



# **BASES DE DATOS DISTRIBUIDAS**

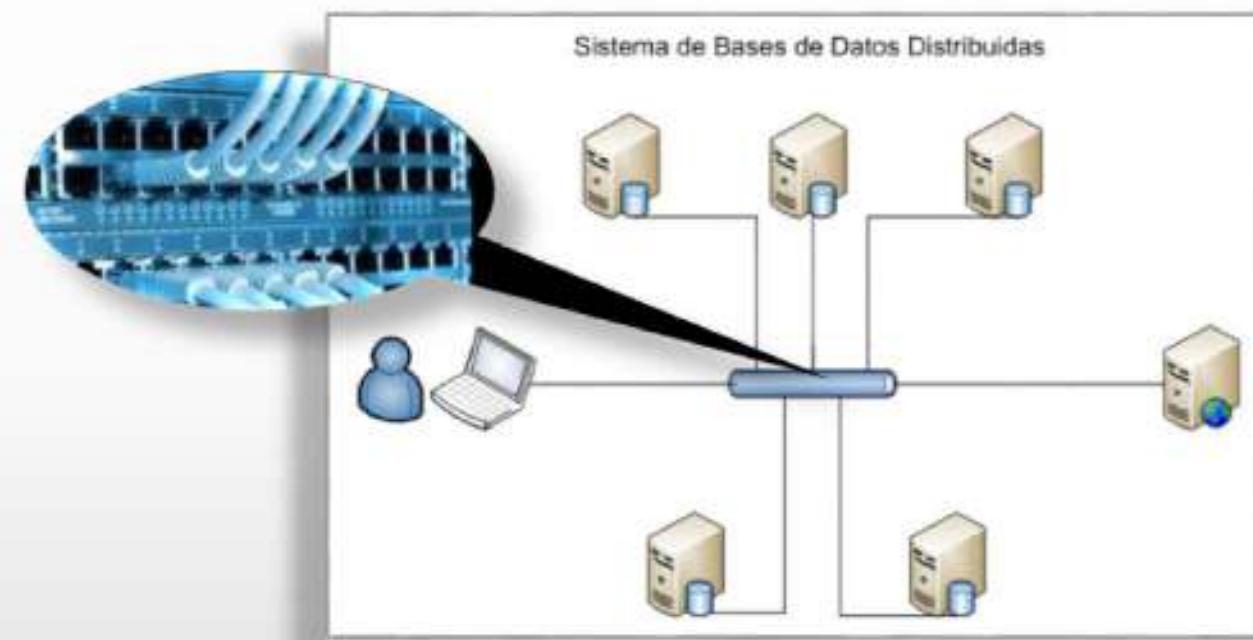
Juan Camilo Camacho  
Julián David C.  
Yohana Delgado Ramos.

# AGENDA

## INTRODUCCIÓN

1. Bases de Datos Homogéneas y Heterogéneas
2. Almacenamiento distribuido de la información
3. Transacciones distribuidas
4. Protocolos de compromiso
5. Control de Conurrencia en Bases de Datos Distribuidas
6. Disponibilidad
7. Procesamiento distribuido de consultas
8. Bases de datos distribuidas heterogéneas
9. Bases de Datos basadas en la nube
10. Estado del arte

# Introducción



Bases de datos centralizadas	Bases de datos distribuidas
Independencia de los datos	Transparencia en la distribución
Reducción de redundancia	Replicación de datos y copias múltiples
Estructura físicas complejas	Optimización global

*"Conjunto de múltiples bases de datos lógicamente relacionadas las cuales están distribuidas entre diferentes sitios Interconectados por una red de comunicaciones"*

<http://froac.manizales.unal.edu.co/roap/scorm/472/2.PNG>

[http://desarrolloik.site90.net/wp-content/uploads/2012/04/Fotolia\\_10772948\\_XS\\_Mant\\_Red.jpg](http://desarrolloik.site90.net/wp-content/uploads/2012/04/Fotolia_10772948_XS_Mant_Red.jpg)

Silberschatz, A., Forth, HF., Sudarshan, S. (2009). *Database System Concepts*. Editorial. McGraw-Hill. 6a ed. Estados Unidos. 13

Toledo, V., Miralles, I. (Sin fecha). *Bases de Datos Distribuidas*. 13pp. <https://iesanvicente.com/colaboraciones/BBDDdistribuidas.pdf>

Gutiérrez, A. (2005). *Diseño de Base de Datos Distribuida (Texto Base)*. [https://lilhectortorres.files.wordpress.com/2010/09/base\\_de\\_datos\\_distribuidas.pdf](https://lilhectortorres.files.wordpress.com/2010/09/base_de_datos_distribuidas.pdf)

# 1. Bases de Datos Homogéneas y Heterogéneas



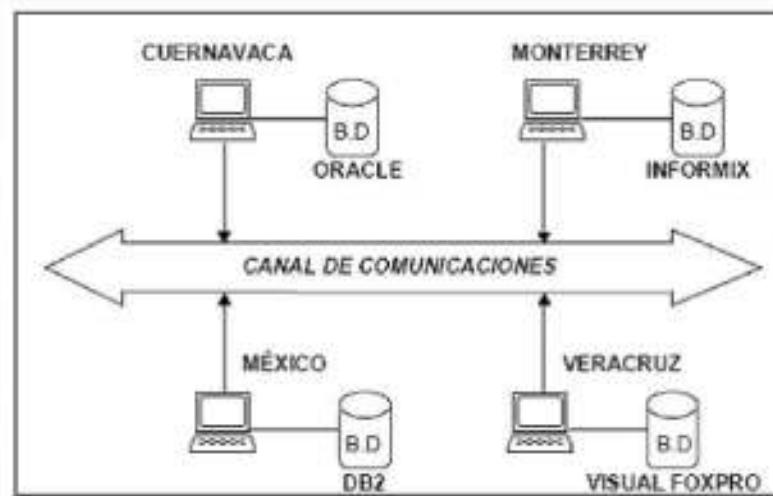
# Bases de Datos Homogéneas y Heterogéneas

## SISTEMAS DE GESTIÓN DE BASES DE DATOS HETEROGENEAS (BASE DE DATOS GLOBAL VIRTUAL)



*"Sistema de bases de datos distribuidas heterogéneas"*

# Bases de Datos Homogéneas y Heterogéneas



*"Sistema de bases de datos distribuidas heterogéneas"*

## **2. Almacenamiento distribuido de la información**



<http://www.channelpartner.es/siteresources/files/496/00.JPG>

# Replicación de la información



Disponibilidad

Paralelismo incrementado

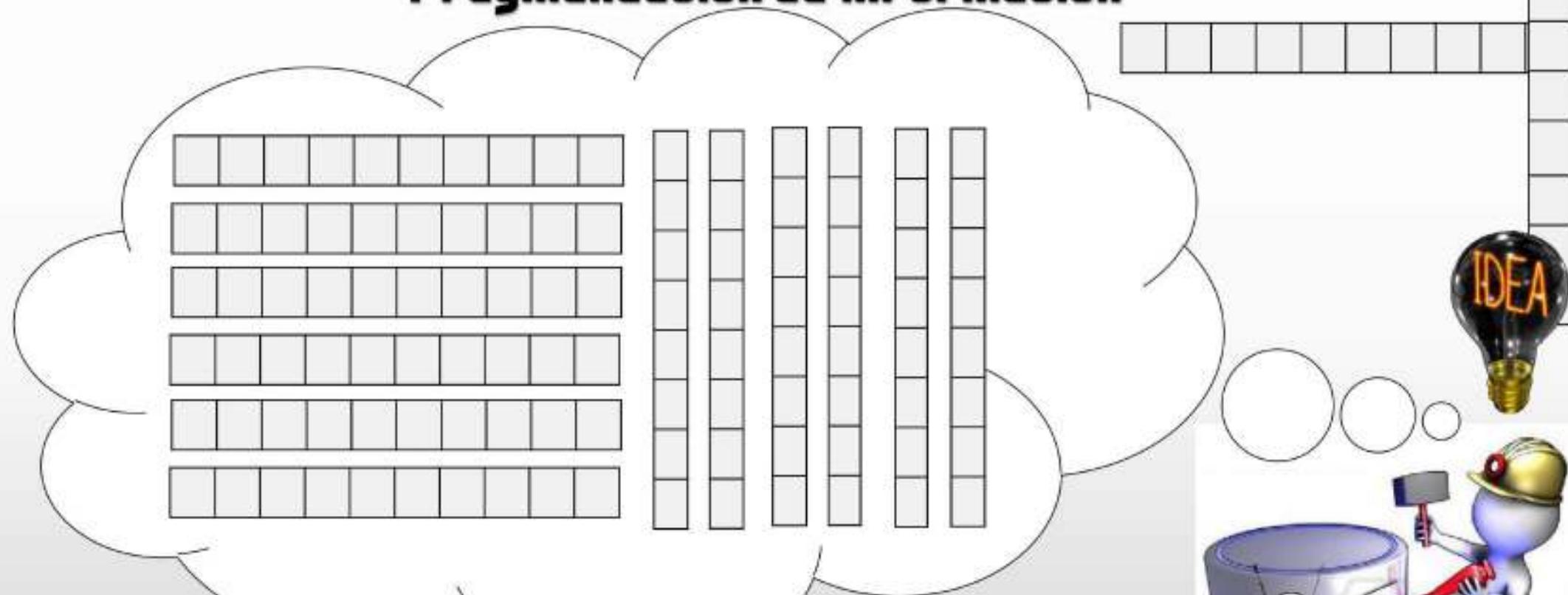
Gastos Generales

[http://blog.gfi.es/wp-content/uploads/2013/06/replicar\\_bbdd.jpg](http://blog.gfi.es/wp-content/uploads/2013/06/replicar_bbdd.jpg)

Silberschatz, A., Forth, HF., Sudarshan, S. (2009). *Database System Concepts*. Editorial. McGraw-Hill. 6a ed. Estados Unidos. 13

Toledo, V., Miralles, I. (Sin fecha). *Bases de Datos Distribuidas*. 13pp. <https://iessanvicente.com/collaboraciones/BBDDdistribuidas.pdf>

# Fragmentación de información



[http://www.topforo.com/crear-foro/gagtis/resps/83911\\_1\\_g.jpg](http://www.topforo.com/crear-foro/gagtis/resps/83911_1_g.jpg)

Silberschatz, A., Forth, HF., Sudarshan, S. (2009). Database System Concepts. Editorial. McGraw-Hill. 6a ed. Estados Unidos. 13

Ricardo, C. (2003). Fragmentación de Datos en Bases de Datos Distribuidas. Tecnología en Marcha. 16:4. pp. 60-67.

Toledo, V., Miralles, I. (Sin fecha). Bases de Datos Distribuidas. 13pp.

<https://iessanvicente.com/collaboraciones/BBDdistribuidas.pdf>

Gutiérrez, A. (2005). Diseño de Base de Datos Distribuida (Texto Base).

[https://lihectortorres.files.wordpress.com/2010/09/base\\_de\\_datos\\_distribuidas.pdf](https://lihectortorres.files.wordpress.com/2010/09/base_de_datos_distribuidas.pdf)

# Fragmentación de información

## Fragmentación horizontal

$$r_i = \sigma_{p_i}(r)$$
$$\downarrow$$
$$r = r_1 \cup r_2 \cup r_3 \cup r_4 \dots \cup r_n$$

Cuenta [separada por ramas en particular]: {Hillside, Valleyview}

$cuenta_1 = \sigma_{nombre\_rama="Hillside"}(cuenta)$

$cuenta_2 = \sigma_{nombre\_rama="Valleyview"}(cuenta)$

# Fragmentación de información

## Fragmentación horizontal

$\sigma_{escuela="EUI"}(alumno)$

DNI	Escuela	Nombre	Nota ingreso	Beca	DNI	Escuela	Nombre	Nota ingreso	Beca
87633483	EUI	Concha Queta	5.6	No	87633483	EUI	Concha Queta	5.6	No
99855743	EUI	Josechu Letón	7.2	Si	99855743	EUI	Josechu Letón	7.2	Si
05399075	EUIT	Oscar Romato	6.1	Si	05399075	EUI	Bill Gates	5.0	No
44543324	EUIT	Bill Gates	5.0	No	44543324	EUI	Maite Clado	7.5	Si
44343234	EUIT	Pepe Pótamo	8.0	No					
44543324	EUI	Maite Clado	7.5	Si					
66553234	EUIT	Ernesto Mate	6.6	No					

$\sigma_{escuela="EUIT"}(alumno)$

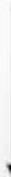
DNI	Escuela	Nombre	Nota ingreso	Beca
33887293	EUIT	Oscar Romato	6.1	Si
44343234	EUIT	Pepe Pótamo	8.0	No
66553234	EUIT	Ernesto Mate	6.6	No

# Fragmentación de información

## Fragmentación vertical

$$R_i = R_1 \cup R_2 \cup R_3 \dots \cup R_n$$

$$r_i = \pi_{R_i}(r)$$



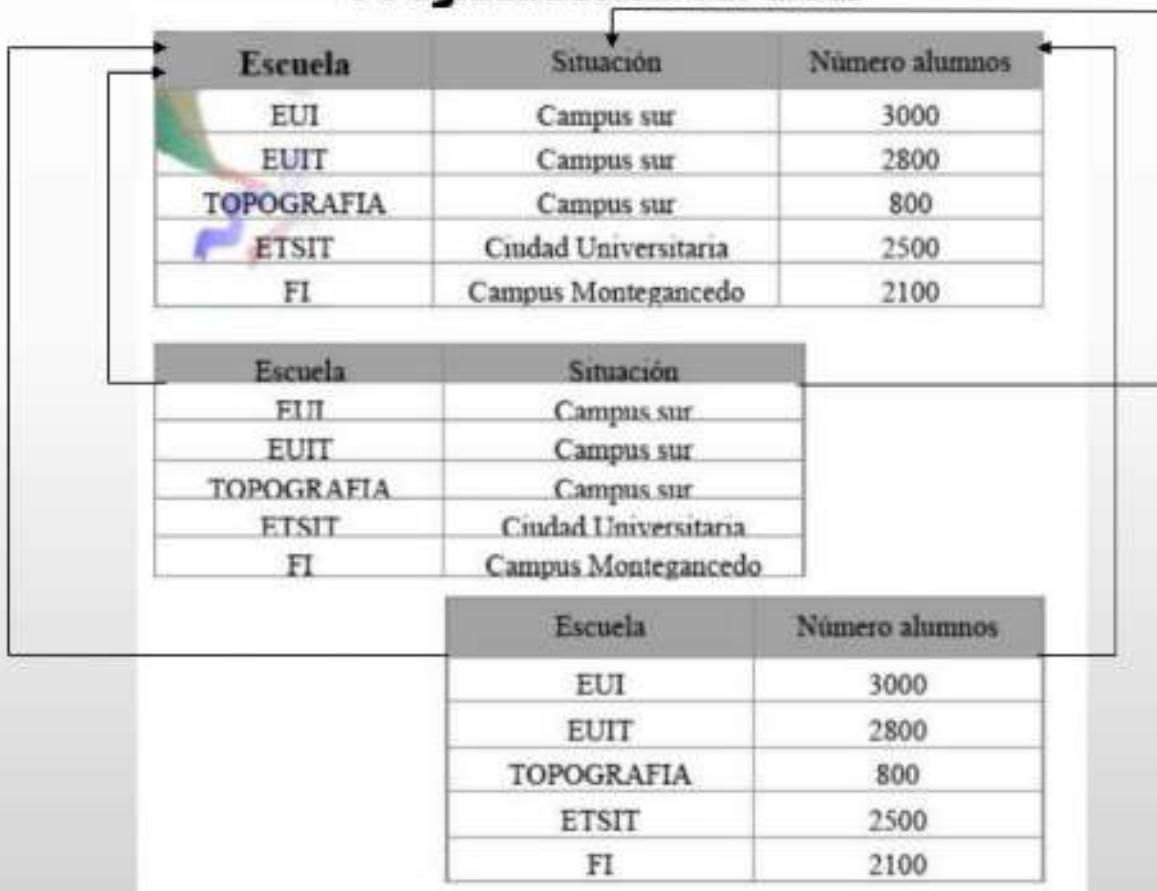
$$r = r_1 \bowtie r_2 \bowtie r_3 \dots \bowtie r_n$$

