Facultad de Ciencia Y Tecnología

Base de Datos Avanzadas

Bases de Datos Distribuidas

DOCENTES:

- TROSSERO, SEBASTIÁN
- SCHMUCKLER, JORGE

Resumen

- Introducción y Conceptos
  - Arquitecturas
  - Concepto BDD/SGBD
  - Motivaciones
  - Elementos y características
  - Distribución de datos
  - Un principio fundamental
- Reglas de Date
- Profundización conceptos Fragmentación y Replicación
- Diccionario de datos
- Procesamiento de consultas
- Transacciones

Introducción y conceptos
Arquitectura de sistemas

Arquitecturas:

- Sistemas Centralizados.
- Sistemas Cliente/Servidor.
- Sistemas Paralelos.
- **Sistemas Distribuidos**

Introducción y conceptos
Arquitectura de sistemas

Sistemas Centralizados:

Los sistemas centralizados son aquellos en los que todos los recursos, datos y procesos se encuentran en un único lugar, es decir, en un servidor central.

Las principales características de estos sistemas son:

- El servidor central es el único punto de acceso a los recursos.
- Todos los datos y aplicaciones residen en el servidor central.
- Los clientes no tienen capacidad de procesamiento, ya que toda la lógica se ejecuta en el servidor.
- La administración y mantenimiento del sistema se realiza en el servidor central.

Introducción y conceptos
Arquitectura de sistemas

Sistemas Cliente/Servidor:

Los sistemas cliente/servidor son aquellos en los que las tareas y responsabilidades se distribuyen entre un servidor y uno o varios clientes.

Las principales características de estos sistemas son:

- Los clientes pueden solicitar y recibir información del servidor, pero también pueden procesar datos y ejecutar algunas funciones localmente.
- La lógica de negocio se encuentra tanto en el servidor como en el cliente, lo que permite una mayor flexibilidad.
- Los clientes pueden trabajar sin conexión al servidor en algunos casos.
- La administración y mantenimiento se distribuye entre el servidor y los clientes.

Introducción y conceptos
Arquitectura de sistemas

Sistemas Paralelos:

Los sistemas paralelos son aquellos que utilizan múltiples procesadores para ejecutar tareas simultáneamente.

Las principales características de estos sistemas son:

- El trabajo se divide en tareas pequeñas y se asignan a diferentes procesadores.
- Los procesadores trabajan de manera simultánea para completar la tarea de forma más rápida.
- La capacidad de procesamiento se incrementa proporcionalmente al número de procesadores.
- La programación y diseño de sistemas paralelos es compleja.

Introducción y conceptos
Arquitectura de sistemas

Sistemas Distribuidos:

Los sistemas distribuidos son aquellos en los que los recursos, datos y procesos se encuentran en diferentes ubicaciones físicas y están conectados a través de una red.

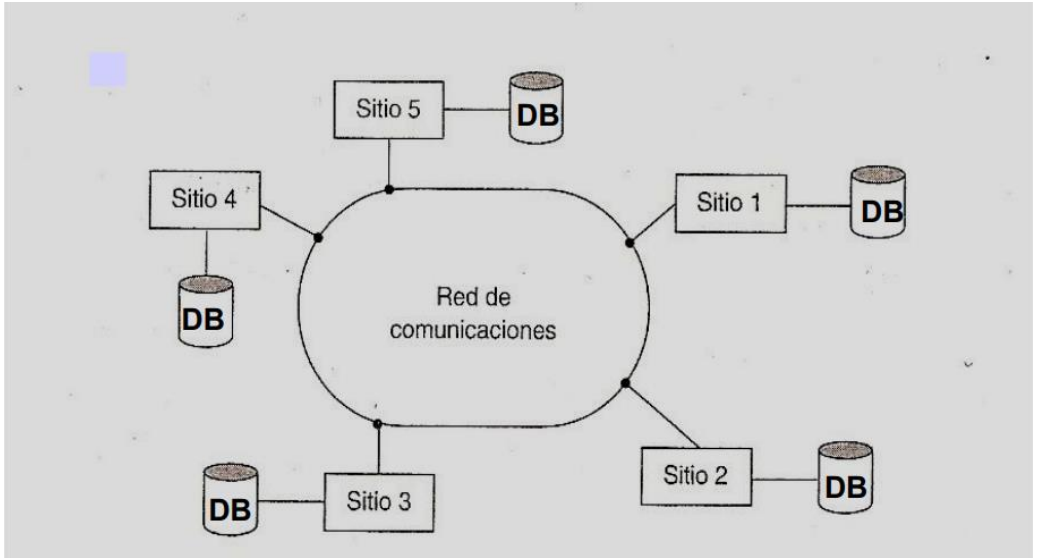
Las principales características de estos sistemas son:

- Los recursos y datos están distribuidos en diferentes lugares.
- Los procesos y tareas se ejecutan de manera concurrente en diferentes nodos de la red.
- La capacidad de procesamiento se puede ampliar mediante la adición de nuevos nodos a la red.
- La administración y mantenimiento es compleja, ya que se requiere coordinación entre los nodos de la red.

Introducción y conceptos

SGBDD

Los sistemas gestores de bases de datos distribuidas son SGBD en los que los datos están almacenados en múltiples nodos en una red, y los nodos están conectados a través de una red de comunicaciones.



Introducción y conceptos SGBDD

Las BDD aportan al dominio de la gestión de bases de datos las ventajas de la computación distribuida.

En la cual un conjunto de elementos de procesamiento (no necesariamente homogéneos) interconectados por una red cooperan en forma coordinada en la ejecución de tareas, dividiendo grandes problemas en piezas más pequeñas que se resuelven en forma coordinada

Introducción y conceptos
Motivaciones

- Se aprovecha más la potencia del ordenador en la resolución de tareas complejas. (Esta potencia se distribuye en los distintos nodos).
- Cada elemento de procesamiento se puede gestionar en forma independiente en el desarrollo de tareas locales (autonomía).
- Permite mayor escalabilidad, se pueden agregar nuevos nodos sin cambios en los actuales o bien reemplazar un nodo sin necesidad de afectar el funcionamiento del resto.

Introducción y conceptos
SGBDD – Elementos y características

- **Nodos**: son las computadoras físicas o virtuales que almacenan datos en el sistema distribuido. Cada nodo de la base de datos puede tener diferentes capacidades de almacenamiento y procesamiento, y puede estar ubicado en diferentes ubicaciones geográficas.

- **Control de transacciones distribuidas**: el SGBDD debe coordinar y controlar las transacciones que ocurren en todo el sistema distribuido. También garantizar que se cumplan las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) de las transacciones.

- **Gestión de fragmentación**: el SGBDD es responsable de dividir las tablas en fragmentos para almacenarlos en diferentes nodos de la base de datos. Además, debe garantizar que los fragmentos estén actualizados y que los cambios realizados en un nodo de la base de datos se propaguen a los demás nodos de manera consistente.

Introducción y conceptos
SGBDD – Elementos y características

- **Gestión de distribución/replicación**: Debe tener la capacidad de copiar y distribuir los datos de una tabla en múltiples nodos de la base de datos según su contenido y reglas establecidas. Además debe garantizar que las réplicas estén actualizadas y sincronizadas en todo momento

- **Control de concurrencia y bloqueos**: también debe garantizar que las transacciones se ejecuten de manera concurrente en el sistema distribuido sin generar conflictos o bloqueos.

- **Servicios de red**: estos servicios permiten la comunicación entre los nodos y permiten la transmisión de datos y mensajes entre los componentes del sistema distribuido.

Distribución de datos

Una de las decisiones que el administrador de SGBD debe tomar es como se van a distribuir los datos, más específicamente:

- ¿Cómo se van a posicionar los datos en el sistema?
- ¿Qué tipo de esquema utilizar?

Introducción y conceptos
Distribución de datos

Estrategias principales:

- Centralización
- Fragmentación
- Replicación
- Híbridas

Introducción y conceptos
Distribución de datos

Centralización:

Es el modelo clásico de Cliente/Servidor debido a que los datos están centralizados en un solo nodo, lo que se distribuye es el procesamiento y gestión de los usuarios.

En otras palabras la estrategia es no distribuir datos.

Introducción y conceptos
Distribución de datos

Fragmentación

La fragmentación consiste en dividir la relación en fragmentos menores, donde cada uno es una relación, estos fragmentos son almacenados en distintos nodos de la BDD.

Tiene como objetivo buscar alternativas para dividir las relaciones en relaciones más pequeñas.

En cada nodo se alojan uno o varios fragmentos.

Las alternativas lógicas son: por tuplas individuales (fragmentación horizontal), por atributos (fragmentación vertical) o una combinación de ambas (fragmentación híbrida)

Introducción y conceptos
Distribución de datos

Ventajas de **fragmentación**:

- Mejorar el rendimiento de las aplicaciones al trabajar con subconjuntos de relaciones.
- Dar una respuesta eficiente a aplicaciones que trabajan con los mismos datos en diferentes nodos
- Permitir el aumento del número de ejecuciones concurrentes.

