Introducción

Entre los años 1960 y 1970 existían computadoras con capacidades limitadas, tanto en procesamiento como en capacidad de almacenamiento. Si bien las aplicaciones no eran tan exigentes como las actuales había que trabajar demasiado para lograr los resultados esperados.

Los datos que se necesitaban almacenar se ubicaban en archivos, con un orden físico dado, por el orden en que se ingresaban o por un orden indicado, que generaba demasiado trabajo. Los registros se conformaban por campos y eran separados con algún símbolo, que luego debían ser detectados para saber donde finalizaba uno y empezaba el otro.

La lectura secuencial de los archivos era muy común, donde se debían recorrer todos los registros para encontrar uno buscado, acarreando problemas en velocidad de respuesta.

También se repetía la situación en la que distintos sectores de una empresa contaban con sus propios datos para resolver los problemas de información que se presentaban. Esto derivaba en que había datos repetidos, sin que hubiera políticas y reglas muy claras, como tampoco personal informático que centralizara el tratamiento de ese tipo de situaciones.

Se necesitaba un esfuerzo mayor al actual en la programación de aplicaciones para acceder a los datos, donde el almacenamiento se centraba en pocos tipos de datos, como numéricos y de texto, lo cual limitaba las capacidades de respuesta a las necesidades de las empresas y exigía algoritmos de programación hasta para controlar y almacenar una fecha.

Actualmente todas las personas interactúan con datos, que de alguna manera deben estar almacenados en un medio físico y asociados a un sistema informático que los registra y accede.

Por ejemplo: cuando desde su casa o desde su teléfono móvil marca un número telefónico está utilizando datos registrados en algún sistema de una empresa telefónica, que reconoce su teléfono, origen de la llamada, y el teléfono del receptor. Eso implica una serie de acciones que desencadenan controles y registros en una base de datos. Como control podemos deducir que debe verificar que el número del receptor exista, de lo contrario es una llamada equivocada, o por ejemplo chequear que no esté ocupado el receptor. Una vez establecida la comunicación el sistema informático registrará mínimamente su número, el número destino, hora de inicio de la llamada y hora final al momento de dar por finalizada la comunicación. Dichos registros almacenados son los que permiten obtener la facturación de los clientes y todos los reportes que se soliciten.

Lo mismo sucede cuando un cliente de un banco interactúa con el cajero automático para retirar dinero. El cliente debe ingresar su tarjeta, que será leída, para luego cruzar la identificación de la tarjeta con la clave que el usuario ingresa. La base de datos mantiene los datos del cliente y sus cuentas asociadas, mientras que una aplicación informática hará controles y mostrará las opciones disponibles al usuario.

En alguna sección de un sistema de gestión de alumnos de la Universidad el docente publica el programa de estudios de la materia específica que dicta, fechas de exámenes parciales y notas de los exámenes parciales de los alumnos de su curso, como también publica mensajes y archivos para que sean accedidos por sus alumnos. Hay personal de la universidad que carga los resultados de los exámenes, indicando las notas de los alumnos presentes y registra los ausentes. Otros usuarios acceden al sistema para cargar asistencias e inasistencias a clase de los alumnos. Por su parte los alumnos ingresan con su legajo y clave a la autogestión que presenta el sistema, vía intranet o vía internet, para ver información de su estado académico, conformado por las materias que cursó y cursa, resultados en exámenes, parciales y finales, cantidad de faltas que posee, e incluso puede inscribirse a un examen de las materias que ya posee regularizada.

¿Qué es entonces una base de datos? Algunos pueden pensar que son sólo datos almacenados en una computadora, como puede ser una planilla de cálculos. Pero es mucho más que eso.

Definición de Base de Datos

Cuando una empresa decide implementar un sistema informático, para cubrir necesidades de información específicas, el equipo de análisis y diseño inicia el proceso de desarrollo del software basándose en las especificaciones que trabaja con el cliente que solicita el sistema, en búsqueda de responder a las necesidades de la empresa.

Durante el proceso se definen archivos como el de Estudiantes para almacenar los datos de todos los alumnos de la Universidad, el de Profesores para almacenar los datos de contacto necesarios, Carreras que propone la Universidad y Materias que conforman cada Plan de Estudios vigentes o no para cada Carrera, etc.

Pero esos archivos nunca quedan aislados de otros datos, es decir que mantienen relaciones que normalmente son transparentes para los usuarios y permiten su aprovechamiento para la obtención de información a través de aplicaciones programadas a tales fines. Por ejemplo es necesario tener algún archivo que mantiene datos de la Inscripción de cada alumno, tanto en las Carreras como en las Materias a cursar, eso permite luego poder reflejar datos de resultados de parciales y exámenes, asistencias, etc.

Entonces podemos ensayar una primera definición diciendo que “una base de datos es un conjunto de datos estructurados y definidos a través de un proceso específico, en búsqueda de evitar la redundancia y para ser almacenados en algún medio de almacenamiento masivo, como actualmente es un disco”.

Cuando decimos redundancia nos referimos a la repetición de datos que puede producirse al definir los almacenamientos de datos. Pretender definir que los datos como apellido y dirección de un alumno estén tanto en un almacenamiento Estudiantes pero también en los almacenamientos relacionados a los exámenes no es correcto, ya que un cambio en la dirección de un alumno puede significar que la modificación se haga en uno de los

almacenamientos pero no en el otro, dando inconvenientes a la hora de devolver información de un alumno determinado.

Sistema de Gestión de Bases de Datos

Cuando el alumno desea información de las notas obtenidas durante el último año de su cursado, que está almacenada en una base de datos, no accede directamente a los datos almacenados. Cada alumno de la Universidad utiliza aplicaciones que han sido desarrolladas para fines específicos, como es obtener un reporte de notas.