Informacion\_empleado: (empleado\_id, nombre, puesto, salario)

→ Informacion\_publica\_empleado[seguridad]: (empleado\_id, nombre, puesto)

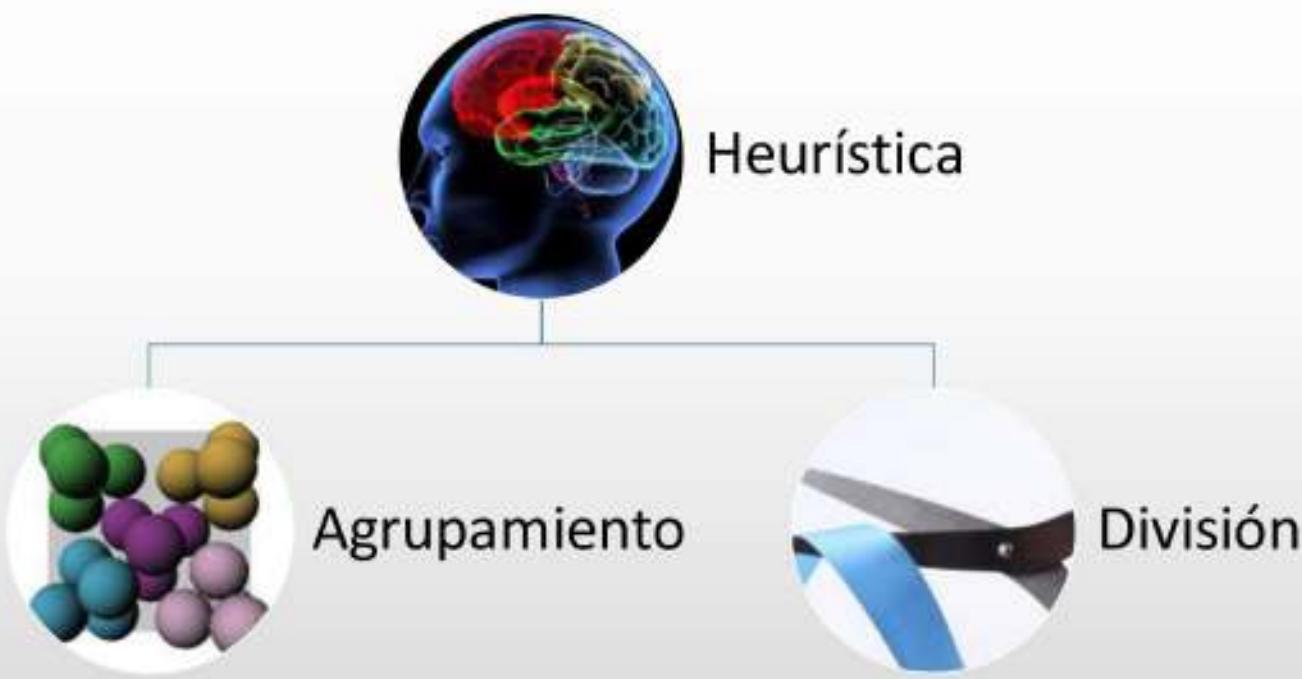
# Fragmentación de información

## Fragmentación vertical



# Fragmentación de información

## Fragmentación vertical



# Fragmentación de información

## Fragmentación vertical

	$S_1$	$S_2$	$S_3$
$q_1$	15	20	10
$q_2$	5	0	0
$q_3$	25	25	25
$q_4$	3	0	0

$$usa(q_i, A_j) = \begin{cases} 1, & \text{si el atributo } A_j \\ & \text{es referido por la consulta } q_i \\ 0, & \text{en caso contrario} \end{cases}$$

	$A_1$	$A_2$	$A_3$	$A_4$
$q_1$	1	0	1	0
$q_2$	0	1	1	0
$q_3$	0	1	0	1
$q_4$	0	0	1	1

$$aff(A_i, A_j) = ref(q_k) * acc(q_k)$$

Suponga que por cada consulta el total de accesos a los atributos es de 1:

$$aff(A_i, A_j) = acc(q_k) \Rightarrow \sum_{l=?}^N \sum_{k=?}^{TS} acc_l(q_k)$$

	$A_1$	$A_2$	$A_3$	$A_4$
$A_1$	45	0	45	0
$A_2$	0	80	5	75
$A_3$	45	5	53	3
$A_4$	0	75	3	78

# Fragmentación de información

## Fragmentación vertical

### Algoritmo BEA

	$A_1$	$A_2$	$A_3$	$A_4$
$A_1$	45	0	45	0
$A_2$	0	80	5	75
$A_3$	45	5	53	3
$A_4$	0	75	3	78

	$A_1$	$A_2$	$A_3$	$A_4$
$A_1$	45	0		
$A_2$	0	80		
$A_3$	45	5		
$A_4$	0	75		

$$cont(A_i, A_k, A_j) = 2bond(A_i, A_k) + 2bond(A_k, A_l) - 2bond(A_i, A_j)$$

$$bond(A_x, A_y) = \sum_{z=1}^n aff(A_x, A_z)aff(A_z, A_y)$$

$$\begin{aligned} cont(A_0, A_3, A_1) &= 2bond(A_0, A_3) + 2bond(A_3, A_1) - 2bond(A_0, A_1) \\ &= 2 * 0 + 2 * 4410 - 2 * 0 = 8820 \end{aligned}$$

$$\begin{aligned} cont(A_1, A_3, A_2) &= 2bond(A_1, A_3) + 2bond(A_3, A_2) - 2bond(A_1, A_2) \\ &= 2 * 4410 + 2 * 890 - 2 * 225 = 10150 \end{aligned}$$

$$\begin{aligned} cont(A_2, A_3, A_4) &= 2bond(A_2, A_3) + 2bond(A_3, A_4) - 2bond(A_2, A_4) \\ &= 1780 \end{aligned}$$

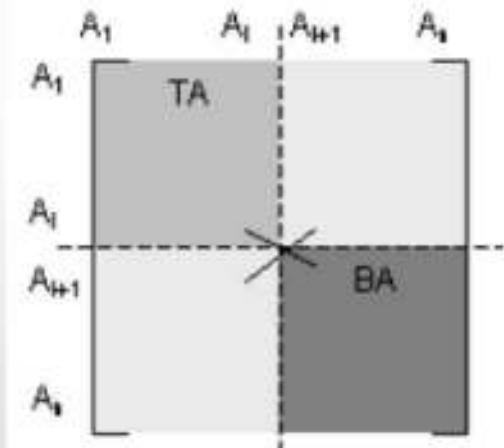
	$A_1$	$A_3$	$A_2$	$A_4$
$A_1$	45	45	0	0
$A_3$	45	53	5	3
$A_2$	0	5	80	75
$A_4$	0	3	75	78

# Fragmentación de información

## Fragmentación vertical

### Algoritmo de particionamiento

función objetivo:  $z = CTQ * CBQ - COQ^2$



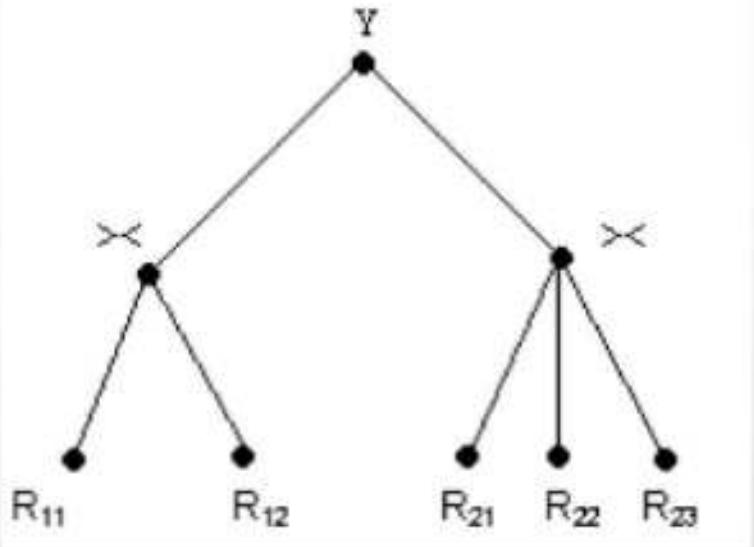
	$A_1$	$A_3$	$A_2$	$A_4$
$A_1$	45	45	0	0
$A_3$	45	53	5	3
$A_2$	0	5	80	75
$A_4$	0	3	75	78

$$r_1 = \{A_1, A_2\}$$
$$r_2 = \{A_1, A_3, A_4\}$$

# Fragmentación de información

## Fragmentación mixta/híbrida

$$r_i = \pi_{R_i}[\sigma_{p_i}(r)]$$



DNI	Escuela	Nombre	Beca
87633483	EUI	Concha Queta	No
99855743	EUI	Josechu Letón	Si
05399075	EUI	Bill Gates	No
44543324	EUI	Maite Clado	Si

$\Pi_{DNI, Escuela, Nombre, Beca}(E)$

Fragmento de la EUI:  $\sigma_{Escuela="EUI"}(T)$

DNI	Escuela	Nombre	Nota ingreso	Beca
87633483	EUI	Concha Queta	5.6	No
99855743	EUI	Josechu Letón	7.2	Si
05399075	EUI	Bill Gates	5.0	No
44543324	EUI	Maite Clado	7.5	Si

$\Pi_{DNI, Escuela, Nombre, Nota ingreso}(E)$

DNI	Escuela	Nombre	Nota ingreso
87633483	EUI	Concha Queta	5.6
99855743	EUI	Josechu Letón	7.2
05399075	EUI	Bill Gates	5.0
44543324	EUI	Maite Clado	7.5

# Fragmentación de información

## Costo Total

$$TOC = \sum_{\forall q_i \in Q} QPC_i + \sum_{\forall s_k \in S} \sum_{\forall F_j \in F} STC_{jk}$$

$$QPC_i = PC_i + TC_i$$

$$STC_{jk} = USC_k * \text{size}(F_j) * x_{jk}$$

$$PC_i = AC_i + IE_i + CC_i$$

- ✓ Procesamiento de consultas
- ✓ Almacenamiento

$$AC_i = \sum_{\forall s_k \in S} \sum_{\forall F_j \in F} (u_{ij} * UR_{ij} + r_{ij} * RR_{ij}) * x_{jk} * LPC_k$$

$$TOC = \sum_{\forall q_i \in Q} \left[ \sum_{\forall s_k \in S} \sum_{\forall F_j \in F} (u_{ij} * UR_{ij} + r_{ij} * RR_{ij}) * x_{jk} * LPC_k + IE_i + CC_i + TC_i + \sum_{\forall s_k \in S} \sum_{\forall F_j \in F} USC_k * \text{size}(F_j) * x_{jk} \right]$$

# Transparencia



### **3. Transacciones Distribuidas**



- Deben preservar las propiedades ACID

# ESTRUCTURA DEL SISTEMA

## GESTOR DE TRANSACCIONES

- ❖ Mantenimiento de un registro histórico con fines de recuperación
- ❖ Participación en un esquema adecuado de control de la concurrencia.

## COORDINADOR DE TRANSACCIONES

- ❖ Inicio de la ejecución de la transacción
- ❖ División de la transacción en varias subtransacciones y distribución a los sitios correspondientes para su ejecución
- ❖ Coordinación de la terminación de la transacción

# Arquitectura del sistema

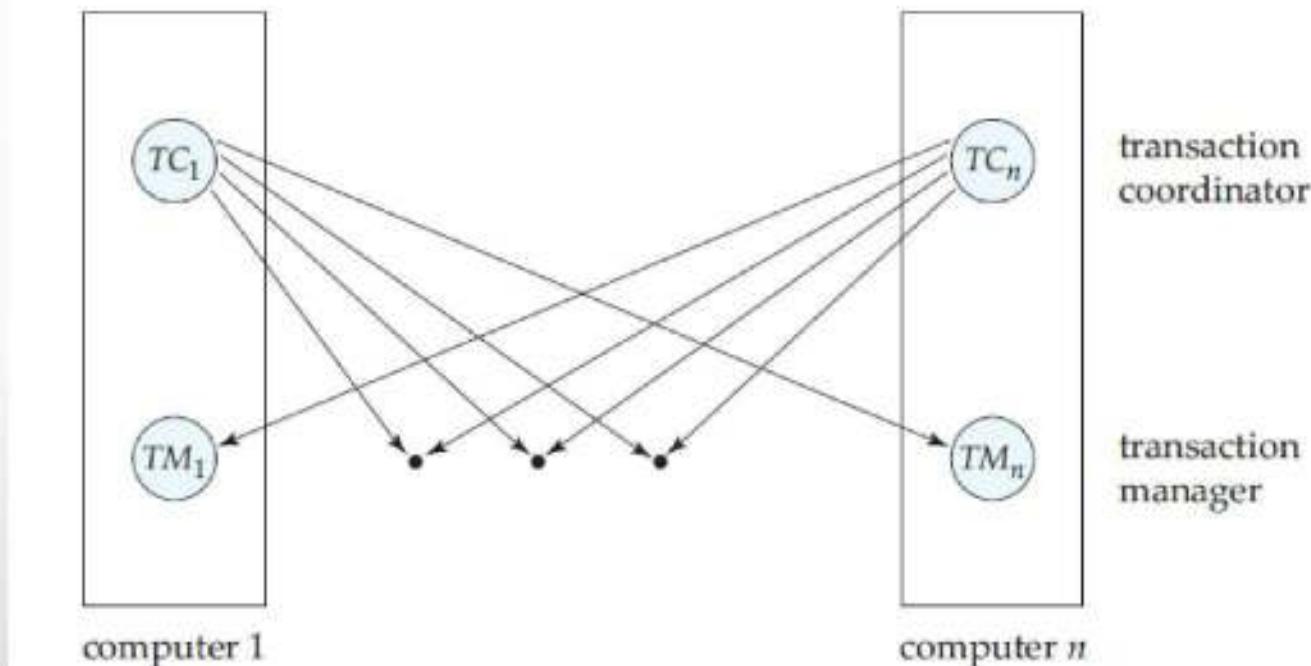


Figure 19.2 System architecture.

Fuente: Silberschatz, Abraham, Henry F. Korth, and S. Sudarshan, *Database System Concepts*, 6th ed (New York: McGraw-Hill, 2011)

## **3.1. Modos de falla del sistema**

- ❖ Fallo de un sitio
- ❖ Pérdida de mensajes
- ❖ Fallo de un enlace de comunicaciones
- ❖ División de la red



# 4. Protocolos de compromiso

## Protocolos de compromiso

Compromiso de dos fases

Protocolo de compromiso

Manejando las fallas

Recuperación y control de concurrencia

Compromiso de tres fases



# Protocolos de compromiso

## Protocolos de compromiso

**Compromiso de dos fases**

**Protocolo de compromiso**

**Manejando las fallas**

**Recuperación y control de concurrencia**

**Compromiso de tres fases**



# Protocolos de compromiso

Asegurar atomicidad

Transacción T debe hacer commit o rollback en todos los sitios

El coordinador de transacción de T ejecuta un protocolo de compromiso



# Protocolos de compromiso

## Protocolos de compromiso

### Compromiso de dos Fases

Protocolo de compromiso

Manejando las fallas

Recuperación y control de concurrencia

### Compromiso de tres Fases



# Protocolo de compromiso de 2 fases

Considerar la transacción T, iniciada en el sitio Si, donde el coordinador de transacción es Ci.



