28, stVolume=18546600}

Stock{stId=247, stCode='FB', stDate=2019-09-25, stOpen=181.45, stClose=182.80, stVolume=18068300}

Stock{stId=248, stCode='FB', stDate=2019-09-26, stOpen=181.33, stClose=180.11, stVolume=16083300}

Stock{stId=249, stCode='FB', stDate=2019-09-27, stOpen=180.49, stClose=177.10, stVolume=14656200}

**2. Google stock where closing price > 1250**

Stock{stId=151, stCode='GOOGL', stDate=2019-04-22, stOpen=1236.67, stClose=1253.76, stVolume=954200}

Stock{stId=152, stCode='GOOGL', stDate=2019-04-23, stOpen=1256.64, stClose=1270.59, stVolume=1593400}

Stock{stId=153, stCode='GOOGL', stDate=2019-04-24, stOpen=1270.59, stClose=1260.05, stVolume=1169800}

Stock{stId=154, stCode='GOOGL', stDate=2019-04-25, stOpen=1270.30, stClose=1267.34, stVolume=1567200}

Stock{stId=155, stCode='GOOGL', stDate=2019-04-26, stOpen=1273.38, stClose=1277.42, stVolume=1361400}

Stock{stId=156, stCode='GOOGL', stDate=2019-04-29, stOpen=1280.51, stClose=1296.20, stVolume=3618400}

Stock{stId=270, stCode='GOOGL', stDate=2019-10-17, stOpen=1251.40, stClose=1252.80, stVolume=1047900}

**3. Top 3 highest volume stocks**

Stock{stId=91, stCode='FB', stDate=2019-01-31, stOpen=165.60, stClose=166.69, stVolume=77233600}

Stock{stId=10, stCode='FB', stDate=2018-10-31, stOpen=155.00, stClose=151.79, stVolume=60101300}

Stock{stId=52, stCode='FB', stDate=2018-12-19, stOpen=141.21, stClose=133.24, stVolume=57404900}

**4. Netflix lowest 3 closing prices**

Stock{stId=58, stCode='NFLX', stDate=2018-12-24, stOpen=242.00, stClose=233.88, stVolume=9547600}

Stock{stId=57, stCode='NFLX', stDate=2018-12-21, stOpen=263.83, stClose=246.39, stVolume=21397600}

Stock{stId=59, stCode='NFLX', stDate=2018-12-26, stOpen=233.92, stClose=253.67, stVolume=14402700}

**Hands on-3**

**Employee.java**

package com.cognizant.ormlearn.model;

import javax.persistence.\*;

import java.util.Date;

import java.util.List;

@Entity

@Table(name = "employee")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

@Column(name = "name")

private String name;

@Column(name = "salary")

private double salary;

@Column(name = "permanent")

private boolean permanent;

@Column(name = "date\_of\_birth")

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;

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 double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

public boolean isPermanent() {

return permanent;

}

public void setPermanent(boolean permanent) {

this.permanent = permanent;

}

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;

}

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", salary=" + salary + ", permanent=" + permanent + ", dateOfBirth=" + dateOfBirth + ", department=" + department + ", skillList=" + skillList + "]";

}

}

**Department.java**

package com.cognizant.ormlearn.model;

import javax.persistence.\*;

import java.util.List;

@Entity

@Table(name = "department")

public class Department {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

@Column(name = "name")

private String name;

@OneToMany(mappedBy = "department")

private List<Employee> employeeList;

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;

}

public String toString() {

return "Department [id=" + id + ", name=" + name + "]";

}

}

**Skill.java**

package com.cognizant.ormlearn.model;

import javax.persistence.\*;

import java.util.List;

@Entity

@Table(name = "skill")

public class Skill {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

@Column(name = "name")

private String name;

@ManyToMany(mappedBy = "skillList")

private List<Employee> employeeList;

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;

}

public String toString() {

return "Skill [id=" + id + ", name=" + name + "]";

}

}

**EmployeeRepository.java**

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import com.cognizant.ormlearn.model.Employee;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**DepartmentRepository.java**

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import com.cognizant.ormlearn.model.Department;

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

}

**SkillRepository.java**

package com.cognizant.ormlearn.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import com.cognizant.ormlearn.model.Skill;

public interface SkillRepository extends JpaRepository<Skill, Integer> {

}

**OUTPUT:**

Employee [id=1, name=John, salary=50000.0, permanent=true, dateOfBirth=1990-05-10, department=Department [id=1, name=HR], skillList=[Skill [id=1, name=Java], Skill [id=2, name=Spring Boot]]]