Por su parte las aplicaciones que usa interactúan con un conjunto de programas aglutinados en lo que se llama el Sistema de Gestión de Bases de Datos (SGBD), Database Management System (DBMS) e incluso llamado Motor de Base de Datos. A este Motor de Base de Datos se lo puede pensar de manera simplificada como una capa de software que controla todos los accesos a la base de datos. Cabe aclarar que un DBMS no se crea para una situación específica de una empresa, sino que debe desempeñarse tanto para fines de un Sistema de Gestión de Alumnos como para un Sistema Bancario, Empresa Telefónica, Comercial, etc.

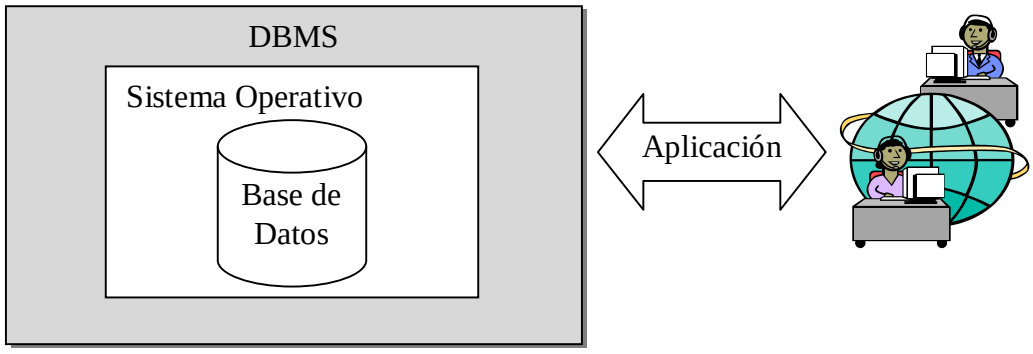

Fig. 6.1

Es decir que la Base de Datos se encuentra protegida por software, conformado por el DBMS y los módulos de seguridad. Si bien las aplicaciones poseen controles cuando el usuario ingresa al sistema, normalmente se define una segunda instancia que es lo definido al DBMS y no hay manera que eso sea evitado.

Así mismo el DBMS no accede directamente a direcciones físicas donde están almacenados los datos, sino que lo hace a través del Sistema Operativo, quien hace de interfaz entre el DBMS y los datos almacenados.

Con esto podemos ampliar la definición de bases de datos enunciada anteriormente con una muy citada por otros autores, que abarca los conceptos mencionados hasta el momento y que es la siguiente:

“Colección de datos interrelacionados almacenados en conjunto sin redundancias perjudiciales o innecesarias; su finalidad es servir a una aplicación o más, de la mejor manera posible; los datos se almacenan de modo que resulten independientes de los

programas que los usan; se emplean métodos bien determinados para incluir nuevos datos y para modificar o extraer los datos almacenados". (Martin, 1975)

Esta definición incluye el concepto de independencia de datos, lo cual significa que las aplicaciones deben ser independientes de las modificaciones que pueden sufrir los datos, tanto en lo lógico como en lo físico. Esto es algo muy deseable de lograr en las bases de datos, pero en algunas situaciones es más o menos complejo de lograr.

Es decir que si, por ejemplo, en el almacenamiento Estudiantes de la base de datos se agrega un dato, como puede ser la cuenta de correo o el número de teléfono móvil, eso no debe implicar hacer cambios en los programas que usan el almacenamiento. Eso es *independencia lógica*. En otras épocas las aplicaciones incluían la definición de la estructura de datos, habiendo una sección donde figuraban todos los datos que poseía cada uno de los archivos de datos que eran accedidos por dicho programa. Significaba que ante un cambio en la estructura, agregado de un nuevo dato o se alteraba la dimensión de un dato existente, las aplicaciones, que podían ser muchas, debían reflejar el cambio en la sección correspondiente.

En cuanto a la *independencia física*, el almacenamiento Estudiantes puede definirse con un acceso rápido por legajo del estudiante, lo que se denomina indexación. Si esa forma de acceso fuera cambiada para mejorar la performance del sistema tampoco debe determinar un cambio en las aplicaciones. Tampoco debería generar cambios en las aplicaciones si se decidiera cambiar de almacenamiento físico a la base de datos.

Usuarios de la Base de Datos

Hay diversas clasificaciones de los usuarios que actúan en un entorno de bases de datos, como las planteadas por grandes autores como C.J. Date[1], Korth[2] y Ramez Elmasri y Shamkant Navathe[3].

Estudiaremos distintos momentos de un sistema con bases de datos e iremos detectando los perfiles que actualmente se repiten en las empresas, a los cuales les asignaremos una denominación y de esa manera quedarán clasificados.

Cuando un nuevo sistema de información desea ser informatizado es conveniente, diríamos imprescindible, que participe un equipo de análisis y diseño de sistemas, conformado por Ingenieros en Sistemas, Analistas de Sistemas, etc. Este equipo es el que toma los requerimientos del cliente –mediante entrevistas, cuestionarios, etc.-, plasma en diagramas y descripciones la interpretación que hicieron de ese relevamiento de necesidades, para luego entrar en un intenso proceso ingenieril, que incluye análisis, creación y diseño. El equipo plantea una propuesta al cliente para ser revisada y mejorada, con los alcances y límites bien determinados. Estando la propuesta aceptada, se mejora el nivel de detalle del diseño, intercambian opiniones con los usuarios futuros del sistema respecto a interfaces y reportes,

[1] C.J.Date-Introducción a los Sistemas de Bases de Datos-7ma. Edición
[2] KORTH/SILBERSCHATZ/SUDARSHAN - Fundamentos de Bases de Datos – Quinta Edición
[3] ELMASRI/NAVATHE - Sistemas de Bases de Datos – Conceptos Fundamentales – Segunda Edición

para llegar a estipular cuáles son las entidades que participan en el desarrollo de las aplicaciones, cuáles las estructuras de cada una, los usuarios y los roles de cada uno, etc.

De esta manera estamos resumiendo en pocas líneas un proceso muy importante, que lleva días y hasta meses de trabajo según complejidad del sistema, pero la intención es indicar que este equipo no interactúa necesariamente con el sistema en producción, sino en la etapa previa a la construcción, lo cual hace que nosotros no los consideremos usuarios del sistema.

