Introducción a las bases de datos

Bases de datos I – UTN FRRO

Giuliano Crenna

Historia de las Bases de Datos

La evolución de las bases de datos es un reflejo del desarrollo tecnológico y las crecientes necesidades de manejo y almacenamiento de información de la humanidad. Desde los primeros sistemas de almacenamiento hasta las sofisticadas bases de datos modernas, esta evolución ha sido impulsada por la búsqueda constante de eficiencia, velocidad y capacidad.

Primeros Sistemas de Almacenamiento de Datos

En la década de 1950, las tarjetas perforadas y las cintas magnéticas eran las principales formas de almacenamiento de datos. Estas primeras formas de almacenamiento permitieron a las organizaciones manejar grandes volúmenes de datos, pero tenían limitaciones significativas en cuanto a la accesibilidad y la velocidad de recuperación de información. Las computadoras mainframe de IBM y otros fabricantes comenzaron a utilizarse en esta época, gestionando datos en un formato que aún estaba lejos de lo que conocemos como bases de datos.

Surgimiento de los Sistemas de Gestión de Bases de Datos

La década de 1960 marcó el comienzo del desarrollo de sistemas de gestión de bases de datos más estructurados. IBM introdujo su Sistema de Gestión de Información (IMS) en 1966, que utilizaba una estructura jerárquica para organizar los datos. Este modelo jerárquico permitía una relación padre-hijo entre los registros, facilitando el acceso y la gestión de la información de manera más organizada. Al mismo tiempo, el Comité de Lenguajes de Sistemas de Datos (CODASYL) desarrolló el modelo de redes, que ofrecía una mayor flexibilidad al permitir relaciones más complejas entre los datos, superando algunas de las limitaciones del modelo jerárquico.

Revolución de las Bases de Datos Relacionales

El verdadero cambio revolucionario en las bases de datos llegó en la década de 1970 con la introducción del modelo relacional por Edgar F. Codd, un científico de IBM. En 1970, Codd publicó su trabajo seminal sobre el modelo relacional, que proponía organizar los

datos en tablas (o relaciones) que podían ser manipuladas utilizando un lenguaje de consulta estructurado. Este concepto simplificó significativamente la forma en que se podían gestionar y recuperar los datos.

A mediados de los años 70, el modelo relacional ganó popularidad rápidamente y se convirtió en el estándar de facto para las bases de datos. IBM desarrolló System R, uno de los primeros sistemas de bases de datos relacionales, que más tarde inspiró el desarrollo de SQL (Structured Query Language). Oracle, fundada en 1977, fue la primera empresa en comercializar un SGBD relacional basado en SQL, sentando las bases para la industria moderna de bases de datos.

Crecimiento y Diversificación

Durante las décadas de 1980 y 1990, el uso de bases de datos relacionales se expandió rápidamente en todas las industrias. Las mejoras en el hardware y las técnicas de optimización permitieron a las bases de datos manejar volúmenes de datos cada vez mayores con mayor eficiencia. Durante este período, surgieron otros SGBD relacionales importantes como Microsoft SQL Server y MySQL, cada uno aportando sus propias innovaciones y mejoras.

La necesidad de manejar datos no estructurados y la evolución de Internet llevaron al desarrollo de nuevas tecnologías de bases de datos en la década de 2000. Surgieron las bases de datos NoSQL, diseñadas para manejar grandes volúmenes de datos no estructurados y distribuidos. MongoDB, CouchDB, y Cassandra son ejemplos de bases de datos NoSQL que ofrecen flexibilidad de esquema y escalabilidad horizontal, permitiendo a las empresas manejar datos a una escala sin precedentes.

Bases de Datos Modernas

Hoy en día, las bases de datos han evolucionado para cubrir una amplia gama de necesidades y aplicaciones. Además de las bases de datos relacionales y NoSQL, las bases de datos orientadas a objetos integran características de la programación orientada a objetos, permitiendo una integración más estrecha entre la base de datos y el código de la aplicación. Las bases de datos distribuidas, como Google Spanner y Amazon DynamoDB, permiten a las organizaciones almacenar datos en múltiples ubicaciones físicas mientras los gestionan como una sola entidad lógica.

