**Bases de Datos**

El término bases de datos fue escuchado por primera vez en un simposio celebrado en California en 1963.

En una primera aproximación, se puede decir que una base de datos es un conjunto de información relacionada que se encuentra agrupada o estructurada.

Desde el punto de vista informático, una base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulen ese conjunto de datos.

Por su parte, un sistema de Gestión de Bases de datos es un tipo de software muy específico dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan; o lo que es lo mismo, una agrupación de programas que sirven para definir, construir y manipular una base de datos, permitiendo así almacenar y posteriormente acceder a los datos de forma rápida y estructurada.

Actualmente, las bases de datos están teniendo un impacto decisivo sobre el creciente uso de las computadoras.

**Historia**

Orígenes

Los orígenes de las bases de datos se remontan a la Antigüedad donde ya existían bibliotecas y toda clase de registros. Además, también se utilizaban para recoger información sobre las cosechas y censos. Sin embargo, su búsqueda era lenta y poco eficaz y no se contaba con la ayuda de máquinas que pudiesen reemplazar el trabajo manual.

Posteriormente, el uso de las bases de datos se desarrolló a partir de las necesidades de almacenar grandes cantidades de información o datos.

En 1884 Herman Hollerith creó la máquina automática de tarjetas perforadas, siendo nombrado así el primer ingeniero estadístico de la historia. En esta época, los censos se realizaban de forma manual.

Ante esta situación, Hollerith comenzó a trabajar en el diseño de una maquina tabuladora o censadora, basada en tarjetas perforadas.

Posteriormente, en la década de los cincuenta se da origen a las cintas magnéticas, para automatizar la información y hacer respaldos. Esto sirvió para suplir las necesidades de información de las nuevas industrias. Y a través de este mecanismo se empezaron a automatizar información, con la desventaja de que solo se podía hacer de forma secuencial.

Década de 1960

Posteriormente en la época de los sesenta, las computadoras bajaron los precios para que las compañías privadas las pudiesen adquirir; dando paso a que se popularizara el uso de los discos, cosa que fue un adelanto muy efectivo en la época, debido a que a partir de este soporte se podía consultar la información directamente, sin tener que saber la ubicación exacta de los datos.

En esta misma época se dio inicio a las primeras generaciones de bases de datos de red y las bases de datos jerárquicas, ya que era posible guardar estructuras de datos en listas y árboles. Otro de los principales logros de los años sesenta fue la alianza de IBM y American Airlines para desarrollar SABRE, un sistema operativo que manejaba las reservas de vuelos, transacciones e informaciones sobre los pasajeros de la compañía American Airlines. Y, posteriormente, en esta misma década, se llevó a cabo el desarrollo del IDS desarrollado por Charles Bachman (que formaba parte de la CODASYL) supuso la creación de un nuevo tipo de sistema de bases de datos conocido como modelo en red que permitió la creación de un standard en los sistemas de bases de datos gracias a la creación de nuevos lenguajes de sistemas de información.

CODASYL (Conference on Data Systems Languages) era un consorcio de industrias informáticas que tenían como objetivo la regularización de un lenguaje de programación estándar que pudiera ser utilizado en multitud de ordenadores.

Década de 1970

Por lo que respecta a la década de los setenta, Edgar Frank Codd, científico informático ingles conocido por sus aportaciones a la teoría de bases de datos relacionales, definió el modelo relacional a la par que publicó una serie de reglas para los sistemas de datos relacionales a través de su artículo “Un modelo relacional de datos para grandes bancos de datos compartidos”.

Este hecho dio paso al nacimiento de la segunda generación de los Sistemas Gestores de Bases de Datos. Como consecuencia de esto, durante la década de 1970, Lawrence J. Ellison, más conocido como Larry Ellison, a partir del trabajo de Edgar F. Codd sobre los sistemas de bases de datos relacionales, desarrolló el Relational Software System, o lo que es lo mismo, lo que actualmente se conoce como Oracle Corporation, desarrollando así un sistema de gestión de bases de datos relacional con el mismo nombre que dicha compañía. Posteriormente en la época de los ochenta también se desarrollará el SQL (Structured Query Language) o lo que es lo mismo un lenguaje de consultas o lenguaje declarativo de acceso a bases de datos relacionales que permite efectuar consultas con el fin de recuperar información de interés de una base de datos y hacer cambios sobre la base de datos de forma sencilla; además de analiza grandes cantidades de información y permitir especificar diversos tipos de operaciones frente a la misma información, a diferencia de las bases de datos de los años ochenta que se diseñaron para aplicaciones de procesamiento de transacciones.

