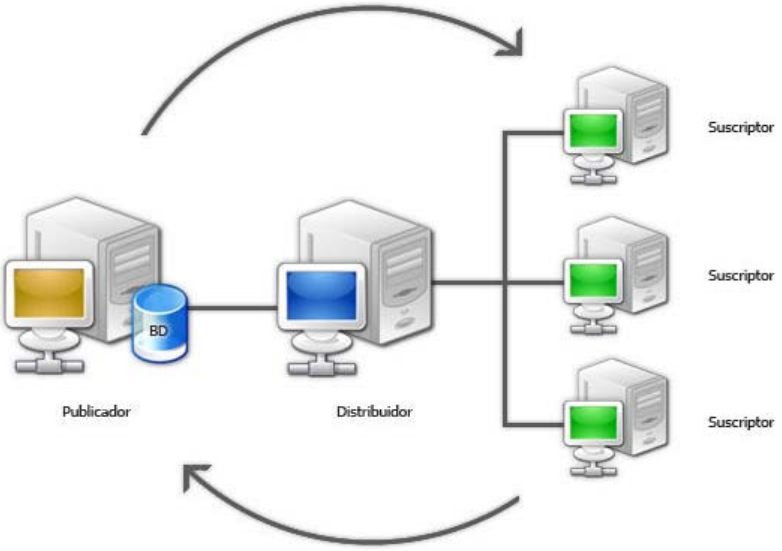
4

ARQUITECTURA DE DISTRIBUCIÓN DE DATOS



Contenido:

CAPITULO IV

1. ARQUITECTURA DE DISTRIBUCIÓN DE DATOS

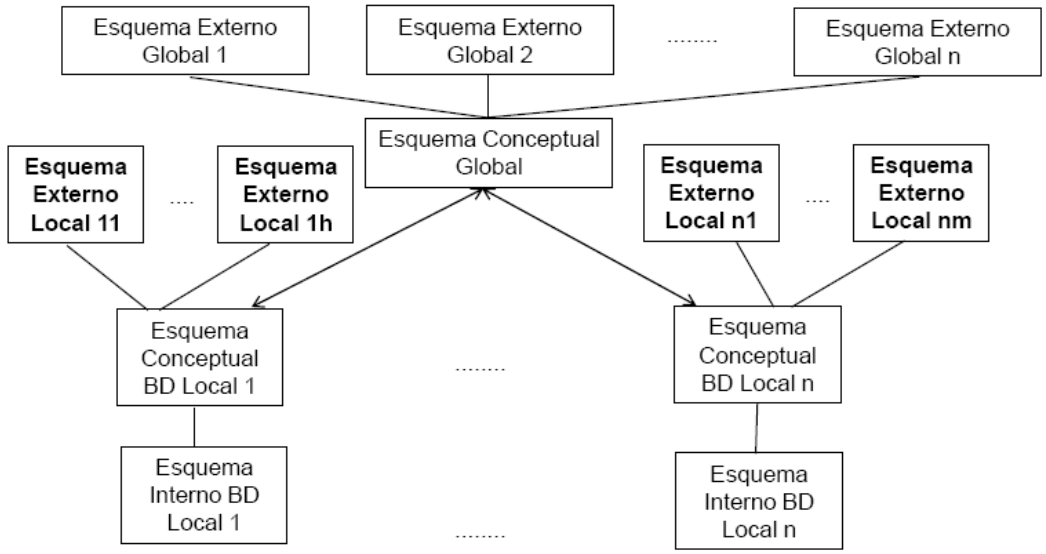
ARQUITECTURA DE UNA BASE DE DATOS DISTRIBUIDA

Se debe tomar en cuenta varios factores para la definición de la arquitectura de un sistema:

1. **Distribución**: Los componentes del sistema están localizados en la misma computadora o en diferente computador.
2. **Heterogeneidad**: Es cuando existen en él componentes que se ejecutan en diversos sistemas operativos.
3. **Autonomía**: Se puede presentar en diferentes niveles, como son:

    - **Autonomía de diseño**: Está relacionadas a su propio diseño.
    - **Autonomía de comunicación**: Es cómo y cuándo comunicarse con otros SMBD.
    - **Autonomía de ejecución**: Ejecutar operaciones locales como quiera.