Las bases de datos en la nube, proporcionadas por servicios como Amazon Web Services (AWS), Google Cloud Platform (GCP) y Microsoft Azure, han facilitado aún más la gestión y escalabilidad de las bases de datos, eliminando la necesidad de infraestructura física y permitiendo a las empresas centrarse en el desarrollo y la innovación.

Definición de bases de datos:

Una **base de datos** es un sistema organizado para recopilar, almacenar, gestionar y recuperar datos de manera eficiente. Los datos en una base de datos están estructurados de tal manera que se pueda acceder a ellos y manipularlos fácilmente.

En informática se conoce como dato a cualquier elemento informativo que tenga relevancia para un usuario. Desde su nacimiento, la informática se ha encargado de proporcionar herramientas que faciliten la manipulación de los datos. Antes de la aparición de las aplicaciones informáticas, las empresas tenían como únicas herramientas de gestión de datos los ficheros con cajones, carpetas y fichas de cartón. En este proceso manual, el tiempo requerido para manipular estos datos era enorme. Pero la propia informática ha adaptado sus herramientas para que los elementos que el usuario utiliza en cuanto a manejo de datos se parezcan a los manuales. Por eso se sigue hablado de ficheros, formularios, carpetas, directorios.

Componentes de una Base de Datos

1. **Tablas**:
    - Estructuras fundamentales en una base de datos relacional.
    - Compuestas por filas (también llamadas registros) y columnas (atributos o campos).
    - Cada tabla representa una entidad específica, como "Usuarios" o "Productos".
2. **Filas**:
    - Cada fila en una tabla representa una única instancia de la entidad.
    - Ejemplo: En una tabla "Usuarios", una fila puede representar un solo usuario.
3. **Columnas**:
    - Cada columna en una tabla representa un atributo de la entidad.
    - Ejemplo: En la tabla "Usuarios", columnas pueden incluir "Nombre", "Edad" y "Correo Electrónico".
4. **Claves Primarias**:
    - Un campo o combinación de campos que identifica de manera única cada fila en una tabla.
    - Ejemplo: En una tabla de "Usuarios", una "ID de Usuario" puede ser la clave primaria.
5. **Claves Foráneas**:
    - Un campo en una tabla que es la clave primaria en otra tabla.
    - Se utilizan para establecer relaciones entre tablas.
    - Ejemplo: En una tabla "Pedidos", una "ID de Usuario" puede ser una clave foránea que relaciona cada pedido con un usuario específico en la tabla "Usuarios".

Tipos de Bases de Datos

1. **Bases de Datos Relacionales (RDBMS)**:
    - Utilizan un modelo basado en tablas para representar datos y sus relaciones.
    - Ejemplos: MySQL, PostgreSQL, Oracle, SQL Server.
    - Lenguaje utilizado: SQL (Structured Query Language).
2. **Bases de Datos NoSQL**:
    - Diseñadas para manejar grandes volúmenes de datos no estructurados y distribuidos.
    - Subtipos:
        - **Documentales**: Almacenan datos en documentos tipo JSON. Ejemplo: MongoDB.
        - **Clave-Valor**: Almacenan datos como pares clave-valor. Ejemplo: Redis.
        - **Columnares**: Almacenan datos en columnas en lugar de filas. Ejemplo: Cassandra.
        - **Grafos**: Almacenan datos en nodos y relaciones. Ejemplo: Neo4j.
3. **Bases de Datos Orientadas a Objetos (OODBMS)**:
    - Integran características de la programación orientada a objetos.
    - Ejemplos: db4o, ObjectDB.
4. **Bases de Datos Distribuidas**:
    - Los datos se almacenan en múltiples ubicaciones físicas pero se gestionan como una sola base de datos lógica.
    - Ejemplos: Google Spanner, Amazon DynamoDB.

