

# Introducción a la base de datos XML: eXistDB

---

**Calbet González, María Victoria**

Trabajo desarrollado para la asignatura de Complementos de Base de Datos

Grado en Ingeniería Informática de Software

1.	Introducción	3
2.	Descripción de la tecnología	3
3.	Herramienta	5
4.	Objetivos a alcanzar	14
5.	Solución propuesta	14
6.	Bibliografía	22

# 1. INTRODUCCIÓN

---

XML es un meta-lenguaje que permite definir lenguaje de marcado desarrollado por W3C, utilizado para almacenar datos en forma legible, proveniente del lenguaje SGML (*Standard Generalized Markup Language*) y permitiendo definir la gramática de lenguajes específicos para estructurar documentos grandes.

Se creó para cumplir varios objetivos:

- Formal y conciso, desde el punto de vista de los datos y la manera de guardarlos.
- Extensible, para poderse utilizar en todos los campos del conocimiento.
- Fácil de leer y editar.
- Fácil de implantar, programar y aplicar a los distintos sistemas.

A diferencia de otros lenguajes de marcado, XML da soporte a bases de datos, de ésta forma, cualquier aplicación podría escribir un documento de texto plano con los datos que estaba manejando en formato XML y otra aplicación recibir esa información y trabajar con ella, siendo útil cuando varias aplicaciones deban comunicarse entre sí o integrar información.

Existen 2 tipos de base de datos XML distintas:

- **XML habilitado:** mapea XML en estructuras tradicionales de bases de datos, como las relacionales, aceptando XML como entrada y formateando en XML la salida. Éstas se adaptan mejor donde la mayoría de datos no están en XML.
- **XML nativo:** usa documentos XML como unidad elemental de almacenamiento, pero no han de almacenarse necesariamente en formato de texto. Éstas se adaptan mejor donde la mayoría de datos están en XML, proporcionan un modelo lógico para agrupar documentos, llamadas “colecciones”. Las bases de datos XML nativo están generalmente asociadas con las bases de datos documentales, éstas soportan al menos XPath, para realizar preguntas a documentos o colección de ellos; XSLT, para transformar documentos o resultados obtenidos de la base de datos y XQuery, semánticamente similar a SQL, proporciona los medios para extraer y manipular información de documentos XML o cualquier fuente de datos que pueda ser representada mediante XML.

## 2. DESCRIPCIÓN DE LA TECNOLOGÍA

---

Para el desarrollo de este proyecto, usaremos eXistdb como base de datos para realizar unas pequeñas consultas a un documento XML.

Se ha escogido esta base de datos, eXistdb, porque es una herramienta software de código abierto, clasificada como un sistema de bases de datos orientado a documentos NoSQL y una base de datos XML nativa, dando soporte para las siguientes tecnologías:

- |          |          |            |
|----------|----------|------------|
| • XPath  | • WebDAV | • SOAP     |
| • XQuery | • XForms | • XACML    |
| • XSLT   | • Rest   | • XInclude |
| • XSL-FO | • RestXQ | • XML-RPC  |

- XProc
- XQuery API para Java

A diferencia de otras bases de datos, su instalación es sencilla, así como su uso es orientativo y existe información suficiente como para crear nuestra base de datos. Al ser una aplicación que se usa de manera local y se permite la instalación de complementos, si en algún determinado momento, no disponemos de una conexión a internet, gracias a este manejo local y esos complementos, tendremos acceso a la información necesaria de forma asequible para poder trabajar.

También da la posibilidad de almacenar nuestra base de datos en un repositorio propio que ofrece la aplicación, así como compartir código o ideas con otros usuarios que manejen la misma herramienta. Otra posibilidad que permite eXistdb es el uso de la herramienta [oXygen XML Editor](#).

Entre todas las tecnologías que podemos emplear en eXistdb, usaremos XQuery.

XQuery es un lenguaje de consulta diseñado para escribir consultas sobre colecciones de datos expresadas en XML, abarcando desde archivos XML hasta bases de datos relacionales con funciones de conversión de registros a XML, su principal función es extraer información de un conjunto de datos organizados como un árbol de etiquetas XML.

Al ser un lenguaje funcional, cada consulta es una expresión que es evaluada y devuelve un resultado, pudiendo combinarse diversas expresiones de una manera flexible con otras expresiones para crear nuevas expresiones más complejas y de mayor potencia semántica.

Una consulta en Xquery es una expresión que lee una secuencia de datos en XML y devuelve como resultado otra secuencia de datos en XML. En Xquery, las expresiones y los valores que devuelven son dependientes del contexto, es decir, los nodos que aparezcan en el resultado dependerán de los namespaces, la posición, la etiqueta raíz, etc.

Las consultas siguen la norma FLWOR (For, Let, Where, Order y Return):

Cláusula	Descripción
<b>For</b>	Crea un flujo de tuplas, en el que cada una está vinculada a una variable
<b>Let</b>	Vincula una variable al resultado completo de una expresión añadiendo esos vínculos a las tuplas generadas por la cláusula <b>for</b> , y si no existe ninguna cláusula <b>for</b> , crea una única tupla que contenga esos vínculos.
<b>Where</b>	Filtra las tuplas
<b>Order by</b>	Ordena las tuplas según el criterio dado
<b>Return</b>	Construye el resultado de la consulta para una tupla dada.

Partiendo de esta base, se puede construir las consultas que uno desee, dependiendo de la consulta o su complicidad, se usarán las funciones que sean necesarias, pero siempre se ha de tener en cuenta la indentación de los nodos padres, nodos hijos, atributos, elementos, etc.

Una vez seleccionada nuestra herramienta y lenguaje de base de datos, deberemos tener a nuestra disposición un archivo XML, por ello, se aplicará en base a un documento de requisitos y a un modelado de procesos de negocio BPMN.

Existen aplicaciones que generan documentos de tipo XML importantes y aplicables al desarrollo cotidiano del trabajo de un Ingeniero de Software. Podemos destacar para esta tarea dos herramientas importantes, REM, conocida herramienta desarrollada para la gestión de requisitos, diseñada para soportar la fase de Ingeniería de Requisitos, y Camunda, aplicación gratuita de software libre y online para crear BPMN. Estas dos herramientas generan documentos XML que servirán para realizar las consultas que se deseen. El documento XML generado por la herramienta REM, no presenta ningún tipo de estructura, en cambio, Camunda, presenta una estructura de datos Schema para BPMN.

### 3. HERRAMIENTA

Hoy en día son numerosas las herramientas usadas que generan documentos XML. Algunas de estas herramientas son usadas por ingenieros informáticos, como Camunda, BonitaSoft o REM, siendo las dos primeras herramientas destinadas para la creación de BPMN y la tercera para desarrollar la gestión de requisitos. Otras herramientas que presentan este tipo de meta-lenguaje pertenecen al Gobierno de España, como son Aemet (Agencia estatal de meteorología) o la Agencia Tributaria; Aemet, permite descargar la predicción detallada de la consulta que se ha realizado en XML y la Agencia Tributaria, que a partir de este mes de Julio, se implantará un cambio en el sistema de gestión actual del IVA dónde actualmente encontrándose en la fase de pruebas, se puede acceder a un cliente básico de servicio web para realizar pruebas de envío de XML de libros de IVA y los documentos, facturas, tendrán una base en XML.

