Chinchilla Arley, Ricardo
Fragmentación de datos en bases de datos distribuidas.
Tecnología en Marcha. Vol. 16 N° 4.

Fragmentación de datos en bases de datos distribuidas

Ricardo Chinchilla Arley [1]

Al estudiar el tema de bases de datos distribuidas (BDD) se deben tratar dos elementos fundamentales: las bases de datos y los sistemas de tecnología de información distribuidos sobre redes de computadoras.

Palabras clave

Bases de datos, almacenamiento distribuido de datos, diseño de baes de datos.

Resumen

La investigación se inicia con una definición de base de datos distribuida, su estructura y los factores por tomar en cuenta para su diseño. Después se centra en la distribución de los datos, la descripción de los tipos de fragmentación (horizontal, vertical y mixta) y sus respectivas operaciones algebraicas. Finalmente, se describe brevemente cómo se realiza la localización de los datos dentro de la base de datos.

Descriptores: Bases de datos, bases de datos distribuidas, redes de computadoras.

Introducción

Al estudiar el tema de bases de datos distribuidas (BDD) se deben tratar dos elementos fundamentales: las bases de datos y los sistemas de tecnología de información distribuidos sobre redes de computadoras. Una base de datos es una colección lógicamente coherente de datos interrelacionados con un significado inherente y creada con un propósito particular. Por su parte, un sistema distribuido interconecta los lugares que tienen diferentes elementos de tecnología de informacion para capturar y almacenar datos, procesarlos y enviarlos a otros sistemas. Ahora bien, una BDD podría definirse como una colección de múltiples bases de datos lógicamente interrelacionadas (denominadas localidades) y distribuidas sobre una red de computadoras. Se podría decir que es la unión de un sistema de base de datos y un sistema de redes de computadoras, lo cual es posible gracias al desarrollo de las telecomunicaciones.

Su funcionamiento básico consiste en el almacenamiento de datos en varios computadores, los cuales se comunican a través de diversos medios, tales como cables telefónicos, coaxiales o fibra óptica. Cada localidad mantiene una base de datos local y puede procesar tanto transacciones locales (acceden a datos locales), como globales (acceden

1 Ricardo Chinchilla Arley, Master en Computación ITCR, funcionario Centro Centroamericano de Población, Director Biblioteca Sede del Atlántico, Universidad de Costa Rica. (*richar@ccp.ucr.ac.cr*).

a datos de varias localidades) (Figura 1). Para manipular la información almacenada, se utiliza un Sistema Administrador de Base de Datos Distribuida (SABDD), el cual constituye el software que permite la administración de la base, de tal forma que su manejo sea transparente al

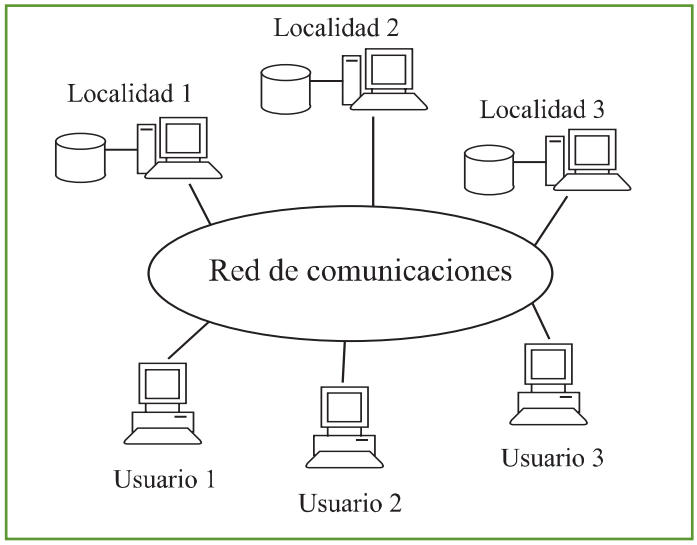

Figura 1
Medio ambiente de una bdd

El principal objetivo de un SABDD es compartir y acceder a la información de una manera confiable y eficaz.

usuario, como si se tratara de una base local.

Las localidades de un SABDD pueden estar dispersas en forma física dentro de una red, ya sea de área local (LAN) o extendida (WAN).

El principal objetivo de un SABDD es compartir y acceder a la información de una manera confiable y eficaz. Si varias localidades diferentes están conectadas entre sí, un usuario de una localidad puede acceder a los datos de otra. Además, si se produce un fallo en una localidad, es posible que las otras localidades continúen trabajando sin contratiempos. Para el usuario, la distribución es totalmente trasparente y maneja la base datos como si estuviese centralizada. De esta manera, cuando realiza una consulta que implica datos de varias localidades, la consulta se divide en subconsultas que se ejecutan en paralelo.

