Exist-DB

Nombre: Caio Medeiros

Índice

Introducción

Exist-db es un sistema de gestión de bases de datos XML no relacional de código abierto. Permite almacenar, buscar y manipular datos en formato XML de manera eficiente. Tiene una interfaz web fácil de usar y ofrece un gran rendimiento al trabajar con grandes conjuntos de datos. También tiene una serie de características avanzadas como la indexación automática, la integración con otras tecnologías (como XQuery y XSLT) y una API para programar acceso a los datos.

Alguna de sus características son:

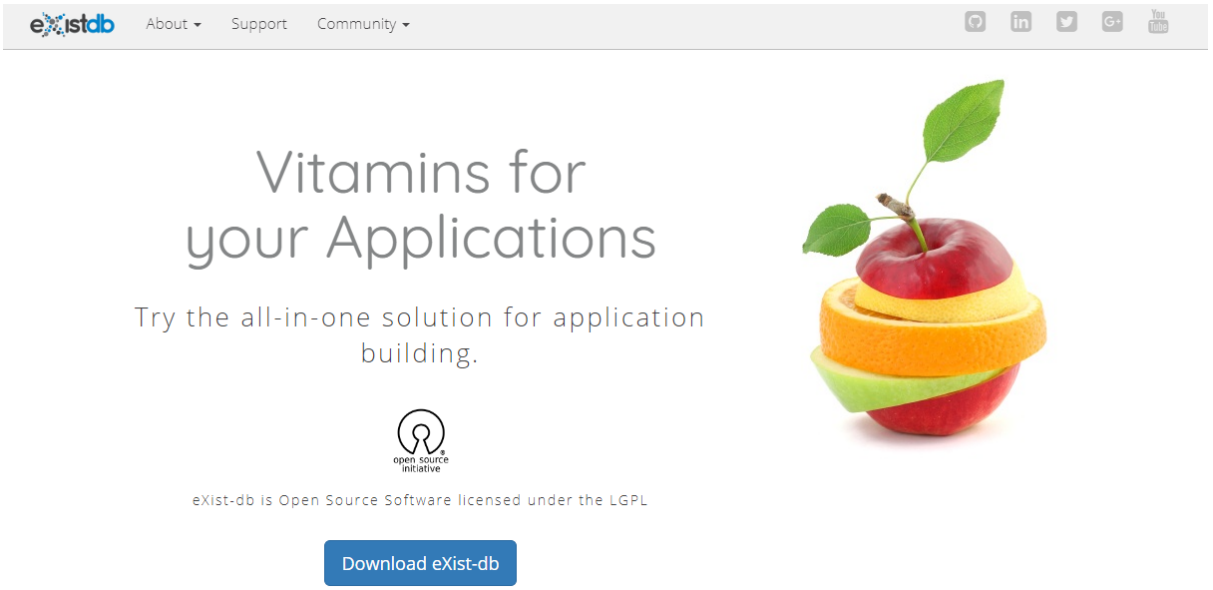
- Almacenamiento de datos en formato XML, lo que permite una estructura jerárquica y una mejor integración con lenguajes de marcas.
- Soporte para consultas XPath y XQuery, lo que facilita la recuperación de datos.
- Integración con tecnologías web, como XSLT, CSS y JavaScript, para la presentación de datos.
- Funcionalidad de búsqueda completa, incluyendo búsqueda de palabras completas, frases y operadores lógicos.
- Sistema de seguridad integrado, con opciones para autenticación y autorización.

- Compatibilidad con una variedad de sistemas operativos, incluyendo Windows, macOS y Linux.
- API disponible para varios lenguajes de programación, incluyendo Java, Python y PHP.