Descargar XML de la predicción detallada de Sevilla XML

Ilustración 1: Aemet

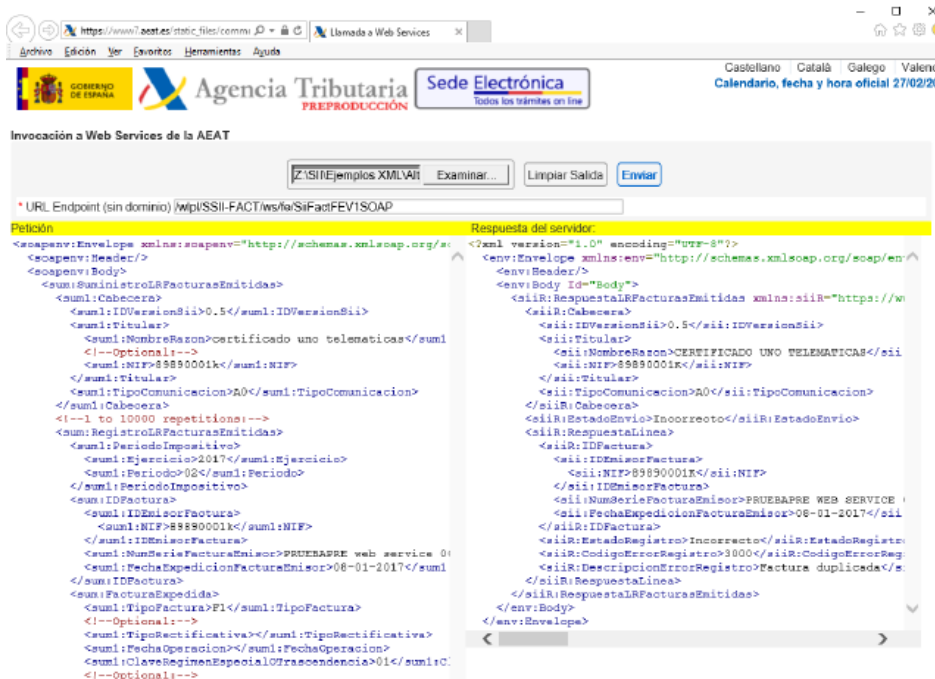
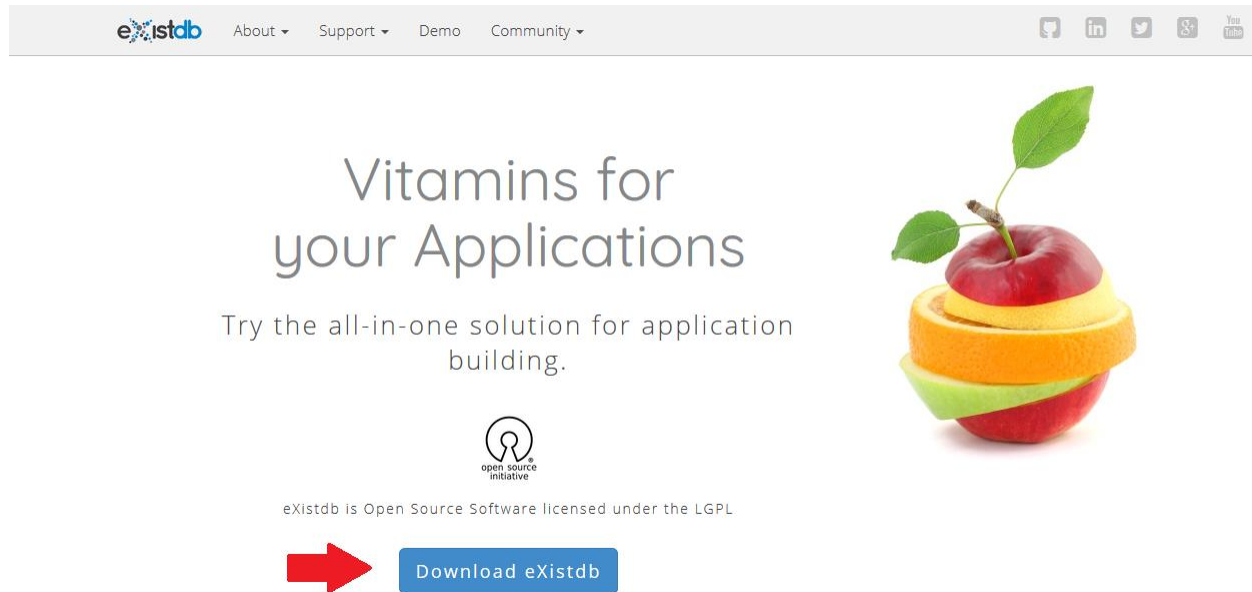


Ilustración 2: Agencia Tributaria

En este trabajo, se manejarán las herramientas para realizar BPMN como Camunda en versión online y REM para la gestión de requisitos.

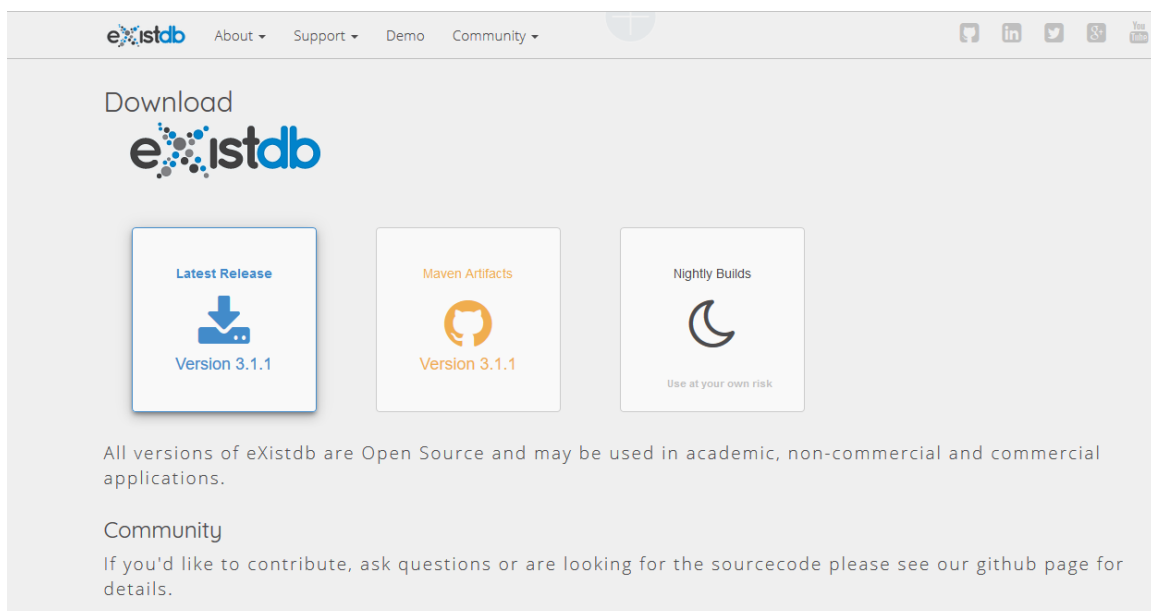
Para usar la herramienta eXistDB, previamente hay que descargarla de su página web oficial:

<http://exist-db.org/exist/apps/homepage/index.html>



*Ilustración 3: Página principal de eXistdb*

Al hacer clic en *Download eXistdb*, le llevará a la siguiente vista:



*Ilustración 4: Download eXistdb*

Haga clic en *Latest Release*, así obtendrá la última versión de esta base de datos, redirigiéndole a la siguiente página: <https://bintray.com/existdb/releases/exist/3.1.1/view> , donde se ha de descargar el archivo *.jar*.

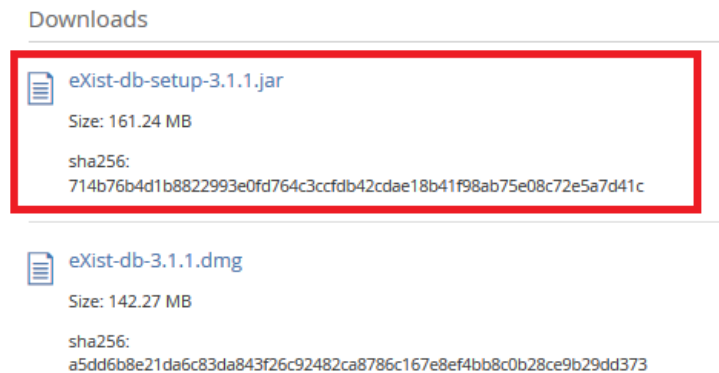


Ilustración 5: Archivo de instalación

Para su instalación y configuración correcta se ha de seguir los siguientes pasos, pero antes de continuar, debe asegurarse de poseer la última actualización de **Java**:

1. Hacer clic sobre el archivo *.jar* de descarga.
2. Hacer clic en descargar *Guardar archivo*.

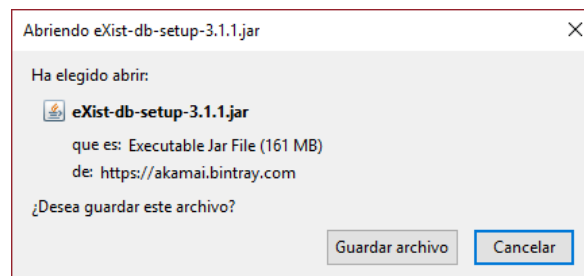


Ilustración 6: Archivo *.jar* de instalación

3. Localice la carpeta donde ha guardado el archivo descargado y ejecútelo. A continuación, se le requerirán permisos de Java en su dispositivo, autorícelo.
4. Haga clic en **Next**.



