Gestión de datos complejos mediante Bases de Datos Relacionales de Objetos

Eduardo Mora Monte
Director de la
Escuela Técnica Superior de Ingenieros
Industriales y de Telecomunicación
Universidad de Cantabria

Para las empresas del sector de las telecomunicaciones, los contenidos implicados pueden llegar a tener incluso mayor interés que la propia operación. Por ello, es muy importante disponer de los sistemas más eficientes en almacenamiento y recuperación de información.

En la actualidad, los Sistemas Administradores de Bases de Datos Relacionales (RDBMS) siguen siendo el soporte de la mayoría de los centros de datos. Estos sistemas admiten sólo ciertos tipos de datos simples y un número restringido de técnicas de indexación. A excepción de los códigos binarios, podría decirse que todos los datos procesados por un ordenador son datos complejos. No obstante, se considera que los caracteres y los números también son tipos de datos simples, ya que son reconocidos por la mayoría de los RDBMS y casi todos los lenguajes de programación y pueden manejarlos.

Al ir ampliándose el campo de aplicación, cada vez es más necesario utilizar otros tipos de datos más complejos, como documentos, imágenes, datos multimedia, series temporales, datos geográficos, etc. que requieren codificaciones binarias especiales para su representación. La representación de los datos es sólo una parte del problema pues los métodos para su tratamiento también son específicos para cada tipo de dato. Actualmente, la mejor representación para estos datos complejos es la de objetos que contienen los detalles de los datos, las estructuras y los métodos referentes a su tratamiento y a la interacción de unos objetos con otros.

Ante este estado de cosas, algunas instituciones se han encaminado hacia sistemas de bases de datos no relacionales, como los Sistemas Administradores de Bases de Datos Orientados a Objetos (OODBMS), para dar soporte a este tipo de datos. Sin embargo, los sistemas no relacionales suelen ser más costosos de desarrollar y mantener que los que siguen el modelo relacional y, además, la ausencia de un lenguaje estándar de definición y manipulación de datos, como el SQL, aumenta mucho la complejidad en el desarrollo de aplicaciones y en la integración con otros sistemas.

Estos son algunos de los motivos por los que la tendencia actual, para gestionar datos complejos, consiste en ampliar las prestaciones de los RDBMS, sin sacrificar la funcionalidad, fiabilidad y escalabilidad que les son propias. Dicho de otro modo, los RDBMS tienden a adoptar ciertas características de los (OODBMS) para convertirse en los denominados Sistemas Administradores de Bases de Datos Relacionales de Objetos (ORDBMS).

Para gestionar eficientemente diferentes tipos de objetos, los RDBMS, fundamentalmente, deben incorporar las tres capacidades siguientes:

- Técnicas de almacenamiento y de creación de índices adecuadas a cada estructura de datos. Por ejemplo, las técnicas de almacenamiento de documentos que tienen en cuenta su estructura y contenido permiten recuperarlos más eficazmente que los que los tratan solamente como Binary Large Objects (BLOB).

- Procedimientos especiales de localización de objetos basados en su contenido. Así, para encontrar documentos por el contenido de su texto, entre otras, se pueden hacer búsquedas: por palabras, por palabras con condiciones boleanas, por una frase exactamente, por una frase aproximadamente, por proximidad, por transposición y sustitución de letras, etc.
- Técnicas eficientes de recuperación de los objetos seleccionados, es decir, con los recursos necesarios para transferirlos, en un tiempo razonable y sin errores.

Es muy importante disponer de los sistemas más eficientes en almacenamiento y recuperación de información

ARQUITECTURA DE UNA ORDBMS Y ACCESO DESDE LA WEB

Las primeras soluciones para disponer de un sistema que pudiera gestionar aceptablemente datos complejos consistieron en diseñar y manejar la lógica de éstos fuera del RDBMS, dejando a este último como un mero sistema de almacenamiento. Ello implicaba la utilización de técnicas propietarias difíciles de coordinar con la funcionalidad del sistema relacional.