Fragmentación

Para acceder a una porción de datos en un SABDD, un programa busca y recupera el dato que necesita. Dicho dato no se mueve por todo el sistema, sino que reside en una localidad. Los procesos que se ejecutan utilizan operaciones algebraicas para determinar su ubicación y recuperarlo.

Una base de datos relacional presenta la información esencialmente en forma de tablas. Este tipo de base de datos se baja en un esquema relacional, el cual se define como un conjunto de atributos $r=\{a_2, ..., a_n\}$ donde cada atributo ai es el nombre de un rol de un dominio y donde dominio $D_i=dom(a_i)$.

Una relación o instancia de relación r de un esquema de relación $r(a_1, a_2, ..., a_n)$ es un conjunto de tuples $\{t_1, ..., t_n\}$. Cada tuple, por su parte, es una lista ordenada de valores

$t= <v_1, ..., v_n>$

y $\forall\ i \in \{1, .., n\} : vi \in dom\ (a_i)\ o\ v_i = null$

Existen dos alternativas para dividir cada tabla en tablas más pequeñas: la división horizontal (tuplas) o la división vertical (atributos). Existe también lo que se conoce como fragmentación híbrida o mixta. Así, sea una relación r, esta se fragmentará en $r_1, r_2, ..., r_n$. Los fragmentos contienen información suficiente para construir la relación r original. Esta reconstrucción puede llevarse a cabo utilizando la operación de unión $\cup$ o un tipo especial de unión sobre los diversos fragmentos denominado *join* $\bowtie$. Al reconstruir la información se debe

evitar producir tuples espurios derivados de un planteo incorrecto de la operación.

El grado de fragmentación puede ir de 0% a 100%, tanto vertical (atributos individuales), como horizontal (tuples individuales). El nivel debe ser definido de acuerdo con la aplicación que se desarrollará.

Ozsu (1991) establece tres reglas para una fragmentación exitosa:

Completitud. Si una relación r es descompuesta en los fragmentos r_1, r_2, ..., r_n cada dato que se encuentra en r también debe ser encontrado un uno o más fragmentos r_i.

Reconstrucción. Si una relación r es fragmentada en r_1, r_2, ..., r_n, es posible definir un operador relacional ∇ tal que

$r = \nabla r_i, \forall r_i \in Fr$

Disjointness. Si una relación r es descompuesta horizontalmente en fragmentos r_1, r_2, ..., r_n, y el dato d_i está en r_j, este dato no va a estar en ningún otro fragmento $r_k (k \neq J)$. Si la relación r es descompuesta verticalmente, su atribuyo llave primaria es repetido en todos los fragmentos. Esto quiere decir que la fragmentación se produce en los atributos que no contienen la llave primaria de la relación.

La finalidad de la fragmentación ha de ser siempre la búsqueda de un mejor rendimiento.

Tomemos el siguiente esquema de relación a manera de ejemplo:

depósito (nom_cliente, num_cta, nom_suc, saldo)

nom_cliente	num_cta	nom_suc	saldo
Luis Chávez	7465	San Pedro	30000
Silvia Díaz	4582	San Pedro	70000
José Mora	3256	Cartago	20000
Julia Meza	2145	Cartago	50000
Juan Vargas	9521	Santa Ana	10000
Xinia Víquez	3469	Santa Ana	40000

Fragmentación horizontal

Tomando la misma relación r, la cual se divide en dos subconjuntos r_1, r_2, ..., r_n, cada tupla de r debe pertenecer a alguno de los fragmentos y cada a_i es un atributo definido sobre un dominio D_i. Así mismo, un fragmento puede definirse como una selección de r y para construir el fragmento utilizamos un predicado p_i ya sea simple o compuesto

$a_i = \sigma p_i(r)$

donde el predicado simple es definido de la forma

$p_i: a_i \theta valor$

donde $\theta \in \{=, <, >, \leq, \geq, \neq\}$ y *valor* es seleccionado del dominio de a_i θ *valor*. Un predicado compuesto es definido por la conjunción de predicados simples.