Ilustración 7: Instalacion eXistdb 1

5. Escoja la carpeta donde quiera almacenar la aplicación. [Nota: En Windows Vista y Windows 7, no instale eXist-db en el directorio "Program Files" a menos que haya desactivado UAC o ejecute siempre eXist-db en modo privilegiado.]

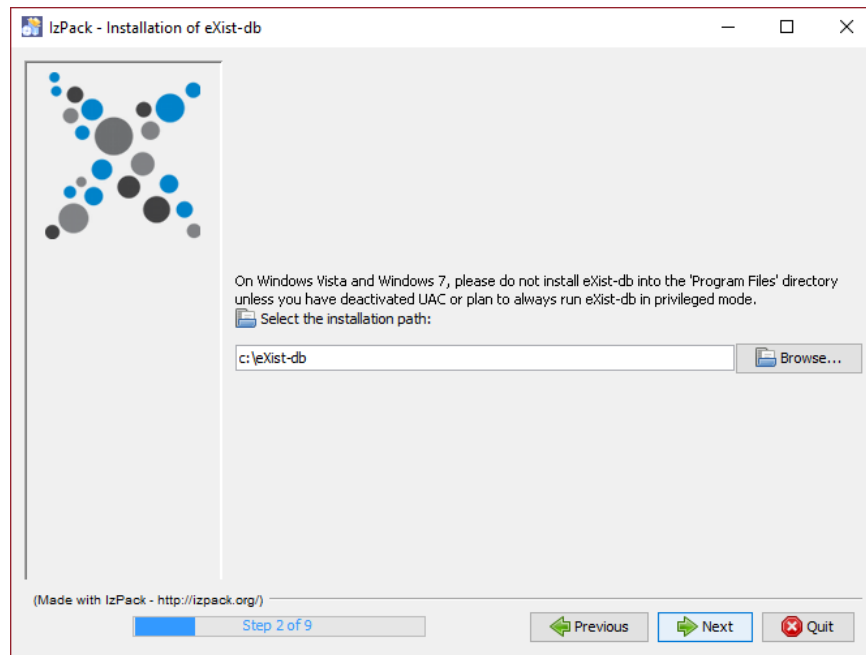


Ilustración 8: Instalación eXistdb 2

6. Haga clic en **Next**. A continuación, seleccione el directorio donde la herramienta guardará los archivos de datos, ha de estar fuera del directorio de "Program Files". [Nota: Asegúrese que eXist-db puede escribir en el directorio en el que se instalará]



7. Escoja la contraseña para su base de datos, por defecto el usuario será *admin*.

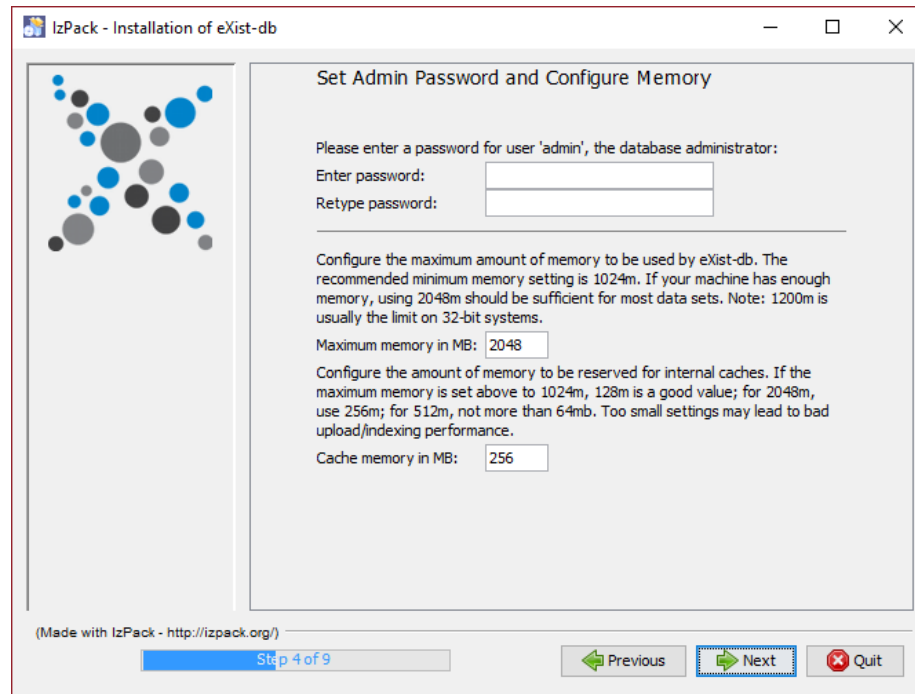


Ilustración 9: Instalación de eXistdb 3

8. Seleccione los paquetes a instalar. [Nota: Déjelos tal como vienen seleccionados por defecto]

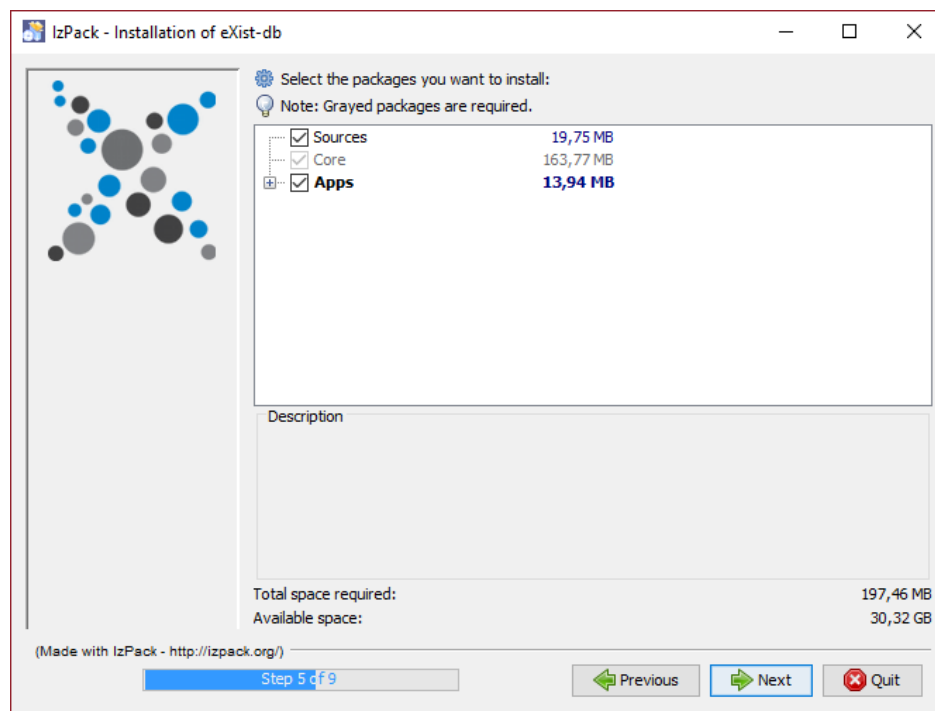


Ilustración 10: Instalación de eXistdb 4

9. Haga clic en **Next**, procederá a la instalación.

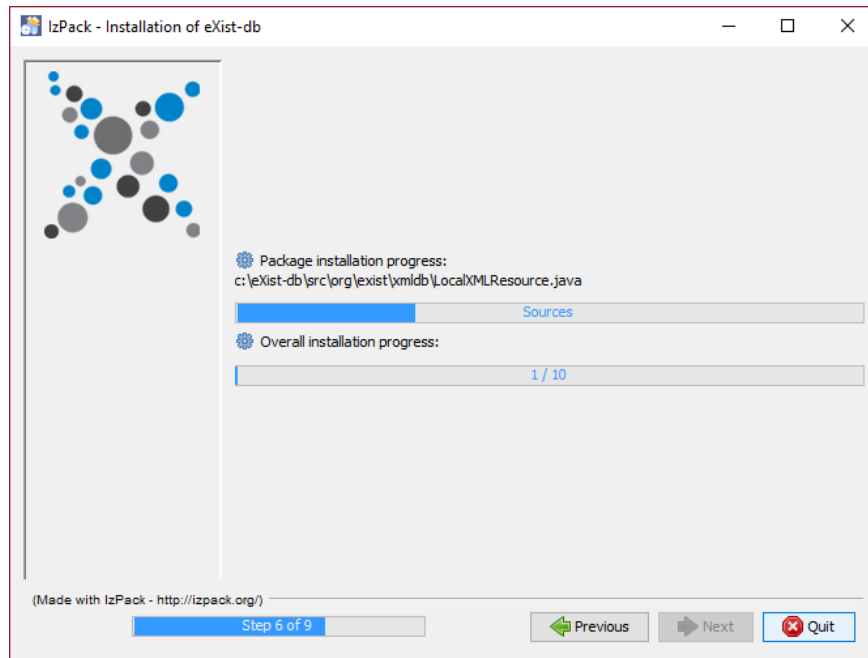


Ilustración 11: Instalación de eXistdb 5

10. Una vez instalado, continúe seleccionando **Next** para finalizar la instalación de la base de datos. Se le facilitará las opciones para poder crear un acceso directo a la aplicación tanto en el escritorio como en Inicio.