Introducción y conceptos
Distribución de datos

Desventajas de **fragmentación**

- Aumento de la complejidad del SGBD.
- La comprobación de las restricciones de integridad es mas costosa.
- La implementación de transacciones es mucho mas compleja, por ende mas costosa.

Introducción y conceptos
Distribución de datos

Reglas de **Fragmentación**

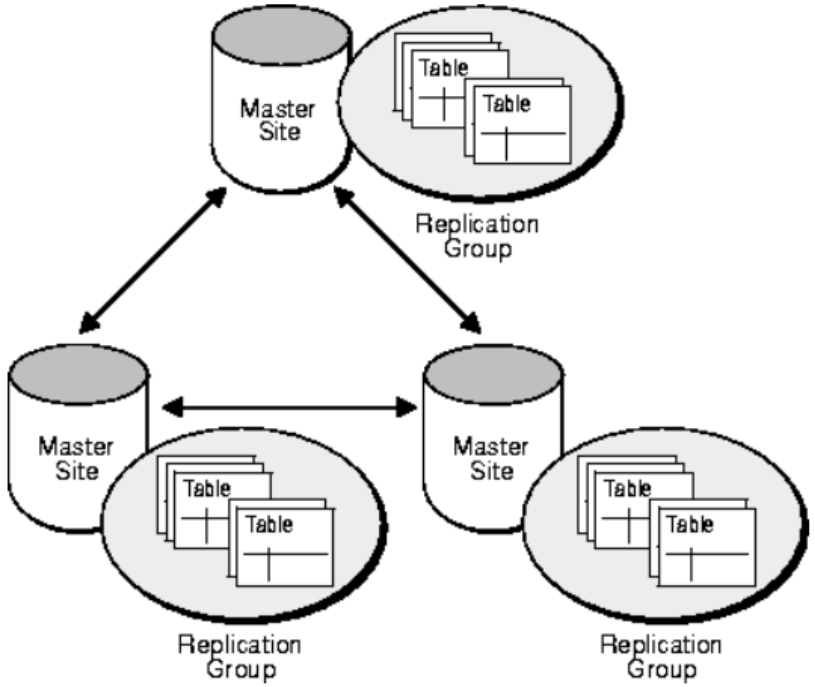
- Completitud: La descomposición de una relación R en los fragmentos R_1, R_2, ..., R_n es completa si y solamente si cada elemento de datos en R se encuentra en algún fragmento R_i.
- Reconstrucción: Si la relación R se descompone en los fragmentos R_1, R_2, ..., R_n, entonces debe existir algún operador (§) que permita reconstruir la Relación Original R.
- Disyunción: Si la relación R se descompone en los fragmentos R_1, R_2, ..., R_n, y el dato di está en R_j, entonces, no debe estar en ningún otro fragmento
  - Fragmentación Horizontal: Intersección Vacía entre los R_i
  - Fragmentación Vertical: Solo se repiten atributos claves.

Introducción y conceptos
Distribución de datos

Replicación

En este modelo, las tablas se replican en diferentes nodos, lo que significa que hay varias copias de la misma tabla almacenadas en diferentes nodos.

Esto se hace para mejorar el rendimiento y garantizar la disponibilidad de los datos.



Introducción y conceptos
Distribución de datos

La **replicación** puede ser:

- Total o completa: La replicación completa significa que la base de datos completa se replica en cada sitio del sistema distribuido. Este esquema maximiza la disponibilidad y la redundancia de los datos en una red de área amplia.
- Parcial: La replicación parcial ocurre cuando solo se replican ciertos fragmentos de la base de datos en función de la importancia de los datos en cada ubicación. Aquí, el número de copias puede variar desde uno hasta el número total de nodos en el sistema distribuido.
- Sin replicación: Define que hay una sola copia de cada elemento de la base de datos. En este caso todos los elementos son disjuntos con excepción de las PK en caso que corresponda.

Introducción y conceptos
Distribución de datos

Ventajas de replicación

- Disponibilidad: El sistema tiene mas posibilidades de seguir funcionando ante la caída de un nodo.
- Aumento de las posibilidades de paralelismo: Varios usuarios de nodos diferentes pueden efectuar consultas sobre la misma relación.
- Además cuanto más replicas existan
  - mas paralelismo (en consultas)
  - menos trafico de red (en consultas).

Introducción y conceptos
Distribución de datos

Desventajas de replicación

- Sobrecarga de las actualizaciones: El sistema debe asegurar que todas las replicas de una relación sean consistentes, es decir, se deben propagar todos los cambios a todas las réplicas. Esto conlleva:
  - Aumento de uso de discos de cada nodo
  - Aumento de consumo de red (en replicaciones)

NOTA: Parece contradictorio que el consumo de red sea ventaja y desventaja al mismo tiempo.
Tener en cuenta que se gana en consultas, pero se pierde en replicación. Se debe analizar ambos aspectos para poder decidir si este modelo es el indicado.

Introducción y conceptos
Distribución de datos

Enfoque híbrido:

El enfoque híbrido propone una combinación de los esquemas de Fragmentación y Replicación.

La estrategia es particionar determinadas relaciones y seleccionar una estrategia de replicación para cada fragmento.

Introducción y conceptos
Un principio fundamental

Ahora podemos establecer lo que puede ser considerado como el principio fundamental de la base de datos distribuida:

"Ante el usuario, un sistema distribuido debe lucir exactamente igual que un sistema que no es distribuido."

C.J.Date

En otras palabras:

Para los usuarios, solo hay una única base de datos

Introducción y conceptos
Un principio fundamental

Todos los problemas de los sistemas distribuidos son, o deberían ser, problemas internos o en el nivel de implementación, y no externos o en el nivel del usuario.

Para tener en cuenta: Cuando decimos "usuarios" nos referimos específicamente a los usuarios finales o programadores de aplicaciones que están realizando operaciones de manipulación de datos.

DML => Sin cambios

DDL => Requieren extender funcionalidades

Introducción y conceptos
Resumen ventajas y desventajas

Ventajas: podemos dividirlas en:

- Organizativas:
  - Adaptar la base de datos a la distribución geográfica de la organización y con flexibilidad a cambios.
  - Almacenar datos dónde son usados/generados.
  - Autonomía local en cada nodo
- Económicas:
  - Solución de compromiso entre Costos de Comunicación y de Creación de Pequeños Sitios (Aunque podemos considerar que los costos de comunicación están siempre presentes)
- Técnicas:
  - Mayor accesibilidad por concurrencia.
  - Mayor fiabilidad/disponibilidad: Varios sitios con réplicas => menor riesgo
  - Mejor rendimiento: BDs más chicas => operaciones de menor volumen
  - Escalabilidad

Resumen
Ventajas y Desventajas

Desventajas:

- Complejidad del software base: sincronización, algoritmos paralelos, detección de caídas, etc.
- Mayor dependencia de comunicaciones.
- Complejidad en etapa de diseño: fases adicionales, modelos de fragmentación y replicación.
- Poca “madurez” de los productos de SGBDD.
- Administración más compleja: sincronización y coordinación.
- Dificultad de cambio: nula o poca existencia de metodologías y estándares.
- Recursos humanos especializado: Tanto del lado de requerir el personal como la escasez actual de profesionales capacitados en estas tecnologías.

Resumen

- Introducción y Conceptos
- Reglas de Date
- Profundización conceptos Fragmentación y Replicación
- Diccionario de datos
- Procesamiento de consultas
- Transacciones

Reglas de Date

El principio fundamental identificado anteriormente nos conduce a doce reglas complementarias u objetivos:

1. Autonomía local.
2. No dependencia de un sitio central.
3. Operación continua.
4. Independencia de ubicación.
5. Independencia de fragmentación.
6. Independencia de replicación.
7. Procesamiento de consultas distribuidas.
8. Administración de transacciones distribuidas.
9. Independencia de hardware.
10. Independencia de sistema operativo.
11. Independencia de red.
12. Independencia de DBMS

Reglas de Date
1 Autonomía local

Los nodos o localidades de una BDD deben ser independientes entre si en el mayor grado posible.

Características de cada nodo:

- Tiene su propio DBMS
- El DBMS controla todos los aspectos del nodo
- Las operaciones de acceso a datos locales utilizan sólo recursos locales.
- La administración de los datos es local.
- Hay cooperación entre los nodos para el acceso distribuido de datos.

Reglas de Date
2 No depender de un sitio central

Todos los sitios/nodos deben ser tratados como iguales

- De existir un sitio central, habría un cuello de botella
- De existir un sitio central, el sistema sería vulnerable, porque una falla haría fallar a todo el sistema
- Para el protocolo de commit de dos fases se necesita un servicio central pero sólo durante la ejecución de una transacción (Coordinardor Lógico Centralizado).