Pero cabe destacar que ORACLE es considerado como uno de los sistemas de bases de datos más completos que existen en el mundo, y aunque su dominio en el mercado de servidores empresariales ha sido casi total hasta hace relativamente poco, actualmente sufre la competencia del SQL Server de la compañía Microsoft y de la oferta de otros Sistemas Administradores de Bases de Datos Relacionales con licencia libre como es el caso de PostgreSQL, MySQL o Firebird que aparecerían posteriormente en la década de 1990.

Década de 1980

Por su parte, a principios de los años ochenta comenzó el auge de la comercialización de los sistemas relacionales, y SQL comenzó a ser el estándar de la industria, ya que las bases de datos relacionales con su sistema de tablas (compuesta por filas y columnas) pudieron competir con las bases jerárquicas y de red, como consecuencia de que su nivel de programación era sencillo y su nivel de programación era relativamente bajo.

Década años 1990

En la década de 1990 la investigación en bases de datos giró en torno a las bases de datos orientadas a objetos. Las cuales han tenido bastante éxito a la hora de gestionar datos complejos en los campos donde las bases de datos relacionales no han podido desarrollarse de forma eficiente.

¿Por qué son necesarias las bases de datos?

Hoy, los sistemas de bases de datos son imprescindibles en numerosos campos. Cualquier tipo de software concebido para las empresas se basa en robustas bases de datos con un gran número de opciones y herramientas para los administradores del sistema. La seguridad de los datos, además, ha ido ganando importancia con el tiempo, y es que en las bases de datos electrónicas se almacenan y cifran contraseñas, datos personales e incluso divisa digital. El sistema financiero moderno, no es más que una red de bases de datos, en la cual la mayor parte de las cuantías monetarias solo existen como unidades electrónicas de información, cuya protección, por medio de bases de datos seguras es una de las tareas principales de las instituciones financieras. Aunque no solo por esto son cruciales las bases de datos electrónicas para la civilización moderna.

Funciones y condiciones de un sistema de gestión de base de datos (SGBD)

Un término muy extendido para describir las funciones y los requisitos de las transacciones en un database management system es el de ACID, acrónimo de atomicity, consistency, isolation y durability (atomicidad, consistencia, aislamiento, durabilidad). Estos cuatro parámetros, cubren los requisitos más importantes de un SGBD (ACID compliant):

* **Atomicidad** designa a la propiedad “todo o nada” de los gestores de bases de datos: para que una consulta sea válida y la transacción se complete correctamente se ha de llevar a cabo en el orden correcto de pasos.
* **La consistencia** (o coherencia) se da cuando al finalizar una transacción, la base de datos sigue siendo estable, lo que requiere la supervisión continua de todas las transacciones.
* **El aislamiento** es la condición que garantiza que las transacciones no se obstaculicen unas a otras, algo que normalmente se logra con ciertas funciones de bloqueo que aíslan los datos que participan en una transacción.
* **La durabilidad** significa que en un SGBD todos los datos se guardan a largo plazo incluso tras concluir una transacción y también, o especialmente, en el caso de fallos del sistema o caídas del SGBD. Para esta condición, son esenciales los registros de transacción, que protocolizan todos los procesos que tienen lugar en el SGBD.

Funciones:

* Almacenar datos
* Editar datos
* Borrar datos
* Gestionar los metadatos
* Seguridad de los datos
* Integridad de los datos
* Función multiusuario
* Optimizar las consultas
* Transparencia del sistema

Modelos

Modelo Jerárquico

Este es el modelo más antiguo, hoy superado en gran medida por el modelo relacional (entre otros), si bien recientemente su empleo ha ido creciendo. XML utiliza este sistema para guardar datos y algunas compañías de seguros y bancos recurren a las bases de datos jerárquicas sobre todo en las aplicaciones más antiguas de base de datos. El sistema de base de datos jerárquico más conocido es IMS/DB de IBM.