ARQUITECTURA ANSI/SPARC



1. Arquitectura ANSI/X3/SPARC para BDD [IMAG.10][1]

La arquitectura ANSI / SPARC se divide en 3 niveles:

1. **EL NIVEL INTERNO.** Es el que se ocupa de la forma como se almacenan físicamente los datos.
2. **EL NIVEL EXTERNO.** Es el que se ocupa de la forma como los usuarios individuales perciben los datos.
3. **EL NIVEL CONCEPTUAL.** Es un nivel de mediación entre los otros dos, es decir define las estructuras de almacenamientos el Administrador de Base de Datos.

No existe un equivalente de una arquitectura estándar para sistemas de manejo de bases de datos distribuidas, cada sistema ha adoptado su propia arquitectura.

[1] http://www.kybele.etsii.urjc.es/docencia/DBD/2009-2010/Material/%5BDBD-2009-10%5DBD_Distribuidas.pdf

Se debe definir un modelo de referencia para un esquema de estandarización en bases de datos distribuidas, cuyo propósito es dividir el trabajo en piezas y esas piezas se relacionan unas con otras. Se sigue los siguientes enfoques:

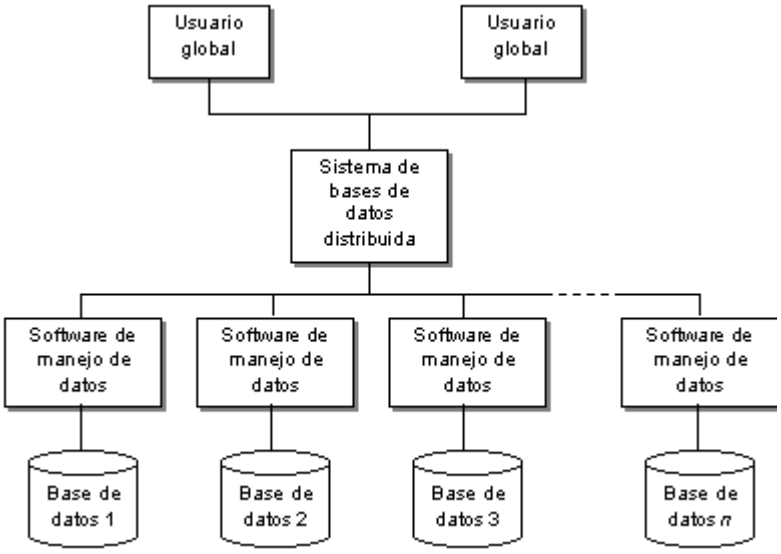
1. **Basado en componentes**. Se definen las componentes del sistema junto con las relaciones entre ellas.
2. **Basado en funciones**. Se identifican las diferentes clases de usuarios junto con la funcionalidad que el sistema ofrecerá para cada clase.
3. **Basado en datos**. Se identifican los diferentes tipos de descripción de datos y se especifica un marco de trabajo arquitectural el cual define las unidades funcionales que realizarán y/o usarán los datos de acuerdo con las diferentes vistas. Este es el enfoque seguido por el modelo ANSI/SPARC.

ARQUITECTURA DE UN SISTEMA MANEJADOR DE BASES DE DATOS DISTRIBUIDAS

Los sistemas de datos distribuidos están divididos en dos clases:

1. Sistemas de manejo de bases de datos distribuidos homogéneos
2. Sistemas de manejo de bases de datos distribuidos heterogéneos

