Accede al documento original de Wuolah

TEMA-14-ACCESO-A-DATOS.pdf

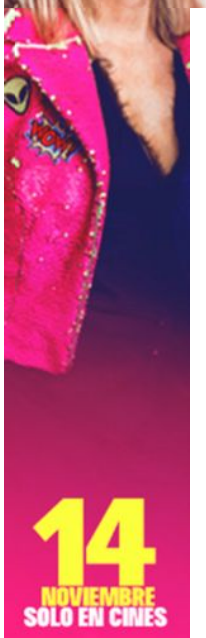
user_4383848

Acceso a datos

1º Desarrollo de Aplicaciones Multiplataforma

Estudios España

Reservados todos los derechos.
No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.



14 NOVIEMBRE SOLO EN CINES
TODOS LOS LADOS DE LA CAMA

TEMA 14 - ACCESO A DATOS
GESTIÓN DE BASES DE DATOS NATIVAS XML

1. BASES DE DATOS XML:

Son bases de datos que almacenan documentos y datos XML de una forma muy eficiente. Permiten que los datos se almacenen, consulten, combinen, aseguren, indexen, etc.
Las bases de datos XML nativas **no se basan en tablas**, sino en los llamados contenedores. Cada contenedor puede almacenar grandes cantidades de documentos XML o datos, que tienen alguna relación entre ellos.
Los contenedores **también pueden tener subcontenedores**. La gran diferencia con las bases de datos relacionales es que la estructura de los datos XML en un contenedor no tiene por qué ser fija. Podremos almacenar distintas unidades de negocio sin mucha relación dentro del mismo contenedor, la infraestructura lo permite, y podemos hacerlo.
Las **bases de datos XML nativas** no son consultadas por sentencias SQL, son consultadas por expresiones Xpath, establecido por el "**W3C**" para navegar a través de documentos XML. Es un lenguaje que se puede utilizar para consultar datos de documentos XML.
Este lenguaje se basa en una representación de árbol del documento XML, y selecciona nodos según diferentes criterios. Cuando se consulta una base de datos XML nativa, el usuario generalmente abre un contenedor y posteriormente, envía dicha expresión Xpath contra todos los documentos XML en la base de datos.

2. VENTAJAS Y DESVENTAJAS DE LAS BASES DE DATOS XML CON RESPECTO A LAS RELACIONALES:

a. Ventajas:
• Las bases de datos XML nativas son capaces de almacenar, mantener y consultar mayores cantidades de documentos XML en comparación a las relacionales.
• A diferencia de las relacionales, no es necesario configurar tablas, y por tanto, no se necesita realizar diseños complicados antes de configurar la base de datos. Una tabla de base de datos clásica tiene la desventaja de ser sólo bidimensional, por lo que la estructura "más profunda" debe implementarse utilizando claves secundarias, lo que puede hacer que el diseño de una base de datos sea bastante complicado.
• La facilidad de importación o exportación hacia/desde otras aplicaciones con otros formatos.

b. Desventajas:
• Las bases de datos relacionales están muy bien establecidas, es tecnología ya probada. Las bases de datos XML son un fenómeno algo más reciente ya que algunas de las primeras bases de datos XML no tienen mucho más de una década de antigüedad, por lo que la experiencia aún es limitada.
• Xpath no es un lenguaje excesivamente fácil de aprender, mientras que SQL está muy extendido. SQL está más relacionado con el lenguaje y la forma humana de 'pedir' cualquier cosa en general, por lo que es algo más sencillo de aprender.
• No todas las aplicaciones gestoras de datos soportan dicho lenguaje, tendríamos que buscar cuales son los compatibles con las bases de datos mencionadas.

WUOLAH

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.