Instalación y descarga

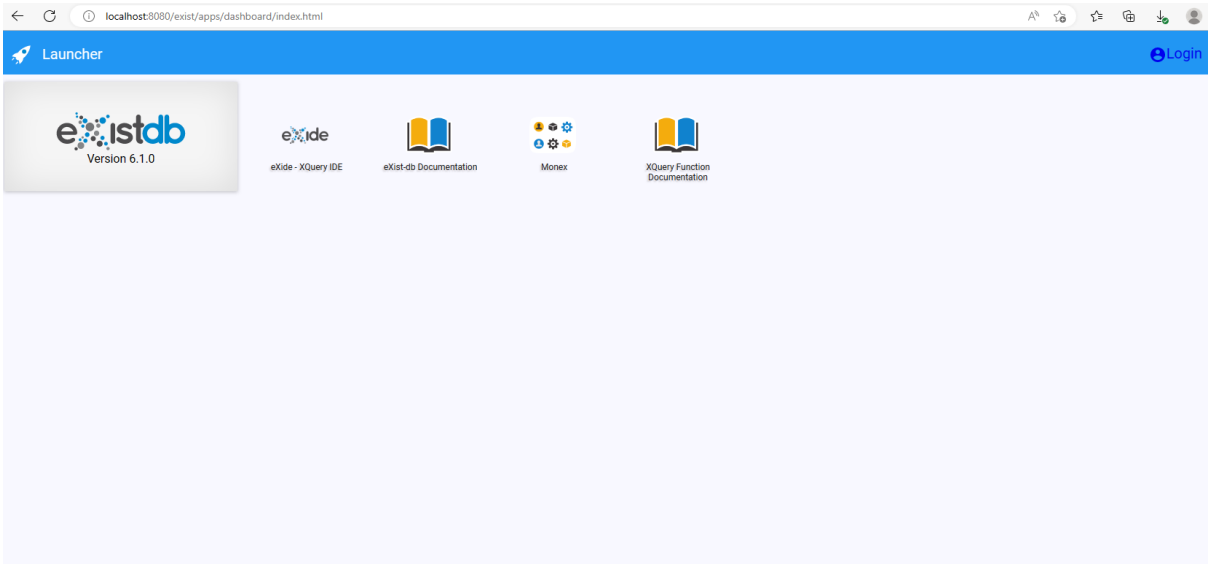
Exist-db se puede descargar e instalar en varios sistemas operativos, incluyendo Windows, MacOS y Linux. El proceso de instalación puede variar ligeramente según el sistema operativo. A continuación, se presenta un resumen general del proceso de instalación en un sistema operativo Windows:

1. Descargue el paquete de instalación de Exist-db desde la página de descargas en el sitio web oficial.



2. Ejecute el paquete de instalación y siga las instrucciones en pantalla para instalar el software.

3. Una vez completada la instalación, abra un navegador web y vaya a la URL "http://localhost:8080" para acceder a la interfaz web de Exist-db.



4. Utilice las credenciales que has colocado en la instalación de exist-db

Requerimientos mínimos

Los requisitos mínimos para ejecutar Exist-db son los siguientes:

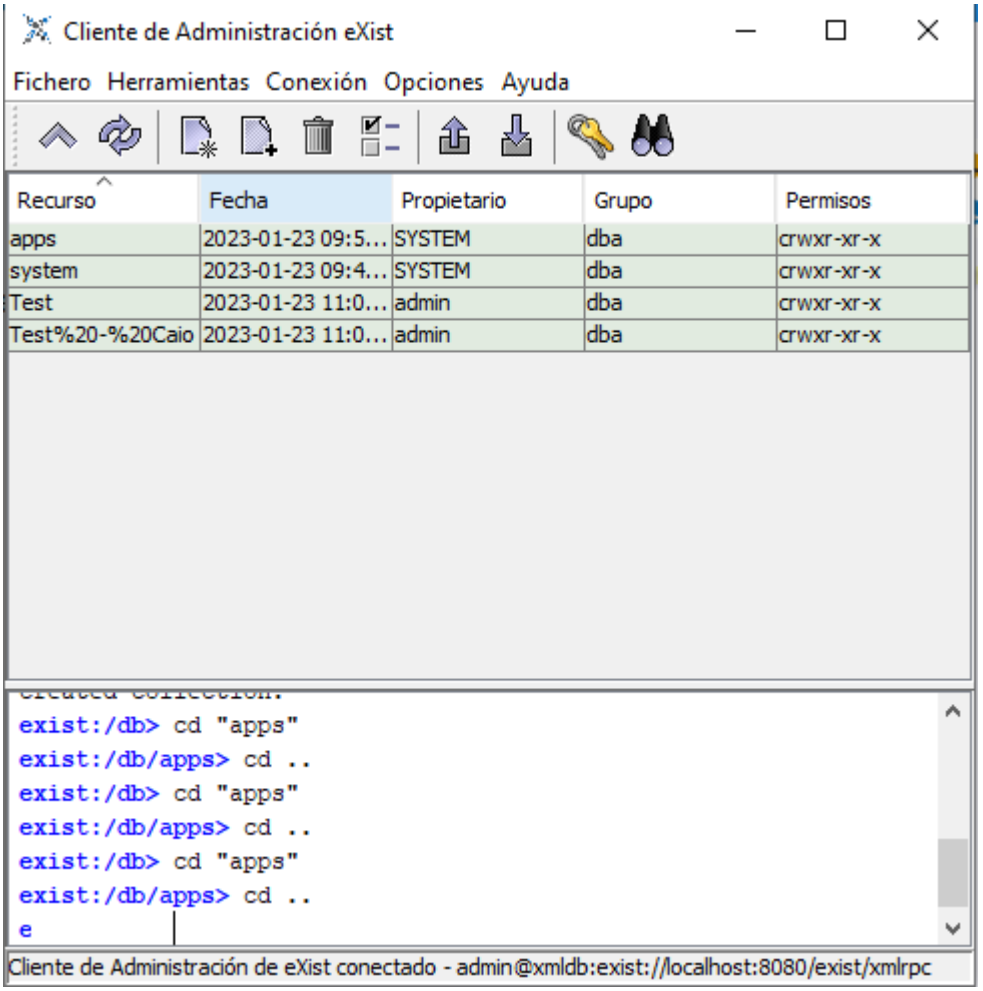
- Sistema operativo: Windows, MacOS o Linux.
- Procesador: Intel Pentium 4 o equivalente.
- Memoria RAM: 2 GB.
- Espacio en disco duro: 1 GB.
- Java: Versión 8 o superior.