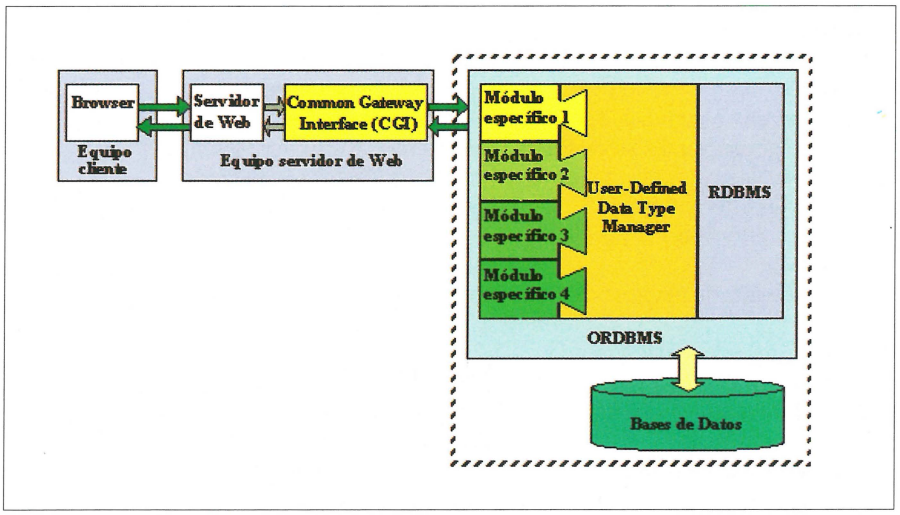


Figura 1. Arquitectura modular de una ORDBMS y su uso desde de la Web

Dado que cada campo de aplicación normalmente precisa tipos de datos determinados cuyo tratamiento es diferente según el caso, no tiene sentido pretender una solución global para la gestión de los posibles datos complejos. La tendencia de las principales empresas del sector, como son Informix, Oracle, IBM o Sybase, consiste en buscar respuestas específicas para cada campo de aplicación mediante el desarrollo de módulos independientes que sean soportados por una capa común, denominada User-Defined Data Type Manager (UDTM), que actúa como interfaz entre cada uno de los módulos y el RDBMS. El desarrollo de uno de esos módulos implica la definición de nuevos tipos de datos y la confección de las rutinas que operen con ellos.

Según parece, Informix es la empresa que proporciona más integración entre sus módulos, denominados DataBlades, y el motor RDBMS, mientras que Sybase es la que plantea una mayor independencia al organizar sus módulos específicos en diferentes servidores y presentarlos unidos a través de su interfaz OpenServer. En la figura 1 se representa la arquitectura completa del sistema cuando el acceso a las bases de datos se realiza a través de la Web.

Cuando el acceso a los datos se realiza a través de la Web, el usuario final observa la información bajo la apariencia de páginas Web. En un caso general, el equipo cliente debe disponer de un Browser que permita la interacción con las páginas Web y el equipo servidor de Web ha de incorporar el correspondiente software de gestión de páginas; además, este equipo requiere de un Common Gateway Interface (CGI), software para acceder a los datos. El proceso completo de obtención de información puede resumirse en los siguientes pasos:

1. Mediante un Universal Resource Locator (URL), el Browser realiza una solicitud al servidor de Web.
2. Si el URL invoca al CGI, éste compone las instrucciones necesarias para acceder a los datos.
3. Esta instrucción es recibida por el ORDBMS.
4. El ORDBMS accede a la base de datos.
5. El ORDBMS retorna los datos al CGI.
6. El CGI procesa la información, incorporándola a una página del Web Server.
7. Esta página es enviada al Browser para ser presentada al usuario.

Esta página puede incluir los tags necesarios para generar un nuevo URL y repetir sucesivamente el proceso anterior.

LOS DATABLADES DE INFORMIX

Todo DataBlade está compuesto por una colección de objetos de Base de Datos y código que extienden la funcionalidad del Gestor Relacional Informix Dynamic Server. En cierta forma puede considerarse como una librería de clases C++ o Java que encapsula tipos de datos especializados, tales como imágenes, textos, etc. Generalmente, la construcción de un

DataBlade implica la definición y el desarrollo de nuevos tipos de datos, rutinas, interfaces, tablas e índices y código de cliente.

Según la información a manejar por el DataBlade, éste deberá proporcionar al usuario nuevos tipos de datos que tratará de igual forma que los tipos de datos básicos, es decir, sus valores podrán ser almacenados, examinados mediante "queries" o llamadas a funciones, pasados como argumento a funciones de base de datos e indexados.

Los nuevos tipos de datos que el DataBlade aporta al sistema se denominan tipos de datos definidos por el usuario, "User-defined DataTypes" (UDT's). Para soportar estos nuevos tipos de datos, Informix Dynamic Server proporciona a los desarrolladores de DataBlades los tipos que aparecen en la figura 2 y son los que realmente se incorporan a la base de datos cuando se registra y se utiliza el DataBlade.

