111---

```
public static void main(String[] args) {
    Logger.getLogger("org.hibernate").setLevel(Level.SEVERE);

    SessionFactory sf = null;
    Session s = null;
    Transaction t = null;

    //  Clase calendar para definir la fecha de la clase Date
    Calendar cal = Calendar.getInstance();
    Date date = new Date();
    cal.setTime(date);
    cal.add(Calendar.DAY_OF_MONTH, -9);
    cal.add(Calendar.MONTH, -1);
    cal.add(Calendar.YEAR, -55);
    try{
        sf = HibernateUtil.getSessionFactory();
        s = HibernateUtil.abrirSesion();
        t = s.beginTransaction();


        //  A) agregamos las 5 tareas usando hibernate
        date = cal.getTime();
        for(int i=1; i<6; i++){
            s.save(new Tareas("Tarea"+i, new Date(), date, false));
        }


        //  B) HQL para agregar los valores de una tabla definida con la misma
        // estructura que la principal
        String hql = "insert into tareas (id, descripcion, fecha_inicio, fecha_fin, \n" +
                "finalizada) select v.id, v.descripcion, v.fecha_inicio, \n" +
                "v.fecha_fin, v.finalizada from tareasValues v";

        Query query = s.createSQLQuery(hql);
        int cant = query.executeUpdate();
        System.out.println("Cantidad de tareas agregada: " + cant);


        //  C) Consulta usando HQL del contenido de Tareas
```

```
//  C) Consulta usando HQL del contenido de Tareas
String hqlConsulta = "from Tareas";
List<Tareas> listaTareas = s.createQuery(hqlConsulta, Tareas.class).getResultList();
listaTareas.forEach(System.out::println);


//  D) guardamos las tareas con ID pedidos
Tareas t1 = s.get(Tareas.class, 2);
Tareas t2 = s.get(Tareas.class, 4);

//  contamos la cantidad de tareas que podremos actualizar
int contAct = actualizarTarea(t1);
contAct += actualizarTarea(t2);

//  Las actualizamos
s.update(t1);
s.update(t2);

System.out.println("Filas Actualizadas: " + contAct);

//  Imprimimos la lista que ya teniamos definida y solo cogemos las 2 tareas que queriamos
listaTareas.stream().filter(x -> (x.getId()%2==0&&x.getId()<5)).forEach(System.out::println);


//  E) Actualizamos de la misma manera pero usando HQL
String hqlUpdate = "update tareas set fecha_fin=:fe, finalizada=:fin where id in (7,9)";
Query queryUp = s.createSQLQuery(hqlUpdate);
queryUp.setParameter("fe", new Date());
queryUp.setParameter("fin", true);

int resUpdate = queryUp.executeUpdate();

System.out.println("Cantidad de filas actualizadas: " + resUpdate);


//  F) Aprovechando la lista que ya teniamos usando hql, la volvemos a mostrar con los valores
//  que nos interesa
```

```
//  F) Aprovechando la lista que ya teniamos usando hql, la volvemos a mostrar con los valores
//  que nos interesa
s.clear();
listaTareas = s.createQuery(hqlConsulta, Tareas.class).getResultList();
listaTareas.stream().filter(x -> (x.getId()==7||x.getId()==9)).forEach(System.out::println);


//  G)  usando HQL eliminamos la terea por el id pedido por consola
System.out.println("Dame el id para eliminar del registro: ");
short id = new Scanner(System.in).nextShort();

int existe = (int)listaTareas.stream().filter(x -> (x.getId()==id)).count();

if(existe>0){
    String hqlDelete = "delete from tareas where id=:id";
    Query queryDelete = s.createSQLQuery(hqlDelete);
    queryDelete.setParameter("id", id);
    int del = queryDelete.executeUpdate();
    System.out.println("Filas borradas: (" + del + ")...");
}else
    System.out.println("No existe el id introducido...!");


    H)  Mostramos las tareas sin la terea borrada, algo curioso es que
    cuando hacemos actualizaciones o inserciones una misma consulta HQL nos sirve
    pero cuando borramos un registro este no lo ve, por lo que lo vuevlo a definir
    para cargarlo con los valores borrados.

listaTareas = s.createQuery(hqlConsulta, Tareas.class).getResultList();
listaTareas.forEach(System.out::println);

t.commit();
```

222---

```
public static void main(String[] args) {
    Logger.getLogger("org.hibernate").setLevel(Level.SEVERE);

    SessionFactory sf = null;
    Session s = null;
    Transaction t = null;

    try{
        sf = HibernateUtil.getSessionFactory();
        s = HibernateUtil.abrirSesion();
        t = s.beginTransaction();

        //  3)
        String hqlConsulta = "select p.codigo, p.nombre, p.precio, p.codigoFabricante, p.fabricante.nombre"
                           + "from producto p join p.fabricante f";
        List<Object[]> productoList = s.createQuery(hqlConsulta, Object[].class).getResultList();


        productoList.forEach(x -> System.out.println("Fabricante: " + x[0] + " Producto: " + x[1]));

        //  4)
        String hql = "from Producto";
        List<Producto> listProducto = s.createQuery(hql, Producto.class).getResultList();
        System.out.println("********************** LISTA DE TODOS LOS PRODUCTOS **********************");
        listProducto.forEach(System.out::println);

        System.out.println("Dame el nombre del fabricante: ");
        String fabr = new Scanner(System.in).nextLine();

        int existe = (int)s.createQuery("from Fabricante", Fabricante.class)
                .getResultList().stream().filter(x -> x.getNombre().equals(fabr.toUpperCase())).count();

        if(existe>0){
            String hqlUpdate = "update producto set precio=precio*1.1 \n" +
                               "where codigo_fabricante in (select codigo " +
                               "from fabricante where nombre=:nom);";
            Query queryUpdate = s.createSQLQuery(hqlUpdate);
            queryUpdate.setParameter("nom", fabr.toUpperCase());
```

```
    Query queryUpdate = s.createSQLQuery(hqlUpdate);
    queryUpdate.setParameter("nom", fabr.toUpperCase());

    int cant = queryUpdate.executeUpdate();
    System.out.println("Cantidad de filas afectadas " + cant);
}else{
    System.out.println("El proveedor no existe en la base de datos");
    System.exit(0);
}

System.out.println("********************* LISTA DE TODOS LOS PRODUCTOS ACTUALIZADOS **********************");
s.clear();
listProducto = s.createQuery(hql, Producto.class).getResultList();
listProducto.forEach(System.out::println);

    //  6) Elimina todos los productos que contengan 'monitor'
    //  en este ejemplo seran los que contengan una 0
    String hqlDelete = "delete from producto where nombre like '%o%'";
    Query queryDelete = s.createSQLQuery(hqlDelete);
    int cantDel = queryDelete.executeUpdate();
    System.out.println("Cantidad de productos borrados: " + cantDel);
    s.clear();
    listProducto = s.createQuery(hql, Producto.class).getResultList();
    listProducto.forEach(System.out::println);

t.commit();
```