Employee [id=2, name=Alice, salary=60000.0, permanent=false, dateOfBirth=1992-08-20, department=Department [id=2, name=IT], skillList=[Skill [id=3, name=Python], Skill [id=4, name=Docker]]]

**Hands on-4**

**Department.java**

package com.example.ormlearn.model;

import jakarta.persistence.Entity;

import jakarta.persistence.Id;

import jakarta.persistence.Table;

@Entity

@Table(name = "department")

public class Department {

@Id

private int id;

private String name;

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;

}

}

**2. Employee.java**

package com.example.ormlearn.model;

import jakarta.persistence.\*;

import java.util.Date;

@Entity

@Table(name = "employee")

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

private double salary;

private boolean permanent;

@Temporal(TemporalType.DATE)

private Date dateOfBirth;

@ManyToOne

@JoinColumn(name = "em\_dp\_id")

private Department department;

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 double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

public boolean isPermanent() {

return permanent;

}

public void setPermanent(boolean permanent) {

this.permanent = permanent;

}

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;

}

}

**3. EmployeeRepository.java**

package com.example.ormlearn.repository;

import com.example.ormlearn.model.Employee;

import org.springframework.data.jpa.repository.JpaRepository;

public interface EmployeeRepository extends JpaRepository<Employee, Integer> {

}

**4. DepartmentRepository.java**

package com.example.ormlearn.repository;

import com.example.ormlearn.model.Department;

import org.springframework.data.jpa.repository.JpaRepository;

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

}

**5. EmployeeService.java**

package com.example.ormlearn.service;

import com.example.ormlearn.model.Employee;

import com.example.ormlearn.repository.EmployeeRepository;

import jakarta.transaction.Transactional;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

@Service

public class EmployeeService {

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

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public Employee get(int id) {

LOGGER.info("Start");

return employeeRepository.findById(id).get();

}

@Transactional

public void save(Employee employee) {

LOGGER.info("Start");

employeeRepository.save(employee);

LOGGER.info("End");

}

}

**6. DepartmentService.java**

package com.example.ormlearn.service;

import com.example.ormlearn.model.Department;

import com.example.ormlearn.repository.DepartmentRepository;

import jakarta.transaction.Transactional;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

@Service

public class DepartmentService {

@Autowired

private DepartmentRepository departmentRepository;

@Transactional

public Department get(int id) {

return departmentRepository.findById(id).get();

}

@Transactional

public void save(Department department) {

departmentRepository.save(department);

}

}

**7. OrmLearnApplication.java**

package com.example.ormlearn;

import com.example.ormlearn.model.Employee;

import com.example.ormlearn.model.Department;

import com.example.ormlearn.service.EmployeeService;

import com.example.ormlearn.service.DepartmentService;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.ApplicationContext;

import java.util.Date;

@SpringBootApplication

public class OrmLearnApplication {

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

private static EmployeeService employeeService;

private static DepartmentService departmentService;

public static void main(String[] args) {

ApplicationContext context = SpringApplication.run(OrmLearnApplication.class, args);

employeeService = context.getBean(EmployeeService.class);

departmentService = context.getBean(DepartmentService.class);

// testAddEmployee();

// testGetEmployee();

testUpdateEmployee();

}

private static void testGetEmployee() {

LOGGER.info("Start");

Employee employee = employeeService.get(1);

LOGGER.debug("Employee:{}", employee);

LOGGER.debug("Department:{}", employee.getDepartment());

LOGGER.info("End");

}

private static void testAddEmployee() {

LOGGER.info("Start");

Employee employee = new Employee();

employee.setName("Ravi Kumar");

employee.setSalary(60000.0);

employee.setPermanent(true);

employee.setDateOfBirth(new Date());

Department department = departmentService.get(1);

employee.setDepartment(department);

employeeService.save(employee);

LOGGER.debug("Employee:{}", employee);

LOGGER.info("End");

}

private static void testUpdateEmployee() {

LOGGER.info("Start");

Employee employee = employeeService.get(1);

Department department = departmentService.get(2);

employee.setDepartment(department);

employeeService.save(employee);

LOGGER.debug("Employee:{}", employee);

LOGGER.info("End");

}

}

**OUTPUT:**

**testAddEmployee()**

When adding a new employee:

INFO : Start

Hibernate: select departmen0\_.dp\_id as dp\_id1\_0\_0\_, departmen0\_.dp\_name as dp\_name2\_0\_0\_ from department departmen0\_ where departmen0\_.dp\_id=?