Reglas de Date
3 Operación Continua

Un sistema BDD no debería estar nunca fuera de servicio, debe operar 7x24

- Soporte para recuperaciones rápidas de BD
- DBMS tolerante a fallos (con Software y Hardware acorde)
- Soporte para backups online (hot backup), total o incremental. (Requesito indispensable en general actualmente mas allá de BDD)

Reglas de Date
4 Independencia de Localización

Los usuarios y las aplicaciones no necesitan conocer la ubicación física de los datos. Actúan como si la BDD es una sola.

Punto crítico: el Diccionario de Datos

Usuarios y aplicaciones referencian objetos por un alias:

- El DD debe mantener una la relación de Objetos-Alias y ubicaciones.
- Un DDBMS debe mantener y utilizar el DD aún cuando los datos se mueven entre localidades
- El DD debe estar replicado en las localidades y las réplicas deben mantenerse actualizadas.

Reglas de Date
5 Independencia de Fragmentación de Datos

Los usuarios pueden comportarse como si los datos no estuvieran fragmentados.

- Usuarios no necesitan conocer los fragmentos físicos en que está dividida cada relación.
- Los datos pueden estar almacenados en la ubicación donde son usados con mayor frecuencia para que la mayoría de las operaciones sean locales y se reduzca el tráfico de la Red.

Reglas de Date
6 Independencia de Replicación de Datos

El usuario debe comportarse como si los datos no estuvieran replicados.

- Probablemente el usuario solamente pueda detectar diferencias solamente en el rendimiento al tener todos los datos en una copia local continuamente.

Reglas de Date
7 Procesamiento de Consultas Distribuidas

La performance promedio de una consulta debe ser independiente del sitio donde se realiza la consulta.

- El SGBDD debe disponer de mecanismos para optimizar las consultas y en especial para reducir la carga de tráfico necesaria

Reglas de Date
8 Gestión de Transacciones Distribuidas

El SGBDD debe disponer de mecanismos adecuados para el control de concurrencia y la recuperación de transacciones distribuidas.

- Debe mantenerse la atomicidad de las transacciones.
- Control de recuperación de información.
- Control de concurrencia.
- Protocolos utilizado para preservar la atomicidad: commit en dos o tres fases los más conocidos.

Reglas de Date
9, 10, 11 y 12 Independencias *

Tener en cuenta: “Recordar la heterogeneidad de los nodos”.

Independencia del Hardware: Es necesario tener la posibilidad de ejecutar el mismo DBMS en diferentes plataformas de Hardware. Exactamente planteado/exigido en Ansi-Sparc (nivel externo, conceptual y nivel interno) para todas las bases de datos.

Independencia del SO: Es necesario tener la posibilidad de ejecutar el mismo DBMS en diferentes Sistemas Operativos. Cada nodo puede tener un SO distinto.

Independencia de la red: El SD debe poder operar con diferentes redes de comunicaciones.

Independencia del DBMS: Garantizar que cada sitio pueda funcionar con un SGBD diferente, incluso basado en un modelo de datos diferente, siempre y cuando compartan una interface común.

Resumen

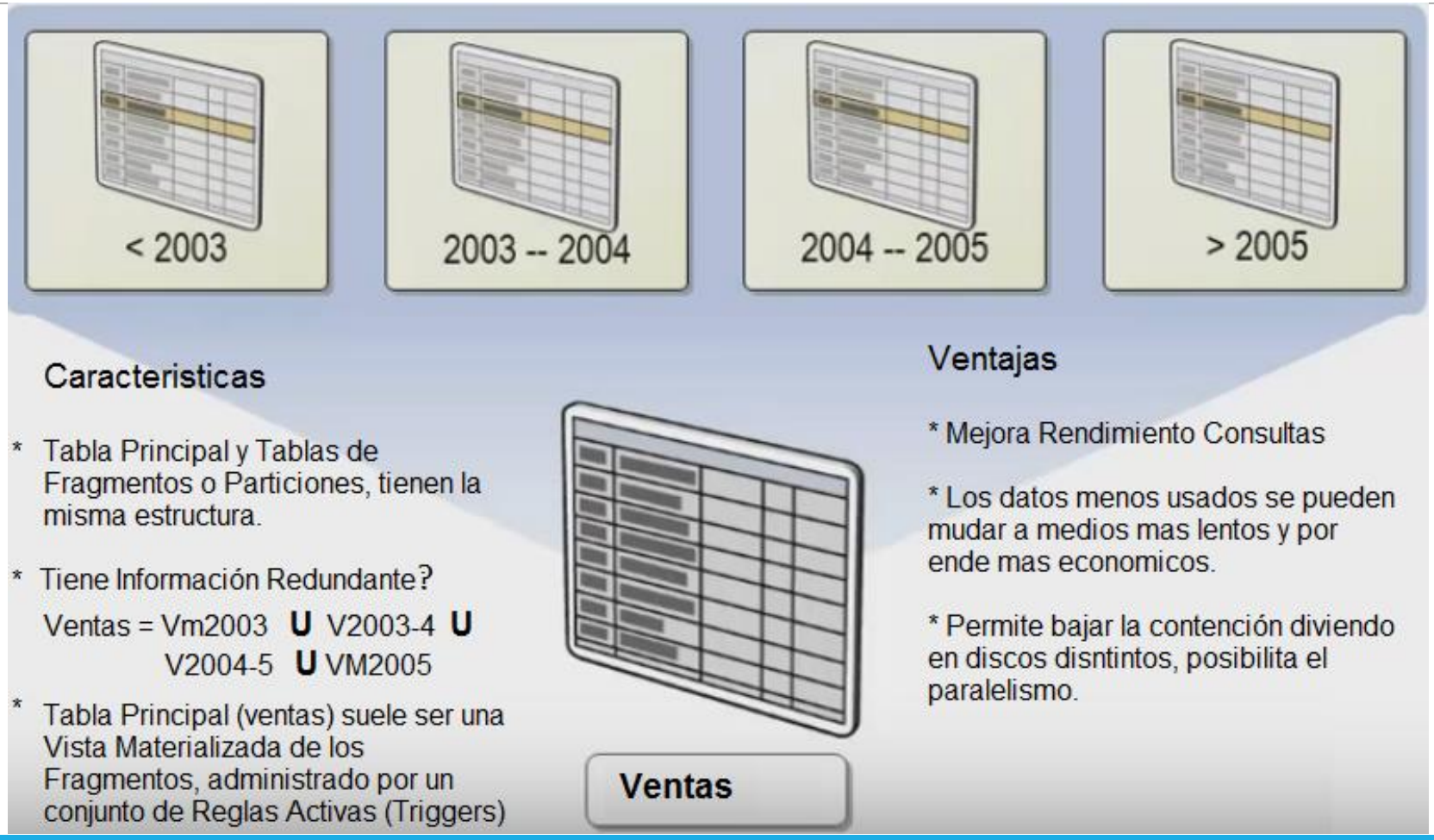
- Introducción y Conceptos
- Reglas de Date
- Profundización conceptos Fragmentación y Replicación
- Diccionario de datos
- Procesamiento de consultas
- Transacciones

Profundización: Fragmentación Historia

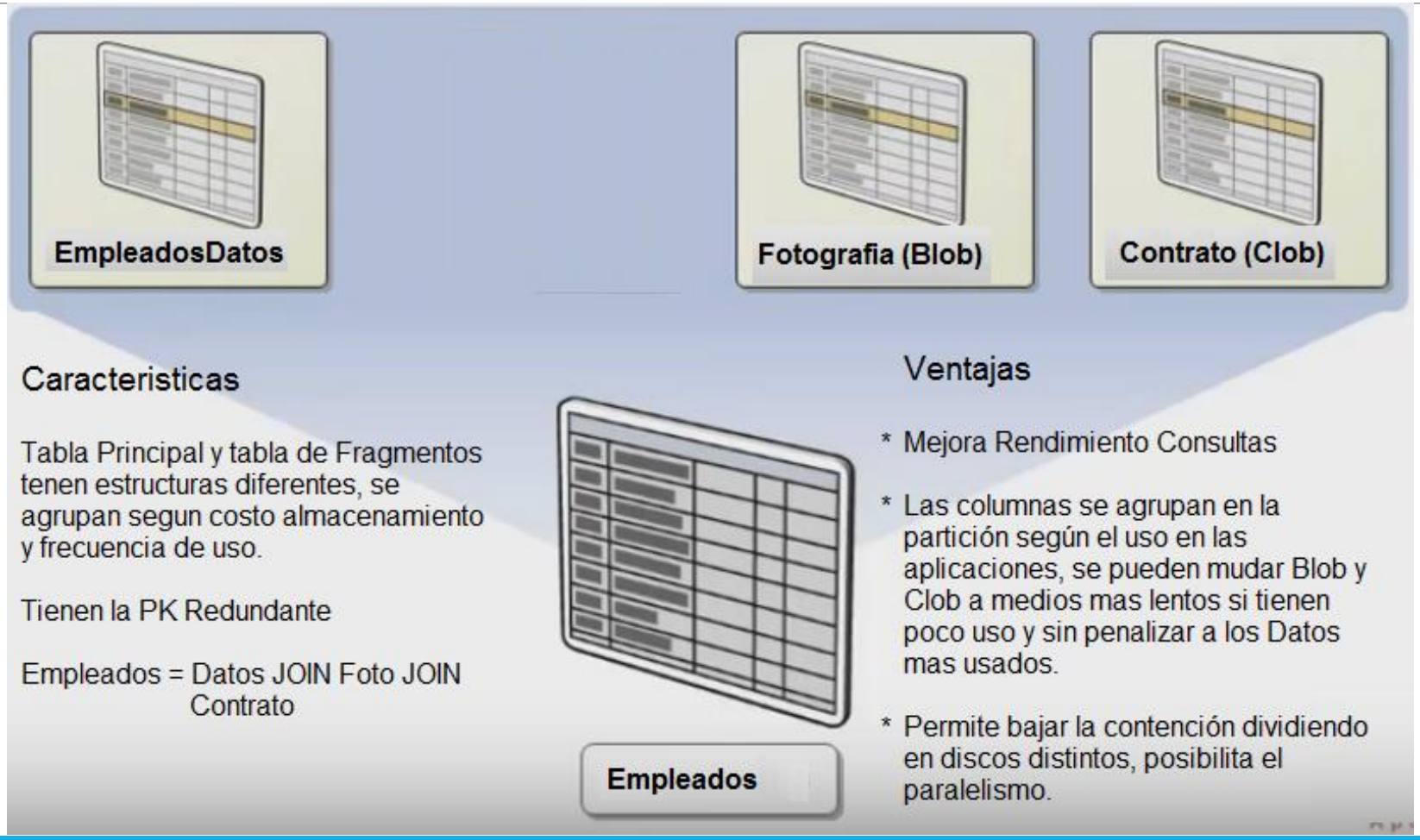
Esta idea nació con las Bases de datos Centralizadas y luego se utilizo en las BDD.

