

# MÓDULO

# BASES DE DATOS

---

EDGAR EUSEBIO LÓPEZ MURILLO  
JOHN JAIRO MONSALVE RESTREPO  
MARGARITA MARÍA HINCAPIÉ VÉLEZ

---



POLITÉCNICO COLOMBIANO  
JAIME ISAZA CADAVID



Alcaldía de Medellín

## PLAN DE TRABAJO

### INTRODUCCIÓN

El programa de Bases de Datos a desarrollarse, comprende cuatro unidades de trabajo teórico prácticas. Con ellas, se pretende que el alumno posea una visión clara, sencilla, general y completa sobre Bases de Datos.

El presente módulo de Bases de Datos, plantea los siguientes interrogantes generales, para ser resueltos en el desarrollo de esta asignatura.

¿Qué son Bases de Datos?

¿Cuáles son los objetivos de los sistemas de bases de datos?

¿Qué es abstracción de la Información?

¿Qué es un modelo de datos?

¿Qué tipos de modelos de datos existen?

¿Qué es un lenguaje para manipulación de datos?

¿Qué personas intervienen en la manipulación de una base de datos?

¿Cuáles son las técnicas para modelar una base de datos?

Los anteriores interrogantes se resolverán en el desarrollo de este módulo de la manera más sencilla posible y didácticamente agrupados en unidades. En cada unidad se colocarán actividades a desarrollar en forma práctica, permitiendo que el estudiante interiorice el conocimiento por sus propias vivencias, de igual manera, que surjan nuevos interrogantes y tenga capacidad para resolverlos con propiedad.

### OBJETIVO GENERAL

Introducir a los estudiantes, en el conocimiento y manejo de las bases de datos.

## OBJETIVOS ESPECÍFICOS

- Adquirir los conocimientos que permitan reconocer los conceptos básicos en el manejo de las bases de datos.
- Adquirir los conocimientos que permitan asimilar los conceptos fundamentales de un modelo Entidad - Relación.
- Adquirir los conocimientos que permitan identificar la estructura de un modelo relacional, así como los lenguajes necesarios para su manipulación.
- Familiarizarse e identificar otros modelos de Bases de Datos

## INTEGRACIÓN DEL MÓDULO POR UNIDADES

- **Unidad 1: Introducción a los conceptos de bases de datos**

Introducción  
Conceptos básicos  
Objetivos de los sistemas de bases de datos  
Abstracción de la información  
Modelos de datos  
Otros conceptos asociados a una base de datos

- **Unidad 2: Modelo Entidad Relación (E-R)**

Modelo entidad relación  
Entidades y conjunto de entidades  
Relaciones y conjunto de relaciones.  
Limitantes de mapeo - Tipos de relaciones  
Llaves primarias  
Diagrama Entidad-Relación  
Reducción de diagramas E-R a tablas  
Generalización y especialización  
Agregación

- **Unidad 3: Modelo relacional**

Modelo Relacional  
Conceptos básicos  
Estructura de las bases de datos relacionales  
Lenguajes de consulta  
Diseño de bases de datos relacionales  
Normalización

- **Unidad 4: Otros modelos de bases de datos**

Modelos de red  
Modelo de datos de red  
Conceptos básicos  
Diagramas de estructura de datos.  
Modelos de datos Jerárquicos  
Modelo de datos jerárquico  
Conceptos Básicos.  
Diagramas de estructura de árbol  
Modelos de datos orientados a objetos  
Base de datos orientados a objetos.  
Jerarquía de clases.  
Herencia  
Consultas orientadas a objetos:  
Complejidad de Modificación

## UNIDAD 1

**INTRODUCCIÓN A LOS CONCEPTOS DE BASES DE DATOS****INTRODUCCIÓN**

Hoy en día las empresas manejan una gran cantidad de datos. Cualquier empresa que se precie debe tener almacenados todos estos datos en una base de datos para poder manejarlos mediante una aplicación profesional; sin esta funcionalidad resultaría imposible tratar y manejar en su totalidad los datos que lleva a cabo la empresa y se perdería un tiempo y un dinero muy valiosos.

Uno de los pasos cruciales en la construcción de una aplicación que maneje una base de datos, es sin duda, el diseño de la base de datos.

Si las tablas no son definidas apropiadamente, podemos tener muchos dolores de cabeza al momento de ejecutar consultas a la base de datos para tratar de obtener algún tipo de información.

No importa si nuestra base de datos tiene sólo 20 registros, o algunos cuantos miles, es importante asegurarnos que nuestra base de datos está correctamente diseñada para que tenga eficiencia y que se pueda seguir utilizando por largo del tiempo.

En esta unidad, se mencionarán algunos principios básicos del diseño de base de datos y se tratarán algunas reglas que se deben seguir cuando se crean bases de datos.

Dependiendo de los requerimientos de la base de datos, el diseño puede ser algo complejo, pero con algunas reglas simples que tengamos en la cabeza será mucho más fácil crear una base de datos perfecta para nuestro siguiente proyecto.

**JUSTIFICACIÓN**

Desde tiempos remotos, los datos han sido registrados por el hombre en algún tipo de soporte (piedra, papel, madera, etc.) a fin de que quedara constancia de un fenómeno o idea. Los datos han de ser interpretados para que se conviertan en información útil, esta interpretación supone un fenómeno de agrupación y clasificación.

En la era actual y con el auge de los medios informáticos aparece el almacenamiento en

soporte electromagnético, ofreciendo mayores posibilidades de almacenaje, ocupando menos espacio y ahorrando un tiempo considerable en la búsqueda y tratamiento de los datos. Es en este momento donde surge el concepto de bases de datos y con ellas las diferentes metodologías de diseño y tratamiento.

El objetivo básico de toda base de datos es el almacenamiento de símbolos, números y letras carentes de un significado en sí, que con un tratamiento adecuado se convierten en información útil. Un ejemplo podría ser el siguiente dato: 19941224, con el tratamiento correcto podría convertirse en la siguiente información: "Fecha de nacimiento: 24 de diciembre de 1994".

Según van evolucionando los tiempos, las necesidades de almacenamiento de datos van creciendo y con ellas las necesidades de transformar los mismos datos en información de muy diversa naturaleza. Esta información es utilizada diariamente como herramientas de trabajo y como soporte para la toma de decisiones por un gran colectivo de profesionales que toman dicha información como base de su negocio. Por este motivo el trabajo del diseñador de bases de datos es cada vez más delicado, un error en el diseño o en la interpretación de datos puede dar lugar a información incorrecta y conducir al usuario a la toma de decisiones equivocadas. Se hace necesaria la creación de un sistema que ayude al diseñador a crear estructuras correctas y fiables, minimizando los tiempos de diseño y explotando todos los datos, nace así la metodología de diseño de bases de datos.

### OBJETIVO GENERAL

Proporcionar los conocimientos que permitan al estudiante reconocer los conceptos básicos en el manejo de las bases de datos.

### OBJETIVOS ESPECÍFICOS

- Reconocer los conceptos involucrados en el manejo de una base de datos.
- Identificar los objetivos de un sistema de base de datos.
- Identificar los conceptos de abstracción de la información
- Identificar los diferentes modelos de bases de datos

## CONTENIDO

1. Conceptos básicos
2. Objetivos de los sistemas de bases de datos
3. Abstracción de la información
4. Modelos de datos
5. Otros conceptos asociados a una base de datos

## CONCEPTOS BÁSICOS

Es necesario iniciar esta unidad con algunos conceptos básicos para el mejor entendimiento de la misma, por lo tanto empezaremos con las definiciones que involucran a las bases de datos.

**Dato:** Conjunto de caracteres con algún significado, pueden ser numéricos, alfabéticos, o alfanuméricos.

**Información:** Es un conjunto ordenado de datos los cuales son manejados según la necesidad del usuario. Para que un conjunto de datos pueda ser procesado eficientemente y pueda dar lugar a información, primero se debe guardar lógicamente en archivos.

Conceptos básicos de archivos computacionales.

**Campo:** Es la unidad más pequeña a la cual el usuario puede referirse en un programa. Desde el punto de vista del programador representa una característica de un individuo u objeto.

**Registro:** Colección de campos de iguales o de diferentes tipos.

**Archivo:** Colección de registros almacenados siguiendo una estructura homogénea.

**Base de datos:** Es una colección de archivos interrelacionados, son creados con un Sistema Manejador de Bases de Datos (DBMS). El contenido de una base de datos engloba a la información concerniente a una organización (almacenada en archivos), de tal manera que los datos estén disponibles para los usuarios. Una finalidad de la base de datos es eliminar la redundancia o al menos minimizarla. Los tres componentes principales de un sistema de base de datos son el hardware, el software DBMS y los datos a manejar, así como el personal encargado del manejo del sistema.

Sistema Manejador de Base de Datos. (DBMS): Un DBMS es una colección de numerosas rutinas de software interrelacionadas, cada una de las cuales es responsable de una tarea específica. El objetivo primordial de un sistema manejador base de datos es proporcionar un contorno que sea a la vez conveniente y eficiente para ser utilizado al extraer, almacenar y manipular información de la base de datos. Todas las peticiones de acceso a la base, se manejan centralizadamente por medio del DBMS, por lo que este paquete funciona como interfase entre los usuarios y la base de datos.

Esquema de base de datos: Es la estructura por la que está formada la base de datos, se especifica por medio de un conjunto de definiciones que se expresa mediante un lenguaje especial llamado lenguaje de definición de datos. (DDL)

Administrador de base de datos (DBA): Es la persona o equipo de personas profesionales responsables del control y manejo del sistema de base de datos, generalmente tiene(n) experiencia en DBMS, diseño de bases de datos, Sistemas operativos, comunicación de datos, hardware y programación.

Los sistemas de base de datos se diseñan para manejar grandes cantidades de información, la manipulación de los datos involucra tanto la definición de estructuras para el almacenamiento de la información como la provisión de mecanismos para la manipulación de la información, además un sistema de base de datos debe de tener implementados mecanismos de seguridad que garanticen la integridad de la información, a pesar de caídas del sistema o intentos de accesos no autorizados.

Un objetivo principal de un sistema de base es proporcionar a los usuarios finales una visión abstracta de los datos, esto se logra escondiendo ciertos detalles de como se almacenan y mantienen los datos.

## OBJETIVOS DE LOS SISTEMAS DE BASES DE DATOS

Los objetivos principales de un sistema de base de datos es disminuir los siguientes aspectos:

### 1. Redundancia e inconsistencia de datos

Puesto que los archivos que mantienen almacenada la información son creados por diferentes tipos de programas de aplicación existe la posibilidad de que si no se controla detalladamente el almacenamiento, se pueda originar un duplicado de información, es decir que la misma información esté más de una vez en un dispositivo de almacenamiento. Esto aumenta los costos de almacenamiento y acceso a los datos, además de que puede originar la inconsistencia de los datos - es decir diversas copias de un mismo dato no concuerdan entre si -, por ejemplo: que se actualice la dirección de un cliente en un archivo y que en otros archivos permanezca la anterior.



## **2. Dificultad para tener acceso a los datos**

Un sistema de base de datos debe contemplar un entorno de datos que le facilite al usuario el manejo de los mismos. Supóngase un banco, y que uno de los gerentes necesita averiguar los nombres de todos los clientes que viven dentro del código postal 78733 de la ciudad. El gerente pide al departamento de procesamiento de datos que genere la lista correspondiente. Puesto que esta situación no fue prevista en el diseño del sistema, no existe ninguna aplicación de consulta que permita este tipo de solicitud, esto ocasiona una deficiencia del sistema.

## **3. Aislamiento de los datos**

Puesto que los datos están repartidos en varios archivos y estos no pueden tener diferentes formatos, es difícil escribir nuevos programas de aplicación para obtener los datos apropiados.

## **4. Anomalías del acceso concurrente**

Para mejorar el funcionamiento global del sistema y obtener un tiempo de respuesta más rápido, muchos sistemas permiten que múltiples usuarios actualicen los datos simultáneamente. En un entorno así la interacción de actualizaciones concurrentes puede dar por resultado datos inconsistentes. Para prevenir esta posibilidad debe mantenerse alguna forma de supervisión en el sistema.

## **5. Problemas de seguridad**

La información de toda empresa es importante, aunque unos datos lo son más que otros, por tal motivo se debe considerar el control de acceso a los mismos, no todos los usuarios pueden visualizar alguna información, por tal motivo para que un sistema de base de datos sea confiable debe mantener un grado de seguridad que garantice la autenticación y protección de los datos. En un banco por ejemplo, el personal de nóminas sólo necesita ver la parte de la base de datos que tiene información acerca de los distintos empleados del banco y no a otro tipo de información.

## **6. Problemas de integridad**

Los valores de datos almacenados en la base de datos deben satisfacer cierto tipo de restricciones de consistencia. Estas restricciones se hacen cumplir en el sistema añadiendo códigos apropiados en los diversos programas de aplicación.

## ABSTRACCIÓN DE LA INFORMACIÓN

Una base de datos es en esencia una colección de archivos relacionados entre sí, de la cual los usuarios pueden extraer información sin considerar las fronteras de los archivos.

Un objetivo importante de un sistema de base de datos es proporcionar a los usuarios una visión abstracta de los datos, es decir, el sistema esconde ciertos detalles de cómo se almacenan y mantienen los datos. Sin embargo para que el sistema sea manejable, los datos se deben extraer eficientemente.

Existen diferentes niveles de abstracción para simplificar la interacción de los usuarios con el sistema: Interno, conceptual y externo, específicamente el de almacenamiento físico, el del usuario y el del programador.

### 1. Nivel físico (interno)

Es la representación del nivel más bajo de abstracción, en éste se describe en detalle la forma como se almacenan los datos en los dispositivos de almacenamiento (por ejemplo, mediante señaladores o índices para el acceso aleatorio a los datos).

### 2. Nivel conceptual

El siguiente nivel más alto de abstracción, describe qué datos son almacenados realmente en la base de datos y las relaciones que existen entre los mismos, describe la base de datos completa en términos de su estructura de diseño. El nivel conceptual de abstracción lo usan los administradores de bases de datos, quienes deben decidir qué información se va a guardar en la base de datos.

Consta de las siguientes definiciones:

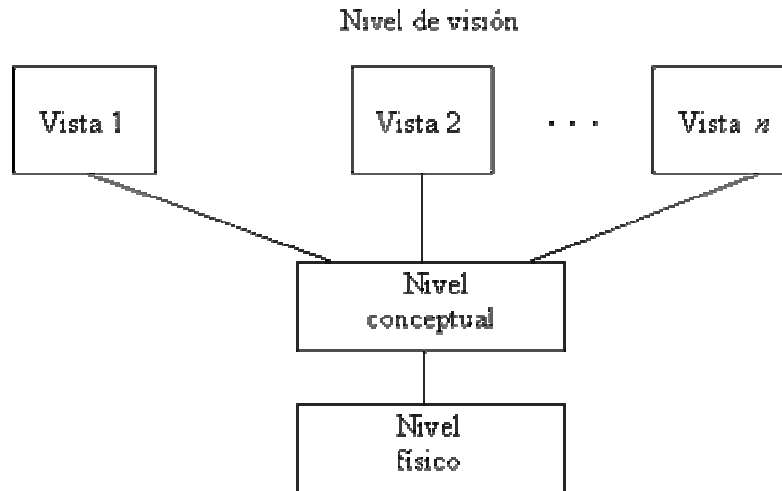
- Definición de los datos: Se describen el tipo de dato y la longitud de campo de todos los elementos direccionables en la base. Los elementos por definir incluyen artículos elementales (atributos), totales de datos y registros conceptuales (entidades).
- Relaciones entre datos: Se definen las relaciones entre datos para enlazar tipos de registros relacionados para el procesamiento de archivos múltiples.

En el nivel conceptual la base de datos aparece como una colección de registros lógicos, sin descriptores de almacenamiento. En realidad los archivos conceptuales no existen físicamente. La transformación de registros conceptuales a registros físicos para el almacenamiento se lleva a cabo por el sistema y es transparente al usuario.

### 3. Nivel de visión (externo)

Nivel más alto de abstracción, es lo que el usuario final puede visualizar del sistema terminado, describe sólo una parte de la base de datos al usuario acreditado para verla. El sistema puede proporcionar muchas visiones para la misma base de datos.

La interrelación entre estos tres niveles de abstracción se ilustra en la siguiente figura.



## MODELOS DE DATOS

Para introducirnos en este tema, empezaremos definiendo que es un modelo.

**Modelo:** Es una representación de la realidad que contiene las características generales de algo que se va a realizar. En base de datos, esta representación se elabora en forma gráfica.

**Modelo de datos:** Es una colección de herramientas conceptuales para describir los datos, las relaciones que existen entre ellos, semántica asociada a los datos y restricciones de consistencia.

Los modelos de datos se dividen en tres grupos:

- Modelos lógicos basados en objetos.
- Modelos lógicos basados en registros.
- Modelos físicos de datos.

## 1. Modelos lógicos basados en objetos

Se usan para describir datos en los niveles conceptual y de visión, es decir, con este modelo representamos los datos de tal forma como nosotros los captamos en el mundo real, tienen una capacidad de estructuración bastante flexible y permiten especificar restricciones de datos explícitamente. Existen diferentes modelos de este tipo, pero el más utilizado por su sencillez y eficiencia es el modelo Entidad-Relación.