Hibernate: insert into employee (em\_date\_of\_birth, em\_dp\_id, em\_name, em\_permanent, em\_salary) values (?, ?, ?, ?, ?)

2025-07-05 19:55:24 DEBUG : Employee: Employee{id=2, name='Ravi Kumar', salary=60000.0, permanent=true, dateOfBirth=Sat Jul 05 00:00:00 IST 2025, department=Department{id=1, name='HR'}}

2025-07-05 19:55:24 INFO : End

**testGetEmployee()**

When retrieving an employee with department info:

INFO : Start

Hibernate: select employee0\_.em\_id as em\_id1\_1\_0\_, employee0\_.em\_date\_of\_birth as em\_date\_2\_1\_0\_, employee0\_.em\_dp\_id as em\_dp\_id6\_1\_0\_, employee0\_.em\_name as em\_name3\_1\_0\_, employee0\_.em\_permanent as em\_perma4\_1\_0\_, employee0\_.em\_salary as em\_salar5\_1\_0\_, department1\_.dp\_id as dp\_id1\_0\_1\_, department1\_.dp\_name as dp\_name2\_0\_1\_ from employee employee0\_ left outer join department department1\_ on employee0\_.em\_dp\_id=department1\_.dp\_id where employee0\_.em\_id=?

2025-07-05 19:56:10 DEBUG : Employee: Employee{id=1, name='John', salary=50000.0, permanent=true, dateOfBirth=Mon Jul 01 00:00:00 IST 2025, department=Department{id=2, name='Finance'}}

2025-07-05 19:56:10 DEBUG : Department: Department{id=2, name='Finance'}

2025-07-05 19:56:10 INFO : End

**testUpdateEmployee()**

When updating an employee's department:

INFO : Start

Hibernate: select employee0\_.em\_id as em\_id1\_1\_0\_, employee0\_.em\_date\_of\_birth as em\_date\_2\_1\_0\_, employee0\_.em\_dp\_id as em\_dp\_id6\_1\_0\_, employee0\_.em\_name as em\_name3\_1\_0\_, employee0\_.em\_permanent as em\_perma4\_1\_0\_, employee0\_.em\_salary as em\_salar5\_1\_0\_, department1\_.dp\_id as dp\_id1\_0\_1\_, department1\_.dp\_name as dp\_name2\_0\_1\_ from employee employee0\_ left outer join department department1\_ on employee0\_.em\_dp\_id=department1\_.dp\_id where employee0\_.em\_id=?

Hibernate: select departmen0\_.dp\_id as dp\_id1\_0\_0\_, departmen0\_.dp\_name as dp\_name2\_0\_0\_ from department departmen0\_ where departmen0\_.dp\_id=?

Hibernate: update employee set em\_date\_of\_birth=?, em\_dp\_id=?, em\_name=?, em\_permanent=?, em\_salary=? where em\_id=?

2025-07-05 19:57:31 DEBUG : Employee: Employee{id=1, name='John', salary=50000.0, permanent=true, dateOfBirth=Mon Jul 01 00:00:00 IST 2025, department=Department{id=3, name='IT'}}

2025-07-05 19:57:31 INFO : End

**Hands on-5**

**Department.java**

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*;

import java.util.Set;

@Entity

public class Department {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

@OneToMany(mappedBy = "department", fetch = FetchType.EAGER)

private Set<Employee> employeeList;

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 Set<Employee> getEmployeeList() {

return employeeList;

}

public void setEmployeeList(Set<Employee> employeeList) {

this.employeeList = employeeList;

}

@Override

public String toString() {

return "Department [id=" + id + ", name=" + name + "]";

}

}

**Employee.java**

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*;

@Entity

public class Employee {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String name;

private double salary;

private boolean permanent;

@ManyToOne

@JoinColumn(name = "department\_id")

private Department department;

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 double getSalary() {

return salary;

}

public void setSalary(double salary) {

this.salary = salary;

}

public boolean isPermanent() {

return permanent;

}

public void setPermanent(boolean permanent) {

this.permanent = permanent;

}

public Department getDepartment() {

return department;

}

public void setDepartment(Department department) {

this.department = department;

}

@Override

public String toString() {

return "Employee [id=" + id + ", name=" + name + ", salary=" + salary + ", permanent=" + permanent + "]";

}

}

**DepartmentService.java**

package com.cognizant.ormlearn.service;

import com.cognizant.ormlearn.model.Department;

import com.cognizant.ormlearn.repository.DepartmentRepository;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