A partir del reconocimiento de las entidades y relaciones entre ellas participan determinados usuarios en la vida del sistema y que podemos definir así:

- **Administrador de la Base de Datos:** comúnmente es el profesional –Ingeniero, Analista-, con perfil técnico, que en el ambiente informático se denomina DBA (Data Base Administrator). Este profesional, muy importante para la empresa, recibe las especificaciones del Equipo de Análisis y Diseño para implementar en un Sistema de Gestión de Base de Datos, como puede ser actualmente un Oracle, SQL Server, DB2, etc. El DBA tiene múltiples funciones y responsabilidades que se detallan en un apartado siguiente.
- **Programador de Aplicaciones:** conociendo los casos a desarrollar -escritos e identificados por el Equipo de Análisis y Diseño- los prototipos de interfaces y las estructuras de los almacenamientos a manipular, este profesional genera las aplicaciones necesarias en el sistema, con el lenguaje de programación que se le indica y conoce, tanto para obtener las entradas de datos que alimentan a la base de datos como para lograr las salidas, como pantallas de resultados o reportes, que se plantearon en la propuesta de solución. Es conveniente aclarar que este usuario no necesita conocer toda la estructura de la base de datos, sino lo que realmente necesita para programar, como también que normalmente, y dependiendo de la envergadura del sistema, trabaja en un equipo de desarrollo.
- **Usuario Final:** es el personal que interactúa con las aplicaciones programadas por el usuario anteriormente nombrado y es el que menos conocimiento técnico posee de todos los usuarios. Si bien hay perfiles distintos entre los usuarios finales que interactúan con el sistema, esta persona no conoce los detalles técnicos del sistema ni de la base de datos. Por ejemplo: un alumno que accede a su autogestión vía internet para conocer su estado académico no necesita conocer la estructura del sistema, sólo utiliza las opciones que se le presentan en el menú y está asociado a un perfil definido para todos los alumnos. Otro usuario final del Sistema de Gestión de Alumnos es el profesor de un curso, quien verá opciones definidas para ese perfil, con posibilidad de cargar notas de parciales, enviar mensajes a todo el curso o imprimir lista de sus alumnos.

  Durante la vida del sistema seguramente se presentan requerimientos que cambian y otros que no fueron tenidos en cuenta al diseñar el sistema. En este último caso es lógico que no estén los programas que determinados usuarios necesitan. Por ejemplo, es muy común que personal directivo especifique una necesidad como "Obtener los datos de los alumnos que cursan el último año de la Carrera, trabajan,

tienen promedio superior a 9 (nueve) y no han rendido materias en los últimos 2 turnos de examen". Este tipo de solicitud no programada, suele dar lugar a que determinados usuarios tengan permisos para acceder a la base de datos mediante consultas que ellos mismos arman en pantalla, indicando los almacenamientos a usar, condiciones a cumplir, datos a mostrar y cálculos a realizar. Si esta solicitud se hace repetida en el tiempo puede llegar a decidirse que es conveniente incluirla como opción del sistema.

Funciones y responsabilidades de un DBA

Habíamos dicho que el DBA (Data Base Administrator) tiene múltiples funciones técnicas y responsabilidades con el sistema. En algunas organizaciones el DBA suele trabajar solo, pero es muy común que necesite de personal a cargo para hacer las tareas operacionales que se necesitan y lograr cubrir la alta demanda de tiempo actual, por ejemplo para atender demandas de empresas con servicios internacionales y uso del sistema en 7díasx24horas, es decir todas las horas de todos los días de la semana del año, como puede ser por ejemplo un sistema de vuelos internacionales.

Entre sus funciones principales podemos citar:

- ***Especificación lógica de la Base de Datos:*** El DBA especifica, mediante la interfaz del DBMS elegido o sentencias de definición de datos, la estructura de la Base de Datos que recibió del Equipo de Análisis y Diseño. Indica cuáles son los datos de cada entidad, los tipos de datos, la dimensión de cada dato, las relaciones entre ellos, claves identificadas, vistas para los usuarios finales, etc. Las sentencias que utiliza el DBA se agrupan en lo que se llama Lenguaje de Definición de Datos (DDL) y está formada por instrucciones para trabajar en la estructura, como para crear objetos, modificar estructuras de los objetos o eliminar objetos que ya no se necesitan en la base de datos.
- ***Especificación física:*** El DBA define el medio físico que almacenará a la base de datos, por ejemplo el disco y partición en el servidor, como también como se almacenarán los archivos de datos y cómo serán accedidos los distintos archivos para poder lograr una mejor performance.
- ***Definición de seguridad:*** Define grupos de usuarios y usuarios individuales, con los perfiles para cada uno, indicando los archivos a los que puede acceder cada usuario y los derechos que posee sobre cada uno. Por ejemplo: define el perfil Estudiante permitiendo el derecho de consultar sobre los archivos que contienen datos de sus evaluaciones, consultar y enviar mensajes con los docentes, tener el derecho de modificar su contraseña, pero no tendrá posibilidad de ver datos personales de sus docentes o el derecho de modificar notas de evaluaciones.
- ***Definir procedimiento de respaldo:*** Es responsabilidad del DBA asegurar que los datos estén respaldados para evitar inconvenientes ante algún tipo de incidente (rotura de medio físico, errores de procedimientos en actualizaciones, robo de hardware, incendio, etc.) por lo que deberá definir la periodicidad de los respaldos, el o los medios para mantener los respaldos (backup), si copiará todo el contenido de

la base de datos cada vez que inicia el respaldo o si será parcial, etc. Con esto no se dice que el DBA lo haga, sino que define el procedimiento y deberá ser ejecutado por personal de soporte en la empresa. Lo mismo sucede con el procedimiento para recuperar los datos y quien lo hará.

