Unidad III. Arquitecturas para Bases de Datos Distribuidas

M. en C. Euler Hernández Contreras

Contenido

1. Modelos arquitectónicos para Sistemas Gestores de Bases de Datos Distribuidas
   a) Autonomía
   b) Distribución
   c) Heterogeneidad
   d) Otras Alternativas

2. Arquitecturas de Sistemas de bases de datos distribuidas
   a) Sistemas Cliente/Servidor
   b) Sistemas de Bases de Datos Distribuidas
   c) Sistemas Multibase

3. Arquitectura de referencia para una base de datos distribuida.
   a) Esquema Global
   b) Esquema de Fragmentación
   c) Esquema de Localización
   d) Características de la arquitectura de referencia.

Referencia Bibliográfica

1. Michael V. Mannino. Administración de bases de datos, diseño y desarrollo de aplicaciones, Tercera Edición. Mc Graw Hill Interamericana, México 2007, 712 págs.

2. M. Tamer Özsu, Patrick Valduriez. Principles of Distributed Database Systems. Second Edition. Prentice-Hall, Estados Unidos. 1999, págs. 666

3. Stefano Ceri, Giuseppe Pelagatti. Distributed Databases Principles & Sistems. Mc Graw-Hill Inc., 1985, págs. 385

Modelos arquitectónicos para Sistemas Gestores de Bases de datos Distribuidas[2]**.**

Se pueden considerar posibles formas en el cual múltiples bases de datos pueden estar unidas para compartir múltiples SGDB. En la Figura 1 se usa una clasificación que organiza los sistemas con las características tales como autonomía de los sistemas locales, su distribución y su heterogeneidad.

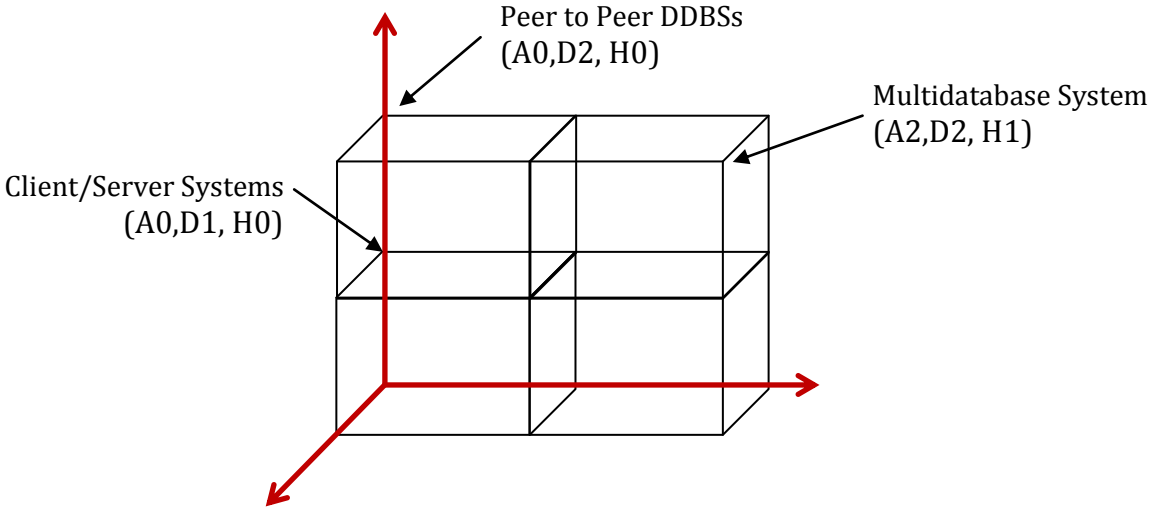

Figura 1. Alternativas de Implementación para Sistemas Gestores de Bases de Datos

Autonomía:
Se refiere al control de distribución mas no a los datos. Indica el grado en el cual un SGBD puede operar independientemente.

Autonomía es en función de un número de factores tales como:

1.- Intercambio de información.
2.- Ejecución independiente de transacciones

Los Requerimientos de un sistema autónomo han sido especificados en diversas maneras:

a) Las operaciones locales de un SGBD no afectan su participación en el sistema multibase.
b) La manera en el cual los SGBD (individuales) procesan y optimizan las consultas, no deberían afectar la ejecución de los "queries" globales que acceden a múltiples BD.
c) La consistencia del sistema u operación no debería afectar cuando un SGBD de forma individual se une o deja de formar parte de un sistema multibase federada.

Dimensiones de la Autonomía.

a) En el diseño:
   Cada SGBD es libre de usar modelos de datos y técnicas de administración de transacciones
b) En la comunicación:
   Cada SGBD es libre de tomar sus propias decisiones como el tipo de información que quieren proporcionan a otros SGBD o el software que controle su ejecución global.
c) En la ejecución:
   Cada SGBD puede ejecutar transacciones que le son enviados y ejecutarlos como el quiera.

Taxonomía de sistemas autónomos.

a) Integración fuerte:
Hace referencia que una simple imagen BD (enteramente) esta disponible a cualquier usuario que quiera compartir información, esta imagen puede residir en múltiples bases de datos. Desde el punto de vista de los usuarios, los datos están lógicamente centralizados en una base de datos.

En un sistema autónomo con integración fuerte los administradores de los datos son implementados de tal manera que uno de ellos controla el proceso de cada solicitud del usuario no importando si requiere el servicio de mas un administrador de datos.

b) Sistemas semiautónomos (sistemas federados):
Consiste de SGBD que pueden operar independiente y cada SGBD ha decidido participar en una federación para hacer sus datos disponibles. Cada SGDB determina que partes de su propia base de datos harán accesibles a los usuarios de otro SGBD. Ellos no son completamente sistemas autónomos debido a que necesitan ser modificados para hacer posible el intercambio de información entre ellos.

c) Aislamiento total:
Son sistemas stand-Alone (SGDB), el cuan no saben la existencia de otro SGBD o bien no hay comunicación entre ellos.

Distribución:

La distribución hace referencia a distribución de los datos en múltiples sitios. Existen diversas maneras en que los SGBD se pueden distribuir.

Distribución:

a) cliente / servidor
   Es el más popular ya que concentra la administración de los datos es por parte del servidor que proporciona dichos datos a los clientes los cuales acceden por medio de una (GUI). La comunicación se establece al compartir información entre

clientes y servidores. Los SGBD cliente / servidor representa los primeros intentos de una funcionalidad distribuida.

b) punto a punto. (distribución completa)
No hay distinción entre máquinas cliente y servidores. Cada máquina tiene una funcionalidad completa del SGBD y puede comunicarse con otras máquinas para ejecutar consultas y transacciones. Estos sistemas también son llamados distribución completa.