Para la reconstrucción de la relación original, simplemente se unen todos los fragmentos

$r = r_1 \cup r_2 \cup ... \cup r_n$

Tomando el ejemplo propuesto y teniendo tres sucursales (San Pedro, Cartago, Santa Ana).

depósito1 = σ *nom_suc = "San Pedro" (depósito)*

depósito2 = σ *nom_suc = "Cartago" (depósito)*

depósito3 = σ *nom_suc = "Santa Ana" (depósito)*

De esta manera, el fragmento *depósito1* se almacena en la localidad San Pedro, el fragmento *depósito2* en la localidad Cartago y el *fragmento3* en la localidad Santa Ana.

Para realizar la fragmentación horizontal, debemos tomar en cuenta dos factores importantes:

- La selectividad del término del predicado, esto es, el número de tuplas de una relación que pueden ser accesadas por una consulta de

depósito 1

nom_cliente	num_cta	nom_suc	saldo
Luis Chávez	7465	San Pedro	30000
Silvia Díaz	4582	San Pedro	70000

depósito 2

nom_cliente	num_cta	nom_suc	saldo
José Mora	3256	Cartago	20000
Julia Meza	2145	Cartago	50000

depósito 3

nom_cliente	num_cta	nom_suc	saldo
Juan Vargas	9521	Santa Ana	10000
Xinia Víquez	3469	Santa Ana	40000

usuario y que satisfacen el término del predicado utilizado.

- La frecuencia de acceso a los datos.

Fragmentación vertical

Para realizar la fragmentación vertical de *r(R)* se debe involucrar la definición de varios subconjuntos R_1, R_2, ..., R_n tales que:

$R = R_1 \cup R_2 \cup ... \cup R_n$

Cada fragmento ri de r se define por:

$r_i = \prod R_i (r)$

La relación *r* puede reconstruirse a partir de los fragmentos realizando la operación de *join*, en la cual se combina una selección de un producto cartesiano

$r = r_1 \bowtie r_2 \bowtie ... \bowtie r_n$

Para realizar la fragmentación vertical es necesario agregar a cada fragmento la llave primaria de la relación, con el fin de comprobar la integridad. Otra alternativa consiste en agregar un atributo adicional, denominado *id_tupla*, el cual será la dirección física lógica de una tupla, lo cual lo convierte en una llave del esquema. De esta manera, si descomponemos verticalmente el ejemplo en

depósito_vert1=(nom_cliente, nom_suc, id_tupla)

depósito_vert2=(num_cta, saldo, id_tupla)

y utilizando la siguiente operación para la fragmentación

depósito4 = $\prod$*depósito_vert1 (depósito)*

depósito5 = $\prod$*depósito_vert2 (depósito)*

tendremos lo siguiente

depósito 4

nom_cliente	nom_suc	id_tupla
Luis Chávez	San Pedro	1
Silvia Díaz	San Pedro	2
José Mora	Cartago	3
Julia Meza	Cartago	4
Juan Vargas	Santa Ana	5
Xinia Víquez	Santa Ana	6

depósito 5

nom_cliente	saldo	id_tupla
Luis Chávez	30000	1
Silvia Díaz	70000	2
José Mora	20000	3
Julia Meza	50000	4
Juan Vargas	10000	5
Xinia Víquez	40000	6

Para la reconstrucción basta con realizar la operación

$\prod$*depósito (depósito 4* $\bowtie$ *depósito 5)*

Fragmentación híbrida o mixta

En algunos casos, la fragmentación horizontal y vertical no es suficiente para satisfacer los requerimientos de los usuarios de la aplicación. Es posible aplicar fragmentación horizontal a un fragmento obtenido con fragmentación vertical o viceversa (figura 2).

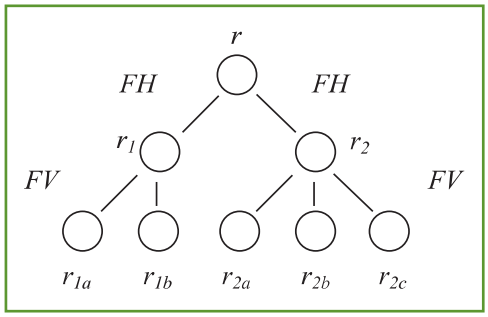

Figura 2
Fragmentación híbrida o mixta

En algunos casos, la fragmentación horizontal y vertical no es suficiente para satisfacer los requerimientos de los usuarios de la aplicación.

Por ejemplo, al tener una relación r dividida en los fragmentos r_1, r_2, ..., r_n, obtenidos al aplicar la fragmentación horizontal o vertical, es posible aplicar una fragmentación a uno de los fragmentos ya obtenidos con antelación. Así, si se desea fragmentar horizontalmente *depósito4* obtenido anteriormente con fragmentación vertical, tendremos

Depósito4a = σ *nom_suc="San Pedro" (depósito4)*

Depósito4b = σ *nom_suc="Cartago" (depósito4)*

Depósito4c = σ *nom_suc="Santa Ana" (depósito4)*

Con lo que la relación *r* se divide en 4 fragmentos: *depósito4a, depósito4b, depósito4c* y *depósito5*.