- ***Implementar reglas de integridad:*** El Equipo de Análisis y Diseño especifica ciertas limitaciones a los datos a almacenar, dados por el mundo real de la organización o por reglas lógicas propias de las bases de datos. Por ejemplo: en determinados países la nota que un estudiante puede obtener está entre el valor 0 (cero) y el 10 (diez), eso implica que el dato Nota nunca podrá estar fuera de esos límites. Otro ejemplo de integridad puede observarse en un archivo de exámenes, donde el estudiante que se registre debe ser parte del alumnado de la Universidad y la materia que rinde debe ser integrante del plan de estudios de la Carrera que cursa el alumno. Si analizamos un sistema bancario también tiene sus restricciones propias del negocio, al igual que una empresa de telefonía también las posee. Estas reglas se definen durante el proceso de creación de las bases de datos y de los archivos que la conforman, incluso puede agregarse algunas nuevas que surgen durante el uso del sistema y mientras los datos lo permitan, ya que algunos seguramente existen.
- ***Monitorear la performance de la Base de Datos:*** El DBA debe monitorear el rendimiento de la Base de Datos, detectando procesos que generen demoras en la devolución de información y mejorando permanentemente la performance general de la Base de Datos. En algunos casos el resultado es que los desarrolladores deben revisar las aplicaciones desarrolladas o la arquitectura del sistema, en otras situaciones los métodos de acceso pueden ser mejorados, incluso la ubicación de los archivos de datos o las capacidades del hardware o de la conectividad influyen en el rendimiento. Hay situaciones que pueden ser resueltas por el DBA, pero otras sólo pueden ser informadas a quienes toman decisiones para que busquen soluciones.

Arquitectura ANSI/SPARC

Como las funciones de los distintos usuarios mencionados de una base de datos son distintas e implica que cada uno puede interactuar de diversas formas con esos datos almacenados, surge esta propuesta para estandarizar los conceptos y permitir una mejor lectura de la independencia de datos, permitiendo ubicar a cada usuario de una base de datos en función de su relación con la misma, ya que no todos tienen igual visión aunque los datos almacenados son únicos.

ANSI (American National Standards Institute) tuvo mucha influencia en la definición de esta arquitectura. Este Instituto fue fundado el 19 de octubre 1918 y se autodefine en *www.ansi.org* como el instituto que *"supervisa la creación, promulgación y el uso de miles de normas y directrices que impactan directamente en casi todos los sectores de las empresas: desde dispositivos acústicos hasta los equipos de construcción, desde la producción lechera y ganadera hasta la distribución de energía, y muchos más"*.

La arquitectura que se presentará en tres niveles tiene su antecedente en el año 1971, cuando Codasyl[4] reconoce la necesidad de una arquitectura en dos niveles, un esquema –vista de sistema- y un sub esquema –vista de usuario-.

Luego de eso, ANSI se dedica al tema, como lo dicen en el libro "Databases, role and structure: an advanced course"[5], donde el autor indica que ANSI cuenta con un comité conocido como X3 dedicado a Computación y procesamiento de información. El SPARC (Standards Planning and Requirements Committee) de ANSI/X3 fija un grupo de estudio de DBMS para especificar modelos para la estandarización de bases de datos. Este comité produce en 1972 un reporte provisional seguido por un reporte final en 1977, donde divide la vista de sistema -destacada por Codasyl- en vista de empresa y vista de almacenamiento, quedando lo que actualmente se conocen como:

- **Nivel Externo:** lo conforman las múltiples vistas de los datos almacenados en la base de datos y que se presentan a los distintos usuarios de múltiples formas, adecuándolas a las necesidades de información que tiene cada uno. A este nivel también se lo denomina Nivel de Visión y se define con el Lenguaje de Manipulación de Datos.
- **Nivel Conceptual:** es la estructura lógica global, que representa las estructuras de datos y las relaciones entre ellos. Hay una única vista en este nivel y se lo define con el Lenguaje de Definición de Datos.
- **Nivel Interno:** se describe como los datos están almacenados físicamente y como son accedidos, por esto también se denomina Nivel Físico.

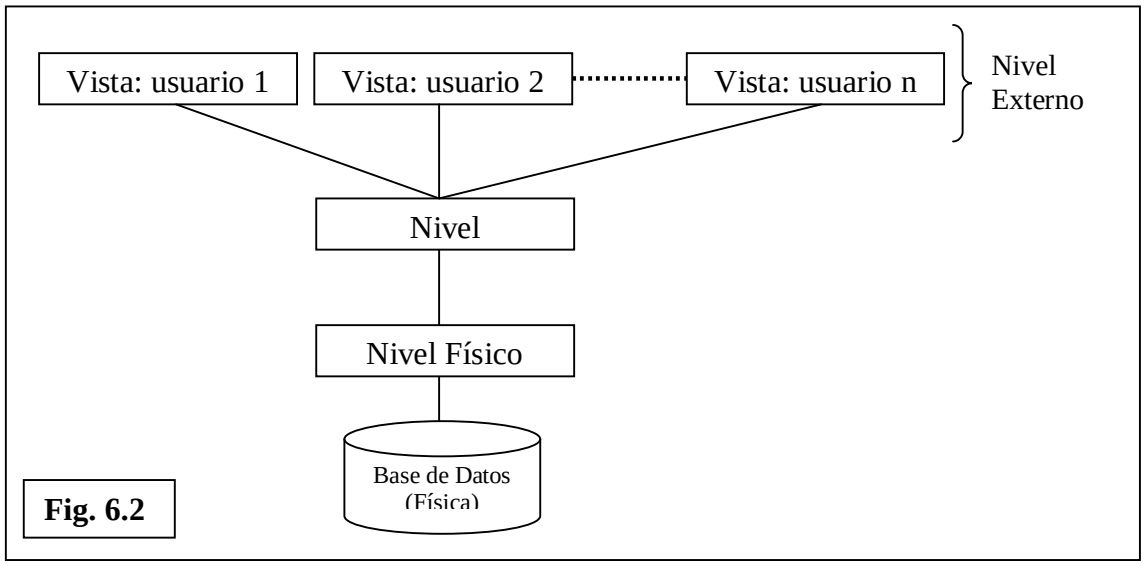

Fig. 6.2

Veamos la relación de los usuarios con estos niveles de abstracción, por ejemplo: los Usuarios Finales y Programadores de Aplicaciones no necesitan conocer como están

[4] CODASYL (Conference on Data Systems Languages): Comité formado por personas de industrias informáticas, con el objetivo de regular el desarrollo de un lenguaje de programación estándar, de donde surge el lenguaje COBOL.