Heterogeneidad:

La heterogeneidad puede ocurrir en varias formas en un sistema distribuido que van desde heterogeneidad en hardware, diferentes protocolos de red y diversos sistemas gestores de bases de datos. También la heterogeneidad se refleja en los modelos de datos, lenguajes de consulta y protocolos de administración de transacciones.

Otras Alternativas:

Considerando las alternativas arquitectónicas mostradas en la Figura 1, y partiendo desde el origen hasta la dimensión de autonomía, usaremos una notación basada en alternativas de tres dimensiones, estas dimensiones son identificadas como A (Autonomía), D (Distribución) y H (Heterogeneidad).

Las alternativas a través de cada dimensión son identificados por números como 0, 1 o 2. Estos números por supuesto tienen diferentes significados para cada dimensión.

Autonomía	0	= Fuerte integración
	1	= Sistema semiautónomo
	2	=Aislamiento total
Distribución	0	=No Distribución
	1	=sistemas cliente/servidor
	2	=Distribución Punto a Punto
Heterogeneidad	0	=Sistema Homogéneo
	1	=Sistema Heterogéneo

En la figura 1, identificamos dos alternativas arquitectónicas (A0,D2,HO) es cual hace referencia a un SGBD distribuido (punto a punto) homogéneo y (A2,D2, H1) el cual representa un sistema distribuido (punto a punto) multibase de datos heterogéneo.

Notaremos que no todas las alternativas arquitectónicas que son identificadas son significativas.

(A0, D0, H0):
Los primeros sistemas son aquellos que están lógicamente integrados. Si en ellos no hay distribución o heterogeneidad, entonces el sistema es un conjunto de múltiples SGBD que están lógicamente integrados. No existen muchos ejemplos de estos sistemas pero se hablan de sistemas multiprocesamiento.

(A0,D0,H1):
Si la heterogeneidad es introducida, puedes tener múltiples SGBD pero proveen una vista integrada al usuario. En el pasado los sistemas eran diseñados para proveer un acceso integrado a bases de datos de red, jerárquicas y relacionales residiendo en una simple máquina.

(A0,D1,H0):
El caso mas interesante es aquel donde la base de datos es distribuida presentándose como una vista integrada al usuario. Esta alternativa representa una distribución cliente/servidor.

(A0,D2,H0):
En este punto el diseño representa un escenario donde el mismo tipo de transparencia es provista al usuario en un ambiente completamente distribuido. No existe distinción entre clientes y servidores, cada sitio provee funcionalidades idénticas.
Se trata de SGDB distribuidas pero sin cualquier heterogeneidad.

(A1,D0,H0):
El siguiente punto en la dimensión de la autonomía son los sistemas semiautónomos comúnmente llamados *SGBD Federados*. Los componentes de un sistema en un ambiente federado tienen autonomía significante en su ejecución pero su participación en una federación indica que ellos deciden cooperar con otros para ejecutar los requerimientos de los usuarios que acceden a múltiples bases de datos.

En este punto en el diseño los componentes del sistema no tienen que ser distribuidos o heterogéneos. Un ejemplo puede ser múltiples instalaciones (sobre la misma máquina) de un SGDB "abierto", el término "abierto" significa que el SGBD sabe como participar en una federación. Cada SGBD está destinado a una función en particular aun en la capa de software proveyendo al usuario una administración integrada.

(A1,D0,H1):
Existen sistemas que introducen la heterogeneidad tan bien como la autonomía llamándose *SGBD heterogéneos federados*. Ejemplos de ellos es fácilmente encontrar en cada día. Suponiendo que existe un SGBD relacional que administra datos estructurados, una imagen de un SGDB que administra imágenes y un servidor de videos. Si queremos proveer una vista integral a los usuarios, es necesario "ocultar" la autonomía y heterogeneidad de los componentes del sistema y establecen una interfaz común.

(A1,D1,H1):
Sistemas de este tipo introducen distribución al colocar componentes del sistema en diferentes máquinas. Estos sistemas pueden ser referenciados como *SGBD distribuidos heterogéneos federados*. Es apropiado establecer que los aspectos de distribución de estos sistemas son menos importantes que su autonomía y heterogeneidad.

(A2,D0,H0):
Si nos movemos completamente a la dimensión autonomía obtendremos lo que llamamos *Arquitectura de un Sistema Multibase* (MDBS). Al identificar las características de estos sistemas, sus componentes no entienden el concepto de cooperación si ellos no saben "como comunicarse entre ellos". Sin heterogeneidad y distribución un MDBS es una conexión interconectada de bases de datos autónomas.

Un Sistema Gestor de BD Mutlibase (multi-DBMS) es el software que es provisto para el manejo de una colección de bases de datos autónomas y transparentes para acceder a ellos. Esto no es una alternativa realista al asumir que no hay heterogeneidad en los componentes del sistema. Esto puede ocurrir en dos casos: cuando tenemos instalados el mismo SGBD y cuando tenemos un conjunto de SGBD idénticos en funcionalidad e interfaz. Ninguna de estas circunstancias son similares de ocurrir.

(A2,D0,H1):
Ese caso es realista ya que puede ocurrir mas que (A1,D0,H1), en que siempre queremos construir aplicaciones que accederán a los datos desde múltiples sistemas de almacenamiento con diferentes características. Algunos sistemas de almacenamiento no necesariamente deben ser SGBD y obviamente no están diseñados y desarrollados con interoperabilidad con algún otro software. El ejemplo que dimos para (A1,D0,H1) se aplica en este caso si asumimos que los componentes del sistema no comprenden el concepto de federación.

(A2,D1,H1) y **(A2,D2,H1)**:
Consideramos ambos casos debido a la similitud del problema. Ambos representan el caso donde los componentes de la base de datos hacen posible que el MDBS es distribuido sobre un número de sitios llamándole *MDBS distribuido*. Como se indico previamente, la solución a los problemas de distribución para ambos caso son similares y el alcance general para llegar a la interoperabilidad no difiere mucho. Quizá la mayor diferencia es en el caso de la distribución cliente/servidor (A2,D1,H1) la interoperabilidad es delgada en los sistemas *middleware* resultante en una arquitectura de tres capas.

Tanto la distribución como la administración en un MDBS es diferente de un SGBD distribuida. En este punto la diferencia fundamental radica en el nivel de autonomía de los SGDB locales. Un Sistema Mutilbase centralizado o distribuido puede ser homogéneo o heterogéneo.

Arquitecturas de Sistemas de Base de datos Distribuida [1]

1. Sistemas Cliente/ Servidor

Como se muestra en la Figura 2, en los sistemas cliente/servidor, la base de datos reside en una máquina en el extremo terminal llamada servidor, y es común que los usuarios accedan a los datos a través de sus estaciones de trabajo, que funcionan como clientes. Las funciones de la base de datos se dividen entre el cliente y el servidor. El cliente proporciona al usuario la interfaz y ejecuta la aplicación lógica, mientras que en el servidor administra los datos y procesa las solicitudes de éstos.

