

Práctica con Eclipse Link. Mapear tablas como objetos con JPA.

**Base datos
bdemp de
mysql**

Por Jorge María Huguet

Índice

1.	Eliminar departamento	3
2.	Modificar departamento.....	4
3.	Lista empleados.....	4
4.	Insertar empleado	5
5.	Modificar empleado	6
6.	Borrar empleado	7

1. Eliminar departamento

Una vez seguidos los pasos guiados de la práctica se crea la clase DeptDelete para borrar departamentos de la tabla Dept.

Para ello en la línea marcada el objeto depDel almacena la información del departamento 70, y con el método remove() usado más abajo lo borramos de la tabla.

```
public class DeptDelete {  
    public static void main(String[] args) {  
  
        EntityManagerFactory emf = Persistence.createEntityManagerFact  
        EntityManager em = emf.createEntityManager();  
        Dept depDel= new Dept();  
        depDel=em.find(Dept.class,70);  
        em.getTransaction().begin();  
        em.remove(depDel);  
        em.getTransaction().commit();  
    }  
}
```

En la clase DeptInsertar.java insertamos el departamento 70 RRHH.

```
run:  
[EL Info]: 2023-01-18 14:15:59.413--Server  
[EL Info]: connection: 2023-01-18 14:16:00  
nombre departamento: ACCOUNTING  
nombre departamento: RESEARCH  
nombre departamento: SALES  
nombre departamento: OPERATIONS  
nombre departamento: RRHH  
BUILD SUCCESSFUL (total time: 3 seconds)
```

Al ejecutar la nueva clase DeptDelete.java el departamento se elimina.

```
run:  
[EL Info]: 2023-01-18 14:17:52.821--Serve  
[EL Info]: connection: 2023-01-18 14:17:5  
nombre departamento: ACCOUNTING  
nombre departamento: RESEARCH  
nombre departamento: SALES  
nombre departamento: OPERATIONS  
BUILD SUCCESSFUL (total time: 3 seconds)
```

2. Modificar departamento

Al igual que en la clase anterior usamos el departamento 70 que va a cambiar de nombre de RRHH a IT.

```
public class DeptModify {
    public static void main(String[] args) {

        EntityManagerFactory emf = Persistence.createEntityManagerFactory("persistence");
        EntityManager em = emf.createEntityManager();
        Dept depDel= new Dept();
        depDel=em.find(Dept.class, 70);
        System.out.println("El departamento se llama: "
            + depDel.getDname());
        depDel.setDname("IT");
        System.out.println("Tras el cambio se llama: "
            + depDel.getDname());
        em.getTransaction().begin();
        em.getTransaction().commit();
    }
}
```

Output - PersistenciaDBemp (run) x

```
run:
[EL Info]: 2023-01-18 14:31:08.439--ServerSession(815320891)--EclipseLink, version 2.6.4
[EL Info]: connection: 2023-01-18 14:31:09.166--ServerSession(815320891)--file:/home/.../persistenceDBemp.jar
El departamento se llama: IT
tras el cambio se llama: RRHH
BUILD SUCCESSFUL (total time: 3 seconds)
```

3. Lista empleados

Se crea la nueva clase para listar los empleados usando las clases creadas anteriormente de persistencia de datos.

```
public class EmpView {
    public static void main(String[] args) {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("persistence");
        EmpJpaController dao = new EmpJpaController(emf);
        List<Emp> lista =dao.findEmpEntities();
        for (Emp empleado : lista) {
            System.out.println("Nombre: " + empleado.getEname()+
                "\tpuesto: "+empleado.getJob());
        }
    }
}
```

Output - PersistenciaDBemp (run) x

```
run:
[EL Info]: 2023-01-18 14:50:38.101--ServerSession(815320891)--EclipseLink, version 2.6.4
[EL Info]: connection: 2023-01-18 14:50:38.746--ServerSession(815320891)--file:/home/.../persistenceDBemp.jar
Nombre: KING    puesto: PRESIDENT
Nombre: BLAKE   puesto: MANAGER
Nombre: CLARK   puesto: MANAGER
Nombre: JONES   puesto: MANAGER
Nombre: MARTIN  puesto: SALESMAN
Nombre: ALLEN   puesto: SALESMAN
Nombre: TURNER  puesto: SALESMAN
Nombre: JAMES   puesto: CLERK
Nombre: WARD    puesto: SALESMAN
Nombre: FORD    puesto: ANALYST
Nombre: SMITH   puesto: CLERK
Nombre: SCOTT   puesto: ANALYST
Nombre: ADAMS   puesto: CLERK
Nombre: MILLER  puesto: CLERK
BUILD SUCCESSFUL (total time: 3 seconds)
```