# Protocolos de compromiso

**Protocolos de compromiso**

**Compromiso de dos Fases**

**Protocolo de compromiso**

**Manejando las fallas**

**Recuperación y control de concurrencia**

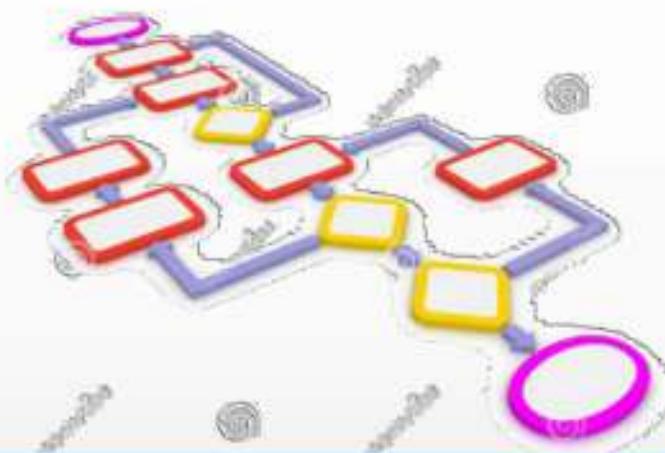
**Compromiso de tres Fases**



# Protocolo de compromiso

## FASE 1

- Ci añade el registro <prepare T>
- Ci envía el mensaje prepare T
- Determinar si Si desea hacer commit sobre T



Si la respuesta es no, se añade el registro <no T>

Se envía el mensaje abort T a Ci

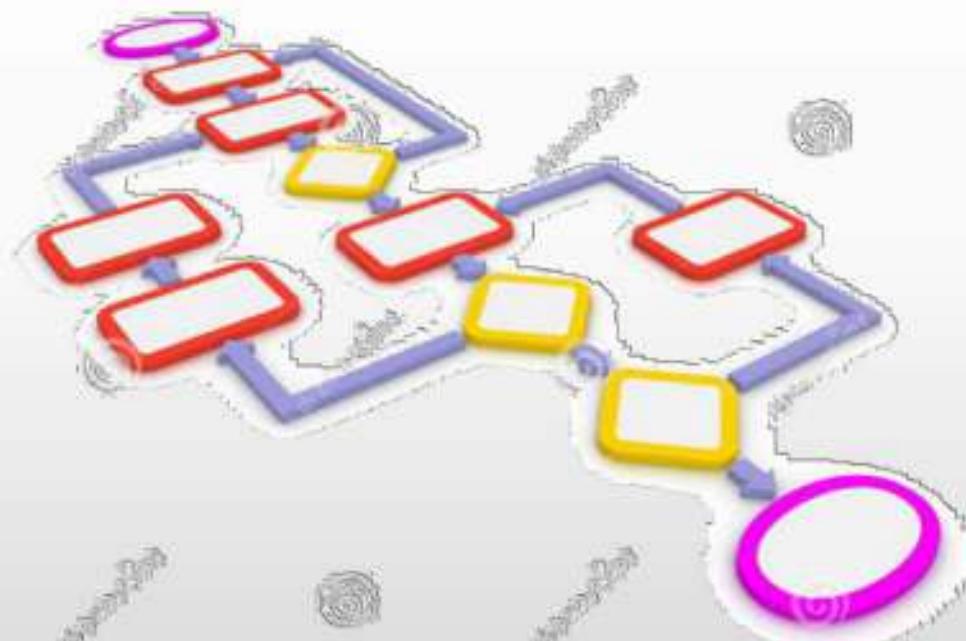
Si la respuesta es si, se añade el registro <ready T>

Se envía el mensaje ready T a Ci

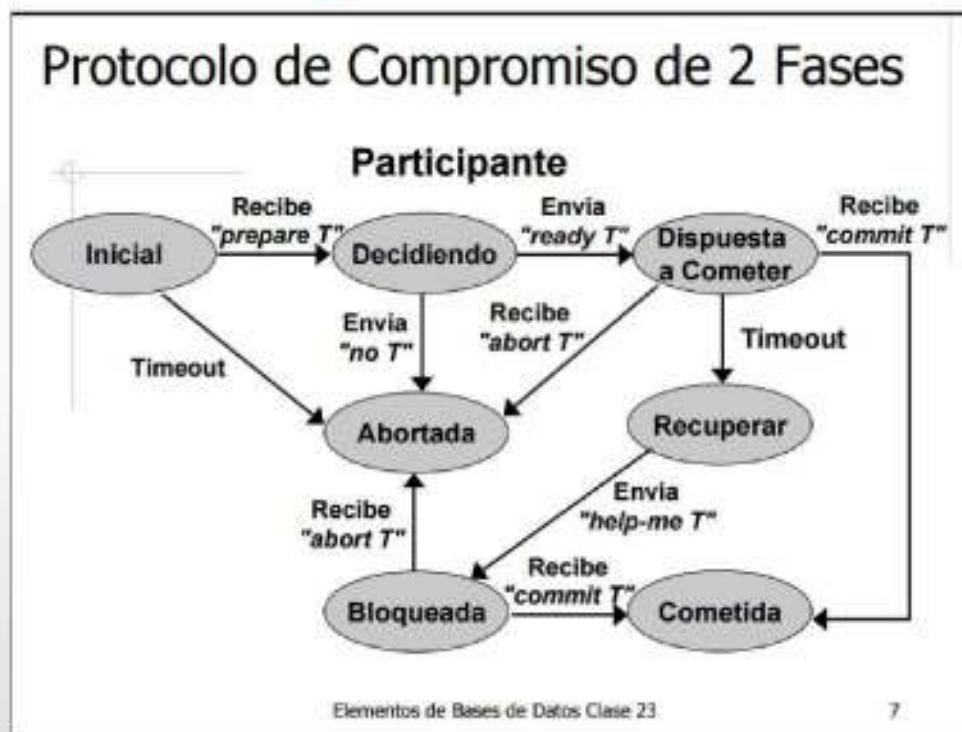
# Protocolo de compromiso

## FASE 2

- Ci determina si T puede hacer commit o abort
- T puede hacer commit si Ci recibió ready T de todos los participantes
- Ci adiciona el registro <commit T> o <abort T>
- Ci envía un mensaje commit T o abort T a todos los participantes
- Cada participante ingresa el mensaje recibido al log



# Protocolo de compromiso de 2 fases (2PC)

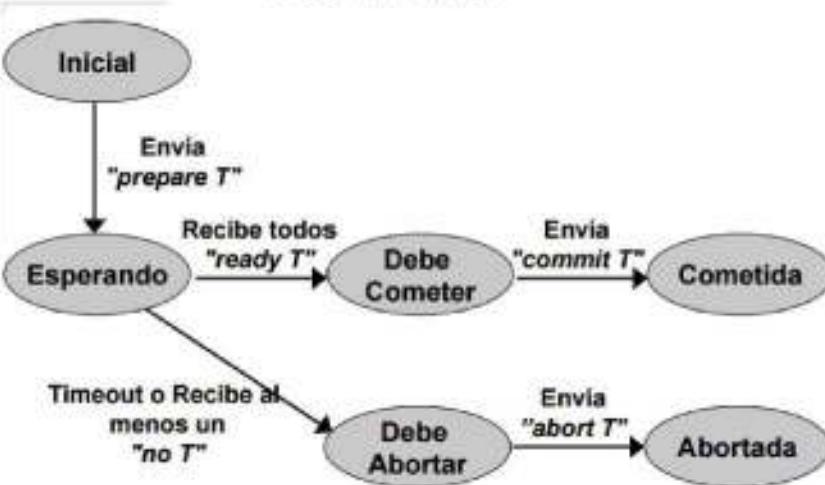


"Protocolo de compromiso de 2 fases "

# Protocolo de compromiso de 2 fases (2PC)

## Protocolo de Compromiso de 2 Fases

Coordinador



Elementos de Bases de Datos Clase 23

8

*"Protocolo de compromiso de 2 fases "*

# Protocolos de compromiso

**Protocolos de compromiso**

**Compromiso de dos Fases**

**Protocolo de compromiso**

**Manejando las fallas**

**Recuperación y control de concurrencia**

**Compromiso de tres Fases**



# Manejo de fallas

## FALLA EN EL SITIO PARTICIPANTE

- Si el sitio fallo antes de responder con un mensaje ready T a Ci -> se asume respuesta abort T
- Si el sitio fallo después de enviar un mensaje ready T -> se continua con la ejecución normal
- Cuando un sitio Sk se recupera
  - El log contiene un <commit T> -> <redo T>
  - El log contiene un <abort T> -> <undo T>
  - El log contiene un <ready T>



# Manejo de fallas

## FALLA EN EL SITIO PARTICIPANTE

- Cuando un sitio Sk se recupera
  - El log contiene un <ready T>
    - Si Ci esta activo -> Ci notifica el estado de T
      - Si el estado de T es commit -> Sk ejecuta <redo T>
      - Si el estado de T es abort -> Sk ejecuta <undo T>
    - Si Ci fallo
      - Sk envia mensaje querystatus a cada sitio participante
  - El log no contiene entradas de control -> Ci aborta T Y Sk ejecuta <undo T>



# Manejo de fallas



## FALLA EN EL COORDINADOR

- Los sitios participantes deben esperar la recuperación de Ci
- Si un sitio activo tiene una entrada <commit T> → T debe estar en estado commit
- Si un sitio activo tiene una entrada <abort T> → T debe estar en estado abort
- Si un sitio activo no tiene una entrada <ready T> → T se debe abortar
- Si no se presenta alguno de los casos anteriores
  - Todos los sitios activos deben tener una entrada <ready T> en su log → se debe esperar la recuperación de Ci

# Manejo de fallas



## PARTICIÓN DE RED

- Si Ci y todos los participantes permanecen en la misma partición
  - > La falla no tiene efecto en el protocolo
- Si Ci y los participantes permanecen en varias particiones
  - Los sitios que no están en la partición con Ci
    - > Ejecutan el protocolo para lidiar con el fallo de Ci
  - Los sitios que comparten la partición con Ci
    - >Continúan la ejecución normal del protocolo

# Protocolos de compromiso

## Protocolos de compromiso

Compromiso de dos Fases

Protocolo de compromiso

Manejando las fallas

Recuperación y control de concurrencia

Compromiso de tres Fases



# Recuperación y control de concurrencia

- Transacción dudosa

Se encuentra una entrada `<ready T>` pero no una entrada `<commit T>` o `<abort T>`

- Determinar el estado de las transacciones dudosas

- Si el coordinador falla y ningún sitio tiene información

-> la recuperación se bloquea -> Inutilización del sitio

- Algoritmos de recuperación

En vez de escribir `<ready T>` escriben `<ready L, T>`

- L es la lista de candados sostenidos por T

- Readquisición de candados sostenidos por T

- T puede empezar en el sitio incluso antes de determinar su estado



# Protocolos de compromiso

## Protocolos de compromiso

**Compromiso de dos Fases**

**Protocolo de compromiso**

**Manejando las Fallas**

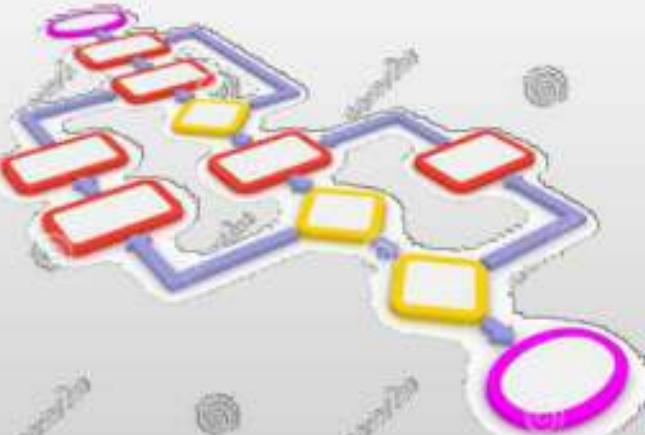
**Recuperación y control de concurrencia**

**Compromiso de tres Fases**

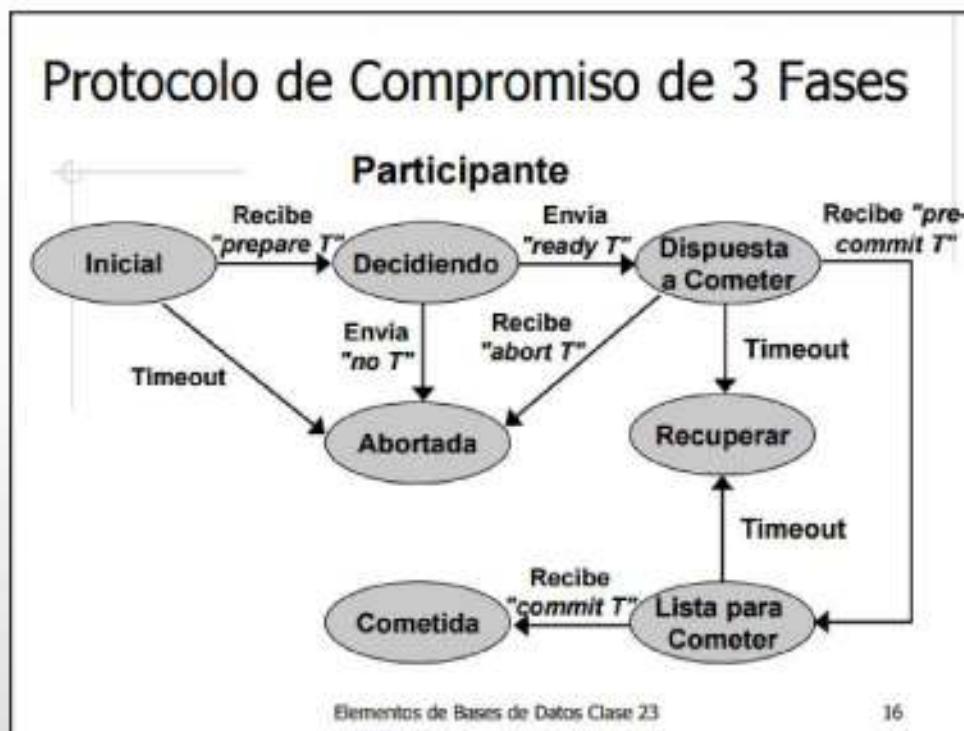


# Compromiso de 3 fases

- Extensión de 2PC
- Se asume que no hay partición de red
- No deben fallar mas de  $K$  sitio,  $K$  nivel de tolerancia
- Involucra una tercera fase donde involucra a varios sitios en la decisión de compromiso
- Ci asegura que  $k$  sitios sepan que se pretendía hacer commit sobre  $T$
- Si Ci Falla
  - Los sitios restantes eligen un nuevo coordinador
  - En caso contrario, Ci aborta  $T$



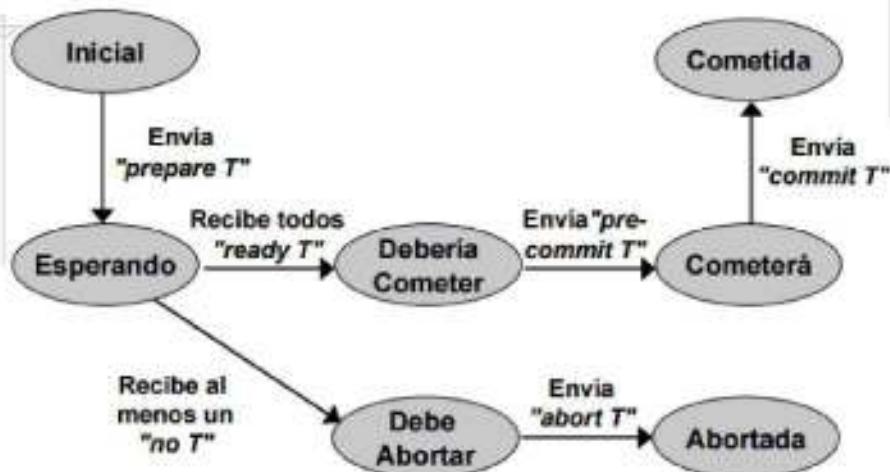
# Protocolo de compromiso de 3 fases (3PC)



"Protocolo de compromiso de 3 fases "

# Protocolo de compromiso de 3 fases (3PC)

Protocolo de Compromiso de 3 Fases  
Coordinador



Elementos de Bases de Datos Clase 23

17

"Protocolo de compromiso de 3 fases "

## 5. Control de la concurrencia en las bases de datos distribuidas

- Cada sitio participa en la ejecución de un protocolo de compromiso para asegurar la atomicidad global de las transacciones
- Los protocolos que se describen necesitan que se hagan actualizaciones de todas las réplicas de los elementos de datos

# 5.1. Protocolo de bloqueo

Se utilizan los mismos protocolos de bloqueo vistos en control de concurrencia. Pero se modifica el modo en que el gestor de bloqueos trata los datos replicados.