En las bases de datos jerárquicas las dependencias son inequívocas. Cada registro tiene solo un precedente (Parent-Child Relationships, PCR) a excepción de la raíz (root), constituyendo un esquema en árbol como el de arriba. Mientras que cada nodo “hijo”, solo puede tener un nodo “padre”, los “padres” pueden tener tantos “hijos” como quieran. Dado el estricto ordenamiento jerárquico, los niveles sin relación directa, no interactúan entre sí y conectar dos árboles diferentes tampoco es fácil. Por todo esto, las estructuras de base de datos jerárquicas son extremadamente inflexibles, pero muy claras.

Los registros con hijos se llaman records y los que no tienen se llaman hojas y son los que suelen contener los documentos. Los records sirven para clasificar las hojas. Las consultas a una base de datos jerárquica alcanzan a las hojas partiendo desde la raíz y pasando por los distintos records.

Interfaz de usuario gráfica, Diagrama, Aplicación

Descripción generada automáticamente

Base de datos en red

El modelo en red se desarrolló casi de forma simultánea al relacional, aunque con el tiempo sería superado por la competencia. A diferencia del modelo jerárquico, aquí los registros o records no revelan relaciones padre-hijo estrictas, sino que cada registro puede tener múltiples precedentes, lo que le da la estructura en red de su nombre. Para acceder a un registro tampoco hay, por eso mismo, un camino único e invariable.

Diagrama

Descripción generada automáticamente

Hoy el modelo de base de datos en red se utiliza, sobre todo, en los grandes ordenadores. En otros campos se sigue confiando en el modelo jerárquico (clientes de IBM, sobre todo) o se ha dado el paso hacia el modelo relacional, mucho más flexible y fácil de utilizar. Algunos modelos conocidos de base de datos en red son el UDS de Siemens y el DMS de Sperry Univac. Con el tiempo, ambos fabricantes han desarrollado también interesantes formas mixtas entre el modelo en red y el relacional, aunque sin lograr arrancar del todo.

Modelo de base de datos relacional

El modelo que goza de más popularidad hoy es el relacional, aunque tampoco queda libre de crítica. Su correspondiente sistema de gestión es más conocido como SGBDR (RDBMS en inglés) y como lenguaje utiliza normalmente SQL. Este modelo basado en tablas, gira en torno al concepto de relación, un término bien definido en matemáticas y que aquí se utiliza como sinónimo de tabla. Para formular las relaciones se utiliza álgebra relacional, con cuya ayuda puede obtenerse la información de estas relaciones. Este es el principio que fundamenta el lenguaje SQL.

Escala de tiempo

Descripción generada automáticamente con confianza baja

El modelo relacional trabaja con tablas independientes que determinan la localización de los datos y sus conexiones. Estos datos conforman un registro (en la imagen, una fila o “tupla”) y se guardan en columnas como atributos (en la imagen, de A1 a An). La relación es lo que resulta de los atributos interrelacionados. Para identificar inequívocamente un registro es elemental la clave primaria, que normalmente se define como el primer atributo (A1) y que no puede cambiarse. Dicho de otra manera, esta clave primaria o ID, define la posición exacta del registro con todos los atributos.

Modelo de base de datos orientado a objetos

Las bases de datos de objetos no nacen hasta finales de 1980 y hasta hoy, solo han encontrado una escasa aplicación. Estas bases de datos, disponibles también en formato open source, suelen utilizarse en plataformas Java y .NET. La más conocida es db4o, que destaca, sobre todo, por un escaso uso de la memoria. Las bases de datos de objetos acostumbran a trabajar con el lenguaje OQL, muy similar a SQL.

Diagrama

Descripción generada automáticamente

En el modelo orientado a objetos, los datos se guardan en un objeto junto con sus funciones (métodos) y los atributos que los describen más en profundidad. En un sistema de gestión de bases de datos de objetos, son los métodos, depositados en el objeto junto con los datos, los que definen cómo se accede al objeto.

Modelo de base de datos orientado a documentos

En este modelo, los documentos son la unidad básica para el almacenamiento de datos. Estas unidades son las que estructuran los datos y no deben confundirse con los documentos de los programas de procesamiento de texto. Aquí, los datos se guardan en los llamados pares clavevalor, comprendiendo así, una “clave” y un “valor”. Como no están definidos ni la estructura ni el número de pares, los documentos que integran una base de datos orientada a documentos pueden resultar muy dispares entre sí. Cada documento es una unidad cerrada en sí misma y establecer relaciones entre documentos no resulta fácil, pero en este modelo no es necesario.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Edgar Frank Codd**

Nacido en Inglaterra el 23 de agosto de 192, Edgar es considera el padre de las bases de datos relacional.