**A. Modelo Entidad-Relación.** Denominado por sus siglas como: E-R. Este modelo representa a la realidad a través de entidades, que son objetos que existen y que se distinguen de otros por sus características, por ejemplo: un alumno se distingue de otro por sus características particulares como lo es el nombre, o el número de control asignado al entrar a una institución educativa, así mismo, un empleado, una materia, etc. Las entidades pueden ser de dos tipos:

- **Tangibles:** Son todos aquellos objetos físicos que podemos ver, tocar o sentir.
- **Intangibles:** Son Todos aquellos eventos u objetos conceptuales que no podemos ver, aun sabiendo que existen, por ejemplo: la entidad materia, sabemos que existe, sin embargo, no la podemos visualizar o tocar.

Las características de las entidades en las base de datos se llaman atributos, por ejemplo el nombre, dirección teléfono, grado, grupo, etc. son atributos de la entidad alumno; Clave, número de seguro social, departamento, etc., son atributos de la entidad empleado. A su vez una entidad se puede asociar o relacionar con más entidades a través de relaciones.


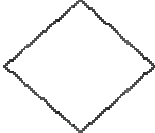


Pero para entender mejor esto, veamos un ejemplo:

Consideremos una empresa que requiere controlar a los vendedores y las ventas que ellos realizan; de este problema determinamos que los objetos o entidades principales a estudiar son el empleado (vendedor) y el artículo (que es el producto en venta), y las características que los identifican son:

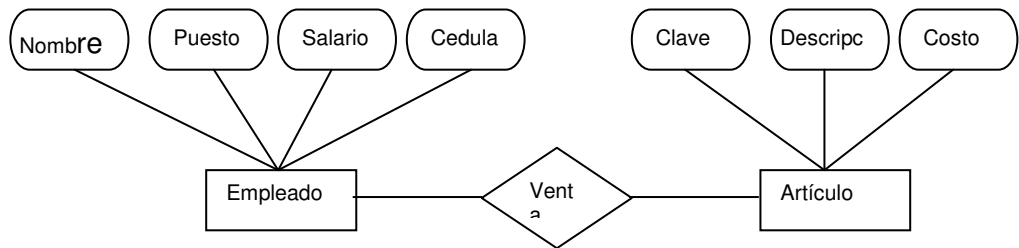
<b>Empleado:</b>	<b>Artículo:</b>
Nombre	Descripción
Puesto	Costo
Salario	Clave
Codigo.	

La relación entre ambas entidades la podemos establecer como Venta.

Bueno, ahora nos falta describir como se representa un modelo E-R gráficamente. La representación es muy sencilla, se emplean símbolos, los cuales son:

<b>Símbolo</b>	<b>Representa</b>
	Entidad
	Relación
	Atributos
	Ligas

Así nuestro ejemplo anterior quedaría representado de la siguiente forma:



Existen más aspectos a considerar con respecto a los modelos entidad relación, los cuales serán considerados en el tema Modelo Entidad Relación.

## 2. Modelos lógicos basados en registros

Se utilizan para describir datos en los niveles conceptual y físico.

Estos modelos utilizan registros e instancias para representar la realidad, así como las relaciones que existen entre estos registros (ligas) o apuntadores. A diferencia de los modelos de datos basados en objetos, se usan para especificar la estructura lógica global de la base de datos y para proporcionar una descripción a nivel más alto de la implementación.

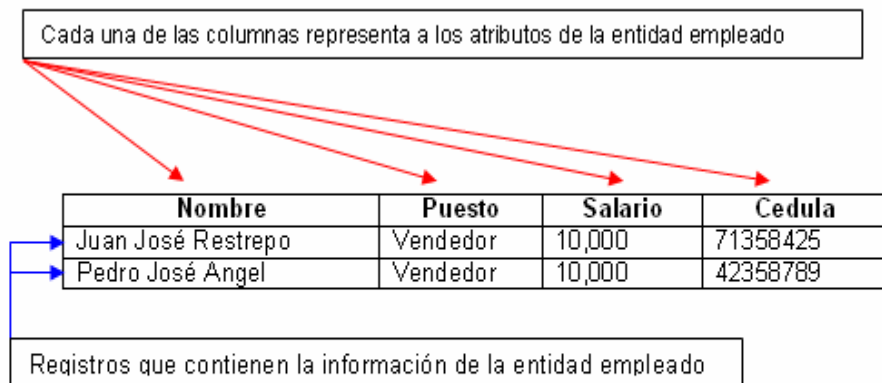
Los tres modelos de datos más ampliamente aceptados son:

- Modelo Relacional
- Modelo de Red
- Modelo Jerárquico

- A. Modelo relacional.** En este modelo se representan los datos y las relaciones entre éstos, a través de una colección de tablas, en las cuales los renglones (tuplas) equivalen a cada uno de los registros que contendrá la base de datos y las columnas corresponden a las características (atributos) de cada registro localizado en la tupla;

Considerando nuestro ejemplo del empleado y el artículo:

**TABLA EMPLEADO**



**TABLA ARTICULO**

Clave	Descripcion	Costo
CO01	Colcha Matrimonial	2,000

Ahora te preguntará ¿cómo se representan las relaciones entre las entidades en este modelo?

Existen dos formas de representarla, pero para ello necesitamos definir que es una llave primaria. Es un atributo el cual definimos como atributo principal, es una forma única de identificar a una entidad. Por ejemplo, la cédula de un empleado lo distingue de otro por que las cédulas no pueden ser iguales.

Ahora si, las formas de representar las relaciones en este modelo son:

Haciendo una tabla que contenga cada una de las llaves primarias de las entidades involucradas en la relación.

Tomando en cuenta que la llave primaria del empleado es su cédula, y la llave primaria del artículo es la Clave.

Cedula	Clave
71358425	C001
42358789	C110

Incluyendo en alguna de las tablas de las entidades involucradas, la llave de la otra tabla.

Nombre	Puesto	Salario	Cedula	Clave
Juan José Restrepo	Vendedor	10,000	71358425	C001
Pedro José Ángel	Vendedor	10,000	42358789	C110

**B. Modelo de red.** Este modelo representa los datos mediante colecciones de registros y sus relaciones se representan por medio de ligas o enlaces, los cuales pueden verse como punteros. Los registros se organizan en un conjunto de gráficas arbitrarias.

Para profundizar en este tema visitemos: Modelo de datos de red (unidad 4)

**C. Modelo jerárquico.** Es similar al modelo de red en cuanto a las relaciones y datos, ya que estos se representan por medio de registros y sus ligas. La diferencia radica en que están organizados por conjuntos de árboles en lugar de gráficas arbitrarias.

Para profundizar más en el tema visita: Modelo de datos jerárquico (unidad 4).

### 3. Modelos físicos de datos

Se usan para describir a los datos en el nivel más bajo, aunque existen muy pocos modelos de este tipo, básicamente capturan aspectos de la implementación de los sistemas de base de datos. Existen dos clasificaciones de este tipo que son:

- Modelo unificador
- Memoria de elementos.

## OTROS CONCEPTOS ASOCIADOS A UNA BASE DE DATOS

### 1. Instancias y esquemas

Con el paso del tiempo la información que se va acumulando y desechando en la base de datos, ocasiona que ésta cambie.

Denominamos

- **Instancia.** Al estado que presenta una base de datos en un momento dado. Veámoslo como una fotografía que tomamos de la base de datos en un tiempo  $t$ , después de que transcurre el tiempo  $t$  la base de datos ya no es la misma.
- **Esquema.** Es la descripción lógica de la base de datos, proporciona los nombres de las entidades y sus atributos especificando las relaciones que existen entre ellos. Es un banco en el que se inscriben los valores que irán formando cada uno de los atributos. El esquema no cambia los que varían son los datos y con esto tenemos una nueva instancia.

*Ejemplo:*

Considerando el ejemplo del vendedor que vende artículos, esquema e instancia según nuestro ejemplo, quedaría:

Esquema:

{ Vendedor : Nombre, puesto, salario, Cedula }  
 { Articulo : Clave, costo, descripción }

Instancia:

José Restrepo	Vendedor	10,000	71358425	—	C001	Colcha Matrimonial	2,000
---------------	----------	--------	----------	---	------	--------------------	-------

Como podemos observar el esquema nos muestra la estructura en el cual se almacenaran los datos, en este caso en registros cuyos nombres de campos son: por parte del vendedor (Nombre, puesto, salario, cédula) y por el artículo (Clave, costo, descripción); La instancia representa a una serie de datos almacenados en los registros establecidos por el esquema, estos datos varían, no permanecen fijos en el tiempo

## 2. Independencia de los datos

Se refiere a la protección contra los programas de aplicación que puedan originar modificaciones cuando se altera la organización física o lógica de la base de datos.

Existen 2 niveles de independencia de datos.

- Independencia física de datos.** Es la capacidad de modificar el esquema físico sin provocar que se vuelvan a escribir los programas de aplicación.
- Independencia lógica de datos.** Capacidad de modificar el esquema conceptual sin provocar que se vuelvan a escribir los programas de aplicación.



### 3. Lenguaje de definición de datos

El lenguaje de definición de datos, denominado por sus siglas como: DDL (Data definition Language), permite definir un esquema de base de datos por medio de una serie de definiciones que se expresan en un lenguaje especial; el resultado de estas definiciones se almacena en un archivo especial llamado diccionario de datos.

### 4. Lenguaje de manipulación de datos

La manipulación de datos se refiere a las operaciones de insertar, recuperar, eliminar o modificar datos; dichas operaciones son realizadas a través del lenguaje de manipulación de datos (DML, Data Manipulation Language), que es quién permite el acceso de los usuarios a los datos.

Existen básicamente dos tipos de lenguajes de manipulación de datos:

- A. **Procedimentales.** Los LMD requieren que el usuario especifique que datos se necesitan y cómo obtenerlos.
- B. **No procedimentales.** Los LMD requieren que el usuario especifique que datos se necesitan y sin especificar cómo obtenerlos.

### 5. Manejador de Bases de Datos

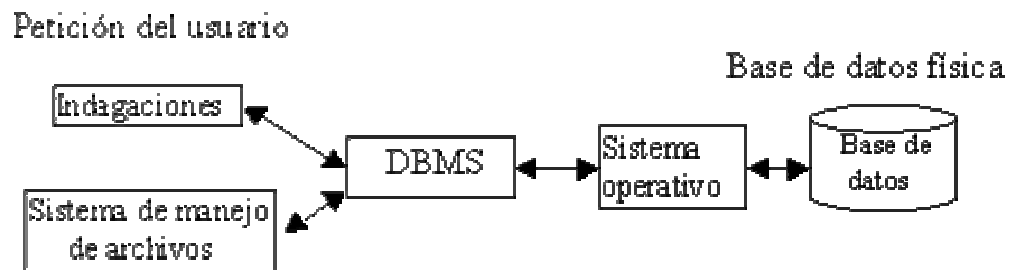
El sistema manejador de bases de datos es la porción más importante del software de un sistema de base de datos. Un DBMS es una colección de numerosas rutinas de software interrelacionadas, cada una de las cuales es responsable de alguna tarea específica.

Las funciones principales de un DBMS son:

- Crear y organizar la Base de datos.
- Establecer y mantener las trayectorias de acceso a la base de datos de tal forma que los datos puedan ser accedados rápidamente.
- Manejar los datos de acuerdo a las peticiones de los usuarios.
- Registrar el uso de las bases de datos.
- Interacción con el manejador de archivos. Esto a través de las sentencias en DML al comando del sistema de archivos. Así el Manejador de base de datos es el responsable del verdadero almacenamiento de los datos.
- Respaldo y recuperación. Consiste en contar con mecanismos implantados que permitan la recuperación fácilmente de los datos en caso de ocurrir fallas en el sistema de base de datos.

- Control de concurrencia. Consiste en controlar la interacción entre los usuarios Concurrentes para no afectar la inconsistencia de los datos.
- Seguridad e integridad. Consiste en contar con mecanismos que permitan el control de la consistencia de los datos evitando que estos se vean perjudicados por cambios no autorizados o previstos.

El DBMS es conocido también como Gestor de Base de datos.



La figura muestra el DBMS como interfase entre la base de datos física y las peticiones del usuario. El DBMS interpreta las peticiones de entrada/salida del usuario y las manda al sistema operativo para la transferencia de datos entre la unidad de memoria secundaria y la memoria principal.

En sí, un sistema manejador de base de datos es el corazón de la base de datos ya que se encarga del control total de los posibles aspectos que la puedan afectar.

## 6. Administrador de Bases de Datos

Denominado por sus siglas como: DBA, Database Administrator. Es la persona encargada y que tiene el control total sobre el sistema de base de datos, sus funciones principales son:

- Definición de esquema. Es el esquema original de la base de datos se crea escribiendo un conjunto de definiciones que son traducidas por el compilador de DDL a un conjunto de tablas que son almacenadas permanentemente en el diccionario de datos.
- Definición de la estructura de almacenamiento del método de acceso. Estructuras de almacenamiento y de acceso adecuados se crean escribiendo un conjunto de definiciones que son traducidas por el compilador del lenguaje de almacenamiento y definición de datos.
- Concesión de autorización para el acceso a los datos. Permite al administrador de la base de datos regular las partes de las bases de datos que van a ser accedidas por varios usuarios.

- Especificación de limitantes de integridad. Es una serie de restricciones que se encuentran almacenados en una estructura especial del sistema que es consultada por el gestor de base de datos cada vez que se realice una actualización al sistema.

## **7. Usuarios de las bases de datos.**

Podemos definir a los usuarios como toda persona que tenga todo tipo de contacto con el sistema de base de datos desde que este se diseña, elabora, termina y se usa.

Los usuarios que accedan una base de datos pueden clasificarse como:

- Programadores de aplicaciones. Los profesionales en computación que interactúan con el sistema por medio de llamadas en DML (Lenguaje de Manipulación de Datos), las cuales están incorporadas en un programa escrito en un lenguaje de programación (Por ejemplo, COBOL, PL/I, Pascal, C, etc.)
- Usuarios sofisticados. Los usuarios sofisticados interactúan con el sistema sin escribir programas. En cambio escriben sus preguntas en un lenguaje de consultas de base de datos.
- Usuarios especializados. Algunos usuarios sofisticados escriben aplicaciones de base de datos especializadas que no encajan en el marco tradicional de procesamiento de datos.
- Usuarios ingenuos. Los usuarios no sofisticados interactúan con el sistema invocando a uno de los programas de aplicación permanentes que se han escrito anteriormente en el sistema de base de datos, podemos mencionar al usuario ingenuo como el usuario final que utiliza el sistema de base de datos sin saber nada del diseño interno del mismo por ejemplo: un cajero.

## **8. Estructura general del sistema.**

Un sistema de base de datos se encuentra dividido en módulos cada uno de los cuales controla una parte de la responsabilidad total de sistema. En la mayoría de los casos, el sistema operativo proporciona únicamente los servicios más básicos y el sistema de la base de datos debe partir de esa base y controlar además el manejo correcto de los datos. Así el diseño de un sistema de base de datos debe incluir la interfaz entre el sistema de base de datos y el sistema operativo.

Los componentes funcionales de un sistema de base de datos son:

- Gestor de archivos: Gestiona la asignación de espacio en la memoria del disco y de las estructuras de datos usadas para representar información.
- Manejador de base de datos: Sirve de interfaz entre los datos y los programas de aplicación.

- **Procesador de consultas:** Traduce las proposiciones en lenguajes de consulta a instrucciones de bajo nivel. Además convierte la solicitud del usuario en una forma más eficiente.
- **Compilador de DDL:** Convierte las proposiciones DDL en un conjunto de tablas que contienen metadatos, estas se almacenan en el diccionario de datos.
- **Archivo de datos:** En él se encuentran almacenados físicamente los datos de una organización.
- **Diccionario de datos:** Contiene la información referente a la estructura de la base de datos.
- **Índices:** Permiten un rápido acceso a registros que contienen valores específicos.

Una forma gráfica de representar los componentes antes mencionados y la relación que existe entre ellos sería la siguiente.



**ACTIVIDAD...****1. Responder al siguiente cuestionario**

1 Un campo (como: Nombre, dirección, tel.) de un registro, representa a:

- A. Una entidad.
- B. Un atributo de una entidad.
- C. Una relación

2 La abstracción de la información se refiere a:

- A. Proporcionar a todos los usuarios de la base de datos la información global de la misma.
- B. Es una serie de datos en un instante de la base de datos.
- C. Proporciona a cada usuario la vista de la información concerniente a una base de datos según su relación con la misma.

3 Una Instancia es:

- A. Una serie de información en un momento dado de la base de datos.
- B. La información proporcionada al realizar una consulta a la base de datos.
- C. La estructura lógica de la base de datos.

4 El DDL es el lenguaje que nos permite:

- A. Realizar consultas, modificaciones y alteraciones a la información de una base de datos.
- B. Establecer la integridad de los datos.
- C. Definir la estructura conceptual y externa de los datos que contendrá la DB.

5 El Manejador de Bases de Datos (DBMS)

- A. Es la persona encargada del sistema de base de datos
- B. Es el software que funciona como interfaz entre los programas de aplicación de bases de datos y los usuarios.
- C. Es un sistema operativo espacial para bases de datos.

**ESTRATEGIAS METODOLÓGICAS**

- Presentación de la unidad a cargo del profesor.