Operaciones en una Base de Datos

1. **Definición de Datos**:
    - Crear y modificar estructuras de datos.
    - Ejemplo: CREATE TABLE, ALTER TABLE, DROP TABLE.
2. **Manipulación de Datos**:
    - Insertar, actualizar, eliminar y consultar datos.
    - Ejemplo: INSERT INTO, UPDATE, DELETE, SELECT.
3. **Control de Acceso**:
    - Gestión de permisos y restricciones de acceso para usuarios.
    - Ejemplo: GRANT, REVOKE.

4. **Integridad de los Datos**:
    - Asegura la consistencia y precisión de los datos mediante restricciones.
    - Ejemplo: Claves primarias y foráneas, restricciones de unicidad.
5. **Gestión de Transacciones**:
    - Garantiza que las operaciones sean ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad).
    - Ejemplo: BEGIN, COMMIT, ROLLBACK.
6. **Recuperación de Datos**:
    - Backup y recuperación ante fallos.
    - Ejemplo: BACKUP DATABASE, RESTORE DATABASE.

Importancia de las Bases de Datos

- **Eficiencia**: Permiten gestionar grandes volúmenes de datos de manera eficiente.
- **Acceso Rápido**: Facilitan el acceso rápido a la información almacenada.
- **Seguridad**: Ofrecen mecanismos robustos de seguridad para proteger los datos.
- **Integridad**: Mantienen la integridad y consistencia de los datos.
- **Escalabilidad**: Pueden escalar para manejar mayores volúmenes de datos y usuarios.

Tipos de Bases de Datos

1. Bases de Datos Relacionales (RDBMS)

Definición

Las bases de datos relacionales utilizan un modelo basado en tablas para representar datos y sus relaciones. Cada tabla representa una entidad y sus atributos se representan como columnas.

Características

- **Estructura**: Organizadas en tablas con filas y columnas.
- **Relaciones**: Soportan relaciones entre tablas a través de claves primarias y foráneas.
- **Integridad de Datos**: Utilizan restricciones como claves primarias, claves foráneas y restricciones de unicidad.
- **Lenguaje de Consulta**: Utilizan SQL (Structured Query Language) para la gestión de datos.

Ejemplos

- **MySQL**: Sistema de gestión de bases de datos relacional de código abierto conocido por su velocidad y facilidad de uso.
- **PostgreSQL**: Sistema avanzado y de código abierto, con soporte para tipos de datos personalizados y transacciones complejas.
- **Oracle**: Sistema robusto y comercial utilizado en grandes empresas, conocido por su escalabilidad y rendimiento.
- **SQL Server**: Sistema de Microsoft con una fuerte integración con otros productos de Microsoft.

Ventajas

- **Estructura Bien Definida**: Claramente organizada, lo que facilita la consulta y manipulación de datos.
- **Integridad de Datos**: Mecanismos robustos para mantener la integridad y consistencia de los datos.
- **SQL**: Lenguaje estándar ampliamente conocido y utilizado.

Desventajas

- **Escalabilidad Horizontal Limitada**: Puede ser difícil escalar horizontalmente en comparación con bases de datos NoSQL.
- **Rendimiento**: Puede disminuir con grandes volúmenes de datos y relaciones complejas.

---

2. Bases de Datos NoSQL

Definición

Las bases de datos NoSQL están diseñadas para manejar grandes volúmenes de datos no estructurados y distribuidos, y no utilizan el modelo relacional tradicional.

Características

- **Flexibilidad de Esquema**: No requieren esquemas predefinidos, lo que permite almacenar datos sin una estructura fija.
- **Escalabilidad Horizontal**: Diseñadas para escalar fácilmente agregando más servidores.
- **Diversidad de Modelos de Datos**: Incluyen diferentes tipos de bases de datos para diferentes necesidades.

