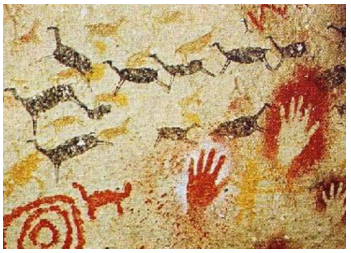
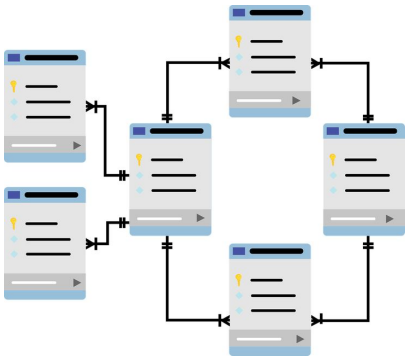
Capítulo 1: Historia y Evolución de las Bases de Datos

Historia



Desde los albores de la humanidad, el ser humano ha tenido la necesidad de almacenar información, ya sea en forma de dibujos rupestres, jeroglíficos, códigos escritos o en la era moderna, datos digitales. El manejo de grandes cantidades de información ha sido un reto constante para la humanidad, que ha desarrollado diversas tecnologías y herramientas para hacer frente a este desafío.

La historia de las bases de datos modernas se remonta a la década de 1960, cuando IBM desarrolló el sistema de procesamiento de datos IMS (Information Management System) para el seguimiento de inventarios. En ese entonces, las bases de datos se basaban en el modelo jerárquico, en el cual los datos se organizan en una estructura en forma de árbol, con un único registro raíz y varios registros secundarios. Este modelo resultaba muy rígido y limitado en la capacidad de almacenamiento y acceso a los datos.



En 1970, Edgar F. Codd, un matemático británico que trabajaba en IBM, propuso un modelo alternativo de bases de datos, basado en la teoría matemática de las relaciones. Este modelo, conocido como modelo de bases de datos relacionales, propone que los datos se almacenen en tablas, que a su vez se relacionan entre sí mediante claves primarias y foráneas. El modelo relacional permitió una mayor flexibilidad en la organización de los datos y un acceso más eficiente a los mismos.

El primer sistema comercial de bases de datos relacionales fue el SQL/DS de IBM, lanzado en 1981. En los años siguientes surgieron otros sistemas de gestión de bases de datos relacionales, como Oracle, Informix y Sybase, entre otros. Estos sistemas de bases de datos comenzaron a ser utilizados por las empresas para la gestión de sus datos empresariales y el almacenamiento de información transaccional.

A finales de los años 80 y principios de los 90, surgieron nuevas tecnologías y herramientas para el manejo de datos, como los sistemas de gestión de bases de datos objeto-relacionales y los sistemas de bases de datos distribuidos. Los sistemas objeto-relacionales permiten el almacenamiento de datos complejos y relacionados de forma más eficiente, mientras que los sistemas distribuidos permiten la gestión de grandes cantidades de datos en varios servidores.
En la década de 2000, surgieron nuevas tecnologías de bases de datos, como las bases de datos NoSQL y las bases de datos en la nube. Las bases de datos NoSQL, como MongoDB y Cassandra, permiten el almacenamiento de grandes cantidades de datos no estructurados, como documentos y datos multimedia. Las bases de datos en la nube, como Amazon Web Services y Microsoft Azure, permiten a las empresas almacenar y procesar grandes cantidades de datos de forma más económica y eficiente.

Arquitectura cliente servidor vs arquitectura basada en la nube

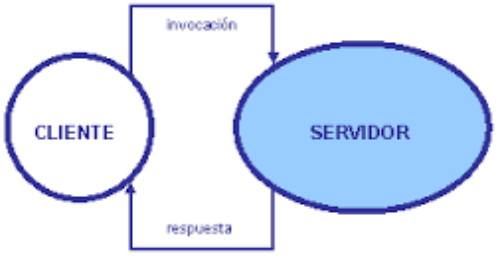
La arquitectura cliente-servidor y la arquitectura en la nube son dos enfoques diferentes para gestionar y utilizar bases de datos y aplicaciones, aunque nosotros nos centraremos solo en bases de datos. Ambas arquitecturas tienen sus ventajas y

desventajas en cuanto a la gestión de bases de datos, y la elección entre ellas dependerá de las necesidades y objetivos específicos de cada organización.