**RECURSOS**

- Humanos: Profesor y alumnos
- Institucionales: Salón de clase
- Materiales: Texto guía

**INDICADORES DE EVALUACIÓN**

- Evaluación escrita sobre la unidad
- Evaluación de las actividades propuestas en la unidad.

**CRITERIOS DE EVALUACIÓN**

- ¿Qué Objetivos tienen las bases de datos?
- ¿En qué consiste la abstracción de la información?
- ¿Cuáles son los modelos de datos?
- ¿Qué es un DBMS?
- ¿Cuáles son las funciones de un DBA?
- ¿Cómo se clasifican los usuarios que accedan una base de datos?

## UNIDAD 2

**MODELO ENTIDAD-RELACIÓN****INTRODUCCIÓN**

Esta unidad busca que el estudiante distinga los conceptos involucrados en la construcción de un modelo Entidad – Relación (E-R), tales como las entidades, relaciones, tipos de relaciones, llaves primarias. Así mismo, como se convierte un modelo E-R a tablas.

**JUSTIFICACIÓN**

Los diagramas Entidad – Relación (E-R) proporcionan una notación para documentar un diseño tentativo de bases de datos. El diseñador capta las características importantes de aplicación con diagramas E-R , que entonces traslada a un esquema específico de bases de datos.

Aun cuando el modelo original E-R da una visión de la forma en que las entidades, relaciones y restricciones pueden captar el significado inherente de una aplicación, su apropiación más significativa es la técnica para representar la aplicación de una forma concisa y descriptiva; este análisis se refiere sólo a diagramas E-R, que son dispositivos de comunicación para diseñadores que trabajan en definir la base de datos. Ningún producto comercial de bases de datos pone en práctica directamente el modelo E-R. Después de definir la aplicación, por medio de diagramas E-R y otra documentación, es necesario que se traslade a un esquema específico de bases de datos.

**OBJETIVO GENERAL**

Proporcionar los conocimientos que permitan al estudiante asimilar los conceptos fundamentales de un modelo Entidad - Relación.



## OBJETIVOS ESPECÍFICOS

- Identificar los conceptos básicos de un modelo E-R
- Reconocer los diferentes tipos de asociación existentes entre dos o más entidades.
- Determinar la importancia de las llaves primarias en un modelo E-R
- Conocer la metodología para reducir un diagrama E-R a tablas
- Identificar conceptos asociados a un modelo E-R de generalización, especialización y agregación.

## CONTENIDO

1. Entidades y conjunto de entidades
2. Relaciones y conjunto de relaciones.
3. Limitantes de mapeo - Tipos de relaciones
4. Llaves primarias
5. Diagrama Entidad-Relación
6. Reducción de diagramas E-R a tablas
7. Generalización y especialización
8. Agregación

## ENTIDADES Y CONJUNTOS DE ENTIDADES

Una entidad es un objeto que existe y se distingue de otros objetos de acuerdo a sus características llamadas atributos. Las entidades pueden ser concretas como una persona o abstractas como una fecha.

Un conjunto de entidades es un grupo de entidades del mismo tipo. Por ejemplo el conjunto de entidades CUENTA, podría representar al conjunto de cuentas de un banco X, o ALUMNO representa a un conjunto de entidades de todos los alumnos que existen en una institución.

Una entidad se caracteriza y distingue de otra por los atributos, en ocasiones llamadas propiedades, que representan las características de una entidad. Los atributos de una entidad pueden tomar un conjunto de valores permitidos al que se le conoce como

dominio del atributo. Así cada entidad se describe por medio de un conjunto de parejas formadas por el nombre del atributo y el valor de dato. Habrá una pareja para cada atributo del conjunto de entidades.

*Ejemplo:*

Hacer una descripción en pareja para la entidad alumno con los atributos Código, Nombre y Especialidad.

<b>Nombre_atributo</b>	,	<b>Valor</b>
Código	,	96310418
Nombre	,	Sánchez Osuna Ana
Especialidad	,	LI

O considerando un ejemplo para Vendedor cuyos atributos son: Cedula, Nombre, Salario.

<b>Nombre_atributo</b>	,	<b>Valor</b>
Cedula	,	70520358
Nombre	,	Daniel Colín Morales
Salario	,	433800

## RELACIONES Y CONJUNTOS DE RELACIONES

Una relación es la asociación que existe entre dos a más entidades.

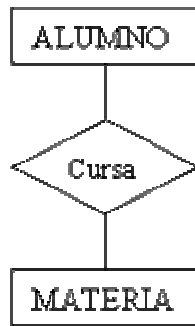
Un conjunto de relaciones es un grupo de relaciones del mismo tipo.

La cantidad de entidades en una relación determina el grado de la relación, por ejemplo la relación ALUMNO-MATERIA es de grado 2, ya que intervienen la entidad ALUMNO y la entidad MATERIA, la relación PADRES, puede ser de grado 3, ya que involucra las entidades PADRE, MADRE e HIJO.

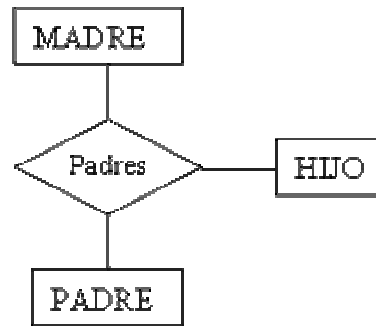
Aunque el modelo E-R permite relaciones de cualquier grado, la mayoría de las aplicaciones del modelo sólo consideran relaciones del grado 2. Cuando son de tal tipo, se denominan relaciones binarias.

La función que tiene una relación se llama papel, generalmente no se especifican los papeles o roles, a menos que se quiera aclarar el significado de una relación.

Diagrama E-R (sin considerar los atributos, sólo las entidades) para los modelos ejemplificados:



Ejemplo de relación de grado 2



Ejemplo de relación de grado 3

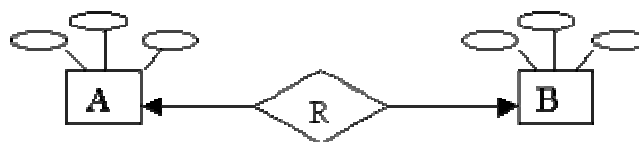
**LIMITANTES DE MAPEO**

Existen 4 tipos de relaciones que pueden establecerse entre entidades, las cuales establecen con cuantas entidades de tipo B se pueden relacionar una entidad de tipo A:

1. **Relación uno a uno.** Se presenta cuando existe una relación como su nombre lo indica uno a uno, denominado también relación de matrimonio. Una entidad del tipo A solo se puede relacionar con una entidad del tipo B, y viceversa.

Por ejemplo: la relación asignación de automóvil que contiene a las entidades EMPLEADO, AUTO, es una relación 1 a 1, ya que asocia a un empleado con un único automóvil por lo tanto ningún empleado posee más de un automóvil asignado, y ningún vehículo se asigna a más de un trabajador.

Es representado gráficamente de la siguiente manera:



- A: Representa a una entidad de cualquier tipo diferente a una entidad B.
- R: En el diagrama representa a la relación que existe entre las entidades.

El extremo de la flecha que se encuentra punteada indica el uno de la relación, en este caso, una entidad A ligada a una entidad B.

2. **Relación uno a muchos.** Significa que una entidad del tipo A puede relacionarse con cualquier cantidad de entidades del tipo B, y una entidad del tipo B solo puede estar relacionada con una entidad del tipo A.

Su representación gráfica es la siguiente:

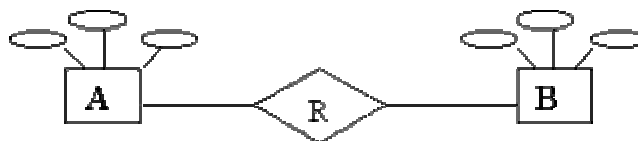


Nótese en este caso que el extremo punteado de la flecha de la relación de A y B, indica una entidad A conectada a muchas entidades B.

3. **Muchos a uno.** Indica que una entidad del tipo B puede relacionarse con cualquier cantidad de entidades del tipo A, mientras que cada entidad del tipo A solo puede relacionarse con solo una entidad del tipo B.



4. **Muchas a muchas.** Establece que cualquier cantidad de entidades del tipo A pueden estar relacionados con cualquier cantidad de entidades del tipo B.



A los tipos de relaciones antes descritos, también se le conoce como cardinalidad.

La cardinalidad nos especifica los tipos de relaciones que existen entre las entidades en el modelo E-R y establecer con esto las validaciones necesarias para conseguir que los datos de la instancia (valor único en un momento dado de una base de datos) correspondan con la realidad.

Algunos ejemplos de cardinalidades de la vida común pueden ser:

**Uno a uno.**

El noviazgo, la cédula de cada persona, El NIT personal, El acta de nacimiento, ya que solo existe un solo documento de este tipo para cada una de las diferentes personas.

**Uno a muchos.**

Cliente – Cuenta en un banco, Padre-Hijos, Camión-Pasajeros, zoológico - animales, árbol – hojas.

**Muchos a muchos.**

Arquitecto – proyectos, fiesta – personas, estudiante – materias.

**NOTA:** Cabe mencionar que la cardinalidad para cada conjunto de entidades depende del punto de vista que se le de al modelo en estudio, claro está, sujetándose a la realidad.

Otra clase de limitantes lo constituye la dependencia de existencia.

Refiriéndonos a las mismas entidades A y B, decimos que si la entidad A depende de la existencia de la entidad B, entonces A es dependiente de existencia por B, si eliminamos a B tendríamos que eliminar por consecuente la entidad A, en este caso B es la entidad Dominante y A es la entidad subordinada.

## LLAVES PRIMARIAS

Como ya se ha mencionado anteriormente, la distinción de una entidad entre otra se debe a sus atributos, lo cual lo hacen único. Una llave primaria es aquel atributo el cual consideramos clave para la identificación de los demás atributos que describen a la entidad. Por ejemplo, si consideramos la entidad ALUMNO del Politécnico Colombiano Jaime Isaza Cadavid, podríamos tener los siguientes atributos: Nombre, Semestre, Especialidad, Dirección, Teléfono, código, de todos estos atributos el que podremos designar como llave primaria es el código, ya que es diferente para cada alumno y este nos identifica en la institución.

Claro que puede haber más de un atributo que pueda identificarse como llave primaria en este caso se selecciona la que consideremos más importante, los demás atributos son denominados llaves secundarias.

Una clave o llave primaria es indicada gráficamente en el modelo E-R con una línea debajo del nombre del atributo.

**DIAGRAMA ENTIDAD - RELACIÓN**

Denominado por sus siglas como E-R; este modelo representa a la realidad a través de un esquema gráfico empleando la terminología de entidades, que son objetos que existen y son los elementos principales que se identifican en el problema a resolver con el diagramado y se distinguen de otros por sus características particulares denominadas atributos, el enlace que rige la unión de las entidades esta representada por la relación del modelo.

Recordemos que un rectángulo nos representa a las entidades; una elipse a los atributos de las entidades, y una etiqueta dentro de un rombo nos indica la relación que existe entre las entidades, destacando con líneas las uniones de estas y que la llave primaria de una entidad es aquel atributo que se encuentra subrayado.

A continuación mostraremos algunos ejemplos de modelos E-R, considerando las cardinalidades que existen entre ellos:

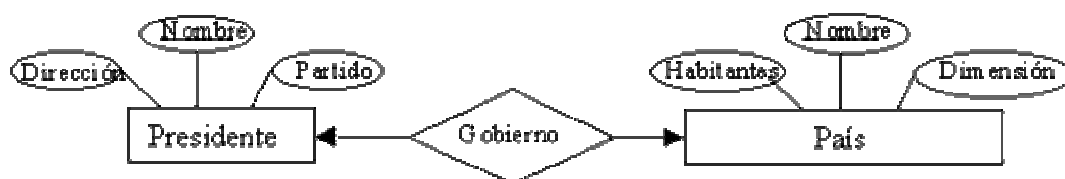
**1. Relación Uno a Uno.**

*Problema:* Diseñar el modelo E-R, para la relación Registro de automóvil que consiste en obtener la tarjeta de circulación de un automóvil con los siguientes datos:-  
Automóvil: Modelo, Placas, Color - Tarjeta de circulación: Propietario, No\_serie, Tipo.



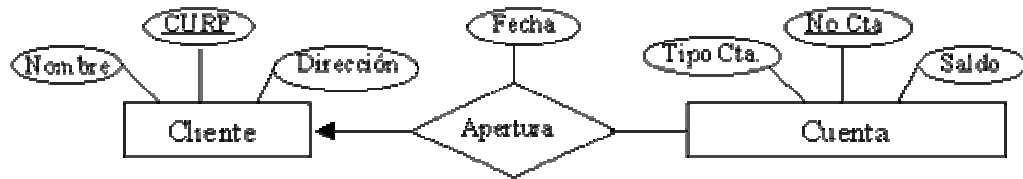
Indicamos con este ejemplo que existe una relación de pertenencia de uno a uno, ya que existe una tarjeta de circulación registrada por cada automóvil.

En este ejemplo, representamos que existe un solo presidente para cada país.



2. Relación muchos a muchos.

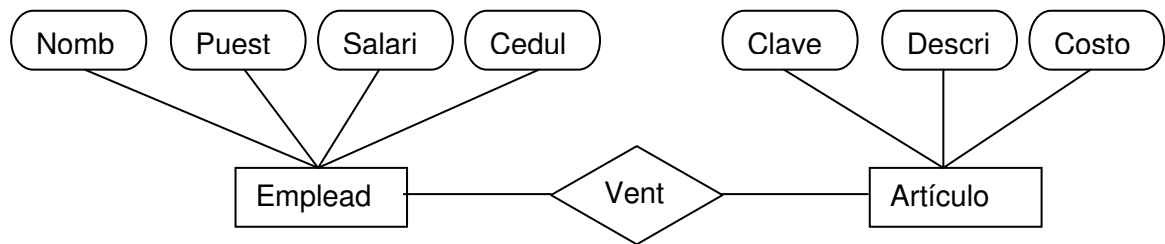
El siguiente ejemplo indica que un cliente puede tener muchas cuentas, pero que una cuenta puede llegar a pertenecer a un solo cliente (Decimos puede, ya que existen cuentas registradas a favor de más de una persona).



**REDUCCIÓN DE DIAGRAMAS E - R A TABLAS**

Un diagrama E-R, puede ser representado también a través de una colección de tablas. Para cada una de las entidades y relaciones existe una tabla única a la que se le asigna como nombre el del conjunto de entidades y de las relaciones respectivamente, cada tabla tiene un número de columnas que son definidas por la cantidad de atributos y las cuales tienen el nombre del atributo.

La transformación de nuestro ejemplo Venta en la que intervienen las entidades de Empleado con los atributos Cedula, nombre, puesto, salario y Artículo con los atributos Clave, descripción, costo y cuyo diagrama E-R es el siguiente:



Entonces las tablas resultantes siguiendo la descripción anterior son:

Tabla Empleado

Nombre	Puesto	Salario	Cedula
Teófilo	Vendedor	2000	70120358
Cesar	Auxiliar ventas	1200	52334873

## Tabla artículo

Clave	Descripción	Costo
A100	Abanico	460
C260	Colcha matrimonial	1200

## Tabla Venta

Cedula	Clave
70120358	C260
52334873	A100

Nótese que en la tabla de relación - Venta -, contiene como atributos a las llaves primarias de las entidades que intervienen en dicha relación, en caso de que exista un atributo en las relaciones, este atributo es anexado como una fila más de la tabla;

Por ejemplo si anexamos el atributo fecha a la relación venta, la tabla que se originaría sería la siguiente:

Cedula	Clave	Fecha
70120358	C260	10/12/96
52334873	A100	11/12/96

## GENERALIZACIÓN Y ESPECIALIZACIÓN

### 1. Generalización.

Es el resultado de la unión de 2 o más conjuntos de entidades (de bajo nivel) para producir un conjunto de entidades de más alto nivel. La generalización se usa para hacer resaltar los parecidos entre tipos de entidades de nivel más bajo y ocultar sus diferencias.

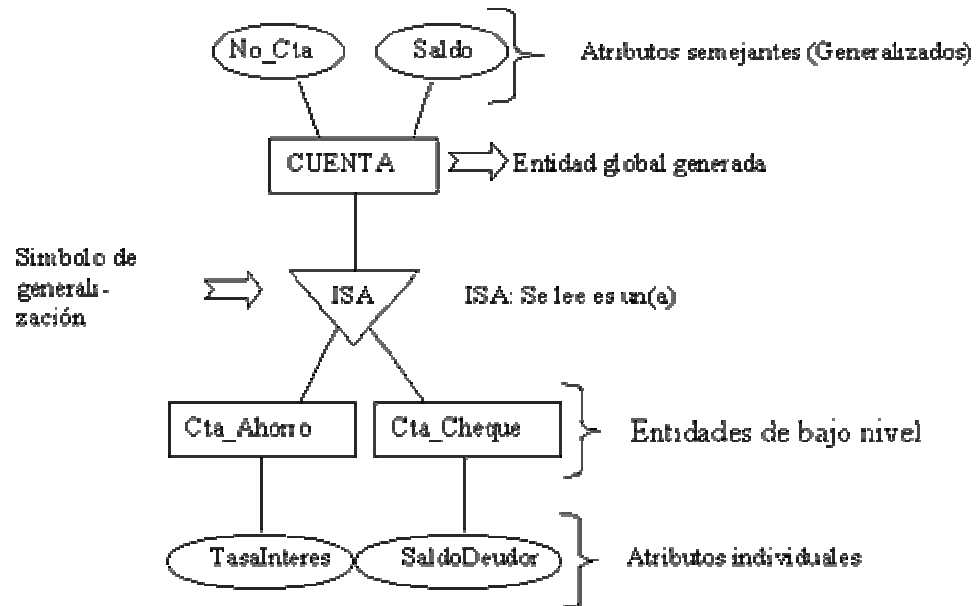
La generalización consiste en identificar todos aquellos atributos iguales de un conjunto de entidades para formar una entidad(es) global(es) con dichos atributos semejantes, dicha entidad(es) global(es) quedará a un nivel más alto al de las entidades origen.