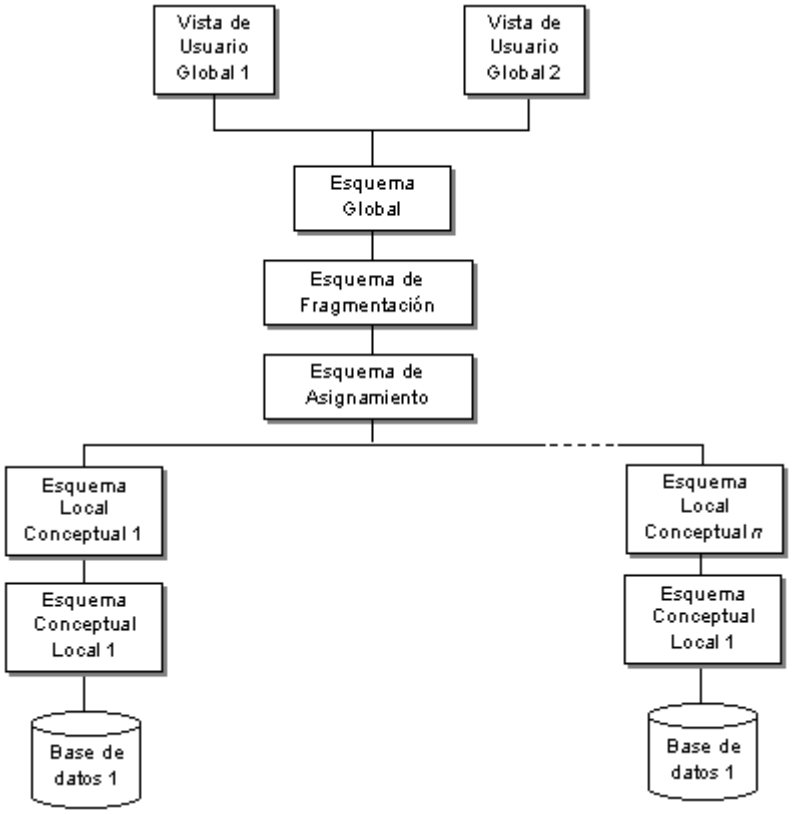
ARQUITECTURA DE UN SISTEMA DE MANEJO DE BASES DE DATOS DISTRIBUIDOS HOMOGÉNEOS



2. Arquitectura de un SMBDD homogéneo.

Los sistemas homogéneos se parece a un sistema centralizado, a diferencia que estos sus datos se distribuyen en varios sitios comunicados por la red. No existen usuarios locales y todos ellos accesan la base de datos a través de una interfaz global.

Para manejar los aspectos de la distribución, se deben agregar dos niveles a la arquitectura estándar ANSI-SPARC, de la siguiente manera, como se muestra en la Figura

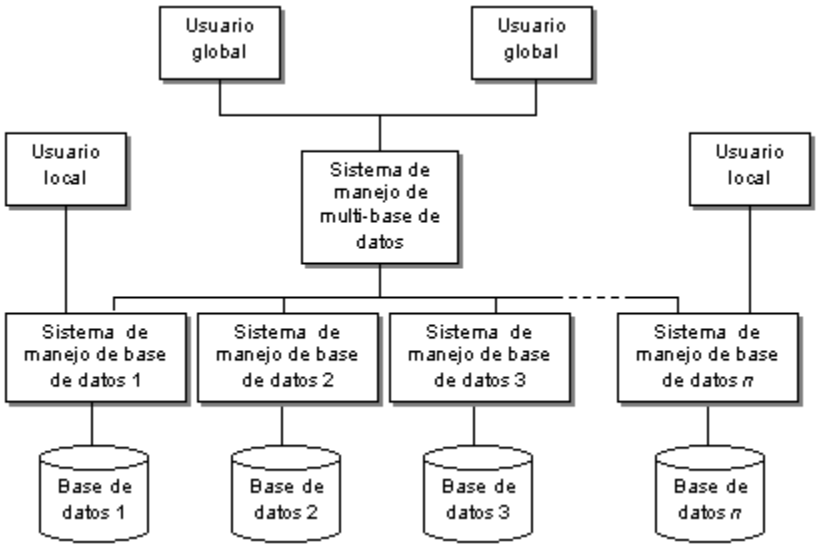


3. Arquitectura de los esquemas de un SMBDD homogéneo

El esquema de fragmentación describe la forma en que las relaciones globales se dividen entre las bases de datos locales.

El esquema de asignamiento especifica el lugar en el cual cada fragmento es almacenado. De aquí, los fragmentos pueden migrar de un sitio a otro en respuesta a cambios en los patrones de acceso.