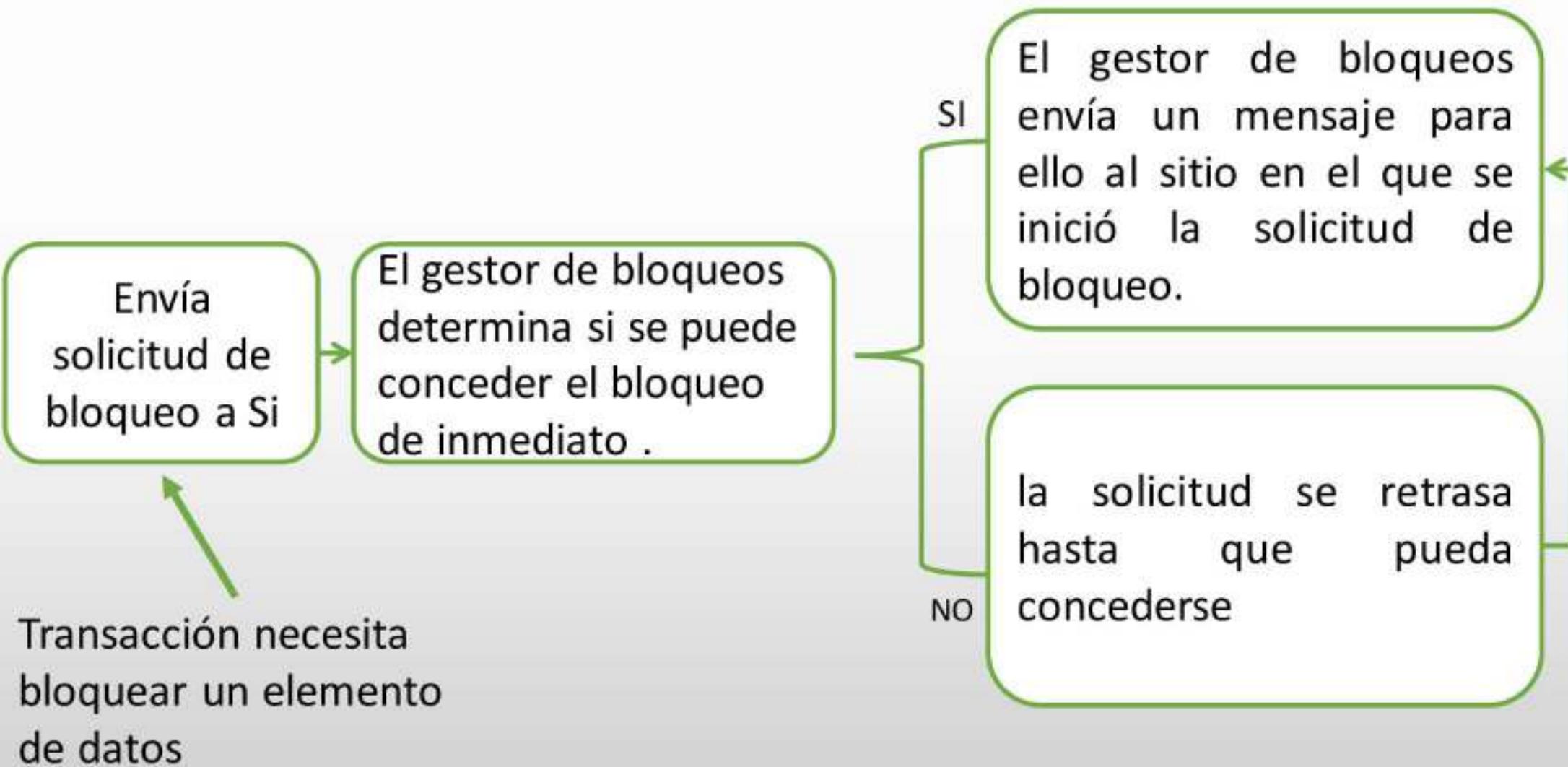
## Modos de bloqueo

**Compartido (locks-s):** Si una transacción  $T_i$  obtiene un **bloqueo en modo compartido** sobre el elemento  $Q$ , entonces  $T$  puede leer  $Q$  pero no lo puede escribir.

**Exclusivo (locks-x):** Si una transacción  $T_i$  obtiene un **bloqueo en modo exclusivo** sobre el elemento  $Q$ , entonces  $T$  puede tanto leer como escribir  $Q$ .

## **5.1.1. Enfoque de gestor único de bloqueos**

- ❖ Un gestor único de bloqueos reside en un sitio único (*Si*).
- ❖ Todas las solicitudes de bloqueo y desbloqueo se realizan en *Si*.



## **Enfoque de gestor único de bloqueos**

### **VENTAJAS**

- ❖ **Implementación sencilla**
- ❖ **Tratamiento sencillo de los interbloqueos**

### **DESVENTAJAS**

- ❖ **Cuello de botella**
- ❖ **Vulnerabilidad**

## **5.1.2. Gestor distribuido de bloqueos**

- ❖ La función del gestor de bloqueos se halla distribuida entre varios sitios.
- ❖ Cada sitio mantiene un gestor de bloqueos local

Transacción desea bloquear un elemento de datos, que no está replicado y reside en el sitio Si

Se envía un mensaje al gestor de bloqueos del sitio Si para solicitarle un bloqueo.

Si el elemento de datos está bloqueado en un modo incompatible

SI

la solicitud se retrasa hasta que pueda concederse

NO

El gestor de bloqueos devuelve un mensaje al sitio que ha iniciado la solicitud que indica que ha concedido la solicitud de bloqueo

# **Gestor distribuido de bloqueos**

## VENTAJAS

- ❖ Implementación sencilla
- ❖ Reduce el grado en el que el coordinador constituye un cuello de botella.
- ❖ Tiene una sobrecarga razonablemente baja

## DESVENTAJAS

- ❖ El tratamiento de los interbloqueos resulta más complejo



### 5.1.3. Copia principal

- ❖ Cuando un sistema utiliza la réplica de datos se puede escoger una de las réplicas como **copia principal**.
- ❖ Para Cada elemento de datos Q la copia principal Q debe residir exactamente en un sitio principal de Q.

Solicita un bloqueo en el sitio principal de  $Q$

Se puede conceder el bloqueo.

SI

El gestor de bloqueos devuelve un mensaje al sitio que ha iniciado la solicitud que indica que ha concedido la solicitud de bloqueo

NO

la solicitud se retrasa hasta que pueda concederse

Cuando una transacción desea bloquear un elemento de datos

# **Copia principal**

## **VENTAJAS**

**Implementación sencilla**

## **DESVENTAJAS**

**Si falla el sitio principal de q, q queda inaccesible**

## **5.1.4. Protocolo de mayoría**

En todos los sitios donde se almacenan elementos de datos existen administradores de bloqueo

Si el elemento de datos  $Q$  se replica en  $n$  sitios diferentes

El mensaje de solicitud de bloqueo se envía a más de la mitad de los  $n$  sitios en donde se almacena  $Q$ .

El gestor de bloqueos determina si se puede conceder el bloqueo de inmediato.

SI

El gestor de bloqueos devuelve un mensaje al sitio que ha iniciado la solicitud que indica que ha concedido la solicitud de bloqueo

NO

la solicitud se retrasa hasta que pueda concederse

NOTA: La transacción no se opera en  $Q$  hasta que logre obtener un bloqueo en la mayoría de las réplicas de  $Q$ .

# **Protocolo de mayoría**

## **VENTAJAS**

- ❖ Puede extender para tratar los fallos de los sitios
- ❖ Evita los inconvenientes del control centralizado

## **DESVENTAJAS**

- ❖ **IMPLEMENTACIÓN:** necesita  $2(n / 2 + 1)$  mensajes para manejar las solicitudes de bloqueo y  $(n / 2 + 1)$  mensajes para manejar las solicitudes de desbloqueo
- ❖ Tratamiento de los interbloqueos

## **5.1.5. Protocolo sesgado**

Se concede un tratamiento más favorable a las solicitudes de bloqueos compartidos que a las solicitudes de bloqueos exclusivos

# Protocolo Sesgado

**Bloqueos compartidos:**  
Solicita un bloqueo de  $Q$  al gestor de bloqueos de un sitio que contenga una réplica de  $Q$ .

**Bloqueos exclusivos:**  
Solicita un bloqueo de  $Q$  al gestor de bloqueos de todos los sitios que contienen una réplica de  $Q$ .

El gestor de bloqueos determina si se puede conceder el bloqueo de inmediato.

SI  
El gestor de bloqueos devuelve un mensaje al sitio que ha iniciado la solicitud que indica que ha concedido la solicitud de bloqueo

NO  
la solicitud se retrasa hasta que pueda concederse

# **Protocolo Sesgado**

## **VENTAJAS**

**Imponer menos sobrecarga  
a las operaciones de lectura**

## **DESVENTAJAS**

- ❖ **Sobrecarga adicional sobre las operaciones de escritura**
- ❖ **Tratamiento de los interbloqueos**

## 5.1.6. Protocolo de consenso de quórum

- Asigna a cada sitio un peso no negativo  $S$ .
- Asigna a las operaciones de lectura y de escritura del elemento  $x$  dos enteros, denominados **quórum de lectura**  $Q_r$  y **quórum de escritura**  $Q_w$ , que deben cumplir:

$$Q_r + Q_w > S \text{ y } 2 * Q_w > S$$

- Para ejecutar la operación:
- **LECTURA (read):**deben Bloquearse suficientes réplicas para que su peso total sea  $\geq Q_r$ ,
  - **Escritura (Write):** bloquear suficientes réplicas como para que su peso total sea  $\geq Q_w$ .

**Ventaja:** Permite reducir de manera selectiva el coste de las operaciones de bloqueo de lectura o de escritura

## 5.2. Marcas temporales

- ❖ Se concede a cada transacción una marca temporal *única* que el sistema utiliza para decidir el orden de secuenciación

Métodos para generar marcas temporales únicas

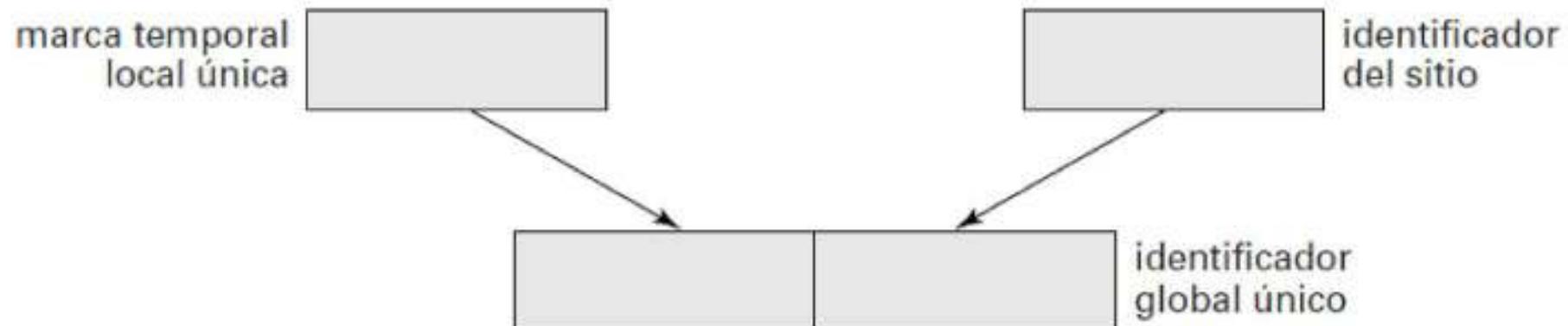
**Esquema centralizado**

Un solo sitio distribuye las marcas temporales

**Esquema distribuido**

Cada sitio genera una marca temporal local única

# Marcas temporales [Generación de marca temporal global única]



Fuente: Silberschatz, Abraham, Henry F. Korth, and S. Sudarshan, *Database System Concepts*, 4th ed (Boston: McGraw-Hill, 2002)

# Marcas temporales [Problemas en la generación de marcas temporales]

Si un sitio genera marcas temporales locales a una velocidad mayor que la de los demás sitios entonces:

se necesita es un mecanismo que asegure que las marcas temporales locales se generen de manera homogénea en todo el sistema

En cada sitio  $S_i$  se define:

un **reloj lógico** ( $LC_i$ ), que genera la marca temporal local única

## **Marcas temporales**

### **Asegurar que los relojes lógicos estén sincronizados**

- El sitio  $S_i$  adelante su reloj lógico siempre que una transacción  $T_j$  con la marca temporal  $\langle x, y \rangle$  visite ese sitio.
  - $x$  sea mayor que el valor actual de  $LC_i$ .
- En este caso el sitio  $S_i$  adelanta su reloj lógico al valor  $x + 1$ .

Nota: Si se utiliza el reloj del sistema para generar marcas temporales, éstas se asignarán de manera homogénea.

## 5.3. Réplica con grado de consistencia bajo

**Replica Maestro-esclavo:** la base de datos permite las actualizaciones en el sitio principal y las propaga de manera automática a las réplicas de los demás sitios.

- Útil para distribuir información
- creación de copias de la base de datos para ejecutar consultas de gran tamaño
- Las actualizaciones deben propagarse de manera periódica de modo que la propagación no interfiera con el procesamiento de las consultas.

## **Réplica con grado de consistencia bajo**

**réplica multamaestro(réplica de actualización distribuida)** :se permiten las actualizaciones en cualquier réplica de los elementos de datos y se propagan de manera automática a todas las réplicas.

Actualización de las réplicas : se aplica la actualización inmediata con el compromiso de dos fases, utilizando una de las técnicas de control de la concurrencia distribuida.

Protocolo sesgado: Las operaciones de **escritura** tienen que bloquear y actualizar todas las réplicas y las operaciones de **lectura** bloquean y leen cualquier réplica

## Réplica con grado de consistencia bajo [Actualización de replicas]

- **Propagación perezosa** : permiten que continúe el procesamiento de las transacciones (incluidas las actualizaciones) aunque un sitio quede desconectado de la red.

Las actualizaciones de las réplicas se traducen en actualizaciones del sitio principal, que se propagan luego de manera perezosa a todas las réplicas

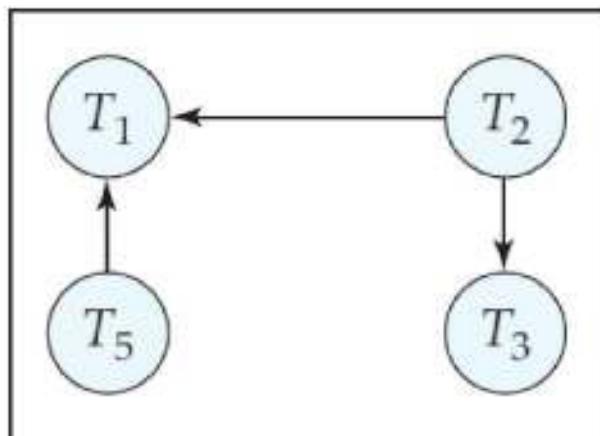
Las actualizaciones se llevan a cabo en cualquier réplica y se propagan a todas las demás.

