



UNIVERSIDAD
DE GRANADA



Administración de Bases de Datos

Grado en Ingeniería Informática

Tema 6 – Sistemas distribuidos de Bases de Datos



I. J. Blanco, A. G. López Herrera

**Departamento de Ciencias de la
Computación e Inteligencia Artificial**
<http://decsai.ugr.es>

- Tradicionalmente, un procesador y varios dispositivos de almacenamiento.
- La *distribución* del proceso se debe a:
 - funcionalidad
 - geografía
 - autonomía
 - fiabilidad
 - escalabilidad



- Un SGBDD se compone de *nodos (localidades)*:
 - el nodo es un SGBD
 - los nodos pueden trabajar aisladamente o intercambiar datos
- *Base de datos distribuida* = virtual = lógica
- Tipos: *homogéneo, heterogéneo*.
- Transparencia.

- Características
- Transparencia
- Diseño
- Recuperación
- Control de concurrencia



Para un usuario debe ser igual trabajar contra un sistema distribuido que contra un sistema aislado



- Conceptos:
 - **Sección posterior:** SGBD
 - **Sección frontal:** aplicaciones (en el SGBD o fuera)
- **Administrador de Comunicaciones de Datos (DC)**

- Procesamiento distribuido:
 - un caso: cliente/servidor
 - aplicaciones en un servidor, SGBD en otro (mejores tiempos)
 - servidor exclusivo para SGBD (mejor rendimiento)
 - cliente exclusivo para aplicaciones (mejor adaptado)
 - varios clientes acceden al mismo servidor
 - varios servidores tras un mismo cliente
 - usuarios de un nodo pueden acceder a datos de otro nodo

- Procesamiento distribuido:
 - un caso: cliente/servidor
 - aplicaciones en un servidor, SGBD en otro (mejores tiempos)
 - servidor exclusivo para SGBD (mejor rendimiento)
 - cliente exclusivo para aplicaciones (mejor adaptado)
 - varios clientes acceden al mismo servidor
 - **varios servidores tras un mismo cliente**
 - usuarios de un nodo pueden acceder a datos de otro nodo

- Ventajas:
 - **uso compartido de datos** (departamentos, divisiones, ...)
 - **fiabilidad y disponibilidad**: aislamiento ante fallos, redundancia (requiere detección de fallos y reintegración)
 - **Agilización de consulta**: dividir consultas en subconsultas para distintos nodos

- Desventajas:
 - Coste de desarrollo de software
 - Mayor probabilidad de error
 - Mayor tiempo de procesamiento
 - Tiempo de la red de comunicación



- Es lo mismo trabajar contra un SGBD que contra un SGBDD.
- Niveles:
 - red
 - nombres de datos
 - copia y fragmentación
 - localización de fragmentos y copias
 - actualizaciones



- Ocultar los detalles de la distribución de la información en la red.
- Concepto relacionado (opuesto): *autonomía local*



- Cada elemento, un solo nombre.
- Una posibilidad, un *asignador de nombres centralizado*:
 - cuello de botella,
 - sensible a caídas,
 - poca autonomía local
- Otra posibilidad, *prefijo de nodo*:
 - mayor autonomía local, menor transparencia de red

- La replica es beneficiosa:
 - lectura sobre datos locales,
 - objeto accesible si una de sus copias lo está
- Transparente al usuario en el acceso (implica actualización de todas las copias)

- Fragmentación:
 - Horizontal (a nivel de instancia)
 - Vertical (a nivel de esquema)
- *Fragmento*: sub-relación obtenida mediante proyección y/o selección
- Es necesaria una nomenclatura unificada, por ejemplo:

nodo<refnodo>.<relacion>.f<idfrag>.r<idcopia>

- La nomenclatura unificada viola la transparencia de red.
- Se usan *alias* simples que el sistema traduce a nomenclatura unificada.
- Requiere de un *esquema de asignación de nombres*.