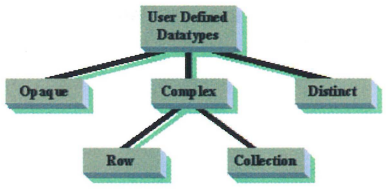


Figura 2. User Defined Datatypes con Informix

- El tipo de dato Row es un tipo de datos que permite agrupar una relación de items de datos, básicos (simples) o definidos por el usuario, bajo un nombre con el que puede ser citado el grupo. Los elementos individuales de un dato de tipo Row normalmente tienen diferentes tipos de dato.
- El tipo de dato Collection es un agrupamiento como el anterior en el que los datos agrupados son del mismo tipo.
- El tipo de dato Distinct permite personalizar un tipo de dato existente. Por ejemplo, el tipo de dato Euro podría crearse desde un tipo de dato decimal.

INFORME ETSIT DE SANTANDER

Nombre del Centro. Escuela Técnica Superior de Ingenieros Industriales y de Telecomunicación

Dirección. Avda. de los Castros, s/n. C.P. 39005 – Santander
Teléfono: 942-201870/71/72 - Fax: 942-201873 - e-mail: etsiiyt@ccaix3.unican.es

Títulos que se otorgan actualmente. Ingeniero Industrial, Ingeniero Químico, Ingeniero de Telecomunicación, Ingeniero Técnico Industrial (especialidad en Electricidad), Ingeniero Técnico Industrial (especialidad en Electrónica Industrial), Ingeniero Técnico Industrial (especialidad en Mecánica), Ingeniero Técnico Industrial (especialidad en Química Industrial), Ingeniero Técnico de Telecomunicación (especialidad en Sistemas Electrónicos).

Fecha de constitución de la Escuela. En 1942 como "Escuela de Peritos Industriales", y en 1994 se denomina como en la actualidad.

Fecha en la que salió la primera promoción. La primera promoción de Ingeniero de Telecomunicación fue en el curso 1993/9

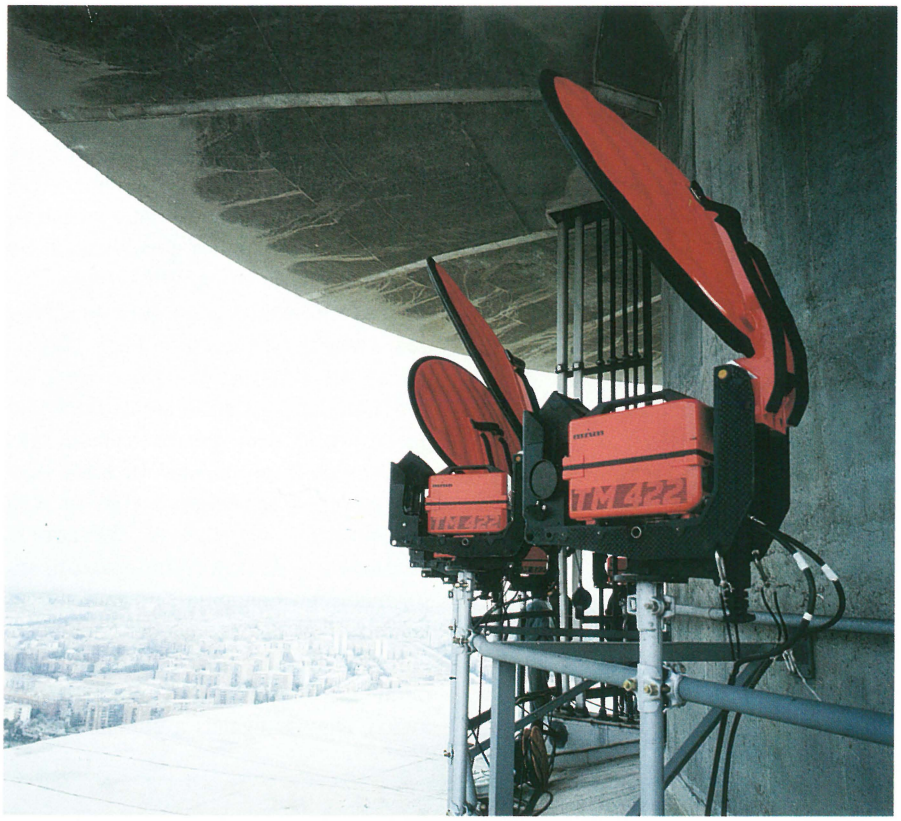
Total de Ingenieros de Telecomunicación salidos de esta Escuela. 243

Situación actual:
- "Claustro de Profesores" se recoge en la figura de "La Junta de Escuela". Su composición es: número total de profesores: 55; Titular de Escuela Universitaria: 11; Catedrático de Escuela Universitaria: 2; Titular de Universidad: 22; Catedrático de Universidad: 19; Profesor Asociado: 1
- Alumnos matriculados en el curso 2000/2001: total de alumnos matriculados de todos los planes en este Centro: 2.802

Total de alumnos matriculados en primer curso de Ingeniero de Telecomunicación. 555. El detalle por año de carrera no se puede establecer con los nuevos planes de estudio que la distribución es por créditos.

