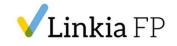


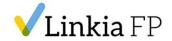
Formación Profesional Oficial a Distancia



DAW - M03 - Clase 10

Base de datos orientada a objetos – db4o





Contenidos

- Db4o una alternativa open source.
- Ventajas.
- Inconvenientes.
- Inicializaciones.
- Grabar.
- Consultas.



db4o

- Db4o (Data Base For Objects) es una implementación OODB libre (bajo GPL – General Public Licence, por lo tanto de código abierto) que permite de manera sencilla hacer persistentes objetos creado con Java y recuperarlos posteriormente en otra ejecución.
- Db4o no es exactamente una base de datos OO que sigue ODMG ya que, entre otras cosas, no implemente un lenguaje OQL.

db40



Ventajas db4o

- Mayor velocidad de desarrollo (transparencia)
- Mejor rendimiento con objetos de negocio complejos (árboles, estructuras anidadas, relaciones N a N, relaciones recursivas)
- Fácil Backup (la base completa está en un solo archivo)
- No necesita administración
- Las búsquedas se hacen directamente usando objetos
- Los cambios en los objetos (agregar o quitar atributos a una clase) se aplican directamente en la base, sin tener que migrar datos ni reconfigurar nada.
- Multiplataforma.



Inconvenientes db4o

- No existe un lenguaje de consultas como sql (se deben realizar programáticamente)
- No existen restricciones, deben programarse (No implementa integridad referencial).
- Tamaño limitado de los ficheros de BBDD (2GB 264GB).
- No permite clustering.
- No existe gestión de permisos.
- No existen aplicaciones para data mining ni informes.
- Para accesos sencillos y persistencia de objetos simples los motores de persistencia ofrecen un rendimiento similar.



Inicialización

```
private ObjectContainer db;
db = Db4oEmbedded.openFile("datos.dat");
// ObjectContainer - Clase principal de
persistencia con db4o
```



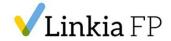
Grabar

```
private void grabarAlumno(Alumno a) {
  db.store(a);
}
```



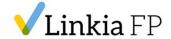
Consulta básica

```
public Alumno selectAlumnoByNombre(Alumno a) {
      Alumno aux;
      Query q = db.query();
      q.constrain(Alumno.class);
      q.descend("nombre").contrain(a.getNombre());
      ObjectSet resultado = q.execute();
      aux = (Alumno) resultado.next();
      return aux;
```



Consulta básica con ArrayList

```
public List<Alumno> selectAllAlumnos() {
   ArrayList<Alumno> alumnos = new ArrayList<>();
   Query q = db.query();
   q.constrain(Alumno.class);
   ObjectSet resultado = q.execute();
   while (resultado.hasNext()) {
      Alumno a = (Alumno) resultado.next();
      alumnos.add(a);
   return alumnos;
```



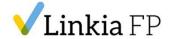
Ejemplo de consultas avanzadas

```
// Query con constraint de edad superior a 20
Query q = db.query();
q.constrain(Alumno.class);
q.descend("nota").constraint(20).greater();
// ordenar por edad
q.constrain(Alumno.class);
q.descend("nota").orderAscending();
//Query con doble constraint, para establecer un rango
q.contraint(Alumno.class);
Constraing edadMin = q.descend("edad").constraing(10).greater();
q.descend("edad").constrain(20).smaller().and(edadMin);
```



Consulta básica con ArrayList

```
public List<Alumno> selectAllAlumnosRange(int edadMinima, int
edadMaxima) {
   ArrayList<Alumno> alumnos = new ArrayList<>();
   Query q = db.query();
   q.constrain(Alumno.class);
   Constraint edadMin =
q.descend("edad").constraint(edadMinima).greater();
 q.descend("edad").constrain(edadMaxima).smaller().and(edadMin);
   ObjectSet resultado = q.execute();
   while (resultado.hasNext()) {
      Alumno a = (Alumno) resultado.next();
      alumnos.add(a);
   return alumnos;
```



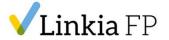
Modificar

```
public void modificarAlumno (Alumno antiguo, Alumno
nuevo) {
   ObjectSet resultado = db.queryByExample(antiguo);
   Alumno aux = (Alumno) resultado.next();
   aux.setNombre(nuevo.getNombre);
   db.store(aux);
```



Borrar

```
private void borrarAlumno(Alumno a) {
     Query q = db.query();
     q.contrain(Alumo.class);
     q.descend("nombre").contraing(a.getNombre());
     ObjectSet resultado = q.execute();
     Alumno aux = (Alumno) resultado.next();
     db.delete(aux);
```



Cerrar la conexión

```
private void cerrarConexion() {
    db.close();
    //Si necesitamos podemos borrar el fichero
    File f = new File("archivo");
    f.delete();
}
```



Formación Profesional Oficial a Distancia