Es importante tener en cuenta que estos son los requisitos mínimos y que el rendimiento de Exist-db puede verse afectado con estos recursos. Para trabajar con grandes conjuntos de datos o cargas de trabajo intensas, se recomienda contar con recursos superiores.

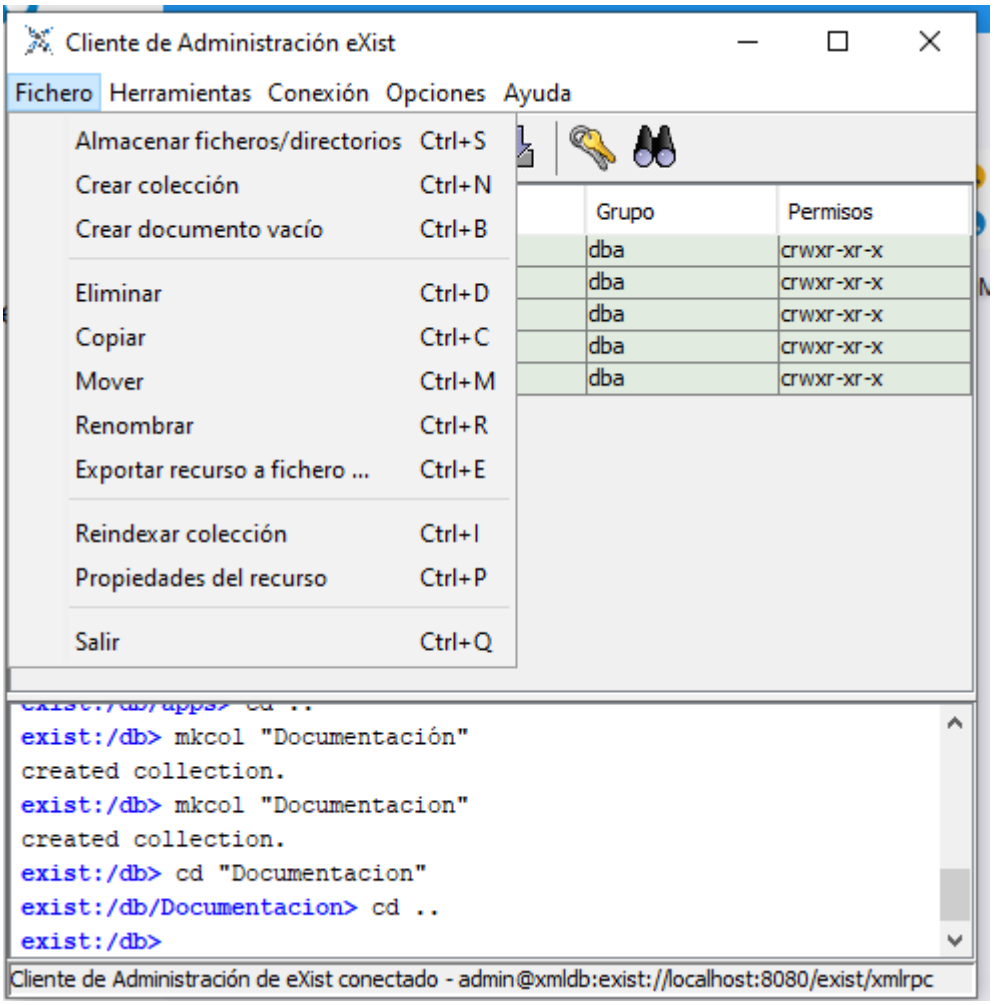
Es importante mencionar que Exist-db está desarrollado en Java, por lo que es necesario tener instalado un JRE o JDK (Java Runtime Environment or Java Development Kit) antes de instalar exist-db, de lo contrario, no podrá ser ejecutado.

