Universidad Autónoma del Estado de México

Centro Universitario UAEM Valle de Teotihuacán

LICENCIATURA EN INFORMÁTICA ADMINISTRATIVA

APUNTES DE

ADMINISTRACIÓN DE BASES DE DATOS

CLAVE	HORAS DE TEORÍA	HORAS DE PRÁCTICA	TOTAL DE HORAS	CRÉDITOS	NÚCLEO DE FORMACIÓN
L30013	2	2	4	6	SUSTANTIVO

CUARTO SEMESTRE

AUTOR:

M. EN D. ED. OSCAR HERNÁNDEZ GÓMEZ

FECHA DE ELABORACIÓN: 2018- A

Las bases de datos han invadido el ámbito de los sistemas informáticos desde sus inicios y se hacen cada vez más notables en nuestra vida cotidiana, éstas son utilizadas como solución, o como soporte a la solución, de la mayor parte de los problemas que se presentan en el mundo real. Por ello, el conocimiento de esta tecnología, de la teoría en la que se soporta y de los modelos de sistemas encargados de la gestión de las mismas es de suma importancia.

Estos apuntes están dirigidos a los alumnos de Administración de Bases de Datos, esta obra tiene fines didácticos, buscados a través del planteamiento teórico. Sin profundizar en las bases teóricas con un formulismo matemático excesivo, sino, por el contrario, centrarnos en los conceptos y reglas fundamentales y, basándose en los conocimientos que se pretenden del resultado de ésta tienen una aplicación práctica directa, se recomienda aplicar los contenidos vistos, ya sea a través de ejercicios o realizando proyectos personales o grupales.

En cuanto a la estructura de estos apuntes, se puso especial énfasis en ubicar los contenidos de forma tal que su ordenamiento final ayude al lector a incorporar conceptos de manera progresiva y, a la vez, ir reforzando lo aprendido a lo largo de las unidades propuestas. Ello permitirá relacionar y elaborar los conceptos fundamentales que favorecerán al momento de aplicar los modelos de bases de datos.

Se sugiere al lector involucrarse de manera activa en los temas que aquí se tratan, ya que le serán de gran ayuda, tanto en el aspecto personal como en el aspecto profesional.

Objetivos

- Conocer los conceptos sobre la organización de los archivos de datos y de las bases de datos, así como los objetivos que se persiguen en su utilización.
- Analizar los modelos de especificación abstractos y conceptuales de las bases de datos.
- Diseño e implementación de base de datos utilizando los distintos modelos de datos existentes.
- Comprender la importancia del proceso de normalización de las tablas que componen una base de datos y lo aplicará.
- Administrar los recursos de una base de datos.
- Diseñar una base de datos a partir de datos reales utilizando los elementos principales de un modelo conceptual.
- Analizar las metodologías para el diseño de bases de datos.
- Diseño e implementación de base de datos utilizando SQL.

El presente, contiene imágenes referentes a aspectos de suma importancia. A continuación se muestra una lista de estos elementos y su significado.

	Representa el objetivo de la unidad
	Contenidos teóricos
	Tareas e investigaciones

UNIDAD I INTRODUCCIÓN

Objetivo: Conocer los conceptos sobre la organización de los archivos de datos y de las bases de datos, así como los objetivos que se persiguen en su utilización.

1.1 Concepto de sistema de información

Toda organización necesita para su funcionamiento un conjunto de informaciones que han de transmitir entre sus distintos elementos y, generalmente también, desde y hacia el exterior del sistema. Una parte de esta comunicación se realiza por medio de contactos interpersonales entre los empleados, es el sistema de información informal.

Pero este tipo de flujo de información, cuando se trata de organismos complejos, se muestra insuficiente y costoso, siendo preciso disponer de un sistema de información formal, también llamado organizacional que, integrado en el sistema de orden superior que es el organismo, aporte a éste la información necesaria.

Sistema: Un sistema es un conjunto de componentes que interaccionan entre sí para lograr un objetivo común.

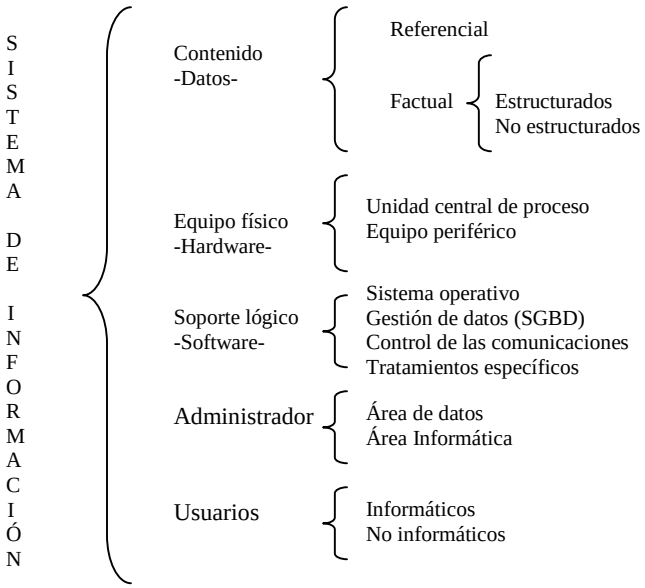
Un **sistema de información** puede definirse técnicamente como un conjunto de componentes interrelacionados que permiten capturar, procesar, almacenar y distribuir la información para apoyar la toma de decisiones y el control en una institución.

Dentro del concepto de sistema de información, una base de datos es también un sistema.

Todo sistema de información formal, de ahora en adelante lo llamaremos simplemente *sistema de información* (SI), se diseña a fin de satisfacer las necesidades de información de una organización (empresa o cualquier tipo de institución pública o privada) y está inmerso en ella. El SI ha de tomar los datos del entorno (la propia organización así como fuentes externas) y sus resultados han de ser la información qué dicha organización necesita para su gestión y toma de decisiones; por otra parte, los directivos de la organización tendrán que marcar los objetivos y directrices por los que se regule el SI.

El SI puede ser comparado con un motor que impulsa la información, haciéndola circular por el organismo, distribuyéndola y aportándola a aquellas áreas donde es necesaria. Para realizar esta función es preciso que el sistema recoja previamente los latos allí donde son generados y los procese para convertirlos en información útil.

Componentes de un sistema de información



Es importante distinguir entre dos tipos de datos: referenciales y factuales. Los **sistemas de información *referencial*** contienen referencias bibliográficas de los documentos donde se puede encontrar la información, pero no la información en sí misma, de modo que una vez recuperado el dato (es decir, la referencia) es preciso conseguir el documento fuente. En cambio, los sistemas de tipo *factual* devuelven la información buscada, la cual puede ser directamente utilizada sin necesidad de acudir a nuevos circuitos informativos.

Otra clasificación, más bien aplicable a los **datos factuales**, se refiere a su formato, según la cual los datos pueden ser *estructurados* o *no estructurados* (también llamados *formateados y no formateados).* Los primeros tienen una cierta estructura o formato en la que los distintos campos ocupan determinadas posiciones fijas (así, en un fichero de personal, el DNI del empleado se puede encontrar en primer lugar, ocupando las ocho primeras posiciones del fichero; el nombre y los apellidos a continuación, etc.). Existen, sin embargo, otros datos cuyo formato no puede ser fijo, como los textos (propios de los sistemas documentales), o los datos multimedia (voz, imagen, etc.); son datos *no estructurados.* Se suele distinguir entre dos tipos de sistemas de gestión distintos según se ocupen del tratamiento de datos estructurados o no estructurados.

Sistemas de Gestión de Bases de Datos (SGBD[1]): se ocupan del tratamiento (definición, actualización y recuperación) de datos estructurados (a ellos está dedicada esta obra).

Sistemas de Recuperación de Información (SRI[2]): se ocupan del tratamiento de datos no estructurados (documentos). Estos sistemas proporcionan facilidades de thesaurus, búsqueda en texto libre, etc.

En la actualidad existe una clara tendencia hacia la convergencia de estos dos de tratamiento, de modo que ambas funcionalidades sean ofrecidas por un único sistema.

El ordenador, que ha de soportar la función de tratamiento o proceso, está integrado por dos subsistemas: el equipo físico *(hardware)* y el soporte lógico (software).

Otro componente fundamental del sistema es el *administrador*, o más bien la unidad de administración (ya que se trata de una función y no de una persona), cuya misión es asegurar la calidad y permitir el uso correcto y permanente de los datos. El administrador no es el propietario de los datos, sino el gestor y custodio de los mismos, cuya responsabilidad se extiende tanto al contenido del sistema como al área informática, si bien estas dos funciones (administración del contenido y administración del SGBD) pueden estar encomendadas a unidades distintas de la organización.

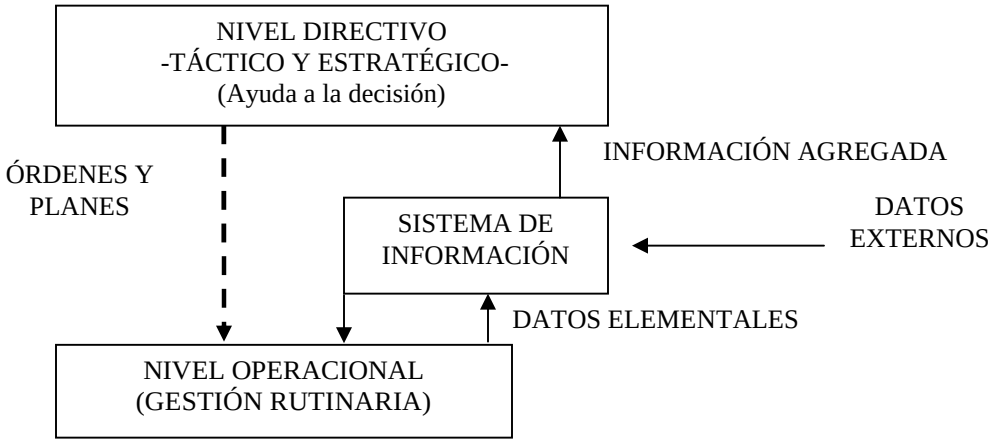
Por último, consideramos como otro decir, a la persona o grupo de personas pueden ser tanto *informáticos* (analistas y pocos conocimientos de informática que generalmente en modo conversacional procedimientos preparados ex profeso, acceden directamente al sistema, pero que componente del sistema a los *usuarios;* es que han de acceder al SI. Estos usuarios programadores) como *usuarios finales* con necesitan consultar o actualizar los datos, y mediante lenguajes muy sencillos o también pueden existir usuarios que no obtienen información del mismo.

