Instituto Tecnológico de Costa Rica

Trabajo de investigación, Bases de Datos Distribuidas

José Morales Vargas, carné 2019024270
Alejandro Soto Chacón, carné 2019008164
Ignacio Vargas Campos, carné 2019053776
José Retana Corrales, carné 2020144743

Área Académica de

Ingeniería en Computadores

Bases de Datos

(CE3101)

Profesor Marco Rivera Meneses

Semestre I 2022

Índice

Introducción

Marco Teórico

Para desarrollar el tema de bases de datos distribuidas es esencial primero plantear que define a un sistema de este tipo. En primer lugar se deben definir algunos conceptos base.

En primer lugar se debe definir el término de base de datos. Una base de datos se define como una colección de datos, con este término refiriéndose a hechos conocidos que pueden ser almacenados y que tienen significado implícito. Una base de datos tiene como propiedades necesarias: representa un aspecto del mundo real (minimundo) y todo cambio al mismo se refleja en ella; es una colección de datos lógicamente coherente con un significado inherente; y está diseñada, construida y populada con datos para un propósito específico con un público meta específico en mente (Elmasri & Navathe, 2016, pp. 4-5).

Se deben separar los conceptos de base de datos y sistema de administración de bases de datos. El sistema de administración de base de datos (DBMS) es un sistema computarizado que permite a los usuarios el crear y mantener una base de datos. Es un sistema de software de propósito general que facilita el proceso de definir, construir, manipular y compartir bases de datos entre varios usuarios y aplicaciones (Elmasri & Navathe, 2016, p. 6).

Finalmente, un sistema distribuido es un ambiente computacional en en que varios componentes están repartidos entre múltiples computadores en una red. Estos dispositivos parten el trabajo, coordinando sus esfuerzos para completar el trabajo de forma más eficiente que la posible si tan solo un dispositivo fuera responsable de la tarea (Splunk, 2022).

Respecto a sistemas distribuidos en específico se deben considerar los fundamentos relacionados al diseño de sistemas que tratan con datos distribuidos: fragmentación y replicación.

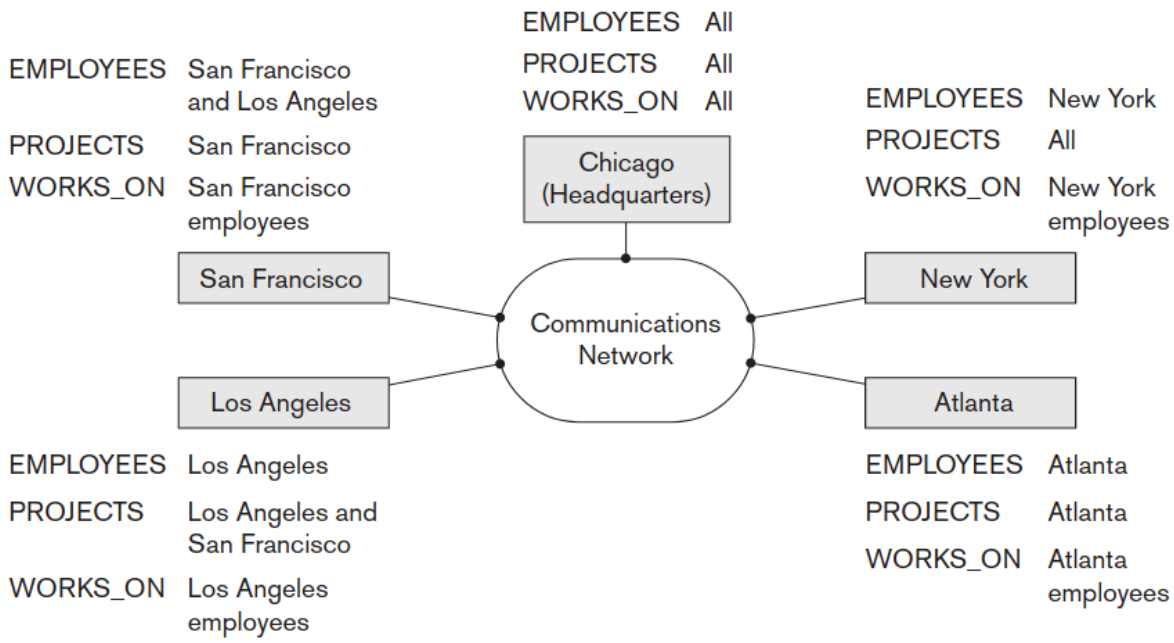

Figura 1: Ejemplo de fragmentación y replicación (Elmasri & Navathe, 2016, p. 844).