Una vez concluido el proceso de instalación, procederemos a ejecutarla para su uso, pero se nos requerirá permisos para Java y para usarla como un servicio, cuando lo hayamos permitido, en la barra de herramientas encontraremos minimizado el icono de eXistdb, pudiendo acceder al panel de administración de la aplicación, configuración del sistema, al cliente de Java, abrir el editor de la aplicación.

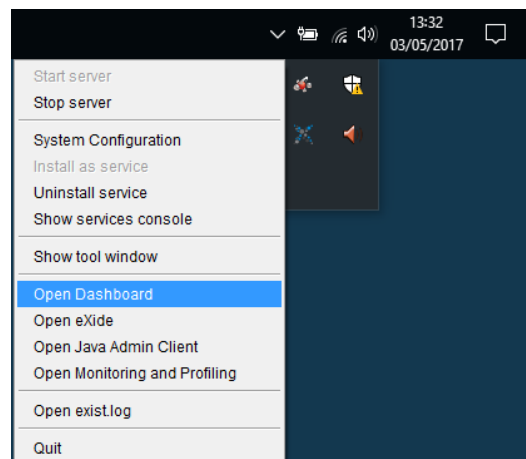


Ilustración 12: Barra de herramientas para el uso de eXistdb

Si abrimos el panel de administración, podremos instalar algunas herramientas, que nos facilitará documentación para poder usar funciones en Xquery o en otra tecnología de las permitidas por eXistdb, para ello, solo bastará con instalar los paquetes pertinentes desde el directorio de paquetes.



Ilustración 13: Panel de administración

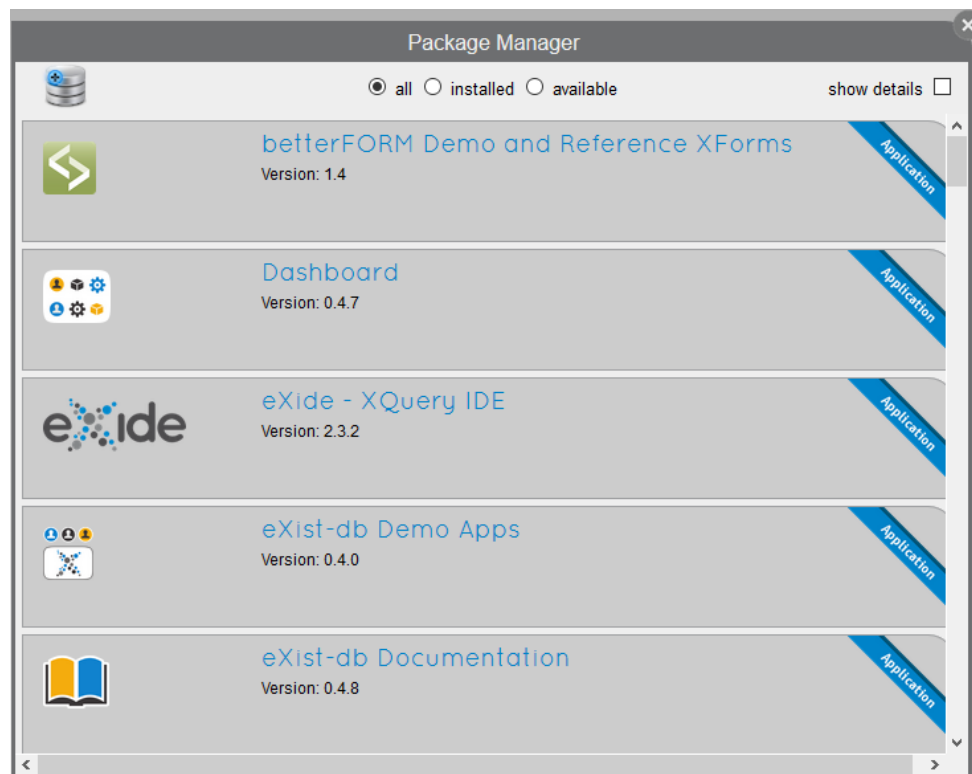


Ilustración 14: Directorio de paquetes de instalación

Para usar la herramienta de eXistdb desde el panel de navegación, sólo se ha de hacer clic en *eXide – Xquery IDE*.

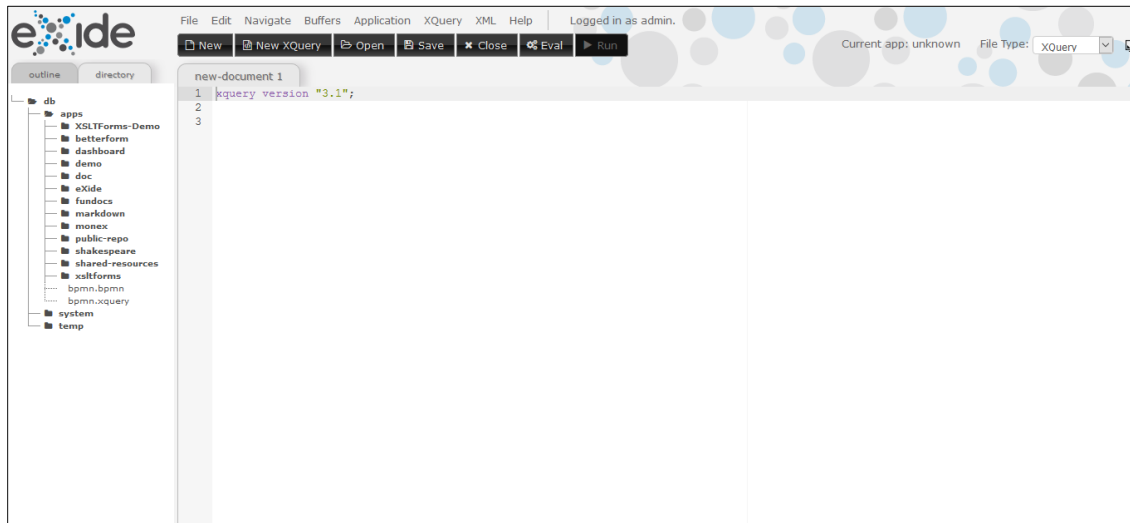
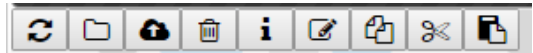


Ilustración 15: eXide-Xquery IDE

Desde la barra de herramientas, podemos acceder al contenido de nuestra base de datos, accediendo a **File > Manage**, se abrirá una pestaña donde podremos observar el contenido de la misma y usarla como

se desee, y mediante el uso de las herramientas disponibles:



se podrá crear nuevas carpetas, subir archivos, borrar, editar, actualizar la base de datos, etc...