[5] Databases, role and structure: an advanced course. Escrito por P. M. Stocker,Peter M. D. Gray,M. P. Atkinson. Univ. Cambridge

almacenados los datos físicamente, sino que tienen una visión alejada del Nivel Físico y personalizada para cubrir las necesidades que tiene cada uno. Por ejemplo: un cajero de una entidad bancaria no necesita conocer los mismos datos que un gerente, ni un estudiante universitario necesita acceder a todas las opciones que usa un profesor de la universidad. Un reporte presentado a un gerente puede ser el resultado de un largo proceso de combinar datos provenientes de distintos archivos y haber aplicado múltiples cálculos, pero eso es trasparente para él, ya que no conoce ni le interesa conocer estos detalles técnicos.

Otro usuario, el DBA, está en condiciones y es el único que debería conocer el Nivel Físico, para poder modificarlo si hiciera falta. Por ejemplo: cambiar el medio de almacenamiento de la base de datos o la forma de acceso a los archivos, manteniendo siempre inalterables las vistas para los demás usuarios.

Sólo el DBA puede introducir cambios en el Nivel Conceptual, pero sí es necesario que mantenga las vistas inalteradas hacia los usuarios finales y programadores de aplicaciones. Por ejemplo: agregar una columna a un archivo de datos cuando este no fue tenido en cuenta inicialmente en el sistema o agregar un archivo de datos con las relaciones que corresponda a los ya existentes.

Un archivo de datos de Productos y otro de Proveedores, de una base de datos de una empresa que vende insumos y artículos informáticos, podrían tener este aspecto:

Ejemplo:

Productos					
CodProd	**DesProd**	**PrecioVta**	**CodProv**	**Stock**	**StMin**
108	IMP. MULTIF. TX410	699,80	19	2	6
230	MON. 19 PULG. LCD W1943C	925,50	32	12	10
110	IMP. MULTIF. C5580-CH.TIN.	769	21	3	6
…	…	…	…	…	…

Proveedores			
CodProv	**RazónSocial**	**Contacto**	**Teléfono**
21	HP Center	…	…
19	Moura EPSON	…	…
32	Asoc. LG	…	…
…	…	…	…

Físicamente estos archivos de la base de datos están almacenados de una única manera, que en definitiva es un conjunto de bytes y no son visibles a ningún tipo de usuario, lo cual se administra a través del sistema operativo. Pero el DBA, utilizando herramientas e instrucciones del DBMS, puede organizarlo indicando en qué disco se almacena la base de datos (disco local o remoto, en uno u otro servidor) y definiendo para cada archivo su

método de acceso, por ejemplo generando un índice por código de producto en el archivo Productos y agilizando así la búsqueda de un producto conociendo su código.

En el Nivel Conceptual o de Estructura Lógica el DBA puede crear las estructuras de estos archivos o modificarlas, por ejemplo indicando que el dato Código del Producto es numérico con 4 dígitos posibles (0 a 9999), se autogenera cada vez que se inserta un nuevo producto y nunca puede repetirse. Aquí también es necesario identificar la relación entre los archivos de Productos y de Proveedores, determinando como se produce una búsqueda o qué sucede cuando un proveedor quiere ser eliminado de la base de datos.

En el Nivel Externo, cada usuario verá lo que necesita y tiene habilitado, por ejemplo:

- El vendedor podrá ver en su pantalla un listado ordenado por descripción del producto, con cabeceras de listado distinta a los nombres de los campos de datos en el archivo origen:

Producto	**Descripción**	**Precio**	**Stock Actual**
108	IMP. MULTIF. TX410	699,80	2
110	IMP. MULTIF. C5580-	769	3
230	CH.TIN.	925,50	12
…	MON. 19 PULG. LCD W1943C	…	…
	…		

  A él no le interesa ni quien es el proveedor ni cuál es el stock mínimo de cada producto.
  Tampoco sabe di todos estos datos provienen de un solo almacenamiento o de una combinación.

- El encargado de compras visualizará un listado con combinación de datos de los dos archivos, con cabeceras más descriptivas e incluyendo el resultado de un cálculo indicando de lo que debería comprar como mínimo para cubrir el stock mínimo por producto:

			Stock			
Producto	**Descripción**	**Precio**	**Actual**	**Mínimo**	**Comprar**	**Proveedor**
108	IMP. MULTIF. TX410	699,80	2	6	4	Moura
230	MON. 19 PULG. LCD	925,50	12	10	0	EPSON
110	W1943C	769	3	6	3	Asoc. LG
…	IMP. MULTIF. C5580-	…	…	…	…	HP Center
	CH.TIN.					…
	…					

- El Gerente de la empresa seguramente necesitará un informe con mayor complejidad de proceso, incluyendo datos estadísticos resultantes de cálculos resumiendo una situación y hasta un gráfico en pantalla, pero en definitiva obtenidos de los datos almacenados en los mismos archivos pero logrando una vista distinta.

Modelos de Datos

Los datos, en la base de datos, serán almacenados según la estructura elegida. Así como tenemos lenguajes de programación estructurados, orientados a aspectos, orientados a objetos, etc., donde cada uno brinda conceptos y ventajas en determinados tipos de desarrollos, existen lo que se llaman modelos de datos para almacenar las bases de datos, algunos que históricamente dieron las bases para el desarrollo de los demás, otros que están actualmente en uso y los que están en proceso de investigación.

Un modelo de datos brinda distintos conceptos, y permite definir reglas y estructuras para almacenar los datos, para después poder manipularlos.