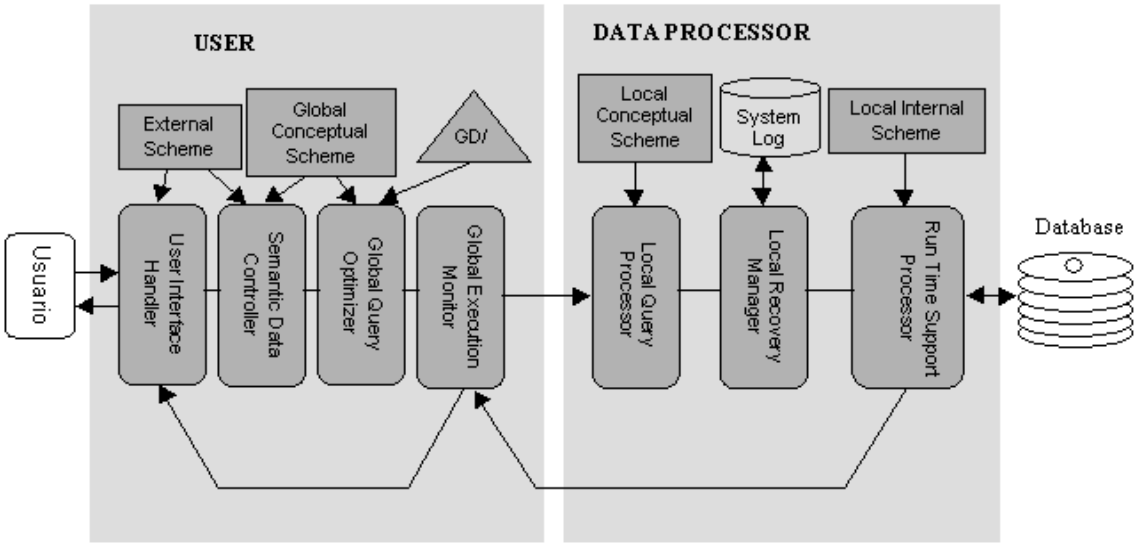
ARQUITECTURA DE UN SISTEMA DE MANEJO DE BASES DE DATOS DISTRIBUIDOS HETEROGÉNEOS



4. Arquitectura de un sistema multi-bases de datos.

Un sistema multi-bases de datos tiene múltiples SMBDs, que pueden ser de tipos diferentes, y múltiples bases de datos existentes. Existen usuarios locales y globales.

ARQUITECTURA BASADA EN COMPONENTES DE UN SISTEMA DE MANEJO DE BASES DE DATOS DISTRIBUIDOS



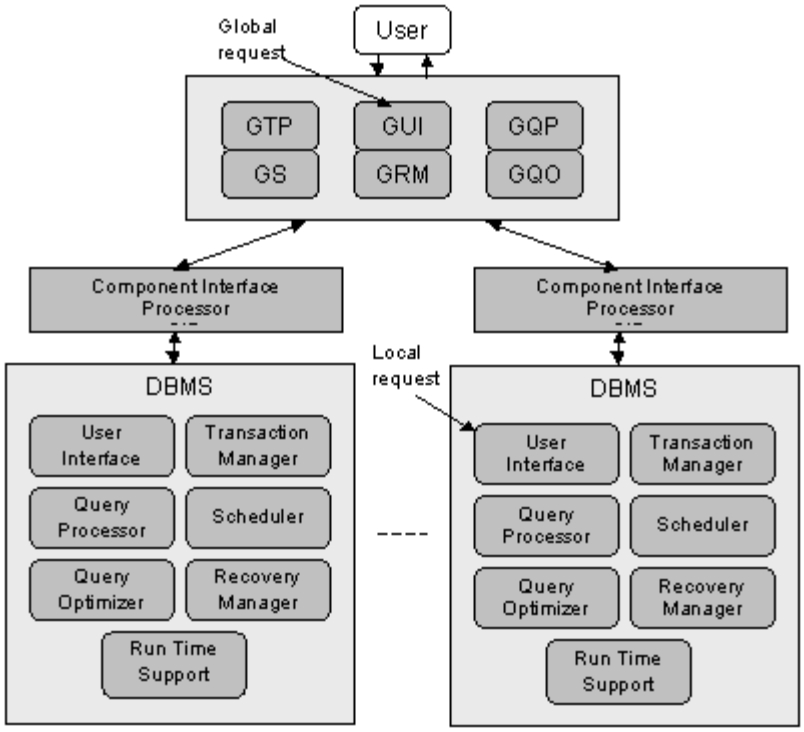
5. Arquitectura basada en componentes de un SMBD distribuido

Consiste en dos partes como son: el procesador de datos y el procesador de usuario.

- El procesador de usuario es el encargado de procesar las solicitudes del usuario, consiste en cuatro partes: un manejador de la interfaz con el usuario, un controlador semántico de datos, un optimizador global de consultas y un supervisor de la ejecución global.
- El procesador de datos existe en cada nodo de la base de datos distribuida. Utiliza un esquema local conceptual y un esquema local interno, el procesador de datos consiste de tres partes: un procesador de consultas locales, un

manejador de recuperación de fallas locales y un procesador de soporte para tiempo de ejecución.