## 5.4. Tratamiento de los interbloqueos

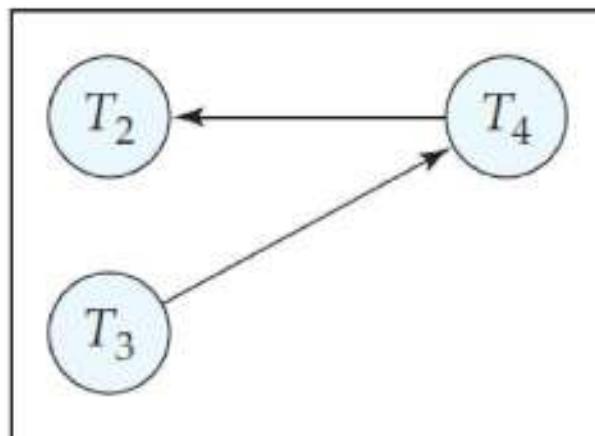
- Si se permite que los interbloqueos se produzcan y se confía en su detección, el problema principal en los sistemas distribuidos es la decisión del modo en que se mantiene el grafo de espera.
- Las técnicas para tratar este problema exigen que cada sitio guarde un **grafo local de espera**.

# Grafos locales de espera

## Ejemplo.



site  $S_1$



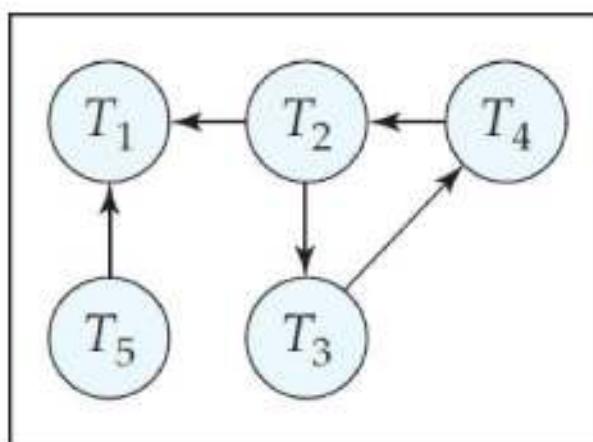
site  $S_2$

NODOS: transacciones  
(locales y no locales)

**Figure 19.4** Local wait-for graphs.

Fuente: Silberschatz, Abraham, Henry F. Korth, and S. Sudarshan, *Database System Concepts*, 6th ed (New York: McGraw-Hill, 2011)

# Grafo global de espera



**Figure 19.5** Global wait-for graph for Figure 19.4.

Fuente: Silberschatz, Abraham, Henry F. Korth, and S. Sudarshan, *Database System Concepts*, 6th ed (New York: McGraw-Hill, 2011)

## Ejemplo:

FIGURA 12.8

Grafos de espera para la detección de candado mortal

Tiempo	Sitio1	Sitio2	Sitio3	Sitio4
t1	T1: Xbloqueo a	T2: Sbloqueo g	T3: Sbloqueo m	T4: Xbloqueo q
t2	T1: Xbloqueo b	T2: Xbloqueo h	T3: Xbloqueo n	T4: Sbloqueo r
t3	T2: solicita Sbloqueo a	T3: solicita Xbloqueo g	T4: solicita Sbloqueo n	T1: solicita Sbloqueo q
t4	... T2espera ...	... T3espera ...	... T4espera ...	... T1espera ...

FIGURA 12.8(a)

Programación global de las transacciones T1, T2, T3 y T4

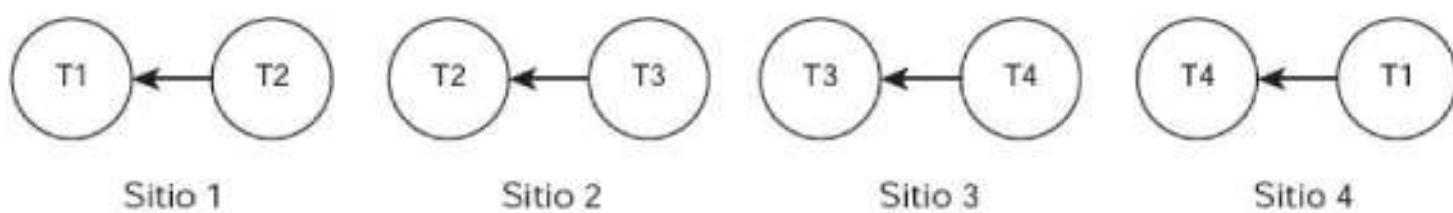
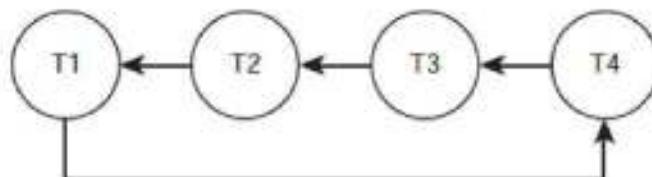


FIGURA 12.8(b)

Grafos de espera locales sin ciclos

FIGURA 12.8(c)

Grafo de espera global con ciclo



Fuente: Ricardo, Catherine M., Víctor Campos Olgún, and Javier Enríquez Brito, *Bases de datos* (México: McGraw-Hill, 2009)

# Tratamiento de los interbloqueos

- Detección centralizada de interbloqueos: el sistema crea y mantiene un **grafo global de espera** en un solo sitio: el coordinador de detección de interbloqueos.

## TIPOS DE GRAFOS DE ESPERA

**grafos reales:** describen el estado real pero desconocido del sistema

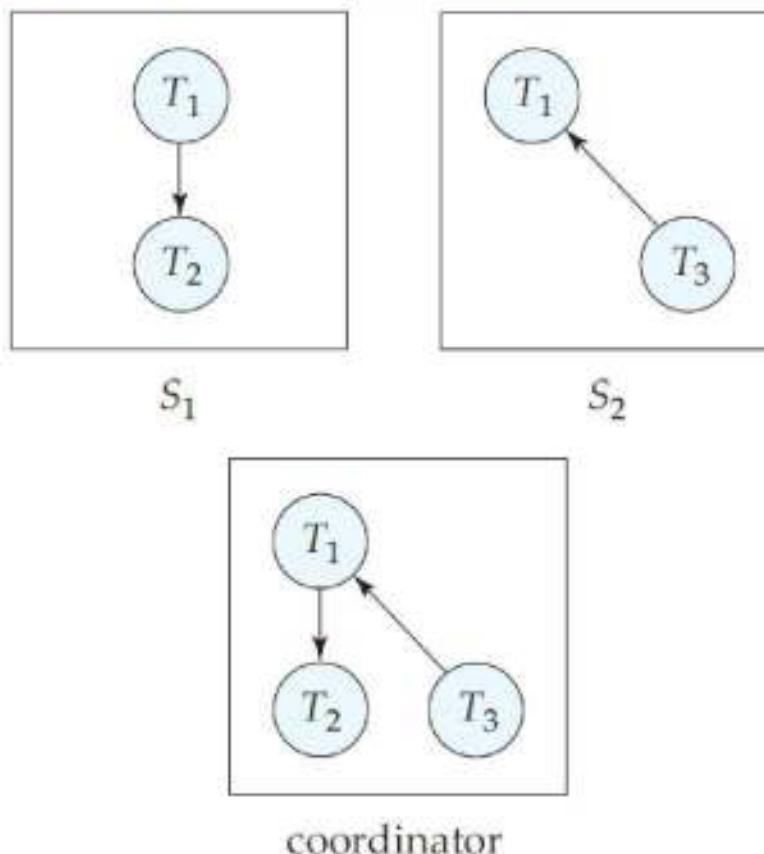
**Grafos Creados:** son una aproximación generada por el controlador durante la ejecución del algoritmo del controlador

## **Condiciones para volver a crear el grafo global de espera:**

- Siempre que se introduzca o se elimine un nuevo arco en alguno de los grafos locales de espera.
- De manera periódica, cuando se hayan producido varias modificaciones en un grafo local de espera.
- Siempre que el coordinador necesite invocar el algoritmo de detección de ciclos.

# Retrocesos innecesarios

## 1. ciclos falsos en el grafo global de espera

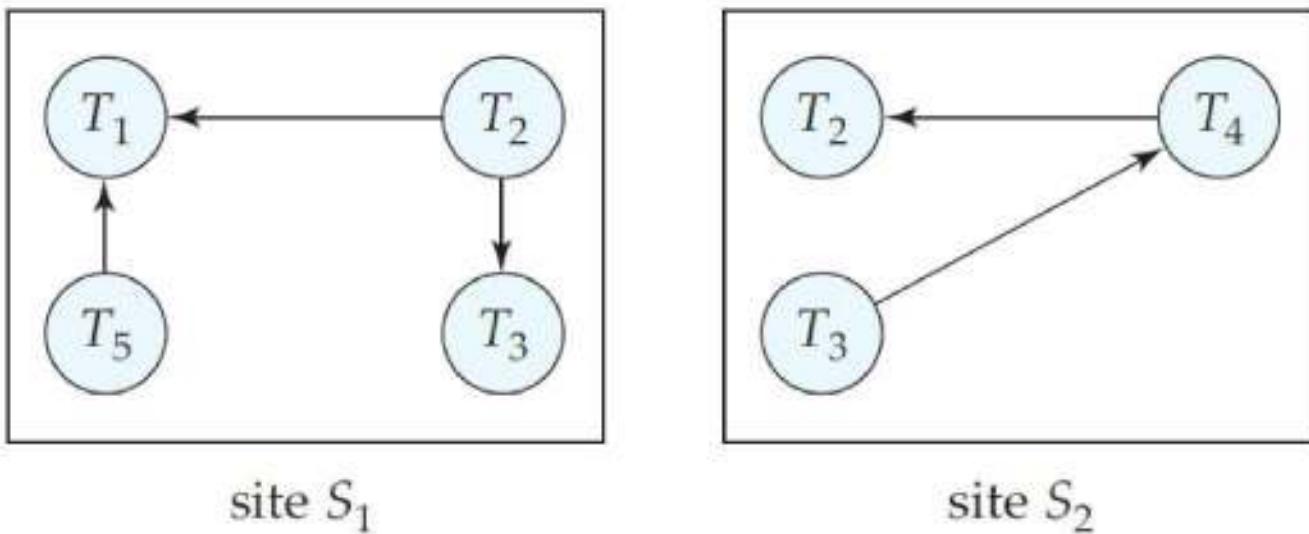


**Figure 19.6** False cycles in the global wait-for graph.

Fuente: Silberschatz, Abraham, Henry F. Korth, and S. Sudarshan, *Database System Concepts*, 6th ed (New York: McGraw-Hill, 2011)

## Retrocesos innecesarios

2. Se produce de verdad un *interbloqueo* y se escoge una víctima cuando se aborta alguna de las transacciones por motivos no relacionados con interbloqueos.



**Figure 19.4** Local wait-for graphs.

Fuente: Silberschatz, Abraham, Henry F. Korth, and S. Sudarshan, Database System Concepts, 6th ed (New York: McGraw-Hill, 2011)

## 6. DISPONIBILIDAD

### Disponibilidad

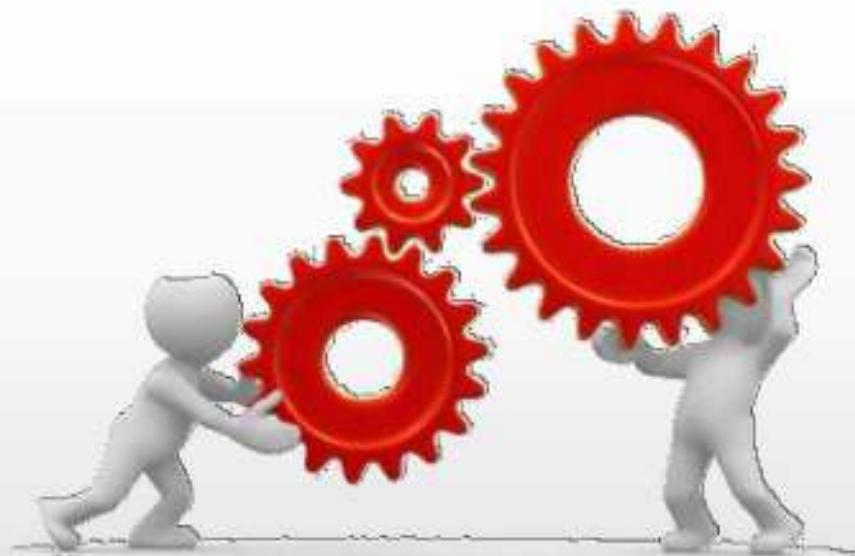
- Enfoque basado en la mayoría
- Enfoque leer uno, escribir en todos
- Reintegración del sitio
- Comparación con Backup remoto
- Selección Coordinador
- Consistencia por disponibilidad



# DISPONIBILIDAD

## Disponibilidad

- Enfoque basado en la mayoría
- Enfoque leer uno, escribir en todos
- Reintegración del sitio
- Comparación con Backup remoto
- Selección Coordinador
- Consistencia por disponibilidad



# DISPONIBILIDAD

- La base de datos debe Funcionar casi todo el tiempo
- Robustez: Capacidad de seguir Funcionando ante varios tipos de Fallos

## Tipos de fallos

- Perdida de mensajes
- Fallo en el enlace
- Partición de la red



# DISPONIBILIDAD

## Disponibilidad

Enfoque basado en la mayoría

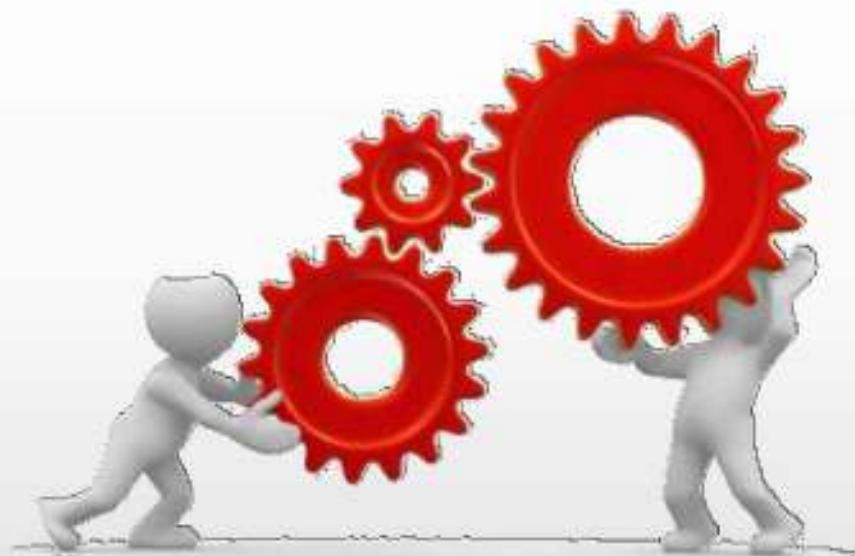
Enfoque leer uno, escribir en todos

Reintegración del sitio

Comparación con Backup remoto

Selección Coordinador

Consistencia por disponibilidad



# Enfoque basado en la mayoría

- Cada objeto de datos se almacena con un numero de versión para detectar cuando fue la ultima escritura

Actualización del numero de versión

- Enviar un mensaje de lock-request a mas de la mitad de los sitios en donde se encuentra replicado el objeto de datos
- La transacción no opera sobre el objeto de datos hasta obtener una respuesta lock de la mayoría de las replicas del objeto de datos



# Enfoque basado en la mayoría

- Operaciones de lectura
  - Lee el lock de la replica que tiene el numero de versión mas alto
  - Opcionalmente se puede escribir el valor a las replicas con numero de versión mas bajo
- Operación de escritura
  - Escribe sobre todas las replicas de las cuales se obtuvo un lock y actualiza el numero de versión



