**Modelo Jerárquico**

Los registros tradicionales en un archivo convencional era contenido fijo y con una estructura esencialmente lineal. La ubicación de los distintos campos dentro del registro no está determinada por razones lógicas sino por todas las posiciones son equivalentes y a lo sumo se busca que ciertos campos estén contiguos por prolijidad. un programa utiliza la información en forma de registros, que físicamente son transferidos desde y hacia los periféricos.  
Si analizamos la forma en que se los utiliza, notamos que no es necesario disponer de todos los datos del registro al mismo tiempo, lo que implica que el registro de un archivo es una unidad demasiado grande para manejarla eficazmente y aparece la idea de particionar la información y agruparla según vaya a utilizarla.

Surgen los registros de base de datos en función del contenido de la agrupación lógica, una podrá ser más relevante que otra, lo que implica un orden jerárquico, y cuanto más significativo sea un dato tanto mayor será el nivel del grupo en que va a aparecer.

El modelo de datos jerárquico representa organizaciones jerárquicas y utiliza un principio de estructuración de datos: la relación padre-hijo. Este vínculo también llamado VPH es una relación de 1: N entre dos tipos de registros. Una ocurrencia del tipo VPH consiste en un registro del tipo reg. Padre y varios registros del tipo reg. hijo.

**Propiedades de los esquemas jerárquicos**

Un grupo del más alto nivel con sus dependientes en jerarquía constituyen un registro jerárquico o de árbol.  
El más alto nivel de jerarquía se llama “Raíz” y se conoce como el nivel 0.  
La raíz no participa como registro hijo en ningún tipo de VPH.  
Todo tipo de registro, con excepción de la raíz, participa como tipo de registro hijo en uno y solo un tipo de VPH.  
Un tipo de registro puede participar como tipo de registro padre en cualquier cantidad de tipos VPH.  
Un tipo de registro que no participa como tipo de registro padre en ningún tipo de VPH se denomina hoja( es el nivel más bajo del esquema jerárquico).

Empleado:

Departamento:

Entonces, como grafico genérico, encontramos lo siguiente.

**Arboles de concurrencias jerárquicas**

En correspondencia con el esquema jerárquico, hay muchas ocurrencias en la base de datos. Cada ocurrencia, es una estructura de árbol cuya raíz es un solo registro de su tipo.

Ejemplo:

Estos árboles se utilizan tanto para las descripciones físicas como para las lógicas.

En las primeras, se los emplea para describir el conjunto de punteros y relaciones entre entradas en los índices.  
En las lógicas sirven para describir relaciones que existen entre tipos de segmentos o tipos de registros.  
En los diagramas de base de datos, cada nudo es un tipo de registro(segmento) diferente. Las relaciones lógicas se producen a través del uso de punteros para evitar la duplicación del mismo segmento denominado “Hijos lógicos”.

En el grafico que se presenta a continuación, el segmento “empleados” y el segmento “cursos” aparece en dos árboles separados (dos bases de datos FISICAS distintas).

En cambio, si los árboles se construyeran como lo marcan las líneas punteadas se evita la redundancia remplazando la repetición de los segmentos indicados por hijos lógicos(punteros) que se dan la dirección de los segmentos empleados y cursos.  
Este último, incluirá datos de intersección (notas) puesto que no pertenecen ni al árbol CATEGO ni al árbol EMPLE, sino a la relación o vínculo entre ambos.

**Base de datos física: CATEGO**

**A**

CATEGORIA

**B**

**D**

**C**

CAPACITACION

CURSOS

SALARIO

EMPLEADOS

**X**

HIJO LOGICO EMPLEADOS

**Base de datos física: EMPLE**

**E**

EMPLEADOS

**H**

**G**

**F**

CREDITOS

EXPERIENCIA

EDUCACION

CURSOS Y NOTAS

**Y**

H.LOG | NOTAS  
CURSOS

El hecho de la existencia de hijos lógicos permite tener distintas visiones oses lógicas a través de su nexo lógico, provocando en muchos casos la invasión de jerarquías en determinados segmentos.