ARQUITECTURA BASADA EN COMPONENTES DE UN SISTEMA MULTI-BASES DE DATOS.



6. Arquitectura basada en componentes sistema multi-bases de datos.

Consta de un sistema de manejo de bases datos para usuarios globales y usuarios locales. Para comunicar el sistema global con los sistemas locales se define una interfaz común entre componentes mediante la cual, las operaciones globales se convierten en una o varias acciones locales.

1.1. TRANSPARENCIA

Se refiere a la división del nivel semántico y la implementación del sistema.

El objetivo es ocultar al usuario los detalles de diseño, es decir, el usuario no tiene que saber que se encuentra trabajando sobre un sistema distribuido, por ejemplo la independencia de los datos es una forma de transparencia

El propósito fundamental de la transparencia en el ambiente distribuido es la independencia de datos.

La transparencia la podemos encontrar en:

- ✓ Manejo de la red de comunicación.
- ✓ Manejo de copias repetidas
- ✓ En la distribución o fragmentación de la información.

INDEPENDENCIA DE DATOS

Es la inmunidad de las aplicaciones de usuarios a los cambios en la definición y organización de los datos y viceversa. La independencia de datos se puede darse en dos aspectos: lógico y físico. **[www.08]**[2]

1. **Independencia lógica de datos**. Se refiere a la inmunidad de las aplicaciones de usuario a los cambios en la estructura lógica de la base de datos. Esto permite que un cambio en la

[2] http://basedatosavanzadaselem.obolog.com/unidad-1-base-datos-distribuidas-523685

definición de un esquema no debe afectar a las aplicaciones de usuario. Por ejemplo, el agregar un nuevo atributo a una relación, la creación de una nueva relación, el reordenamiento lógico de algunos atributos. **[www.09]**[3]

2. **Independencia física de datos**. Se refiere al ocultamiento de los detalles sobre las estructuras de almacenamiento a las aplicaciones de usuario. Esto es, la descripción física de datos puede cambiar sin afectar a las aplicaciones de usuario. Por ejemplo, los datos pueden ser movidos de un disco a otro, o la organización de los datos puede cambiar. **[www.10]**[4]

NIVELES DE TRANSPARENCIA.

El propósito de establecer una arquitectura de un sistema de bases de datos distribuidas es ofrecer un nivel de transparencia adecuado para el manejo de la información.

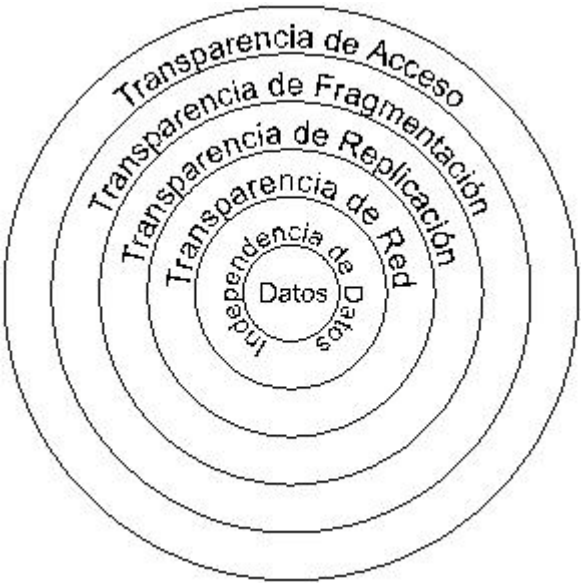
Primer Nivel. Se soporta la transparencia de red.

Segundo Nivel. Se permite la transparencia de replicación de datos.

Tercer Nivel. Se permite la transparencia de la fragmentación.

Cuarto Nivel. Se permite transparencia de acceso (por medio de un lenguaje de manipulación de datos)

[3] http://basedatosavanzadaselem.obolog.com/unidad-1-base-datos-distribuidas-523685
[4] http://basedatosavanzadaselem.obolog.com/unidad-1-base-datos-distribuidas-523685



7. Organización en Capas de niveles de Transparencia. [IMAG.16][5]

1.1.1. TRANSPARENCIA DE LOCALIZACIÓN.

La trasparencia de localización le permite acceder al usuario a los datos sin tener en cuenta la ubicación de estos, es decir debe ser transparente al usuario, ya q este no necesita saber dónde está el dato para utilizarlo. Se consigue cuando los administradores de transacciones distribuidas pueden determinar la localización de los datos y emitir acciones a los administradores apropiados, esto se puede ejecutar cuando los administradores de transacciones tienen acceso a los directorios de localizaciones de los datos.

