ASSIGNMENT 9

Name :- Prajapati Meet B

Roll No :- CE050

Batch :- A3

Employee.java

```java
package com.ddu.ce.crud_demo.entity;
import jakarta.persistence.*;
@Entity
@Table(name="Employees")
public class Employee {

	@Id
	@GeneratedValue(strategy = GenerationType.IDENTITY)
	@Column(name = "Emp_id")
	private int id;
	@Column(name = "Emp_name")
	private String name;
	@Column(name = "Emp_dep")
	private String department;
	@Column(name = "Emp_sal")
	private double salary;

	public Employee() {}

	public Employee(String name, String department, double salary) {
		super();
		this.name = name;
		this.department = department;
		this.salary = salary;
	}
	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 String getDepartment() {
		return department;
	}
	public void setDepartment(String department) {
		this.department = department;
	}
	public double getSalary() {
		return salary;
	}
	public void setSalary(double salary) {
		this.salary = salary;
	}
	@Override
	public String toString() {
		return "Employee [id=" + id + ", name=" + name + ",
department=" + department + ", salary=" + salary + "]";
	}



}
```

Employee_imp.java

```
package com.ddu.ce.crud_demo.dao;
```

```
import com.ddu.ce.crud_demo.entity.Employee;
import java.util.List;
public interface Employee_imp {
	void save(Employee theStudent);
	Employee findById(Integer id);
	List<Employee> findAll();
	void update(Employee theStudent);
	void delete(Integer id);

}
```

Employee_detail.java

```
package com.ddu.ce.crud_demo.dao;
import com.ddu.ce.crud_demo.entity.Employee;
import jakarta.persistence.EntityManager;
import jakarta.persistence.TypedQuery;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;
import org.springframework.transaction.annotation.Transactional;
import java.util.List;
@Repository
public class Employee_detail implements Employee_imp{

	// define field for entity manager
		private EntityManager entityManager;
		// inject entity manager using constructor injection
		public Employee_detail(@Autowired EntityManager
entityManager) {
			this.entityManager = entityManager;
		}
		// implement save method
```

```
	@Override
	@Transactional
	public void save(Employee Emp) {
		entityManager.persist(Emp);
	}
	@Override
	public Employee findById(Integer id) {
		return entityManager.find(Employee.class, id);
	}
	@Override
	public List<Employee> findAll() {
		// create query
		TypedQuery<Employee> theQuery =
entityManager.createQuery("FROM Employee", Employee.class);
		// return query results
		return theQuery.getResultList();
	}
	@Override
	@Transactional
	public void update(Employee Emp) {
		entityManager.merge(Emp);
	}
	@Override
	@Transactional
	public void delete(Integer id) {
		// retrieve the student
		Employee Emp = entityManager.find(Employee.class,
id);
		// delete the student
		entityManager.remove(Emp);
	}

}
```