Ejemplo: Base lógica CATEGO Base lógica EMPLE

**E**

**GC**

**H**

**A**

**F**

**B**

**D**

**CC**

**Y**

**D**

**E**

**X**

**A**

**F**

0

**C**

**BC**

**H**

**G**

**Descripción de datos:**

El administrador de datos debe estar en condiciones de hacer la descripción lógica global de los datos(esquema), como así también las vistas parciales para cada programador de aplicaciones(subesquema). Algunas veces tendrá que escribir tipos de relaciones o características de datos que el programador de aplicaciones no describe. Para esta finalidad es necesario un lenguaje de descripción de datos (DDL). Este lenguaje podrá ser una ampliación de un lenguaje de programación, o un recurso del sistema administración.

**Tipos de descripción o bloques de control:**

La explotación de sistemas de base de datos se concreta con el uso de dos tipos de bloque de control llamados DESCRIPCION DE LA BASE DE DATOS-DB- y BLOQUE DE ESPECIFICACION DE PROGRAMA -PSB-.  
a su vez el bloque PSB está formado por uno o más BLOQUES DE CUMINICACION DE PROGRAMA -PCB-

**DB:** contiene la descripción general de la base de datos incluyendo características físicas, tales como métodos de acceso, diseño de segmentos, detalles de los campos, longitud de bloques, aspectos de seguridad y control de accesos etc.

**PSB:** muestra la “visión del programador” o estructura lógica de datos a procesar, respondiendo a un bloque DBD e incluyendo todos o una parte de los segmentos de varios DBD, agregando la especificación del tipo de procesamiento autorizado en cada caso: solo lectura, borrado, actualización, etc.  
Un bloque PSB es un conjunto de 1 o más bloques de comunicación del programa -PCBs-

**PSB:** es la descripción parcial de la base de datos, aquella a la que es sensible el programa. Además, permite especificar para cada segmento que tipo de preparaciones puede realizar el programa sobre él.

**SENSIBILIDAD:** es el conjunto de segmentos que utiliza el programa.

Vamos a ejemplificar el siguiente caso:

El programa procesará 3 bases de datos, por lo tanto, el diagrama será así:

**DB3**

**DB2**

**DB1**

**DB3**

**DB1**

**DB2**

**PCB1**

**PCB2**

**PCB3**

**PSB**

**Programa**

**Lenguaje de manipuleo de datos:**

El lenguaje de manipuleo de datos (DML): sirve como una interfase entre un programa de aplicación y una base de datos.  
Cuando en un sistema tradicional un programa requiere una operación de entrada/salida sobre un archivo, dicho programa le concederá el control al supervisor, quien manejará dicha situación.  
Cuando se trabaja con DML, el programa de aplicación se conecta con él y llegado el momento, es este quien emitirá el llamado al supervisor.  
En general, las órdenes del lenguaje deben estar incorporadas en un lenguaje de programación de aplicación general, llamado anfitrión.

En un lenguaje de registro por registro, una operación de obtención de base de datos coloca los registros obtenidos en variables de programa.  
Después de cada orden de base de datos, el ultimo registro al que tuvo acceso la orden es el registro actual de la base de datos.

**Supervisor**

**LMD**

**Programa de aplicacion**

**Panorama del sistema de base de datos jerárquicas ims**

Investigar sobre el sistema jerárquico information management system (IMS), sistema dominante para el manejo de sistema de contabilidad y de inventarios a gran escala.  
Arquitectura básica.  
Organización lógica de los datos.

**Modelo Plex o Red**

**Definición:**

Como segundo tipo de organización de base de datos aparece la base de datos Plex o Red, que, junto con el modelo jerárquico, fue importante para implementar un gran número de sistemas comerciales.

Si en una relación entre datos, un hijo tiene más de un padre, ya no se cumple las características de una estructura jerárquica. Por lo tanto, en ese tipo de base la diferencia fundamental es que cada hijo puede tener más de un padre.  
Cualquier componente puede vincularse con otro.  
Como en el caso de árbol, la estructura plex puede ser descripta en términos de padre e hijos.

**4**

**1**

**2**

**3**

Hay dos estructuras de datos básicas que se manejan en el modo de red: registros y conjuntos.