*Ejemplo:*



Tomando el ejemplo donde: Se tiene las entidades Cta\_Ahorro y Cta\_Cheques, ambas tienen los atributos semejantes de No\_Cta y Saldo, aunque además de estos dos atributos, Cta\_Ahorro tiene el atributo Tasa\_Interes y Cta\_Cheques el atributo Saldo\_Deudor. De todos estos atributos podemos juntar (generalizar) No\_Cta y Saldo que son iguales en ambas entidades.

Entonces tenemos:



Podemos leer esta gráfica como: La entidad Cta\_Ahorro hereda de la entidad CUENTA los atributos No\_Cta y Saldo, además del atributo de TasaInteres, de forma semejante Cta\_Cheque tiene los atributos de No\_Cta, Saldo y SaldoDeudor.

Como podemos observar la Generalización trata de eliminar la redundancia (repetición) de atributos, al englobar los atributos semejantes. La entidad(es) de bajo nivel cuentan (heredan) todos los atributos correspondientes.

## 2. Especialización:

Es el resultado de tomar un subconjunto de entidades de alto nivel para formar un conjunto de entidades de más bajo nivel.

En la generalización cada entidad de alto nivel debe ser también una entidad de bajo nivel. La especialización no tiene este limitante.

Se representa por medio de un triángulo denominado con la etiqueta "ISA", se distingue de la generalización por el grosor de las líneas que conectan al triángulo con las entidades.

La especialización denota la diferencia entre los conjuntos de entidades de alto y bajo nivel.

**AGREGACIÓN**

La agregación surge de la limitación que existe en el modelado de E-R, al no permitir expresar las relaciones entre relaciones de un modelo E-R en el caso de que una relación X se quiera unir con una entidad cualquiera para formar otra relación.

La Generalización consiste en agrupar por medio de un rectángulo a la relación (representada por un rombo) junto con las entidades y atributos involucrados en ella, para formar un grupo que es considerado una entidad y ahora sí podemos relacionarla con otra entidad.

Para ejemplificar lo anterior consideremos el ejemplo en donde el problema consiste en que existen trabajando muchos empleados que trabajan en diferentes proyectos, pero dependiendo del trabajo que realizan pueden llegar a utilizar un equipo o maquinaria; en este problema intervienen 3 entidades: Empleado, Proyecto y Maquinaria, el diagrama E-R correspondiente es:

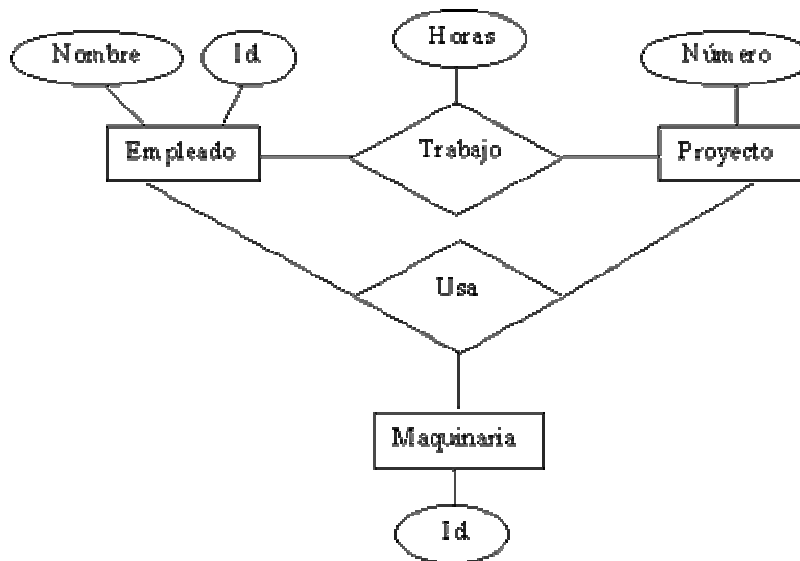


Diagrama E-R con relaciones redundantes

Como el modelo E-R no permite la unión entre dos o más relaciones, la relación trabajo es englobada como si fuera una entidad más de la relación usa, gráficamente queda como:

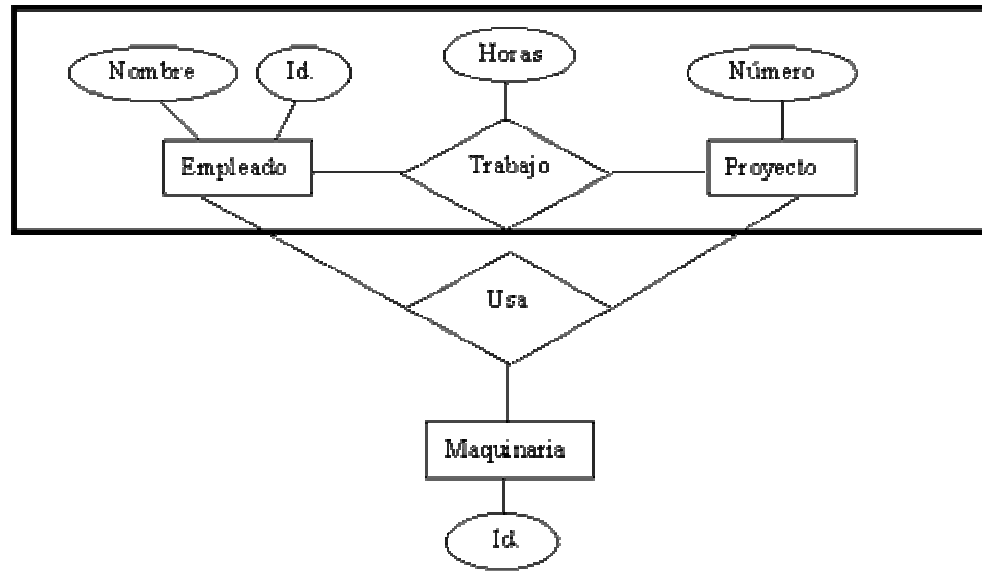


Diagrama E-R con agregación

Ahora podemos decir que la entidad trabajo se relaciona con la entidad maquinaria a través de la relación usar, para indicarnos que un trabajo usa un determinado equipo o maquinaria según el tipo de trabajo que se trate.

### ACTIVIDAD...

2. Identificar Entidades, atributos, relaciones para un Acuario, en el cual las especificaciones son las siguientes:

- A. El acuario mantiene muchos tanques, de varios nombres, colores y volúmenes, y cada tanque contiene una variedad de peces.
- B. Cada pez tiene un nombre, color y peso, y cada uno representa uno de muchas especies.
- C. Para cada especie, se mantiene el nombre y la preferencia de dieta.
- D. Durante su vida en el acuario, cada pez experimenta muchos eventos, tales como nacimiento, muerte, alimentación y encuentros belicosos con otros peces. Cada evento pertenece a un pez en particular, y se guarda la fecha y un comentario textual de la naturaleza del evento

2. Elaborar el diagrama E-R para el caso del acuario (anterior)

3. Elaborar la reducción del diagrama E-R del numeral anterior a tablas.

### ESTRATEGIAS METODOLÓGICAS

- Presentación de la unidad a cargo del profesor.
- Desarrollo del modelo E-R aplicado a un caso específico a cargo del docente.
- Realización por parte de los alumnos de las actividades propuestas en la unidad.

### RECURSOS

- Humanos: Profesor y alumnos
- Institucionales: Salón de clase
- Materiales: Texto guía

### INDICADORES DE EVALUACIÓN

- Evaluación escrita sobre la unidad
- Evaluación de las actividades propuestas en la unidad.

### CRITERIOS DE EVALUACIÓN

- ¿Qué importancia tiene el modelo Entidad - Relación?
- ¿Cuáles son los elementos de un modelo E-R?
- ¿Cuáles son los tipos de relaciones que se pueden dar entre dos o más entidades?
- ¿Cuáles son pasos para reducir un diagrama E-R a tablas?

## UNIDAD 3

**MODELO RELACIONAL****INTRODUCCIÓN**

Esta unidad busca que el estudiante distinga los conceptos involucrados en la construcción de un modelo relacional, los lenguajes de consulta y modificación de la base de datos, así mismo los conceptos básicos necesarios para el diseño de una base de datos relacional.

**JUSTIFICACIÓN**

Un modelo de bases de datos describe una estructura para almacenar y manipular información en una base de datos. El modelo relacional utiliza una estructura particularmente sencilla, en el cual una base de datos es un conjunto de tablas, cada una de ellas estructuralmente semejante a un archivo de registros de longitud fija. Debido a esto, el modelo relacional es accesible a una amplia variedad de usuarios no técnicos.

Las bases de datos relacionales son el tipo de bases de datos actualmente más difundido. Los motivos de este éxito son fundamentalmente dos:

1. Ofrecen sistemas simples y eficaces para representar y manipular los datos
2. Se basan en un modelo, el relacional, con sólidas bases teóricas

El modelo relacional fue propuesto originariamente por E.F. Codd en un ya famoso artículo de 1970. Gracias a su coherencia y facilidad de uso, el modelo se ha convertido en los años 80 en el más usado para la producción de DBMS.

**OBJETIVO GENERAL**

Proporcionar los conocimientos que permitan al estudiante identificar la estructura de un modelo relacional, así como los lenguajes necesarios para su manipulación.

**OBJETIVOS ESPECÍFICOS**

- Reconocer la estructura de una base de datos relacional.
- Identificar los lenguajes formales y comerciales para la manipulación de una base de datos.
- Conocer las instrucciones necesarias para la modificación de una base de datos.
- Conocer los peligros existentes en el diseño de una base de datos relacional
- Identificar y aplicar los pasos para la normalización de una base de datos relacional.

**CONTENIDO**

1. Conceptos básicos
2. Estructura de las bases de datos relacionales
3. Lenguajes de consulta
4. Diseño de bases de datos relacionales
5. Normalización

**CONCEPTOS BÁSICOS**

La ventaja del modelo relacional es que los datos se almacenan, al menos conceptualmente, de un modo en que los usuarios entienden con mayor facilidad. Los datos se almacenan como tablas y las relaciones entre las filas y las tablas son visibles en los datos. Este enfoque permite a los usuarios obtener información de la base de datos sin asistencia de sistemas profesionales de administración de información.

Las características más importantes de los modelos relacionales son:

Es importante saber que las entradas en la tabla tienen un solo valor (son atómicos); no se admiten valores múltiples, por lo tanto la intersección de un renglón con una columna tiene un solo valor, nunca un conjunto de valores.

Todas las entradas de cualquier columna son de un solo tipo. Por ejemplo, una columna puede contener nombres de clientes, y en otra puede tener fechas de nacimiento. Cada columna posee un nombre único, el orden de las columnas no es de importancia para la tabla, las columnas de una tabla se conocen como atributos. Cada atributo tiene un dominio, que es una descripción física y lógica de valores permitidos.

No existen 2 filas en la tabla que sean idénticas.

La información en las bases de datos son representados como datos explícitos, no existen apuntadores o ligas entre las tablas.

El enfoque relacional es sustancialmente distinto de otros enfoques en términos de sus estructuras lógicas y del modo de las operaciones de entrada/salida. En el enfoque relacional, los datos se organizan en tablas llamadas relaciones, cada una de las cuales se implanta como un archivo. En terminología relacional una fila en una relación representa un registro o una entidad; cada columna en una relación representa un campo o un atributo.

Así, una relación se compone de una colección de entidades (o registros) cuyos propietarios están descritos por cierto número de atributos predeterminados implantados como campos.

## ESTRUCTURA DE LAS BASES DE DATOS RELACIONALES

La arquitectura relacional se puede expresar en términos de tres niveles de abstracción: nivel interno, conceptual y de visión.

La arquitectura relacional consta de los siguientes componentes:

**Modelo relacional de datos.** En el nivel conceptual, el modelo relacional de datos está representado por una colección de relaciones almacenadas. Cada registro de tipo conceptual en un modelo relacional de datos se implanta como un archivo almacenado distinto.

**Submodelo de datos.** Los esquemas externos de un sistema relacional se llaman submodelos relacionales de datos; cada uno consta de uno a más escenarios (vistas) para describir los datos requeridos por una aplicación dada. Un escenario puede incluir datos de una o más tablas de datos. Cada programa de aplicación está provisto de un buffer ("Área de trabajo de usuario") donde el DBMS puede depositar los datos recuperados de la base para su procesamiento, o puede guardar temporalmente sus salidas antes de que el DBMS las escriba en la base de datos.

Esquema de almacenamiento. En el nivel interno, cada tabla base se implanta como un archivo almacenado. Para las recuperaciones sobre las claves principal o secundaria se pueden establecer uno o más índices para acceder un archivo almacenado.

Sublenguaje de datos. Es un lenguaje de manejo de datos para el sistema relacional. El álgebra relacional y cálculo relacional, son lenguajes "relacionalmente completos", esto es, cualquier relación que pueda derivarse de una o más tablas de datos, también se puede derivar con un solo comando del sublenguaje. Por tanto, el modo de operación de entrada/Salida en un sistema relacional se puede procesar en la forma una tabla a la vez en lugar de un registro a la vez; en otras palabras, se puede recuperar una tabla en vez de un solo registro con la ejecución de un comando del sublenguaje de datos.

## LENGUAJES DE CONSULTA

### 1. LENGUAJES DE CONSULTA FORMALES

Los lenguajes de consulta son los lenguajes en el que los usuarios solicitan información de la base de datos. Estos lenguajes son generalmente de más alto nivel que los lenguajes de programación. Los lenguajes de consulta pueden clasificarse como procedimentales y no procedimentales.

En el lenguaje del tipo procedimental el usuario da las instrucciones al sistema para que realice una secuencia de operaciones en la base de datos para calcular el resultado deseado.

En el lenguaje no procedimental, el usuario describe la información deseada sin dar un procedimiento específico para obtener dicha información.

El álgebra relacional es un lenguaje de consulta formal procedimental, el álgebra relacional define operadores que funcionan sobre las tablas (de una manera similar a los operadores +, -, etc. del álgebra común) para llegar al resultado deseado. El álgebra relacional es difícil de utilizar, debido en parte a que es procedimental, esto es, al utilizar el álgebra relacional no sólo debemos saber lo que queremos, también cómo obtenerlo.

En el proceso de bases de datos comerciales el álgebra relacional se utiliza de manera poco frecuente. Aunque unos cuantos productos exitosos DBMS sí tienen opciones del álgebra relacional, éstas son poco utilizadas en vista de su complejidad.

El álgebra relacional toma dos o más tablas como entrada y produce una nueva tabla como resultado de la serie de operaciones. Las operaciones fundamentales en el álgebra relacional son seleccionar, proyectar, producto cartesiano, renombrar, unión y diferencia de conjuntos. Además de las operaciones fundamentales existen otras



operaciones como son: intersección de conjuntos, producto natural, división y asignación.

## 2. LENGUAJES DE CONSULTAS COMERCIALES

Un lenguaje de consulta comercial proporciona una interfaz más amigable al usuario. Un ejemplo de este tipo de lenguaje es el SQL, (Structured Query Language, Lenguaje de Consulta Estructurado).

Las partes más importantes del SQL son:

DDL: Lenguaje de definición de datos (que nos permite crear las estructuras)

DML: Lenguaje de manipulación de datos (que nos permite tener acceso a las estructuras para suprimir, modificar e insertar)

La estructura básica de una expresión en SQL contiene 3 partes, Select, From y Where.

La cláusula Select se usa para listar los atributos que se desean en el resultado de una consulta.

El From, lista las relaciones que se van a examinar en la evaluación de la expresión.

El Where, es la definición de las condiciones a las que puede estar sujeta una consulta.

La consulta típica de SQL tiene la siguiente forma:

```
Select A1,A2,A3...An  
From r1,r2,r3...rm  
Where Condición(es)
```

Donde:

A1,A2,A3...An: Representan a cada atributo(s) o campos de las tablas de la base de datos relacional.

r1,r2,r3...rm: Representan a la(s) tabla(s) involucradas en la consulta.

Condición: Es el enunciado que rige el resultado de la consulta.

Si se omite la cláusula Where, la condición es considerada como verdadera, la lista de atributos (A1,A2..An) puede sustituirse por un asterisco (\*), para seleccionar todos los atributos de todas las tablas que aparecen en la cláusula From.

### Funcionamiento del SQL

El SQL forma el producto cartesiano de las tablas involucradas en la cláusula From, cumpliendo con la condición establecida en la orden Where y después proyecta el resultado con la orden Select.