- **Horizontal:** Esta modalidad consiste en tener varias tablas con las mismas columnas en cada una de ellas y distribuir la cantidad de filas en estas tablas (generalmente se parte separando los datos por años, meses, u otro criterio).
- **Vertical:** Esta modalidad se aplica por ejemplo cuando tenemos una columna de tipo BLOB con una fotografía o un texto muy largo (CLOB), que tiene baja probabilidad de ser recuperado con el resto de los datos y decidimos ponerlo en otra tabla asociada con la misma clave primaria. También se utiliza en caso de querer evitar demasiadas columnas en una tabla y separar algunas por algún criterio.

Profundización: Fragmentación Idea - horizontal



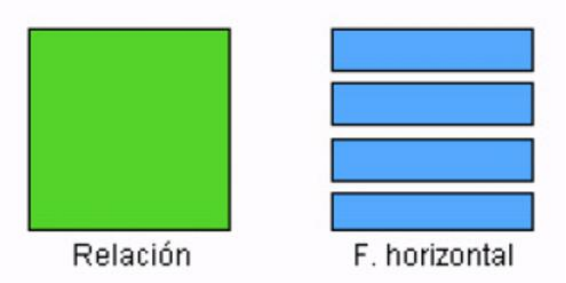
Profundización: Fragmentación Idea - vertical



Profundización: Fragmentación Horizontal

La Fragmentación tanto Horizontal como Vertical surgen como una técnica general de minimizar la contención y achicar los arboles de índices.

Fragmentar Horizontalmente una relación, implica realizar una selección sobre las tuplas, ocupando la operación de *restricción/selección* (where) sobre uno o mas atributos:



Restricción

$\sigma_{c_i="xxx"}(R)$

WHERE

C_i = “xxx”

Es decir cada fragmento dará origen a una nueva relación (una porción o subconjunto de la relación original).

La relación R se divide en los subconjuntos R_1, R_2, …, R_n.

Profundización: Fragmentación Horizontal - Ejemplo

Considere la relación Alumnos:

Jno	NOMBRE	NOTA	ESCUELA
J1	LUIS YANEZ	8	CIME
J2	ERIKA QUIROZ	8	CIME
J3	DANIEL MURILLO	9	EISIC
J4	MARIA JOSE MENDEZ	10	EISIC
J5	ADONIS PABON	9	EISIC

Alumnos1

Jno	NOMBRE	NOTA	ESCUELA
J1	LUIS YANEZ	8	CIME
J2	ERIKA QUIROZ	8	CIME

Alumnos2

Jno	NOMBRE	NOTA	ESCUELA
J3	DANIEL MURILLO	9	EISIC
J4	MARIA JOSE MENDEZ	10	EISIC
J5	ADONIS PABON	9	EISIC

Profundización: Fragmentación Horizontal - Ejemplo

Alumnos(Jno, nombre, nota, escuela)

Se aplica restricción: $\sigma_{escuela="CIME"}\ (Alumnos)$

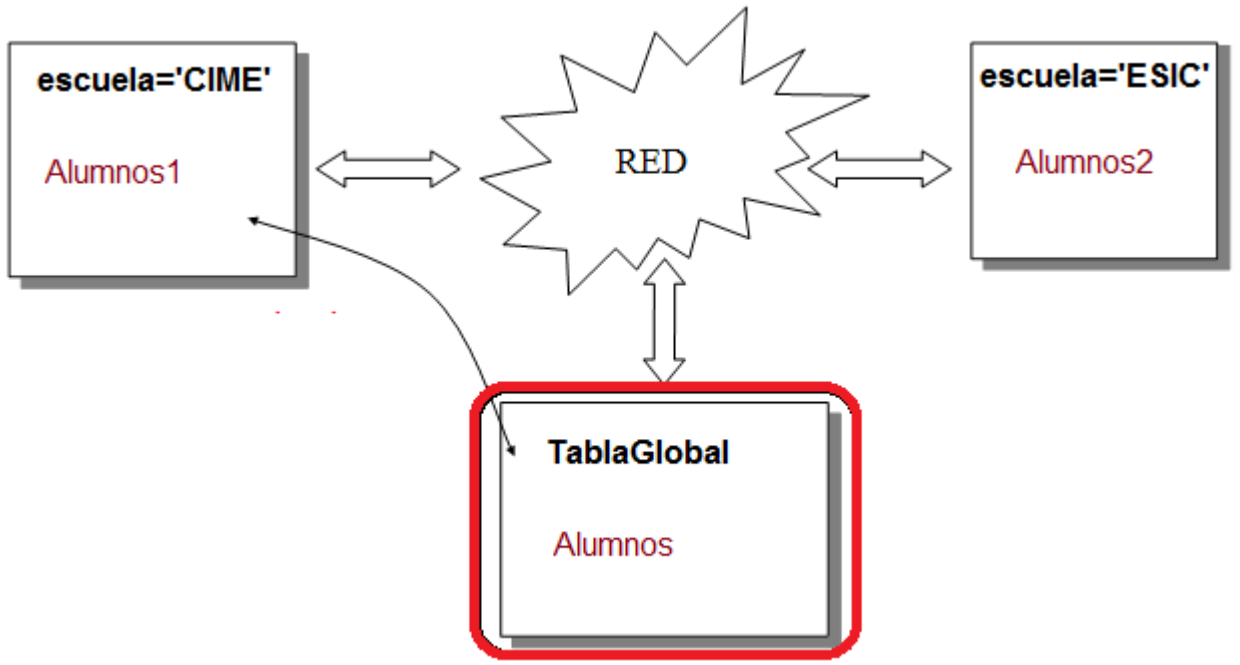
```
SELECT * FROM Alumnos WHERE escuela = 'CIME‘
```

Da origen a Alumnos1

Jno	NOMBRE	NOTA	ESCUELA
J1	LUIS YANEZ	8	CIME
J2	ERIKA QUIROZ	8	CIME

Profundización: Fragmentación Horizontal - Ejemplo

Recordemos que para el usuario en BDD la fragmentación deberá ser transparente, por lo que las DML se realizarán sobre la relación original “Alumnos”, y dependiendo en qué fragmento se encuentre, el SGBDD deberá acceder al nodo correspondiente para recuperar la información.



Profundización: Fragmentación Horizontal - Reconstrucción

Reconstrucción: Si la relación R se descompone en los fragmentos R_1, R_2, ..., R_n, entonces debe existir algún operador que permita reconstruir la relación original R.

El operador sería: U -> Unión

Alumnos = Alumnos1 U Alumnos2

Jno	NOMBRE	NOTA	ESCUELA
J1	LUIS YANEZ	8	CIME
J2	ERIKA QUIROZ	8	CIME
J3	DANIEL MURILLO	9	EISIC
J4	MARIA JOSE MENDEZ	10	EISIC
J5	ADONIS PABON	9	EISIC

Profundización: Fragmentación Horizontal – Primaria y derivada

Existen dos tipos de fragmentaciones horizontales:

- **Fragmentación Horizontal Primaria:** Aplicado a relaciones que tienen baja dependencia con otras. Básicamente el criterio de partición depende de la misma relación en la que se está efectuando.
- **Fragmentación Horizontal Derivada:** Consiste en dividir una relación partiendo de los predicados definidos sobre alguna otra, debido a que la relación R depende de la relación Q, sobre cuyos atributos está definido el predicado de la fragmentación.