- Todas las copias deben actualizarse.
- El problema está relacionado con el problema de actualización de vistas.



- *Catálogo global*: contiene los metadatos sobre la distribución.
- Una posibilidad: en un solo nodo. ¡Vulnerable!
- Otra posibilidad: copias en todos los nodos. ¡Poco eficiente!
- Solución: catálogo distribuido (cada nodo tiene información de sus objetos, las copias y los fragmentos)

- Autonomía local
- Independencia de nodo central
- Operación continua
- Independencia con respecto a la localización
- Independencia con respecto a la fragmentación
- Independencia de réplica
- Procesamiento distribuido de consultas

- N1.EMP, 5000 registros, 80 B / registro
- N2.DEPTO, 30 registros, 40 B / registro
- Nodo N3 ejecuta la operación
`SELECT #emp, EmpName, DeptoName FROM EMP e,
DEPTO d WHERE e.#dpto=d.#dpto`
 - EMP y DEPTO a N3: 401200 B transferidos
 - EMP a N2 (400000 B), resolver allí y enviar a N3 el resultado (200000 B)
 - DPTO a N1 (1200 B), resolver allí y enviar a N3 el resultado (200000 B)

- Autonomía local
- Independencia de nodo central
- Operación continua
- Independencia con respecto a la localización
- Independencia con respecto a la fragmentación
- Independencia de réplica
- Procesamiento distribuido de consultas
- **Manejo distribuido de transacciones**

- Autonomía local
- Independencia de nodo central
- Operación continua
- Independencia con respecto a la localización
- Independencia con respecto a la fragmentación
- Independencia de réplica
- Procesamiento distribuido de consultas
- Manejo distribuido de transacciones
- **Independencia con respecto al equipo**



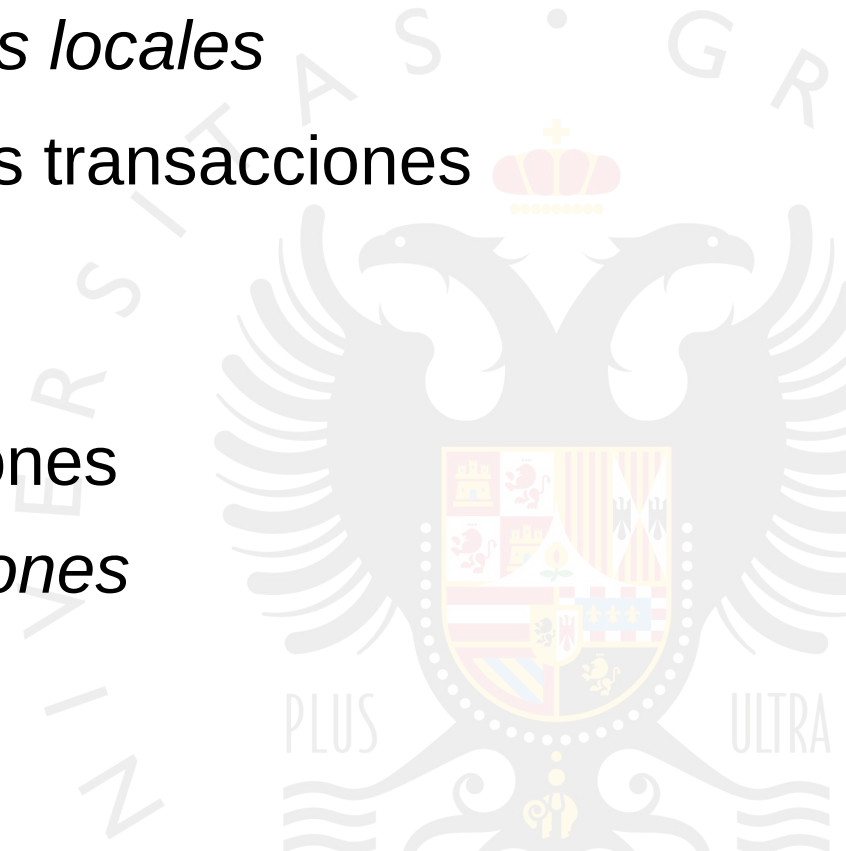
- ...
- Independencia con respecto del S. O.
- Independencia con respecto de la red
- ...