44444

```
public static void main(String[] args) {
    Logger.getLogger("org.hibernate").setLevel(Level.SEVERE);

    SessionFactory sf = null;
    Session s = null;
    Transaction t = null;

    try{
        sf = HibernateUtil.getSessionFactory();
        s = HibernateUtil.abrirSesion();
        t = s.beginTransaction();

        //  Creacion Sede y Departamento
          Sede sede = new Sede("Sevilla");
          s.save(sede);
          Departamento dep = new Departamento(sede, "I+D");
          s.save(dep);

          Empleado e1 = new Empleado("111111111", dep, "Pepe");
          Empleado e2 = new Empleado("222222222", dep, "Ana");
          Empleado e3 = new Empleado("333333333", dep, "Fiona");

          s.save(e1);
          s.save(e2);
          s.save(e3);


        /* Datos Prueba
        Sede sede = new Sede("Sevilla");
        s.save(sede);
        Departamento dep = new Departamento(sede, "I+D");
        s.save(dep);

        Empleado e1 = new Empleado("111111111", dep, "Pepe");
        Empleado e2 = new Empleado("222222222", dep, "Ana");
        Empleado e3 = new Empleado("333333333", dep, "Fiona");

        s.save(e1);
```

```
            //  Consulta a 2 tablas prueba, creamos una consulta adaptada a HQL sobre 2 tablas relacionadas y este
            //  nos devolvera un array de Object con las clases de las tablas seleccionadas, en este caso Empleado y
            //  Departamento que luego iteramos para ver los datos de estos con la relacion que queriamos.
//            String hqlCons = "FROM Empleado e, Departamento d WHERE e.departamento.idDepto=d.idDepto";
//            List<Object[]> empleados = s.createQuery(hqlCons, Object[].class).getResultList();
//            empleados.forEach(a -> System.out.println(a[0]+"-"+a[1]));

            //  Mostrar todos los empleados del departamento (1) en este caso le pasamos el id para que nos devuelva
            //  el objeto con el id, so podria comprobar con un if() si este existe pero paso de ponerlo xd
            Departamento depto = s.get(Departamento.class, 1);
            s.refresh(depto);

            System.out.println("************** DATOS DE EMPLEADOS **************");
            List<Empleado> empleados = s.createQuery("FROM Empleado WHERE departamento.idDepto = :id", Empleado.class).setParameter("id", depto.getIdDepto()).getResultList();
            System.out.println("empleados totales: " + empleados.size());
            empleados.forEach(emp -> System.out.println(emp.getNomEmp()));

            //  Mostrar Departamentos sobreescribiendo el metodo toString()
            String hql = "FROM Departamento";
            Query q = s.createQuery(hql);

            System.out.println("************** LEEMOS TODOS LOS DEPARTAMENTOS **************");
            System.out.println("**************************** DATOS ****************************");

            List <Departamento> listResultados = q.list();

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


            //  Primera forma aprendida en clase, el problema es que en la misma ejecucion no te coge el departamento
            //  que acabas de crear, mientras que con la consulta manual HQL esta si la reconoce.
//            Departamento deptoPrueba = s.get(Departamento.class, 10);
//            s.refresh(deptoPrueba);
//
//            if(deptoPrueba != null){
//                System.out.println("************** DATOS DE EMPLEADOS **************");
//
//                List<Empleado> empleados2 = new ArrayList<>(deptoPrueba.getEmpleados());
```

Activar Windov
Ve a Configuración

```
//                List<Empleado> empleados2 = new ArrayList<>(deptoPrueba.getEmpleados());
//                empleados2.forEach(emp -> System.out.println(emp.getNomEmp()));
//
//                String hql2 = "FROM Departamento";
//                Query q2 = s.createQuery(hql2);
//
//                System.out.println("************** LEEMOS TODOS LOS DEPARTAMENTOS **************");
//                System.out.println("**************************** DATOS ****************************");
//
//                List <Departamento> listResultados2 = q2.list();
//
//                listResultados2.forEach(System.out::println);
//
//            }else
//                System.out.println("El departamento no existe en la base de datos");
//
            t.commit();
        }catch(HibernateException he){
            System.err.println("Error Hibernate: " + he.getMessage());

            if(t!=null)t.rollback();
        }finally{
            HibernateUtil.cerrarSession(s);
            HibernateUtil.cerrarSessionFactory();
        }
    }
}
```