En una transacción interactiva común, el usuario interactúa con la estación de trabajo cliente por medio de una interfaz gráfica provista ya sea por el sistema de base de datos o por un tercer proveedor. Además de manejar la aplicación lógica, el cliente realiza la edición inicial de las solicitudes de datos (por lo general, SQL) revisa la sintaxis de la solicitud y genera otra para la base de datos que es enviada al servidor a través de la red. Éste valida la solicitud por medio de la revisión del diccionario de datos, autorización y las restricciones de integridad; optimiza la consulta; aplica controles de concurrencia y técnicas de recuperación; recupera los datos y los envía de regreso al cliente, que los presenta al usuario. En el cliente también se ejecutan programas de aplicación, en los que las solicitudes de datos pasan por una interfaz de programa de aplicación (IPA) hacia el servidor en una forma similar a la descrita. Si el cliente se apega a un estándar como el ODBC o el JDBC, se comunica con cualquier servidor que provea una interfaz estándar. A diferencia del ambiente de base de datos centralizada, el servidor no procesa aplicaciones por sí misma.

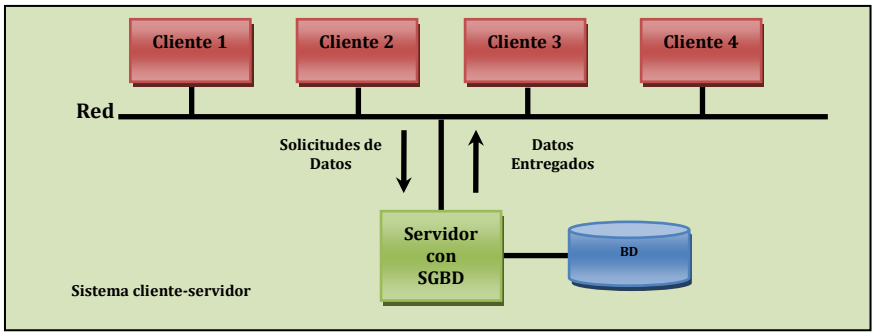

Figura 2. Sistema Cliente/Servidor

Como se puede observar en la Figura 3, la funcionalidad entre clientes y servidores difieren del tipo de SGBD (por ejemplo relacional versus orientado a objetos). En los sistemas relacionales, los servidores únicamente trabajan con la administración de los datos, esto significa que el procesamiento de consultas, optimización, trabajo con las transacciones, y almacenamiento se lleva a cabo del lado del servidor; en el caso de los clientes la aplicación y las interfaces tienen un

componente del SGBD orientado al cliente que es responsable de administrar los datos que son capturados por el cliente y (algunas veces) por bloqueos en las transacciones, existe un sistema operativo y el software de comunicación que corren tanto del lado del cliente como del lado del servidor.

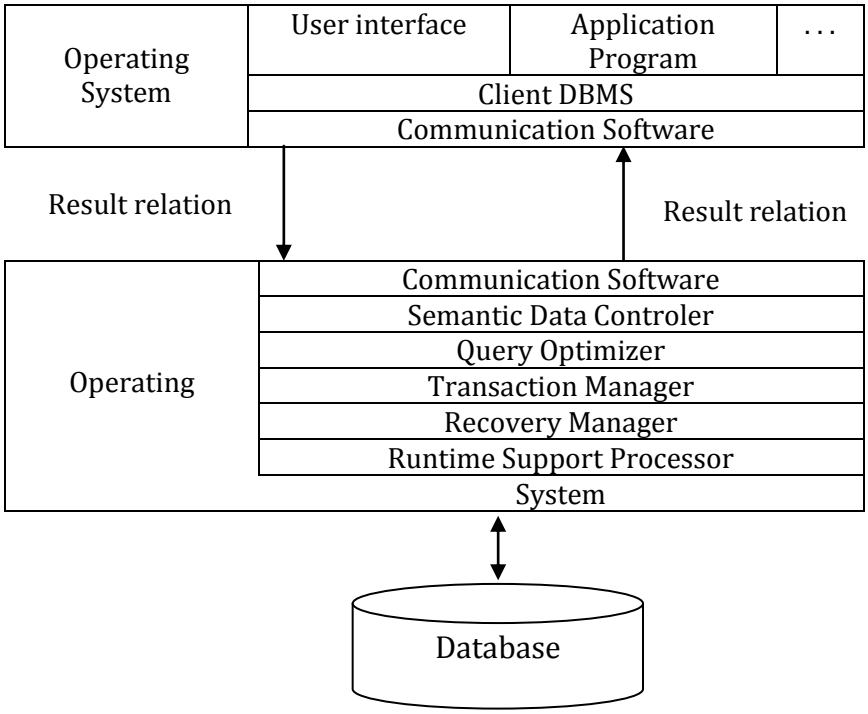

Figura 3. Client/Server Reference Architecture

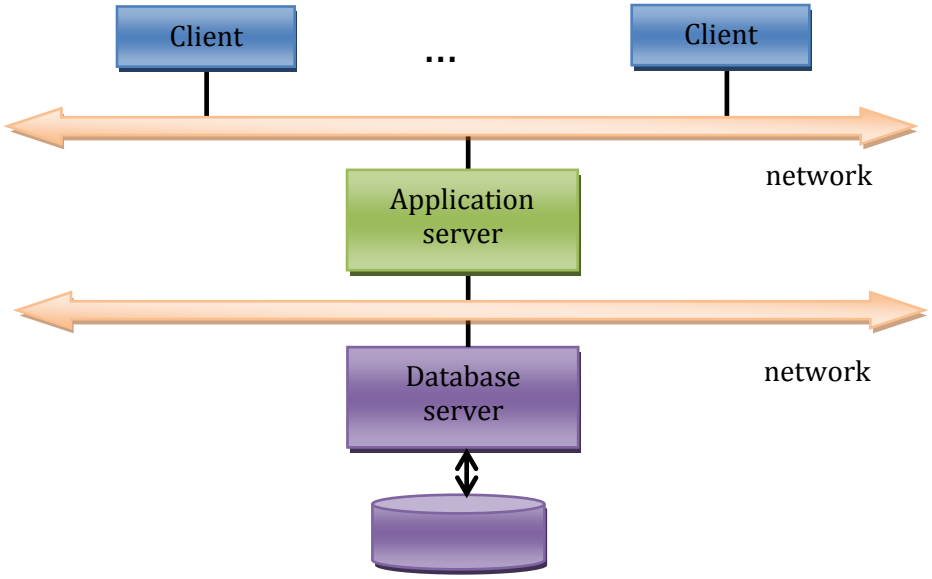

Figura 4. Database Server Approach

En la Figura 4 muestra un simple punto de vista de esta aproximación de la arquitectura cliente/servidor con servidores de aplicaciones conectados a un único servidor de datos a través de una red de comunicación.