Los modelos de datos se clasifican de diversas formas, pero hay coincidencia entre los autores en clasificarlos según los conceptos que son la base de cada uno. Por ejemplo, tenemos los modelos de datos basados en objetos y los basados en registros, como también el modelo físico de datos que apunta al almacenamiento de los datos en un nivel bajo de abstracción, relacionado al formato y camino de acceso a los datos.

En los modelos de datos basados en registros los datos se estructuran en un conjunto de campos que conforman un registro. Tal es el caso de un registro Estudiante conformado por campos como legajo, apellido, nombres, fecha y lugar de nacimiento.

En el caso de los modelos basados en objetos han aprovechado conceptos utilizados en el paradigma de programación orientada a objetos, como sucede en las áreas de ingeniería de software. Sólo que en las Bases de Datos Orientadas a Objetos se da la posibilidad de crear objetos para que tengan un almacenamiento que persista en el tiempo y puedan ser utilizados.

Modelos de Datos Orientado a Objetos

Este modelo nace con el objetivo de permitir el almacenamiento de datos para aplicaciones complejas que, como dicen Elisa Bertino y Lorenzo Martino[6], no se orientan a aplicaciones del área comercial y administrativa, sino a aplicaciones de las ingenierías, tales como CAD/CAM, CASE, CIM, sistemas multimediales y sistemas de gestión de imágenes, donde la estructura de los datos es compleja y las operaciones se pueden definir en función de la necesidad de las aplicaciones.

El Grupo de Gestión de Bases de Datos Orientadas a Objetos (ODMG), conformado por usuarios y productores de Sistemas de Gestión de Bases de Datos Orientadas a Objetos (OODBMS), es el encargado de estandarizar tanto el modelo de datos, como el lenguaje de Definición de Objetos (ODL) y el Lenguaje de Consulta de Objetos (OQL).

[6] Elisa Bertino y Lorenzo Martino: Sistemas de bases de datos orientadas a objetos.

Los datos se almacenan de manera persistente en objetos, los cuales son identificados unívocamente por un OID (IDentificador de Objeto) generado por el sistema.

Los objetos se definen con una estructura (OID, Constructor_Tipo, Estado), donde:

- OID: identificador único en el sistema, incluso cuando se elimina un objeto es conveniente que el OID no se reutilice.
- Constructor_Tipo: es la definición de la construcción del estado del objeto.
- Estado: se define a través del Constructor_Tipo.

Por ejemplo: si el Constructor_Tipo es de Tupla, significa que el Estado tendrá la estructura $(atr_1 : oid_1, atr_2 : oid_2, atr_3 : oid_3, ..., atr_n : oid_n)$. Donde atr_i es un nombre de atributo y oid_i es un OID.

Si el Constructor_Tipo es Átomo significa que el Estado será un valor.

Así el Constructor también puede ser un conjunto, una lista, bolsa y array, determinando que el estado se refiera a OIDs de objetos. Sólo el átomo se puede referir a un valor, aunque hasta este puede referirse a otro objeto mediante el OID del mismo.

Con estos constructores se producen anidamientos y se logra una estructura de alta complejidad, además de la posibilidad de incorporar conceptos propios de la programación orientada a objetos, como clase, herencia, polimorfismo, etc.

La definición en ODL de una clase Personas puede ser:

```
Class Personas
{
attribute string apellido,
attribute string nombres,
attribute enum PosiblesSexo {Masculino, Femenino} sexo,
attribute struct Domicilio
        {string calle, short numero, string ciudad,
attribute date fecha_nac;
short edad();
};
```

Para consultar la base de datos se utiliza el lenguaje OQL, propuesto por ODMG. Las sentencias de este lenguaje son similares a las de SQL y pueden ser insertadas en programas desarrollados con lenguajes de la Programación Orientada a Objetos, como ese el caso de C++ o Java. El resultado es un conjunto de objetos que pueden ser manipulados directamente en esos lenguajes y otros que sigan el paradigma.

Una consulta OQL puede ser:

```
SELECT p.apellido
FROM p in Personas
WHERE p.sexo=”Masculino”
```

En esta expresión aparece p como variable iterador que asume los valores de cada objeto y la consulta devuelve una bolsa string, porque es una colección de apellidos que pueden repetirse.
Como ventajas de este modelo pueden distinguirse la posibilidad de manipular objetos complejos con buen rendimiento, integrar la persistencia de datos a la programación orientada a objetos, menor costo y esfuerzo en el desarrollo de las aplicaciones y en el mantenimiento.
La aparición de este tipo de Sistemas de Gestión de Bases de Datos exigió a los RDBMS (Sistemas de Gestión de Bases de Datos Relacionales) a que incorporen objetos en sus estructuras y flexibilicen las posibilidades de almacenamiento, de ello surge el concepto actual de Sistemas de Gestión de Bases de Datos Objeto-Relacionales.

Actualmente se conocen prototipos y productos de gestión de bases de datos orientadas a objetos, entre estos están:

- Jasmine: tecnología orientada a objetos que permite tratar datos multimedia y complejas informaciones relacionadas. Se trata de un producto Java que puede ser utilizado en entornos intranet, Internet, cliente/servidor y a través de cualquier navegador.
- GemStone: fue introducido en 1987 y es el ODBMS comercial con más tiempo en el mercado. Soporta acceso concurrente de múltiples lenguajes, incluyendo Smalltalk-80, Smalltalk/V, C++ y C.
- Versant Object Database (V/OD): ofrece características importantes a los desarrolladores C + +, Java o. NET, el apoyo a la concurrencia masiva y grandes conjuntos de datos.

Este es un tema tan interesante como extenso, por ello dejaremos el tratamiento de este modelo para otro capítulo y empezaremos a dar las bases de conceptos para llegar al Modelo Relacional.

Modelos de Datos Basados en Registros

Estos modelos de datos tienen sus inicios con los Modelos de Red y Jerárquicos, hoy obsoletos desde el punto de vista tecnológicos.

Ambos modelos, Red y Jerárquicos, compartían el concepto de registro de datos y los enlaces entre ellos, pero difieren en sus restricciones y representación.