También desde **File** o de la barra de herramientas de acceso rápido, se puede crear un nuevo documento

Xquery o de otro tipo:

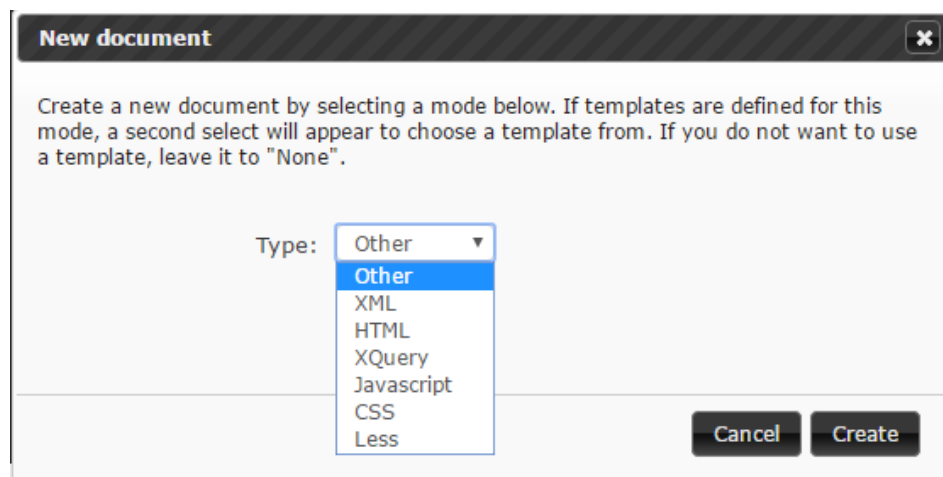


Ilustración 16: Nuevo documento eXistdb

Si queremos añadir algún *namespace* a nuestro documento, accediendo a **Navigate > Go to/import module** se nos facilitarán todos aquellos que podemos usar de forma cómoda y sencilla para su uso.

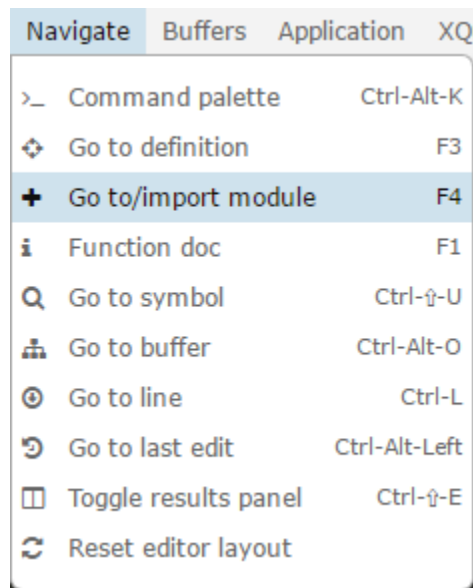


Ilustración 17: Navigate

Para crear las colecciones en nuestra base de datos, se seguirán los pasos indicados en [la solución propuesta](#).

Si vamos a utilizar las tecnologías REST, podremos dar uso a la pestaña Application. [Nota: Este apartado no se abordará, pues no es de relevancia para el desarrollo del contenido de consultas en base de datos XML que se va a realizar]

En cuanto a la herramienta para crear el bpmn, solo se ha de acceder a la página web de Camunda: <https://bpmn.io/> y acceder al editor de bpmn haciendo clic en *Try Online*.

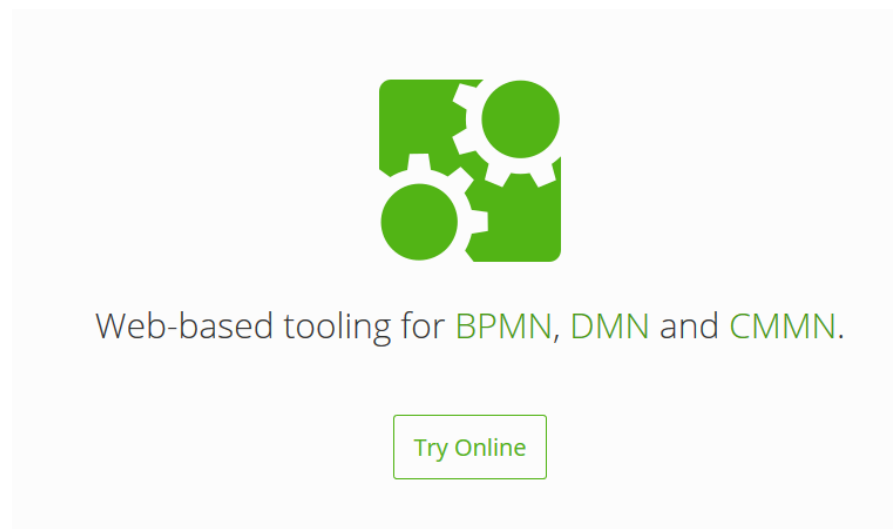


Ilustración 18: Camunda

Cuando hayamos creado nuestro bpmn, se guardará el archivo en nuestro ordenador, y lo abriremos como un archivo *.txt*, y bien se puede copiar su contenido y lo pegaremos en un nuevo documento en nuestra herramienta de eXistdb o bien importando el documento, en eXide-Xquery IDE, que guardaremos en nuestra base de datos para poder manejarlo posteriormente.

Para dar uso a la herramienta REM, bastará con instalarla desde: [Herramienta REM -LSI](#), para acceder a la descarga del programa, hará falta la contraseña proporcionada por algún miembro del departamento LSI de la facultad de Ingeniería Informática de Sevilla. Una vez descargada la herramienta e instalada, se podrá acceder a la aplicación y comenzar a usarla; cuando se haya finalizado su uso, se guardará el trabajo realizado, dicho trabajo se guardará en distintos formatos, uno de ellos es en XML, que será aquel que se usará para realizar las consultas en Xquery.

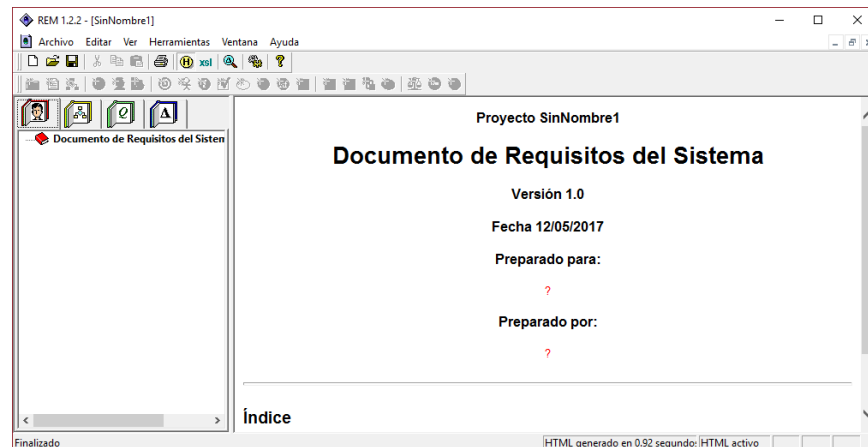


Ilustración 19: REM

## 4. OBJETIVOS A ALCANZAR

---

Con este trabajo, se pretende alcanzar los siguientes objetivos:

- Manejar el uso de la base de datos eXistdb para XML.
- Aprender y manejar de forma básica Xquery.
- Aplicar este tipo de bases de datos al uso cotidiano en herramientas para la ingeniería de software.
- Distinción entre las posibilidades que ofrecen unas bases de datos y otras, para saber cuál se adapta mejor a nuestras necesidades.

Una vez alcanzados, tendremos una noción suficiente como para poder escoger la base de datos y el tipo de lenguaje que mejor se adapte a las necesidades que se nos presenten.

## 5. SOLUCIÓN PROPUESTA

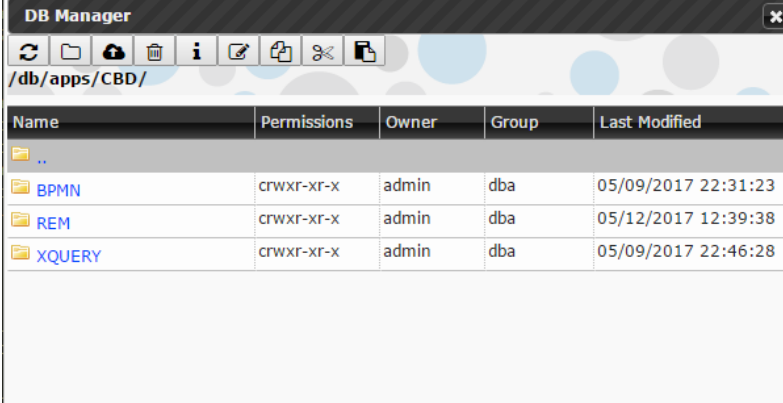
---

La solución propuesta es a partir del XML generado de nuestra herramienta REM y un modelado de proceso de negocio ficticio, realizar varias consultas en Xquery, viendo cómo cambian los resultados en función del código que se escriba.