- Independencia con respecto del SGBD:
 - diversos SGBD entienden el mismo estándar de lenguaje pero permiten variaciones:
 - la traducción la realiza una *compuerta* (programa que hace que un SGBD actúe como otro distinto para permitir la conexión)
 - si un usuario accede por un SGBD concreto, la transparencia garantiza que verá toda la BDD como se ve en su SGBD de entrada
 - es responsabilidad de su SGBD proveer de la compuerta

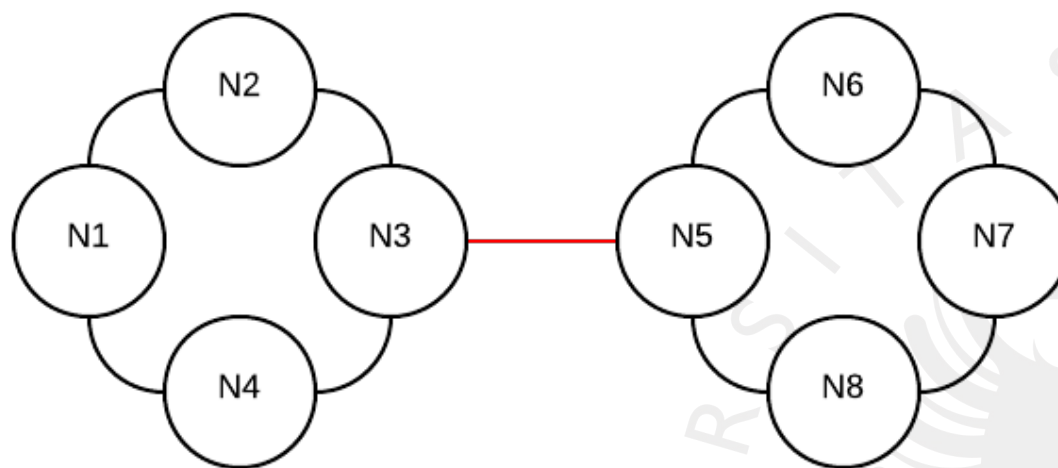
- Debe incluir protocolos para el intercambio de información con el otro SGBD (de manejo y de resultados)
- Actuar como servidor para el otro SGBD
- Proveer correspondencia de tipos
- Proveer correspondencia de dialectos SQL
- Proveer correspondencia de *feedback*
- Proveer correspondencia de diccionarios
- Proveer de protocolos de dos fases para las transacciones compatible
- Proveer de bloqueo de datos en el SGBD de destino

- Dos gestores:
 - *Gestor de transacciones distribuidas*
 - *Gestores de transacciones locales*
- Todos colaboran para que las transacciones globales sean correctas.
- Cada nodo tiene:
 - Gestor local de transacciones
 - *Coordinador de transacciones*



- Los fallos del SGBDD en su totalidad son los mismos que los de un SGBD además de:
 - fallo total en una localidad (nodo)
 - interrupción de línea de comunicación
 - pérdida de mensajes
 - fragmentación de red
- La detección temprana es esencial.
- La reintegración debe ser automática

- ¿Qué hacer cuando un nodo no responde?
 - Si contiene información repetida, actualizar el catálogo y eliminar referencias
 - Si había transacciones activas en el nodo, abortarlas (prevenir bloqueos)
 - Si el nodo era un distribuidor (nombre, copias, concurrencia), designar otro nodo para sustituirle



- Cuando se ejecutan transacciones distribuidas es necesario el compromiso (ejecutada o abortada para todos)
- Dos tipos de protocolo:
 - En dos fases
 - En tres fases



- T transacción en el nodo N_i con coordinador C_i
- Fase 1:
 - C_i añade “Preparar T” al log y graba
 - C_i envía “Preparar T” a todas las localidades N_j con agente
 - ...

