7. Poblar base de datos a partir de documentos XML validados con esquemas XML

En este capítulo se aborda el tema de la población de la base de datos que se creó con el constructor automático de bases de datos. Presentado el algoritmo que identifica los diferentes tipos de etiquetas que pueden aparecer en un documento XML, cómo se crean las instrucciones SQL y finalmente cómo se puebla la base de datos creada con anterioridad.

Población de la base de datos

Una vez que se tenía definida la forma en que funcionaría el constructor automático de la base de datos, el siguiente paso fue crear el alimentador de dicha base, utilizando para esto documentos XML válidos.

Se buscaron diferentes alternativas, buscando la mejor manera en la que ambas partes –constructor y alimentador de la base de datos- pudieran interactuar. Obteniéndose como resultado el algoritmo de la Figura 7.1, que muestra el funcionamiento en forma general del alimentador de la base de datos, con documentos XML.

Algoritmo para analizar los documentos XML

1. Recibir datos de entrada (path de archivos, nodo principal, nombre BD)
2. Acondicionar información, quitando espacios en blanco, comentarios, etc.,

almacenando la información en *Lista1*

3. Busca elemento de referencia (el mismo que sirvió para crear la BD), en caso de encontrarlos ir a 4; si no, presentar mensaje “no se encontró elemento”
4. En una sola cadena de caracteres obtener la información desde el elemento buscado hasta su etiqueta de finalización

   Ejemplo_Cadena: <elemento><uno>dos</uno></elemento>
5. Una vez obtenida la cadena, revisar etiqueta por etiqueta contenida en la cadena; pueden presentarse las siguientes opciones o tipos de etiquetas.

   a) Si la etiqueta contiene atributos dentro (tipo1)

      <etiqueta atributos=” algo ” … ><otra_etiqueta> …

   b) La etiqueta contiene atributos y tiene autocierre (tipo2)

      <etiqueta atributos=” algo” …/><otra_etiqueta> …

   c) No contiene atributos y después de la etiqueta no hay otra etiqueta (contiene texto) (tipo3)

      <etiqueta atributos=”” …>información …

   d) Después de la etiqueta sigue otra etiqueta (tipo4)

      <etiqueta><otra_etiqueta> …

   e) La etiqueta contiene atributos mas información (tipo5)

      <etiqueta atributos=” ” ..>información <…
6. De acuerdo al tipo de etiqueta, la información se extrae de diferente manera, y por lo tanto se crean diferentes comandos SQL, que alimentan la base de datos, almacenándose en una lista llamada *Lista2*

7. Por ultimo, se revisa si existen varios `INSERT`, para una sola tabla; si sí, ir a 8 , y si no, dejar intacta *Lista2*
8. Modificar *Lista2,* agrupando `INSERT`s, que contengan el mismo nombre de tabla
9. Posteriormente se compara que los resultados obtenidos, producto de recorrer el documento XML, sean congruentes con la base de datos. Haciendo los ajustes mínimos, para que se pueda introducir la información correctamente, ayudándose de las instrucciones que generaron la base de datos contenida, el archivo que creó la base de datos.
10. Crear los comandos SQL, para alimentar la base de datos

Figura 7.1. Algoritmo para analizar los documentos XML

Lo primero que debe hacer al alimentador es recibir los datos de entrada: *path_archivo* y *elemento_raiz o nodo*. El path o ruta indica donde se encuentran el o los archivos XML que van a alimentar la base datos. Y el elemento raíz o nodo, es el mismo que se usó para la creación de la base de datos.

El poblador o alimentador puede tener dos alternativas dependiendo de la información que se ingrese en *path_archivo*, ya que puede leer varios archivos en un proceso automático o solamente un archivo XML. Para que el alimentador realice la primera alternativa se debe teclear *"**`*.xml`**",* lo que da como resultado la búsqueda de todos los archivos dentro del directorio del path con extensión XML.

Todos los nombres de los archivos XML se almacenan en una *lista2*. Si se quiere que el alimentador realice la segunda opción, entonces se ingresa el nombre del *"documento.xml"* en donde *documento* es el nombre de un archivo con extensión XML, lo que almacena sólo un elemento a la *lista2*. En caso de encontrarse el o los archivos XML, el siguiente paso es leer la información contenida dentro de dicho archivo. A continuación se explica con detalle.