Esta aproximación del servidor de aplicación puede ser extendida al introducir múltiples servidores de datos y servidores de aplicaciones como se puede ver en la Figura 5 donde cada servidor de aplicaciones es dedicado para una o más aplicaciones, mientras que el servidor de datos opera en múltiples servidores.

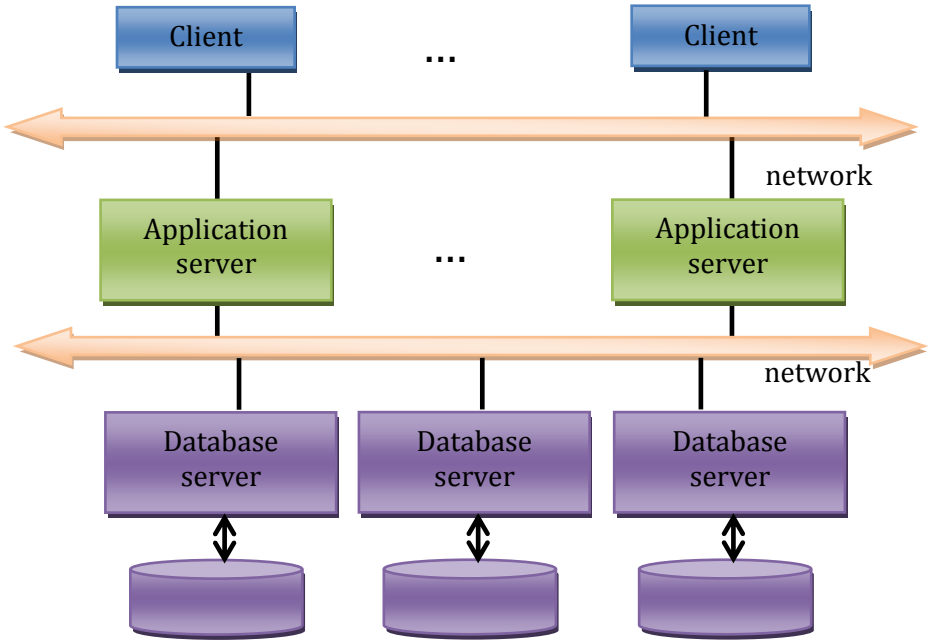

Figura 5. Distributed Database Servers

2. Sistemas Punto a punto (peer-to-peer)

En una arquitectura punto a punto no existe diferencia en la funcionalidad de cada componente, es decir, entre los servidores y los clientes, en la Figura 6 se describe con detalle los elementos de esta arquitectura.

La Figura 6 muestra que la organización física de cada máquina puede ser diferente, esto significa que es necesario tener un esquema interno por cada sitio el cual es llamado Esquema Interno Local (*local internal schema* -LIS-); el punto de vista global de los datos es descrita por el Esquema Conceptual Global (*global conceptual schema* –GCS-), este es global debido a que describe la estructura lógica de los datos de todos los sitios.

Para manejar la replicación y fragmentación de los datos es necesario conocer la organización lógica de los datos en cada sitio por lo que es requerida una tercera capa en esta arquitectura llamada Esquema Conceptual Local (*local conceptual schema* –LCS-).

En la arquitectura descrita debemos tomar en cuenta que el Esquema Conceptual Global (GCS) es la unión de todos los Esquemas Conceptuales Locales (LCS); finalmente, las aplicaciones de los usuarios y el acceso a las bases de datos es soportado por los Esquemas Externos (*External Schemas* -ESs-) definidas arriba de la capa del GCS.

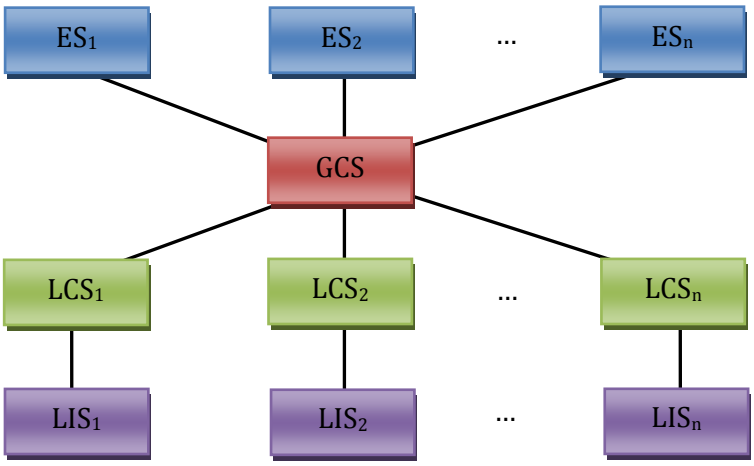

Figura 6. Arquitectura de referencia de Bases de datos distribuida

En un sistema de bases de datos distribuidas las consultas globales son direccionadas a un conjunto de consultas locales las cuales serán ejecutadas por los componentes del SGBD existentes en cada sitio para comunicarse entre ellos.

Los componentes del SGBD es mostrada en la Figura 7, donde uno de los principales componentes maneja la interacción con los usuarios y el otro será responsable del almacenamiento.

El primer componente le llamaremos Procesador del usuario (user processor), el cual consiste de cuatro elementos:

1. Manejador de interfaces de usuario (*User interface Handler*), es el responsable de interpretar los comandos del usuario y dar el formato a los datos resultantes que son enviados al usuario.
2. Controlador de datos semánticos (*smantic user controller*), usa las restricciones de integridad y autorizaciones que son definidas por el esquema conceptual global para verificar si una consulta de usuario puede ser procesada, también es el responsable de las funciones de autorización y otras funciones.
3. Optimizador de consulta global (*global query optimizar and decomposer*), determina la estrategia de ejecución y minimiza los costos, traduce la consulta global en casa subconsulta que será ejecutada en casa sitio local empleando tanto los esquemas locales como el esquema global. El optimizador de consulta global es el responsable entre otras cosas, generar la mejor estrategia de ejecución de operaciones distribuidas.

4. Monitor de ejecución distribuido (*distributed execution monitor*), coordina la ejecución distribuida de una solicitud del usuario, también es llamada administrador de transacciones distribuidas (*distributed transaction manager*).