4. Insertar empleado

Se crea la clase EmpInsert y se llaman a los diferentes métodos set para insertar al empleado.

```
public class EmpInsert {
    public static void main(String[] args) {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("persistence");
        EntityManager em = emf.createEntityManager();
        Emp insertEmp = new Emp();
        insertEmp.setEmpno(9000);
        insertEmp.setName("BOND");
        insertEmp.setJob("CLERK");
        insertEmp.setMgr(4500);
        Date hiredate = new Date(1988, 9, 30);
        insertEmp.setHiredate(hiredate);
        BigDecimal sal = new BigDecimal(4500);
        insertEmp.setSal(sal);
        BigDecimal comm = new BigDecimal(500);
        insertEmp.setComm(comm);
        insertEmp.setDeptno(70);

        em.getTransaction().begin();
        em.persist(insertEmp);
        em.getTransaction().commit();
    }
}
```

persistenceadbemp.EmpInsert > main > hiredate

out x

persistenceADBemp (run) x SQL 1 execution x

Nombre: MARTIN	puesto: SALESMAN
Nombre: ALLEN	puesto: SALESMAN
Nombre: TURNER	puesto: SALESMAN
Nombre: JAMES	puesto: CLERK
Nombre: WARD	puesto: SALESMAN
Nombre: FORD	puesto: ANALYST
Nombre: SMITH	puesto: CLERK
Nombre: SCOTT	puesto: ANALYST
Nombre: ADAMS	puesto: CLERK
Nombre: MILLER	puesto: CLERK
Nombre: BOND	puesto: CLERK

BUILD SUCCESSFUL (total time: 2 seconds)

5. Modificar empleado

Al igual que con la clase de DeptModify, se crea un objeto Emp y se incluye la información del empleado 9000, posteriormente se modifica el registro con el método setName().

```
public class EmpModify {  
  
    public static void main(String[] args) {  
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("persistence");  
        EntityManager em = emf.createEntityManager();  
        Emp modEmp = new Emp();  
        modEmp = em.find(Emp.class, 9000);  
        System.out.println("El nombre actual es: " + modEmp.getName());  
  
        modEmp.setName("STARK");  
        System.out.println("Se ha modificado por: " + modEmp.getName());  
  
        em.getTransaction().begin();  
        em.persist(modEmp);  
        em.getTransaction().commit();  
    }  
}
```

persistenciadbemp.EmpModify >

Output - PersistenciaDBemp (run) x

run:
[EL Info]: 2023-01-18 15:15:32.933--ServerSession(815320891)--E
[EL Info]: connection: 2023-01-18 15:15:33.608--ServerSession(81
El nombre actual es: BOND
Se ha modificado por: STARK
BUILD SUCCESSFUL (total time: 2 seconds)

6. Borrar empleado

En el objeto delEmp de la clase Emp, se almacena la información del empleado 9000, con el método remove() se elimina de la tabla. Al ejecutar el la clase EmpView se muestran todos los empleados , como STARK fue eliminado no aparece.

```
public class EmpDelete {
    public static void main(String[] args) {
        EntityManagerFactory emf = Persistence.createEntityManagerFactory("persistence");
        EntityManager em = emf.createEntityManager();
        Emp delEmp = new Emp();
        delEmp = em.find(Emp.class, 9000);

        em.getTransaction().begin();
        em.remove(delEmp);
        em.getTransaction().commit();
    }
}
```

persistenciadbemp.EmpDelete > main >

Output - PersistenciaDBemp (run) x

```
run:
[EL Info]: 2023-01-18 15:25:45.966--ServerSession(815320891)--Ec
[EL Info]: connection: 2023-01-18 15:25:46.615--ServerSession(81
Nombre: KING      puesto: PRESIDENT
Nombre: BLAKE     puesto: MANAGER
Nombre: CLARK     puesto: MANAGER
Nombre: JONES     puesto: MANAGER
Nombre: MARTIN    puesto: SALESMAN
Nombre: ALLEN     puesto: SALESMAN
Nombre: TURNER    puesto: SALESMAN
Nombre: JAMES     puesto: CLERK
Nombre: WARD      puesto: SALESMAN
Nombre: FORD      puesto: ANALYST
Nombre: SMITH     puesto: CLERK
Nombre: SCOTT     puesto: ANALYST
Nombre: ADAMS     puesto: CLERK
Nombre: MILLER    puesto: CLERK
BUILD SUCCESSFUL (total time: 3 seconds)
```