La fragmentación se refiere al como se puede partir la información para almacenarla en lugares distintos. Asumiendo que se trabaja con un modelo relacional. Se dice entonces que existen dos tipos de fragmentación: horizontal y vertical. La fragmentación horizontal se refiere a dividir los registros por tuplas, es decir, distintos equipos contienen cada uno una cierta cantidad de tuplas de una relación. En el caso de la fragmentación vertical, el término se refiere a un esquema más complejo en que se divide una relación por atributos. Cada subdivisión debe contener la llave primaria de la relación de forma que se pueda reconstruir cada tupla completa (Tupper, 2011, pp. 387-388).

La replicación se refiere a la creación de datos redundantes que permiten a los procesos que necesitan acceder a la información el proceder de forma suave y efectivamente. Maximiza la disponibilidad pero hay desventajas asociadas, por ejemplo, se puede llegar a un caso en que haya redundancia en cada equipo, de manera que al momento de refrescar los datos se incurre en un gran costo computacional. Antes de aplicar la técnica de replicación se debe considerar cuidadosamente los tiempos de refrescamiento y formas de mantener la integridad referencial en la base de datos (Tupper, 2011, p. 388)

Otro fundamento teórico importante a tomar en cuenta para el estudio de las bases de datos distribuidas es el control de concurrencia, problema que se vuelve más complejo cuando los datos están descentralizados. Harrington (2016, pp. 450-451) menciona que el objetivo de todo sistema de control de concurrencia es la ejecución de transacciones ACID, que se refiere a las propiedades de atomicidad, consistencia, aislamiento (isolation) y durabilidad.

Atomicidad significa que la transacción es una sola unidad, falla o es exitosa como un todo, pero no puede finalizar en un estado intermedio.

Consistencia se refiere a que toda transacción debe resultar en un base de datos con un estado consistente, es decir, que cumple todas las restricciones.

Aislamiento se refiere a la propiedad de las transacciones de ser serializables. El efecto de correr ambas transacciones al mismo tiempo debería ser el mismo que si se ejecutaran en serie.

Finalmente, una transacción debe tener durabilidad, es decir, una vez que se ejecuta, sus resultados son persistentes y no se les puede hacer un rollback.

Sistemas de Bases de Datos Distribuidas

Cuando se habla de una base de datos distribuida (DDB) el término en cuestión se refiere a una colección de múltiples, lógicamente interrelacionadas bases de datos distribuidas en una red de computadores. Usualmente, la discusión respecto a las mismas también involucra a los sistemas de administración de bases de datos distribuidas (DDBMS), homólogos a los DBMS comunes pero especializados para hacer de la complejidad de los datos distribuidos algo transparente al usuario (Özsu, 2003, p. 674). Se llama sistema de base de datos distribuida(DDBS) a la combinación de una DDB y DDBMS.

Elmasri y Navathe (2016, p. 873) señalan que para poder clasificar a una base de datos como un DDB se deben cumplir las siguientes condiciones:

- Conexión de nodos de bases de datos sobre una red de computadores: Se refiere a que deben haber varios equipos, llamados sitios o nodos. Estos sitios deben estar conectados por una red para transmitir datos y comandos entre sitios.
- Interrelación lógica de las bases de datos conectadas: Es necesario que la información en los distintos nodos esté relacionada de forma lógica.
- Posible ausencia de homogeneidad entre nodos conectados: No necesariamente todos los nodos son idénticos en términos de datos, hardware y software.

La última condición es particularmente importante. No necesariamente una base de datos distribuida es de una arquitectura uniforme. Incluso la estructura de los datos puede variar de un sitio que almacena los datos en un modo relacional y otro que almacena datos en una estructura llave-valor.

Algunos ejemplos de bases de datos distribuidas incluyen opciones de bases de datos NoSQL distribuidas como Cassandra, ScyllaDB y mongoDB. también hay ejemplos SQL como Google Spanner, crateDB, CockroachDB, Yugabyte y Amazon Aurora (Scylla, 2022).

En la implementación de sistemas de bases de datos distribuidas pueden haber una variedad de sistemas operativos en uso, desde sistemas de uso general como Windows y Linux, hasta sistemas especializados como IBM i.