- Fase 1:
 - ...
 - Si el gestor de transacciones de N_j :
 - no puede cumplir la sub-transacción, anota “No T” en su log y envía mensaje “Abortar T” a C_i
 - puede cumplir la sub-transacción, anota “T lista” en su log y graba y envía mensaje “T lista” a C_i

- Fase 2:
 - Cuando los N_j han respondido al “Preparar T” de C_i o después de un intervalo de tiempo, C_i decide:
 - si recibe el “T lista” de todos los N_j , se anota “ejecutar T” en el log y se graba, y se ejecuta
 - si no recibe algún “T lista”, se anota “abortar T” en el log y se graba
 - ...

- Fase 2:
 - ...
 - A continuación, se envía la decisión a todos los N_j



- Gestión de fallos:
 - en nodo participante: cuando se recupera, debe revisar todos sus agentes interrumpidos
 - si hay un “Ejecutar T”, realiza un *redo(T)*
 - si hay un “Abortar T”, realiza un *undo(T)*
 - si no hay entradas sobre T, la transacción fue abortada por no responder a tiempo a “preparar T”
 - ...

- Gestión de fallos:
 - en nodo participante: cuando se recupera, debe revisar todos sus agentes interrumpidos
 - ...
 - hay un registro “T lista”, debe consultar al coordinador Ci para ver el estado final de T:
 - si Ci está activo, le responderá sobre el estado
 - si Ci no está activo, preguntará a los otros nodos

- Gestión de fallos:
 - en una línea de comunicación: un nodo sólo puede interpretar que ha ocurrido lo anterior de modo que se aplica todo lo anterior
 - fragmentación de red:
 - si el coordinador y los nodos están en el mismo fragmento, no existe problema
 - si no están en el mismo fragmento, se pierden mensajes y se aplica todo lo anterior.

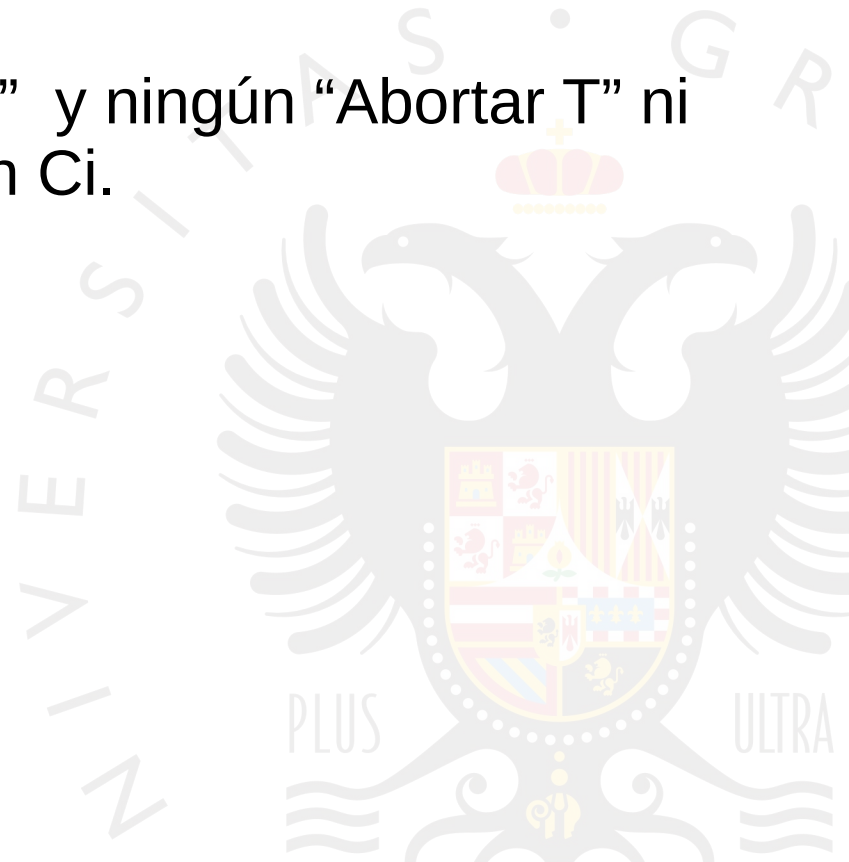
- T transacción en el nodo N_i con coordinador C_i
- Fase 1: igual a la del protocolo en dos fases
- Fase 2:
 - si C_i recibe un “Abortar T” de cualquier N_j o no le responden en un plazo, se aborta la transacción igual que en el protocolo en dos fases
 - ...