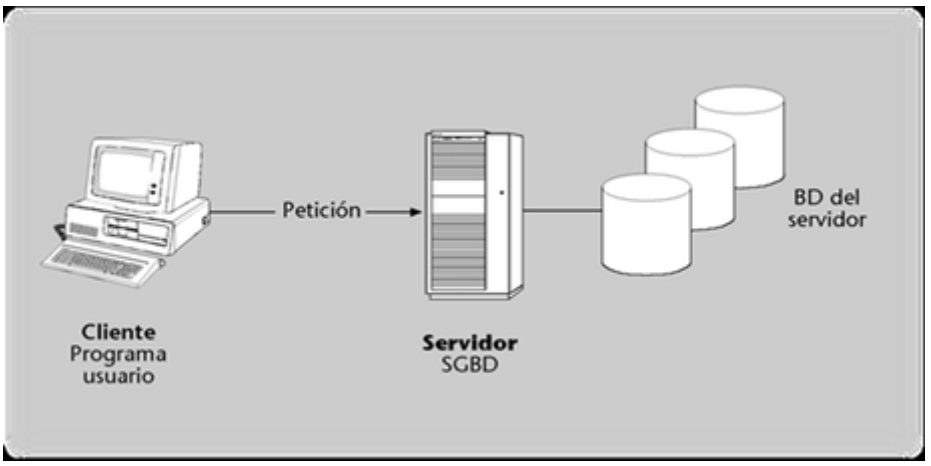
Arquitectura Cliente-Servidor

La arquitectura cliente-servidor se basa en la idea de que hay dos roles en un sistema informático: el cliente y el servidor. El cliente es la aplicación o el software que utiliza el usuario para acceder y manipular los datos, mientras que el servidor es la aplicación o el software que se ejecuta en un servidor remoto y que gestiona los datos.

En la arquitectura cliente-servidor, los clientes envían solicitudes al servidor para acceder a los datos y el servidor responde con la información solicitada. Los clientes pueden ser programas de software instalados en los ordenadores de los usuarios o aplicaciones web que se ejecutan en un navegador web.



En la arquitectura cliente-servidor, los SGBD (Sistemas de gestión de bases de datos) se ejecutan en el servidor y los clientes se conectan al servidor para acceder a los datos almacenados en la base de datos. El servidor de la base de datos es responsable de recibir solicitudes de los clientes, procesar las solicitudes y enviar los resultados de vuelta al cliente.



Ventajas de la arquitectura cliente-servidor para las bases de datos:

- Mayor control y seguridad de los datos ya que los datos están centralizados en un servidor remoto
- En algunos casos, menor tiempo de respuesta. Ya que el servidor puede estar alojado de manera local, evitando la latencia.

Desventajas de la arquitectura cliente-servidor:

- Mayor costo de hardware y software ya que se necesita un servidor dedicado para alojar la base de datos
- Mayor complejidad en la gestión y configuración del servidor y los clientes
- Si no se hace una inversión y planificación de gran envergadura, menor disponibilidad.

Arquitectura basada en la Nube

Antes de adentrarnos a la arquitectura basada en la nube, les voy a contar una pequeña historia de cómo fue evolucionando la arquitectura cliente servidor en términos de alta disponibilidad, para llegar a lo que hoy es la arquitectura basada en la nube.

En un comienzo, si yo quería tener un sitio web por ejemplo, destinaba una PC, “común y corriente” a ser un servidor. Le instalaba los programas necesarios para ser un servidor web, y desde cualquier navegador, de cualquier parte del mundo me podía conectar a ella para ver el sitio web.

Eso es la arquitectura cliente servidor. Por un lado un servidor ofreciendo un servicio (en este ejemplo web), y por otro lado un navegador que funciona como cliente para conectarse a ese servidor.

El gran inconveniente de ello, es que si por ejemplo se corta la energía eléctrica, “se cae el servidor” y nadie puede acceder al sitio web, entonces se comenzó a mejorar las condiciones donde se encontraba instalado ese servidor para que ello no suceda. Uno de los primeros cambios fue incorporar una UPS (un banco de baterías), para que ante un corte de energía siga alimentado mediante baterías. Pero esto seguía siendo deficiente ya que si se quemaba por ejemplo la fuente de alimentación, sucedía lo mismo, es por ello que los servidores fueron evolucionando y comenzaron a tener por ejemplo 2 ó mas fuentes de alimentación.