**Registros:** los datos se almacenan en registros; cada registro consiste en un grupo de valores de datos relacionados entre sí. Con el modelo de red es posible definir elementos de información compleja.  
estructuras Plex simples y complejas  
Lamamos estructura plex simple a aquella que no hay correspondencia compleja en ambos sentidos.  
Ejemplo:

**A**

**B**

**PAIS**

La correspondencia entre padre (PAIS) y el hijo (PROVINCIA) es simple ya que un país le corresponden varias provincias, pero una provincia se corresponde solo con un país.

**PROVINCIA**

Llamamos estructura plex compleja a aquella que se corresponden en ambos sentidos.

**A**

**B**

Estas estructuras requieren métodos más elaborados para su representación física y, por ejemplo, el lenguaje para la descripción de datos propuestos por CODASYL no sirve para describirlas. En estos casos, se recurre a la transformar dichas estructuras en simple.  
Ej.:

**Articulo**

Esto indica:  
1 articulo 🡪 puede figurar en varias ordenes de compra.  
1 orden de compra🡪 puede contener varios artículos.

**Orden de compra**

Para transformar esta estructura compleja en simple, separamos por articulo y por orden de compra:

**Articulo C**

**Articulo B**

**Articulo A**

**Orden de compra 1**

**Orden de compra 4**

**Orden de compra 3**

**Orden de compra 2**

Un artículo aparece en varias órdenes de compra y a su vez una orden de compra puede contener diferentes artículos.

**Organización física**

PUNTEROS HIJOS MULTIPLES

Cualquier registro, excepto los de nivel inferior, puede tener muchos hijos; por lo tanto, necesitamos incluir en los registros, listas de punteros de longitud variable, a veces muy extensas.  
Estas listas de punteros resultan difíciles de manejar cuando las inserciones y eliminaciones son frecuentes. Es por eso que se evita en lo posible el uso de este método.

PUNTEROS HIJOS Y MELLIZOS

Se necesitan solo dos punteros por registro y uno solo para los registros de ultimo nivel. El problema está en que se forman largas cadenas de punteros cuando hay muchos mellizos

**El conjunto: SET**

Es un anillo y tiene por tanto un registro principal llamado registro propietario u owner y registros miembros o members.

El SET puede definir relaciones lógicas entre los registros.  
Un conjunto puede ser vacío porque contiene solo al registro propietario y carece de miembros, sin embargo, ningún SET puede prescindir del registro propietario.

Existe una relación de dependencia entre el registro dueño y los registros miembros. Los registros miembros pueden estar conectados siguiendo algún criterio de ordenamiento previamente definido.

Un set puede representarse esquemáticamente por un arco, o una flecha entre dos tipos de registros.

**Propietario**

Set X

**Miembro**

Hay 7 reglas de base para el establecimiento de las relaciones entre los diferentes tipos de registros de la base de datos por medio de los sets.

1. Todo registro puede ser miembro de varios SETS diferentes:

Set B

Set A

**Miembro**

**Propietario**

**Propietario**

1. Todo registro puede ser propietario de varios SETS diferentes:

Set C

Set E

Set D

**Miembro**

**Miembro**

**Miembro**

**Propietario**

1. Todo registro puede ser a la ves:
   * Miembro de uno o varios sets
   * Propietario de uno o varios set.

Set F

Set G

**Miembro y propietario**

**Propietario**

**Miembro**

1. Un SET solo puede tener un tipo de registro como propietario, pero puede tener varios tipos de registros diferentes como miembro:

Set H

**Propietario**

**Miembro**

**Miembro**

1. Dos tipos de registros pueden conectarse entre ellos por un numero cualquiera de sets diferentes.

Propietario

Set J

Set K

Miembro

1. Puede existir un tipo de registro en la base de datos sin participar en ningún SET:
2. Un registro puede declararse miembro opcional. La participación de las ocurrencias del miembro se establece entonces o se suprime dinámicamente en el seno del programa utilizador.  
     
     
     
     
     
     
     
     
   La implementación interna del SET se realiza por un encadenamiento entre la ocurrencia del propietario y las diferentes ocurrencias de los miembros.

Set L

Miembro

Propietario

Las ocurrencias de los miembros se relacionan automáticamente unas con otras por apuntadores “hacia el que sigue”. El usuario puede, si quiere, solicitar que estas ocurrencias estén igualmente relacionadas por apuntadores “hacia el que precede” de tal forma que la cadena pueda igualmente ser recorrida a la inversa.