Profundización: Fragmentación Horizontal – Primaria y derivada

Las tres entradas necesarias para desarrollar la fragmentación horizontal derivada son las siguientes:

- El conjunto de fragmentos de la relación principal,
- La relación miembro o secundaria
- El conjunto de predicados resultados de aplicar el “join” entre la relación principal y secundaria.

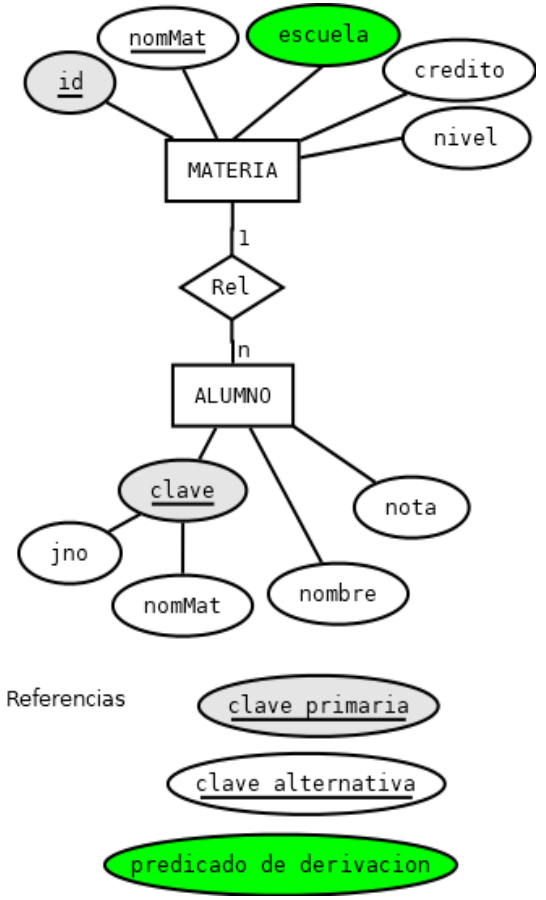
Es decir la fragmentación de la tabla principal, se aplica a tablas secundarias, o bien se debe partir de una fragmentación horizontal primaria para derivarla.

Profundización: Fragmentación Horizontal – Primaria y derivada

Ejemplo: Considere las relaciones MATERIA y ALUMNO

ID	MATERIA	ESCUELA	CRÉDITOS	NIVEL
1	Análisis Matemático	EISIC	4	1
2	Sistemas Operativos	EISIC	6	3
3	Programación II	CIME	6	2
4	Tecnología Eléctrica	EISIC	4	2
5	Técnicas de Aprendizaje	CIME	4	1
6	Dibujo Mecánico	CIME	6	3

Jno	NOMBRE	MATERIA	NOTA
J1	LUIS YANEZ	Sistemas Operativos	8
J2	ERIKA QUIROZ	Dibujo Mecánico	8
J3	DANIEL MURILLO	Técnicas de Aprendizaje	9
J4	MARIA JOSE MENDEZ	Análisis Matemático	10
J5	ADONIS PABON	Programación II	9



Profundización: Fragmentación Horizontal – Primaria y derivada

Se quiere fragmentar horizontalmente basado en la escuela en la cual el alumno está matriculado. La escuela no es un atributo de ALUMNO.

Escuela = EISIC (Fragmentación Horizontal primaria)

ID	MATERIA	ESCUELA	CRÉDITOS	NIVEL
1	Análisis Matemático	EISIC	4	1
2	Sistemas Operativos	EISIC	6	3
4	Tecnología Eléctrica	EISIC	4	2

Fragmentación Horizontal derivada

Jno	NOMBRE	MATERIA	NOTA
J1	LUIS YANEZ	Sistemas Operativos	8
J4	MARIA JOSE MENDEZ	Análisis Matemático	10

Escuela = CIME (Fragmentación Horizontal primaria)

ID	MATERIA	ESCUELA	CRÉDITOS	NIVEL
3	Programación II	CIME	6	2
5	Técnicas de Aprendizaje	CIME	4	1
6	Dibujo Mecánico	CIME	6	3

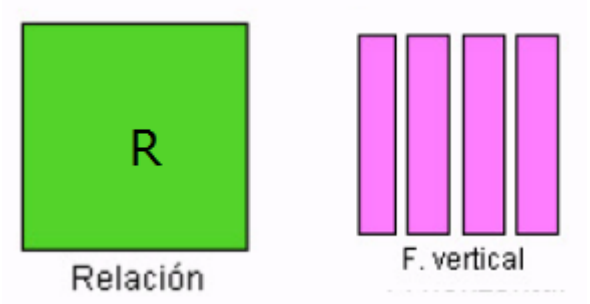
Fragmentación Horizontal derivada

Jno	NOMBRE	MATERIA	NOTA
J2	ERIKA QUIROZ	Dibujo Mecánico	8
J3	DANIEL MURILLO	Técnicas de Aprendizaje	9
J5	ADONIS PABON	Programación II	9

Profundización: Fragmentación Vertical

También la Fragmentación Vertical surge como técnica de las bases de datos centralizadas.

Fragmentar Verticalmente una relación, dijimos, se obtiene realizando una descomposición o división por atributos, de acuerdo a la operación proyección sobre estos atributos:



Proyección $\pi_{L_i}(R)$

La Fragmentación significa dividir el conjunto de atributos en subconjuntos donde solo se repite la *clave primaria* o restricción *UNIQUE*.

Profundización: Fragmentación Vertical - Ejemplo

Considere la Relación Alumnos:

Jno	NOMBRE	NOTA	ESCUELA
J1	LUIS YANEZ	8	CIME
J2	ERIKA QUIROZ	8	CIME
J3	DANIEL MURILLO	9	EISIC
J4	MARIA JOSE MENDEZ	10	EISIC
J5	ADONIS PABON	9	EISIC

Información de notas:

Jno	NOTA
J1	8
J2	8
J3	9
J4	10
J5	9

Información de Nombres y Escuelas

Jno	NOMBRE	ESCUELA
J1	LUIS YANEZ	CIME
J2	ERIKA QUIROZ	CIME
J3	DANIEL MURILLO	EISIC
J4	MARIA JOSE MENDEZ	EISIC
J5	ADONIS PABON	EISIC

Profundización: Fragmentación Vertical - Ejemplo

Alumnos(Jno, nombre, nota, escuela)

Se proyecta $\pi_{nombre,\, escuela}\ (Alumnos)$

```
SELECT Jno, nombre, escuela FROM Alumnos
```

Da origen a **Alumnos1**

Jno	NOMBRE	NOTA	ESCUELA
J1	LUIS YANEZ	8	CIME
J2	ERIKA QUIROZ	8	CIME

Se proyecta $\pi_{nota}\ (Alumnos)$

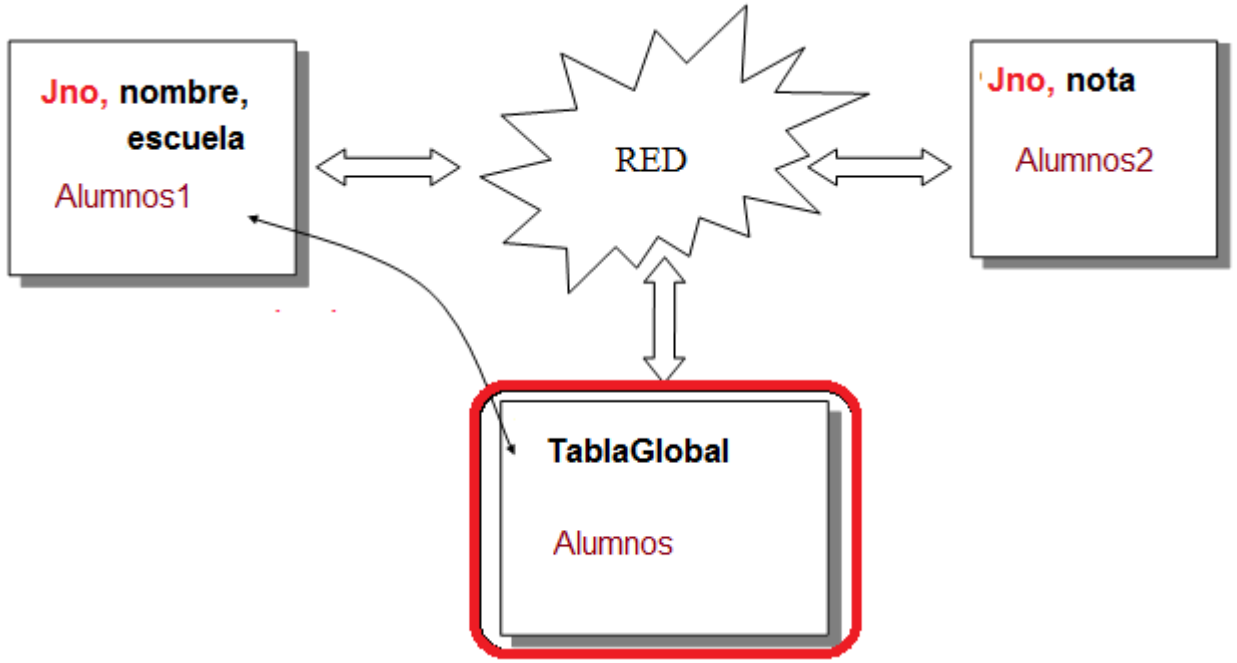
```
SELECT Jno, nota FROM Alumnos
```

Da origen a **Alumnos2**

Jno	NOTA
J1	8
J2	8
J3	9
J4	10
J5	9

Profundización: Fragmentación Vertical - Ejemplo

Igual que la fragmentación horizontal, el usuario queda ajeno a las particiones y solamente consulta sobre la tabla alumnos.