Cada uno de estos fragmentos se almacena en localidades diferentes. Para la reconstrucción, se debe realizar un *join* entre los fragmentos generados por la fragmentación vertical y la horizontal (figura 3).

depósito 4a

nom_cliente	nom_suc	id_tupla
Luis Chávez	San Pedro	1
Silvia Díaz	San Pedro	2

depósito 4b

nom_cliente	nom_suc	id_tupla
José Mora	Cartago	3
Julia Meza	Cartago	4

depósito 4c

nom_cliente	nom_suc	id_tupla
Juan Vargas	Santa Ana	5
Xinia Víquez	Santa Ana	6

depósito 5

nom_cliente	saldo	id_tupla
Luis Chávez	30000	1
Silvia Díaz	70000	2
José Mora	20000	3
Julia Meza	50000	4
Juan Vargas	10000	5
Xinia Víquez	40000	6

Localización de los datos

Todas las consultas realizadas por un usuario del sistema se ejecutan en forma atómica, es decir, se ejecutan todas las instrucciones de la consulta o no se

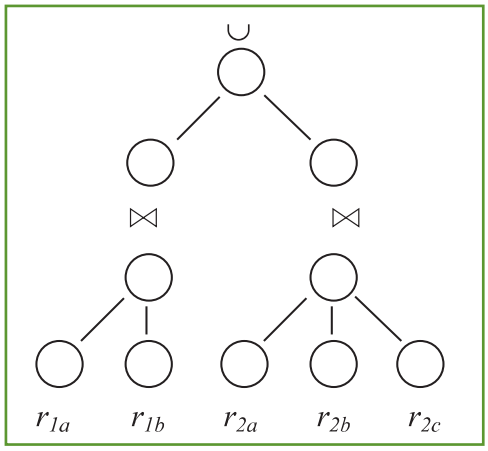

Figura 3
Reconstrucción de la fragmentación híbrida o mixta

Todas las consultas realizadas por un usuario del sistema se ejecutan en forma atómica, es decir, se ejecutan todas las instrucciones de la consulta o no se ejecuta ninguna.

ejecuta ninguna. Para un sistema distribuido es bastante complicado garantizar la atomicidad de las consultas puesto que es posible que participen varias localidades en su ejecución. El fallo de una de esta localidades o de la línea de comunicación puede dar como resultado cálculos erróneos. Para enfrentar este problema, cada localidad cuenta con un administrador de transacciones el cual se encarga de gestionar la ejecución de aquellas transacciones que accedan a datos almacenados en esa localidad. Los administradores de las diferentes localidades cooperan para ejecutar las transacciones globales.

Como se ha visto hasta este momento, cualquier entidad fragmentada puede ser reconstruida nuevamente. El proceso de reconstrucción o fragmentación reversiva se denomina programa de localización.

Al realizar una consulta, la cual es formulada en un lenguaje de consulta de alto nivel, es automáticamente traducida en un plan de ejecución de bajo nivel. Esta transformación tiene dos requerimientos: la consulta se debe haber formulado correctamente, para que el plan de ejecución produzca los resutados esperados, y debe ir en concordancia de la minimización de costos por el consumo de recursos.

Como el direccionamiento de la consulta tiene dos requerimientos, esta debe concentrarse en dos pasos secuenciales: localización de los datos y optimización global. Estos pasos son precedidos por su descomposición y reescritura en álgebra relacional. La localización de los datos transforma una consulta algebraica en su consulta fragmentada equivalente (expresada en fragmentos de la base almacenada en diferentes localidades). La optimización global genera un óptimo plan de ejecución para la consulta fragmentada, creando decisiones relativas a la operación de ordenamiento, movimiento de los datos entre las localidades, y la selección de los algoritmos distribuidos y locales para la operación de los datos.

Discusión

Las bases de datos distribuidas han permitido que los sistemas de base de datos puedan crecer considerablemente en la cantidad de información que manejan, al permitir la integración de múltiples computadoras, distribuidas en distintas partes de un área geográfica determinada y cada una tener los recursos suficientes para contener y manejar por sí sola grandes cantidades de información. Se comportan de la misma forma que las bases de datos relacionales, incluyendo la arquitectura de tres niveles, afectando solamente al nivel físico la manera de almacenar sus datos.