1.1. 2 Sistemas de Información para la gestión para la ayuda y toma de decisiones

En toda organización se suele distinguir tres niveles de gestión (operacional, táctico y estratégico), por lo que el SI estará compuesto por tres subsistemas estructurados jerárquicamente y que se corresponden con cada uno de estos tres niveles. En el plano operacional, los usuarios necesitan datos puntuales (elementales) que describan los sucesos que, de una forma u otra, caracterizan las actividades de la organización, por lo que este subsistema de información será muy voluminoso. De él, mediante un proceso de elaboración adecuado (en general de

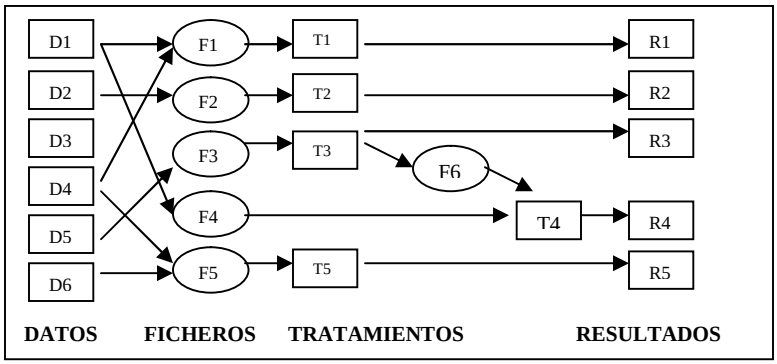
[1] Las siglas inglesas son DBMS, *Datábase Management Systems.*
[2] Las siglas inglesas son IRS, *Information Retrieval Systems.*

agregación), se podrán, obtener los datos necesarios (junto con los aportados desde el exterior) para el funcionamiento de los otros dos subsistemas, cuyos usuarios tienen unas exigencias.



Sistema de información único (nivel directivo y operacional)

1.2 Sistemas de bases de datos frente a sistemas de archivos



Este sistema de procesamiento de archivos típico que se acaba de describir. Los registros permanentes son almacenados en varios archivos y se escriben diferentes programas de aplicación para extraer registros y para añadir registros a los archivos adecuados. Antes de la llegada de los sistemas de gestión de bases de datos (SGBDs), las organizaciones normalmente han almacenado la información usando tales sistemas.

Los datos se recogen varias veces y se encuentran repetidos en los distintos ficheros. Esta redundancia, además de malgastar recursos, origina a menudo divergencias en los resultados.

Este planteamiento produce, además de una ocupación inútil de memoria secundaria, un aumento de los tiempos de proceso, al repetirse los mismos controles y operaciones en los distintos ficheros. Pero más graves todavía son las inconsistencias que a menudo se presentan en estos sistemas, debido a que la actualización de los mismos datos, cuando se encuentran en más de un fichero, no se suele realizar de forma simultánea en todos ellos.

Mantener información de la organización en un sistema de procesamiento de archivos tiene una serie de inconvenientes importantes:

Inconsistencia de datos. Es decir, las diversas copias de los mismos datos pueden no coincidir. Por ejemplo, un cambio en la dirección del cliente puede estar reflejado en los registros de las cuentas de ahorro pero no estarlo en el resto del sistema.

Dificultad en el acceso a los datos. Supóngase que uno de los empleados del banco necesita averiguar los nombres de todos los clientes que viven en el distrito postal 28733 de la ciudad. El empleado pide al departamento de procesamiento de datos que genere dicha lista. Debido a que esta petición no fue prevista cuando el sistema original fue diseñado, no hay un programa de aplicación a mano para satisfacerla.

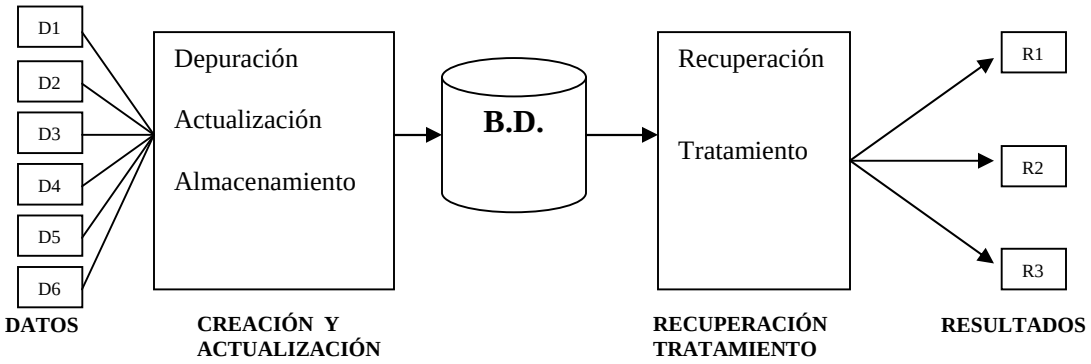
Aislamiento de datos. Debido a que los datos están dispersos en varios archivos.

Problemas de integridad. Los valores de los datos almacenados en la base de datos deben satisfacer ciertos tipos de restricciones de consistencia.

Problemas de atomicidad. Esta debe ocurrir en ellos por completo o no ocurrir en absoluto. Es difícil asegurar esta propiedad en un sistema de procesamiento de archivos convencional.

Problemas de seguridad. No todos los usuarios de un sistema de bases de datos deberían poder acceder a todos los datos.

De este análisis se deduce claramente la necesidad de una gestión más racional del conjunto de datos, surgiendo así un nuevo enfoque que se apoya sobre una base de datos, en la cual los datos son recogidos y almacenados una sola vez, con independencia de los tratamientos (véase figura).



Organización en B.D.: Sistemas orientados a los datos

INVESTIGAR, HISTORIA DE LAS BASES DE DATOS

1.3 Definición de Base de Datos

Se define una base de datos como una serie de datos organizados y relacionados entre si, los cuales son recolectados y explotados por el sistema de información de una empresa o negocio en particular.

1.3.1 Conceptos básicos de Bases de Datos

Ventajas de las Bases de Datos

- Globalización de la información
- Eliminar información redundante
- Eliminar información incongruente
- Permite compartir información
- Permite mantener la integridad de la información
- Independencia de datos
- Eliminar la dificultad en el acceso de los datos

1.4 Visión de los datos

Un **sistema de bases de datos** es una colección de archivos interrelacionados y un conjunto de programas que permitan a los usuarios acceder y modificar estos archivos.

Uno de **los propósitos principales de un sistema de bases 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.

Los usuarios del sistema pueden realizar una variedad de operaciones sobre dichos archivos, por ejemplo:

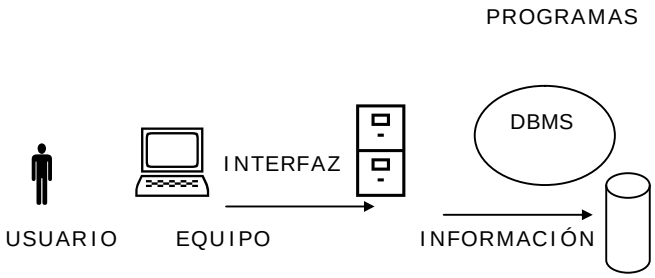
- Agregar nuevos archivos vacíos a la base de datos;
- Insertar datos dentro de los archivos existentes;
- Recuperar datos de los archivos existentes;
- Modificar datos en archivos existentes;
- Eliminar datos de los archivos existentes;
- Eliminar archivos existentes de la base de datos.

Aplicaciones de los sistemas de bases de datos

Banca. Para información de los clientes, cuentas y préstamos, y transacciones bancarias.
Líneas aéreas. Para reservas e información de planificación.
Universidades. Para información de los estudiantes, matrículas de las asignaturas y cursos.
Transacciones de tarjetas de crédito. Para compras con tarjeta de crédito y generación mensual de extractos.

1.4.1 Componentes Principales de un Sistema de Base de Datos

Datos: Son la base de datos propiamente dicha.
Hardware: Son los dispositivos de almacenamiento en donde reside la base de datos.
Software: Conjunto de programas conocidos como sistema manejador de base de datos (DBMS).
Usuarios: Tres categorías (programador de aplicaciones, usuario final y el administrador de la base de datos).



1.4.2 Características de los sistemas de bases de datos

- Los sistemas de base de datos se diseñan para manejar grandes cantidades de información.
- El manejo de los datos implica tanto la definición de estructuras para el almacenamiento como la creación de mecanismos para manejar la información.
- El sistema de base de datos debe cuidar la seguridad de la información almacenada en la BD, previniendo caídas del sistema o intentos de acceso no autorizados.
- Si se comparte la información entre varios usuarios el sistema debe evitar posibles resultados anómalos.

INVESTIGAR, ARQUITECTURA ANSI/X3/SPARC

1.5 Arquitectura de los sistemas de bases de datos

Ahora se esta en condiciones de presentar la arquitectura para un sistema de base de datos. El objetivo al presentar esta arquitectura es ofrecer una infraestructura en la que puedan basarse las unidades siguientes. Dicha infraestructura resulta útil para describir los conceptos generales de las bases de datos y para explicar la estructura de sistemas de bases de datos específicos; no se afirma que todo sistema pueda coincidir enteramente con esta infraestructura en particular, ni se quiere sugerir que esta arquitectura represente la única infraestructura posible, sin embargo, la arquitectura parece ajustarse a la mayoría de los sistemas.

1.5.1 Niveles de la arquitectura

1.5.1. 1 Nivel de Vistas. Es aquel en el que se presenta al usuario final y que puede combinaciones o relaciones entre los datos que conforman a la base de datos global. Puede definirse como la forma en el que el usuario aprecia la información y sus relaciones. Oculta parte de la información a los usuarios, es decir hace visible sólo una parte de la base de datos.

Este usuario debe *ver* la información que maneja como un registro, una ficha de datos (un formulario) con independencia de a qué objeto pertenecen los ítems de datos, correspondientes a ese registro, en el dominio del problema (sistema) y en qué relaciones se ven implicados esos datos.

1.5.1. 2 Nivel Lógico o Conceptual. Es aquel en el que se definen las estructuras lógicas de almacenamiento y las relaciones que se darán entre ellas. Determina la organización de los archivos. Índices, llaves, orden de campos, tipos de datos.