3. GESTORES XML LIBRE Y COMERCIALES:

a.Sistemas gestores de bases de datos XML nativas de pago:
• **Excelon XIS:** Con este gestor, las empresas pueden aprovechar de manera completa y rentable la extensibilidad y flexibilidad de XML para crear, auditar y cambiar continuamente aplicaciones que almacenan y manipulan cantidades limitadas de datos XML.
• **GoXML DB:** Es una base de datos XML nativa con un motor de consultas de alto rendimiento. Los documentos XML se almacenan directamente, lo que elimina la necesidad de descomponer datos o configurar cómo se almacenarán los datos.
• **Infonyte DB:** Se distingue por el soporte nativo para XML, la independencia de la plataforma y el uso moderado de los recursos del sistema. Se adapta perfectamente a los requisitos tanto de arquitecturas de componentes como de dispositivos pequeños.

b. Algunas de las opciones que tenemos libres o de código abierto son las siguientes:
• **ExistDB:** Es un proyecto de software de código abierto para bases de datos NoSQL construido sobre tecnología XML. Está clasificado como un sistema de base de datos orientado a documentos NoSQL y una base de datos XML nativa. También proporciona soporte para documentos XML, JSON, HTML y binarios. A diferencia de la mayoría de los sistemas de administración de bases de datos relacionales, proporciona Xquery y XSLT como lenguajes de programación de aplicaciones y consultas.
• **X-Hibe DB:** Es una poderosa base de datos XML nativa diseñada para desarrolladores de software que requieren funciones avanzadas de procesamiento y almacenamiento de datos XML dentro de sus aplicaciones.
• **Tamino DB:** Es una base de datos XML nativa. Si la comparamos con una base de datos relacional, tiene la desventaja de no ser muy popular.

4. INSTALACIÓN Y CONFIGURACIÓN DE EXISTDB:

Deberemos de tener en cuenta unos requisitos de sistema recomendados.
• Si instalamos una versión final superior a la versión 3.0 tendremos que tener instalado previamente el **JDK Java 8.**
• Debemos asignar **128 MB** de memoria caché.
• Debemos de tener **1024 MB de memoria RAM** disponibles para poderle asignar al aplicativo.
Una vez descargado el archivo .jar lo instalamos. Finalmente, en este punto, es necesario configurar y establecer la cantidad de memoria RAM que queremos darle a nuestro proceso Java y a nuestra memoria caché.

4.1. EL PAQUETE DE INSTALACIÓN:

Hay distintos paquetes que podemos agregar a la instalación.
• **El paquete “core”** Es uno de los elementos que nos permitirá ‘correr’ la base de datos.
• **El paquete “sources”** es opcional deseleccionaríamos dicho paquete si tuviéramos problemas de espacio
• **El paquete “app”** nos permite seleccionar o deseleccionar una serie de apps.

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.

WUOLAH

Elimina la publicidad de este documento con 1 coin

5. ESTRATEGIAS DE ALMACENAMIENTO:

Las bases de datos nativas XML pueden ser clasificadas dependiendo el tipo de almacenamiento:

A) Almacenamiento basado en texto: Esta modalidad consiste en almacenar el documento completo en base de datos, y dotar a la misma, de algún tipo de funcionalidad para que se pueda acceder fácilmente a él. En este tipo de almacenamiento suele realizarse la compresión de ficheros. También añadir algunos índices específicos para aumentar la eficiencia. Las posibilidades para esto serían dos:
• Almacenar el fichero binario de tipo BLOB en un sistema relacional.
• Guardar dicho fichero en un soporte o almacén orientado a dicha operación con índices, soporte de transacciones etc.
B) Almacenamiento basado en el modelo: Para este caso, se utilizaría un modelo de datos lógico como por ejemplo puede ser DOM, para definir la estructura y la jerarquía de los documentos XML que se vayan a almacenar. Se guardaría el modelo del documento en un almacén definido previamente.
Para esto, tenemos algunas posibilidades como:
• Traducir desde DOM a tablas de una base de datos convencional relacional.
• Traducir el objeto DOM a objetos en una base de datos orientada a objetos.
• Usar un almacén de datos creado específicamente para esta utilidad.
C) Almacenar una forma modificada del documento en local: Podremos usarlo, cuando la cantidad de ficheros a almacenar no sea muy elevada, y no se realicen numerosas actualizaciones ni transferencias de los mismos. Realmente consiste en almacenar una forma modificada del fichero XML base completo, directamente en sistema de archivos. Evidentemente tiene diferentes limitaciones como escalabilidad, flexibilidad, recuperación y seguridad.