Subtipos

a. Documentales

- **Definición**: Almacenan datos en documentos tipo JSON o BSON.
- **Ejemplo**: MongoDB.

- **Ventajas**: Flexible, permite almacenar datos anidados y variados.
- **Uso**: Aplicaciones con datos semiestructurados.

b. Clave-Valor

- **Definición**: Almacenan datos como pares clave-valor.
- **Ejemplo**: Redis.
- **Ventajas**: Rápido acceso a datos, ideal para caché.
- **Uso**: Sesiones de usuario, perfiles de usuario, caché.

c. Columnares

- **Definición**: Almacenan datos en columnas en lugar de filas.
- **Ejemplo**: Cassandra.
- **Ventajas**: Optimizado para consultas de agregación y grandes volúmenes de datos.
- **Uso**: Análisis de datos, big data.

d. Grafos

- **Definición**: Almacenan datos en nodos y relaciones (aristas).
- **Ejemplo**: Neo4j.
- **Ventajas**: Excelente para datos interconectados y relaciones complejas.
- **Uso**: Redes sociales, recomendaciones, sistemas de gestión de identidad y acceso.

---

3. Bases de Datos Orientadas a Objetos (OODBMS)

Definición

Las bases de datos orientadas a objetos integran características de la programación orientada a objetos con la gestión de bases de datos. Los datos se almacenan como objetos, similar a los objetos en los lenguajes de programación orientados a objetos.

Características

- **Persistencia de Objetos**: Los objetos se almacenan de manera persistente en la base de datos.
- **Herencia y Polimorfismo**: Soportan características de orientación a objetos como herencia y polimorfismo.
- **Integración con Lenguajes OOP**: Diseñadas para trabajar directamente con lenguajes de programación orientados a objetos.

Ejemplos

- **db4o**: Base de datos orientada a objetos de código abierto.

- **ObjectDB**: Base de datos orientada a objetos para Java.

Ventajas

- **Compatibilidad con OOP**: Facilita el desarrollo de aplicaciones en lenguajes orientados a objetos.
- **Reducción de Impedancia**: Elimina la necesidad de conversión entre estructuras de datos de la aplicación y la base de datos.

Desventajas

- **Adopción Limitada**: Menos común que las bases de datos relacionales y NoSQL.
- **Herramientas y Soporte**: Menor cantidad de herramientas y soporte en comparación con otros tipos de bases de datos.

---

4. Bases de Datos Distribuidas

Definición

Las bases de datos distribuidas almacenan datos en múltiples ubicaciones físicas pero se gestionan como una sola base de datos lógica. Esto permite distribuir la carga de trabajo y mejorar la disponibilidad y rendimiento.

Características

- **Distribución de Datos**: Los datos se distribuyen en múltiples servidores o nodos.
- **Alta Disponibilidad**: Diseñadas para ser altamente disponibles y tolerantes a fallos.
- **Consistencia Eventual**: Algunos sistemas priorizan la disponibilidad y partición sobre la consistencia (según el teorema CAP).

Ejemplos

- **Google Spanner**: Base de datos distribuida que combina las ventajas de las bases de datos relacionales y NoSQL.
- **Amazon DynamoDB**: Base de datos NoSQL gestionada y distribuida, diseñada para alto rendimiento y escalabilidad.

Ventajas

- **Escalabilidad Horizontal**: Fácil de escalar añadiendo más nodos.
- **Tolerancia a Fallos**: Mejora la disponibilidad y recuperación ante desastres.

Desventajas

- **Complejidad de Gestión**: Requiere gestión y configuración más complejas.
- **Consistencia**: Puede haber compromisos en la consistencia de los datos según el teorema CAP (Consistency, Availability, Partition tolerance).

Bibliografía:

- INTRODUCCIÓN A LAS BASES DE DATOS - MAX JOSÉ BERMÚDEZ LEÓN
- Introducción a las bases de datos - Rafael Camps Paré