Introducción de datos utilizando la GUI del sistema

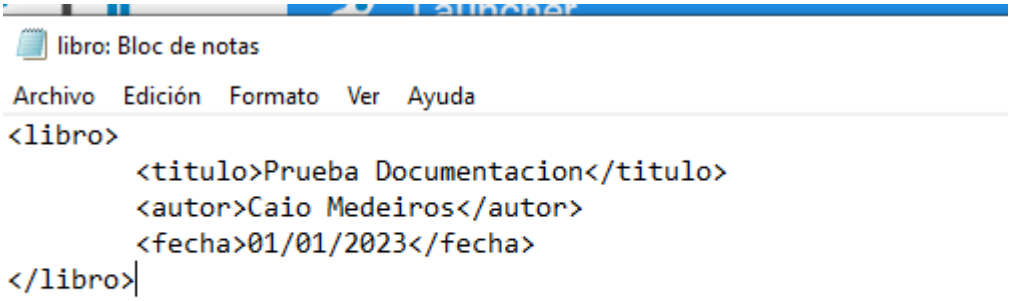
1. Iniciar el ejecutable de exist-db.



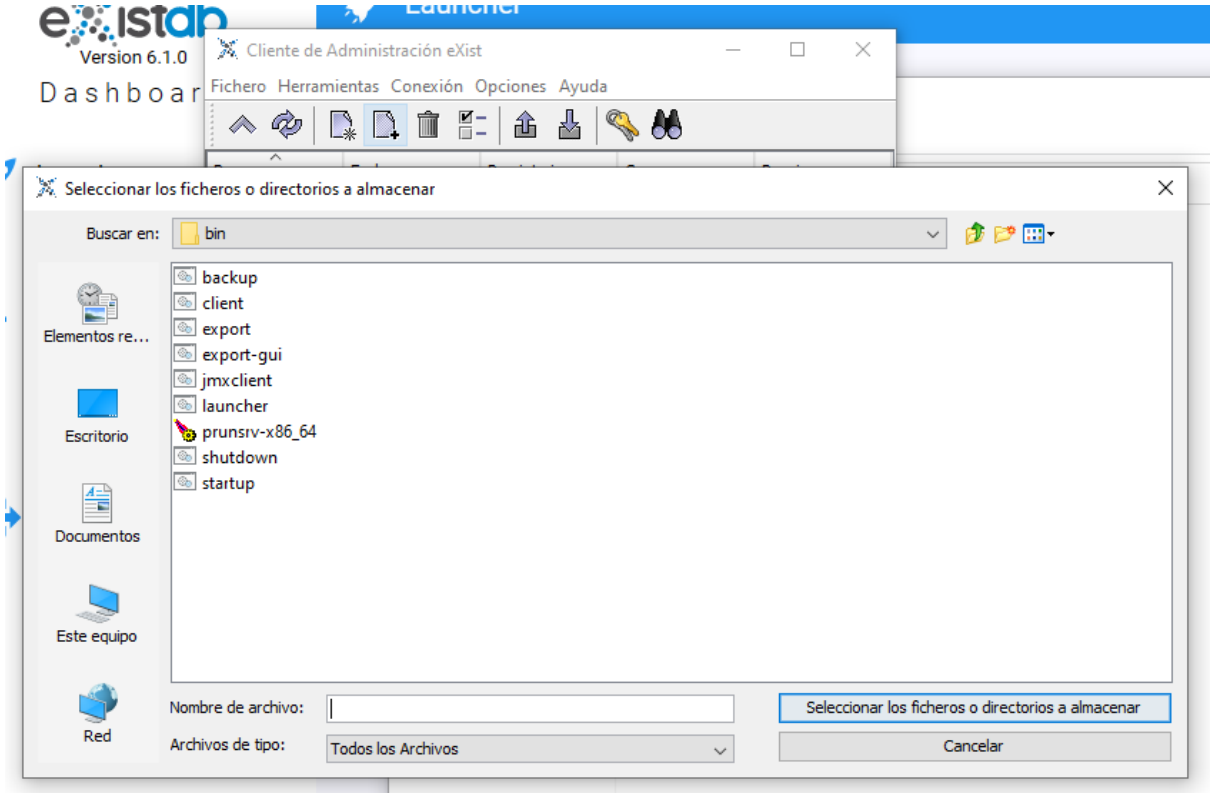
2. Creamos una nueva colección llamada “Documentacion”:



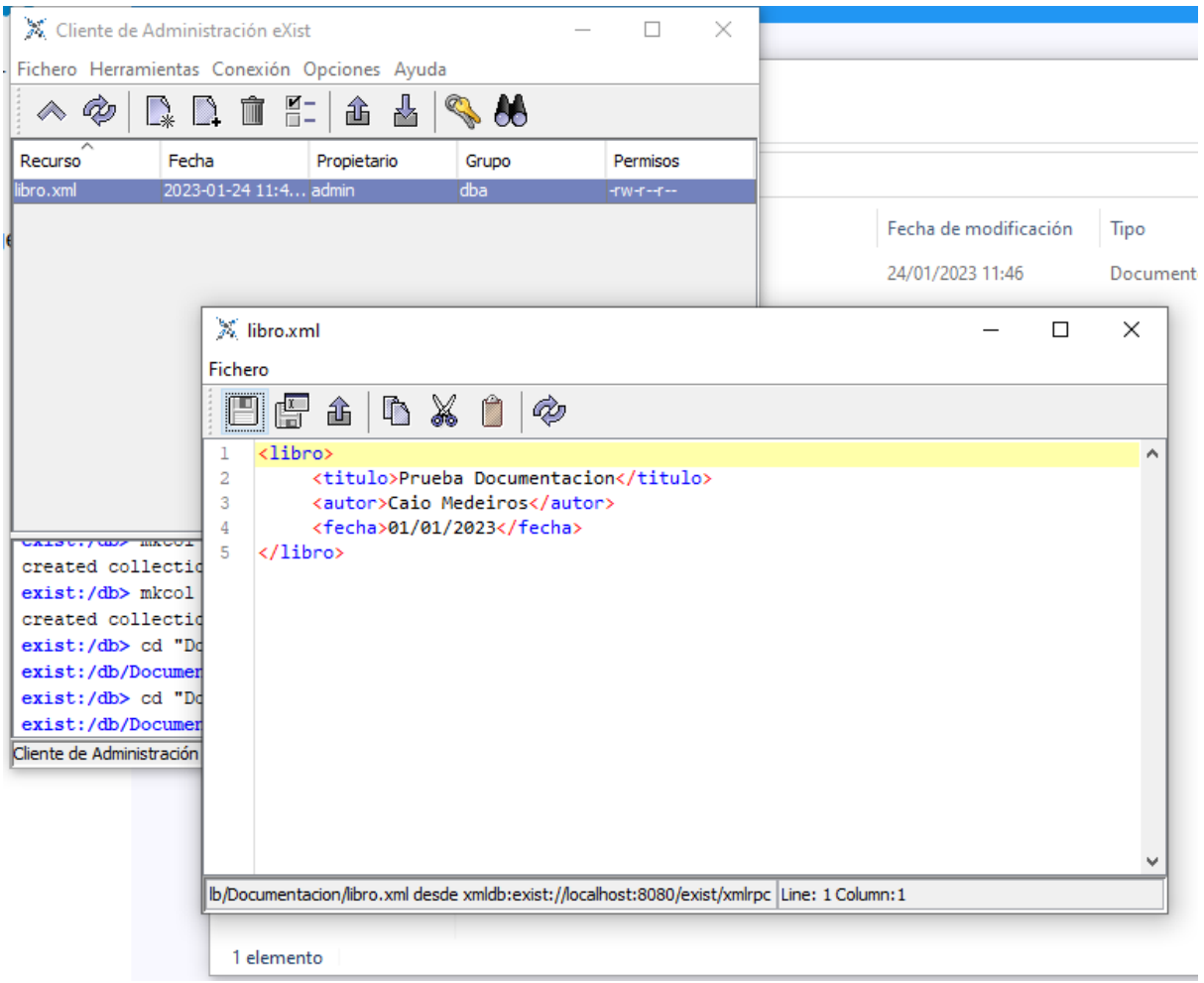
3. Creamos un XML de ejemplo.