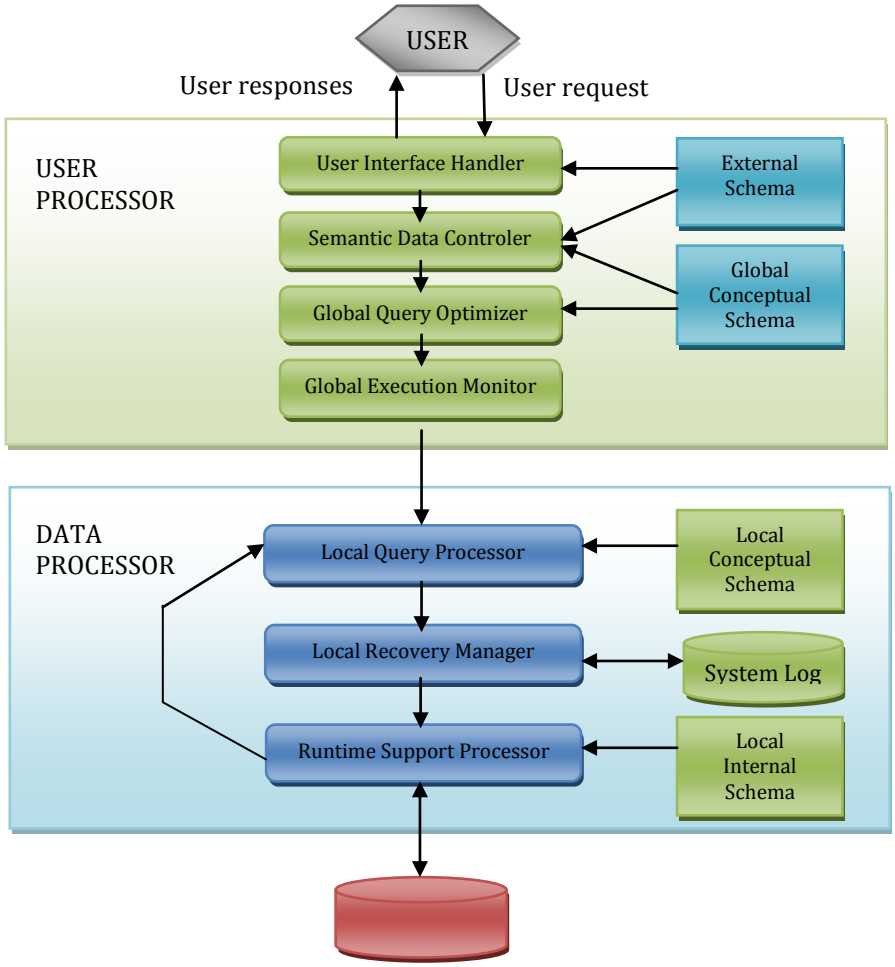

Figura 7. Componentes de un SGBD distribuida

El segundo componente de un SGBD distribuido es el procesador de datos (*data processor*) el cual consiste de tres elementos:

1. Optimizador de consultas locales (*local query optimizer*), el cual actua como el selector de camino de acceso (*access path selector*), el cual es el responsable de elegir la mejor vía de acceso a los datos.
2. Gestor de recuperación local (*local recovery manager*), el cual es el responsable de asegurar que la base de datos local quede de forma consistente cuando un fallo ocurre.
3. Procesador de soporte en tiempo de ejecución (*run-time support processor*), físicamente accede a la base de datos con base a los comandos definidos por el optimizador de consultas, este procesador es la interface del sistema operativo y el contenedor del buffer de la base de datos (*database buffer -or cache- manager*) o el gestor de memoria, el cual es responsable para mantener la memoria principal y el acceso a los datos.

3. Arquitectura de un Sistema Multibase de datos

Los sistemas multibase de datos (*Multidatabase System –MDBS-*) representan el caso donde cada SGBD de forma individual (si es distribuido o no) son completamente autónomos y no tienen un concepto de cooperación, ellos no tienen conocimiento de la existencia de otro SGBD o el poderse comunicar entre ellos; en la mayoría de la literatura encontraras la referencia a sistema de integración de datos (*data integration system*) Figura 8.