# DISPONIBILIDAD

## Disponibilidad

**Enfoque basado en la mayoría**

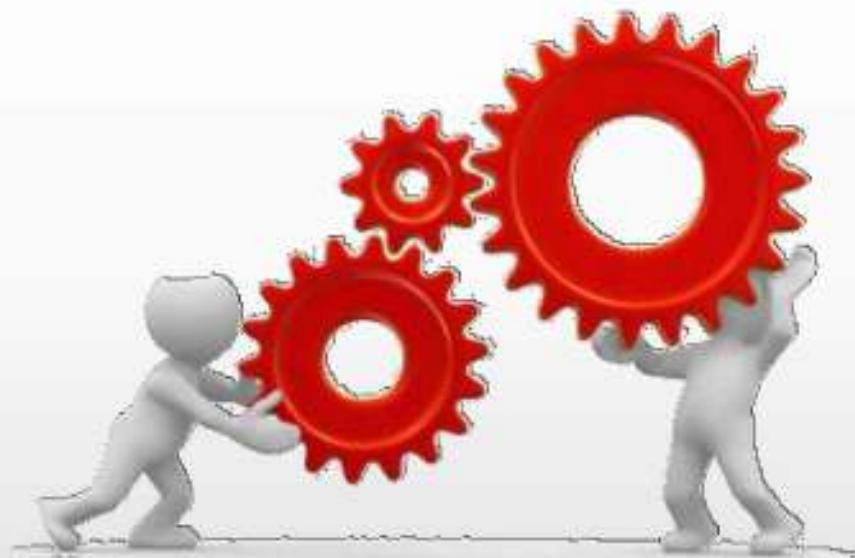
**Enfoque leer uno, escribir en todos**

**Reintegración del sitio**

**Comparación con Backup remoto**

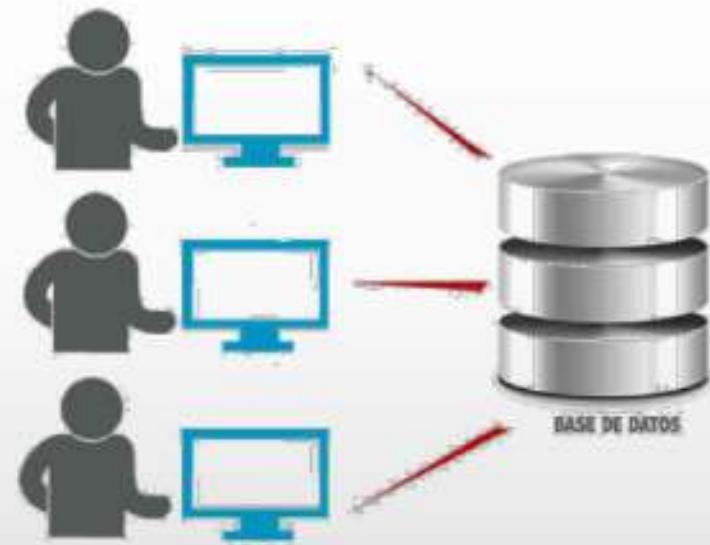
**Selección Coordinador**

**Consistencia por disponibilidad**



# Enfoque leer uno, escribir en todos

- Asignar pesos unitarios a los sitios
- El quorum de lectura es 1
- El quorum de escritura es n
- No hay necesidad de utilizar numero de version
- Si un sitio que contiene el elemento de datos falla  
→ La escritura no puede proceder porque el quorum de escritura no esta disponible



# DISPONIBILIDAD

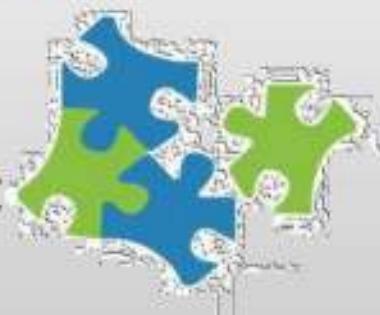
## Disponibilidad

- Enfoque basado en la mayoría
- Enfoque leer uno, escribir en todos
- Reintegración del sitio
- Comparación con Backup remoto
- Selección Coordinador
- Consistencia por disponibilidad



# Reintegración del sitio

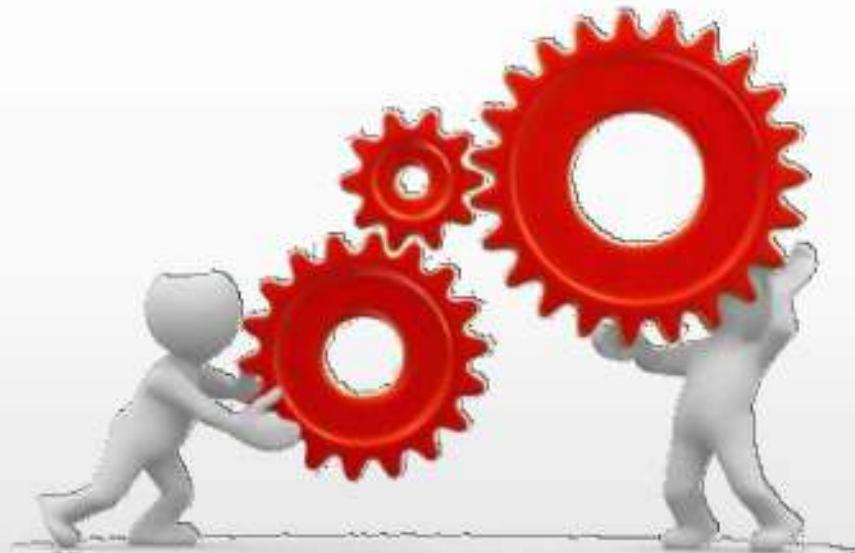
- Cuando un sitio se recupera se deben actualizar sus tablas para reflejar los cambios realizados mientras estaba caido.
- Si el sitio tiene replicas de objetos de datos, debe obtener los valores actuales.
- La reintegración puede ser complicada debido a que los objetos de datos procesados pueden sufrir cambio mientras el sitio se esta actualizando.
- Una solución es detener el sistema mientras el sitio se recupera.
- Antes de que un bloqueo se otorgue, el sitio debe asegurar que se ha puesto al dia con la actualización de los datos.



# DISPONIBILIDAD

## Disponibilidad

- Enfoque basado en la mayoría
- Enfoque leer uno, escribir en todos
- Reintegración del sitio
- Comparación con Backup remoto
- Selección Coordinador
- Consistencia por disponibilidad



# Comparación con Backup remoto

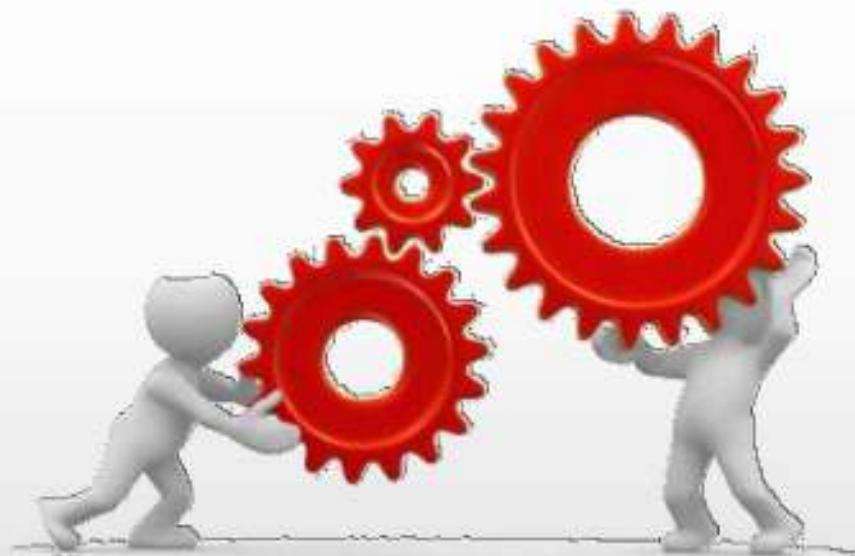
- Enfoques para proporcionar alta disponibilidad
- El control de concurrencia y recuperación se realiza en un solo sitio
- Las transacciones se comunican con un solo sitio
- Ofrecen un enfoque de bajo costo para la alta disponibilidad en comparación con las bases de datos distribuidas



# DISPONIBILIDAD

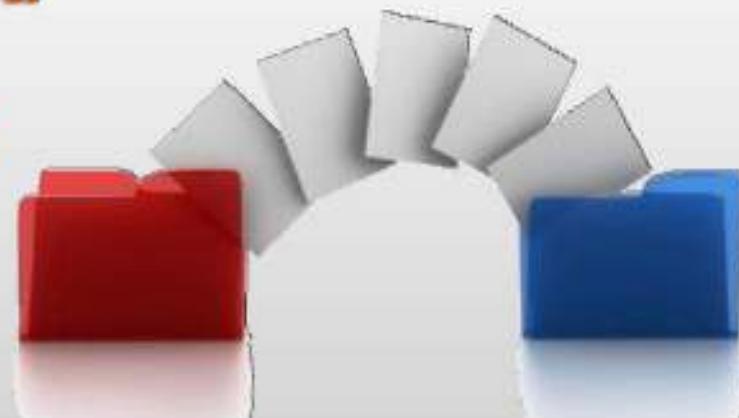
## Disponibilidad

- Enfoque basado en la mayoría
- Enfoque leer uno, escribir en todos
- Reintegración del sitio
- Comparación con Backup remoto
- Selección Coordinador
- Consistencia por disponibilidad



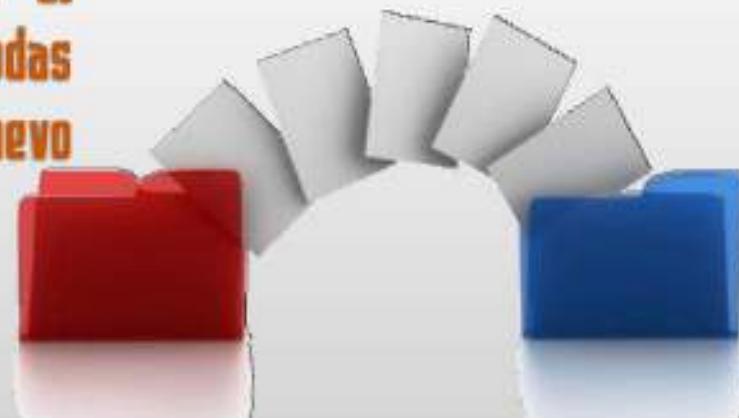
# Selección del coordinador

- Una manera de reemplazar al coordinador fallido es mediante un coordinador de copia de seguridad
- El coordinador de copia de seguridad es un sitio que mantiene la información necesaria para asumir el papel de coordinador
- Todos los mensajes dirigidos al coordinador se reciben tanto por el coordinador como por su copia de seguridad
- La copia de seguridad no toma ninguna acción que afecte a otros sitios



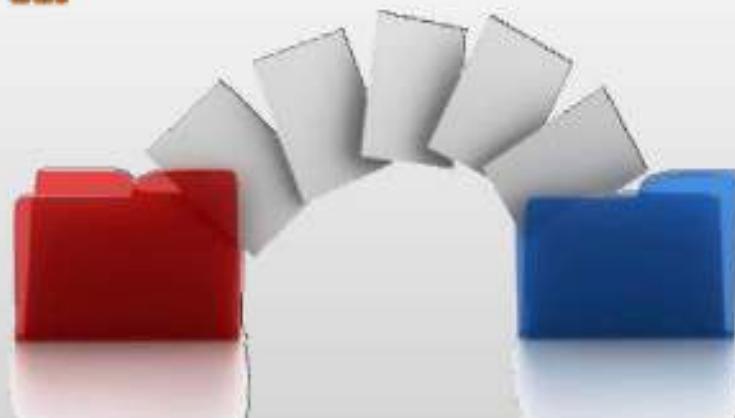
# Selección del coordinador

- Si una copia de seguridad no está preparada, se nombre un coordinador y este debe recobrar información de todos los sitios en el sistema para ejecutar las tareas de coordinador
- En algunas ocasiones la única fuente de información es el coordinador fallido por lo cual se es necesario abortar una o todas las transacciones y reiniciarlas bajo el control del nuevo coordinador



# Selección del coordinador

- La comunicación entre el coordinador y su copia de seguridad debe ser constante para asegurar que sus actividades estén sincronizadas.
- Los algoritmos de elección requieren que cada sitio activo del sistema tenga asociado un número de identificación único.



# Selección del coordinador

## ALGORITMO BULLY

- Asume que el numero de identificación del sitio Si es i
- El sitio elegido es aquel con el identificador mas grande
- El algoritmo debe proporcionar un mecanismo por el cual el sitio que se recupera de un accidente pueda identificar al coordinador actual.



# Selección del coordinador

## ALGORITMO BULLY

- El sitio Si envía un mensaje de elección a cada sitio con numero de identificación superior
- Si espera una respuesta un tiempo T
- Si no recibe respuesta
  - > Se asume que todos los sitios Fallaron
  - > Se elige a si mismo como nuevo coordinador
- Si recibe una respuesta



# Selección del coordinador

## ALGORITMO BULLY

- Si recibe una respuesta
  - >Comienza un tiempo T
  - >Espera recibir un mensaje informándole de que un sitio con numero de identificación superior ha sido elegido
- Si no recibe respuesta en el tiempo T
  - >Si asume que el sitio fallo
  - >Si reinicia el algoritmo



# Selección del coordinador

## ALGORITMO BULLY [Algoritmo agresor]

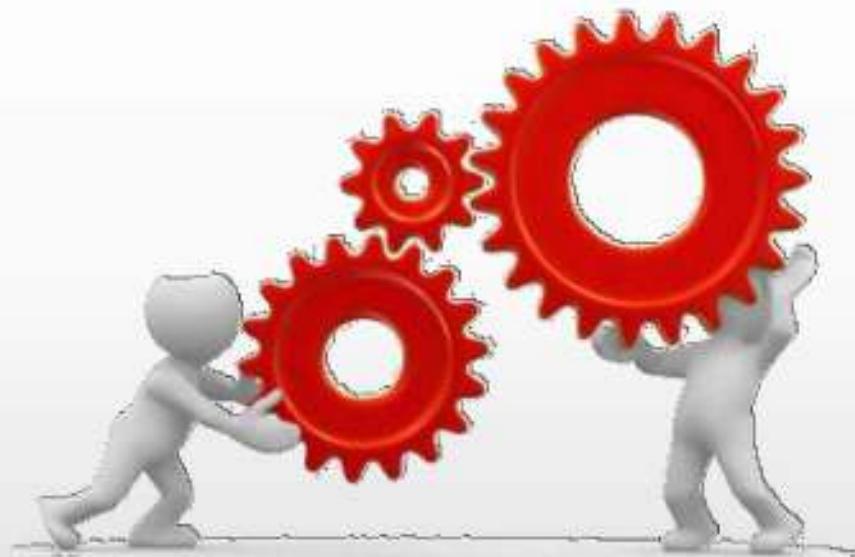
- Despues de que un sitio se recupera, el algoritmo se ejecuta automáticamente
- Si no hay sitios activos con números de identificación superior, el sitio recuperado se convierte en el nuevo coordinador incluso si hay un coordinador activo.



# DISPONIBILIDAD

## Disponibilidad

- Enfoque basado en la mayoría
- Enfoque leer uno, escribir en todos
- Reintegración del sitio
- Comparación con Backup remoto
- Selección Coordinador
- Consistencia por disponibilidad



# Consistencia por disponibilidad

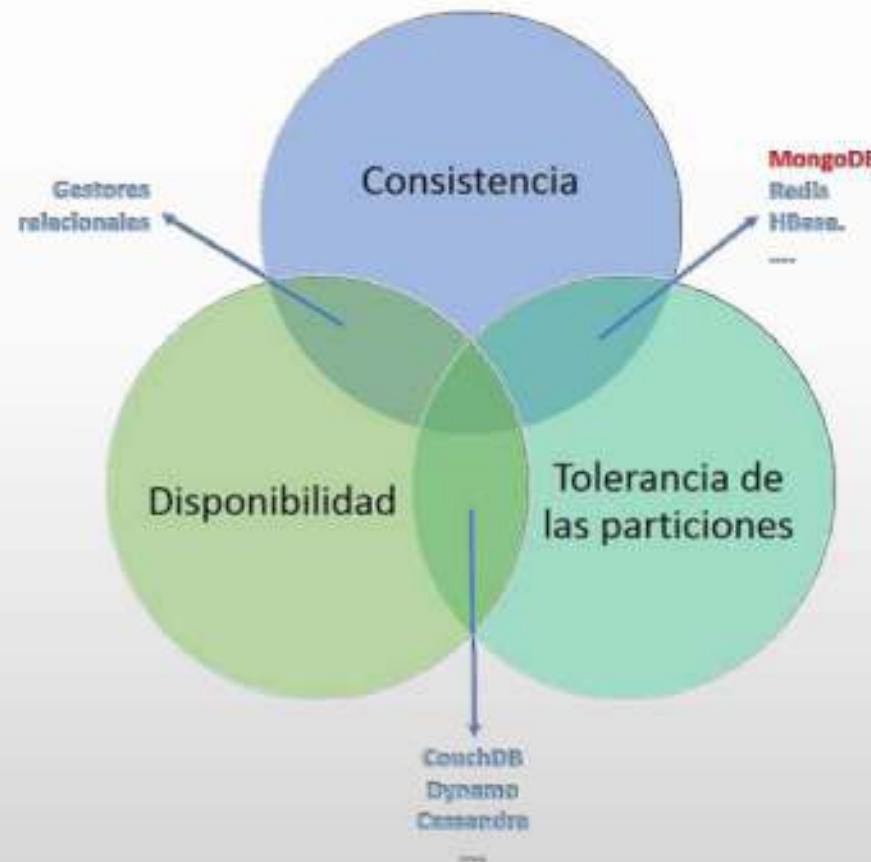
- Si existe un fallo de red con mas de 2 particiones  
→ Ninguna partición tiene la mayoría de los sitios  
→ No se puede procesar actualizaciones
- El protocolo de leer uno y escribir en todos los disponibles proporciona disponibilidad pero no consistencia.