Para nuestros ejemplos consideremos una tabla llamada CURSO, que contiene los siguientes campos:

Nombre del campo	Descripción
NumC	Número del curso, único para identificar cada curso
NombreC	Nombre del curso, también es único
DescC	Descripción del curso
Creditos	Créditos, número de éstos que gana el estudiante al cursarlo
Costo	Costo del curso.
Depto	Departamento académico que ofrece el curso.

Datos contenidos en la tabla CURSO

NumC	NombreC	DescC	Creditos	Costo	Depto
A01	Liderazgo	Para público General	10	100.00	Admón.
S01	Introducción a la inteligencia artificial	Para ISC y LI	10	90.00	Sistemas.
C01	Construcción de torres	Para IC y Arquitectura	8	0.00	Ciencias
B01	Situación actual y perspectivas de la alimentación y la nutrición	Para IB	8	80.00	Bioquímica
E01	Historia presente y	IE e II	10	100.00	Electromecánica.

	futuro de la energía solar				
S02	Tecnología OLAP	Para ISC y LI	8	100.00	Sistemas
C02	Tecnología del concreto y de las Estructuras	Para IC	10	100.00	Ciencias
B02	Metabolismo de lípidos en el camarón	Para IB	10	0.00	Bioquímica
E02	Los sistemas eléctricos de potencia	Para IE	10	100.00	Electromecánica
S03	Estructura de datos	Para ISC y LI	8	0.00	Sistemas
A01	Diseño bioclimático	Para Arquitectura	10	0.00	Arquitectura
C03	Matemáticas discretas	General	8	0.00	Ciencias
S04	Circuitos digitales	Para ISC	10	0.00	Sistemas
S05	Arquitectura de Computadoras	Para ISC	10	50.00	Sistemas
I01	Base de Datos Relacionales	Para ISC y LI	10	150.00	Informática

### Ejemplos de consultas:

#### A. OBTENCIÓN DE UNA TABLA ENTERA

Obtener toda la información disponible sobre un curso donde Costo sea 0.

```
SELECT *
FROM CURSO
WHERE Costo=0.00
```

Resultado de la consulta anterior.

NumC	NombreC	DescC	Creditos	Costo	Depto
C01	Construcción de torres	Para IC y Arquitectura	8	0.00	Ciencias
B02	Metabolismo de lípidos en el camarón	Para IB	10	0.00	Bioquímica
S03	Estructura de datos	Para ISC y LI	8	0.00	Sistemas
A01	Diseño bioclimático	Para Arquitectura	10	0.00	Arquitectura
C03	Matemáticas discretas	General	8	0.00	Ciencias

Colocamos un \* debido a que no nos limitan la información de la tabla, es decir nos piden que mostremos todos los datos o atributos de la tabla CURSO.

Como la única condición en la sentencia WHERE es que la tarifa del curso sea igual a 0, esta consulta regresa todas las tuplas donde se encuentre que Costo = 0.00.

Debido a que Costo es un campo numérico, la condición solo puede comparar con campos del mismo tipo. Para representar valores negativos se antepone a la izquierda el signo (-), en este ejemplo se considera solo el signo (=) para establecer la condición, sin embargo otros operadores que se pueden utilizar son:

Menor que	<
Mayor que	>
Menor o igual que	<=
Mayor o igual que	>=
Diferente	<>

Además de los operadores booleanos AND, NOT, OR.

Cabe señalar que en la sentencia Where cuando se requiere establecer condiciones con cadenas, estas son delimitadas por apóstrofes ("). Las expresiones de cadenas son comparadas carácter por carácter, dos cadenas son iguales solo si coinciden todos los caracteres de las mismas.

## B. EJEMPLOS DE CONSULTAS CON CADENAS

Obtener toda la información sobre cualquier curso que ofrezca el departamento de Ciencias.

```
SELECT *
FROM CURSO
WHERE Depto = 'Ciencias';
```

Resultado de la consulta.

NumC	NombreC	DescC	Creditos	Costo	Depto
C01	Construcción de torres	Para IC y Arquitectura	8	0.00	Ciencias
C02	Tecnología del concreto y de las Estructuras	Para IC	10	100.00	Ciencias
C03	Matemáticas discretas	General	8	0.00	Ciencias

### C. VISUALIZACIÓN DE COLUMNAS ESPECIFICADAS

En los ejemplos anteriores obteníamos toda la tabla completa, ahora veremos como mostrar sólo algunos atributos específicos de una tabla.

Obtener los valores NumC, NombreC y Depto, en este orden de toda la tabla curso.

```
SELECT NumC, NombreC, Depto
FROM CURSO;
```

Resultado de la consulta:

NumC	NombreC	Depto
A01	Liderazgo	Admón.
S01	Introducción a la inteligencia artificial	Sistemas.
C01	Construcción de torres	Ciencias
B01	Situación actual y perspectivas de la alimentación y la nutrición	Bioquímica
E01	Historia presente y futuro de la energía solar	Electromecánica.

S02	Tecnología OLAP	Sistemas
C02	Tecnología del concreto y de las Estructuras	Ciencias
B02	Metabolismo de lípidos en el camarón	Bioquímica
E02	Los sistemas eléctricos de potencia	Electromecánica
S03	Estructura de datos	Sistemas
A01	Diseño bioclimático	Arquitectura
C03	Matemáticas discretas	Ciencias
S04	Circuitos digitales	Sistemas
S05	Arquitectura de Computadoras	Sistemas
I01	Base de Datos Relacionales	Informática

Observamos que en este caso no se tiene la sentencia Where, no existe condición, por lo tanto, todas las filas de la tabla CURSO se recuperan, pero sólo se visualizarán las tres columnas especificadas.

Así mismo, empleamos la (,) para separar los campos que deseamos visualizar.

#### D. VISUALIZACIÓN DE UN SUBCONJUNTO DE FILAS Y COLUMNAS

Seleccionar los valores NumC, Depto y Costo para todos los cursos que tengan un Costo inferior a \$100

```
SELECT NumC, Depto, Costo
FROM CURSO
WHERE Costo < 100.00
```

Como resultado de esta consulta se obtendrán todas aquellas tuplas que tengan en el campo Costo un valor menor que 100, y se visualizarán sólo los campos de NumC, Depto, Costo.

Podemos observar que este ejemplo cubre el formato general de una consulta SQL.

#### E. LA PALABRA CLAVE DISTINCT

DISTINCT, es una palabra reservada que elimina las filas duplicadas en el resultado de una consulta.

Visualizar todos los departamentos académicos que ofrezcan cursos, rechazando los valores duplicados.

```
SELECT DISTINCT Depto
FROM CURSO;
```

Resultado de la consulta

Depto
Administración
Sistemas
Ciencias
Bioquímica
electromecánica
Arquitectura
Informática

La palabra DISTINCT va estrictamente después de la palabra SELECT.

De no haberse utilizado la palabra DISTINCT, el resultado hubiera mostrado todas las tuplas del atributo Depto que se encontraran, es decir, se hubiera visualizado la columna de Depto completamente.

## F. EMPLEO DE LOS CONECTORES BOOLEANOS (AND, OR, NOT)

Para emplear las condiciones múltiples dentro de la sentencia WHERE, utilizamos los conectores lógicos.

El conector AND. Este conector pide al sistema que seleccione una sola columna únicamente si ambas condiciones se cumplen.

Obtener toda la información sobre todos los cursos que ofrece el departamento Sistemas que tengan una tarifa igual a 0.

```
SELECT *
FROM CURSO
WHERE Depto='Sistemas' AND Costo=0.00;
```

El resultado de esta consulta sería todas aquellas tuplas que cumplan exactamente con las dos condiciones establecidas.

El conector OR. Este conector al igual que el AND permite conectar condiciones múltiples en la sentencia WHERE, a diferencia del conector AND, el OR permite la selección de filas que cumplan con una sola de las condiciones establecidas a través de este conector.

Obtener toda la información existente sobre cualquier curso ofrecido por los departamentos Arquitectura o Bioquímica.

```
SELECT *  
FROM CURSO  
WHERE Depto = 'Arquitectura' OR Depto= 'Bioquímica';
```

El resultado de esta consulta será la de visualizar todas aquellas tuplas donde se cumpla cualquiera de las 2 condiciones, es decir mostrará todas las tuplas que tengan en el atributo Depto = Arquitectura o Bioquímica.

El conector NOT. Este nos permite marcar aquellas tuplas que por alguna razón no deseamos visualizar.

Obtener el nombre del curso y del departamento de todos los cursos que no sean ofrecidos por el departamento Sistemas.

```
SELECT NombreC, Depto  
FROM CURSO  
WHERE NOT (Depto='Sistemas');
```

## G. JERARQUIA DE OPERADORES BOOLEANOS

En orden descendente (de mayor a menor prioridad)

```
NOT  
AND  
OR
```

## H. FUNCIONES AVANZADAS APLICABLES A CONSULTAS

Existen funciones que permiten la agilización de consultas similares a una hoja de cálculo, ya que trabajan con base en renglones y columnas.

```
COUNT ( ): Cuenta el número de tuplas en la columna establecida  
MIN ( ): Localiza el valor mínimo de la columna establecida  
MAX ( ): Localiza el valor máximo de la columna establecida.  
AVG ( ): Obtiene el promedio de valores de la columna establecida
```



SUM ( ): Obtiene el valor total que implican los valores obtenidos en la columna establecida.

Para ejemplificar las instrucciones siguientes consideremos las siguientes tablas:

ALUMNO	CURSA	MATERIA,	que tienen los siguientes atributos:
NControl	NControl	Clave	
NombreA	Clave	NombreM	
Especialidad	Calif	Creditos	
Dirección			

*Ejemplos:*

Obtener el número de alumnos que existen en la carrera de Ingeniería en Sistemas Computacionales.

```
SELECT Count (*)
FROM Alumno
WHERE especialidad='ISC';
```

Obtener la máximo calificación que ha obtenido J.M. Cadena.

```
SELECT Max(Calif)
FROM Cursa
WHERE NControl IN (SELECT NControl
FROM Alumno
WHERE NombreA= 'J.M. Cadena ');
```

Obtener el promedio de calificaciones de Salvador Chávez.

```
SELECT Avg (Calif)
FROM Cursa
WHERE NCotrol IN (SELECT NControl
FROM Alumno
WHERE NombreA='Salvador Chávez');
```

Obtener la suma total de las calificaciones obtenidas por Daniel Colín.

```
SELECT Sum (Calif)
FROM Cursa
WHERE NControl IN (SELECT NControl
FROM Alumno
WHERE NombreA='Daniel Colín');
```

Hasta aquí hemos visto el manejo sencillo de realizar consultas con SQL, hay que destacar que en la realización de consultas anidadas se tiene que poner cuidado a la

prioridad de los operadores, teniendo cuidado también al momento de agrupar los paréntesis que involucran las condiciones con los operadores.

### 3. MODIFICACIÓN DE LA BASE DE DATOS

Como se mencionó al inicio de este apartado del SQL, éste cuenta con comandos DDL, para la definición de datos que nos permite crear o modificar la estructura de las tablas.

Las instrucciones para realizar estas operaciones son:

CREATE TABLE: Nos permite crear una tabla de datos vacía.  
CREATE INDEX: Crea un índice que nos puede auxiliar para las consultas.  
DROP TABLE: Permite borrar una tabla.  
DROP INDEX: Borra el índice indicado.

Instrucciones de DML que permiten insertar, borrar y actualizar registros de una tabla de la base de datos, tales como:

INSERT: Permite almacenar registros en una tabla creada.  
UPDATE: Permite modificar datos de registros almacenados en la tabla.  
DELETE: Borra un registro entero o grupo de registros de una tabla.

#### A. Estructura de la sentencia CREATE TABLE

```
CREATE TABLE <Nombre de la tabla>
(
  Atributo1: tipo de dato longitud,
  Atributo2: tipo de dato longitud,
  Atributo3: tipo de dato longitud,
  :
  :
  Atributon: tipo de dato longitud,
  PRIMARY KEY (Opcional) );
```

Los campos pueden definirse como NOT NULL de manera opcional excepto en la llave primaria para lo cual es obligatorio. Además al definir la llave primaria se genera automáticamente un índice con respecto al campo llave; para definir la llave la denotamos dentro de los paréntesis de PRIMARY KEY.

*Ejemplo:*

Crear la tabla alumno con los atributos antes descritos, tomando como llave el numero de control.

```
CREATE TABLE Alumno
(
NControl char(8) NOT NULL,
NombreA char(20),
Especialidad char(3),
Dirección char(30),
PRIMARY KEY (NControl) );
```

Tabla Alumno:

NControl	NombreA	Especialidad	Dirección

Puede existir tantos campos Not Null (No nulos) como se requieran. En si estructurar la creación de una tabla es siempre parecida al ejemplo anterior.

## B. Estructura de la sentencia INSERT

```
INSERT
INTO Nombre de la tabla a la que se le va a insertar el registro
VALUES (Conjunto de valores del registro) ;
```

*Ejemplo:*

Insertar en la tabla Alumno, antes creada los datos del alumno Daniel Colín, con número de control 95310518 de la especialidad de Ingeniería civil, con domicilio Abasolo Norte #45.

```
INSERT
INTO Alumno
VALUES("95310518","Daniel Colín","IC","Abasolo Norte #45") ;
```

Nótese que la inserción de los datos se realiza conforme la estructura que se implantó en la tabla, es decir en el orden en que se creó dicha tabla. En caso de querer omitir un dato que no sea no nulo solamente se ponen las comillas indicando el vacío de la cadena.

## C. Estructura de la Sentencia CREATE INDEX

```
CREATE INDEX Nombre que se le asignará al índice.
ON Nombre de la tabla a la cual se le crea el índice (Campo(s) por el cual se
crea el índice);
```

*Ejemplo:*

Crear un índice de la tabla Alumno por el campo Especialidad.

```
CREATE INDEX Indice1  
ON Alumno(Especialidad);
```

Este índice contendrá a todos los alumnos ordenados por el campo especialidad.

```
CREATE INDEX UNIQUE INDEX Indice2  
ON Alumno (Especialidad);
```

En la creación de este índice utilizamos la sentencia UNIQUE, es un indicador para permitir que se cree un índice único por especialidad, esta sentencia siempre se coloca antes de CREATE INDEX. En este ejemplo se creará un índice que contenga un alumno por especialidad existente.

#### **D. Estructura de la sentencia UPDATE**

```
UPDATE Nombre de la tabla en donde se modificaran los datos  
SET Valores  
WHERE (Condición);
```

*Ejemplo:*

Modificar el número de control del registro de Daniel Colín de la tabla Alumno por el número 96310518.

```
UPDATE Alumno  
SET NControl '96310518'  
WHERE NombreA='Daniel Colín';
```

#### **E. Estructura de la sentencia DROP TABLE**

```
DROP TABLE Nombre de la tabla a borrar ;
```

*Ejemplo:*

Borrar la tabla Alumno creada anteriormente.

```
DROP TABLE Alumno;
```

#### **F. Estructura de la sentencia DROP INDEX**

```
DROP INDEX Nombre del índice a borrar;
```

*Ejemplo:*

Borrar el índice Indice1 creado anteriormente.

```
DROP INDEX Indice1;
```

### G. Estructura de la sentencia DELETE

```
DELETE  
FROM Nombre de la tabla  
WHERE Condición;
```

*Ejemplos:*

Borrar el registro cuyo número de control es 95310386.

```
DELETE  
FROM Alumno  
WHERE Control='95310386';
```

Borrar todos los registros de la tabla alumno.

```
DELETE  
FROM Alumno;
```

En el primer ejemplo, se borrará todo el registro (todos los datos), del alumno con número de control = 95310386.

En el segundo ejemplo se borrarán todos los registros de la tabla alumno, pero sin borrar la estructura de la tabla, ya que la orden Delete solo borra registros; la sentencia Drop Table borra toda la estructura de la tabla junto con los registros de la misma.

## 4. VISTAS

Una vista se define en SQL usando la orden CREATE VIEW. Para definir una vista debemos dar a la vista un nombre y declarar la consulta que calcula la vista. Una vez que establecemos una vista, podemos ejecutar una sentencia SELECT que referencie a esa vista. El sistema asociará la vista SQL con una tabla base y extraerá y visualizará, entonces, los datos de la tabla base.

Esto significa que una vista no contiene datos duplicados de una tabla base. No tiene absolutamente ningún dato, puesto que no es una tabla real, todo el proceso se realiza con los datos almacenados en la tabla base, es decir, se percibe como una tabla virtual.

Las órdenes que se utilizan para la manipulación de vistas son:

CREATE VIEW: Crea una tabla virtual.

DROP VIEW: Elimina una vista creada anteriormente.

**A. Estructura de la sentencia CREATE VIEW.**

CREATE VIEW Nombre de la vista AS (Expresión de consulta);

Para nuestros ejemplos consideremos de nuevo la tabla llamada CURSO, que contiene los siguientes campos:

Nombre del campo	Descripción
NumC	Número del curso, único para identificar cada curso
NombreC	Nombre del curso, también es único
DescC	Descripción del curso
Creditos	Créditos, número de estos que gana al estudiante al cursarlo
Costo	Costo del curso.
Depto	Departamento académico que ofrece el curso.

Que contiene los siguientes datos:

NumC	NombreC	DescC	Creditos	Costo	Depto
A01	Liderazgo	Para público General	10	100.00	Admón.
S01	Introducción a la inteligencia artificial	Para ISC y LI	10	90.00	Sistemas.
C01	Construcción de torres	Para IC y Arquitectura	8	0.00	Ciencias
B01	Situación actual y perspectivas de la alimentación y la nutrición	Para IB	8	80.00	Bioquímica
E01	Historia presente y futuro de la energía solar	IE e II	10	100.00	Electromecánica.