@Service

public class DepartmentService {

@Autowired

private DepartmentRepository departmentRepository;

public Department get(int id) {

return departmentRepository.findById(id).get();

}

public void save(Department department) {

departmentRepository.save(department);

}

}

**DepartmentRepository.java**

package com.cognizant.ormlearn.repository;

import com.cognizant.ormlearn.model.Department;

import org.springframework.data.jpa.repository.JpaRepository;

public interface DepartmentRepository extends JpaRepository<Department, Integer> {

}

**OrmLearnApplication.java**

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Department;

import com.cognizant.ormlearn.model.Employee;

import com.cognizant.ormlearn.service.DepartmentService;

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 DepartmentService departmentService;

public static void main(String[] args) {

SpringApplication.run(OrmLearnApplication.class, args);

}

@Override

public void run(String... args) throws Exception {

testGetDepartment();

}

void testGetDepartment() {

Department department = departmentService.get(1);

System.out.println(department);

System.out.println(department.getEmployeeList());

for (Employee employee : department.getEmployeeList()) {

System.out.println(employee);

}

}

}

**OUTPUT:**

Department [id=1, name=HR]

[Employee [id=101, name=John, salary=50000.0, permanent=true], Employee [id=102, name=Jane, salary=60000.0, permanent=false]]

Employee [id=101, name=John, salary=50000.0, permanent=true]

Employee [id=102, name=Jane, salary=60000.0, permanent=false]

**Hands on-6**

**Employee.java**

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*;

import java.util.Set;

@Entity

public class Employee {

@Id

private int id;

private String name;

private double salary;

private boolean permanent;

@ManyToOne

@JoinColumn(name = "dp\_id")

private Department department;

@ManyToMany(fetch = FetchType.EAGER)

@JoinTable(name = "employee\_skill",

joinColumns = @JoinColumn(name = "es\_em\_id"),

inverseJoinColumns = @JoinColumn(name = "es\_sk\_id"))

private Set<Skill> skillList;

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 double getSalary() { return salary; }

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

public boolean isPermanent() { return permanent; }

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

public Department getDepartment() { return department; }

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

public Set<Skill> getSkillList() { return skillList; }

public void setSkillList(Set<Skill> skillList) { this.skillList = skillList; }

}

**2. Skill.java**

package com.cognizant.ormlearn.model;

import jakarta.persistence.\*;

import java.util.Set;

@Entity

public class Skill {

@Id

private int id;

private String name;

@ManyToMany(mappedBy = "skillList")

private Set<Employee> employeeList;

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 Set<Employee> getEmployeeList() { return employeeList; }

public void setEmployeeList(Set<Employee> employeeList) { this.employeeList = employeeList; }

}

**3. OrmLearnApplication.java**

package com.cognizant.ormlearn;

import com.cognizant.ormlearn.model.Employee;

import com.cognizant.ormlearn.model.Skill;

import com.cognizant.ormlearn.service.EmployeeService;

import com.cognizant.ormlearn.service.SkillService;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import jakarta.transaction.Transactional;

@SpringBootApplication

public class OrmLearnApplication {

@Autowired

private EmployeeService employeeService;

@Autowired

private SkillService skillService;

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

public static void main(String[] args) {

SpringApplication.run(OrmLearnApplication.class, args);

}

@Transactional

public void testGetEmployee() {

Employee employee = employeeService.get(1);

LOGGER.debug("Employee:{}", employee);

LOGGER.debug("Department:{}", employee.getDepartment());

LOGGER.debug("Skills:{}", employee.getSkillList());

}

@Transactional

public void testAddSkillToEmployee() {

Employee employee = employeeService.get(1);

Skill skill = skillService.get(2);

employee.getSkillList().add(skill);

employeeService.save(employee);

}

}

**OUTPUT:**

**testGetEmployee():**

DEBUG com.cognizant.ormlearn.OrmLearnApplication - Employee:Employee{id=1, name='John', salary=60000.0, permanent=true}

DEBUG com.cognizant.ormlearn.OrmLearnApplication - Department:Department{id=1, name='Technology'}

DEBUG com.cognizant.ormlearn.OrmLearnApplication - Skills:[Skill{id=1, name='Java'}, Skill{id=2, name='Spring'}]

**testAddSkillToEmployee() :**

DEBUG com.cognizant.ormlearn.OrmLearnApplication - Employee:Employee{id=1, name='John', salary=60000.0, permanent=true}

DEBUG com.cognizant.ormlearn.OrmLearnApplication - Skill:Skill{id=3, name='SQL'}