El ejemplo de sistemas distribuidos permite dividir las consultas para ser realizadas a una mayor velocidad por distintas máquinas o procesadores, incrementando enormemente el rendimiento del sistema de base de datos. Ofrece también la ventaja de una

mayor confiabilidad, debido a la redundancia. Por otro lado, el desarrollo del Internet facilita aún más la aplicación de este esquema, pero también aumenta sus riesgos y obliga a invertir más en seguridad.

Se debe considerar para el diseño de la distribución de los datos, los factores que contribuyen a un diseño óptimo. La organización lógica de la base de datos, la localización de las aplicaciones, las características de acceso de las aplicaciones a la base de datos y las características del sistema en cada sitio, tienen una decisiva influencia sobre la distribución. La información necesaria para el diseño de la distribución puede dividirse en cuatro categorías: la información de la base de datos, la información de la aplicación, la información sobre la red y la información sobre las máquinas conectadas a esta. Las dos últimas son de carácter cuantitativo y servirán, principalmente, para desarrollar el proceso de asignación.

La mayor ventaja que presenta esta tecnología radica en el desarrollo y crecimiento modular, donde distintas localidades podrán ser incluidas conforme va creciento en sistema.

La mayor ventaja que presenta esta tecnología radica en el desarrollo y crecimiento modular, donde distintas localidades podrán ser incluidas conforme va creciento en sistema, lo cual definitivamente influye de manera muy positiva en el costo final del proyecto.

Al iniciar el estudio de un desarrollo sobre esta tecnología, se deben tener presentes algunos aspectos que han de ser manejados adecuadamente. Es preciso tomar las decisiones sobre la distribución física de la información y de qué manera será fragmentada. Se debe determinar y distribuir las copias de los fragmentos en distintas localidades, la forma en que se va a manejar la posibilidad de modificaciones concurrentes de la misma información y el control de las actualizaciones de la información duplicada para mantener la consistencia.

Otro aspecto de gran importancia es el de mecanismos de recuperación después de fallas, tales como la posibilidad de que, debido a problemas en la comunicación, el sistema quede dividido en dos o más *sub-redes* fragmentadas.

Aunque las teorías de sistemas distribuidos tienen un camino largo en el ámbito de la investigación, al parecer hasta hace poco se han aplicado estos sistemas al ámbito comercial, donde existe un nicho muy grande de aplicación. Dicho nicho tiende a crecer en la medida en que se adentra cada vez más en un mundo globalizado, donde la información es utilizada y creada de manera global.

Bibliografía

Bases de datos distribuidas. s.l.: Escuela Universitaria de Informática, s.f. http://www-oei.eui.upm.es/Asignaturas/BD/Distribuidas.htm

Bases de datos distribuidas: doce normas para su manejo. Mundo de la computación 4(22):19:21, 1990.

Bouguettaya, Athman; Papazoglou, Mike; King, Roger. *On building a hyperdistributed database. Information systems*. 20(7):557-577, 1995.

Cahn, Robert. *Wide area network desing: concepts and tools form optimization*. San Francisco: Morgan Kaufmann Publishers, 1998.

González Alvarado, Carlos. *Sistemas de bases de datos*. Cartago: Editorial Tecnológica de Costa Rica, 1996.

González Martín, Oscar. *Arquitecturas de sistemas de bases de datos*. Castilla La Mancha, España: Universidad de Castilla La Mancha, 2000. http://alarcos.inf-cr.uclm.es/doc/bda/doc/trab/T9900_OGonzalez.pdf

Hua, Kien; Lee, Chiang; Young, Honestly. *Data partitioning for multicomputer database systems: a cell-based approach. Información system*. 18(5):329-342, 1993.

Korth, Henry; Silberschatz, Abraham. *Fundamentos de bases de datos*. Madrid: McGraw-Hill, 1993.

Nitzberg, Bill; Lo, Virginia. *Distributed shared memory: a survey of issues and algorithms*. Computer. 24(8):52-60, 1991.

Rodríguez Santos, Jorge. [helmantika@terra.es] *Diseño de bases de datos distribuidas*. [http://usurios.lycos.es/ jrodr35/]

Ozsu, Tammer; Valduriez, Patrick. *Principles of distributed database systems*. New Jersey: Prentice-Hall, 1991.

Tanenbaum, Andrew. *Redes de computadoras*. 3 ed. México: Prentice-Hall, 1997.

Vargas, Carlos. *Base de datos*. Cartago: ITCR, 1995.