Los administradores de transacciones necesitan conocer si los datos cambian de lugar, ya que las transacciones ignoran la modificación en la localización.

[5] http://usuarios.multimania.es/admin2master/documentos/descargas/ppt/BASES_DE_DATOS_DISTRIBUIDAS.PPT

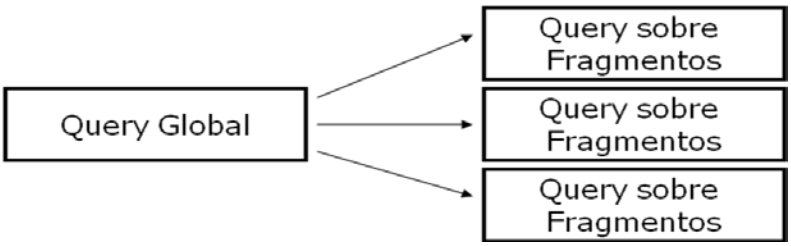
1.1.2. TRANSPARENCIA DE FRAGMENTACIÓN

El acceso a una base de datos distribuida debe hacerse en forma transparente. Los usuarios deben comportarse, como si los datos en realidad no estuvieran fragmentados, lo cual es necesario por razones de rendimiento.

El programador no necesita saber si la base de datos es distribuida o está fragmentada.

Los accesos a la base de datos es de forma global, es decir el usuario no necesita especificar los nombres de los fragmentos ni la ubicaciones de los datos.

El sistema maneja la conversión de consultas de usuario definidas sobre relaciones globales a consultas definidas sobre fragmentos. Las respuestas a consultas fragmentadas para obtener una sola respuesta a una consulta global. **[www.11]**[6]



8. Muestra la Transparencia de Fragmentación [IMAG.17][7]

[6] http://www.google.com.ec/url?sa=t&source=web&cd=4&ved=0CB8QFjAD&url=http%3A%2F%2Fdircompucv.ciens.ucv.ve%2Fgenerador%2Fsites%2Fbases-de-datos-dist%2Farchivos%2FBDD%2520-%2520Clase%25202.pps&rct=j&q=%BFQue+es+independencia+de+datos%3F++Es+la+inmunidad+de+las+aplicaciones+de+usuarios+a+los+cambios+en+la+definici%F3n+y+organizaci%F3n+de+los+datos+y+viceversa.&ei=00YmTKXaI8KAlAeq7P2lAw&usg=AFQjCNH8JW6hh0LX5fMedB9AwofatAvO-A

[7] http://www.google.com.ec/url?sa=t&source=web&cd=4&ved=0CB8QFjAD&url=http%3A%2F%2Fdircompucv.ciens.ucv.ve%2Fgenerador%2Fsites%2Fbases-de-datos-dist%2Farchivos%2FBDD%2520-%2520Clase%25202.pps&rct=j&q=%BFQue+es+independencia+de+datos%3F++Es+la+inmunidad+de+las+aplicaciones+

1.1.3. TRANSPARENCIA DE REPLICA

La transparencia sobre replicación de datos se refiere a que si existen réplicas de objetos de la base de datos, esta debe ser controlada por el SMBDD, más no por el usuario.

El usuario no necesita saber sobre la replicación de los datos, la función principal de la transparencia de replicación es la de mantener la consistencia entre las copias, esta funciona en forma transparente a las aplicaciones.

La replicación puede mejorar el funcionamiento y proteger la disponibilidad de las aplicaciones, porque alterna opciones de acceso de los datos existente.