El usuario puede especificar que cada ocurrencia del miembro apunte directamente hacia el propietario, de tal forma que no sea necesario continuar con la cadena de las ocurrencias de los miembros antes de llegar al propietario.

Las opciones de relaciones son en consecuencia:

Apuntador hacia el que sigue.  
Apuntador hacia el que sigue y el que precede.  
Apuntador hacia el que sigue y el propietario.  
Apuntador hacia el que sigue, el que precede y el propietario.

A nivel del ordenamiento del SET

**S**

Propietario

**P**

**Pr**

Miembro

**P**

**S**

**S**

**P**

**Pr**

Miembro

Pueden utilizarse varias opciones para ordenar las diferentes ocurrencias de un mimbro en el seno de un mismo SET.  
Hay que notar que este orden lógico de las ocurrencias de un miembro en un SET es totalmente independiente de su localización física en la base de datos.   
La posición que ocupara una nueva ocurrencia entre las otras puede definirse como la que debe ser la primera de la cadena de ocurrencias, sea la última, sea aún la posición que precede o sigue inmediatamente tal ocurrencia determinada, indicada por el programa de aplicación, sea finalmente la posición que resulta de un criterio de elección determinado.

De igual manera, se puede exigir también que las ocurrencias de un miembro de un SET se ordenen en base a un atributo del miembro en forma ascendente o descendente.

Ejemplo de estructuración de datos por medio de SETs.

Set A

Set B

Clientes

Pedidos

Productos

Facturas

Pagos

Set D

Set C

Cantidades

**Definición de datos**

Después de diseñar un esquema de base de datos de red, debemos declarar a sistema de gestión de base de datos todos los tipos de registros, tipos de conjuntos, definiciones de elementos de información y restricciones del esquema. Para ello usaremos el DDL de red.

**Lenguaje de manipulación de datos en red**

El DML (Lenguaje de manipulación de datos) asociado al modelo de red consiste en órdenes de registro por registro incorporadas en un lenguaje de aplicación general denominado lenguaje anfitrión. El sistema de base de datos y el lenguaje de programación anfitrión son dos sistemas de software distintos que se enlazan mediante una interfaz común y se comunican solamente a través de esa interfaz. Como las ordenes del DML son registro por registro, es necesario identificar registros específicos de la base de datos como registros actuales. El propio sistema de gestión de base de datos lleva el control de varios registros y ocurrencias de conjunto por medio de un mecanismo denominado indicadores de actualidad. Se utilizan varios de ellos:

**Actual de tipo de registro:** por cada tipo de registros, se sigue la pista al último registro de ese tipo al que se tuvo acceso. Si todavía no se ha leído ninguno de ese tipo, el registro actual no está definido.

**Actual del tipo de conjuntos:** por cada tipo de conjuntos en el esquema, se sigue la pista de la ultima ocurrencia de conjunto de ese tipo de conjuntos a la que se tuvo acceso. La ocurrencia se especifica con un solo registro de ese conjunto, que es el propietario o uno de los registros miembros. Si todavía no se ha leído ninguno de ese tipo, el actual conjunto no está definido.

Actual de unidad de ejecución (CRU – current of run unit): es un programa de acceso a la base de datos que el computador esta ejecutando. Por cada unidad de ejecución, el CRU sigue la pista al ultimo registro al que tuvo acceso el programa.

Cada vez que el programa ejecuta una orden de DML, el sistema de gestión de base de datos actualiza los indicadores de actualidad para los tipos de registros y de conjuntos afectados por esa orden.

Hay varios indicadores de estado que devuelven una señal de éxito o fracaso después de ejecutarse cada orden de DML. El programa puede revisar los valores de estos indicadores de estado y emprender acciones apropiadas: ya sea continuar o transferir el control a una rutina de manejo de errores.

**Un sistema de base de datos de red: IDMS**

Investigar sobre el sistema que se basa en el modelo de red: el infomation batabase management system (IDMS: sistema de gestión de base de datos integrados).  
Arquitectura básica.  
Organización lógica de los datos.