El grado de homogeneidad de una base de datos es uno de los primeros factores al clasificar este tipo de sistemas. Se tienen entonces dos variaciones en base a este criterio: bases de datos con un modelo distribuido homogéneo, y bases de datos con un modelo distribuido federado o heterogéneo.

Cuando un modelo de datos distribuido se refiere a datos fragmentados en dispositivos similares en diferentes localidades, es llamado modelo homogéneo (Tupper, 2011). Lo anterior significa que el sistema es consistente en plataformas, protocolos e interfaces de comunicación, lo que también simplifica sustancialmente la implementación de este tipo de sistema. Una de las ventajas que presenta este tipo de sistema es que no hay necesidad de traducción o re-formateo de datos, lo que puede ser una tarea sumamente complicada.

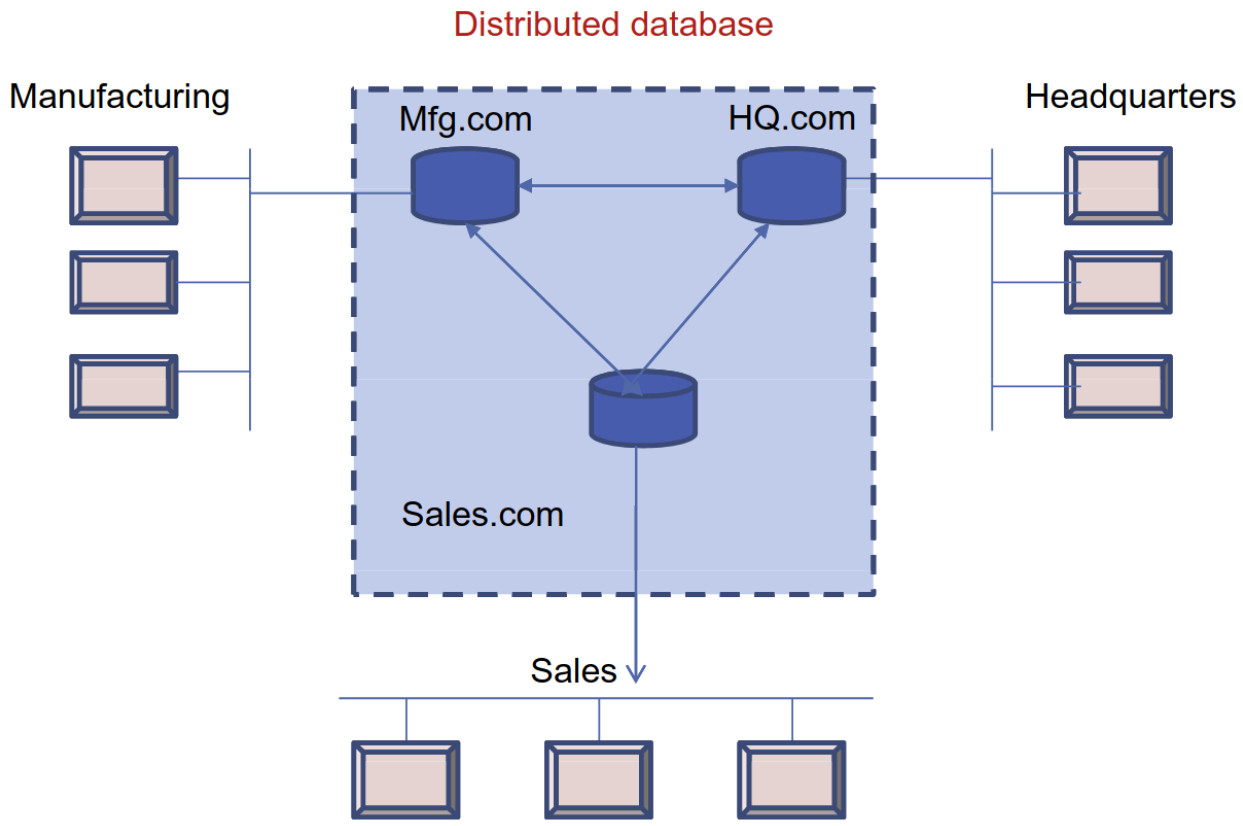

Figura 2: Modelo de una base de datos distribuida homogénea. Obtenido de Tupper, (2011).