- Fase 2:
 - ...
 - si C_i recibe un “T lista” de todos los N_j , anota un “Preejecutar T” y graba, y envía un mensaje “Preejecutar T” a los N_j informando de que todos están listos
 - cada N_j que recibe el mensaje, anota “Preejecutar T”, graba y responde con “Registrado” pero espera.

- Fase 3:
 - Ci espera a recibir un número de “Registrado”, anota un “Ejecutar T”, graba, envía un “Ejecutar T” a los Ni y ejecuta
 - Nj anota “Ejecutar T”, graba y ejecuta
- Antes de enviar “T lista”, cualquier Nj puede abortar enviando el mensaje correspondiente
- El mensaje “Preejecutar T” es un compromiso del coordinador de terminar la transacción.

- Fallo en nodo participante: cuando se recupera, debe revisar todos sus agentes interrumpidos
 - si hay un “Ejecutar T”, realiza un *redo(T)*
 - si hay un “Abortar T”, realiza un *undo(T)*
 - hay un registro “T lista” y ningún “Abortar T”, debe consultar al coordinador C_i para ver el estado final de T:
 - si C_i responde con un “Preejecutar T”, el nodo continua con el protocolo donde se quedó
 - si C_i responde con un “Ejecutar T”, se realiza un *redo(T)*

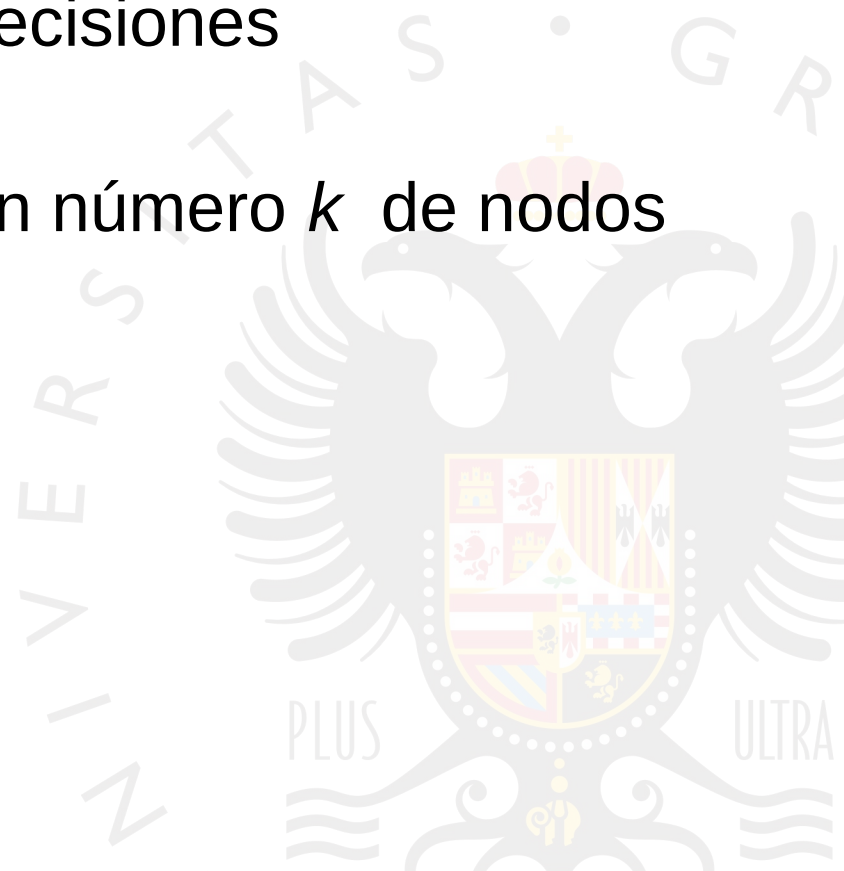
- Fallo en nodo participante: cuando se recupera, debe revisar todos sus agentes interrumpidos
 - ...
 - si N_j tiene un “Preejecutar T” y ningún “Abortar T” ni “Ejecutar T”, se consulta con C_i .