**12 reglas de Codd**

Codd se percató de que existían bases de datos en el mercado las cuales decían ser relacionales, pero lo único que hacían era guardar la información en las tablas, sin estar estas tablas literalmente normalizadas; entonces éste publicó 12 reglas que un verdadero sistema relacional debería tener, aunque en la práctica algunas de ellas son difíciles de realizar. Un sistema podrá considerarse “más relacional” cuanto más siga estas reglas.

* ***Regla 0***: el sistema debe ser relacional, base de datos y administrador de sistema. Ese sistema debe utilizar sus facilidades relacionales (exclusivamente) para manejar la base de datos.
* ***Regla 1***: la regla de la información, toda la información en la base de datos es representada unidireccionalmente, por valores en posiciones de las columnas dentro de filas de tablas. Toda la información en una base de datos relacional se representa explícitamente en el nivel lógico exactamente de una manera: con valores en tablas.
* ***Regla 2***: la regla del acceso garantizado, todos los datos deben ser accesibles sin ambigüedad. Esta regla es esencialmente una nueva exposición del requisito fundamental paralas llaves primarias. Dice que cada valor escalar individual en la base de datos debe ser lógicamente direccionable especificando el nombre de la tabla, la columna que lo contiene y la llave primaria.
* ***Regla 3***: tratamiento sistemático de valores nulos, el sistema de gestión de base de datos debe permitir que haya campos nulos. Debe tener una representación de la “información que falta y de la información inaplicable” que es sistemática, distinto de todos los valores regulares.
* ***Regla 4***: catálogo dinámico en línea basado en el modelo relacional, el sistema debe soportar un catálogo en línea, el catálogo relacional debe ser accesible a los usuarios autorizados. Es decir, los usuarios deben poder tener acceso a la estructura de la base de datos (catálogo).
* ***Regla 5***: la regla comprensiva del sublenguaje de los datos, el sistema debe soportar por lo menos un lenguaje relacional que:
  + Tenga una sintaxis lineal.
  + Puede ser utilizado de manera interactiva.
  + Soporte operaciones de definición de datos, operaciones de manipulación de datos (actualización, así como la recuperación), seguridad e integridad y operaciones de administración de transacciones.
* ***Regla 6***: regla de actualización, todas las vistas que son teóricamente actualizables deben ser actualizables por el sistema.
* ***Regla 7***: alto nivel de inserción, actualización, y cancelación, el sistema debe soportar suministrar datos en el mismo tiempo que se inserte, actualiza o esté borrando. Esto significa que los datos se pueden recuperar de una base de datos relacional en los sistemas construidos de datos de filas múltiples y/o de tablas múltiples.
* ***Regla 8***: independencia física de los datos, los programas de aplicación y actividades del terminal permanecen inalterados a nivel lógico cuandoquiera que se realicen cambios en las representaciones de almacenamiento o métodos de acceso.
* ***Regla 9***: independencia lógica de los datos, los cambios al nivel lógico (tablas, columnas, filas, etc.) no deben requerir un cambio a una solicitud basada en la estructura. La independencia de datos lógica es más difícil de lograr que la independencia física de datos.
* ***Regla 10***: independencia de la integridad, las limitaciones de la integridad se deben especificar por separado de los programas de la aplicación y se almacenan en la base de datos. Debe ser posible cambiar esas limitaciones sin afectar innecesariamente las aplicaciones existentes.
* ***Regla 11***: independencia de la distribución, la distribución de las porciones de la base de datos a las varias localizaciones debe ser invisible a los usuarios de la base de datos. Los usos existentes deben continuar funcionando con éxito:
  + Cuando una versión distribuida del SGBD se introdujo por primera vez
  + cuando se distribuyen los datos existentes se redistribuyen en todo el sistema.
* ***Regla 12***: la regla de la no subversión, si el sistema proporciona una interfaz de bajo nivel de registro, a parte de una interfaz relacional, que esa interfaz de bajo nivel no se pueda utilizar para subvertir el sistema, por ejemplo: sin pasar por seguridad relacional o limitación de integridad. Esto es debido a que existen sistemas anteriormente no relacionales que añadieron una interfaz relacional, pero con la interfaz nativa existe la posibilidad de trabajar no relacionalmente.