Trabajo Práctico

Materia: Laboratorio IV

Tema: Bases de datos distribuidas

Profesora: Marcela Russo

Alumno:

- Nicolás Ezequiel Salvia

Año: 2023

Base de datos distribuidas

Descripción de la arquitectura, las características y sus funcionalidades.

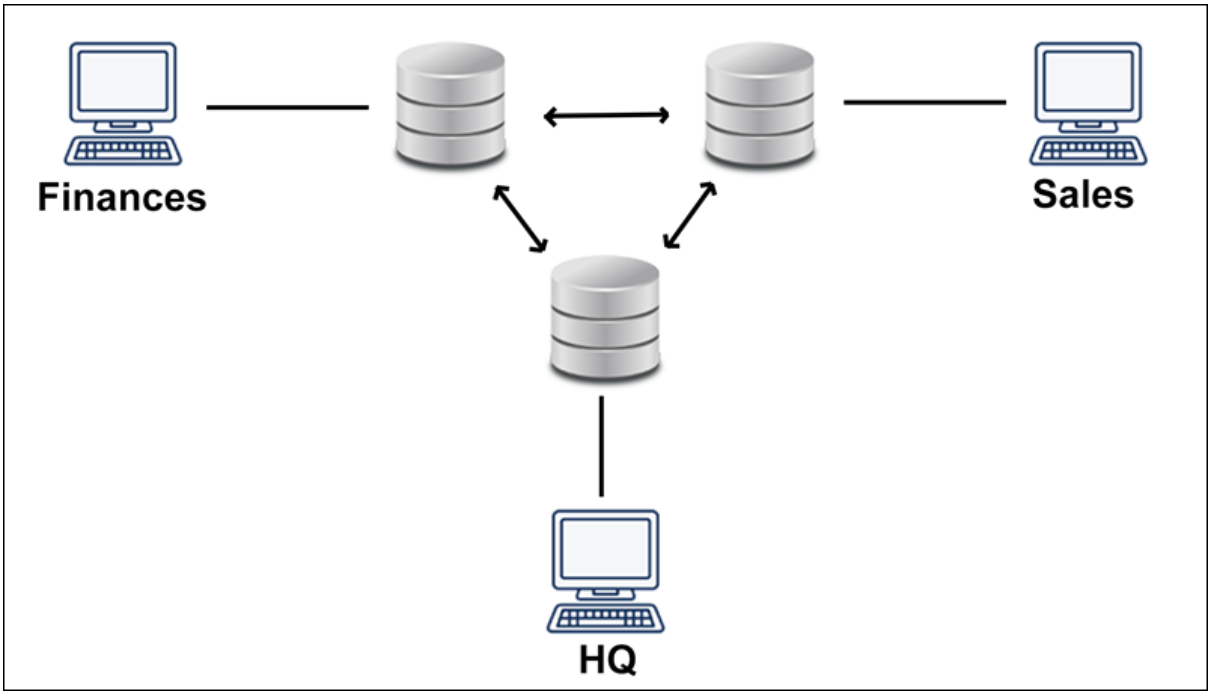
Una base de datos distribuida es una base de datos que corre y almacena los datos en múltiples computadoras. Estas bases de datos operan en dos o más servidores interconectados a una red. A cada una de estas bases de datos se las conoce como instancias o nodos.