# Consistencia por disponibilidad

## TEOREMA CAP

- Establece que una base de datos distribuida solo puede tener dos de las siguientes propiedades:
  1. Consistencia
  2. Disponibilidad
  3. Tolerancia a la partición
- Las particiones de la red no se pueden prevenir por lo cual se debe sacrificar disponibilidad o consistencia



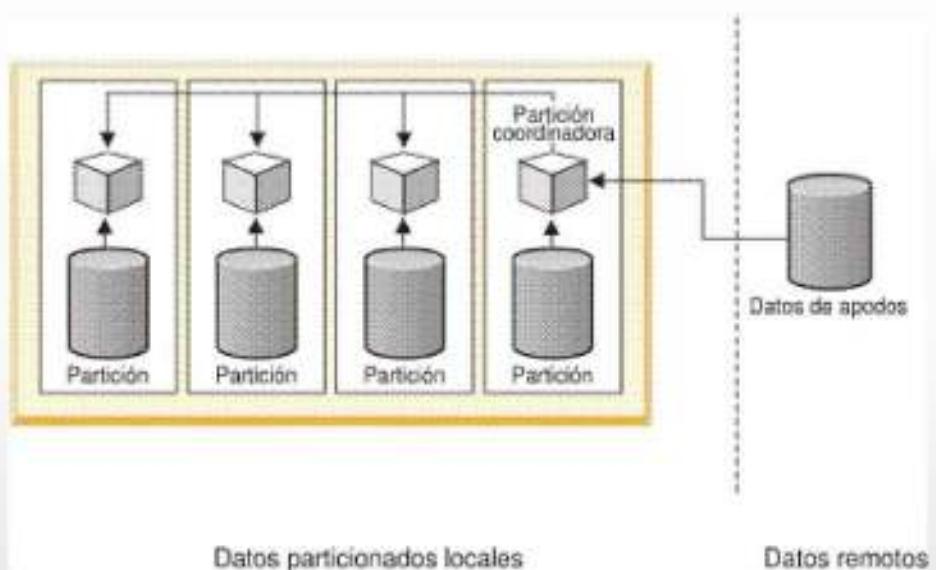
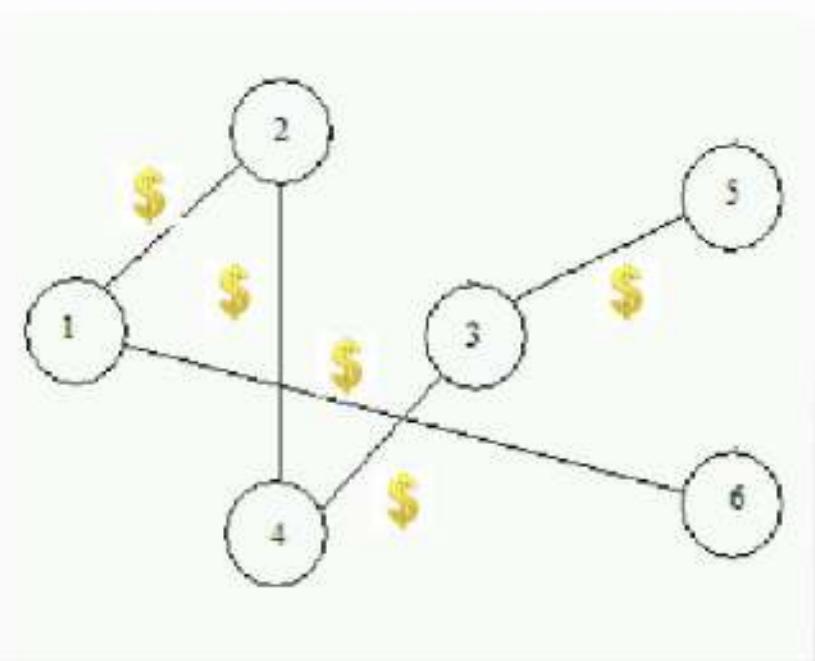
# Consistencia por disponibilidad

## PROPIEDADES BASE

- Básicamente disponible [A costa de la consistencia]**  
Permite la actualización incluso en el caso de particiones
- Estado soft**  
El estado de la base de datos no puede definirse con precisión
- Eventualmente consistente**  
Solucionada la partición, todas las replicas deben ser consistentes entre si



# 7. Procesamiento distribuido de consultas



<http://structio.sourceforge.net/guias/proglín/proglín006.gif>

[http://www-01.ibm.com/support/knowledgecenter/SSEPGG\\_9.5.0/com.ibm.swg.im.iis.fed.tuning.doc/images/ftpprt01b.gif](http://www-01.ibm.com/support/knowledgecenter/SSEPGG_9.5.0/com.ibm.swg.im.iis.fed.tuning.doc/images/ftpprt01b.gif)

Silberschatz, A., Forth, HF., Sudarshan, S. (2009). *Database System Concepts*. Editorial. McGraw-Hill. 6a ed. Estados Unidos. 13

# Transformación de consultas

Fácil: "Buscar todas las tuplas de la relación cuenta"



¿Por qué siempre piden lo más difícil?

$\sigma_{nombre\_rama="Hillside"}(cuenta)$   
 $\sigma_{nombre\_rama="Hillside"}(cuenta_1 \cup cuenta_2)$



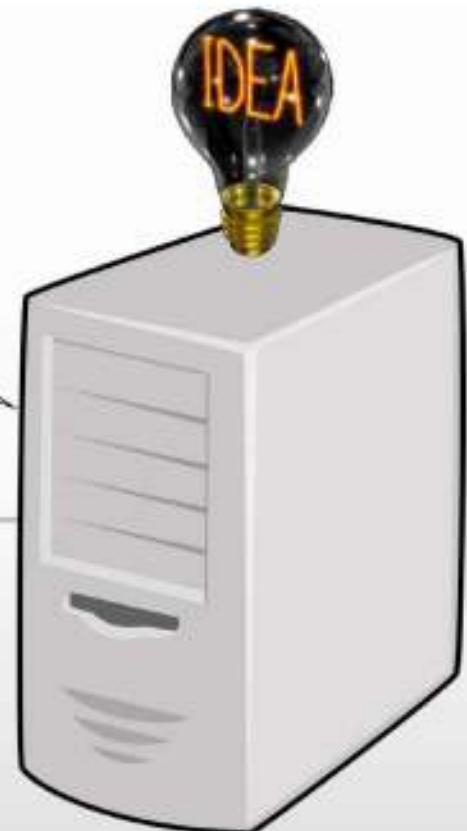
Transparencia en la fragmentación

$\sigma_{nombre\_rama="Hillside"}(cuenta_1) \cup \sigma_{nombre\_rama="Hillside"}(cuenta_2)$

[https://pixabay.com/p-567943/?no\\_redirect](https://pixabay.com/p-567943/?no_redirect)

<http://2.bp.blogspot.com/-vn-dS20cEp8/U4nuPKy12I/AAAAAAAABCc/iOMJyFMyVm/g/s400/Credito.jpg>

Silberschatz, A., Forth, HF., Sudarshan, S. (2009). Database System Concepts. Editorial. McGraw-Hill. 6a ed. Estados Unidos. 13



# Sencillo procesamiento Join



Opción 1



Opción 2



Opción 3

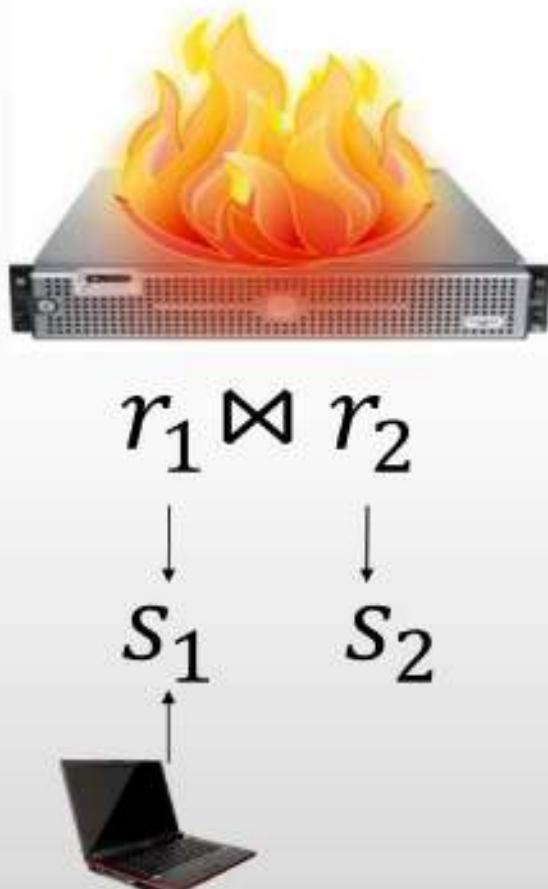
- ✓ No replicadas
- ✓ No fragmentadas

<http://d-maps.com/m/america/colombia/colombie/colombie42s.gif>

[https://encrypted-tbn2.gstatic.com/images?q=tbn:ANd9GcQ0vMRy5Dy\\_dCPgInnjtkVXceWRK0Sepa8gnXuMnlmqbjil8LcntzYX2Tc](https://encrypted-tbn2.gstatic.com/images?q=tbn:ANd9GcQ0vMRy5Dy_dCPgInnjtkVXceWRK0Sepa8gnXuMnlmqbjil8LcntzYX2Tc)

Silberschatz, A., Forth, HF., Sudarshan, S. (2009). Database System Concepts. Editorial. McGraw-Hill. 6a ed. Estados Unidos. 13

# Estrategias semi-join



*Computar  $temp_1 \leftarrow \pi_{R_1 \cap R_2}(r_1)$  en  $S_1$*

*Enviar  $temp_1$  de  $S_1$  a  $S_2$*

*Computar  $temp_2 \leftarrow r_2 \bowtie temp_1$  en  $S_2$*

*Enviar  $temp_2$  de  $S_2$  a  $S_1$*

*Computar  $r_1 \bowtie temp_2$  en  $S_1$*

## Estrategias Join que explotan el paralelismo

$$r_1 \bowtie r_2 \bowtie r_3 \bowtie r_4$$

$$(r_1 \bowtie r_2) \bowtie (r_3 \bowtie r_4)$$

# 8. BASES DE DATOS DISTRIBUIDAS HETEROGÉNEAS

**Bases de datos distribuidas heterogéneas**

**Vista de datos unificada**

**Procesamiento de consultas**

**Administración de transacciones en bases de datos múltiples**



# **BASES DE DATOS DISTRIBUIDAS HETEROGÉNEAS**

**Bases de datos distribuidas heterogéneas**

**Vista de datos unificada**

**Procesamiento de consultas**

**Administración de transacciones en bases de datos múltiples**



# BASES DE DATOS DISTRIBUIDAS HETEROGÉNEAS

- La manipulación de bases de datos distribuidas heterogéneas requiere de una capa de software adicional (Sistema de datos existentes)
- Integrar un sistema heterogéneo en un sistema homogéneo se difícil o imposible:
  - Dificultades técnicas: Costo de conversión de aplicaciones
  - Dificultades de la organización: Problemas debido a las políticas de las empresas u organizaciones dueñas de los sistemas de BD existentes.



# **BASES DE DATOS DISTRIBUIDAS HETEROGÉNEAS**

**Bases de datos distribuidas heterogéneas**

**Vista de datos unificada**

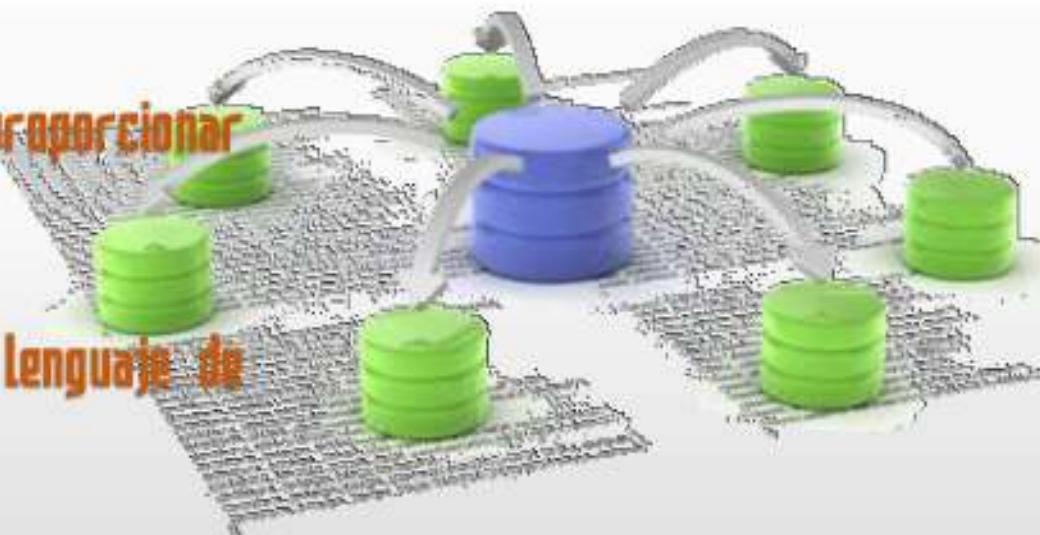
**Procesamiento de consultas**

**Administración de transacciones en bases de datos múltiples**



# Vista de datos unificada

- Manejo de modelos de datos diferentes
- Se debe utilizar un mismo modelo de datos para proporcionar la ilusión de un único sistema de bases de datos
- Una opción es el modelo relación y SQL como lenguaje de consultas común



# Vista de datos unificada

- La integración de esquemas es una tarea complicada debido a la heterogeneidad semántica
- La integración no es simplemente la traducción directa entre lenguajes de definición de datos
  - Los nombres de atributos pueden tener significados diferentes
  - Falta de compatibilidad de los tipos de datos
  - La traducción de los tipos de datos puede no ser simple



# BASES DE DATOS DISTRIBUIDAS HETEROGÉNEAS

**Bases de datos distribuidas heterogéneas**

**Vista de datos unificada**

**Procesamiento de consultas**

**Administración de transacciones en bases de datos múltiples**



# Procesamiento de consultas

- Una consulta en un esquema global debe ser traducida al esquema local, en cada uno de los sitios en donde la consulta deba ser ejecutada
- Los resultados de la consulta deben ser traducidos al esquema global
- Procesamiento de resultados para eliminar duplicados
- La optimización de las consultas se hace de acuerdo a la optimización local



# **BASES DE DATOS DISTRIBUIDAS HETEROGÉNEAS**

**Bases de datos distribuidas heterogéneas**

**Vista de datos unificada**

**Procesamiento de consultas**

**Administración de transacciones en bases de datos múltiples**



# Administración de transacciones en bases de datos múltiples

Un sistema de múltiples bases de datos es compatible con 2 tipos de transacciones

- Transacciones locales**

*Sin control del sistema de múltiples bases de datos*

- Transacciones globales**

*Se ejecuta bajo el control del sistema de múltiples bases de datos*

**Cada sistema local debe utilizar un esquema de control de concurrencia**



## **9. Cloud-Based Databases [Bases de datos basadas en la nube ]**

- Valoran mas la disponibilidad y escalabilidad sobre la consistencia
- Tienen características de sistemas homogéneos y heterogéneos



# Sistemas de almacenamiento de datos en la nube

- Tienen requerimientos altos de escalabilidad
- Los datos deben dividirse en miles de procesadores.



