Proyecto XML

Fernando Ortega castro

1. La importancia de las bbdd nativas en XML en general.

La principal función de las bbdd nativas en XML es proporcionar un lenguaje estructurado más sencillo de leer y de entender, además de muy útil en el envio de información entre programas. Esto otorga ventajas, principalmente a la hora de intercambiar información, a partir de las bases de datos en XML se pueden generar archivos con otras extensiones para ser compartidos y la información sea más fácil de leer, por ejemplo documentos PDF.

Por otro lado cuenta con otras ventajas como la facilidad para añadir nuevas etiquetas, importar y exportar los datos, muy manejables y otras ventajas.

2. Por qué he usado eXist-db.

La base de datos que elegí es eXist-db, es una base de datos XML nativa de código abierto. eXist-db es conocida por su flexibilidad y su conjunto completo de características, incluyendo un potente motor de indexación y búsqueda, soporte para XQuery, y RESTful API. También ofrece una interfaz web para la administración de datos y una de escritorio.

La elegí esta BD por que otorga soporte para diversos documentos además de diversos modos de acceso y compatibilidad tanto para windows como para ubuntu.

Otro factor en la elección fue el que ya había trabajado con alguna de las mas famosas como Base X y había tenido problemas asi que decidí aventurarme por esta de codigo abierto y con una documentación bastante completa.

3. CRUD en la bbdd.

Mi base de datos eXist-db trabaja por colecciones para el gestionar los datos lo mejor es usar el cliente para crear la colección de la base de datos y luego la colección en mi caso de perfiles, dentro de esta colección ya esta perfiles.xml y los que se deseen introducir.

La lectura e inserción de los datos se pueden hacer con consultas como XPath o editando el xml. Puedes importar o crear los archivos o colecciones.

El CRUD en mi programa es el siguiente:

/// **Lectura**
////////// Codigo programa
//
public static List<Perfil> getPerfilesFromCollection (Collection col) throws Exception{

```
        // Leer preparando una consulta de XPath
        XPathQueryService xpqs =
(XPathQueryService)col.getService("XPathQueryService", "1.0");
        xpqs.setProperty("indent", "yes");

        // Guardaremos los perfiles obtenidos de la consulta
        List<Perfil> perfiles = new ArrayList<>();
        Perfil p;

        // Consulta para obtener todos los perfiles de la coleccion
        ResourceSet result = xpqs.query("for $perfil in collection(\"/\")/perfiles/perfil\n"+
        "return concat($perfil/id, ', ', $perfil/login, ', ', $perfil/contrasenia, ', ',
$perfil/nombreUsuario, ', ', $perfil/email)");

        // Iteramos en el resultado de la consulta
        ResourceIterator i = result.getIterator();

        Resource res = null;
        while(i.hasMoreResources()) {
            try {

                // Obtenemos el resultado de la consulta
                res = i.nextResource();

                // DEBUG
                //System.out.println("-->"+res.getContent().toString());
```

```
                // Dividimos el resultado en cada atributo
                String [] datos = res.getContent().toString().split(", ");

                // Rellenamos el perfil con los datos obtenidos
                p = new Perfil(Long.parseLong(datos[0].trim()), datos[1].trim(),
datos[2].trim(), datos[3].trim(), datos[4].trim());

                // Añadimos el perfil al vector
                perfiles.add(p);

            } finally {
                // Limpiamos el resultado
                try { ((EXistResource)res).freeResources(); } catch(XMLDBException xe)
{xe.printStackTrace();}
            }

        }

        // Retornamos los perfiles obtenidos
        return perfiles;
    }

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/// Insertado
////////// Codigo programa
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
public static void setNewPerfilToCollection (Collection col, Perfil newp) throws
XMLDBException{

        // Obtenemos el xml de perfiles
        XMLResource res = (XMLResource) col.getResource("perfiles.xml");

        // Pasamos los perfiles a String para manejarlos
        String perfilesBD = (String) res.getContent();

        // Para agregar el nuevo perfil lo insertamos en el string para mas tarde
reemplazzar el archivo
        perfilesBD = perfilesBD.replace("</perfiles>", perfilToXML(newp)+"</perfiles>");

        // Reemplazamos el contenido del archivo por el nuevo contenido con el perfil
ya insertado
```

```
        res.setContent(perfilesBD);

        // Guardamos el archivo
        col.storeResource(res);

    }


////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/// Modificado
////////// Codigo programa
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
    public static void savePerfilToCollection (Collection col, Perfil pmod) throws
XMLDBException{

        // Obtenemos el xml de perfiles
        XMLResource res = (XMLResource) col.getResource("perfiles.xml");

        // Pasamos los perfiles a String para manejarlos
        String perfilesBD = (String) res.getContent();

        // reemplazamos el perfil seleccionado por sus nuevos datos
        perfilesBD =
perfilesBD.replaceAll("<perfil>[\\s]*?<id>"+pmod.getId()+"</id>[\\s\\S]*?</perfil>",
perfilToXML(pmod));

        // Reemplazamos el contenido del archivo por el nuevo contenido con el perfil
ya modificado
        res.setContent(perfilesBD);

        // Guardamos el archivo
        col.storeResource(res);

    }

////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

/// Eliminado
////////// Codigo programa
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
    public static void deletePerfilToCollection (Collection col, Perfil pdel) throws
XMLDBException{
```

```
        // Obtenemos el xml de perfiles
        XMLResource res = (XMLResource) col.getResource("perfiles.xml");

        // Pasamos los perfiles a String para manejarlos
        String perfilesBD = (String) res.getContent();

        // eliminamos el perfil seleccionado
        perfilesBD =
perfilesBD.replaceAll("<perfil>[\\s]*?<id>"+pdel.getId()+"</id>[\\s\\S]*?</perfil>", "");


        // Reemplazamos el contenido del archivo por el nuevo contenido sin el perfil
        res.setContent(perfilesBD);

        // Guardamos el archivo
        col.storeResource(res);

    }
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

4. Programa.

Para configurar el acceso a la base de datos debemos de añadir las siguientes dependencias:

```
//////////////////////////////////////
<dependencies>
        <dependency>
            <groupId>net.sf.xmldb-org</groupId>
            <artifactId>xmldb-api</artifactId>
            <version>1.7.0</version>
        </dependency>

        <dependency>
            <groupId>org.exist-db</groupId>
            <artifactId>exist-core</artifactId>
            <version>6.2.0</version>
            <type>jar</type>
        </dependency>
    </dependencies>
//////////////////////////////////////
```

Usaremos el siguiente driver: “org.exist.xmldb.DatabaseImpl”

Y la ruta que nos ofrece eXist-db de base para el acceso es la siguiente:
“xmldb:exist://localhost:8080/exist/xmlrpc/db/”

Imports necesarios:

```
///
import org.xmldb.api.base.*;
import org.xmldb.api.modules.*;
import org.xmldb.api.*;
import org.exist.xmldb.EXistResource;
///

//////////// Ejecución del codigo del programa de prueba
//////////// (módulos del CRUD explicados en el anterior apartado)
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////


/**
 *
 * @author diosfer
 */
public class PruebaBDXML {

    private static final String URI =
"xmldb:exist://localhost:8080/exist/xmlrpc/db/PruebaXML";

    private static final String collection = "/Perfiles";

    private static List<Perfil> perfilesSistema;

    public static void main(String[] args) throws Exception{

        // Driver
        final String driver = "org.exist.xmldb.DatabaseImpl";

        // initialize database driver
        Class cl = Class.forName(driver);
        Database database = (Database) cl.newInstance();
        database.setProperty("create-database", "true");
        DatabaseManager.registerDatabase(database);
```

```
        // Coleccion que obtendremos
        Collection col = null;

        try {

            // Obtenemos la coleccion
            col = DatabaseManager.getCollection(URI+collection, "admin",
"contrasenia1234");


            /// LECTURA ///

            // Obtenemos los perfiles de la coleccion a traves del metodo
            perfilesSistema = getPerfilesFromCollection(col);

            /// DEBUG Comprobamos que se hallan obtenido los perfiles
            for (Perfil p : perfilesSistema){
                System.out.println(p);
            }


            /// INSERTADO ///

            // Creamos un nuevo perfil

            System.out.println("Insertamos un nuevo perfil");

            Perfil newp = new Perfil ("nuevoPerfilLogin", "nuevoPerfilContrasenia",
"nuevoPerfilUserName", "nuevoPerfilEmail");

            setNewPerfilToCollection(col, newp);


            /// MODIFICADO ///

            System.out.println("Modificamos el perfil perfiamente añadido");

            Perfil pmod = newp;
            pmod.setLogin("loginMod");
            pmod.setPasword("contraseniaMod");
            pmod.setEmail("emailMod");
            pmod.setNombreUsuario("userNameMod");

            savePerfilToCollection(col, pmod);
```

```
        /// ELIMINADO ///

        System.out.println("Eliminamos el perfil perfiamente modificado");

        Perfil pdel = pmod;
        deletePerfilToCollection(col, pdel);

    }
    catch (Exception e){
        e.printStackTrace();
    }
    finally {
        //Cerramos la coleccion
        if(col != null) {
            //Cerramos la coleccion
            try { col.close(); } catch(XMLDBException xe) {xe.printStackTrace();}
        }
    }
}
```

//

5. Conclusión.

Las bases de datos en xml tienen sus ventajas y desventajas como bin hable de ellas en el primer apartado. Para resumir su mayor ventaja la claridad y la compatibilidad se pueden ver opacadas por su lentitud y diferencias con las bases de datos relacionales se pueden llegar a ver opacadas por las otras bases de datos aunque tengan relevancia en su propio sector.

Las otras bases de datos tambien tienen sus ventajas y desventajas como la flexibilidad de mongo o el orden y optimización de sql.

En mi opinión y para concluir me iría mas hacia una base de datos relacional por el orden y tipo de gestión que maneja pero siguen siendo una opción valida para almacenar nuestros datos.

6. Webgrafía.

1 Documentació oficial eXist-db.
2 Instalación
3 Datos de las BBDD