Profundización: Fragmentación Vertical - Reconstrucción

Reconstrucción: Si la relación R se descompone en los fragmentos R_1, R_2, ..., R_n, entonces debe existir algún operador que permita reconstruir la relación original R.

El operador es: ⋈ -> reunión (natural join)

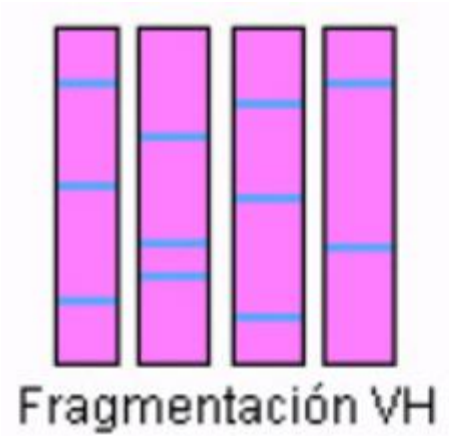
$$Alumnos = Alumnos1 \bowtie Alumnos2$$

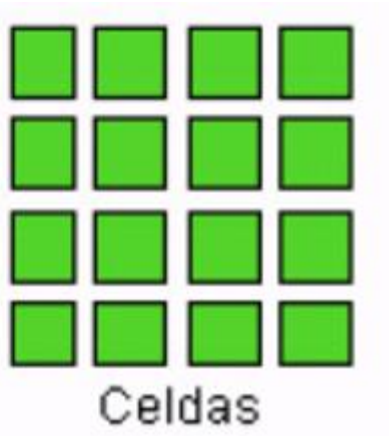
Jno	NOMBRE	NOTA	ESCUELA
J1	LUIS YANEZ	8	CIME
J2	ERIKA QUIROZ	8	CIME
J3	DANIEL MURILLO	9	EISIC
J4	MARIA JOSE MENDEZ	10	EISIC
J5	ADONIS PABON	9	EISIC

Profundización: Fragmentación Mixta o híbrida

Este tipo de fragmentación surge cuando los dos tipos anteriores se combinan.

En tal caso, la relación original puede reconstruirse aplicando las operaciones de **Unión** y **Reunión** en el orden apropiado.





Profundización: Fragmentación Mixta o híbrida

Se puede llevar a cabo de 3 maneras:

- Desarrollando primero la fragmentación vertical y, posteriormente, aplicando la fragmentación horizontal sobre los fragmentos verticales (denominada partición VH),
- Aplicando primero una división horizontal para luego, sobre los fragmentos generados, desarrollar una fragmentación vertical (llamada partición HV),
- De forma directa considerando la semántica de las transacciones y respetando las 3 reglas de fragmentación: Completitud, Reconstrucción y Disyunción.

Profundización: Fragmentación Mixta o híbrida

Combinación de FH + FV => Combinación de Selección + Proyección

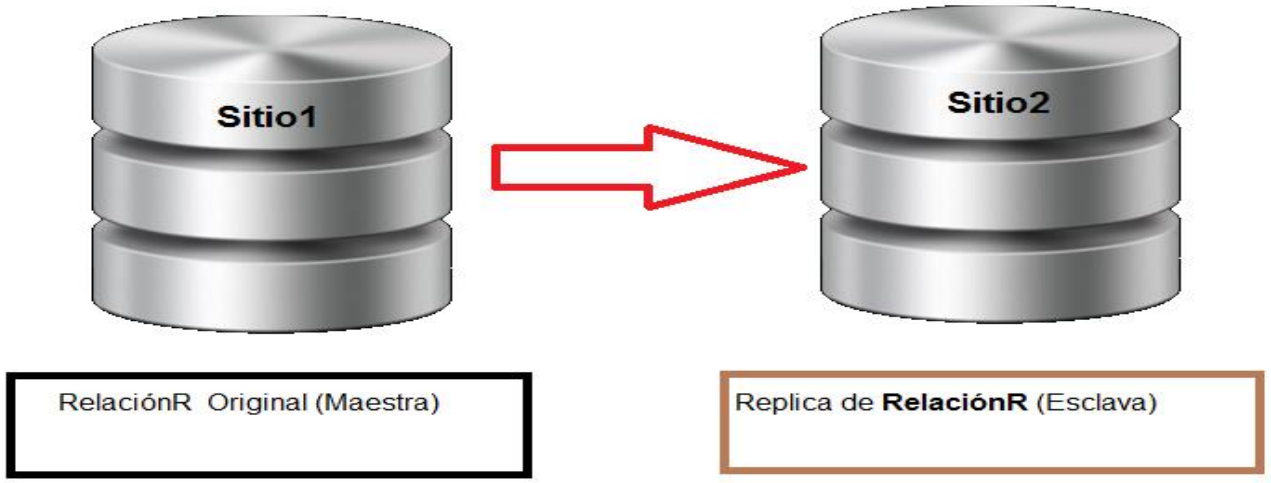
$$\pi_F\ (\sigma_C\ (R))$$

Por lo tanto:

- IF C = true AND F = attrs(R) => Relación Original
- IF C = true AND F ≠ attrs(R) => Fragmento Vertical
- IF C ≠ true AND F = attrs(R) => Fragmento Horizontal
- IF C ≠ true AND F ≠ attrs(R) => Fragmento Mixto

Profundización: Replicación
Concepto base

Antes de analizar mas profundamente este tema, definamos algunos conceptos:



- **Relación Maestra:** Relación Original.
- **Relación Esclava:** Replica de Relación Original.

Nota: Algunos casos pueden tener solo varios maestros, o un maestro y varios esclavos.

Profundización: Replicación
Concepto base

Roles:

- **Relación Maestra (Master):** El nodo maestro es la fuente primaria de los datos. Es el responsable de aceptar y procesar las operaciones de escritura (como inserciones, actualizaciones y eliminaciones) en la base de datos. El nodo maestro es la autoridad para la modificación de los datos y se encarga de enviar los cambios realizados a los nodos esclavos.

- **Relación Esclava (Slave):** Los nodos esclavos son copias de solo lectura de los datos del nodo maestro. Reciben los cambios realizados en el nodo maestro y los aplican en su propia copia local de los datos. Los nodos esclavos permiten consultas de solo lectura y no aceptan operaciones de escritura directas. Su propósito principal es proporcionar redundancia, escalabilidad y mejorar el rendimiento al distribuir la carga de trabajo.

Profundización: Replicación
Concepto base

La relación **maestro-esclavo** establece un flujo unidireccional de datos desde el nodo maestro hacia los nodos esclavos.

Cuando se realiza una operación de escritura en el nodo maestro, se registra el cambio y se envía a los nodos esclavos para que lo apliquen en sus copias locales.

Esto garantiza que los nodos esclavos mantengan los mismos datos que el nodo maestro.

Este tipo de replicación es muy utilizada en un esquema dónde las operaciones DML: INSERT, UPDATE y DELETE se realizan en un solo nodo y se replica la información al resto de los nodos a modo de “solo lectura”.

Profundización: Replicación
Concepto base

La replicación **maestro-maestro** permite que múltiples nodos sean capaces de aceptar operaciones de escritura y asegura que los cambios realizados en un nodo maestro se propaguen a todos los demás nodos maestros en la topología de replicación.

Esta modalidad aumenta considerablemente la complejidad del sistema tanto en diseño como en coordinación entre los nodos.

Además de requerir de una red de comunicaciones confiable para poder implementar diferentes protocolos que permitan que las transacciones se ejecuten de manera correcta.

Profundización: Replicación

Alcance de replicación

Replicación Completa:

Toda la BD es replicada en todos los nodos.

Esto mejora la disponibilidad al extremo, la BD puede seguir funcionando con un solo sitio activo.

Mejora el rendimiento de consultas, ya que todas las consultas pasan a ser locales.

La desventaja de la replicación completa es que penaliza las actualización, una actualización deberá realizarse en todas y cada una de las copias de la BD a fin de mantener la consistencia.

Profundización: Replicación
Alcance de replicación

Replicación Parcial:

Solo se seleccionan para replicar algunas partes de la BD en algunos sitios.