Con esa misma lógica fueron incorporando componentes redundantes y desarrollando sistemas operativos preparados para servidores (por ejemplo Windows Server, Ubuntu Server, entre otros), los cuales son tolerantes a fallas de hardware.

Entonces por ejemplo si se rompe una fuente de alimentación, el servidor sigue operando con otra, lo mismo si se estropea un procesador o una memoria ram, el mismo lanza una alerta pero el servidor sigue funcionando.

En mi trabajo del día a día manipulo servidores físicos y a menudo se rompen, sobre todo discos rígidos, y se encuentra todo tan bien diseñado que no es un problema de indisponibilidad del servidor. Si bien no es un tema de la asignatura, a modo de colación por si quieren ahondar, en lo que es discos rígidos se utilizan sistemas RAID. En mi caso uso mucho RAID 5, que es una configuración de disco que permite por ejemplo que si tenes 3 discos de 1 TB (3 TB en total), puedas aprovechas solo 2 TB de espacio y el espacio restante se utilice para tolerancia a fallos, entonces en caso de que se rompa cualquiera de los 3 discos, simplemente se reemplaza el dañado por uno nuevo y es todo transparente para el administrador de servidores..

Con esto se logró la tolerancia a fallos del hardware de un servidor y se comenzó a trabajar con alta disponibilidad.

Con el correr del tiempo los sistemas fueron evolucionando, aparecieron grandes sistemas que no podían permitirse estar fuera de servicio en ningún momento del año y se dieron cuenta que por más que haya muchas redundancias, incluso hasta en las conexiones a internet, ante un desastre natural no tenían alta disponibilidad.

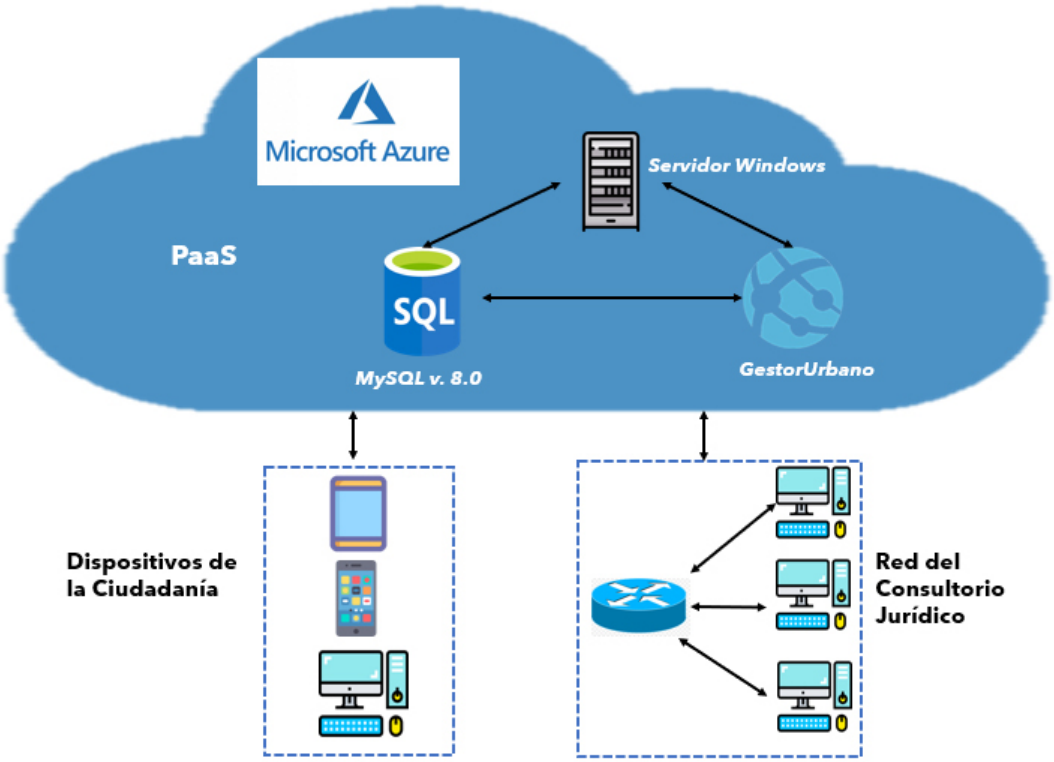
Es allí donde nace la idea de crear cluster de servidores, básicamente la idea consiste en que en lugar de tener un servidor, tengas por ejemplo 5 servidores y que puedas instalar un hypervisor (controlador de máquinas virtuales) sobre los 5 servidores, de modo tal que se puedan sumar los recursos de ellos para responder a los servicios que soporten y que ante la salida de servicio de uno de ellos el resto siguen funcionando. Lo que se logra con eso es que yo pueda poner por ejemplo distribuir los servidores en ubicaciones geográficas diferentes, obteniendo como resultado que por más que un tsunami, huracan, o cualquier tipo de accidente o desastre natural me deje fuera de servicio uno o algunos de los servidores puedo seguir brindando el servicio desde los otros.