Características del Modelo de Datos en Red

Los datos se representaban como colección de registros y las relaciones entre ellos mediante enlaces, que se implementaban como campos que almacenan punteros.

Cada registro es una colección de campos y cada campo almacena un dato.

Gráficamente se los representaba con un diagrama, donde la estructura de datos estaba formada por cajas y líneas, las cajas son los registros y las líneas indicaban sus relaciones.

El siguiente diagrama representa a 3 registros de Clientes y 4 registros de cuentas bancarias.

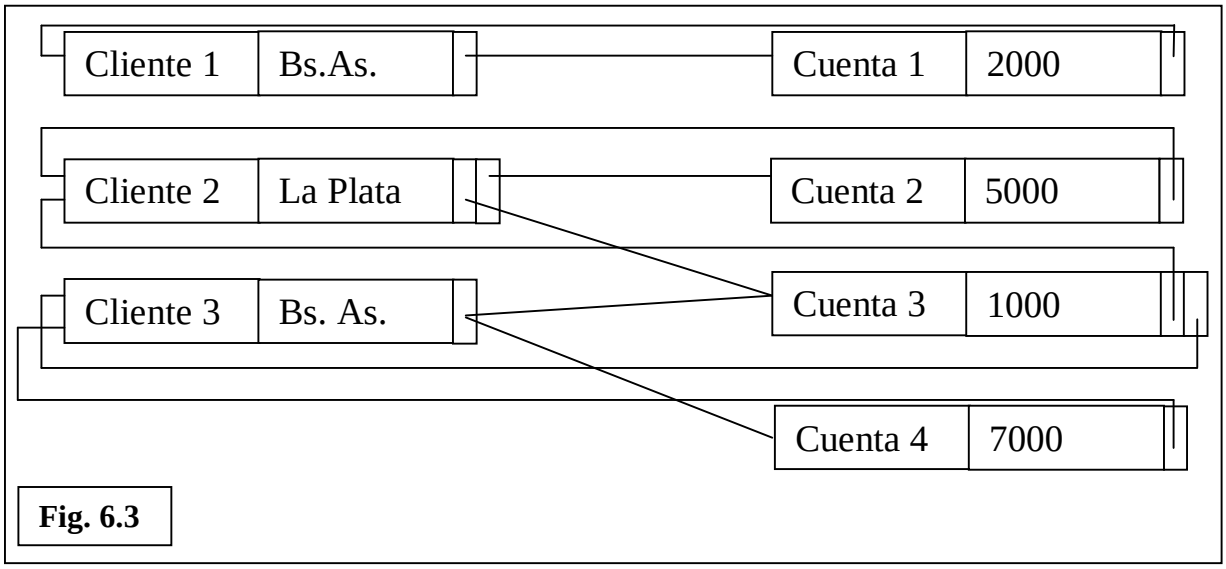

Fig. 6.3

Donde:

- Cliente 1 posee una sola cuenta con un saldo de $2000.- Observe que este registro tiene un campo para un puntero del Cliente a la Cuenta 1 y en el registro Cuenta 1 se visualiza un campo con un puntero hacia el Cliente 1.
- Cliente 2 tiene una Cuenta 2 a su nombre y comparte la Cuenta 3 con el Cliente 3. En este caso en Cliente 2 hay 2 campos, desde donde parten 2 punteros.
- Cliente 3 comparte la Cuenta 3 con el Cliente 2 y además posee su propia cuenta, que es la Cuenta 4.

Veamos que en el registro de la Cuenta 3 hay 2 campos, porque tiene un puntero hacia el Cliente 2 y otro hacia el Cliente 3.

Los campos que almacenaban punteros hacían que los registros fueran de longitud variable, porque en algunos casos se necesitaba sólo uno y en otros más.

La flexibilidad del modelo hizo que la estructura de datos fuera compleja, ya que no había restricciones, donde se puede observar cierta estructura padre-hijo.

Alguna implementación del modelo especificó una única restricción y fue que no podía insertarse un hijo si no había un padre con quien relacionarlo, como también definieron distintos enlaces para evitar la complejidad, como por ejemplo: estructura en anillo, anillo con puntero al padre, agrupados, etc.

Las acciones de manipulación de datos en la programación implicaba órdenes que significaban movimientos físicos, como por ejemplo:

- Find: localiza un registro según alguna condición y el puntero actual en el archivo queda apuntando a ese registro.
- Get: copia registro señalado por el puntero actual al área de trabajo-buffer de memoria.
- Store: crea un registro nuevo, en un archivo, con valores de datos ubicados en memoria que maneja el usuario.
- Modify: modifica contenido. Implica encontrarlo, subirlo a memoria y recién modificarlo.
- Erase: elimina registro, con previa búsqueda del mismo.
- Connect: conecta un registro con un conjunto de registros de un archivo.
- Disconnect: desconecta un registro de un conjunto archivo.
- Reconnect: mueve un registro de un conjunto a otro. Por ejemplo cambiar una cuenta de una sucursal a otra.

Al crear los conjuntos archivos se debe indicar que hacer ante las siguientes acciones:

- Inserción: si la inserción es manual debe existir en el programa una instrucción connect para enlazar el nuevo registro y si no puede ser automática, donde al ejecutar store el registro se almacena en el conjunto y es enlazado.
- Eliminación: depende si se definió como:
  - opcional, donde pueden existir registros sin conexión al conjunto (elimina padre y desconecta todos los hijos),
  - fijo, donde no puede haber registros miembros sin conexión al grupo y además no pueden reconectarse con otra ocurrencia de un conjunto (elimina padre y elimina todos los hijos),
  - obligatorio, tampoco puede haber registros sin conexión al grupo, pero si pueden moverse de un conjunto a otro (no deja eliminar el registro)
- Ordenación: en el conjunto archivo debe definirse que debe hacerse al añadir un nuevo registro. Las opciones son que al insertarlo pase a alguna de estas posiciones en el conjunto:
  - primero del conjunto,
  - último,
  - siguiente al registro actual del conjunto,
  - anterior al registro actual,
  - arbitrario, según determinación del sistema de gestión,
  - ordenado según algún criterio y el sistema de gestión debe mantener el orden.