# REPRESENTACION DE DATOS.

Funciones de bases de datos en la nube:

`put(key, value)`: que se utiliza para almacenar valores con una clave asociada.

`get(key)`: que recupera el valor almacenado asociado con la clave especificada



# Arquitectura de un sistema de almacenamiento de datos de nube

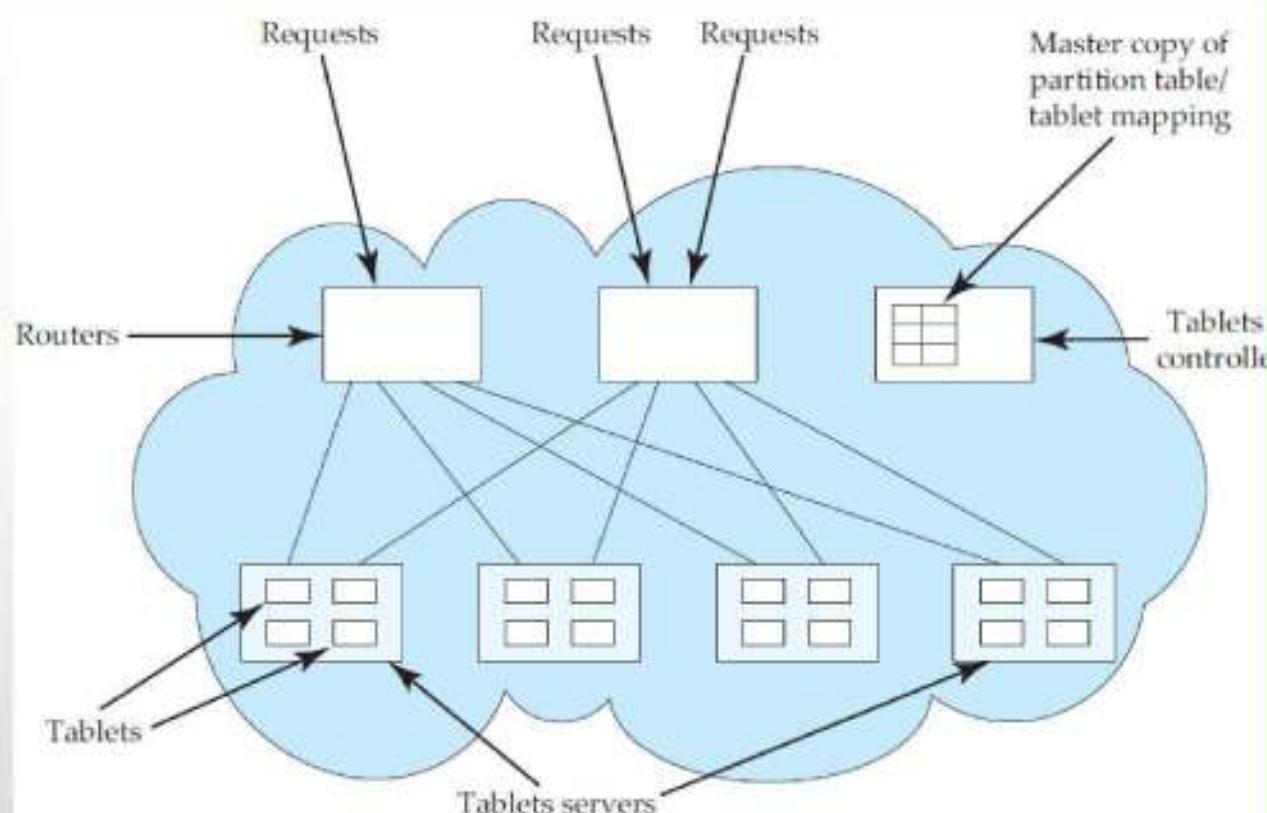


Figure 19.7 Architecture of a cloud data storage system.

Tablets: partición de datos en unidades relativamente pequeñas.

Sitio maestro: tiene asignado un Tablet.

Requests: solicitudes.

Función de mapeo: sirve para saber que sitios son responsables de la tableta.

## **Las transacciones y replicación**

Dado que cada tableta se controla mediante un solo sitio maestro, si el sitio falla la tableta debe ser reasignada a un sitio diferente que tiene una copia de la tableta, que se convierte en el nuevo sitio maestro para la tableta. Las actualizaciones de una tableta se registran, y el registro es a su vez replicado.

# Bases de datos tradicionales en la Nube

- Utilización de Maquinas virtuales.



## Desafíos

- Falta de control de los datos.
- Replicaciones de los datos en distintos lugares.
- Propiedades ACID.

# 11. Estado del Arte



*Un acercamiento de participación vertical dinámica para sistemas de bases de datos.*



*Sistema de búsqueda en bases de datos distribuidas basado en Alchemi.*



*Sistemas de bases de datos a gran escala basadas en redes P2P semánticas.*



*Búsqueda del algoritmo de optimización de consultas en sistemas de bases de datos distribuidas.*

# Un acercamiento de participación vertical dinámica para sistemas de bases de datos.

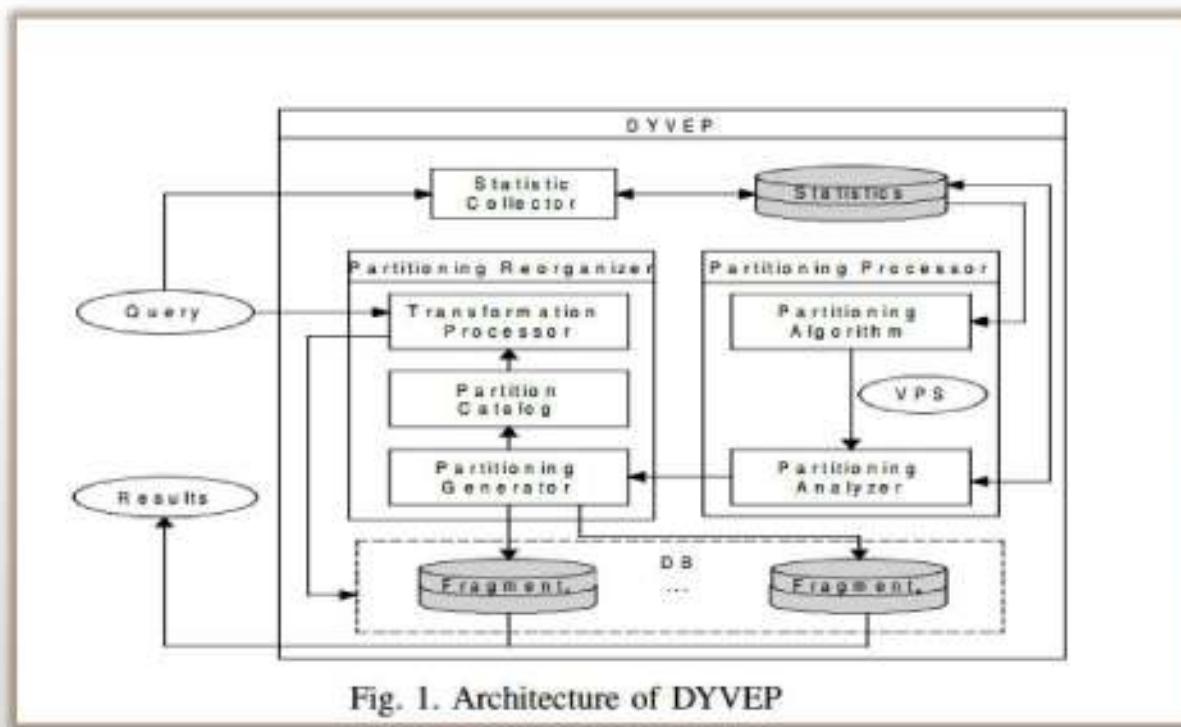
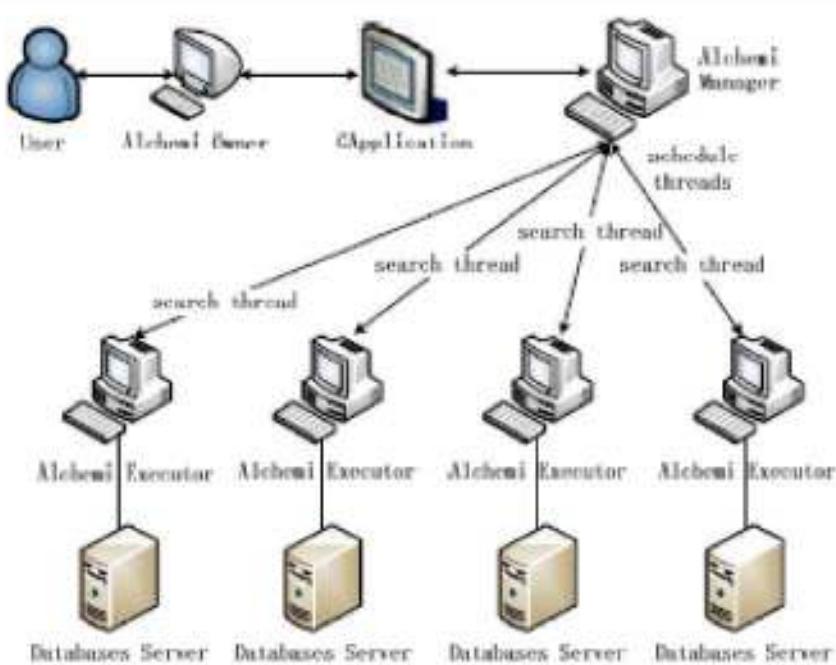


Fig. 1. Architecture of DYVEP

Rodríguez, L., Li, X. (2011). A Dynamic Vertical Partitioning Approach for Distributed Database System. International Conference on Systems, Man and Cybernetics. pp. 1853-1858.

# Sistema de búsqueda en bases de datos distribuidas basado en Alchemi.



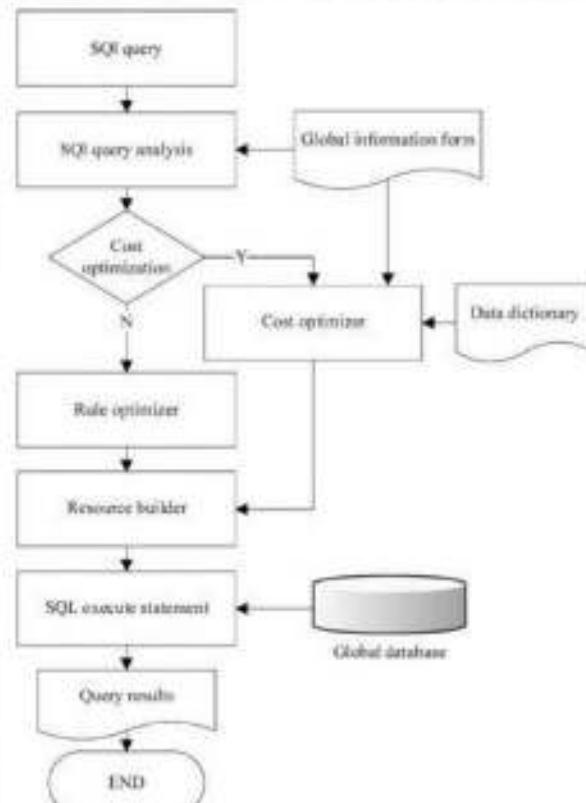
- Hongxia, X., Weifeng, L. (2009). *Distributed Database Searching System based on Alchemi*. International Forum on Computer Science-Technology and Applications. Vol. 1. pp. 160-163.

# **Sistemas de bases de datos a gran escala basadas en redes P2P semánticas.**



*Huang, L. (2010). Large Scale Distributed Database Systems Based on Semantic P2P Networks. First International Conference on Networking and Distributed Computing. pp. 407-409.*

# Búsqueda del algoritmo de optimización de consultas en sistemas de bases de datos distribuidas.



Query scale	Query efficiency of traditional algorithm	Query efficiency of optimized algorithm
1000	0.47	0.48
2000	0.47	0.49
5000	0.45	0.52
8000	0.43	0.56

Yuanyuan, F., Xifeng, M. (2010). Distributed Database System Query Optimization Algorithm Research. International Conference on Computer Science and Information Technology. pp. 657-660.

# **Aplicaciones**

## **[Ambientes más frecuentes]**

- ✓ *Cualquier organización que tiene una estructura descentralizada*
- ✓ *Organismos gubernamentales y de servicio público*
- ✓ *La industria de manufactura*
- ✓ *Control y comando militar*
- ✓ *Líneas de transportación aérea*
- ✓ *Cadenas hoteleras*
- ✓ *Servicios bancarios y financieros*

# Ventajas y Desventajas

## Ventajas

- *Datos cercanos*
- *Procesamiento rápido*
- *Adición de nodos*
- *Menor costo de operación*
- *Amigable al usuario*
- *Falla de un nodo*
- *Autonomía de nodos*



## Desventajas

- *Control y manejo de datos*
- *Habilidad en la integridad*



# Conclusiones

- ❖ La implementación de bases de datos distribuidas son de gran beneficio en cuanto mejoran el rendimiento y la escalabilidad de los datos.
- ❖ El estudio previo a la implementación debe ser detallado y sustentado.
- ❖ Las nuevas tecnologías acompañadas de un amplio campo matemático proveen herramientas que facilitan una correcta implementación.

# Bibliografía

- Silberschatz, A., Forth, HF., Sudarshan, S. (2009). *Database System Concepts*. Editorial McGraw-Hill. 6a ed. Estados Unidos. 1349 pp.
- Rodríguez, L., Li, X. (2011). A Dynamic Vertical Partitioning Approach for Distributed Database System. International Conference on Systems, Man and Cybernetics. pp. 1853-1858.
- Hongxia, X., Weifeng, L. (2009). Distributed Database Searching System based on Alchemi. International Forum on Computer Science-Technology and Applications. Vol. 1. pp. 160-163.
- Huang, L. (2010). Large Scale Distributed Database Systems Based on Semantic P2P Networks. First International Conference on Networking and Distributed Computing. pp. 407-409.
- Yuanyuan, F., Xifeng, M. (2010). Distributed Database System Query Optimization Algorithm Research. International Conference on Computer Science and Information Technology. pp. 657-660.
- Toledo, V., Miralles, I. (Sin fecha). Bases de Datos Distribuidas. 13pp.  
<https://iessanvicente.com/colaboraciones/BBDDdistribuidas.pdf>
- Gutiérrez, A. (2005). *Diseño de Base de Datos Distribuida (Texto Base)*.  
[https://lilhectortorres.files.wordpress.com/2010/09/base\\_de\\_datos\\_distribuidas.pdf](https://lilhectortorres.files.wordpress.com/2010/09/base_de_datos_distribuidas.pdf)
- Ricardo, C. (2003). Fragmentación de Datos en Bases de Datos Distribuidas. *Tecnología en Marcha*. 16:4. pp. 60-67.

# Bibliografía

- *Bases de datos heterogéneas.* Tomado de:  
<http://alfa.facyt.uc.edu.ve/computacion/pensum/cs0347/download/exposiciones2006-2007/BDH.pdf>
- *5. Distributed Databases.* Tomado de: <http://carlosproal.com/bda/bda05.html>
- *Universidad Nacional del Sur. Departamento de Ciencias e Ingeniería de la Computación. Elementos de Bases de Datos.* 2004. Tomado de: <http://cs.uns.edu.ar/~gis/ebd/Archivos/Clases/EBD%20-%20Clase%2023%202004%20BN.pdf>

**GRACIAS**