Características

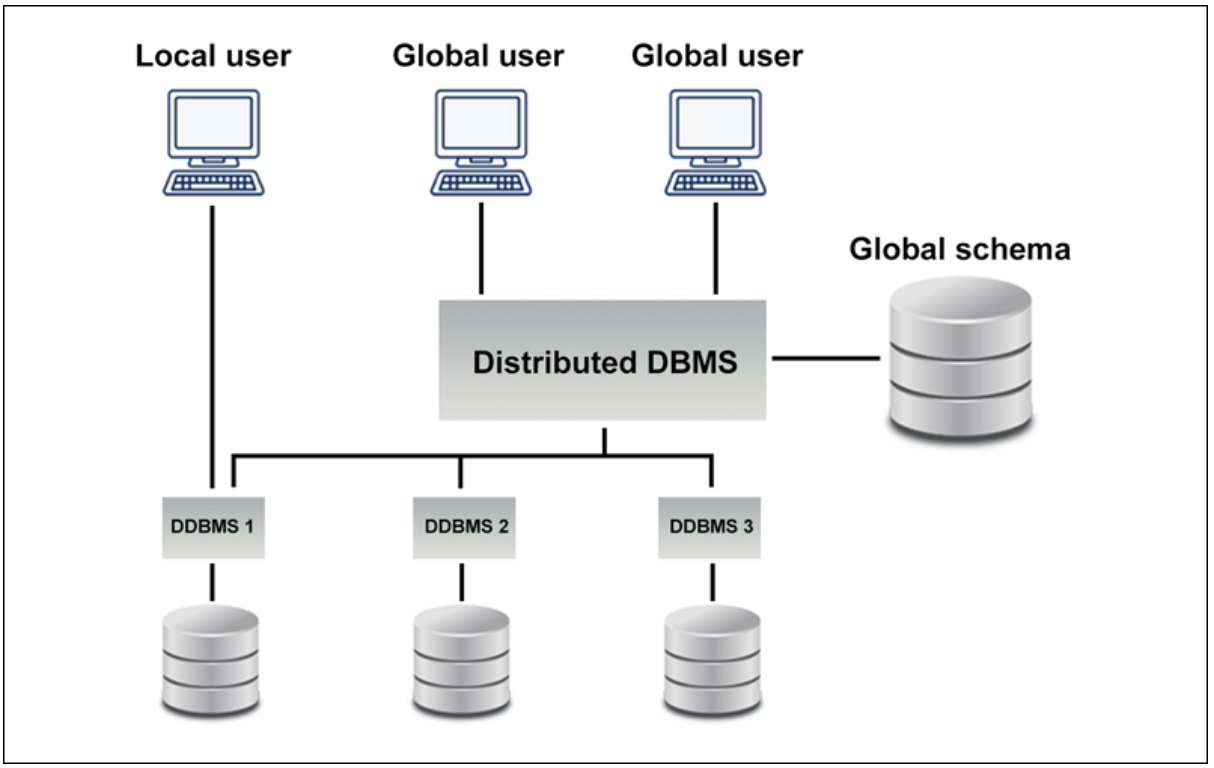
- **Independiente de la ubicación.** Datos físicos almacenados en múltiples sitios y manejados por DDBMS independientes
- **Procesamientos de queries distribuidos.**
- **Manejo de transacciones distribuidos.** Provee consistencia a través de protocolos de commit, técnicas de control de concurrencia distribuidas y métodos de recuperación distribuidos
- Las bases de datos se representan como una única base de datos lógica interconectadas
- Todas las bases de datos en una colección están linkeadas por una red y se comunican entre ellas.
- **Transaction processing.** Programa que incluye una colección de operaciones de una o más base de datos. Consiste en un proceso atómico donde se ejecuta todo o nada.

Tipos

- **Homogénea:** Red de idénticas bases de datos en múltiples sitios. Todas tienen el mismo sistema operativo, DDBMS y estructura de datos haciéndola más fácil de manejar.



- **Heterogénea:** Usan diferentes schemas, sistema operativo, DDBMS y diferentes modelos de datos.



Almacenado en base de datos distribuidos

Replicación

El sistema almacena copias de datos en diferentes sitios. La ventaja es que aumenta la disponibilidad de datos en diferentes sitios y permite realizar queries en paralelo. Sin embargo, esto implica que están constantemente actualizando y sincronizando las unas con otras para mantener una copia exacta de la base de datos. Esto genera un complicado control de concurrencias.

Podemos tener **replicación completa** en donde toda la base de datos se encuentra en todos los sitios o podemos tener **replicación parcial** en donde algunos fragmentos de datos son replicados y otros no.

Tipos de replicación para datos			
	Descripción	**Utilidad**	**Desventaja**
Replicación transaccional	Los usuarios reciben una copia inicial completa y van recibiendo actualizaciones a medida que los datos cambian. Funciona con un sistema publicador-suscriptor para garantizar la consistencia.	Se utiliza en entornos servidor-servidor.	
Replicación Snapshot	Distribuye una copia exacta de la base de datos en un momento dado y no chequea por cambios en los datos.	Generalmente se usan cuando los cambios en los datos no son frecuentes.	Más lento que el anterior dado que tiene que enviar instantáneas de un punto a otro.
Replicación merge	Los datos de 1 o más bases de datos se combinan en una sola base de datos.	Es el tipo de replicación más complejo porque permite al publicador y al suscriptor realizar cambios en la base de manera independiente.	Se usa en entornos clientes-servidor

Fragmentación

Las tablas están fragmentadas. Esto quiere decir que están separados en pequeñas partes y cada fragmento se almacena en sitios diferentes según sea requerido. El requisito es que los fragmentos luego puedan ser reconstruidos en una relación original sin perder datos. La ventaja que tienen es que previene la inconsistencia de datos. Existen dos tipos de fragmentaciones:

Horizontal

Es el proceso de dividir una tabla horizontalmente asignando a cada fila o a un grupo de filas de relaciones, uno o más fragmentos. Estos fragmentos se asignan a distintos lados de una base de datos distribuida.