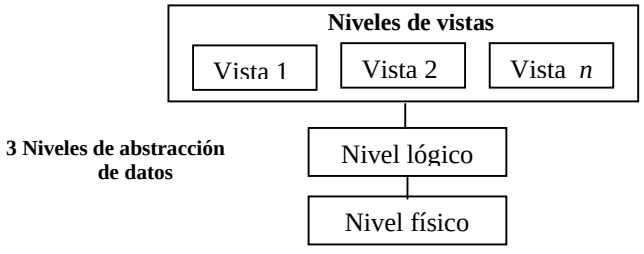
La visión conceptual de una base de datos es una representación abstracta del problema e independiente, en principio, de cómo va a ser tratada esta información, de qué visiones externas pueda tener y de cómo esta información pueda ser almacenada físicamente. Así, la visión conceptual de una base de datos no cambia a no ser que cambie la naturaleza del problema.

```
EMPLEADO (
   NUMERO_EMPLEADO          CHARACTER (6)
   NUMERO_DEPARTAMENTO      CHARACTER (4)
   SALARIO                  NUMERIC (5)
            );
```

1.5.1.3 Nivel Físico. Es el nivel mas bajo de abstracción, determina como están almacenados físicamente los datos (pistas, sectores, cilindros), volúmenes, archivos, tipos de datos, punteros, etc., representa el nivel más bajo.

La visión física de una base de datos es la representación de cómo la información es almacenada en los dispositivos de almacenamiento. Esta visión describe las estructuras u organizaciones físicas, dispositivos, volúmenes, archivos, tipos de datos, punteros, etc., estructuras de mayor o menor complejidad que representan el dominio del problema de una forma entendible por el sistema informático.

```
EMP# TYPE=BYTES (6), OFFSET=6, INDEX= EMPX
DEPT# TYPE= FULLWORD, OFFSET=16
```



Como se describirá más adelante, una base de datos tiene diferentes tipos de usuarios, los cuales sólo observan aquella parte de la base de datos que les interesa y a un nivel de abstracción que son capaces de entender.

1.5.2 Ejemplares, esquemas e independencia

Las bases de datos van cambiando a lo largo del tiempo conforme la información se inserta y se borra, la colección de información almacenada en la base de datos en un momento particular se denomina un ***ejemplar de base de datos (ocurrencia del esquema).*** El diseño completo de la base de datos se llama el ***esquema*** de la base de datos. Los esquemas son raramente modificados.

El concepto de esquemas y ejemplares de bases de datos se puede entender por analogía por un programa escrito en un lenguaje de programación. Un esquema de B.D. corresponde a las declaraciones de variables (junto con definiciones de tipos asociados), En un programa. Cada variable tiene un valor particular en un instante de tiempo. Los valores de las variables en un programa en un instante de tiempo corresponde a un ejemplar de un esquema de bases de datos.

Una base de datos puede tener varios esquemas en el nivel de vistas, a menudo denominados subesquemas, de éstos, el esquema lógico es con mucho el más importante, en términos de su efecto en los programas de aplicación, ya que los programadores construyen las aplicaciones usando el esquema lógico.

Como veremos más adelante no se trata de modelizar "todo" el mundo sino sólo la parte "importante" y "pertinente" para alcanzar los objetivos funcionales del sistema. Esa parte del mundo que nos interesa la llamaremos **espacio del problema**.

Independencia física de los datos: Es la habilidad para modificar el esquema físico sin cambiar el esquema lógico.

Las aplicaciones dependen del esquema lógico.

En general, las interfaces entre los diferentes niveles y componentes deberán estar bien definidos para que los cambios en algunas partes no afecten de forma importante a otras.

HACER UN CUADRO SINÓPTICO DE PAG. 25, 26,27, 28, 29, 30, 31, 32, BASES DE DATOS DESDE CHEN HASTA CODD CON ORACLE

1.5.3 Sistema de Gestión de Bases de Datos

Es importante conocer la diferencia entre lo que es **una base de datos** y lo que es un ***Sistema de Gestión de Bases de Datos (SGBD)***, términos que se confunden muy a menudo cuando se está trabajando con la información haciendo uso de esta tecnología.

Cuando se habla de bases de datos se habla de información que está almacenada cumpliendo toda una serie de características y restricciones como las que se han expuesto.
Pero para que la información pueda ser almacenada como se ha descrito y el acceso a la misma satisfaga las características exigidas a una base de datos para ser denominada como tal, es necesario que exista una serie de procedimientos (un sistema software) que sea capaz de llevar a cabo tal labor. A este sistema software es a lo que se le denomina *Sistema de Gestión de Bases de Datos**.

Así, **un *SGBD*** es una colección de programas de aplicación que proporcionan al usuario de la base de datos los medios necesarios para realizar las siguientes tareas:

Definición de los datos a los distintos niveles de abstracción (físico, lógico y externo).

Manipulación de los datos en la base de datos. Es decir, la inserción, modificación, borrado y acceso o consulta a los mismos.
Mantenimiento de la integridad de la base de datos. Integridad en cuanto a los datos en sí, sus valores y las relaciones entre ellos.

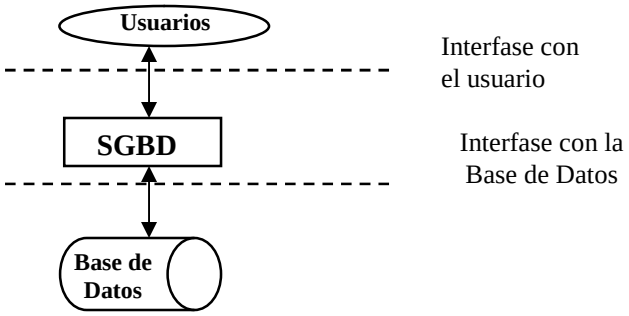
Control de la privacidad y seguridad de los datos en la base de datos.
Y, en definitiva, los medios necesarios para el establecimiento de todas aquellas características exigibles a una base de datos.

El objetivo primordial de una *SGBD* es crear un ambiente en el que sea posible almacenar y recuperar información en forma eficiente y conveniente.
Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información.

El *SGBD* es responsable de:
- ✓ Mantener las relaciones entre la información y la base de datos.
- ✓ Asegurar propiedades de atomicidad y durabilidad de la información, es decir recuperar toda la información en un punto conocido en caso de que el sistema falle.
- ✓ Simplificar y facilitar el acceso a los datos, con un buen almacenamiento, recuperación y actualización de los datos.

Proceso para acceder a la información de una BD.

1. El usuario solicita cierta información de la base de datos.
2. El ***SGBD*** intercepta este requerimiento y lo interpreta.
3. El ***SGBD*** realiza las operaciones necesarias para acceder y / o actualizar la información solicitada.



Dos tipos de comandos:

DDL: Permiten crear y definir nuevas bases de datos, campos e índices.
DML: Permiten generar consultas para ordenar, filtrar y extraer datos de la base de datos.

Comandos DDL

CREATE: Crear nuevas tablas, campos e índices.
DROP: Eliminar tablas e índices.
ALTER: Modificar las tablas agregando campos o cambiando la definición de los campos.l

Comandos DML

SELECT: Consultar registros de la base de datos que satisfagan un criterio determinado.
INSERT: Insertar registros en las tablas.
UPDATE: Modificar los valores de los campos y registros especificados.
DELETE: Eliminar registros de las tablas.

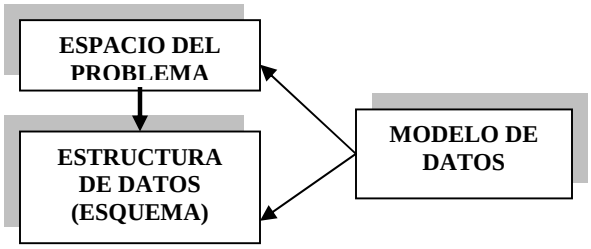
INVESTIGAR COMPONENTES DE LOS SGBD O DBMS

UNIDAD II MODELOS DE BASES DE DATOS

Objetivo: Analizar los modelos de especificación abstractos y conceptuales de las bases de datos.

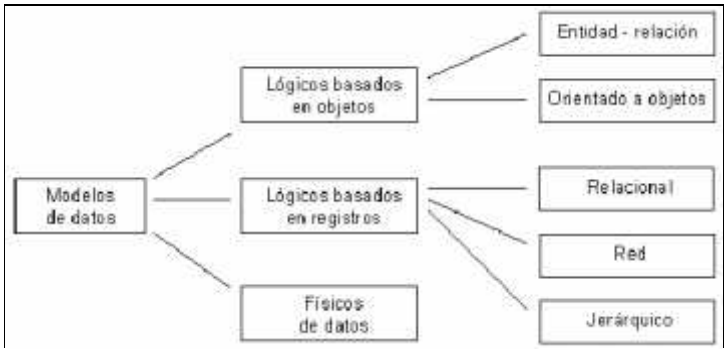
2 Modelo de datos

Se utilizará para significar una descripción conceptual del espacio del problema mediante una colección de herramientas conceptuales para describir los datos, las relaciones, la semántica y las restricciones de consistencia.



Aplicación de un modelo de datos a un mundo real para obtener un esquema

Tipos de modelos de datos



Para ilustrar el concepto de un modelo de datos, describimos dos modelos de datos en este apartado: el modelo entidad-relación y el modelo relacional.

Tarea Hacer un resumen de PAG. 40, 41, 42. , **del libro. Bases de datos desde CHEN HASTA CODD Con ORACLE**

2.1 Modelo entidad-relación

Fue propuesto por *Peter Chen* a mediados de 1970 para la representación conceptual de los problemas y como un medio para representar la visión de un sistema de forma global. Se trata de un modelo muy extendido que ha experimentado a lo largo de los años una serie de ampliaciones, lo que ha originado un medio muy potente para la representación de los datos correspondientes a un problema. Las características actuales de este modelo permiten la representación de cualquier tipo de sistema y a cualquier nivel de abstracción o refinamiento,

El modelo de datos entidad-relación (E-R) está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados *entidades, y* de *relaciones* entre estos objetos.

A demás este se utiliza habitualmente en el proceso de diseño de bases de datos.
Antes de comenzar a describir el modelo E-R, es necesario introducir una serie de conceptos básicos que son utilizados por el mismo. Así, se define:

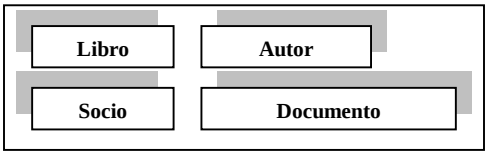
Entidad: Es una «cosa», «persona» u «objeto» en el mundo real que es distinguible de todos los demás objetos. ANSI (1977).