Se aconseja una vez se haya instalado la herramienta eXistdb, realizar los siguientes pasos:

1. File > Manage
2. (Quizás le pida el usuario y la contraseña, hay que ingresarlo)
3. Clickear en “apps”

4. Create collection (icono de una carpeta) y nombrar como “CBD”
5. Clickear en la carpeta creada
6. Realizar el punto 4, tres veces, cada carpeta con el siguiente nombre “BPMN”, “REM” y “XQUERY”



Name	Permissions	Owner	Group	Last Modified
..				
BPMN	crwxr-xr-x	admin	dba	05/09/2017 22:31:23
REM	crwxr-xr-x	admin	dba	05/12/2017 12:39:38
XQUERY	crwxr-xr-x	admin	dba	05/09/2017 22:46:28

Ilustración 20: Carpetas a almacenar en apps/CBD

7. Introducir en la carpeta BPMN el archivo “bpmn.xml”, haciendo clic en “upload files” (icono de la nube) y seleccionándolo de la carpeta dónde se encuentre almacenado.
8. Introducir en la carpeta REM el archivo “Requisitos.xml”, haciendo clic en “upload files” (icono de la nube) y seleccionándolo de la carpeta dónde se encuentre almacenado.
9. Introducir en la carpeta XQUERY el archivo “bpmn.xquery” y “rem.xquery”, haciendo clic en “upload files” (icono de la nube) y seleccionándolos de la carpeta dónde se encuentre almacenado.

En cuanto al documento de Rem, como se puede apreciar en [la Imagen 21](#), esa es la herramienta con el contenido que se va a examinar a partir del XML generado.

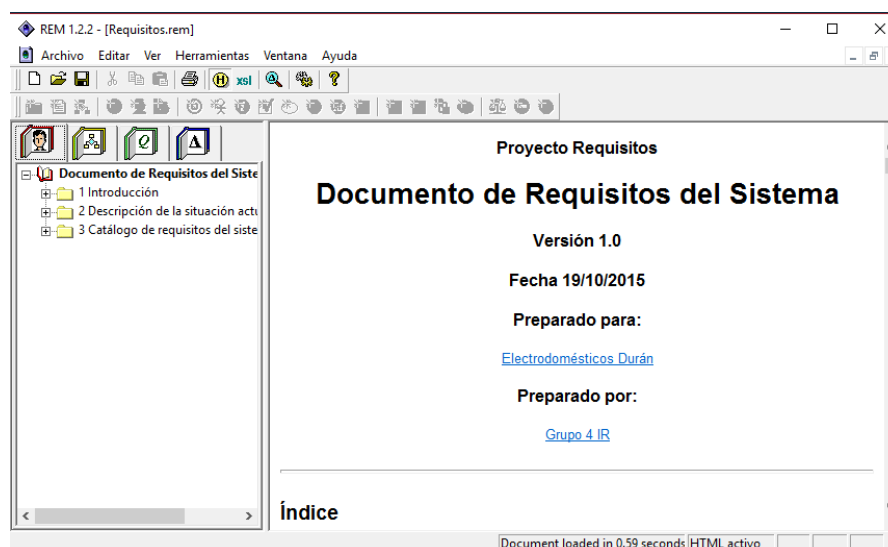


Ilustración 21: Herramienta REM

Se han realizado las distintas consultas siguiendo la estructura **FLOWR** [Nota: los comentarios en el código van entre los caracteres '(:' y ':)']:

```
xquery version "3.1";

(: Declaración de Los namespaces :)
declare namespace rem = "http://rem.lsi.us.es";
```

### 1. Visualizar el xml completo que vamos a usar, el documento

```
let $r := doc("/db/apps/CBD/REM/Requisitos.xml")/*
return $r
```

Resultado:

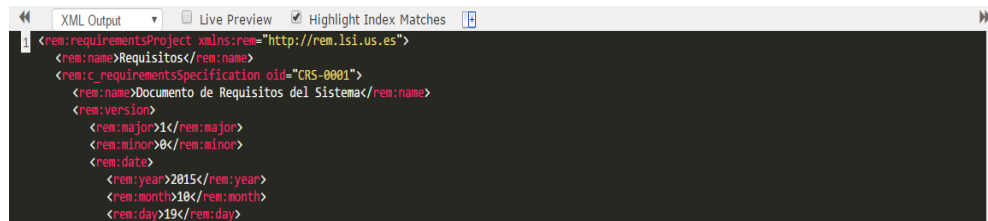


Ilustración 22: Visualizar documento XML completo a usar

### 2. Devuelve los nombres de los stakeholders

```
let $t := doc("/db/apps/CBD/REM/Requisitos.xml") (: se guarda en la variable el documento :)
let $p := $t//element(rem:stakeholder)//element(rem:name) (: Se guarda en la variable el nombre de Los stakeholders :)
let $r := distinct-values($p) (: Si hay algun elemento repetido, lo devuelve una sola vez :)
for $te in $r (: recorre los nombres :)
order by $te (: Ordena por nombre :)
return <name>{$te}</name>
```

Resultado:

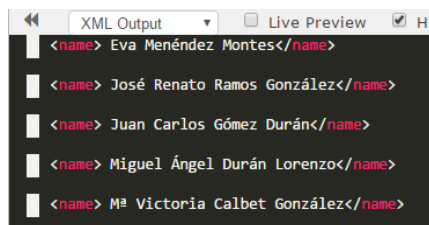


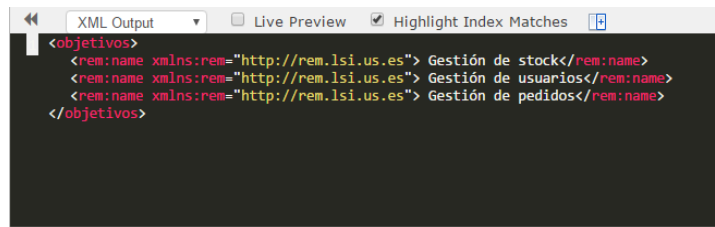
Ilustración 23: Nombres 'stakeholders'

### 3. Dar los nombres de los objetivos

```
for $t in doc("/db/apps/CBD/REM/Requisitos.xml")
let $e := $t//element(rem:section)[@oid="SEC-0025"]//element(rem:objective)//element(rem:name) (: Recorre el documento y busca la seccion que tenga un atributo SEC-0025 dando el nombre del objetivo :)
return
  <objetivos> {$e} </objetivos>
```



Resultado:



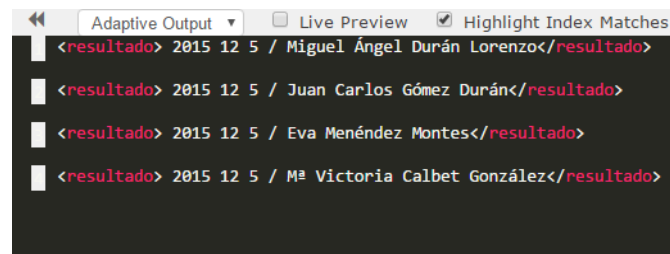
```
<objetivos>
  <rem:name xmlns:rem="http://rem.lsi.us.es"> Gestión de stock</rem:name>
  <rem:name xmlns:rem="http://rem.lsi.us.es"> Gestión de usuarios</rem:name>
  <rem:name xmlns:rem="http://rem.lsi.us.es"> Gestión de pedidos</rem:name>
</objetivos>
```

Ilustración 24: Nombres de los objetivos

4. Devuelve aquellos stakeholders que el día de su creación sea menor o igual a 5 y lo concatena con el nombre q le pertenece y la fecha

```
for $b in doc("/db/apps/CBD/REM/Requisitos.xml")//element(rem:stakeholder) (: recorre todos los stakeholders :)
where $b//element(rem:day)<=5 (: día menor o igual a 5 :)
return <resultado> {concat($b//element(rem:date),"/",
$b//element(rem:name))}</resultado> (: da el resultado de la fecha y el nombre, si comparamos con el resultado en la consulta de devolver todos los stakeholders, no estan todos :)
```

Resultado:



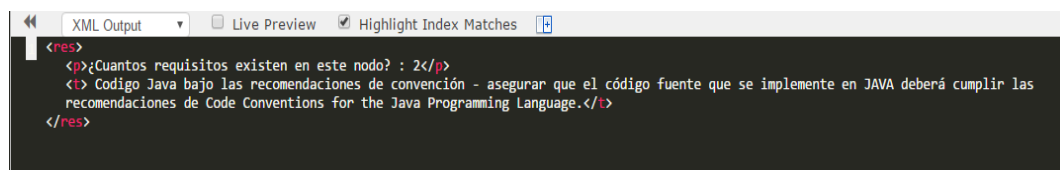
```
<resultado> 2015 12 5 / Miguel Ángel Durán Lorenzo</resultado>
<resultado> 2015 12 5 / Juan Carlos Gómez Durán</resultado>
<resultado> 2015 12 5 / Eva Menéndez Montes</resultado>
<resultado> 2015 12 5 / Mª Victoria Calbet González</resultado>
```

Ilustración 25: Devuelve los 'stakeholders' y su día de creación menor o igual a 5

5. Devuelve descripción y nombre de requisitos asociados a un id de una sección y cuántas son

```
for $b in doc("/db/apps/CBD/REM/Requisitos.xml")//element(rem:section)[@oid="SEC-0037"]
let $t := $b//element(rem:nonFunctionalRequirement)//element(rem:name)
let $req := $b//element(rem:nonFunctionalRequirement)//element(rem:description)
let $c :=count($t) (: Devuelve la cantidad que existen en esta consulta:)
return
  <res>
    <p>{ concat("¿Cuántos requisitos existen en este nodo? : " , $c) } </p>
    <t>{concat($t[1]," - ", $req[1])}</t> (: Accedemos a la posición 1 del elemento :)
  </res>
```

Resultado:



```
<res>
  <p>¿Cuántos requisitos existen en este nodo? : 2</p>
  <t> Código Java bajo las recomendaciones de convención - asegurar que el código fuente que se implemente en JAVA deberá cumplir las recomendaciones de Code Conventions for the Java Programming Language.</t>
</res>
```

Ilustración 26: Devuelve los nombres y descripciones de los requisitos asociados a un id en una sección y cuántas son

#### 6. Devuelve los target asociados a un source en una matriz

```
for $b in doc("/db/apps/CBD/REM/Requisitos.xml")//element(rem:trace)
let $e := $b/@source (: selecciona el atributo source :)
let $w := $b/@target (: selecciona el atributo target :)
where every $p in $e satisfies contains($p,"IRQ-0004") (: se debe satisfacer que
source sea igual IRQ-0004 :)
return <res>{$w}</res> (: devuelve los target que tengan como source IRQ-0004 :)
```

Resultado:

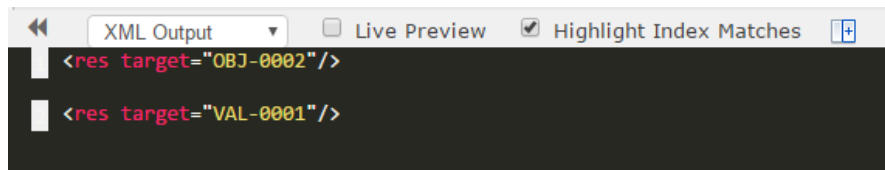


Ilustración 27: Devuelve los target asociados a un source en una matriz

#### 7. Devuelve en formato html, los requisitos funcionales, su oid y su nombre

```
<html>
  <head>
    <title> Requisitos funcionales </title>
  <body>
    <table>
      {
        for $b in
doc("/db/apps/CBD/REM/Requisitos.xml")//element(rem:section)[@oid="SEC-
0033"]//element(rem:functionalRequirement)
        return
          <tr>
            <td>
              <I> { string( concat($b/@oid , " - " , $b//element(rem:name)) ) } </I>
            </td>
          </tr>
        }
      </table>
    </body>
  </html>
```

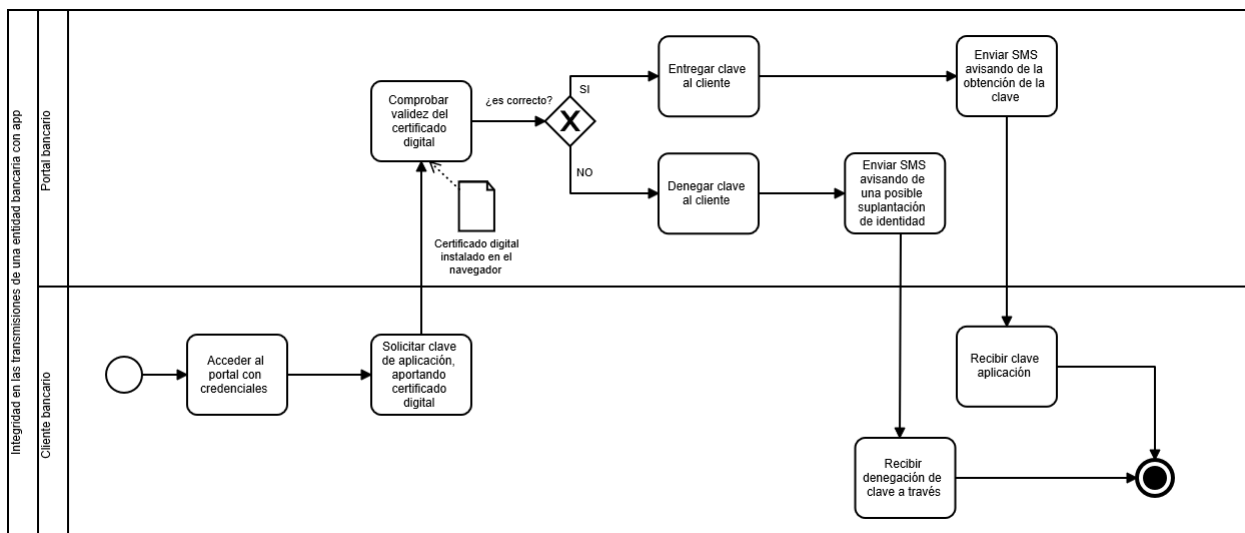
Resultado:

Direct Output   Live Preview   Highlight Index Matches

FRQ-0001 - Consultar catálogo  
FRQ-0002 - Buscar producto por campo  
FRQ-0003 - Añadir producto al catálogo de la compra  
FRQ-0004 - Eliminar producto  
FRQ-0005 - Modificar un producto  
FRQ-0006 - Modificar datos personales  
FRQ-0007 - Consultar 'Contacto' y 'Preguntas frecuentes'  
FRQ-0008 - Modificar 'Contacto' y 'Preguntas Frecuentes'  
FRQ-0009 - Consultar pedidos propios  
FRQ-0010 - Consultar todos los pedidos  
FRQ-0011 - Consultar pedidos pendientes  
FRQ-0012 - Consultar facturas  
FRQ-0013 - Consultar producto más vendido  
FRQ-0014 - Consultar cliente que realiza más compras  
FRQ-0015 - Modificar sliders  
FRQ-0016 - Consultar ventas realizadas.

Ilustración 28: Devuelve en formato html, los requisitos funcionales, su oid y su nombre

El BPMN creado indica un proceso en el cuál se ha de mantener la integridad en las transmisiones de una entidad bancaria desde una aplicación:



Seguiente la estructura **FLWOR**, se ha procedido a realizar las siguientes consultas [Nota: los comentarios en el código van entre los caracteres '{:' y ':}']:

```
xquery version "3.1";
```

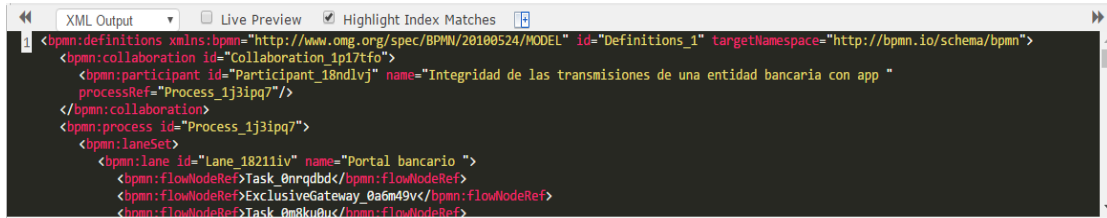
```
(: Declaración de Los namespaces :)
```

```
declare namespace xsi = "http://www.w3.org/2001/XMLSchema-instance";
declare namespace bpmn = "http://www.omg.org/spec/BPMN/20100524/MODEL";
declare namespace di = "http://www.omg.org/spec/DD/20100524/DI";
declare namespace bpmndi = "http://www.omg.org/spec/BPMN/20100524/DI";
declare namespace dc = "http://www.omg.org/spec/DD/20100524/DC";
declare namespace h = "http://www.w3.org/2005/xpath-functions";
```