Programas de doctorado

Departamentos	Título del Programa	Tesis/curso 99-00
Química	Ingeniería Química/ Química	3
Ingeniería de Comunicaciones	Ingeniería de Comunicaciones	1
Ingeniería Estructural y Mecánica	Ingeniería Estructural y Mecánica	0
Ingeniería Eléctrica y Energética	Ingeniería Eléctrica y Energética	3
Transportes y Tecnología de Proyectos y Procesos	Transportes y Tecnología de Proyectos y Procesos	0
Filología	Pensamiento, Lengua y Cultura	0
Administración de Empresas	Administración de Empresas y Organización Industrial	1
Ciencia e Ingeniería del Terreno y de los Materiales	Ciencia e Ingeniería del Terreno y de los Materiales	0
Ciencias de la Tierra y Física de la Materia Condensada	Ciencias de la Tierra y Física de la Materia Condensada	0
Electrónica y Computadores	Electrónica y Computadores	1
T.E.I.S.A.	Tecnología Electrónica, Ingeniería de Sistemas y Automática	1
Ingeniería Geográfica y Técnicas de Expresión Gráfica	Ingeniería Geográfica y Expresión Gráfica en la Ingeniería	0
Matemática Aplicada y Ciencias de la Computación	Matemática Aplicada y Ciencias de la Computación	1



– Los datos de tipo Opaque sirven para almacenar directamente estructuras de C, C++, Java. El sistema solamente almacena en la base de datos el contenido de la estructura, sin interpretarlo. El acceso al contenido de datos de este tipo ha de realizarse a través de rutinas escritas por el usuario.

Otros componentes de un DataBlade son las rutinas que operan sobre los datos. Éstas pueden actuar sobre datos de tipos definidos por el desarrollador del DataBlade o sobre cualquier otro tipo de dato reconocido por el servidor, incluyéndose otros definidos en otros DataBlades. Las rutinas amplían las capacidades de procesado de la base de datos, suministrando nuevas funcionalidades propias de un campo de aplicación.

Rutina definida por el usuario (User-defined routine) es un término que se utiliza en SQL3 para referirse a procedimientos y funciones definidos por el usuario. La diferencia entre ambos consiste en que un procedimiento no puede retornar un valor, mientras que una función sí. Estas rutinas pueden ser escritas en SPL (Store Procedure Language de Informix) o en C, C++ o Java. Las rutinas SPL se almacenan en la propia base de datos mientras que las escritas otros lenguajes son cargadas como un fichero de objetos compartidos o una librería dinámica (DLL). Las rutinas definidas por el usuario admiten la siguiente clasificación:

Cada campo de aplicación normalmente precisa tipos de datos determinados cuyo tratamiento es diferente

– Rutinas para soportar datos definidos por el usuario, que son análogas a los métodos para clases de C++. Es el caso de los datos de tipo Opaque, que requieren rutinas escritas por el usuario para su manejo.
– Funciones Cast, que sirven para convertir datos de un tipo en otro.
– Funciones de usuario que pueden ser utilizadas en expresiones de instrucciones SQL. Por ejemplo en la lista de una instrucción SELECT o en cláusulas WHERE, GROUP BY o HAVING.
– Rutinas de soporte de Métodos de Acceso Definidos por el Usuario.

Los métodos de acceso operan sobre tablas e índices que son gestionados por el servidor de bases de datos. El DataBlade puede utilizar los métodos de acceso existentes o incorporar los suyos propios. La definición de un método de acceso conlleva la inclusión de varias operaciones como las de abrir un índice, buscar el siguiente registro, insertar un registro, borrarlo o modificarlo y cerrar un índice.

Mientras que los índices basados en B-trees se utilizan para agilizar las búsquedas en datos linealmente ordenados, muchos de los nuevos tipos de datos toman ventaja de una ordenación no lineal por lo que se ven beneficiados si se define un método de acceso que mejore su rendimiento.

Para la utilización de estos elementos deben existir unas reglas que todos los DataBlades han de cumplir. Todo DataBlade debe disponer de una interfaz, es decir, de un conjunto de funciones que, siguiendo la especificación definida en un estándar, permita compartir con otros DataBlades los servicios que ofrecen.

Los DataBlades, para sus propios procesos, generan tablas e índices que se incorporan a la base de datos. Por ejemplo, un DataBlade que trabaja con imágenes registra en una tabla los formatos de imagen que procesa. Esto, además, facilita a los desarrolladores la extensión del DataBlade a nuevos tipos, simplemente con añadir un nuevo formato.

Por último, el DataBlade puede ir acompañado de una herramienta cliente que sirva de ayuda al desarrollo de aplicaciones.