S02	Tecnología OLAP	Para ISC y LI	8	100.00	Sistemas
C02	Tecnología del concreto y de las Estructuras	Para IC	10	100.00	Ciencias
B02	Metabolismo de lípidos en el camarón	Para IB	10	0.00	Bioquímica
E02	Los sistemas eléctricos de potencia	Para IE	10	100.00	Electromecánica
S03	Estructura de datos	Para ISC y LI	8	0.00	Sistemas
A01	Diseño bioclimático	Para Arquitectura	10	0.00	Arquitectura
C03	Matemáticas discretas	General	8	0.00	Ciencias
S04	Circuitos digitales	Para ISC	10	0.00	Sistemas
S05	Arquitectura de Computadoras	Para ISC	10	50.00	Sistemas
I01	Base de Datos Relacionales	Para ISC y LI	10	150.00	Informática

*Ejemplos:*

Crear una vista (tabla virtual), denominada CursosS, que contenga sólo las filas correspondientes a cursos ofrecidos por el departamento Sistemas. La vista deberá contener todas las columnas de la tabla CURSO, con la excepción de la columna Depto, la secuencia, de izquierda a derecha de las columnas, deberá ser: NombreC, NumC, Creditos, Costo y DescC.

```
CREATE VIEW CursosS AS
SELECT NombreC, NumC, Creditos, Costo, DescC
FROM CURSO
WHERE DescC='Sistemas';
```

Observemos que después del nombre de la vista ponemos la sentencia AS, esto para definir la estructura de la vista, la estructura en sí de la vista está formada por la consulta anteriormente vista utilizando la orden SELECT.

Crear una vista denominada CursosCaros, correspondientes a las filas de la tabla CURSO, en donde la tarifa exceda de \$150; las columnas de la vista deberán tener los nombres ClaveCurso, NombreCurso y CostoCaro.

```
CREATE VIEW
CursosSCaros(ClaveCurso, NombreCurso, CostoCaro) As
SELECT NumC, NombreC, Costo
FROM Curso
WHERE Costo > 150;
```

Observamos que después del nombre de la vista CursosCaros ponemos los nombres de los campos que se nos pidieron que tuviera la vista(ClaveCurso,...), después se realiza la consulta correspondiente para generar el resultado deseado.

#### **B. Visualizar las vistas**

Creamos una tabla virtual que contiene los datos de las consultas que deseamos, ahora nos falta visualizar estos datos, para ello utilizamos la sentencia SELECT y realizamos la consulta:

```
SELECT *
FROM CursosCaros;
```

De esta consulta podemos observar que mostramos todos los campos que la vista contiene, aunque podemos visualizar sólo algunos de ellos; también observamos que sustituimos el nombre de la tabla por el de la vista junto a la sentencia FROM, esto es por que una vista es una tabla virtual, pero guarda los datos como cualquier tabla normal.

#### **C. Eliminar una vista**

Como si fuera una tabla normal, las vistas también pueden borrarse, para ello utilizamos la sentencia DROP VIEW.

Estructura de la sentencia DROP VIEW.

DROP VIEW Nombre de la vista a borrar;

*Ejemplo:*

Borrar la vista CursosCaros creada anteriormente.

```
DROP VIEW CursosCaros;
```



## DISEÑO DE BASES DE DATOS RELACIONALES

Uno de los retos en el diseño de la base de datos relacional es el de obtener una estructura estable y lógica tal que:

- El sistema de base de datos no sufra de anomalías de almacenamiento.
- El modelo lógico pueda modificarse fácilmente para admitir nuevos requerimientos.

Una base de datos implantada sobre un modelo bien diseñado tiene mayor esperanza de vida aun en un ambiente dinámico, que una base de datos con un diseño pobre. En promedio, una base de datos experimenta una reorganización general cada seis años, dependiendo de lo dinámico de los requerimientos de los usuarios. Una base de datos bien diseñada tendrá un buen desempeño aunque aumente su tamaño, y será lo suficientemente flexible para incorporar nuevos requerimientos o características adicionales.

Existen diversos riesgos en el diseño de las bases de datos relacionales que afectan la funcionalidad de la misma, los riesgos generalmente son la redundancia de información y la inconsistencia de datos.

## NORMALIZACIÓN

La normalización es el proceso de simplificar la relación entre los campos de un registro.

Por medio de la normalización un conjunto de datos en un registro se reemplaza por varios registros que son más simples y predecibles y, por lo tanto, más manejables. La normalización se lleva a cabo por cuatro razones:

- Estructurar los datos de forma que se puedan representar las relaciones pertinentes entre los datos.
- Permitir la recuperación sencilla de los datos en respuesta a las solicitudes de consultas y reportes.
- Simplificar el mantenimiento de los datos actualizándolos, insertándolos y borrándolos.
- Reducir la necesidad de reestructurar o reorganizar los datos cuando surjan nuevas aplicaciones.

En términos más sencillos la normalización trata de simplificar el diseño de una base de datos, ésto a través de la búsqueda de la mejor estructuración que pueda utilizarse con las entidades involucradas en ella.

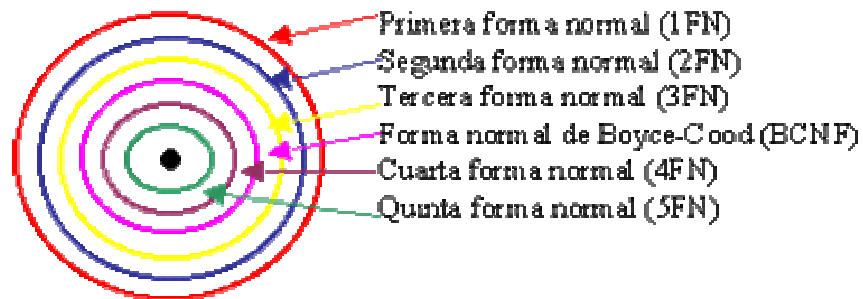
### 1. Pasos de la normalización

- Descomponer todos los grupos de datos en registros bidimensionales.
- Eliminar todas las relaciones en la que los datos no dependan completamente de la llave primaria del registro.
- Eliminar todas las relaciones que contengan dependencias transitivas.

La teoría de normalización tiene como fundamento el concepto de formas normales; se dice que una relación está en una determinada forma normal si satisface un conjunto de restricciones.

En resumen podemos decir que la normalización son las técnicas para prevenir las anomalías en las tablas. Dependiendo de su estructura, una tabla puede estar en primera forma normal, segunda forma normal o en cualquier otra.

Relación entre las formas normales:



#### a. Primera forma normal

*Definición formal:*

Una relación R se encuentra en 1FN si y sólo si por cada renglón columna contiene valores atómicos.

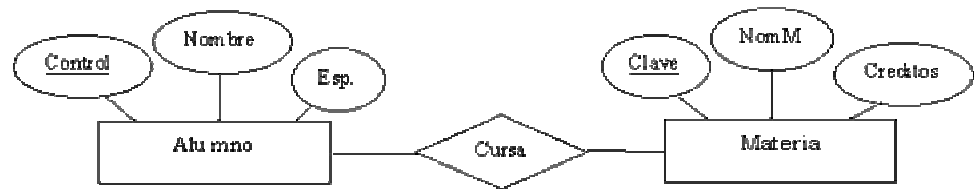
Abreviada como 1FN, se considera que una relación se encuentra en la primera forma normal cuando cumple lo siguiente:

- Las celdas de las tablas poseen valores simples y no se permiten grupos ni arreglos repetidos como valores, es decir, contienen un solo valor por cada celda.
- Todos los ingresos en cualquier columna (atributo) deben ser del mismo tipo.
- Cada columna debe tener un nombre único, el orden de las columnas en la tabla no es importante.

Dos filas o renglones de una misma tabla no deben ser idénticas, aunque el orden de las filas no es importante.

Por lo general la mayoría de las relaciones cumplen con estas características, así que podemos decir que la mayoría de las relaciones se encuentran en la primera forma normal.

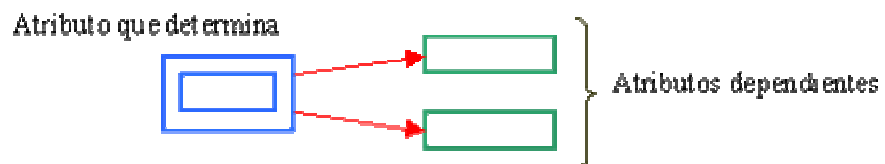
Para ejemplificar como se representan gráficamente las relaciones en primera forma normal consideremos la relación alumno cursa materia cuyo diagrama E-R es el siguiente:



Como esta relación maneja valores atómicos, es decir un solo valor por cada uno de los campos que conforman a los atributos de las entidades, ya se encuentra en primera forma normal, gráficamente así representamos a las relaciones en 1FN.

**b. Segunda forma normal**

Para definir formalmente la segunda forma normal requerimos saber que es una dependencia funcional. Consiste en identificar que atributos dependen de otro(s) atributo(s).

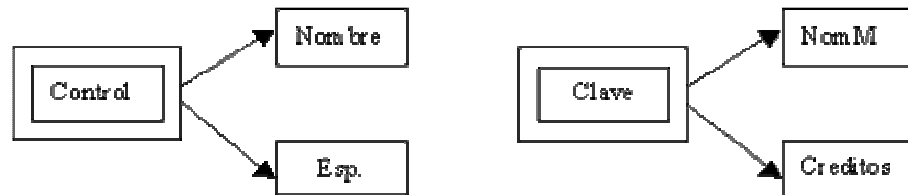


*Definición formal:*

Una relación R está en 2FN si y sólo si está en 1FN y los atributos no primos dependen funcionalmente de la llave primaria.

Una relación se encuentra en segunda forma normal, cuando cumple con las reglas de la primera forma normal y todos sus atributos que no son claves (llaves) dependen por completo de la clave. De acuerdo con esta definición, cada tabla que tiene un atributo único como clave, está en segunda forma normal.

La segunda forma normal se representa por dependencias funcionales como:



Nótese que las llaves primarias están representadas con doble cuadro, las flechas nos indican que de estos atributos se puede referenciar a los otros atributos que dependen funcionalmente de la llave primaria.

### c. Tercera Forma Normal

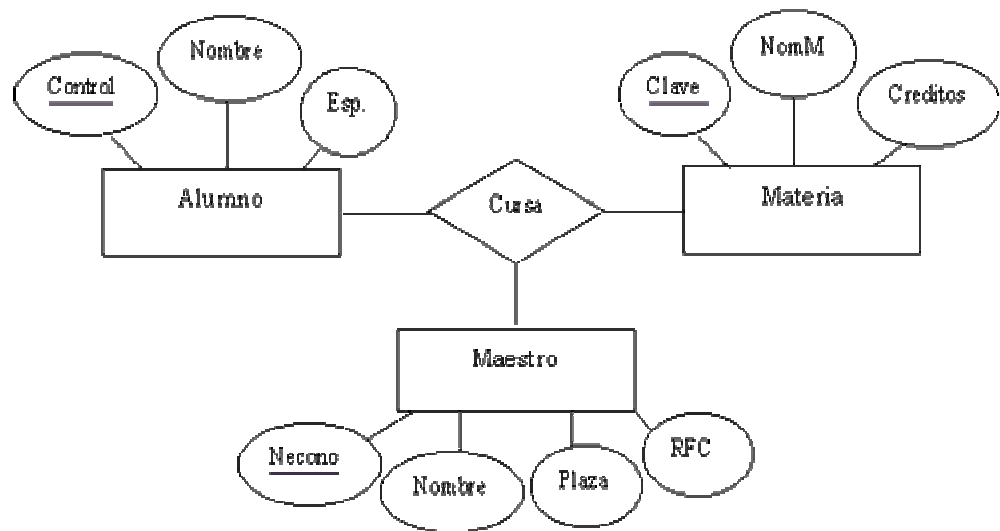
Para definir formalmente la 3FN necesitamos definir dependencia transitiva. En una afinidad (tabla bidimensional) que tiene por lo menos 3 atributos (A,B,C) en donde A determina a B, B determina a C pero no determina a A.

*Definición formal:*

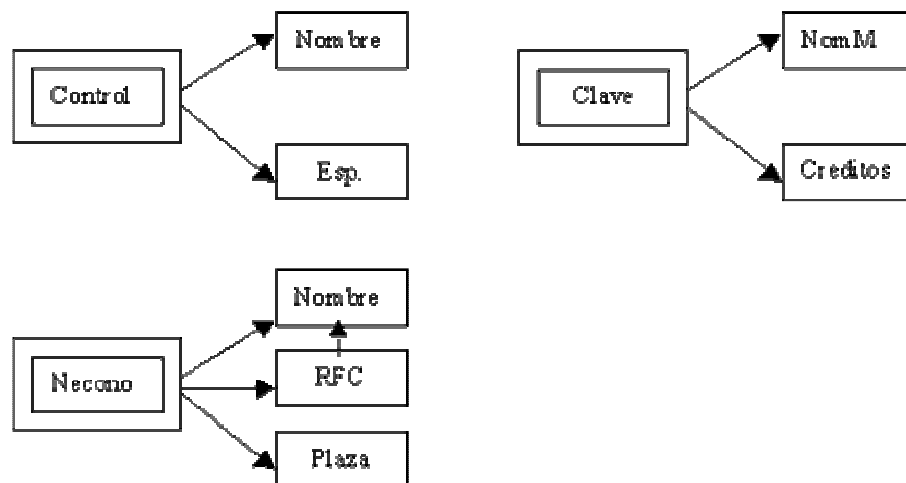
Una relación R está en 3FN si y solo si esta en 2FN y todos sus atributos no primos dependen no transitivamente de la llave primaria.

Consiste en eliminar la dependencia transitiva que queda en una segunda forma normal; en pocas palabras una relación está en tercera forma normal si está en segunda forma normal y no existen dependencias transitivas entre los atributos; nos referimos a dependencias transitivas cuando existe más de una forma de llegar a referenciar a un atributo de una relación.

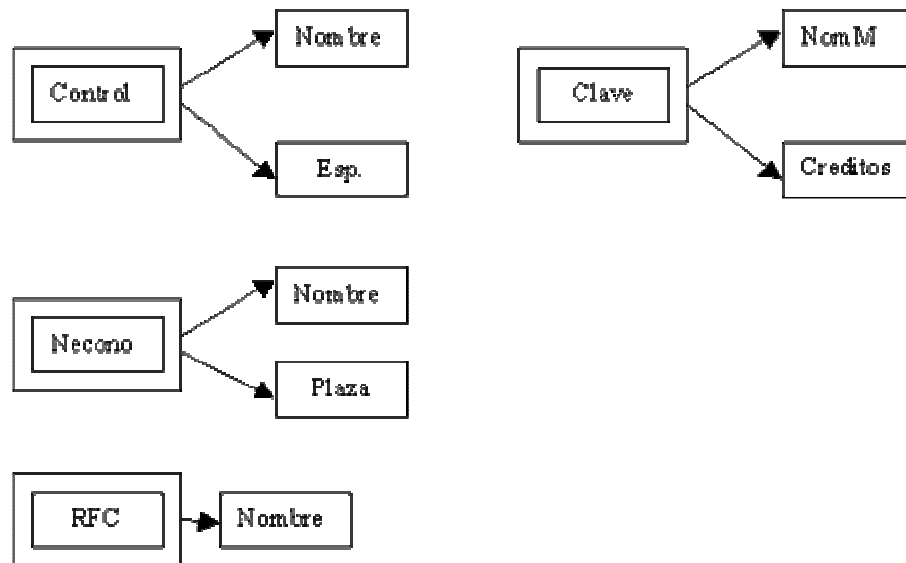
Por ejemplo, consideremos el siguiente caso:



Tenemos la relación alumno-cursa-materia manejada anteriormente, pero ahora consideramos al elemento maestro, gráficamente lo podemos representar de la siguiente manera:



Podemos darnos cuenta que se encuentra graficado en segunda forma normal, es decir que todos los atributos llave están indicados en doble cuadro indicando los atributos que dependen de dichas llaves, sin embargo en la llave Necono tiene como dependientes a 3 atributos en el cual el nombre puede ser referenciado por dos atributos: Necono y RFC (Existe dependencia transitiva), podemos solucionar esto aplicando la tercera forma normal que consiste en eliminar estas dependencias separando los atributos, entonces tenemos:



### ACTIVIDAD...

1. *Elaborar el modelo relacional para el ejercicio planteado en la unidad anterior (Acuario)*
2. *Normalizar la base de datos anterior*

### ESTRATEGIAS METODOLÓGICAS

- Presentación de la unidad a cargo del profesor.
- Desarrollo del modelo Relacional aplicado a un caso específico a cargo del docente.
- Realización de ejercicios de normalización a cargo del docente

**RECURSOS**

- Humanos: Profesor y alumnos
- Institucionales: Salón de clase
- Materiales: Texto guía

**INDICADORES DE EVALUACIÓN**

- Evaluación escrita sobre la unidad
- Evaluación de las actividades propuestas en la unidad.