Ejemplo:
Consideremos una tabla empleado

Eno	**Ename**	**Design**	**Salary**	**Dep**
101	A	abc	3000	1
102	B	abc	4000	1
103	C	abc	5500	2
104	D	abc	5000	2
105	E	abc	2000	2

Podemos dividir la tabla de la siguiente manera:

T1				
Eno	**Ename**	**Design**	**Salary**	**Dep**
101	A	abc	3000	1
102	B	abc	4000	1

T2				
Eno	**Ename**	**Design**	**Salary**	**Dep**
103	C	abc	5500	2
104	D	abc	5000	2
105	E	abc	2000	2

Donde para reconstruirla hacemos T = T1 U T2

Vertical

Es el proceso de descomponer las columnas por columnas. Esta fragmentación es útil cuando algunos sitios no necesitan todas als columnas. Para recomponer la tabla, es necesario que se tenga la primary key. Este proceso debe poder lograrse mediante un JOIN natural por lo que es necesario agregar una tupla-id para lograrlo.

Ejemplo:
Utilizando la misma tabla del ejemplo anterior, podemos dividir la tabla en

Eno	**Ename**	**Design**	**Tuple_id**
101	A	abc	1
102	B	abc	2
103	C	abc	3
104	D	abc	4
105	E	abc	5

Salary	**Dep**	**Tuple_id**
3000	1	1
4000	2	2
5500	3	3
5000	1	4
2000	4	5

Fragmentación Mixta

Se realiza la fragmentación en ambos sentidos. Esto se puede realizar de dos maneras

1. Se realiza una fragmentación horizontal y luego una fragmentación vertical.
2. Se realiza mediante una fragmentación vertical y luego una fragmentación horizontal.

Ejemplo:

Ename	**Design**
A	abc
B	abc
C	abc

Configuraciones de replicación

- **Maestro-Esclavo:** Una base de datos es configurada como maestro y las otras como esclavos. El maestro recibe todas las operaciones de escritura y los esclavos una copia de los datos
- **Replicación Multi-Maestro:** Todos los servidores son maestros, en todos se les escribe y todos los cambios en un maestro serán replicados en los otros maestros.
- **Replicación peer-to-peer:** Todos los servidores actúan como maestro-esclavo y los datos son replicados de manera peer-to-peer
- **Replicación de fuente única:** Una sola fuente de base de datos es replicada en múltiples bases de datos a las que se les apunta.
- **Multiactiva:** Utilizado por CockroachDB. Se diferencia del sistema **activo-activo** ya que este solo aplica la escritura cuando se logra un consenso entre las replicas, es decir, la mayoría de las réplicas pudieron escribir el cambio exitosamente. Si la mayoría de las replicas están offline, la base de datos no estará disponible.

Ventajas y desventajas.

Ventajas y desventajas generales	
Ventajas	**Desventajas**
Incrementa la resiliencia y disminuye los riesgos. Dado que trabajan con réplicas de los mismos datos en múltiples instancias, si uno de los nodos cae, la aplicación puede seguir trabajando.	**Incremento de la complejidad operacional.**
Distribuir la base de datos puede mejorar la performance. Permite distribuir todo el trabajo en varias bases de datos evitando un cuello de botella por tener que realizar todas las operaciones de lectura y escritura en la misma pc.	**Incremento en la curva de aprendizaje**
Distribuir la base de datos geográficamente permite reducir la latencia.	