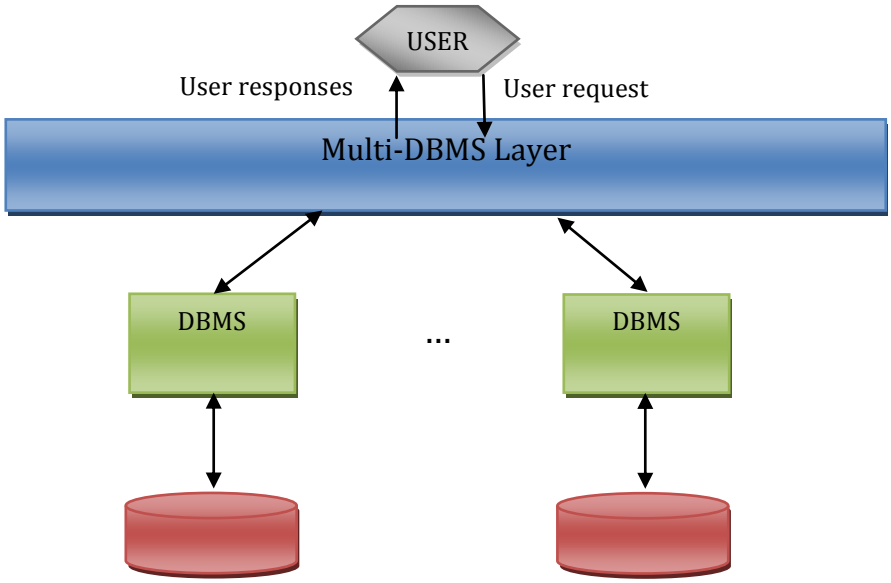

Figura 8. Componentes de un MDBS

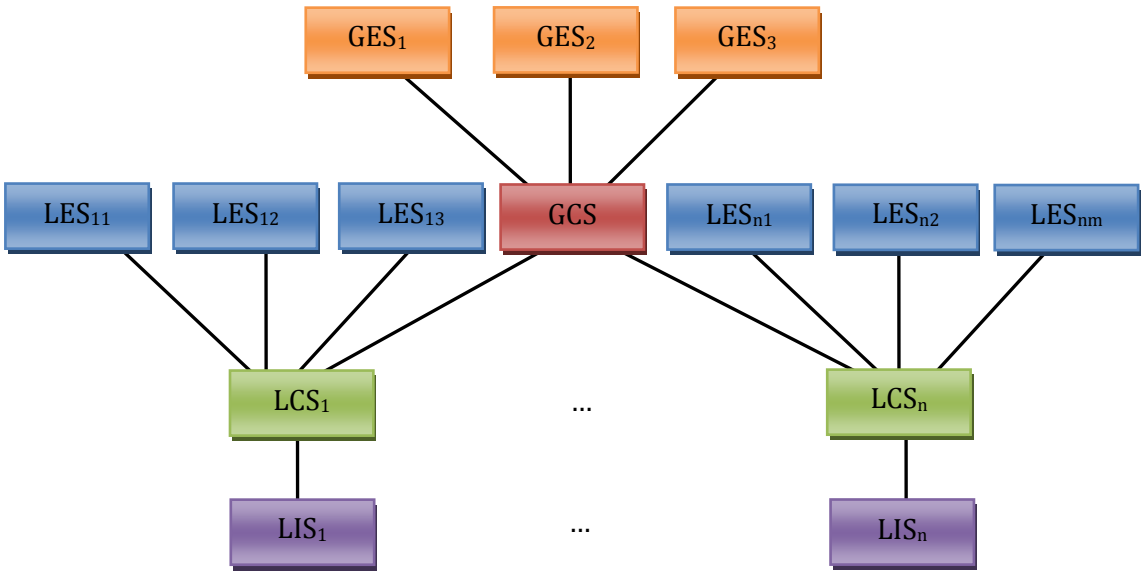

Figura 9. Arquitectura Multibase con un GCS

Las diferencias en el nivel de autonomía entre los sistemas multibase y los sistemas de bases distribuidas son reflejados en su arquitectura, esta diferencia radica en la definición del esquema conceptual global (GCS), para el caso de los sistemas distribuidos el GCS define una vista conceptual de la base de datos completamente (*entire database*), mientras que en un sistema multibase, el GCS se representa solo por la colección de algunas bases de datos (*some of the local database*) locales que cada SGBD quiere compartir.

Los SGBD individuales pueden seleccionar que datos pueden disponer para ser accedidos al definir un esquema de exportación (*export schema*); asi que la definición de una base de datos global (*global database*) difiere para un sistema multibase que para un sistema distribuido, para un sistema distribuido, una base de datos global es igual a la unión de todas las bases de datos locales o probablemente un subjconjunto de la misma unión. En un sistema multibase, el GCS es definido al integrar los esquemas externos de las bases de datos locales y autónomas o (posiblemente parte de ellas) los esquemas conceptuales locales.

4. Sistemas de Bases de datos Distribuidas (paralelas)

En la arquitectura de bases de datos paralelas hay procesadores múltiples que controlan unidades de disco múltiples que contienen a la base de datos, que puede estar particionada en los discos, o tal vez duplicada. Si la tolerancia a las fallas tiene gran prioridad, el sistema se prepara de modo que cada componente sirva como respaldo para los demás componentes del mismo tipo y se haga cargo de las funciones de cualquier componente similar que falle. Las arquitecturas de sistemas de bases de datos paralelas son de **memoria compartida**, **disco compartido, nada compartido**, o **jerárquicas**, que también se llaman **cluster**.

En un sistema de **memoria compartida**, todos los procesadores tienen acceso a la misma memoria y a los discos compartidos, como se ilustra en la Figura 10. La base de datos reside en los discos, ya sea que esté duplicada en ellos o particionada entre todos. Cuando un procesador hace una solicitud de datos, éstos son enviados desde cualquiera de los discos hacia los búferes de la memoria compartidos por todos los procesadores. El SGBD informa al procesador cuál página de la memoria contiene la página de datos solicitada.

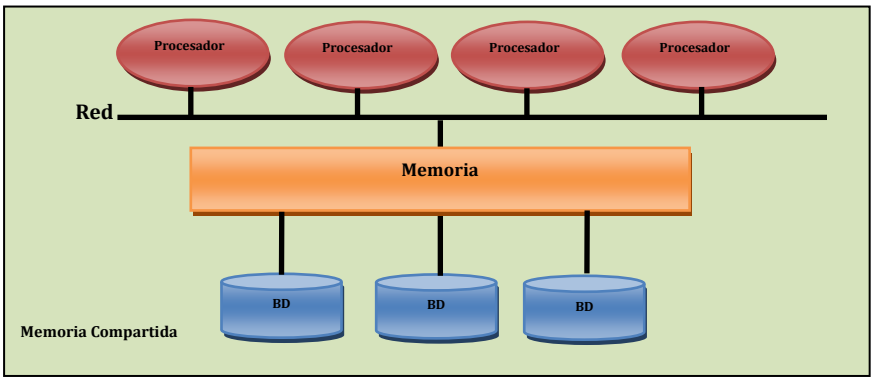

Figura 10. Sistema de Memoria Compartida

En el diseño de **disco compartido**, que se muestra en la Figura 11, cada procesador tiene acceso exclusivo a su propia memoria, pero todos los procesadores tienen acceso a las unidades de disco compartidas. Cuando un procesador solicita datos, las páginas respectivas son llevadas a la memoria del procesador.

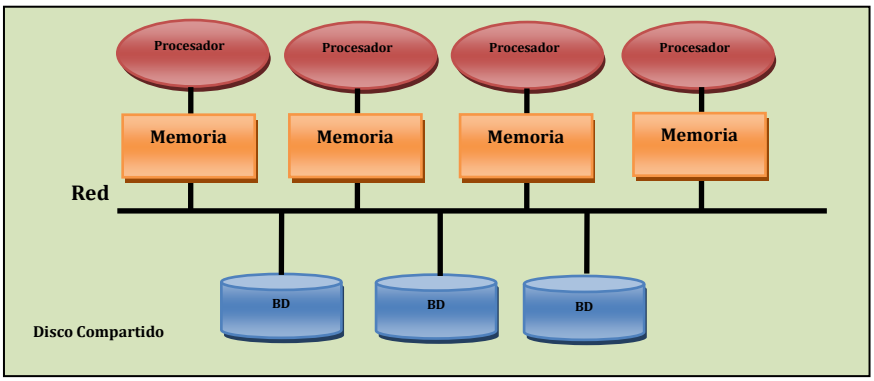

Figura 11. Diseño de disco compartido

En los sistemas de **nada compartido**, cada procesador tiene control exclusivo de su propia unidad o unidades de disco y de su memoria, como se aprecia en la Figura 12, pero los procesadores se comunican uno con otro.

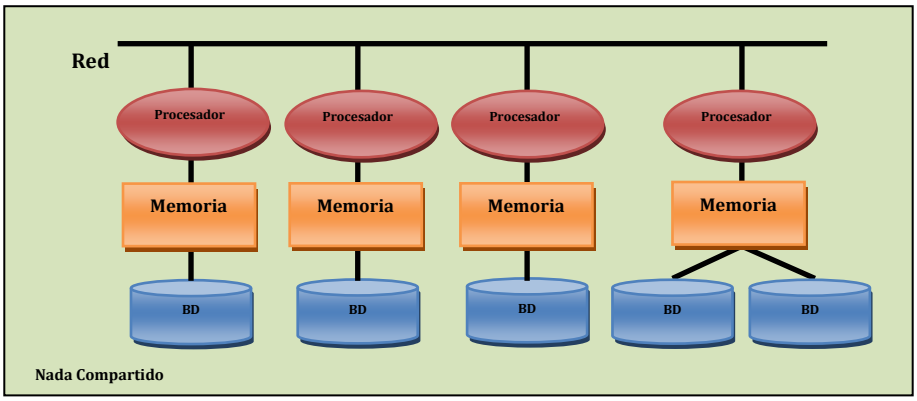

Figura 12. Sistemas de nada compartido

En la arquitectura **jerárquica** o **cluster**, los sistemas constituidos por nodos de memoria compartida están conectados por medio de una red, como se ve en la Figura 13. Los sistemas sólo comparten comunicaciones entre sí, y las arquitecturas general entre sistemas no comparten nada.