**CRITERIOS DE EVALUACIÓN**

- ¿Qué importancia tiene el modelo Relacional?
- ¿Cuáles son las diferentes cláusulas para un lenguaje de consulta SQL ?
- ¿Cuáles son los pasos de normalización?

## UNIDAD 4

**OTROS MODELOS DE BASES DE DATOS****INTRODUCCIÓN**

Esta unidad busca que el estudiante distinga los conceptos asociados a otros modelos de Bases de Datos, tales como: Red, Jerárquicos y Orientados a Objetos, reconociendo sus principales características, que le permitan diferenciar a uno de otro.

**JUSTIFICACIÓN**

Los Modelos de Bases de Datos en Red, Jerárquicos y Orientados a Objetos, no corresponden al objetivo de este curso, pero su existencia sigue dándose (Red y Jerárquicos), o están en el inicio de su apogeo (Orientados a Objetos), por tal motivo, es indispensable que el alumno conozca de su existencia y pueda diferenciarlos.

**OBJETIVO GENERAL**

Identificar otros modelos de Bases de Datos

**OBJETIVOS ESPECÍFICOS**

- Reconocer la estructura de una base de datos en red.
- Reconocer la estructura de una base de datos jerárquica
- Reconocer la estructura de una base de datos orientada a objetos



<b>CONTENIDO</b>	
<ol style="list-style-type: none"><li>1. Modelo de Bases de datos de red</li><li>2. Modelo de Bases de datos Jerárquica</li><li>3. Modelo de Bases de datos orientadas a Objetos</li></ol>	

## MODELO DE BASES DE DATOS DE RED

### **Conceptos básicos.**

Una base de datos de red como su nombre lo indica, está formado por una colección de registros, los cuales están conectados entre sí por medio de enlaces. El registro es similar a una entidad como las empleadas en el modelo entidad-relación.

Un registro es una colección de campos (atributos), cada uno de los cuales contiene un solo valor; un enlace es la asociación entre dos registros exclusivamente, así que podemos verla como una relación estrictamente binaria.

Una estructura de datos de red, llamada algunas veces estructura plex, abarca más que la estructura de árbol porque un nodo hijo en la estructura de red puede tener más de un padre. En otras palabras, la restricción de que en un árbol jerárquico cada hijo puede tener un solo padre, se hace menos severa. Así, la estructura de árbol se puede considerar como un caso especial de la estructura de red.

### **Diagramas de estructura de datos.**

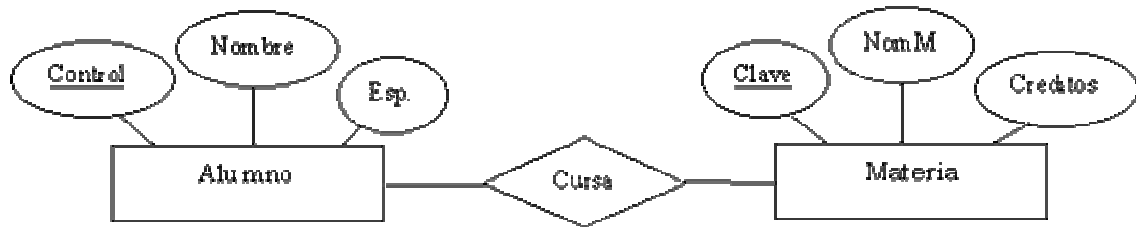
Un diagrama de estructura de datos es un esquema que representa el diseño de una base de datos de red. Este modelo se basa en la representación de relaciones entre registros por medio de ligas. Existen relaciones en las que participan sólo dos entidades (binarias) y relaciones en las que participan más de dos entidades (generales) ya sea con o sin atributo descriptivo en la relación.

Los diagramas constan de dos componentes básicos:

- Celdas: representan a los campos del registro.
- Líneas: representan a los enlaces entre los registros.

Un diagrama de estructura de datos de red, especifica la estructura lógica global de la base de datos; su representación gráfica se basa en el acomodo de los campos de un registro en un conjunto de celdas que se ligan con otro(s) registro(s). Un ejemplo de esto sería el siguiente:

Consideremos la relación alumno-curso-materia donde la relación curso no tiene atributos descriptivos:



Las estructuras de datos según la cardinalidad se representan en los siguientes casos: Cuando el enlace no tiene atributos descriptivos

**Caso 1. Cardinalidad Uno a Uno.**



Diagrama de estructura de datos

**Caso 2. Cardinalidad Uno a Muchos.**



Diagrama de estructura de datos

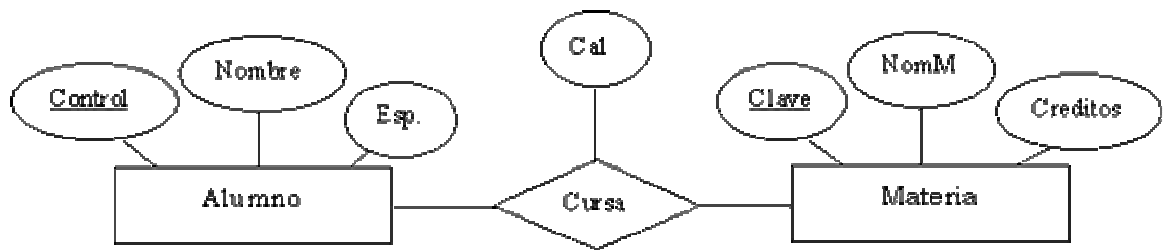
**Caso 3. Cardinalidad Muchos a muchos.**



Diagrama de estructura de datos

Cuando el enlace tiene atributos descriptivos.

Consideremos que a la relación cursa le agregamos el atributo Cal (calificación), nuestro modelo E-R quedaría de la siguiente manera:



La forma de convertir a diagramas de estructura de datos consiste en realizar lo siguiente:

- Sustituir las entidades alumno y materia por los tipos de registro correspondientes.
- Crear nuevo registro, denominado Calif, para este caso, con un solo campo, el de cal (calif).
- Crear los siguientes enlaces muchos a uno:

AluCal, del registro Calif al registro Alumno.  
 MatCal, del registro Calif al registro Materia.

AluCal y MatCal son sólo los nombres que emplearemos para identificar el enlace; pueden ser otros si no son empleados para otra cosa.

El diagrama de estructura de datos resultante es:

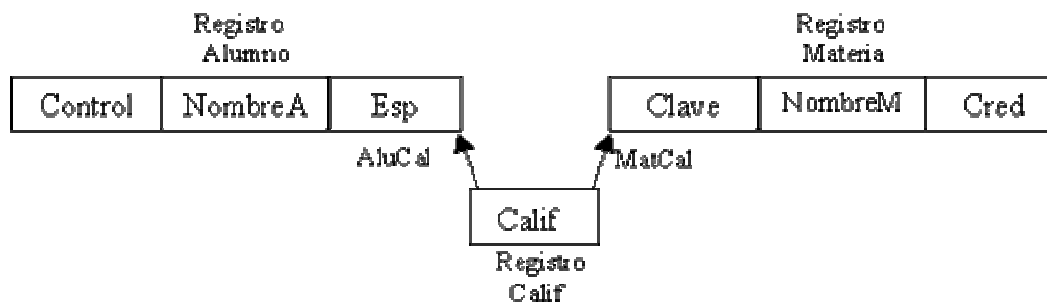
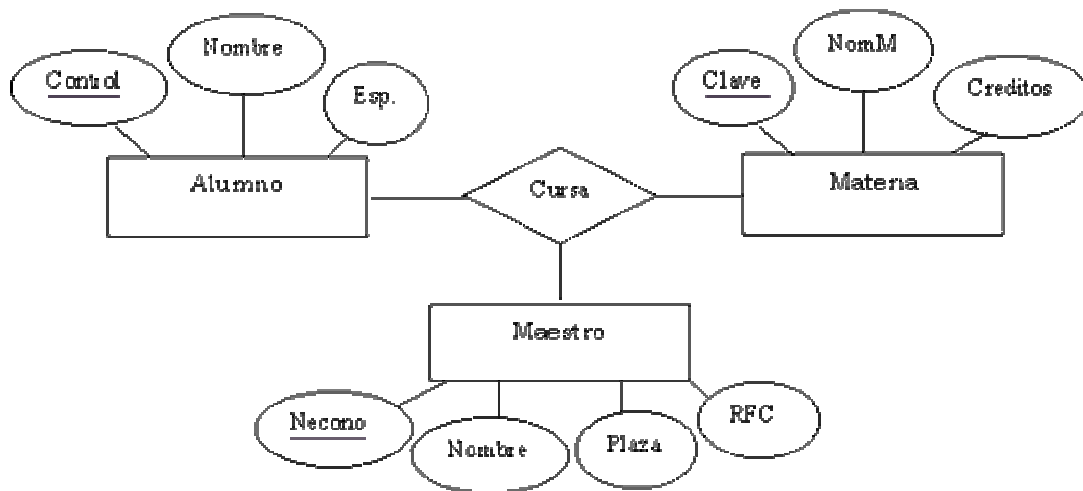


Diagrama de estructura de datos

Diagramas de estructura de datos cuando intervienen más de dos entidades y el enlace no tiene atributos descriptivos.

Consideremos que a la relación alumno-cursa-materia le agregamos la entidad maestro, quien es el que imparte dicha materia.

Nuestro diagrama E-R quedaría de la siguiente manera:



La transformación a diagramas de estructura de datos se realiza mediante los siguientes pasos:

- Crear los respectivos registros para cada una de las entidades que intervienen en el modelo.
- Crear un nuevo tipo de registro que llamaremos Reenlace, que puede no tener campos o tener sólo uno que contenga un identificador único; el identificador lo proporcionará el sistema y no lo utiliza directamente el programa de aplicación. A este registro se le denomina también como registro ficticio o de enlace o de unión.
- Crear los siguientes enlaces del tipo muchos a uno:

ARenl del tipo de registro Reenlace al tipo de registro Alumno.  
 MaRenl del tipo de registro Reenlace al tipo de registro Maestro.  
 MRenl del tipo de registro Reenlace al tipo de registro Materia.

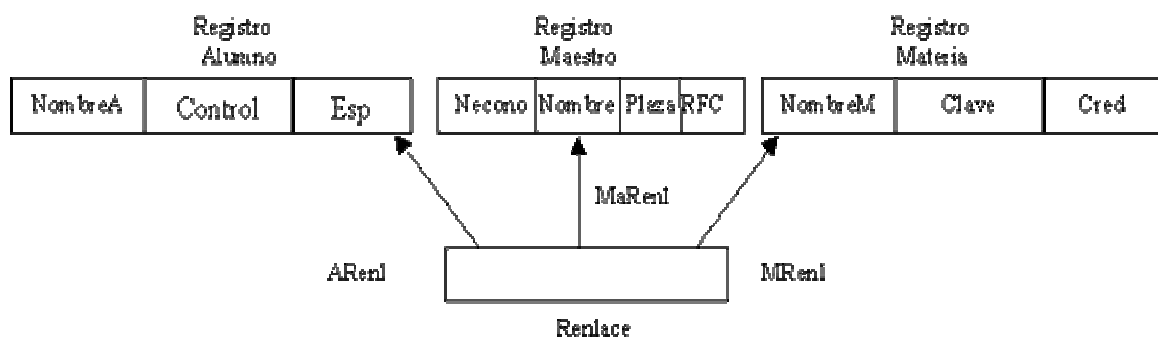


Diagrama de estructura de datos

Ahora si nuestro enlace tuviera atributos descriptivos, se crea el registro con los campos respectivos y se liga indicando el tipo de cardinalidad.

En este caso tomamos el ejemplo anterior y le agregamos a la relación el atributo calif. (Calificación).

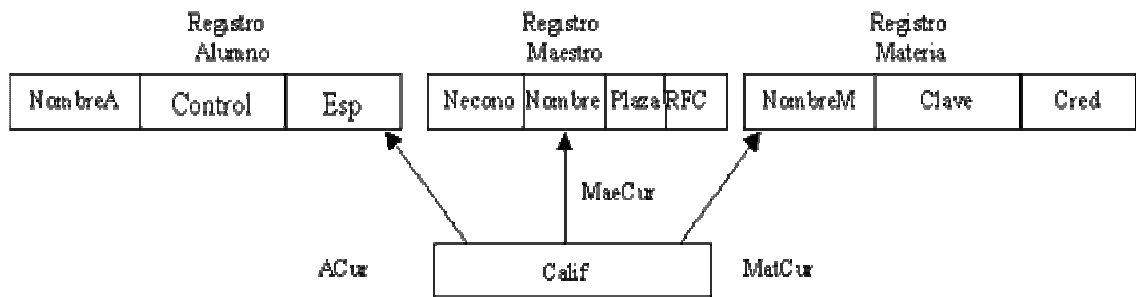
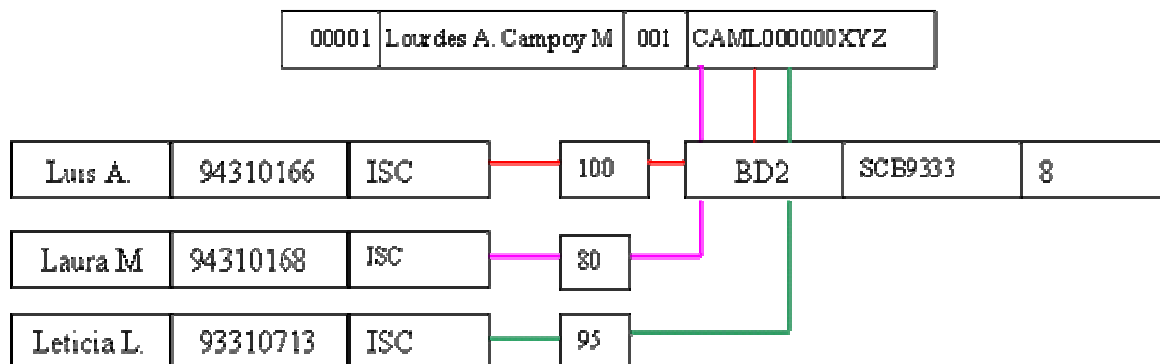


Diagrama de estructura de datos

Considerando el anterior diagrama de estructura de datos, una instancia de este sería:



Este diagrama nos indica que los alumnos Luís A., Laura M. y Leticia L. cursaron la materia Base de datos 2 con la maestra Ing. Lourdes A. Campoy M. obteniendo una calificación de 100, 80,95 respectivamente.

## ACTIVIDAD...

Responder al siguiente cuestionario

1 La base de datos de red puede llamarse algunas veces como:

- A. Una estructura plex.
- B. Un árbol.
- C. Una relación de tablas bidimensionales

- 2 *Una base de datos de red está formada por una colección de:*
  - A. *Nodos*
  - B. *Matrices*
  - C. *Registros*
  
- 3 *Una forma de representar el diseño de una base de datos de red es :*
  - A. *Una colección de información.*
  - B. *Un diagrama de estructura de datos.*
  - C. *La estructura grafica.*
  
- 4 *Este modelo de base de datos esta basado en:*
  - A. *Representaciones entre registros por medio de ligas.*
  - B. *Representaciones entre registros por medio binario.*
  - C. *Representaciones entre registros por medio de encuestas.*
  
- 5 *La forma de representar el diagramado de una bases de datos de red es*
  - A. *Líneas y líneas*
  - B. *Celdas y líneas.*
  - C. *Gráfico y líneas*

## MODELO DE DATOS JERÁRQUICO

### Conceptos Básicos.

Una base de datos jerárquica consiste en una colección de registros que se conectan entre sí por medio de enlaces. Los registros son similares a los expuestos en el modelo de red que se estructura como un grafo.

Cada registro es una colección de campos (atributos), que contienen un solo valor cada uno de ellos. Un enlace es una asociación o unión entre dos registros exclusivamente. Por tanto, este concepto es similar al de enlace para modelos de red, con la diferencia de que el hijo depende del padre estructuralmente hablando.

Consideremos la base de datos, nuevamente, que contiene la relación alumno - materia de un sistema escolar. Existen dos tipos de registros en este sistema, alumno y materia. El registro Alumno consta de tres campos: NombreA, Control y Esp; el registro Materia esta compuesto de tres campos: Clave, NombreM y Cred.

En este tipo de modelos la organización se establece en forma de árbol, donde la raíz es un nodo ficticio (se denominaría actividad) que es lo que requieren estos registros. Así tenemos que, una base de datos jerárquica es una colección de árboles de este tipo.

El contenido de un registro específico puede repetirse en varios sitios (en el mismo árbol o en varios árboles).

La repetición de los registros tiene dos desventajas principales:

- Puede producirse una inconsistencia de datos.
- El desperdicio de espacio.

### Diagramas de estructura de árbol

Un diagrama de estructura de árbol es la representación de un esquema de la base de datos jerárquica, de ahí el nombre, ya que un árbol está desarrollado precisamente en orden descendente formando una estructura jerárquica.

Este tipo de diagrama está formado por dos componentes básicos:

- Rectángulos: que representan a los registros.
- Líneas: que representan a los enlaces o ligas entre los registros.

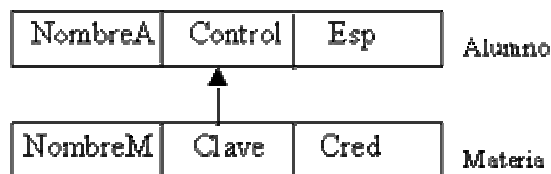
Un diagrama de árbol tiene el propósito de especificar la estructura lógica global de la base de datos.