El acercamiento contrario a un modelo homogéneo es el modelo heterogéneo o federado, en el que hay una distribución diversa de datos entre sistemas sustancialmente distintos en términos

de software, estructura de datos, entre otras. Encontrar modelos relacionales, jerárquicos o de red no es algo poco común en un modelo heterogéneo.

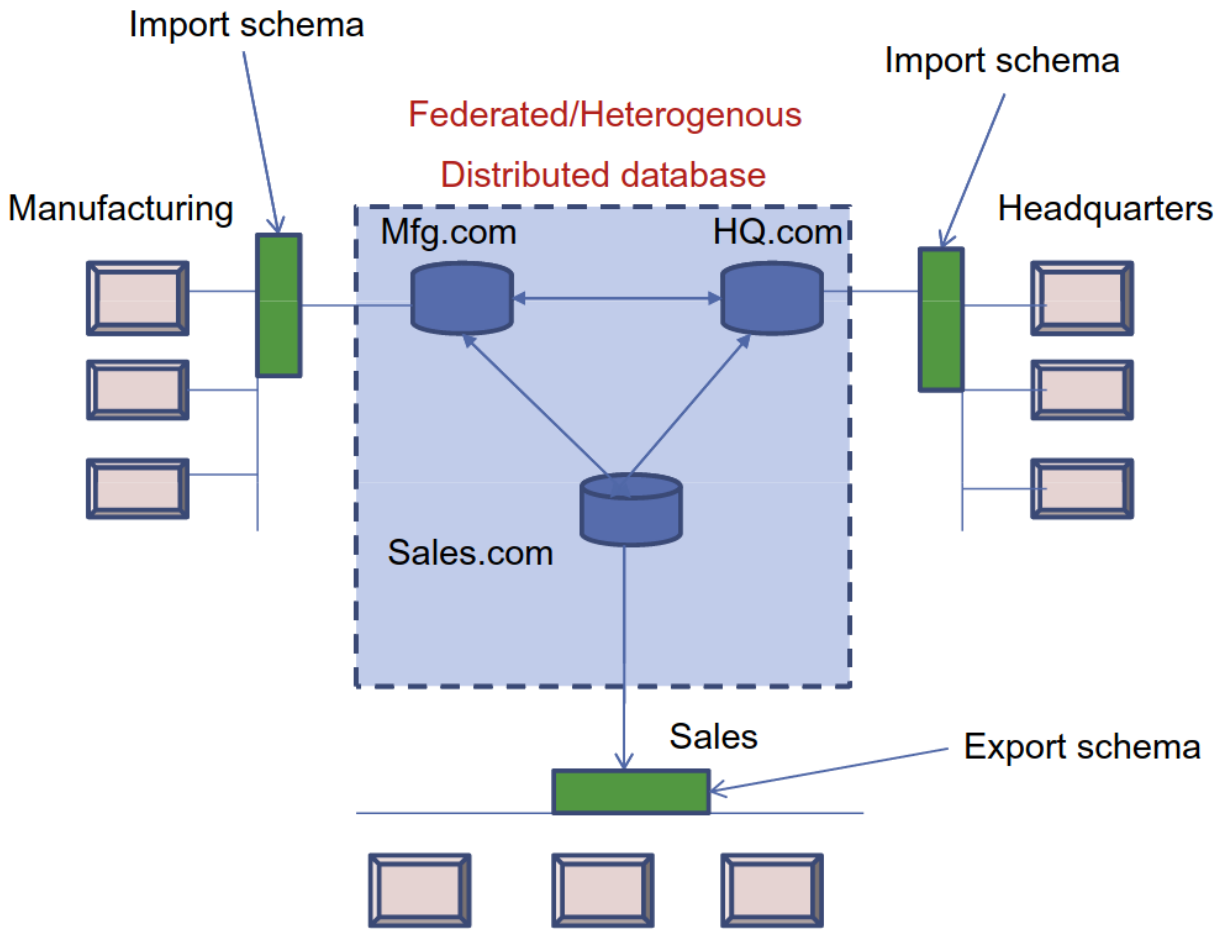

Figura 3: Modelo de una base de datos distribuida federada/heterogénea.Obtenido de Tupper, (2011).

Uno de los retos de los modelos heterogéneos es la heterogeneidad semántica. La misma ocurre cuando hay diferencias en significado, interpretación y uso estipulado de los datos. Este tipo de heterogeneidad impone la mayor dificultad en el diseño de esquemas globales (Elmasri & Navathe, 2016, p. 898).