Demo_Employee.java

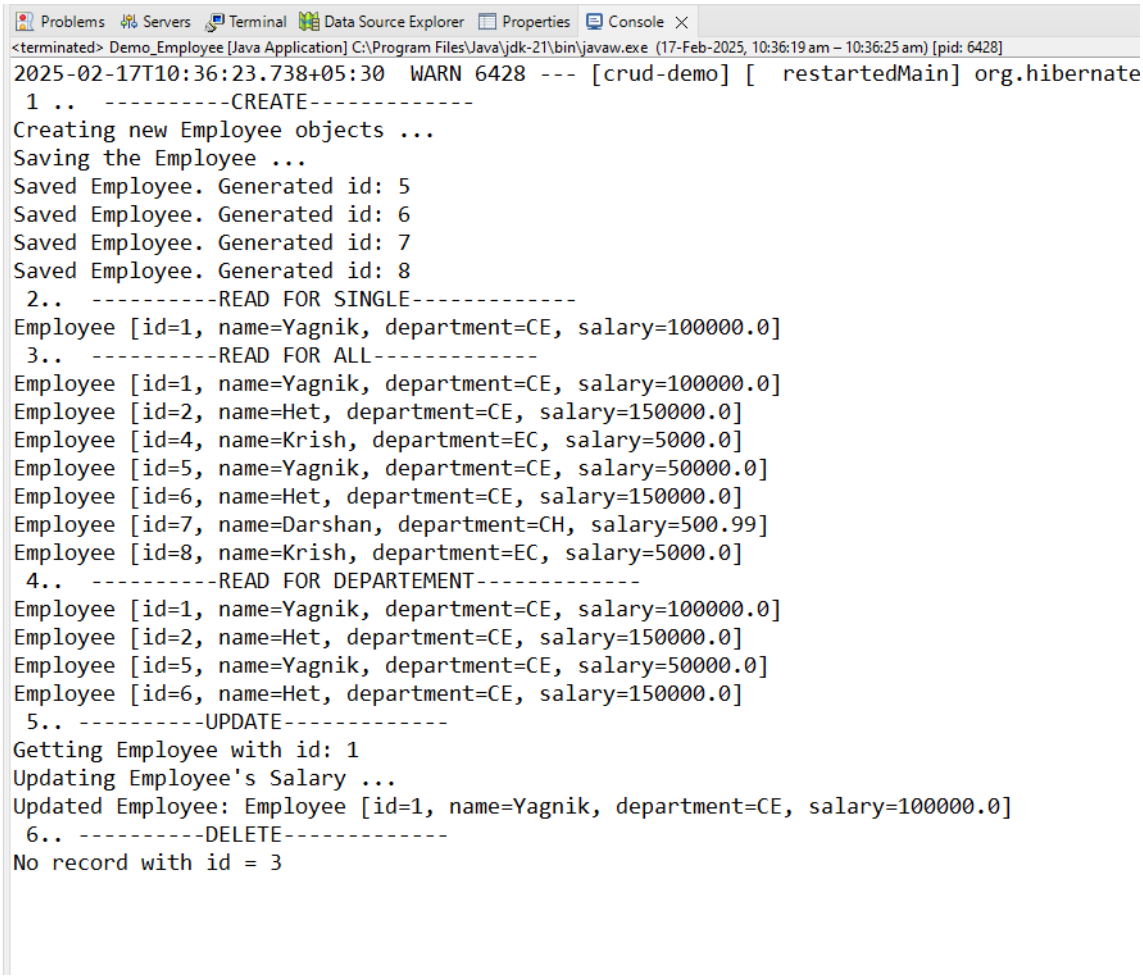
```
package com.ddu.ce.crud_demo;

import java.util.List;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;
import com.ddu.ce.crud_demo.dao.Employee_imp;
import com.ddu.ce.crud_demo.entity.Employee;

@SpringBootApplication
public class Demo_Employee {
	public static void main(String[] args) {
		SpringApplication.run(Demo_Employee.class, args);
	}
	@Bean
	CommandLineRunner commandLineRunner(Employee_imp
Emp) {
		return runner -> {
			System.out.println(" 1 ..  ----------CREATE ----------- ");
			createEmployee(Emp);
			System.out.println(" 2..  ----------READ FOR
SINGLE ------------");
			readEmployee(Emp);
			System.out.println(" 3..  ----------READ FOR
ALL ------------ ");
			readEmployeeForAll(Emp);
			System.out.println(" 4..  ----------READ FOR
DEPARTEMENT ------------ ");
```

```
                read_dep(Emp,"CE");
                System.out.println(" 5.. ----------UPDATE ----------");
                updateEmployee(Emp);
                System.out.println(" 6.. ----------DELETE---------- ");
                deleteEmployee(Emp);
        };
}
private void deleteEmployee(Employee_imp Emp) {
        int EmpId = 3;
        Employee myEmp = Emp.findById(EmpId);
        if (myEmp == null)
                System.out.println("No record with id = " + EmpId);
        else {
                System.out.println("Deleting Employee id: " + EmpId);
                Emp.delete(EmpId);
        }
}
private void updateEmployee(Employee_imp Emp) {
        // retrieve student based on the id: primary key
        int EmpId = 1;
        System.out.println("Getting Employee with id: " + EmpId);
        Employee myEmp = Emp.findById(EmpId);
        // change first name to "John"
        System.out.println("Updating Employee's Salary ...");
        myEmp.setSalary(100000.00);
        // update the student
        Emp.update(myEmp);
        // display the updated student
        System.out.println("Updated Employee: " + myEmp);
}
private void readEmployeeForAll(Employee_imp Emp) {
        // get a list of students
```