**Generalidades de la administración de las bases de datos**

**Estructura del sistema**

La estructura interna de un sistema de gestión de base de datos (SGBD) es bastante compleja. Aludiendo a uno general (ninguno en particular), se puede considerar constituido por tres componentes principales:

* Servicios del sistema: el cual hace posible la operación del sistema, la comunicación con el operador, la conexión de usuarios, y funciones similares.
* Servicios de bloqueo: el cual suministra los controles necesarios para manejar el acceso a la información.
* Servicios de base de datos: el cual hace posible la definición, recuperación y actualización de los datos de los usuarios y del sistema.

Analizaremos en detalle el componente de servicios de base de datos.

**Servicios de base de datos**

Este componente, a su vez, se subdivide en cinco subcomponentes:

* Pre-compilador: es un preprocesador de los programas de aplicación.
* Ligador: compila uno o mas módulos de solicitudes a la base de datos.
* Supervisor durante la ejecución: vigila los programas mientras se ejecutan.
* Manejador de datos almacenados: administra la base de datos encargándose de recuperar y actualizar registros.
* Manejador de buffers: se encarga de transferir físicamente los datos entre el disco y la memoria principal.

**Arquitectura de un SGBD**

Hay tres características para destacar inherentes al enfoque de las bases de datos, que son.

1. La independencia de datos con respecto a los programas.
2. El manejo de múltiples vistas de usuario.
3. El empleo de un cátalo para almacenar la descripción de la base de datos.

Los sistemas pequeños quizá no añadan todos estos aspectos de una arquitectura. Sin embargo, la arquitectura en cuestión se adecua a la mayor parte de los sistemas.  
La arquitectura propuesta para los sistemas de base de datos se llama arquitectura de tres esquemas o ANSI/SPARC, ya que ese es el nombre del grupo de estudio que la propuso.

Es indispensable para entender en su totalidad, la estructura y las posibilidades de los sistemas modernos de base de datos.  
El objetivo consiste en formar una separación entre las aplicaciones del usuario y la base de datos física.

**La arquitectura se divide en tres niveles:** interno, conceptual y externo.

**Externo:**  es el del usuario individual. Para él será o bien un lenguaje de consulta, o algún lenguaje de aplicación especial. Es el que se ocupa de la forma como los usuarios individuales perciben los datos.  
El usuario dispondrá de un lenguaje:

* En caso del programador de aplicaciones, dicho lenguaje podrá ser de programación convencional (COBOL o PL/I), o bien un lenguaje propio especifico par el sistema en cuestión.  
  Para el usuario final será un lenguaje de consulta manejado mediante formas o menús, adaptado a los requerimientos de ese usuario.  
  este lenguaje se ocupa de manera especifica a los objetos y operaciones de la base de datos (SQL).

Este tipo de lenguaje será una combinación de por lo menos dos lenguajes subordinados: un lenguaje de definición de datos (DDL, data definition lenguage), con el cual es posible definir y declarar los objetos de la base de datos, y un lenguaje de manipulación de datos (DML, data manipulation lenguage) con el que es posible manejar o procesar dichos objetos.

En general, una viste externa se compone de varias ocurrencias de varios tipos de registros externos. Por ejemplo, una operación de consulta en DML extraerá una ocurrencia de un registro externo (registro lógico).

**Conceptual:**  
Es una representación de toda la información contenida en la base de datos.   
es la vista comunitaria de los usuarios, o sea un nivel de mediación.  
Oculta los detalles de las estructuras físicas de almacenamiento y se concentra en describir entidades, tipos de datos, vínculos, operaciones de los usuarios y restricciones.

**Interno:**es el que se ocupa de la forma como se almacenan físicamente los datos. Es una representación de bajo nivel. Emplea un modelo físico de datos y describe todos los detalles para su almacenamiento, y los caminos de acceso para la base de datos.   
La siguiente ilustración muestra la relación entre los tres niveles.

Usuarios finales

Vista Externa 1

Vista Externa N

Nivel conceptual

Nivel Interno

Base de datos almacenada

Sistema de administración de bases de datos

El DBMS es el conjunto de programas que maneja todo el acceso a la base de datos.  
Ejemplo:   
1) un usuario solicita acceso, empleado algún sublenguaje de datos determinado (SQL).