Características del Modelo de Datos Jerárquico

Este modelo puede considerarse una implementación particular del Modelo en Red, donde la estructura física también se basa en registros y punteros, pero con forma de árbol invertido y restricciones en las relaciones.

Las relaciones se representan mediante enlaces, implementados como campos que almacenan punteros.

Los grafos son organizados como árboles con un nodo raíz.

Un registro es una colección de campos y cada uno almacena un dato, donde a la colección de registros se la denomina Tipo de registro que son los nodos en el árbol.

Las relaciones que se mantienen entre Tipos de registros son del tipo 1:N, donde se distinguen un registro Padre y N registros Hijos.

En el Diagrama Jerárquico se muestra la estructura de árbol, con cajas que representan los Tipos de registros y líneas que representan las relaciones Padre-Hijo.

Los niveles del árbol no tienen limitación y el Nodo Raíz está en el nivel 0 (cero).

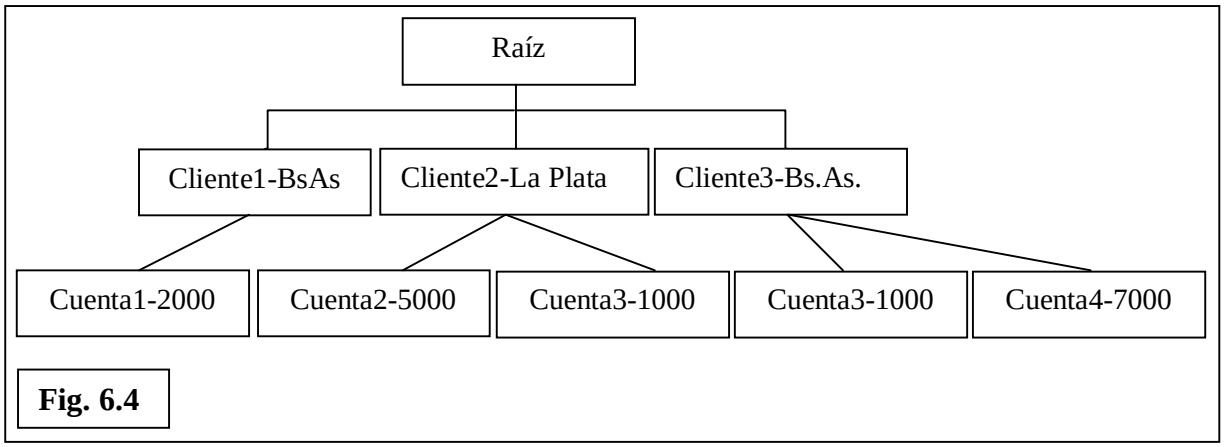

Fig. 6.4

Algunas restricciones del modelo son:

- El Nodo Raíz no participa como hijo en ninguna relación y los Tipos de Registros son siempre hijos de un solo Tipo de Registros,
- Un nodo padre puede tener un número ilimitado de hijos pero un nodo hijo puede, y debe, tener sólo un nodo padre.
- Si un Tipo de Registro Padre posee más de un Tipo de Registro Hijo, los Tipos de Registros Hijo estarán ordenados de izquierda a derecha.
- Se puede eliminar un registro hijo y el registro padre hasta puede quedar con 0 (cero) hijos, pero si se desea eliminar el registro padre no puede quedar un registro hijo sin padre, por lo cual se elimina el padre y todos los hijos asociados.

Para que un registro hijo tenga más de un registro padre debe duplicarse, con la consiguiente redundancia de datos. Algunas implementaciones evitan esto mediante uso de registros virtuales con punteros.

Ese tipo de situaciones muestran una desventaja del modelo jerárquico y que es la poca flexibilidad que posee para representar estructuras que en la vida real hacen falta. Que dos o

más clientes de un banco compartan una cuenta es muy común, como también en una estructura de productos existen productos que son piezas de otro producto (estructura recursiva de datos).

Algunas acciones que podemos mencionar en la manipulación de datos de este modelo:

- Get: localiza y copia un registro al área de trabajo, las condiciones de búsqueda se agregan con la cláusula where. Con un ciclo se podía recorrer toda una rama del árbol, leyendo todos los registros al recorrerla, sin importar la altura del árbol.
- Insert: añade un registro al árbol. Primero se localiza a su registro padre y luego se emite la orden de inserción.
- Replace: modifica uno o más campos de un registro, que primeramente debe ser ubicado como el registro actual de la base de datos.
- Delete: elimina un registro de la base de datos, que esté como registro actual.

Wikipedia dice: "La primera implementación de este metamodelo fue IMS (Information Management System). Se trata de un diseño de IBM y otros colaboradores en 1966 para el Programa Apollo de la NASA. IMS aún se encuentra activo."

Referencias Bibliográficas

[ANSI, 2010] www.ansi.org. Visitado en Enero 2010.

[Bertino-Martino, 1995] Sistemas de bases de datos orientadas a objetos, Conceptos y arquitecturas.

[De Miguel-Piattini, 1999] Fundamentos y modelos de Bases de Datos, Segunda Edición.

[Elmasri-Navathe, 2002] Fundamentos de Sistemas de Bases de Datos, Tercera Edición.

[GemStone, 2009] http://www.gemstone.com/. Visitado en Diciembre 2009.

[Jasmine, 2009] http://www.idg.es/dealerworld/CA-y-Fujitsu-presentan-Jasmine Visitado en Diciembre 2009.

[KORTH/SILBERSCHATZ/SUDARSHAN, - Fundamentos de Bases de Datos – Quinta Edición

[Marqués, 2002] Bases de datos orientadas a objetos.
http://www3.uji.es/~mmarques/e16/teoria/cap2.pdf. Visitado en Febrero 2010.

[Stocker-Gray-Atkinson, 1984] Databases, role and structure: an advanced course

[Wikipedia, 2009] http://en.wikipedia.org/wiki/Object_database. Visitado en diciembre 2009.