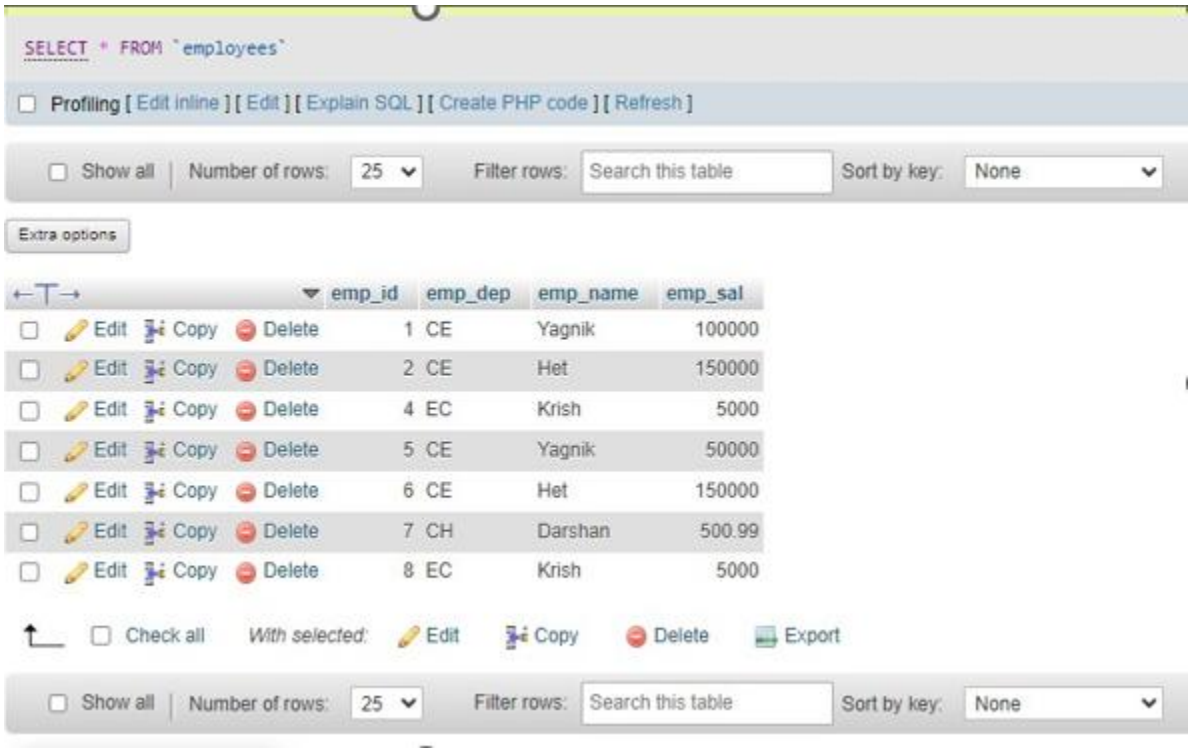
```
            List<Employee> theEmp = Emp.findAll();
            // display list of students
            for (Employee tempEmp : theEmp) {
                  System.out.println(tempEmp);
            }
      }
      private void createEmployee(Employee_imp Emp) {
            // create the student object
            System.out.println("Creating new Employee objects ...");
            Employee tempEmp1 = new Employee("Yagnik",
"CE",50000.00);
            Employee tempEmp2 = new Employee("Het",
"CE",150000.00);
            Employee tempEmp4 = new Employee("Krish",
"EC",5000.00);
            Employee tempEmp3 = new Employee("Darshan",
"CH",500.99);
            // save the student object
            System.out.println("Saving the Employee ...");
            Emp.save(tempEmp1);
            Emp.save(tempEmp2);
            Emp.save(tempEmp3);
            Emp.save(tempEmp4);
            // display id of the saved student
            System.out.println("Saved Employee. Generated id: " +
tempEmp1.getId());
            System.out.println("Saved Employee. Generated id: " +
tempEmp2.getId());
            System.out.println("Saved Employee. Generated id: " +
tempEmp3.getId());
            System.out.println("Saved Employee. Generated id: " +
tempEmp4.getId());
```

```
        }

        private void readEmployee(Employee_imp Emp) {
                // get a list of students
                int Eid = 1;
                Employee theEmp = Emp.findById(Eid);
                // display list of students

                System.out.println(theEmp);
        }

        private void read_dep(Employee_imp Emp,String dep) {

                List<Employee> theEmp = Emp.findAll();
                // display list of students
                for (Employee tempEmp : theEmp) {
                        if(tempEmp.getDepartment().equals(dep))
                        System.out.println(tempEmp);
                }

        }
}
```