### 1. Muestra el xml que se va a usar

```
Let $e := doc('/db/apps/CBD/BPMN/bpmn.xml')/*
return $e
```

Resultado:



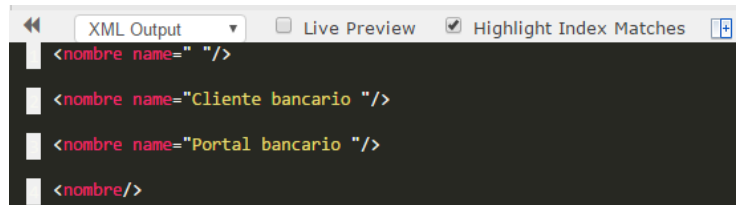
```
<bpmn:definitions xmlns:bpmn="http://www.omg.org/spec/BPMN/20100524/MODEL" id="Definitions_1" targetNamespace="http://bpmn.io/schema/bpmn">
  <bpmn:collaboration id="Collaboration_1p17tfo">
    <bpmn:participant id="Participant_18ndlvj" name="Integridad de las transmisiones de una entidad bancaria con app ">
      processRef="Process_1j3ipq7"/>
    </bpmn:participant>
  </bpmn:collaboration>
  <bpmn:process id="Process_1j3ipq7">
    <bpmn:laneSet>
      <bpmn:lane id="Lane_18211iv" name="Portal bancario ">
        <bpmn:flowNodeRef>Task_0nrqdbd</bpmn:flowNodeRef>
        <bpmn:flowNodeRef>ExclusiveGateway_0a6m49v</bpmn:flowNodeRef>
        <bpmn:flowNodeRef>Task_0m8ku0uc</bpmn:flowNodeRef>
      </bpmn:lane>
    </bpmn:laneSet>
  </bpmn:process>
</bpmn:definitions>
```

Ilustración 29: Muestra el documento XML completo que se va a usar

### 2. Nos devuelve los nombres de los lanes por orden alfabetico

```
for $k in doc("/db/apps/CBD/BPMN/bpmn.xml")//element(bpmn:Lane)
  Let $nombre := $k/@name
  order by $nombre
  return <nombre>{$nombre}</nombre>
```

Resultado:



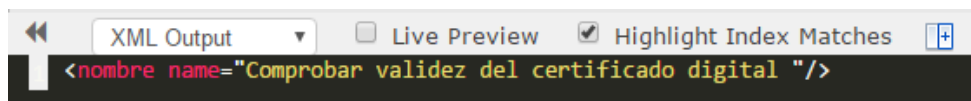
```
<nombre name=" " />
<nombre name="Cliente bancario " />
<nombre name="Portal bancario " />
<nombre />
```

Ilustración 30: Devuelve los nombres de los lanes en orden alfabético

### 3. Filtrar por tarea con un id específico

```
for $t in doc("/db/apps/CBD/BPMN/bpmn.xml")//element(bpmn:task)
  Let $nombre := $t/@name
  where $t/@id="Task_0nrqdbd"
  order by $nombre
  return
  <nombre>{$nombre}</nombre>
```

Resultado:



```
<nombre name="Comprobar validez del certificado digital " />
```

Ilustración 31: Filtra por tarea con un id específico

### 4. Devuelve aquellas aristas con una x e y especifica

```
for $b in doc("/db/apps/CBD/BPMN/bpmn.xml")//element(bpmndi:BPMNEdge)
  where some $a in $b satisfies (($a//element(di:waypoint)/@x < 960) and
  ($a//element(di:waypoint)/@y > 409)) (: Se puede ir cambiando el rango de x e y:)
  return data($b/@bpmnElement)
```

Resultado:

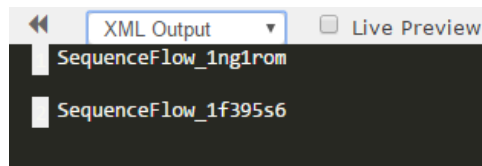


Ilustración 32: Devuelve las aristas con una x e y específica

##### 5. Devuelve aquellas tareas una x e y específica

```
for $b in doc("/db/apps/CBD/BPMN/bpmn.xml")//element(bpmndi:BPMNShape)
let $f := doc("/db/apps/CBD/BPMN/bpmn.xml")//element(bpmn:task)
where some $a in $b satisfies (($a//element(dc:Bounds)/@x < 410) and
($a//element(dc:Bounds)/@y > 100)) (: se puede ir cambiando el rango de x e y:)
let $e := data($b/@bpmnElement) (: devuelve el atributo bpmnElement:)
where starts-with($e,"Task_") (: si en $e algun elemento guardado que empiece con
"task_":)
return $e (: devuelve aquellos que dependen de la comparacion:)
```

Resultado:

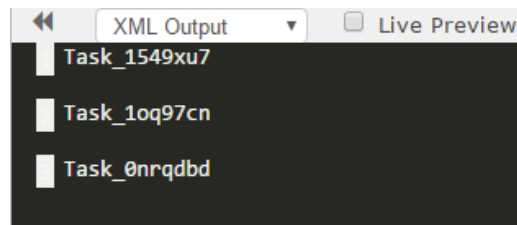


Ilustración 33: Devuelve las tareas con una x e y específica

##### 6. Devuelve las tareas

```
for $i in distinct-values(doc("/db/apps/CBD/BPMN/bpmn.xml")//(element(bpmn:process)
union element(bpmn:task)) /@name) (: unimos los nodos del proceso con las tareas y
filtramos por nombre:)
order by $i (: ordenamos por nombre :)
return data($i)
```

Resultado:

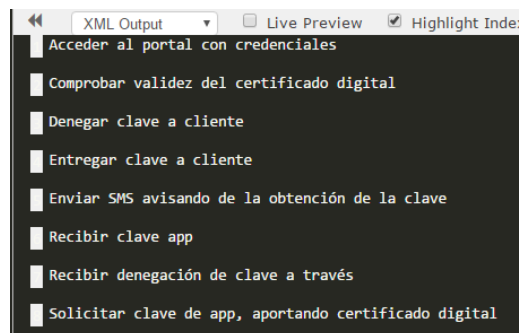


Ilustración 34: Devuelve las tareas ordenadas por nombre

#### 7. Devuelve la tarea asociada a ese id

```
for $b in doc("/db/apps/CBD/BPMN/bpmn.xml")//element(bpmn:task)
where every $a in $b satisfies ($a/@id="Task_0nrqdbd")
return data($b/@name)
```

Resultado:

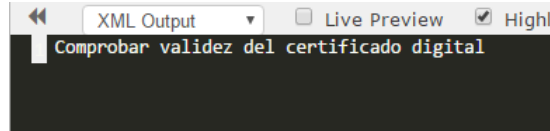


Ilustración 35: Devuelve una tarea asociada a un id

[Nota: El código para ejecutarse, tal como está realizado, se ha de ejecutar por partes, es decir, lo que no quiera ejecutarse se ha de comentar. Existen 2 formas de poder visualizar lo que devuelven las consultas, haciendo clic en "Eval" (se visualiza por consola, se puede escoger cómo ver la salida) o en "Run" (se abre una nueva ventana mostrando el documento en XML), si se escoge la opción de Run, indicar que dependiendo del navegador no se puede visualizar, se aconseja usar para ello Google Chrome.]

## 6. BIBLIOGRAFÍA

---

- eXistdb: <http://exist-db.org/exist/apps/homepage/index.html>
- Documentación eXistdb: <http://exist-db.org/exist/apps/doc/documentation.xml>
- Camunda: <https://bpmn.io/>
- REM: [https://www.lsi.us.es/descargas/descarga\\_programas.php?id=3](https://www.lsi.us.es/descargas/descarga_programas.php?id=3)
- Xquery-Sequences: <https://en.wikibooks.org/wiki/XQuery/Sequences>
- Xquery-Expressions: [https://en.wikibooks.org/wiki/XQuery/Quantified\\_Expressions](https://en.wikibooks.org/wiki/XQuery/Quantified_Expressions)
- Xquery-LSI: <https://www.lsi.us.es/docs/informes/LSI-2005-02.pdf>
- Xquery-Examples: [https://www.w3schools.com/xml/xquery\\_example.asp](https://www.w3schools.com/xml/xquery_example.asp)