- Fallo en coordinador:
 - se escoge uno nuevo entre los participantes
 - el nuevo coordinador C_i' pregunta a los participantes por el estado local de T
 - Los estados de cada nodo son: ejecutada, abortada, lista, preejecutada, no lista

- Fallo en coordinador:
 - ...
 - Ci' decide:
 - si hay algún “Ejecutada”, se ejecuta
 - si hay algún “Abortada”, se aborta
 - si hay algún “Preejecutada”, se envían mensajes “Preejecutar T”
 - en cualquier otro caso, se aborta

- Fallo en coordinador:
 - el protocolo es incompatible con fragmentación (dos coordinadores con decisiones contradictorias)
 - el protocolo requiere de un número k de nodos activos



- Transacciones locales y distribuidas no pueden interferir entre sí.
- Dos tipos (recordatorio del tema 5):
 - Ordenar por hora de llegada
 - Mecanismo de bloqueo en dos fases

- Control de concurrencia basado en horas:
 - Problema: conflictos implican retrocesos
 - Para evitarlo, se retrasan las lecturas hasta que no provoquen problemas (combinando con el protocolo de dos fases)



- Peticiones de bloqueo
- Esquemas sin réplicas:
 - Hay un gestor de bloqueo local
 - Cuando un nodo N_i recibe una solicitud de bloqueo de un átomo a por parte de un N_j para una transacción dada en un modo, que es incompatible con los actuales, se pospone.
 - Cuando un nodo bloqueado se libera, el gestor debe comunicar a N_j

- Enfoque de coordinador único:
 - Gestiona todas las peticiones
 - Admite replicación por lo que la lectura se realiza en una de las copias, y la escritura en todas
 - Es sencillo de implementar (sólo dos mensajes) y en el manejo del bloqueo (se aplica el mismo algoritmo de bloqueo en dos fases centralizado)
 - Tiene desventajas: cuello de botella y vulnerabilidad (por un solo nodo)

- Protocolo de mayoría:
 - Modificación del esquema sin réplica
 - Cuando se desea bloquear el átomo a , debe enviar mensajes a más de la mitad de los nodos en los que se replica.
 - Cada gestor responde si puede o no garantizar.
 - Si no se garantizan todas las peticiones, se posterga.
 - Evita el problema del control central

- Protocolo de mayoría:
 - Tiene defectos:
 - Muchos más mensajes en la red
 - Alta probabilidad de interbloqueo



- Protocolo preferencial:
 - Similar al de mayoría pero con preferencia a las solicitudes de bloqueo compartido sobre las de exclusivo:
 - lectura: se solicita a algún nodo que tenga una copia
 - escritura: se solicita a todos los nodos que tengan copia

- Protocolo de copia primaria:
 - Cada conjunto de copias tiene una copia primaria en un nodo concreto.
 - Se solicita el bloque al nodo de la copia primaria
 - Para lectura, se lee la copia primaria
 - Para escritura, la copia primaria se encarga de actualizar las demás