La lectura del archivo XML que posteriormente poblará la base de datos, comienza preparando la información para que pueda ser manejada correctamente, quitando tabulaciones, espacios en blanco innecesarios, comentarios, y todo aquello que puede provocar un error en la correcta lectura del documento XML. El poblador barre el documento buscando el elemento raíz especificado en la línea de comando. El contenido de este elemento (etiqueta: <*elemento_raíz …* >), que puede ser una estructura compleja se concatena en una sola cadena de caracteres; termina de agregar etiquetas a la cadena al encontrar la etiqueta de cierre del elemento raíz (… <*/elemento_raiz*>). Una vez que todo está unido en una sola cadena de caracteres, el alimentador busca el inicio y final de cada etiqueta, contenida dentro de la cadena, y dependiendo de la información que contenga cada etiqueta y los caracteres que sigan después del fin de la etiqueta (puede ser el inicio de otra '<' o caracteres diferentes a este). El alimentador se comporta de acuerdo al algoritmo (Figura 7.1), dando como resultado la creación de los comandos SQL correspondientes para poblar la base de datos, almacenándose en una lista llamada *ListaSQL*.

Por último, el alimentador hace una revisión dentro de las instrucciones SQL generadas para verificar que no existan comandos *INSERT INTO* con el mismo nombre de la tabla (*INSERT INTO tabla1(…), INSERT INTO tabla1(…)*). Esto es importante porqué nos evita que se añada la información a la base de datos de manera incorrecta y la información que debería estar en una sola fila, se agregue en diferentes filas de la tabla, provocando con esto errores en el llenado de las tablas de la base de datos.

Para solucionar este problema, en caso de encontrar varios comandos *INSERT INTO tabla (…)* con el mismo nombre de tabla, el alimentador las une. A continuación se muestra un ejemplo donde al principio aparecen varios inserciones a una tabla de la base de datos y después de hacer la unión de los comandos SQL, queda en una sola instrucción.

Ejemplo

sin unir comandos SQL

```
INSERT INTO encabezado(id,permisos)
VALUES ('0','todos');
INSERT INTO encabezado(titulo)
VALUES ('Acta de independencia del Imperio Mexicano');
INSERT INTO encabezado(institucion)
VALUES ('Junta Soberana del Imperio Mexicano');
INSERT INTO encabezado(ciudadPublicacion)
VALUES ('ciudad de México');
INSERT INTO encabezado(fechaPublicacion)
VALUES ('1821');
```

después de unir comandos SQL

```
INSERT INTO encabezado(id, permisos, titulo, institucion,
ciudadPublicacion, fechaPublicacion)
VALUES('0','todos','Acta de independencia del Imperio Mexicano','Junta
Soberana del Imperio Mexicano','ciudad de México','1821');
```

Por último, el alimentador de la base de datos ejecuta las instrucciones para poblar la base de datos. En este contexto debo señalar lo siguiente: la única diferencia entre leer uno o varios archivos es que en la primera sólo hay una iteración (porqué *lista2* contiene sólo un elemento*)* y para más de un archivo las iteraciones son igual al número de archivos almacenados en la *lista2* que contiene los nombres de todos los archivos XML que alimentarán la base de datos. La iteración comienza en la sección donde se acondiciona (quitar tabulaciones, etc., y hacer la cadena de caracteres) la información de los documentos XML y termina en la revisión de duplicidad de instrucciones.

Hay que mencionar que en el caso de que se quiera leer varios archivos *(*.xml)*, el *elemento_raíz,* o *nodo*, es el mismo para todos los archivos que van a poblar la base de datos y si no se encontraran los archivos XML aparecería un mensaje en donde se indica que no se encontró el archivo XML. Y por último, en caso de omitirse la extensión '.XML', al momento de escribir el nombre del o los archivos, el alimentador de la base de datos muestra un mensaje que indica "sólo se usan archivos XML para alimentar la base de datos".

Así concluye la explicación del funcionamiento de alimentador de la base de datos. El siguiente capítulo trata de las conclusiones finales, donde se realiza una síntesis de los capítulos anteriores, se abunda sobre las ventajas, desventajas de las decisiones tomadas y se mencionan otras opciones que pudieran utilizarse para consultar documentos XML.