Ejemplo: una persona.

Una entidad tiene un conjunto de propiedades, y los valores para algún conjunto de propiedades pueden identificar una entidad de forma unívoca.

Ejemplo: El número de cuenta 22546 identifica unívocamente una persona particular de la universidad.

Una entidad puede ser concreta, como una persona o un libro o puede ser abstracta como un préstamo.



Conjunto de entidades: Es un conjunto de entidades del mismo tipo que comparten las mismas propiedades o atributos.

Ejemplo: El conjunto de todas las personas que son estudiantes de la universidad, se puede definir como el conjunto de entidades alumno.

Tipo de entidad: Representa la clasificación de entidades individuales.
Extensión del conjunto de entidades: Son las entidades individuales que constituyen un conjunto.
Ocurrencia de entidad: es cada una re las realizaciones concretas de ese tipo de entidad.
Ejemplo: cada alumno en concreto, Marcos Martínez Pérez

Una entidad se representa mediante un conjunto de atributos.

Ejemplo: Los posibles atributos del conjunto de entidades alumno son *id_alumno, nombre_alumno, calle_alumno y ciudad_alumno.*

Dominio: Es el conjunto de valores permitidos para cada atributo.

Atributos: describen propiedades que posee cada miembro de un conjunto de entidades.

Cada entidad tiene un valor para cada uno de sus atributos.

Ejemplo: El valor 22546 para el id_alumno, el valor Juan Pérez santos para nombre_alumno, el valor gasa para la calle y México para el valor *ciudad_alumno.*

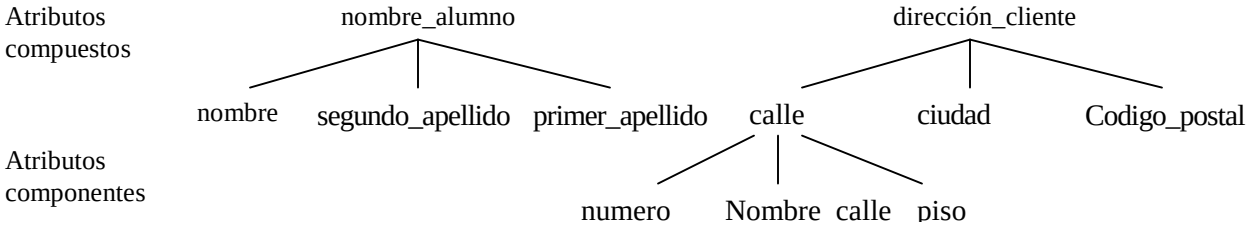
Para cada atributo hay un conjunto de valores permitidos, llamados el dominio, o el conjunto de valores, de ese atributo.

Ejemplo: El dominio del atributo id_alumno podría ser el conjunto de todas las cadenas de la forma *"p-n"* donde n es un entero positivo.

Un atributo, se puede caracterizar por los siguientes tipos de atributo.

Atributos simples y compuestos.
Atributos simples: No están divididos en subpartes.
Ejemplo: *nombre_alumno.*

Atributos compuestos: Se pueden dividir en subpartes (es decir, en otros atributos).
Ejemplo: nombre_alumno podría ser estructurado como un atributo compuesto consistente en nombre, primer_apellido y segundo_apellido.



Atributos Monovalorados y Multivalorados.

Monovalorados: Tienen un sólo valor para una entidad concreta.
Ejemplo: El atributo *id_alumno* referencia a una única persona.
Multivalorados: Tiene un conjunto de valores para una entidad específica.

Ejemplo: Un conjunto de entidades alumno con el atributo numero_telefono, cualquier alumno en particular puede tener cero o varios números telefónicos.

Atributos derivados: El valor para este tipo de atributo se puede derivar de los valores de otros atributos o entidades relacionados.

Ejemplo: Si el conjunto de entidades alumno tiene un atributo fecha_nacimiento, se puede calcular la edad a partir de la fecha de nacimiento y de la fecha actual.

Un atributo toma un valor nulo cuando una entidad no tiene un valor para un atributo. El valor nulo también puede indicar «no aplicable», es decir, que el valor no existe para la entidad.

Ejemplo: Una persona pude no tener segundo nombre de pila.

Valor perdido: valor existe pero no se tiene esa información.

Ejemplo: Si el valor nombre para un alumno es nulo, se asume que el valor es perdido ya que cada alumno debe tener un nombre.

2.1.2 Conjuntos de relaciones

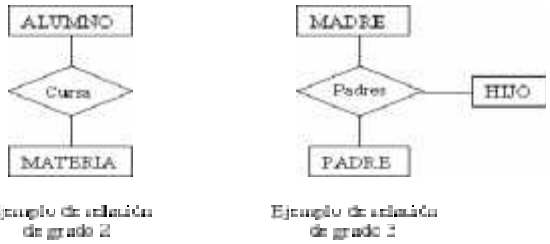
Una *relación* es la asociación que existe entre dos a más entidades.

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. un conjunto de relaciones ternario tiene grado 3.

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

Conjunto de relaciones: Conjunto de relaciones del mismo tipo. Formalmente es una relación matemática con $n >= 2$ de conjuntos de entidades (posiblemente no distintos). Si $E_1,E_2,...,E_n$ son conjuntos de entidades, entonces un conjunto de relaciones R es un subconjunto de:
$\{(e_1,e_2,...,e_n)\ e_1 \in E_1, e_2 \in E_2,...,e_n \in E_n$
donde $(e_1,e_2,...e_n)$ es una relación.
Considérense las dos entidades alumno y materia. Se define el conjunto de relaciones *tomamateria* para denotar la asociación entre alumnos y materias.

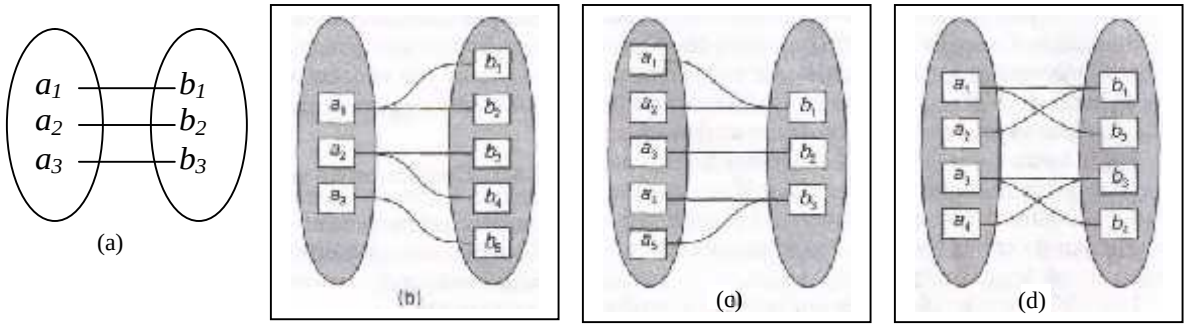
La asociación entre conjuntos de entidades se conoce como *participación;* es decir, los conjuntos de entidades $E_1,E_2,...,E_n$ participan en el conjunto de relaciones R.

2.1.3 Restricciones

Un esquema de desarrollo E-R puede definir ciertas restricciones a las que los contenidos de la base de datos se deben adaptar. En este apartado se examina la correspondencia de cardinalidades y las restricciones de participación, que son dos de los tipos más importantes de restricciones.

2.1.3.1 Correspondencia de cardinalidades

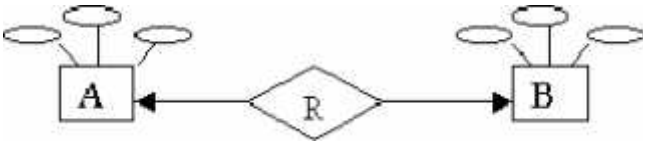
También llamada razón de cardinalidad, expresa el número de entidades a las que otra entidad puede estar asociada vía un conjunto de relaciones.



Uno a uno. Una entidad en *A* se asocia con *a lo sumo* una entidad en *B,* y una entidad en *B* se asocia con *a lo sumo* una entidad en *A* (véase la Figura (a)).

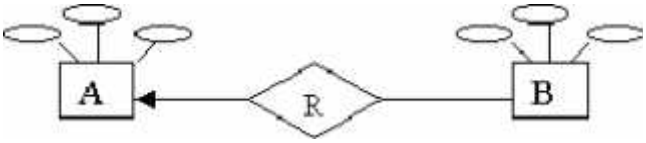
Por ejemplo: la relación *asignación de automóvil* que contiene a las entidades EMPLEADO, AUTO, es una relación 1 a 1, ya que asocia a un empleado con un único automóvil por lo tanto ningún empleado posee más de un automóvil asignado, y ningún vehículo se asigna a más de un trabajador.

Es representado gráficamente de la siguiente manera:



Ejemplos., el RFC de cada persona, El CURP personal, El acta de nacimiento, ya que solo existe un solo documento de este tipo para cada una de las diferentes personas.

Uno a varios. Una entidad en *A* se asocia con cualquier número de entidades en *B* (ninguna o varias). Una entidad en *B*, sin embargo, se puede asociar con *a lo sumo* una entidad en *A* (véase la Figura (b)).

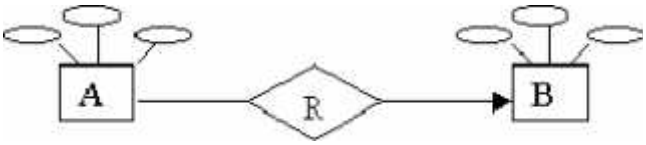


Nótese en este caso que el extremo punteado de la flecha de la relación de *A* y *B*, indica una entidad *A* conectada a muchas entidades *B*.

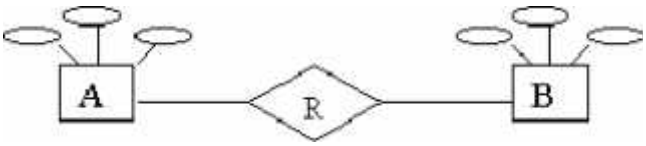
Ejemplos., Cliente – Cuenta en un banco, Padre-Hijos, Camión-Pasajeros, zoológico- animales, árbol – hojas.