Output :-


Problems
Servers
Terminal
Data Source Explorer
Properties
Console
<terminated> Demo_Employee [Java Application] C:\Program Files\Java\jdk-21\bin\javaw.exe (17-Feb-2025, 10:36:19 am – 10:36:25 am) [pid: 6428]
2025-02-17T10:36:23.738+05:30 WARN 6428 --- [crud-demo] [restartedMain] org.hibernate
1 .. ----------CREATE-------------
Creating new Employee objects ...
Saving the Employee ...
Saved Employee. Generated id: 5
Saved Employee. Generated id: 6
Saved Employee. Generated id: 7
Saved Employee. Generated id: 8
2.. ----------READ FOR SINGLE-------------
Employee [id=1, name=Yagnik, department=CE, salary=100000.0]
3.. ----------READ FOR ALL-------------
Employee [id=1, name=Yagnik, department=CE, salary=100000.0]
Employee [id=2, name=Het, department=CE, salary=150000.0]
Employee [id=4, name=Krish, department=EC, salary=5000.0]
Employee [id=5, name=Yagnik, department=CE, salary=50000.0]
Employee [id=6, name=Het, department=CE, salary=150000.0]
Employee [id=7, name=Darshan, department=CH, salary=500.99]
Employee [id=8, name=Krish, department=EC, salary=5000.0]
4.. ----------READ FOR DEPARTEMENT-------------
Employee [id=1, name=Yagnik, department=CE, salary=100000.0]
Employee [id=2, name=Het, department=CE, salary=150000.0]
Employee [id=5, name=Yagnik, department=CE, salary=50000.0]
Employee [id=6, name=Het, department=CE, salary=150000.0]
5.. ----------UPDATE-------------
Getting Employee with id: 1
Updating Employee's Salary ...
Updated Employee: Employee [id=1, name=Yagnik, department=CE, salary=100000.0]
6.. ----------DELETE-------------
No record with id = 3

DATABASE :-


SELECT * FROM `employees`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None
Extra options
emp_id emp_dep emp_name emp_sal
Edit Copy Delete 1 CE Yagnik 100000
Edit Copy Delete 2 CE Het 150000
Edit Copy Delete 4 EC Krish 5000
Edit Copy Delete 5 CE Yagnik 50000
Edit Copy Delete 6 CE Het 150000
Edit Copy Delete 7 CH Darshan 500.99
Edit Copy Delete 8 EC Krish 5000
Check all With selected: Edit Copy Delete Export
Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None