Un diagrama de estructura de árbol es similar a un diagrama de estructura de datos en el modelo de red. La principal diferencia es que en el modelo de red los registros se organizan en forma de un grafo arbitrario, mientras que en el modelo de estructura de árbol los registros se organizan en forma de un árbol con raíz.

### Características de las estructuras de árbol:

El árbol no puede contener ciclos.

Las relaciones que existen en la estructura deben ser de tal forma que sólo existan relaciones uno a muchos o uno a uno entre un padre y un hijo.

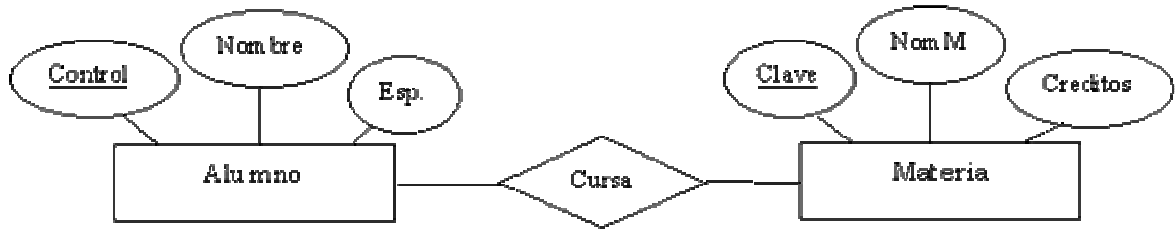


En este diagrama podemos observar que las flechas están apuntando de padres a hijos. Un padre (origen de una rama) puede tener una flecha apuntando a un hijo, pero un hijo siempre puede tener una flecha apuntando a su padre.

El esquema de una base de datos se representa como una colección de diagramas de estructura de árbol. Para cada diagrama existe una única instancia de árbol de base de datos. La raíz de este árbol es un nodo ficticio. Los hijos de ese nodo son instancias de los registros de la base de datos. Cada una de las instancias que son hijos pueden tener a su vez, varias instancias de varios registros.

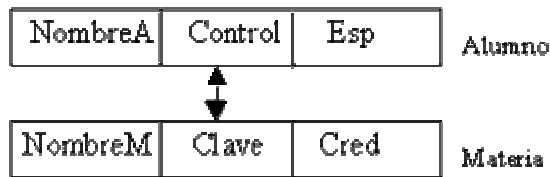
Las representaciones según las cardinalidades son:

Consideremos la relación alumno-materia sin atributo descriptivo.

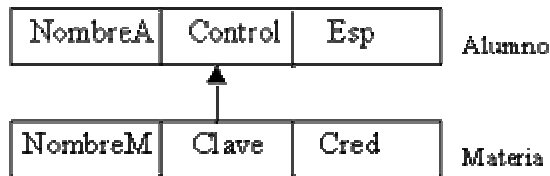


La transformación según las cardinalidades sería:

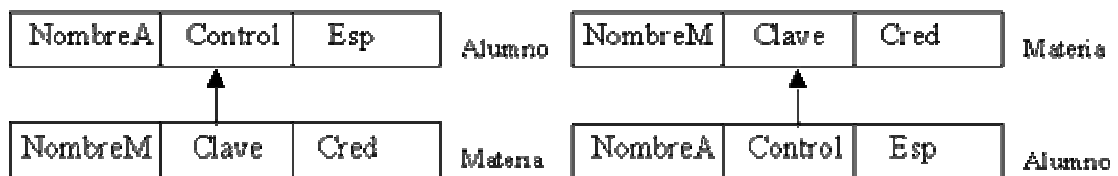
Cuando la relación es uno a uno.



Cuando la relación es uno a muchos.



Cuando la relación es muchos a muchos.



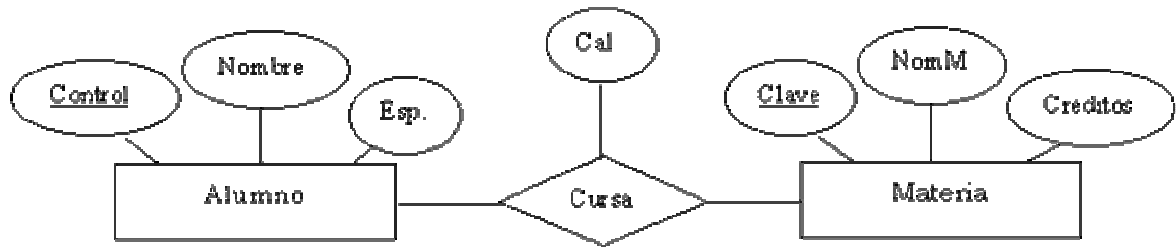
Cuando la relación tiene atributos descriptivos, la transformación de un diagrama E-R a estructura de árbol se lleva a cabo cubriendo los siguientes pasos:



- Crear un nuevo tipo de registro.
- Crear los enlaces correspondientes.

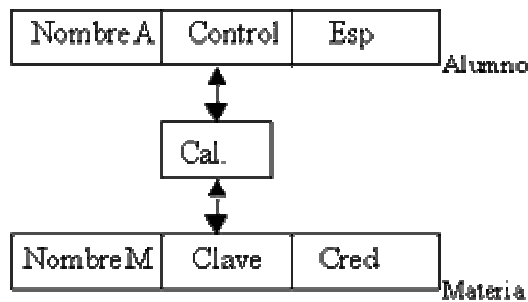
Consideremos que a la relación Alumno-Materia añadimos el atributo Cal a la relación que existe entre ambas, entonces nuestro modelo E-R resulta:

Añadir el diagrama E-R

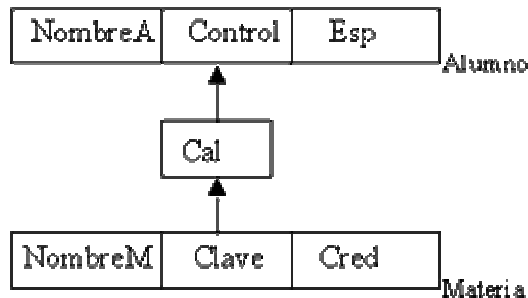


Según las cardinalidades los diagramas de estructura de árbol pueden quedar de la siguiente manera:

Cuando la relación es uno a uno.



Cuando la relación es uno a muchos.



Cuando la relación es Muchos a Muchos.

Si la relación es muchos a muchos entonces la transformación a diagramas de árbol es un poco más compleja debido a que el modelo jerárquico solo puede representar relaciones uno a uno o uno a muchos.

Existen varias formas distintas de transformar este tipo de relaciones a estructura de árbol, sin embargo todas las formas constituyen la repetición de algunos registros.

La decisión de qué método de transformación debe utilizarse depende de muchos factores, entre los que se incluyen:

- El tipo de consultas esperadas en la base de datos.
- El grado al que el esquema global de base de datos que se está modelando se ajusta al diagrama E-R dado.

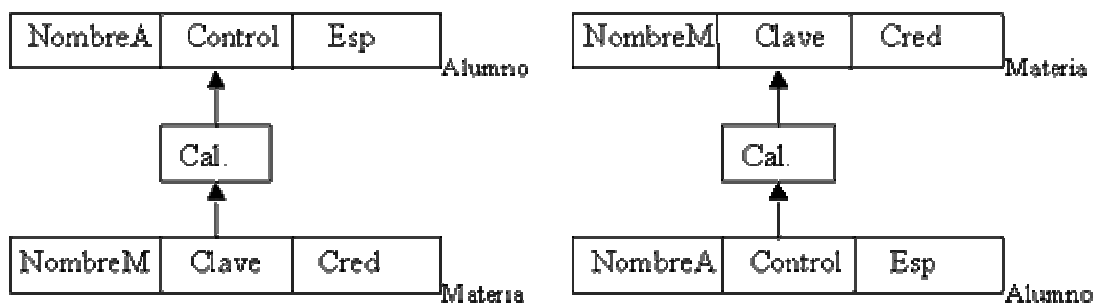
A continuación se describe la forma de transformar un diagrama E-R a estructura de árbol con relaciones muchos a muchos. Suponemos el ejemplo de la relación alumno-materia.

Crear dos diagramas de estructura de árbol distintos T1 y T2, cada uno de los cuales incluye los tipos de registro alumno y materia, en el árbol T1 la raíz es alumno y en T2 la raíz es materia.

Crear los siguientes enlaces:

- Un enlace muchos a uno del registro materia al registro Alumno, en T1
- Un enlace muchos a uno del tipo de registro alumno al tipo de registro materia en T2.

Como se muestra en el siguiente diagrama:



### ACTIVIDAD...

*Responder al siguiente cuestionario*

- 1 *Como se puede representar una base de datos jerárquica:*

- A. *Con una entidad.*
- B. *Con un árbol.*
- C. *Con un enlace*

2 *Una definición de base de datos jerárquica es:*

- A. *Una colección de registros que se conectan entre sí por medio de una estructura de árbol.*
- B. *Es una serie de datos en un instante de la base de datos.*
- C. *Una base de datos normalizada.*

3 *Una base de datos jerárquica indica que sus múltiples ocurrencias se representan en forma de árbol:*

- A. *Un hijo debe existir antes del padre.*
- B. *Un padre debe existir antes del hijo.*
- C. *Un padre puede existir antes del hijo.*

## **BASE DE DATOS ORIENTADOS A OBJETOS**

Las aplicaciones de las bases de datos en áreas como el diseño asistido por computadora, la ingeniería de software y el procesamiento de documentos no se ajustan al conjunto de suposiciones que se hacen para aplicaciones del estilo de procesamiento de datos. El modelo de datos orientado a objetos se ha propuesto para tratar algunos de estos nuevos tipos de aplicaciones.

El modelo de bases de datos orientado a objetos es una adaptación a los sistemas de bases de datos. Se basa en el concepto de encapsulamiento de datos y código que opera sobre éstos en un objeto. Los objetos estructurados se agrupan en clases. El conjunto de clases está estructurado en sub y superclases basado en una extensión del concepto ISA del modelo Entidad - Relación. Puesto que el valor de un dato en un objeto también es un objeto, es posible representar el contenido del objeto dando como resultado un objeto compuesto.

El propósito de los sistemas de bases de datos es la gestión de grandes cantidades de información. Las primeras bases de datos surgieron del desarrollo de los sistemas de gestión de archivos. Estos sistemas primero evolucionaron en bases de datos de red o en bases de datos jerárquicas y, más tarde, en bases de datos relacionales.

## Estructura de objetos

El modelo orientado a objetos se basa en encapsular código y datos en una única unidad, llamada objeto. La interfaz entre un objeto y el resto del sistema se define mediante un conjunto de mensajes.

Un objeto tiene asociado:

- Un conjunto de atributos (variables) que contienen los datos del objeto. El valor de cada variable es un objeto.
- Un conjunto de mensajes a los que el objeto responde.
- Un método (procesos), que es un trozo de código para implementar cada mensaje. Un método devuelve un valor como respuesta al mensaje.

El término mensaje en un contexto orientado a objetos, no implica el uso de un mensaje físico en una red de computadoras, sino que se refiere al paso de solicitudes entre objetos sin tener en cuenta detalles específicos de implementación.

La capacidad de modificar la definición de un objeto sin afectar al resto del sistema está considerada como una de las mayores ventajas del modelo de programación orientado a objetos.

## Jerarquía de clases

En una base de datos existen objetos que responden a los mismos mensajes, utilizan los mismos métodos y tienen variables del mismo nombre y tipo. Sería inútil definir cada uno de estos objetos por separado por lo tanto se agrupan los objetos similares para que formen una clase, a cada uno de estos objetos se le llama instancia de su clase. Todos los objetos de su clase comparten una definición común, aunque difieran en los valores asignados a las variables.

Así que básicamente las bases de datos orientados a objetos tienen la finalidad de agrupar aquellos elementos que sean semejantes en las entidades para formar una clase, dejando por separado aquellas que no lo son en otra clase.

Por ejemplo: Retomemos la relación alumno-curso-materia agregándole la entidad maestro; donde los atributos considerados para cada uno son Alumno: Nombre, Dirección, Teléfono, Especialidad, Semestre, Grupo; Maestro: Nombre, Dirección, Teléfono, Número económico, Plaza, RFC; Materia: Nombre, Créditos, Clave.

Los atributos de nombre, dirección y teléfono se repiten en la entidad alumno y maestro, así que podemos agrupar estos elementos para formar la clase Persona con dichos campos, quedando por separado en alumno: Especialidad, semestre, Grupo y en maestro: Número económico, Plaza y RFC; la materia no entra en la agrupación (Clase persona) ya que la clase especifica los datos de sólo personas, así que queda como clase materia.

## Herencia

Las clases en un sistema orientado a objetos se representan en forma jerárquica, así que las propiedades o características del elemento persona las contendrán (heredarán) los elementos alumno y maestro. Decimos que tanto la entidad Alumno y Maestro son subclases de la clase Persona; este concepto es similar al utilizado en la de especialización (la relación ISA) del modelo E-R.

Se pueden crear muchas agrupaciones (clases) para simplificar un modelo así que una jerarquía (en forma gráfica) puede quedar muy extensa, en estos casos tenemos que tener bien delimitados los elementos que intervienen en una clase y aquellos objetos que las heredan.

### Consultas orientadas a objetos:

Los lenguajes de programación orientados a objetos requieren que toda la interacción con objetos se realice mediante el envío de mensajes.

Consideremos el ejemplo de alumno-curso-materia deseamos realizar la consulta de los alumnos que cursan la materia de Base de Datos 1. Para realizar esta consulta se tendría que enviar un mensaje a cada instancia alumno.

Así un lenguaje de consultas para un sistema de bases de datos orientado a objetos debe incluir tanto el modelo de pasar el mensaje de objeto a objeto como el modelo de pasar el mensaje de conjunto en conjunto.

### Complejidad de Modificación

En bases de datos orientadas a objetos pueden existir los siguientes cambios:

- Adición de una nueva clase: Para realizar este proceso, la nueva clase debe colocarse en la jerarquía de clase o subclase cuidando las variables o métodos de herencia correspondientes.
- Eliminación de una clase: Se requiere la realización de varias operaciones, se deben cambiar los elementos que se han heredado de esa clase a otras y reestructurar la jerarquía.

En sí la estructuración de modelos orientados a objetos simplifica una estructura evitando elementos o variables repetidas en diversas entidades, sin embargo el precio de esto es dedicarle un minucioso cuidado a las relaciones entre las clases cuando el modelo es complejo, la dificultad del manejo de objetos radica en la complejidad de las modificaciones y eliminaciones de clases, ya que de tener variables que heredan otros objetos se tiene que realizar una reestructuración que involucra una serie de pasos complejos.

**ACTIVIDAD...**

*Responder al siguiente cuestionario*

- 1 Las bases de datos orientados a objetos se encargan de identificar
  - A. *Un objeto único.*
  - B. *Un atributo de una entidad.*
  - C. *Una relación*
  
- 2 *Un motivo de éxito al establecer los requerimientos radica en:*
  - A. *Conformar registros.*
  - B. *Conformar clases.*
  - C. *Conformar relaciones*
  
- 3 *Un objeto estructural se debe agrupar en:*
  - A. *Una base de datos.*
  - B. *Una clase.*
  - C. *Un registro*
  
- 4 *Las bases de datos orientadas a objetos permiten encapsular:*
  - A. *Los datos y las variables.*
  - B. *Las operaciones y los procesos.*
  - C. *Los datos y los códigos.*

**REFERENCIAS BIBLIOGRÁFICAS.**

Bases de Datos, Modelos, Lenguajes, Diseño. James L. Johnson, Oxford University Press. México, 2000.

Procesamiento de Bases de Datos, Fundamentos, diseño e instrumentación. Quinta edición. David M. Kroenke. Ed. Prentice Hall. México 1996.

Sistemas de Bases de Datos, Administración y uso. Alice Y. H. Tsai, Editorial Prentice Hall, México 1990

DB2/SQL, Manual para programadores. Tim Martyn Tim Hartley, Editorial Mc.Graw Hill España 1991

Fundamentos de Bases de Datos, Segunda edición, Henry F. Korth Abraham Silberschatz. Editorial Mc.Graw Hill

### ESTRATEGIAS METODOLÓGICAS

- Presentación de la unidad a cargo del profesor.
- Realización por parte de los alumnos de las actividades propuestas en la unidad

### RECURSOS

- Humanos: Profesor y alumnos
- Institucionales: Salón de clase
- Materiales: Texto guía

### INDICADORES DE EVALUACIÓN

- Evaluación escrita sobre la unidad
- Evaluación de las actividades propuestas en la unidad.

### CRITERIOS DE EVALUACIÓN

- ¿Cómo esta conformada una base de datos de red?

- ¿Una base de datos de red cómo se puede convertir a diagramas de estructura de datos?
- ¿En qué se basa la representación entre registros del modelo de red?
- ¿Qué Objetivos tienen las bases de datos jerárquicas?
- ¿En qué consiste la jerarquía de la información?
- ¿Cuáles son enlaces de datos?
- ¿Qué es una representación de árbol?
- ¿Cómo se clasifican los usuarios que accedan una base de datos?
- ¿Qué Objetivos tienen las bases de datos orientadas a objetos?
- ¿En qué consiste la abstracción de la información?
- ¿Qué es el concepto de encapsulamiento de datos y de código?
- ¿Cuáles son las funciones de una clase?