Varios a uno. Una entidad en *A* se asocia con *a lo sumo* una entidad en *B*. Una entidad en *B*, sin embargo, se puede asociar con cualquier número de entidades (ninguna o varias) en *A* (véase la Figura (c)).

Indica que una entidad del tipo B puede relacionarse con cualquier cantidad de entidades del tipo A, mientras que cada entidad del tipo A sólo puede relacionarse con sólo una entidad del tipo B.



Varios a varios. Una entidad en *A* se asocia con cualquier número de entidades (ninguna o varias) en B, y una entidad en *B* se asocia con cualquier número de entidades (ninguna o varias) en A (véase la Figura (d)).



Ejemplo: Arquitectos – proyectos

2.1.4 Restricciones de participación

La participación de un conjunto de entidades *E* en un conjunto de relaciones *R* se dice que es total si cada entidad en *E* participa al menos en una relación en *R*. Si sólo algunas entidades en *E* participan en relaciones en *R*, la participación del conjunto de entidades *E* en la relación *R* se llama parcial.

Por ejemplo, se puede esperar que cada entidad materia esté relacionada con al menos un alumno mediante la relación *toma_curso*. Por lo tanto, la participación de *toma_curso* en el conjunto de relaciones *toma_curso* es total. En cambio, un individuo puede ser socio de una biblioteca y tenga o no tenga un préstamo en la biblioteca.

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.*

2.1.5 Claves

Los valores de los atributos de una entidad deben ser tales que permitan *identificar unívocamente* a la entidad. En otras palabras, no se permite que ningún par de entidades tengan exactamente los mismos valores de sus atributos.

Una clave permite identificar un conjunto de atributos suficiente para distinguir las entidades entre sí. Las claves también ayudan a identificar unívocamente a las relaciones y así a distinguir las relaciones entre sí.

Una superclave es un conjunto de uno o más atributos que, tomados colectivamente, **permiten identificar de forma única una entidad en el conjunto de entidades.** Por ejemplo, el atributo *id_alumno* del conjunto de entidades alumno es suficiente para distinguir una entidad alumno de las otras. Así, *id_alumno* es una superclave. Análogamente, la combinación de *nombre_alumno* e *id_alumno* es una superclave del conjunto de entidades alumno. El atributo *nombre_alumno* de alumno no es una superclave, porque varias personas podrían tener el mismo nombre.

El concepto de una superclave no es suficiente para lo que aquí se propone, ya que, como se ha visto, una superclave puede contener atributos innecesarios. Si *K* es una superclave, entonces también lo es cualquier superconjunto de *K.* A menudo interesan las superclaves tales que los subconjuntos propios de ellas no son superclave. Tales superclaves mínimas se llaman claves candidatas.

Es posible que conjuntos distintos de atributos pudieran servir como **clave candidata**. Supóngase que una combinación de *nombre_alumno y calle_aluno* es suficiente para distinguir entre los miembros del conjunto de entidades alumno. Entonces, los conjuntos *{id_alumno}* y *{nombre_alumno, calle_alumno}* son claves candidatas. Aunque los atributos *id_alumno y nombre_alumno* juntos puedan distinguir entidades alumno, su combinación no forma una clave candidata, ya que el atributo *id_alumno* por sí sólo es una clave candidata.

Se usará el término clave primaria para denotar una clave candidata que es elegida por el diseñador de la base de datos como elemento principal para identificar las entidades dentro de un conjunto de entidades. Una clave (primaria, candidata y superclave) es una propiedad del conjunto de entidades, más que de las entidades individuales. Cualesquiera dos entidades individuales en el conjunto no pueden tener el mismo valor en sus atributos clave al mismo tiempo. La designación de una clave representa una restricción en el desarrollo del mundo real que se modela.

La clave primaria se debería elegir de manera que sus atributos nunca, o muy raramente, cambien. Por ejemplo, el campo dirección de una persona no debería formar parte de una clave primaria, porque probablemente cambiará. Los números de cuenta del alumno, por otra parte, es seguro que no cambiarán. Los identificadores únicos generados por empresas generalmente no cambian, excepto si se fusionan dos empresas; en tal caso el mismo identíficador puede haber sido emitido por ambas empresas y es necesario la reasignación de identificadores para asegurarse de que sean únicos.

Por ejemplo, si consideramos la entidad ALUMNO de la UAPVT, podríamos tener los siguientes atributos: Nombre, Semestre, Especialidad, Dirección, Teléfono, Número de cuenta, de todos estos atributos el que podremos designar como clave primaria es el número de cuenta, ya que es diferente para cada alumno y este nos identifica en la institución.

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

REALIZAR UN RESUMEN DE LA PAG. 25, 26, 27. FUNDAMENTOS DE B.D KORTH

2.1.6 Diagrama entidad-relación

Denominado por sus siglas como: E-R; Este modelo representa a la realidad a través de un esquema gráfico empleando los 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.

La estructura lógica general de una base de datos se puede expresar gráficamente mediante un *diagrama* E-R, que consta de los siguientes componentes:

Rectángulos, que representan conjuntos de entidades.
Elipses, que representan atributos.
Rombos, que representan relaciones entre conjuntos de entidades.
Líneas, que unen los atributos con los conjuntos de entidades y los conjuntos de entidades con las relaciones.
Subrayado: india que un atributo es una clave primaria

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.

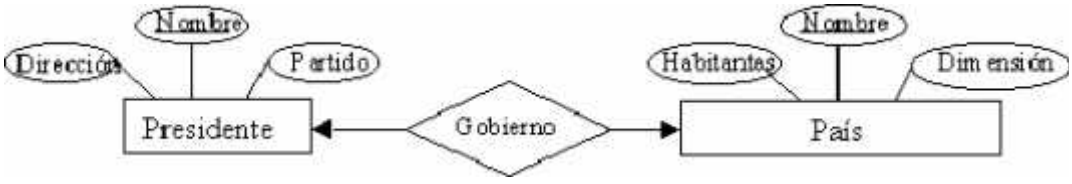
A continuación mostraremos algunos ejemplos de modelos E-R, considerando las cardinalidades que existen entre ellos:

Relación Uno a Uno.

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.



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).

2.1.7 Generalización y especialización

Generalización.

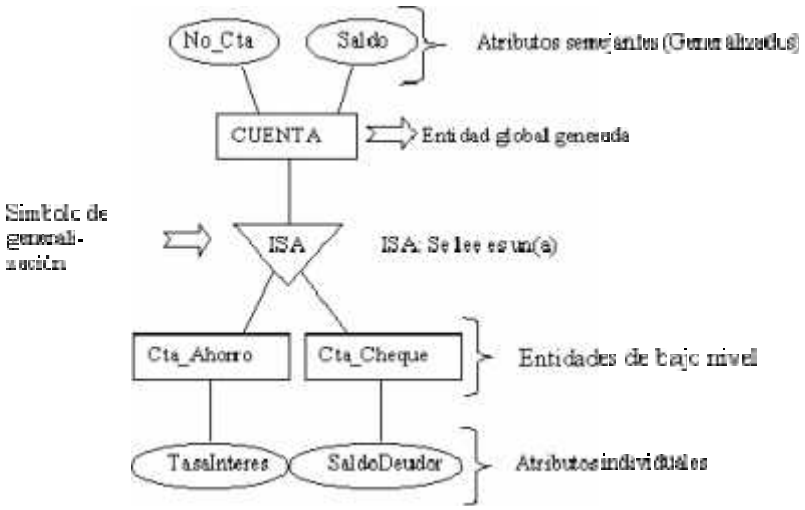
Es el resultado de la unión de 2 o más conjuntos de entidades (de bajo nivel) para producir un conjunto de entidades de más alto nivel. **La generalización se usa para hacer resaltar los parecidos entre tipos de entidades de nivel más bajo y ocultar sus diferencias.**

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) quedara a un nivel más alto al de las entidades origen.

Ejemplo:
Tomando el ejemplo del libro de fundamentos de base de datos de Henry F. Korth.

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 TasaInteres y Cta_Cheques el atributo SaldoDeudor. De todos estos atributos podemos juntar (generalizar) No_Cta y Saldo que son iguales en ambas entidades.

Entonces tenemos:



Herencia de atributos: Un conjunto entidad de nivel más bajo hereda todos los atributos y participaciones en asociaciones del conjunto entidad de nivel superior al que está enlazado.

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.

Especialización:

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

1. En la generalización cada entidad de alto nivel debe ser también una entidad de bajo nivel. La especialización no tiene este limitante.
2. 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.
3. 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 del libro de fundamentos de Base de Datos de Henry F. Korth. En donde el problema consiste en que existen trabajando muchos empleados que trabajan en diferentes proyectos, pero dependiendo del trabajo que realiza en pueden llegar a utilizar un equipo o maquinaria; en este problema intervienen 3 entidades: Empleado, Proyecto y Maquinaria, el diagrama E-R correspondiente es:

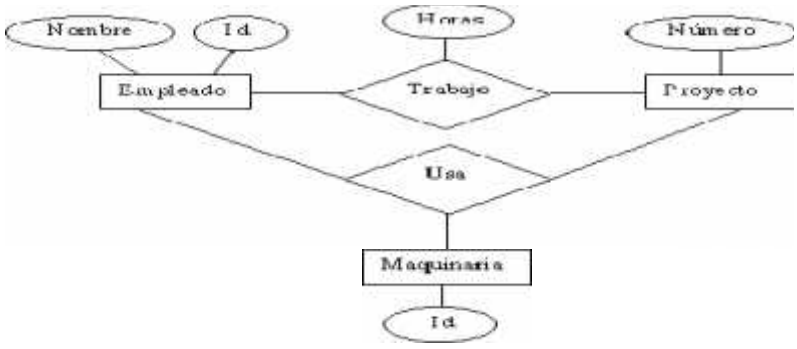


Diagrama E-R con relaciones redundantes

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:

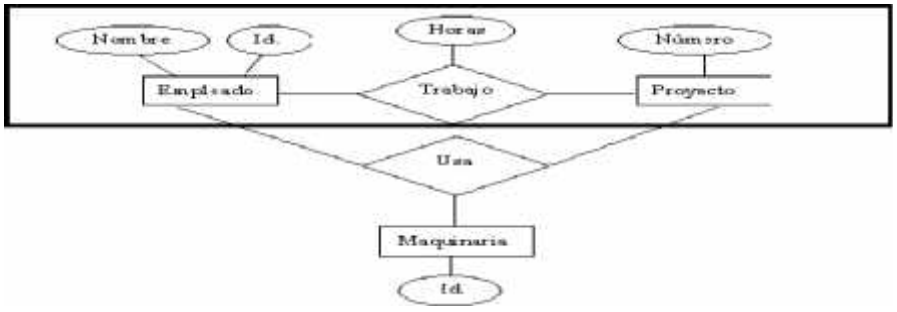


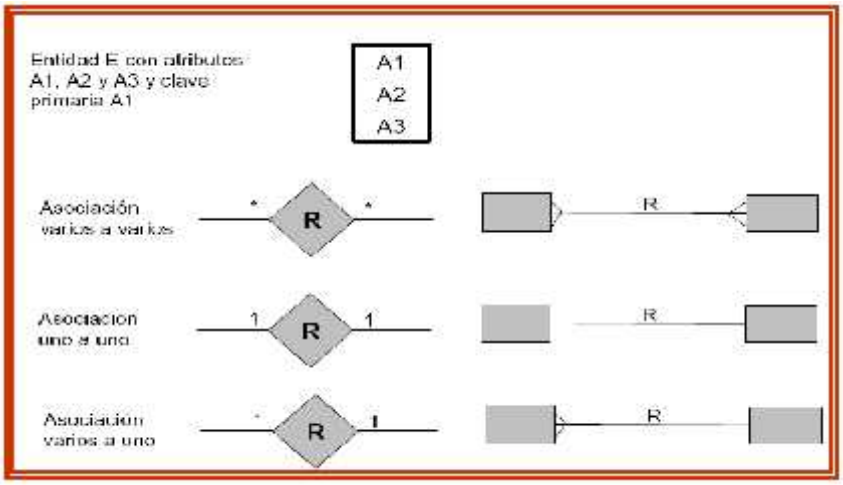
Diagrama E-R con agregación

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.

Instancias.
Cada entidad debe tener múltiples ocurrencias o instancias. Por ejemplo, la entidad Empleado, tiene una ocurrencia (o instancia) por cada empleado en la empresa. Cada instancia de la entidad, tiene valores específicos para los atributos de la entidad.

Observaciones:
Las instancias a menudo son confundidas con las entidades.
Una entidad es una clase o categoría de algo, por ejemplo EMPLEADO.
Una instancia es una "cosa" específica, por ejemplo, el empleado Juan Bravo.

2.1.7.1 Notaciones E-R alternativas

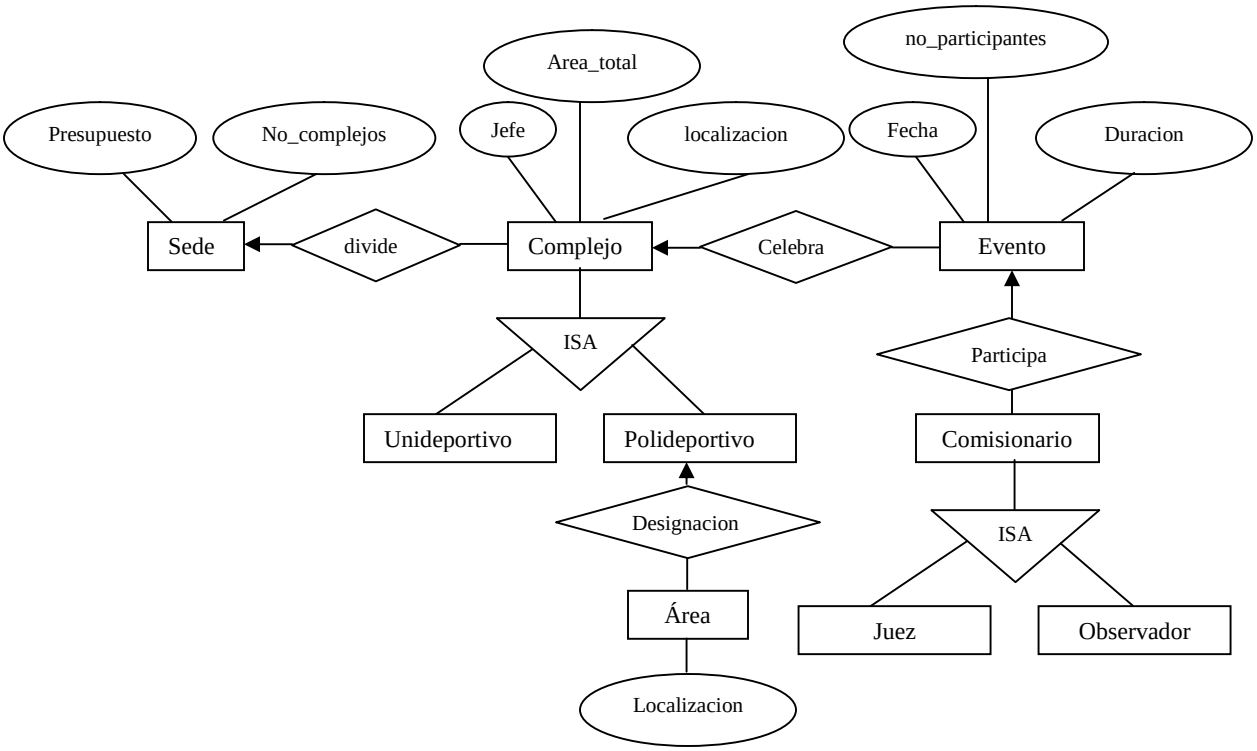


Ejemplo, modelo Entidad Relación utilizando generalización y especialización

Las sedes olímpicas se dividen en complejos deportivos. Los complejos deportivos se subdividen en aquellos en los que se desarrolla un único deporte y en los polideportivos. Los complejos polideportivos tienen áreas designadas para cada deporte. Un complejo tiene una localización, un jefe de organización individual y un área total ocupada.

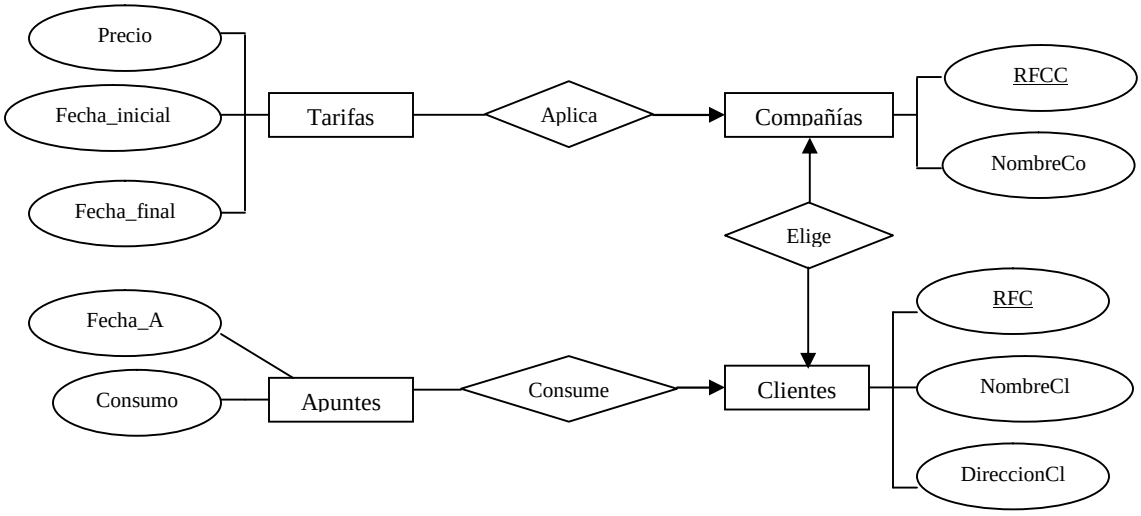
Los dos tipos de complejos (deporte único y polideportivo) tendrán diferentes tipos de información. Para cada tipo de sede, se conservará el número de complejos junto con su presupuesto aproximado.

Cada complejo celebra una serie de eventos (ejemplo: la pista del estadio puede celebrar muchas carreras distintas.). Para cada evento está prevista una fecha, duración, número de participantes. Una lista de todos los comisarios se conservará junto con la lista de los eventos en los que esté involucrado cada comisario ya sea cumpliendo la tarea de juez u observador.



Ejemplo modelo E-R. Facturación de consumos

Usted trabaja en una consultoría y se le ha encargado que realice la facturación de los consumos que se producen en una red eléctrica compartida por diferentes compañías. Se desea tener información sobre los clientes (con su RFC, nombre y Dirección) que consumen la electricidad, las compañías (con su RIF y nombre) y los apuntes que representan el consumo (en kW/h) de un cliente en una fecha en concreto. Los clientes eligen la compañía que les presta el servicio entre dos fechas (pueden cambiar en cualquier momento de compañía). Las compañías aplican tarifas válidas entre dos fechas que indican el coste por kW/h. Una compañía sólo puede aplicar una tarifa cada día, independientemente del cliente. Un cliente se entiende como tal si ha elegido una compañía.



2.2 Modelo relacional

Fue E.F. Codd quien desarrolló en IBM- San José (California) el modelo de datos relacional. Tiene asociada la teoría de normalización de las relaciones que tienen por objeto la eliminación de los comportamientos anómalos de las relaciones durante el proceso de manejo de la información.
El modelo relacional se ha establecido actualmente como el principal modelo de datos para las aplicaciones de procesamiento de datos. Se basa en un conjunto de tablas. El usuario del sistema de datos puede consultar esas tablas, insertar nuevas tuplas, borrar tuplas y actualizar (modificar) las tuplas.

Codd propuso que los sistemas de bases de datos deberían presentarse a los usuarios con una visión de los datos organizados en estructuras llamadas *relaciones*, definidas como conjuntos de tuplas (filas) y no como series o secuencias de objetos, con lo que el orden no es importante. Por tanto, detrás de una relación puede haber cualquier estructura de datos compleja que permita una respuesta rápida a una variedad de consultas. Codd hizo entonces énfasis en que el usuario de un sistema relacional sólo debía preocuparse por el qué consultar y no el cómo de las estructuras de almacenamiento.