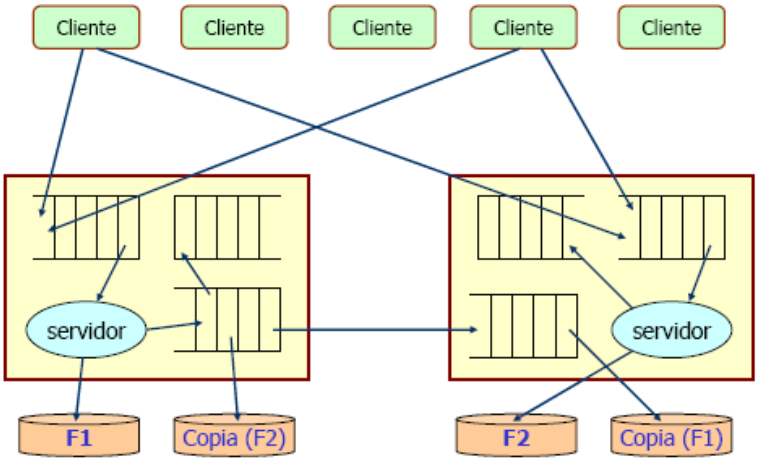
Existe una copia principal y varias copias secundarias, las que se extienden a lo largo de las modificaciones en forma asíncrona. Existen dos tipos de propagación de modificaciones: **[www.11]**[8]

- ✓ Incremental: La información que se envía desde la copia principal a las secundarias son las variaciones en los datos.
- ✓ Total: Se envía toda la copia principal completa.

Su principal utilidad es hacer que el sistema sea menos sensible a los fallos, ya que si la copia principal no está disponible se puede seguir usando alguna de las copias secundarias.

de+usuarios+a+los+cambios+en+la+definici%F3n+y+organizaci%F3n+de+los+datos+y+viceversa.&ei=00YmTKXaI8KAlAeq7P2lAw&usg=AFQjCNH8JW6hh0LX5fMedB9AwofatAvO-A

[8] http://alarcos.inf-cr.uclm.es/doc/aplicabbdd/Documentos/teoria/arquitecturas%20para%20bases%20de%20datos.pdf



9. Principal utilidad de la Transparencia de Replicación [IMAG.18][9]

La replicación es necesaria por las siguientes razones:

- ✓ Mayor rendimiento, debido a que se dispone de copias locales.
- ✓ Mayor disponibilidad, ya que los datos son accesibles siempre al tenerse varias copias.

La principal desventaja, es que hay que mantener actualizadas todas las copias de ese objeto o dato replicado. Esto nos lleva al problema de la "propagación de las actualizaciones".

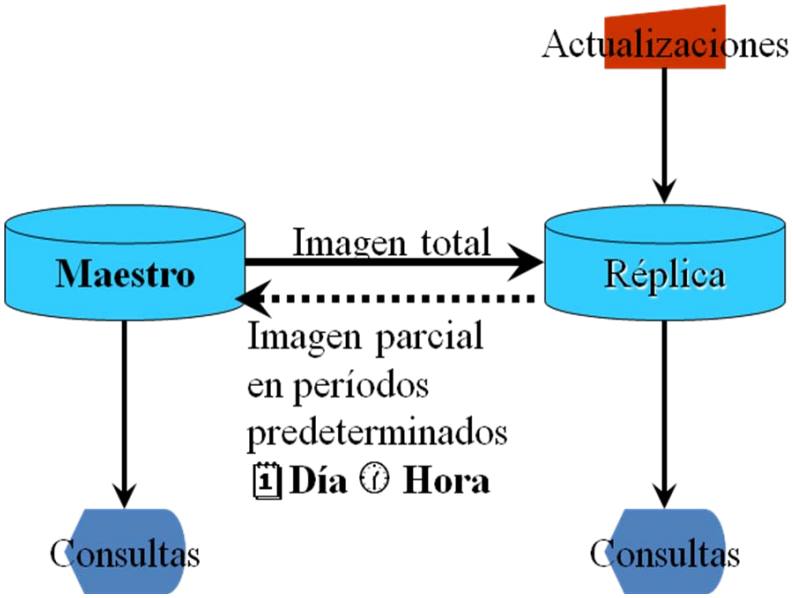
[9] http://alarcos.inf-cr.uclm.es/doc/aplicabbdd/Documentos/teoria/arquitecturas%20para%20bases%20de%20datos.pdf

1.2. FORMAS DE RÉPLICAS

Las réplicas pueden asumir los siguientes formatos:

- ✓ Periódica
- ✓ Continua
- ✓ Check-in / Check-out

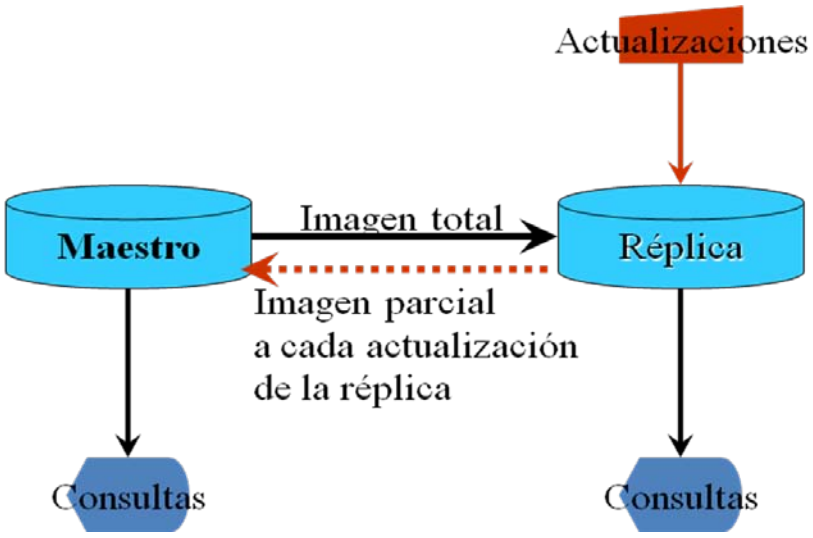
RÉPLICA PERIÓDICA



10. Replica Periódica [IMAG.19][10]

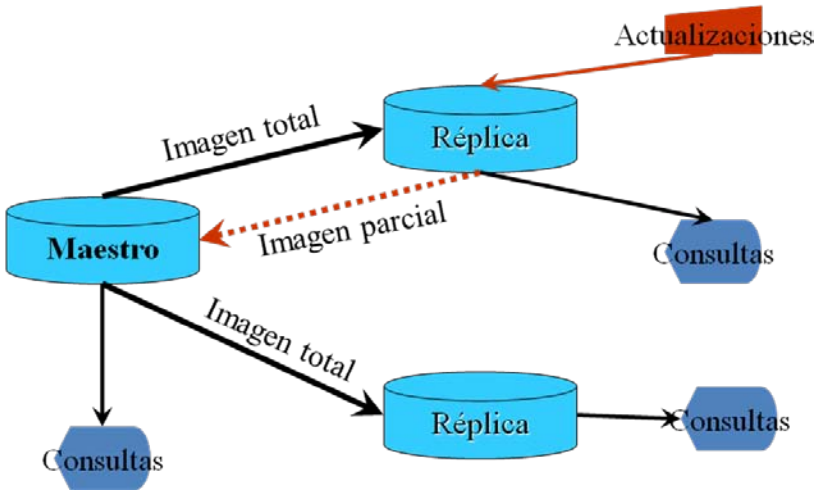
[10] http://www.cyta.com.ar/elearn/bd/curso_archivos/bddistribuida.ppt

RÉPLICA CONTINUA



11. Replica Continua [IMAG.20][11]

RÉPLICA CHECK - IN/OUT



12. Replica Check – IN/OUT [IMAG.21][12]

[11] http://www.cyta.com.ar/elearn/bd/curso_archivos/bddistribuida.ppt
[12] http://www.cyta.com.ar/elearn/bd/curso_archivos/bddistribuida.ppt

CREACIÓN DE RÉPLICAS

Es posible crear réplicas completas o parciales:

Réplica completa: Contiene todos los documentos y elementos de diseño de la base de datos como son: formularios y agentes.

Réplica parcial: Contiene documentos abreviados, resúmenes de documentos o documentos cuyos campos contienen información especificada, documentos preseleccionados, documentos de carpetas o vistas especificadas.

Este tipo de réplica le resultará útil cuando sólo necesite una parte de la base de datos y quiere ahorrar espacio en el disco.

NOTA: Las réplicas que no contienen documentos se denominan réplicas vacías. Si intenta abrir una réplica vacía, se mostrará un mensaje diciendo que la base de datos no está inicializada y debe replicarse.

CARACTERÍSTICAS

VENTAJAS

- ✓ Disponibilidad frente a fallos de la red.
- ✓ Paralelismo, las peticiones se pueden procesar en varios nodos en paralelo.
- ✓ Transferencia de datos reducida

INCONVENIENTES

- ✓ Se eleva el coste de la actualizaciones
- ✓ Se complica el control de la concurrencia