Eso último es la base de la computación en la nube, y por eso es que vemos que los servicios que ofrecen grandes empresas como Google, Microsoft, Meta, etc, nunca (o casi nunca) se caen, porque tienen servidores y datacenters distribuidos por todo el planeta que componen un cluster que soporta los servicios ofrecidos.

Ahora sí, vamos de lleno con la arquitectura basada en la nube.

La arquitectura basada en la nube es una **alternativa a la arquitectura cliente-servidor** en la que los recursos de hardware y software se proporcionan como servicios a través de Internet. **En la arquitectura en la nube, los SGBD y los datos se alojan en servidores remotos que están disponibles para los clientes a través de Internet.**

En la arquitectura en la nube, los clientes se conectan a los servidores remotos para acceder a la base de datos y realizar operaciones en ella. Los clientes pueden ser programas de software instalados en los ordenadores de los usuarios o aplicaciones web que se ejecutan en un navegador web.



Ventajas de la arquitectura en la nube para las bases de datos:

- Mayor flexibilidad y escalabilidad ya que los recursos pueden ser asignados y desasignados según las necesidades de la organización de forma muy dinámica.

- Menor costo de hardware y software ya que no se necesita un servidor dedicado para alojar la base de datos. Se paga por lo que se usa.
- Mayor disponibilidad y acceso a los datos desde cualquier lugar con conexión a Internet.

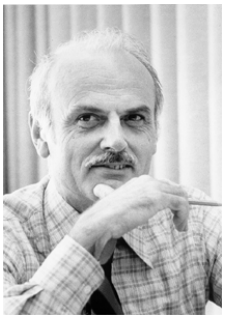
Desventajas de la arquitectura en la nube:

- Menor control y seguridad de los datos ya que los datos se almacenan en servidores remotos controlados por terceros.
- Mayor dependencia de la conectividad a Internet para acceder a la base de datos y realizar operaciones en ella.

La elección entre la arquitectura cliente-servidor y la arquitectura en la nube para la gestión de bases de datos dependerá de las necesidades y objetivos específicos de cada organización. Ambas arquitecturas tienen sus ventajas y desventajas en cuanto a la gestión, el control, la seguridad, la escalabilidad y el costo de la base de datos.

La arquitectura cliente-servidor es más adecuada para organizaciones que requieren un mayor control y seguridad de sus datos, para aquellas que manejan grandes conjuntos de datos que requieren alto rendimiento en forma local, y para aquellas que no se pueden dar el lujo de perder acceso a sus servidores por falta de conectividad a internet. Por otro lado, la arquitectura en la nube es más adecuada para organizaciones que necesitan flexibilidad, escalabilidad y accesibilidad a los datos desde cualquier lugar con conexión a Internet. Es una alternativa que reduce los costos y permite escalar progresivamente.

Referentes



Entre los grandes referentes en la historia de las bases de datos se encuentran Edgar F. Codd (quien se observa en la imagen), un ingeniero de la firma IBM, fue quien propuso el modelo de bases de datos relacionales; Michael Stonebraker, creador de varios sistemas de bases de datos relacionales y objeto-relacionales; y James Gray, quien contribuyó en el desarrollo de sistemas de

bases de datos distribuidos. Además, empresas como IBM, Oracle, Microsoft y Amazon han sido líderes en el desarrollo de tecnologías y herramientas de bases