Esta replicación se define a través del esquema de replicación, este esquema tiene una descripción declarativa de la replicación de los fragmentos.

Cada copia o replica de un fragmento se debe asignar a un determinado sitio del sistema distribuido. Este proceso se denomina Distribución o Asignación de los Datos.

Profundización: Replicación
Alcance de replicación

Sin Replicación:

Caso extremo a la replicación total, solo hay una copia de cada fragmento en un único sitio.

Todos los fragmentos son disjuntos, con excepción de la porción de claves primarias en fragmentos verticales o mixtos.

Entre estos dos extremos (replicación nula o total), existe una amplia gama de replicación parcial de los datos.

Profundización: Replicación

Tipos de replicación

- **Solo Lectura (read only):** Replicas de sólo lectura cuyos datos se refrescan a intervalos especificados. Se usan básicamente por seguridad (backup) y alta disponibilidad.
- **Actualizables (updateable):** Se permite la modificación de datos sobre la instantánea. Estas modificaciones se propagan hasta tabla maestra (versión original de la relación)
- **Replicación avanzada**: replicación de los datos en varios sitios maestros. Se puede actualizar una tabla en cualquiera de los sitios maestros y la actualización se propaga al resto de los sitios.

Profundización: Replicación
Metodología

En Términos Metodológicos Prácticos

La elección de sitios y el grado de replicación dependen de los objetivos en cuanto a rendimiento y disponibilidad del sistema y de los tipos y frecuencias de transacciones previstas para cada sitio.

Encontrar una solución óptima, o incluso buena, para la asignación y replicación de datos distribuidos es un problema de optimización muy complejo.

Profundización:
Fragmentación y Replicación

Las técnicas de réplica y fragmentación se pueden aplicar a la misma relación de partida.

Cuando las tecnicas con tilde se combinan, potencian las posibilidades de efectuar consultas locales

. Consecuencias:

- Mayor Disponibilidad
- Mayor Seguridad
- Seguimos penalizando las actualizaciones.

Resumen

- Introducción y Conceptos
- Reglas de Date
- Profundización conceptos Fragmentación y Replicación
- Diccionario de datos
- Procesamiento de consultas
- Transacciones

Diccionario de datos

Definición Conceptual

Un Diccionario de BDD exige implementar más capacidades que un Diccionario de BD no Distribuida. A este lo llamaremos Diccionario Global o Extendido.

Administra el esquema de *distribución, localización o ubicación, replicación y seguridad* (usuarios / roles / permisos) de una BDD.

Es decir este elemento hace posible **la transparencia de una BDD**.

Diccionario de Datos Global Transparencia de una BDD

Transparencia de la Fragmentación: No se exige a los usuarios que conozcan el modo en que se ha fragmentado la relación.

Transparencia de la Replicación: Los usuarios “ven” a cada objeto como único.

Transparencia de la Ubicación o Localización: No se exige a los usuarios que conozcan la ubicación física de los datos

Diccionario de Datos Global
Información que administra

Información habitual de un diccionario o catálogo:

- Relaciones
- Índices
- Usuarios
- etc

Información de control:

- Transparencia de fragmentación
- Transparencia de réplica
- Transparencia de localización

Diccionario de Datos Global
Almacenamiento

- **Centralizado**: Viola regla 2 “no dependencia de un sitio central”.
- **Réplicas en cada lugar**: Viola regla: “autonomía”, costoso.
- **Catálogo dividido entre todos lugares**: Operaciones remotas costosas.
- **Catálogo dividido y una copia global en un sitio**: Viola 2.

Conclusión: todos estos métodos tienen problemas.

Diccionario de Datos Global Almacenamiento

¿Pero como se puede hacer de estos requerimientos una implementación real?

La respuesta a esto es un problema del Diccionario Global Extendido, trabaja a modo “Servicio de nombres”.

Cada Sitio/Nodo se registra en el Servicio de nombres del diccionario.

Resumen

- Introducción y Conceptos
- Reglas de Date
- Profundización conceptos Fragmentación y Replicación
- Diccionario de datos
- Procesamiento de consultas
- Transacciones

Procesamiento de consultas

Recordemos para este punto que los datos pueden estar fragmentados de manera vertical, horizontal o híbrida.

Al mismo tiempo pueden o no estar replicados parcial o totalmente.

El SGBDD deberá resolver las consultas de la manera más eficiente posible.

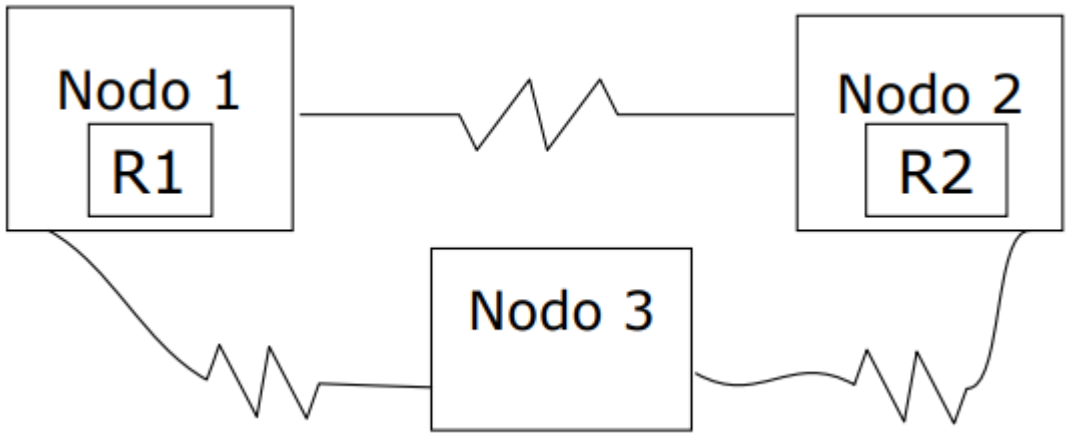
Procesamiento de consultas
Procesamiento Distribuido de Consultas

Proceso Básico de Optimización:

Consiste en evaluar el peso de las comunicaciones de cada alternativa.

Consiste en evaluar 3 componentes de costos:

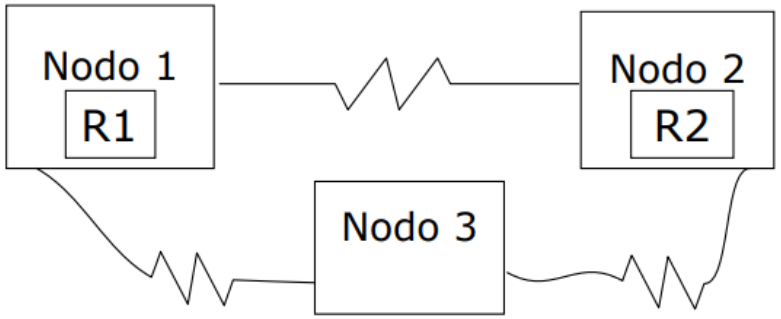
Costo total = costo I/O + Costo Procesamiento + Costo de Comunicaciones



Procesamiento de consultas
Procesamiento Distribuido de Consultas

Si el costo de I/O y de Procesamiento es igual para todos los nodos:

Consiste en evaluar el peso de las comunicaciones de cada alternativa



Consulta en Nodo3 R1 U R2 => <decisión>

- R1 viaja a N2
  viaja Res a N3
- R2 viaja a N1
  viaje Res a N3
- R1, R2 viajan a N3

Procesamiento de consultas
Procesamiento Distribuido de Consultas

Se estudia el costo de las comunicaciones.

Objetivo: la reducción de la cantidad de datos transferidos

Optimización mediante operación de semijoin

Obtenemos solo los atributos de R que participan en el semijoin (por izquierda), usando natural join:

$R \ltimes S = \Pi a_1,..,a_n (R * S)$; donde a_1, ..., a_n son atributos de R

$\rtimes$ => semijoin por derecha

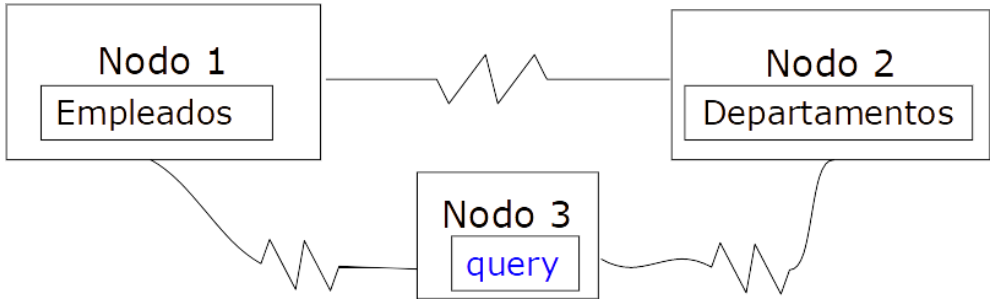
Procesamiento de consultas
Procesamiento Distribuido de Consultas

Nodo1: Empleado => 10.000 tuplas. (100 byte x tupla)

Nodo2: Departamento => 100 tuplas. (35 byte x tupla)

Nodo3: En este nodo se realiza la consulta

*"Por cada empleado, obtener el **nombre del empleado** y el **nombre del departamento** al que pertenece"*



Procesamiento de consultas
Procesamiento Distribuido de Consultas

Nodo1: Empleado

Nombre	Apellido	COD	Dir	Sexo	Sueldo	fechaNac.	Dpto.