La estructura fundamental del modelo relacional es la *relación*, es decir una tabla bidimensional constituida por filas (tuplas) y columnas (atributos). Las relaciones representan las entidades que se consideran interesantes en la base de datos. Cada instancia de la entidad encontrará sitio en una tupla de la relación, mientras que los atributos de la relación representan las propiedades de la entidad. Por ejemplo, si en la base de datos se tienen que representar personas, podrá definirse una relación llamada "Personas", cuyos atributos describen las características de las personas. Cada tupla de la relación "Personas" representará una persona concreta. Por ejemplo, la relación:

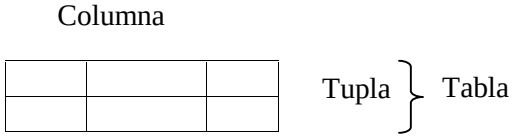
Personas (RFC, nombre, apellido, sexo, estadoCivil, fechaNacimiento)
Es apenas una definición de la estructura de la tabla, es decir su nombre y la lista de atributos que la componen.

Aunque una **relación** es más conocida como ***tabla***, las **tuplas** como ***filas*** y los **atributos** como ***columnas***.

En las bases de Codd se definían los objetivos de este modelo:

- **Independencia física.** La forma de almacenar los datos, no debe influir en su manipulación lógica
- **Independencia lógica.** Las aplicaciones que utilizan la base de datos no deben ser modificadas por que se modifiquen elementos de la base de datos.
- **Flexibilidad.** La base de datos ofrece fácilmente distintas vistas en función de los usuarios y aplicaciones.
- **Uniformidad.** Las estructuras lógicas siempre tienen una única forma conceptual (las tablas)
- **Sencillez.**

2.2.1 Estructura general de la BD relacional



MODELO E-R	MODELO RELACIONAL
Conjunto de entidad o relación	Tabla
Atributo	Columna
Entidad o relación	Tupla

2.2.2 Terminología del modelo relacional

Una de las formas más naturales, y por tanto fácilmente entendibles por el usuario, de representar la información es sobre la base del uso de una representación tabular plana de la misma. Una tabla bidimensional mediante la cual se represente tanto los objetos como las relaciones entre ellos existentes en el dominio del problema.
Tupla. Cada fila de la tabla (cada ejemplar que la tabla representa)
Atributo. Cada columna de la tabla
Grado. Número de atributos de la tabla
Cardinalidad. Número de tuplas de una tabla
Dominio. Conjunto válido de valores representables por un atributo.

2.2.3 Tipos de tablas

- **Persistentes.** Sólo pueden ser borradas por los usuarios:
    - **Vistas.** Son tablas que sólo almacenan una definición de consulta, resultado de la cual se produce una tabla cuyos datos proceden de las bases o de otras vistas e instantáneas. Si los datos de las tablas base cambian, los de la vista que utiliza esos datos también cambia.
    - **Instantáneas.** Son vistas (creadas de la misma forma) que sí que almacenan los datos que muestra, además de la consulta que dio lugar a esa vista. Sólo modifican su resultado (actualizan los datos) siendo refrescadas por el sistema cada cierto tiempo.
- **Temporales.** Son tablas que se eliminan automáticamente por el sistema. Pueden ser de cualquiera de los tipos anterior.

2.2.4 Relación

Dada una serie de conjuntos $D_1, D_2, D_3, ..., D_n$, no necesariamente distintos, se dice que R es una relación entre estos n conjuntos si es un conjunto de n tuplas no ordenadas $(d_1, d_2, d_3, ...d_n)$ tales que $d_1 \in D_1, d_2 \in D_2, d_3 \in D_3, ..., d_n \in D_n$. a los conjuntos $D_1, D_2, D_3, ..., D_n$ se les denomina dominios de R, y el valor de n es el grado de la relación R

ALUMNO

matricula#	nombre	apellido	curso	calificación
3456	José	Pérez de la Lastra	1	5.25
0101	Maria	Atunes Argote	2	7.8
8743	Lourdes	Soria Madrid	1	4.5
1234	Antonio	González Silos	3	6.35

Ejemplo de relación

Se trata de una tabla denominada ALUMNO, se tienen cuatro tuplas. La relación es de grado 5, los 5 dominios son conjuntos de valores que representan.

Al número de tuplas de una relación en un instante dado se le denomina *cardinalidad* de la relación. Así la relación alumno tiene una cardinalidad de cuatro. Al número de columnas de una relación se le denomina grado de la relación. Así, la relación ALUMNO tiene un grado cinco.

2.2.5 Integridad de los esquemas relaciónales

La intención de un esquema relacional, es decir, el conjunto de las intenciones de las relaciones consideradas en el esquema, debe satisfacer las siguientes reglas de integridad mediante las cuales se garantiza la consistencia de la información que pueda ser manejada sobre la base de ese esquema.

Restricciones: Se trata de unas condiciones de obligado cumplimiento por los datos de la base de datos.
Las hay de varios tipos.

- **Inherentes:** Son aquellas que no son determinadas por los usuarios, sino que son definidas por el hecho de que la base de datos sea relacional. Por ejemplo:
    - No puede haber dos tuplas iguales
    - El orden de las tuplas no importa
    - El orden de los atributos no importa
    - Cada atributo sólo puede tomar un valor en el dominio en el que está inscrito

- **Semánticas:** El modelo relacional permite a los usuarios incorporar restricciones personales a los datos.

Las principales son:

- **Clave primaria.** Hace que los atributos marcados como clave primaria no puedan repetir valores.
- **Unicidad.** Impide que los valores de los atributos marcados de esa forma, puedan repetirse.
- **Obligatoriedad.** Prohíbe que el atributo marcado de esta forma no tenga ningún valor
- **Regla de validación.** Condición que debe de cumplir un dato concreto para que sea actualizado.
- **Integridad referencial.** Prohíbe colocar valores en una clave externa que no estén reflejados en la tabla donde ese atributo es clave primaria.

La integridad referencial garantiza que los cambios efectuados en una tabla no afectaran a la relación. Además impide:

- Añadir un registro a la tabla seleccionada si el valor del campo de relación no existe en la tabla principal.
- Eliminar un registro de la tabla principal si tiene registros relacionados en la tabla relacionada. Esto quiere decir, siguiendo con el ejemplo anterior, que antes de borrar a una persona de la tabla de clientes, habría que borrarla de la tabla de ventas, ya que no pueden existir ventas de un cliente inexistente.
- Modificar el campo de relación en la tabla principal si tiene registros relacionados en la tabla relacionada.
- Modificar el campo de relación en la tabla relacionada si el nuevo valor que se indexa en el no existe en la tabla principal, ya que este debe estar también en la tabla principal.

Si exigimos integridad referencial se van a activar las siguientes opciones:

- **Actualizar en cascada los campos relacionados:** Al activar esta opción, si realizamos alguna modificación en el campo clave principal de la tabla principal, Access realizara una actualización automática en el campo relacionado de la tabla relacionada.

- **Eliminar en cascada los registros relacionados:** Si eliminamos un registro en la tabla principal, se eliminaran automáticamente todos los registros relacionados en las demás tablas relacionadas.

INVESTIGAR, TIPOS DE DATOS PARA LOS CAMPOS DE UNA TABLA

Recordar que:

Una tabla esta estructurada en filas y columnas. Cada fila es un registro de datos, ya que contiene datos. Los datos son la intersección de una fila con una columna y es la unidad mínima con la que podemos trabajar en una base

de datos. El gestor Access colocará los datos en una matriz donde las filas serán los registros y las columnas los campos.

***Clave Principal*:** Campo de la tabla que identifica inequívocamente un registro, no pudiendo existir dos registros con la misma clave principal. Esta clave sirve para identificar un registro en concreto. También llamada Primary Key.

El campo clave no puede tener un valor nulo (NULL). Su valor debe ser conocido. El contenido del campo no deber ser extenso, el espacio que ocupe deberá ser mínimo. Su utilización en otras tablas (Claves Ajenas) sirve para crear una relación entre ellas.

***Clave Ajena o foránea*:** Duplicación en la tabla relacionada de la clave principal. Si el campo definido como clave principal o primaria en una tabla forma parte también de otra tabla se llamará clave ajena. También llamada Foreign Key.

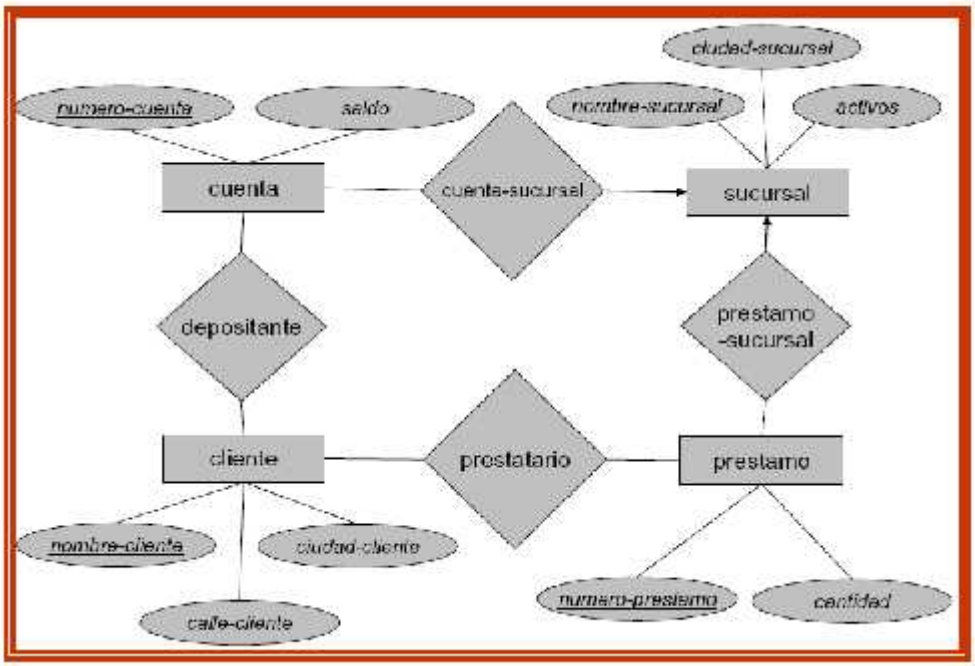
- ✓ Es muy conveniente que cada tabla contenga una clave principal.