4. Entramos a la colección creado y almacenamos el fichero.



5. Visualizamos que se haya subido correctamente.


Cliente de Administración eXist
Fichero Herramientas Conexión Opciones Ayuda
Recurso Fecha Propietario Grupo Permisos
libro.xml 2023-01-24 11:4... admin dba -rw-r--r--
Fecha de modificación Tipo
24/01/2023 11:46 Document
libro.xml
Fichero
1 <libro>
2 <titulo>Prueba Documentacion</titulo>
3 <autor>Caio Medeiros</autor>
4 <fecha>01/01/2023</fecha>
5 </libro>
exist:/db> mkcol
exist:/db> cd "Do
lb/Documentacion/libro.xml desde xmldb:exist://localhost:8080/exist/xmlrpc
Line: 1 Column: 1
1 elemento

Configuración de un proyecto Java con dicho sistema

1. Asegurarse de que se tiene una versión funcional de Exist-db. Para ello, comprobar que se puede acceder correctamente a "http://localhost:8080"
2. Crear un proyecto Java en tu IDE de preferencia
3. Si has creado un proyecto Maven, agregar la siguiente dependencia en el pom.xml

```
<dependency>
     <groupId>org.exist-db</groupId>
     <artifactId>exist-core</artifactId>
     <version>6.1.0</version>
     <type>jar</type>
 </dependency>
```

En cambio, si se ha creado un proyecto Gradle, agregar la siguiente dependencia al archivo build.gradle.

```
dependencies {
    implementation 'org.exist:exist-core:6.1.0'
}
```

4. Hacer un "Clean & Build" desde Netbeans para que se instalen las dependencias.

Explicación y código de las operaciones CRUD

Antes de comenzar con las operaciones CRUD, hay que configurar el acceso a la base de datos por código. Para ello hay que inicializar el driver.

```
// Configuración de la base de datos
final String URI = "xmldb:exist://192.168.152.129:8080/exist/xmlrpc";
final String driver = "org.exist.xmldb.DatabaseImpl";

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

Ahora sí, comenzamos con la primera operación CRUD, "CREATE".

CREATE

1. Conexión a la base de datos.

```
// Conexión a la base de datos.
// Primer parámetro: Ruta de la colección
// Segundo parámetro: Usuario
// Tercer parámetro: Contraseña para el usuario
Collection col = DatabaseManager.getCollection(URI + "/OperacionesCRUD", "admin", "admin");
```

2. Creación del documento XML que se desee insertar.

```
// Creación del elemento XML
String xml = "<libro><autor>Caio Medeiros</autor></libro>";
```

3. Creación de un objeto Resource con el documento XML.

```
// Creación del objeto Resource con el XML
// Primer parámetro: Nombre del recurso que se quiere guardar
// Segundo parámetro: Tipo de recurso (XMLResource o BinaryResource)
Resource res = col.createResource("miLibro.xml", "XMLResource");
res.setContent(xml);
```

4. Agregar el recurso a la colección.

```
// Agregar el recurso a la colección
col.storeResource(res);
```

5. Comprobar que el recurso se ha subido correctamente.
   Con un “System.out.println()” podemos comprobar que se ha subido correctamente.

```
[miLibro.xml]
```

READ

1. Conexión a la base de datos.

```
// Conexión a la base de datos.
// Primer parámetro: Ruta de la colección
// Segundo parámetro: Usuario
// Tercer parámetro: Contraseña para el usuario
Collection col = DatabaseManager.getCollection(URI + "/OperacionesCRUD", "admin", "admin");
```

2. Uso de método .listResources()

```
// Listar los recursos dentro de una coleccion
System.out.println(Arrays.toString(col.listResources()));
```

3. Ver contenido del recurso.