6. ESTABLECIMIENTO Y CIERRE DE CONEXIONES:

• **Org.xmldb.api:** Nos enriquecerá el código con las Interfaces y DatabaseManager
• **Org.xmldb.api.base:** Con esta librería accederemos a las colecciones, los objetos Database, Resource, ResourceIterator, ResourceSet y algunos más.
• **Org.xmldb.api.modules:** Accederemos a los servicios de transacciones, XMLResource, servicios de XPathQueryService y otros varios relacionados.

• **Indexación:** Permiten la creación de índices para acelerar las consultas frecuentes.
• **Identificador único:** Cada documento XML está asociado con un identificador único, a través del cual se puede identificar en el repositorio.

7. AGREGAR, MODIFICAR Y ELIMINAR RECURSOS:

También es posible agregar nuevos recursos XML y no XML a la colección (objeto de “Collection”). Para esto, se necesitan las siguientes clases y métodos:
La clase Collection representa una colección de recursos (resource) almacenados en una base de datos XML. El método más relevante para agregar nuevos recursos en esta clase es:

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.

WUOLAH

Elimina la publicidad de este documento con 1 coin

• **storeResource(resource res)**: Almacena el recurso res proporcionado por el parámetro en la colección.
• **removeResource(recurso res)**: Elimina el recurso res pasado al recurso a través de parámetros de la colección.
• El otro método interesante de la colección (útil para crear y eliminar nuevos recursos) es **listResources()**, que devuelve una matriz de cadenas que contiene los ds de todos los recursos que tiene la colección; **getResourceCount()** obtiene la cantidad de recursos almacenados en la colección; y **createResource** (java.lang.String id, java.lang. String type), crea un nuevo recurso vacío en la colección, cuyo ID y tipo son pasados por parámetros.
Si ya se comprende el proceso de creación de un nuevo recurso, se puede definir el proceso de eliminación más fácilmente. La forma principal de eliminar recursos es: **removeResource(resource res)**, que elimina resource res de la colección

8. MODIFICACIÓN DE CONTENIDOS XML:

A) Modificación del valor del texto asociado a una etiqueta → "**setTextContent**"
B) Modificación del valor de un atributo asociado a una etiqueta → **"setAttribute(String)**"

9. TRANSACCIONES Y EXCEPCIONES:

A) Transacciones XML
En algunos de los ejemplos expuestos en el tema, existe el concepto de transacción: un **conjunto de declaraciones ejecutadas que forman inseparablemente una unidad de trabajo, de modo que todas se ejecutan o no se ejecutan**. Al administrar las transacciones, el administrador XML proporciona acceso simultáneo a los datos almacenados, mantiene la integridad y seguridad de los datos, y proporciona un mecanismo de recuperación de fallos en la base de datos.
Exist-db admite transacciones compatibles con ACID:
• **Atomicidad.** Se deben completar todas las operaciones de la transacción, o no realizar ninguna operación, no se puede dejar a la mitad.
• **Consistencia.** La transacción finaliza sólo cuando la base de datos permanece en un estado coherente.
• **Aislamiento.** Las transacciones sobre una misma información deben ser independientes para que no interfieran con sus operaciones, y no generen ningún tipo de error.
• **Durabilidad.** Al final de la transacción, el resultado de la misma se conservará y no se podrá deshacer incluso si el sistema falla.

B) Tratamiento excepciones
La clase **XMLDBException captura todos los errores que ocurren cuando se usa XML** (DB para procesar la base de datos). Esta excepción contendría dos códigos de error:
• Un código de error especificado por cada sistema de procesamiento XML local (fabricante-proveedor).
• Un código de error definido en la clase ErrorCodes. Si el error que ocurrió en un momento dado es parte del sistema de administración XML, ErrorCode debe tener un valor de errorCodes, VENDOR_ ERROR. La clase ErrorCodes define los diferentes códigos de error XML: DB utilizado por el atributo errorCodes de la excepción **XMLDBException.**

WUOLAH

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.