Ventajas y desventajas de replicación	
Ventajas	**Desventajas**
Mejora del performance ya que los datos pueden leerse de una copia local en vez de una remota	Incrementa la complejidad ya que el proceso de replicación necesita configurarse y mantenerse
Incrementa la disponibilidad de datos ya que permite tenerla en caso de que falle la principal	Incrementa el riesgo de inconsistencias ya que los datos pueden ser actualizados en distintas replicas
Mejora la escalabilidad ya que la carga en la base primaria se reduce mediante la lectura de copias	Incrementa el consumo de almacenamiento y uso de red ya que las copias deben ser almacenadas y transmitidas

Ventajas y desventajas de fragmentación	
Ventajas	**Desventajas**
Aumenta la eficiencia si los datos están cerca del sitio de uso	Velocidad de acceso puede ser muy alta si se necesitan datos de distintos fragmentos
Optimización de queries locales puede ser suficiente para algunas queries ya que los datos están disponibles localmente	Puede ser muy costoso utilizar fragmentación recursiva
Facilita mantener la seguridad y la privacidad ya que los datos se encuentran fragmentados	

Comparación contra los otros tipos de datos, cuándo conviene usarla y cuándo no.

No hay mucha para comparar dado que estamos hablando de un tipo de arquitectura de base de datos más que de un motor de base de datos.

Cuando usar una base de datos distribuidas:

- Cuando queremos tener una buena disponibilidad (probabilidad de correr continuamente durante un intervalo de tiempo) y fiabilidad (probabilidad de correr en determinado tiempo)
- Cuando los tiempos de respuesta son importantes. Con esta base de datos podemos fragmentar los datos y los usuarios pueden realizar solicitudes donde los únicos datos que se necesiten sean los locales.

Cuando no usar una base de datos distribuidas:

- Cuando no se cumple ninguna de las anteriores

Modo de licenciamiento, de ser posible comentar acerca de los precios.

- **Apache Ignite** Apache Software Foundation
- **Apache Cassandra** Apache Software Foundation
- **Apache HBase** Apache Software Foundation
- **Couchbase Server** Couchbase Inc.
- **Amazon SimpleDB** Amazon
- **FoundationDB** Apple
- **CockroachDB** Cockroach Labs

Mostrar la instalación del motor de la BD siempre que sea posible.

https://linuxhint.com/install-use-apache-cassandra-ubuntu-22-04/

https://docs.datastax.com/en/cassandra-oss/3.0/cassandra/initialize/initSingleDS.html

Años de antigüedad de su creación, principales empresas en donde funciona, proveedor que facilita el producto.

Durante los años 60, en plena guerra fría, un investigador llamado Paul Baran desarrolló una idea de comunicación distribuida en donde los mensajes viajaban a través de una red de nodo hasta que finalmente llegaran a su destino. Estos nodos seria computadoras en vez de switches telefonicos. Estos nodos crearian múltiples caminos y fragmentarian el mensaje de tal manera que su transmisión fuera sencilla y eficiente.
Con el objetivo de lograr la mayor eficiencia posible de las instalaciones, se creo un proyecto llamado ARPANET que tenía como objetivo conectarlas todas a una red.
A medida que la red fue creciendo se empezaron a investigar diferentes tipos de redes. En conjunto con Europa, se empezo a crear protocolos y sistemas de direccionamiento entre otras cosas que permitieron la aparición de Internet.

Los primeros programas distribuidos se conocieron como Creeper and Reaper. Creeper utilizaba los ciclos de procesador que no se utilizaban para copiarse al próximo sistema y borrarlo del anterior. Creeper fue modificado para que no se borrará del sistema anterior y así surgió Reaper que se encargaba de borrar todas las copias de Creeper.

En 1988, el DEC(Digital Equipment Corporation) System Research Center comenzó un proyecto donde le enviaba a los voluntarios una aplicación que debian ejecutar el momentos libres y luego reenviar esos resultados. Este proyecto llego a tener cerca de 100 usuarios en 1990.

Más adelante, el DEC y otros grupos comenzaron a utilizar Internet con el objetivo de distribuir material para ser calculado. Eran similares a rompecabezas que consiste en factorizar números, buscar números primos y crackeo de encriptaciones. Estos rompecabezas tenían incentivos monetarios ya que requieren de mucho tiempo para poder resolver estos problemas.

Fuentes:

- https://www.cockroachlabs.com/blog/what-is-a-distributed-database/
- https://phoenixnap.com/kb/distributed-database
- https://www.geeksforgeeks.org/fragmentation-in-distributed-dbms/
- https://www.geeksforgeeks.org/data-replication-in-dbms/
- https://www.techtarget.com/searchoracle/definition/distributed-database