```
// Ver contenido del recurso
System.out.println(col.getResource(col.listResources()[0]).getContent().toString());
```

4. Ver resultado

```
] --- exec-maven-plugin:3.0.0:exec (default-cli) @
<libro><autor>Caio Medeiros</autor></libro>
------------------------------------------------
```

UPDATE

Para actualizar un recurso, simplemente hay que utilizar los mismo métodos que han sido usados para el “CREATE” indicando como id el nombre del archivo ya existente que se quiere modificar.

```
// UPDATE
// Actualización de un recurso
String xml = "<libro><autor>Gustavo</autor></libro>";
Resource res = col.createResource("miLibro.xml", "XMLResource");
res.setContent(xml);
col.storeResource(res);
```

Comprobamos que se ha actualizado correctamente.

```
<libro><autor>Gustavo</autor></libro>
```

DELETE

1. Obtener el recurso que se quiera eliminar.

```
// DELETE
// Obtener el recurso que se quiera eliminar.
// Primer parámetro: ID del recurso
Resource eliminar = col.getResource("miLibro.xml");
```

2. Hacer el borrado.

```
// Borrado
col.removeResource(eliminar);
```

3. Comprobar que se ha borrado correctamente.

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ AD_2Evaluacion ---
[]
------------------------------------------------------------------------
BUILD SUCCESS
```

Tipos de consultas e implementación

Para hacer consultas desde código en Exist-db, se utiliza la API XQuery de Exist-db. La API XQuery te permite crear consultas utilizando el lenguaje de consulta XQuery y ejecutarlas en la base de datos.

Para realizar consultas se tienen que seguir los siguientes pasos.

1. Obtener el servicio de XQuery.

```
// Obtener el servicio de XQuery
Collection col = DatabaseManager.getCollection(URI + "/OperacionesCRUD", "admin", "admin");
XQueryService service = (XQueryService) col.getService("XQueryService", "1.0");
service.setProperty("indent", "yes");
```

2. Escribir la consulta en XQuery.

```
// Escribir la consulta en XQuery
String query = "for $book in //book where $book/author = 'Gustavo' return $book";
```

3. Realizar la consulta.

```
// Realizar la consulta
ResourceSet result = service.query(query);
```

4. Comprobar los resultados.

```
// Comprobar los resultados
ResourceIterator i = result.getIterator();
while (i.hasMoreResources()) {
    Resource r = i.nextResource();
    System.out.println((String) r.getContent());
}
```

```
] --- exec-maven-plugin:3.0.0:exec (default-cli) @ AD_2Evaluacion ---
<libro>
        <titulo>Prueba Documentacion</titulo>
        <autor>Caio Medeiros</autor>
        <fecha>01/01/2023</fecha>
</libro>
```

Ahora que sabemos hacer una consulta, vamos a entrar más en profundidad con los tipos de consultas.

Consulta general

```
String query = "for $book in //libros return $book";
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ AD_2Evaluacion -
<libros>
        <libro>
                <titulo>Prueba Documentacion</titulo>
                <autor>Caio Medeiros</autor>
                <fecha>01/01/2023</fecha>
        </libro>
        <libro>
                <titulo>Prueba Documentacion 2</titulo>
                <autor>David Pérez Lledó</autor>
                <fecha>01/01/2023</fecha>
        </libro>
        <libro>
                <titulo>Prueba Documentacion 3</titulo>
                <autor>Eduardo Gil Calvo</autor>
                <fecha>01/01/2023</fecha>
        </libro>
        <libro>
                <titulo>Prueba Documentacion 4</titulo>
                <autor>Adrián Sorroche</autor>
                <fecha>01/01/2023</fecha>
        </libro>
</libros>
```

Consulta con condición

```
String query = "for $book in /libros/libro where $book/titulo = 'Prueba Documentacion' return $book";
```

```
] --- exec-maven-plugin:3.0.0:exec (default-cli) @ AD_2Evaluacion ---
  <libro>
                <titulo>Prueba Documentacion</titulo>
                <autor>Caio Medeiros</autor>
                <fecha>01/01/2023</fecha>
        </libro>
```

Proyecto elegido

El proyecto que he desarrollado sirve para administrar de manera muy básica una escuela de verano.

Las colecciones que serán usadas son escuela, alumno, curso y monitor.

Diagrama Entidad-Relación