En un modelo heterogéneo se busca optimizar tanto autonomía de cada componente así como transparencia de la arquitectura general. Lo anterior se debe a que usualmente un modelo heterogéneo surge a partir de querer unificar varios sistemas distintos que han surgido por su propia cuenta, y debido a los requisitos funcionales de cada sistema, mantener su autonomía local

es una prioridad y la forma en la que se trate de unificar a los distintos sistemas debe tratar de adaptarse a este requisito pero sin comprometer su objetivo original de ser una interfaz simple a los datos de todos los sistemas en conjunto.

Uno de los retos más complejos de los sistemas de bases de datos distribuidos es el control de concurrencia. La propiedad de serialización solo se cumple para un grupo de transacciones cuando este es serializable en cada sitio, y el orden de la serialización es el mismo para todos los sitios. A esto último se le llama serializabilidad global (Özsu, 2003). Existen tres formas de asegurar esta propiedad: lock centralizado, lock de copia primaria, y algoritmos de locking distribuido.

Para el caso del lock centralizado, consiste en que una base de datos contiene una sola tabla de locks para todo el sistema. Es la solución más simple pero tiene el problema de que presenta un único punto de falla y provoca un efecto de cuello de botella.

El lock de copia primaria consiste en designar uno de los registros replicados como el primario, de forma que ese es el único que se necesita proteger para realizar una transacción.

El último acercamiento a control de concurrencia son los algoritmos de locking distribuido, los cuales no tienen el problema de causar un problema de cuello de botella, pero lo que se ahorra en poder computacional se paga en complejidad de operación.

Muchos problemas adicionales relacionados al control de concurrencia surgen en el entorno de DDBMS que no se encuentran en DMBS centralizados, entre ellos Elmasri y Navathe (Elmasri & Navathe, 2016, p. 854) mencionan:

- El tener que lidiar con múltiples copias de los ítems de datos: Se debe mantener la consistencia entre copia de los datos que se encuentran en distintos sitios.
- Fallas en sitios individuales: El sistema debería seguir funcionando a pesar de la falla de un sub-nodo.
- Fallas en enlaces de comunicación: El sistema debe ser capaz de lidiar con situaciones en que la comunicación entre los distintos sitios es comprometida.
- Ejecución de transacción distribuida: Se debe poder lidiar con la posibilidad de que un paso de una transacción multi-sitio falle en uno o más sitios.

- Deadlocks distribuidos: Los algoritmos de locking se deben adaptar de forma que se pueda evitar una situación de deadlock distribuido.

Hasta este punto se ha discutido los retos que caracterizan a los sistemas de bases de datos distribuidas, por lo que se debe plantear cuales son las ventajas de estos sistemas.

Según Splunk (2022), Los sistemas de bases de datos distribuidas ofrecen las siguientes ventajas:

- Mayor flexibilidad: Es más fácil agregar mayor poder computacional al sistema según las necesidades.
- Fiabilidad: Un sistema distribuido bien diseñado soportar daños en uno o varios nodos sin ver su desempeño impactado con severidad.
- Rapidez aumentada: Mientras que en sistemas centralizados hay un efecto de cuello de botella entre más tráfico de datos se tenga, en el caso de los sistemas distribuidos escalar la capacidad de trafico es más fácil
- Geo-distribución: La entrega de contenido distribuido es esencial tanto para los usuarios finales como las organizaciones.

Análisis de resultados

Conclusiones

Recomendaciones

Referencias

Elmasri, R., & Navathe, S. (2016). *Fundamentals of Database Systems*. Pearson.

Harrington, J. L. (2016). Concurrency Control. *Relational Database Design and Implementation*, 449-470. https://doi.org/10.1016/b978-0-12-804399-8.00022-3

Özsu, M. T. (2003). Distributed Databases. *Encyclopedia of Information Systems*, 673-682. https://doi.org/10.1016/b0-12-227240-4/00046-0

Scylla. (2022). *What is a Distributed Database? Definition & FAQs*. https://www.scylladb.com/glossary/distributed-database/

Splunk. (2022). *What Are Distributed Systems? An Introduction*. https://www.splunk.com/en_us/data-insider/what-are-distributed-systems.html

Tupper, C. D. (2011). Distributed Databases. *Data Architecture*, 385-400. https://doi.org/10.1016/b978-0-12-385126-0.00022-x