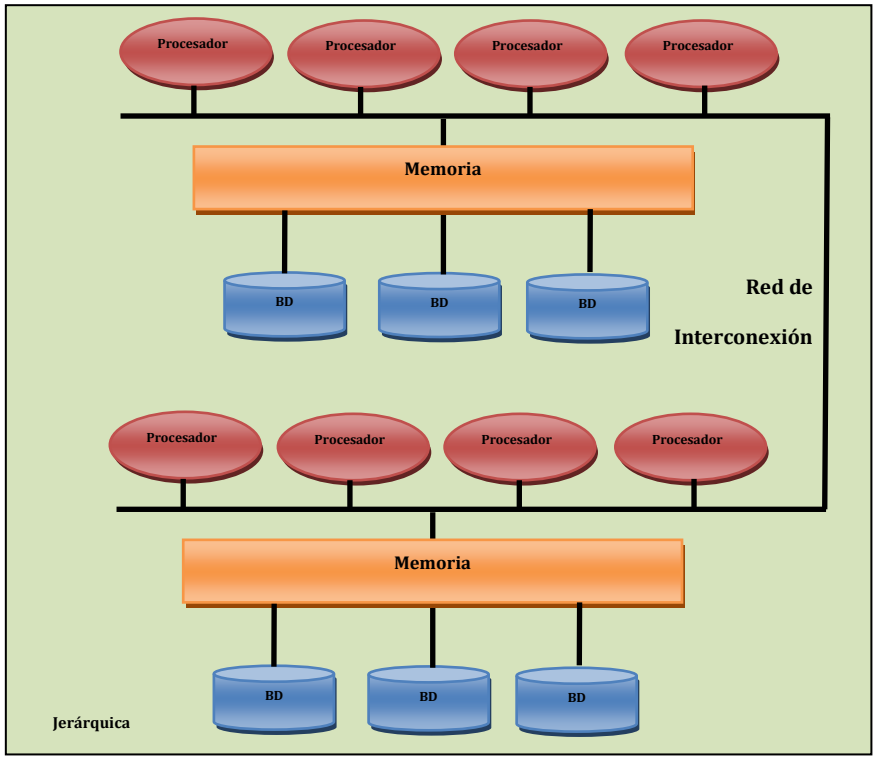

Figura 13. Arquitectura jerárquica o cluster

Arquitectura de referencia para una Base de datos Distribuida [3]

a) Esta arquitectura permite determinar fácilmente los distintos niveles de transparencia en la distribución.

b) Estos niveles son relevantes para entender la organización de cualquier sistema de BD Distribuido.

c) Arquitectura que no se encuentra implementado tal como se indican en los sistemas de BD Distribuidas.

Los componentes de esta arquitecta se muestra en la siguiente figura (Ver Figura 14).

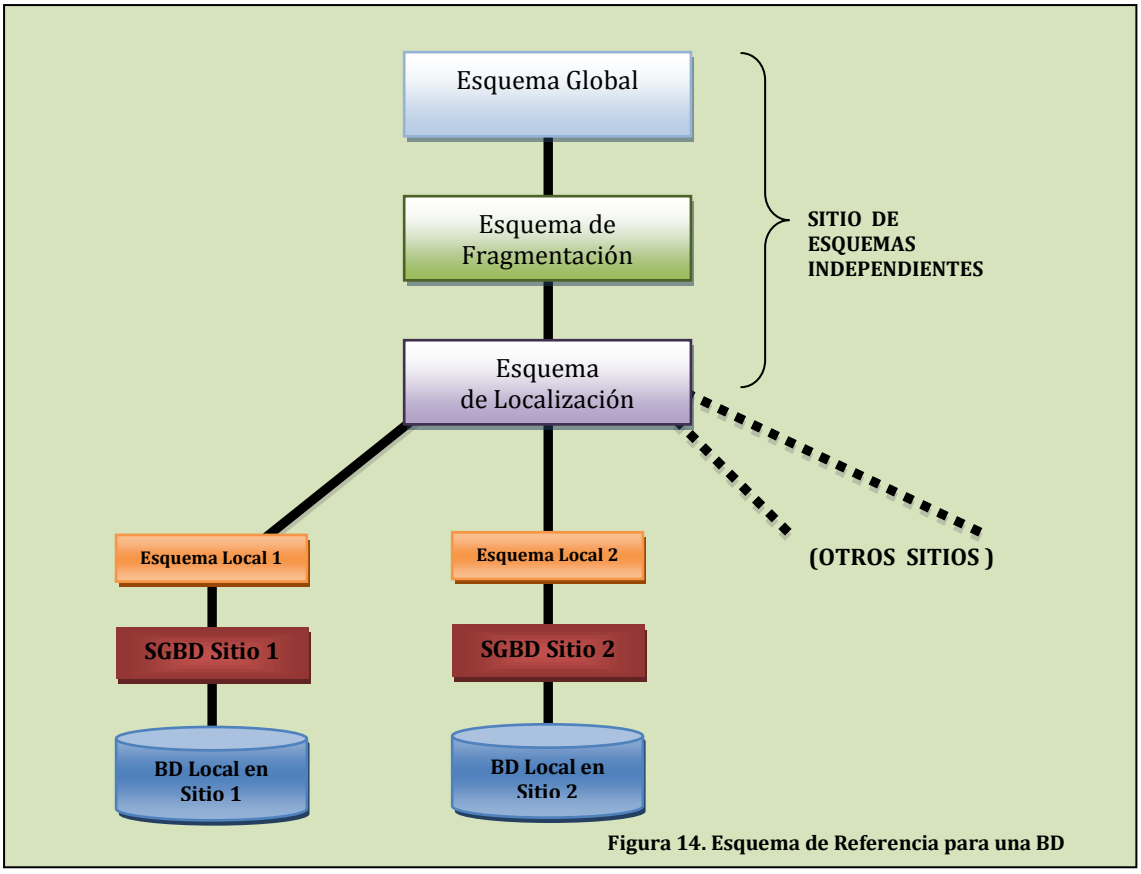

Figura 14. Esquema de Referencia para una BD

Esquema Global.

c) Define los datos que contendrá un sistema de BDD como si la BD no fuera distribuida.
d) Se define de la misma manera que una BD.
e) El modelos de datos usado para definir el esquema global debería ser el conveniente a los sistemas de BD. (Modelo Relacional) -> define un conjunto de relaciones globales.
f) Cada relación global puede ser dividido en diversas porciones llamados Fragmentos.