10.000 tuplas.

Cada tupla tiene 100 bytes de longitud.

- El campo COD tiene 9 bytes de longitud.
- El campo Dpto tiene 4 bytes de longitud.
- El campo Nombre tiene 15 bytes de longitud.
- El campo Apellido tiene 15 bytes de longitud.
- Etc.

Tamaño de la relación: 100 * 10.000 = 10^6 bytes

Procesamiento de consultas
Procesamiento Distribuido de Consultas

Nodo2: Departamento

NombreDpto	NDpto	Responsable	Edificio

100 tuplas.

Cada tupla tiene 35 bytes de longitud.

- El campo NombreDpto tiene 10 bytes de longitud.
- El campo NDpto tiene 4 bytes de longitud.
- El campo Responsable tiene 9 bytes de longitud.

Tamaño de la relación: 35 * 100 = 3500 bytes

Procesamiento de consultas
Procesamiento Distribuido de Consultas

*"Por cada empleado, obtener el **nombre del empleado** y el **nombre del departamento** al que pertenece"*

```
SELECT nombre, apellido, nombredpto
  FROM empleado
  JOIN departamento ON ndpto = dpto
```

Se consulta desde nodo3 que no tiene los datos.

- Cantidad de Filas: 10.000 tuplas.
- Cantidad de Byte de cada tupla: 40 bytes.
- Tamaño del Resultado: 400.000 bytes.

Existen **tres alternativas** para resolver la consulta.

Procesamiento de consultas
Procesamiento Distribuido de Consultas

Primera alternativa:

Transferir toda la información de las relaciones EMPLEADO y DEPARTAMENTO al nodo respuesta (nodo3) y realizar allí mismo la operación de join.

En éste caso se transfieren:

1.000.000 + 3.500 = 1.003.500 bytes.

Procesamiento de consultas
Procesamiento Distribuido de Consultas

Segunda alternativa:

Transferir la relación EMPLEADO al nodo2, ejecutar el join en este nodo y enviar el resultado al nodo3.

En éste caso se transfieren:

1.000.000 + 400.000 (resultado) = 1.400.000 bytes

Procesamiento de consultas
Procesamiento Distribuido de Consultas

Tercera alternativa:

Transferir la relación DEPARTAMENTO al nodo1, ejecutar el join en este nodo y enviar el resultado al nodo3.

En éste caso se transfieren:

3.500 + 400.000 (resultado) = 403.500 bytes.

Solución optima => alternativa 3

Resumen

- Introducción y Conceptos
- Reglas de Date
- Profundización conceptos Fragmentación y Replicación
- Diccionario de datos
- Procesamiento de consultas
- Transacciones

Transacciones distribuidas

Los componentes clave y los conceptos asociados con las transacciones distribuidas:

- **Coordinador de transacciones:** Coordina y supervisa la ejecución de la transacción distribuida, es responsable de iniciar la transacción, coordinar las operaciones en los nodos participantes y asegurar ACID de la transacción global.
- **Participantes**: Son los nodos o sitios de la base de datos distribuida que participan en la transacción.
- **Protocolo de coordinación**: Es el conjunto de reglas y procedimientos utilizados por el coordinador y los participantes para garantizar la coherencia y el cumplimiento de las propiedades ACID.

Transacciones distribuidas
Protocolo de coordinación

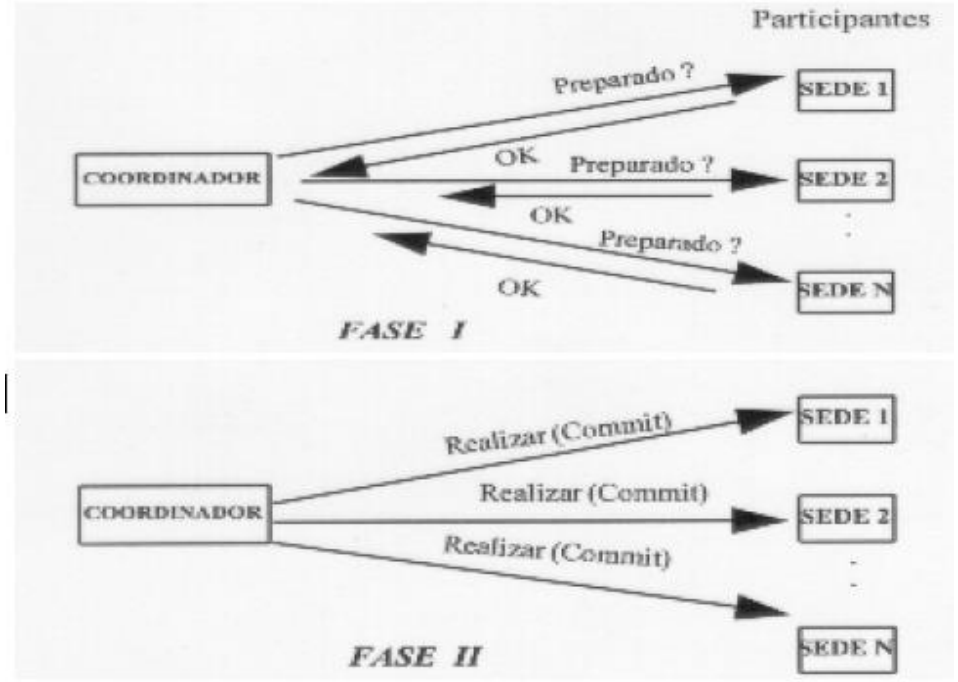
Protocolo de dos fases (2PC)

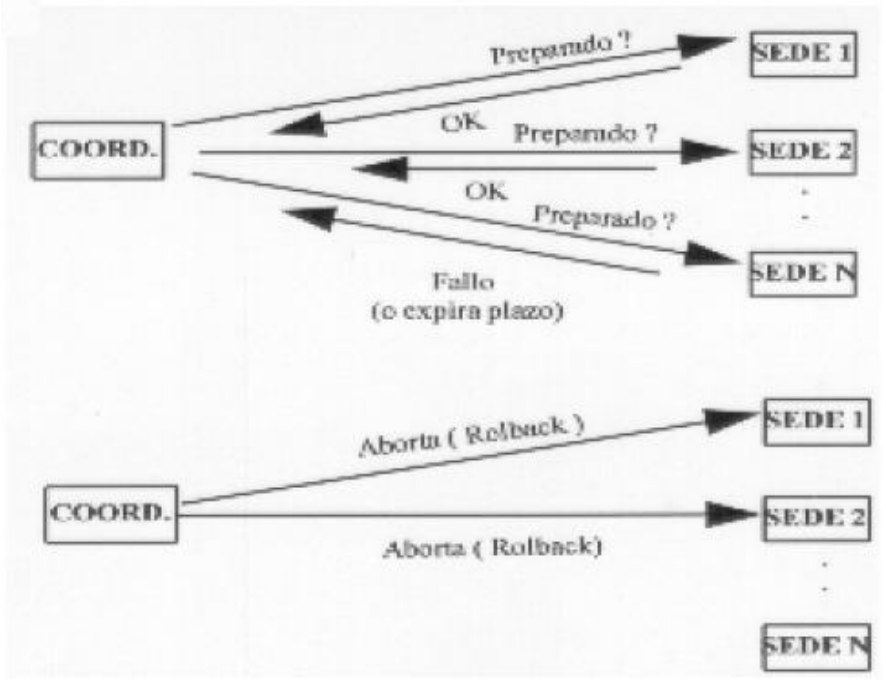
1. **Fase de preparación:** El coordinador envía una solicitud de preparación a todos los participantes de la transacción. Cada participante ejecuta su operación y responde al coordinador con un voto de preparación (ya sea "preparado" o "no preparado").

2. **Fase de confirmación:**
   - Si todos los participantes votan "preparado", el coordinador envía una solicitud de confirmación a todos los participantes. En esta etapa, los participantes confirman de manera definitiva que pueden comprometer la transacción y realizar cambios permanentes en sus datos.
   - Si algún participante vota "no preparado" o si no se recibe respuesta, el coordinador envía una solicitud de aborto a todos los participantes.

Transacciones distribuidas
Protocolo de coordinación

Protocolo de dos fases (2PC)





Transacciones distribuidas
Protocolo de coordinación

Protocolo de dos fases (3PC)

El Protocolo de tres fases introduce en primera instancia una fase de votación para detectar la falta de respuesta o la incapacidad de los participantes para votar, esto evitaría un bloqueo potencial que puede ocurrir en el 2PC debido a la falta de respuesta de un participante.

Si un participante no responde en la fase de votación, se considera un voto de "no“.