Esquema de Fragmentación

- Es el mapeo entre las relaciones globales y los fragmentos.
- El mapeo es de uno a muchos, es decir, muchos fragmentos corresponden a una relación global (R), pero una sola relación global corresponde a un fragmento.
- Los fragmentos son indicados por el nombre de una relación global con un índice (índice de fragmento).

  $R_i \Rightarrow i-ésimo$ fragmento de la relación global R.

  Fragmento: Son porciones lógicas de las relaciones globales, los cuales están físicamente localizados en uno o diferentes sitios de la red.

Esquema de localización.

g) Define el sitio donde un fragmento es localizado.
h) El tipo de mapeo definido en el esquema de localización determina si la BDD
   - a) Es redundante (mapeo de 1 a muchos)
   - b) No redundante (mapeo de uno a uno).

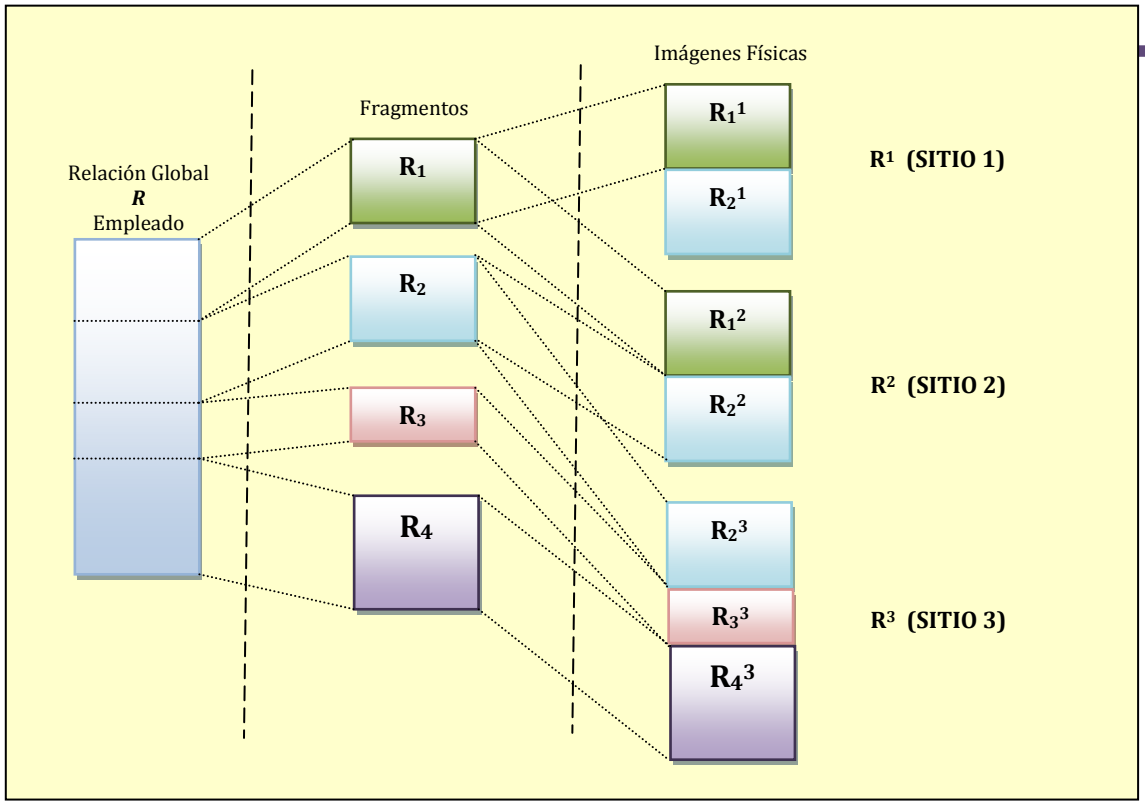

Figura 15. Fragmentos e imágenes físicas para una relación global.

i) Todos los fragmentos corresponden a la misma relación global R y son localizados en el mismo sitio J que sustituyen la imagen física de la relación global R en el sitio J.
j) La imagen física puede ser indicada por el nombre de la relación global y el índice del sitio.

$R^j \Rightarrow$ imagen física de la relación global R en el sitio J.

Ejemplo:

Una relación global R que esta dividido en 4 fragmentos: R_1, R_2, R_3 y R_4, que están localizados redundantemente en 3 sitios en una res de computadoras, los cuales constituyen las 3 imágenes físicas: R^1, R^2 y R^3.

k) La copia de un fragmento en un sitio determinado, es denotado por el nombre de una relación global y dos índices (índice del fragmento y el índice del sitio).

Ejemplo:

$R_3^{\ 2} \Rightarrow$ La copia del fragmento 3 se realiza en el sitio 2.

l) Dos imágenes físicas pueden ser idénticas; en este caso diremos que una imagen física tiene una copia de otra imagen física.

Ejemplo:

R^1 es una copia de R^2

Características de la arquitectura de referencia.

1. Separación del concepto de fragmentación de datos del concepto de localización de datos.

   Esta separación nos permite distinguir dos diferentes niveles de transparencia en la distribución: *transparencia de fragmentación* y *transparencia de localización*.

   La transparencia de fragmentación es el grado más alto de transparencia y consiste en el hecho de que el usuario o los programas de aplicación trabajan sobre relaciones globales.

   La transparencia de localización es el grado mas bajo de transparencia y requiere que el usuario o los programas de aplicación trabajen con los fragmentos en vez de relaciones globales sin conocer donde se encuentran localizados.

2. Control de concurrencia explícito.

   Esta arquitectura proporciona explícitamente del control de concurrencia a nivel fragmento. Por ejemplo en la Figura 8, dos imágenes físicas R^2 y R^3 son traslapadas debido a que contienen datos en común. La definición de fragmentos disjuntos como construcción de bloques de imágenes físicas nos permite referirnos explícitamente a este traslape: replicar el fragmento R_2.

3. Independencia en los SGBD locales.
   Esta característica llamada *transparencia en el mapeo local* nos permite considerar diversos problemas de administración en bases de datos distribuidas sin tomar en cuenta los modelos de datos de cada SGBD local. Claramente, en un sistema homogéneo es posible definir con el mismo modelo de datos los esquemas independientes para los SGBD locales, reduciendo la complejidad de dicho mapeo.