INVESTIGAR las 12 reglas de CODD

2.2.6 Conversión de esquemas E-R a relaciones

- ✓ Las claves primarias permiten representar tanto los conjuntos entidad como los conjuntos asociación como *relaciones* que representan los contenidos de la base de datos.
- ✓ Una base de datos que sigue el esquema E-R se puede representar mediante un conjunto de relaciones.
- ✓ Para cada conjunto entidad y cada conjunto asociación existe una única relación a la que se le asigna el nombre del conjunto entidad o conjunto asociación correspondiente.
- ✓ Cada relación tiene columnas (normalmente una por atributo), que tienen nombres únicos.
- ✓ Convertir un diagrama E-R a relaciones es la base para conseguir un diseño relacional a partir de ese diseño E-R.

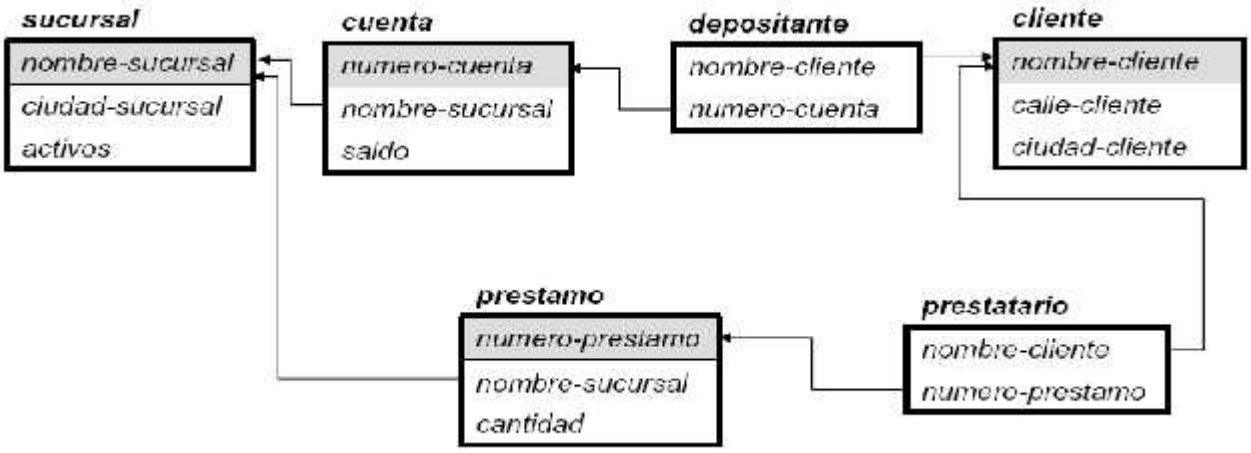
Ejemplo, Diagrama E-R para entidad bancaria



La relación *cliente*

nombre_clente	calle_clente	ciudad_cliente
Suárez López Luís	Principe	Mexico
Pazo Rial		Tijana

Diagrama del esquema para la entidad Bancaria



2.2.7 Pasos del modelo E-R al modelo relacional

Transformación de las entidades
Todas las entidades regulares presentes en el modelo E/R se transforman en tablas en el modelo relacional, manteniendo el número y tipo de los atributos, así como las claves primarias.

Las entidades débiles también se convierten en tablas en el modelo relacional, manteniendo el número y tipo de los atributos, pero su clave primaria se forma por la composición de su clave primaria con la clave primaria de la entidad reglar de la cual depende.

Transformación de las relaciones uno a uno (1:1)

Si en la relación binaria, las dos entidades participan con cardinalidad máxima y mínima igual a uno, entonces:

- ✓ **Si las dos entidades tienen el mismo identificador**, entonces se transforman en una única tabla por la agregación de los atributos de las dos entidades y la clave es la clave de las entidades (es la misma en ambas).

- ✓ **Si las dos entidades tienen distinto identificador**, entonces cada entidad se transforma en una tabla con clave principal el identificador de la entidad correspondiente y cada tabla tendrá como clave ajena el identificador de la otra tabla con la cual está relacionada.

Si en la relación binaria, alguna de las entidades participa con cardinalidad mínima igual a cero, entonces:

- ✓ Cada entidad se transforma en una tabla con clave principal el identificador de la entidad correspondiente.
- ✓ Se construye una nueva tabla correspondiente a la relación, la clave de la misma estaría formada por las claves de cada tabla y los atributos de la relación (si los hay).

Transformación de las relaciones uno a varios (1:N)

Si en la relación binaria 1: N, la entidad que participa con cardinalidad máxima uno, lo hace también con cardinalidad mínima uno, entonces cada entidad se transforma en una tabla con clave principal el identificador de la entidad correspondiente y la clave de la entidad que participa con cardinalidad máxima uno pasa como clave ajena de la otra tabla con la cual está relacionada. Si la relación tuviera atributos, estos pasan a formar parte de la tabla correspondiente a la entidad que participa con cardinalidad máxima N.

Si en la relación binaria 1: N, la entidad que participa con cardinalidad máxima uno, lo hace con cardinalidad mínima cero, entonces cada entidad se transforma en una tabla con clave principal el identificador de la entidad correspondiente y se construye una nueva tabla correspondiente a la relación, formada por las claves de cada tabla y los atributos de la relación. La clave de esta nueva tabla será el identificador de la entidad que participa con cardinalidad máxima N y tendría como clave ajena el identificador de la otra entidad.

Las relaciones débiles no sufren ningún tipo de transformación, simplemente desaparecen en el modelo relacional.

Transformación de las relaciones varios a varios (N: M)

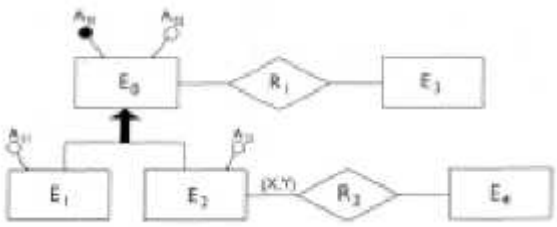
En la relación binaria N:M, cada entidad se transforma en una tabla con clave principal el identificador de la entidad correspondiente y se construye una nueva tabla correspondiente a la relación, que tendrá los atributos correspondientes a la relación y cuya clave estará formada por la composición de los identificadores de las entidades que participan en la relación.

Transformación de las relaciones N-arias

En las relaciones N-arias se aplica la misma regla que en las relaciones binarias N: M.

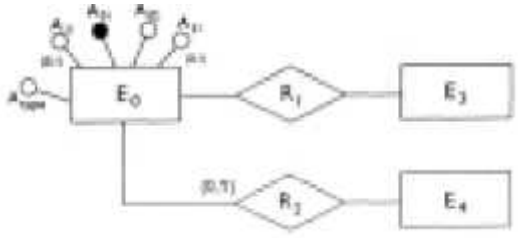
2.2.7.1 Eliminación de generalizaciones

El modelo relacional no puede representarlas directamente.



Principales alternativas para transformar generalizaciones:

1 Plegar las entidades hijo en la entidad padre.
Adecuada cuando las operaciones implican a los atributos de E0, E1 y E2 más o menos de la misma forma. Aunque mayor espacio de almacenamiento, requiere menos accesos.

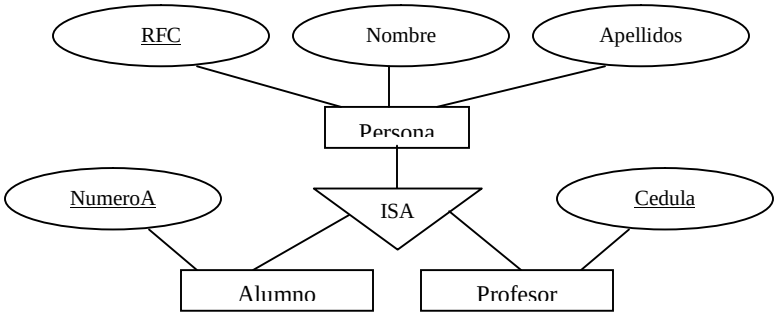


2 Sustituir la generalización por relaciones.

Adecuada cuando la generalización no es total y las operaciones se refieren a atributos de E1(E2) o de E0 (se distinguen los padres de los hijos). Se ahorra espacio con respecto a la alternativa 1 (no hay atributos NULL), pero hay más accesos para mantener la consistencia.

Ejemplo de generalizaciones

- ✓ Se tratan igual que en el caso de las entidades débiles.
- ✓ La relación **ISA** no se transforma en relación.



Personas (RFC, Nombre, Apellidos);
Alumnos (**RFC**, NumeroA);
Profesores (**RFC**, Cedula);

EL ALUMNO INVESTIGARÁ Y EXPONDRÁ LOS SIGUIENTES MODELOS DE BASES DE DATOS. SE PODRÁ APOYAR DEL LIBRO PETER ROB/ CARLOS CORONEL, "SISTEMAS DE BASES DE DATOS", E.D. THOMSON.

2.3 Modelo de redes
2.4 Modelo jerárquico
2.5 Modelo de archivos
2.6 Modelo de thesaurus

UNIDAD III. NORMALIZACIÓN.

Objetivo: Se comprenderá la importancia del proceso de Normalización de las tablas que componen una base de datos y lo aplicará.

3 Introducción

La *normalización,* o sea la razón y uso de las formas normales, es evitar la repetición innecesaria de datos (redundancia). Una solución a este problema es repartirlos en varias relaciones y utilizar referencias por valor entre ellas.

La traducción del esquema conceptual al lógico no es única. Es necesario contar con una medida de la calidad de la agrupación de los atributos en relaciones. Como herramienta principal se usan las dependencias funcionales para agrupar los atributos en esquemas de relación, que se dice que se encuentran en una determinada forma normal. La forma normal de un esquema de relación determina su grado de calidad con respecto a reducir dos efectos no deseados: la redundancia de datos y las anomalías que produce esta redundancia. En este tema se estudia el análisis de esquemas de relación para determinar en qué forma normal se encuentra y el procedimiento, conocido como normalización, para descomponerlos en otros esquemas de relación que se encuentren en formas normales más exigentes.

3.1 Dependencias funcionales

Uno de los retos en el diseño de la base de datos es el de obtener una estructura estable y lógica tal que:
1. El sistema de base de datos no sufra de anomalías de almacenamiento.
2. El modelo lógico pueda modificarse fácilmente para admitir nuevos requerimientos.

La teoría de normalización esta basada en la *teoría de las dependencias*; se dice que una relación está en una determinada forma normal si satisface un conjunto de restricciones.

Dada una relación R, se dice que el atributo **R.y** £ **R** es funcionalmente dependiente de otro atributo **R.x** £ **R**, y se expresa en la forma **R.x** ⟶ **R.y** si, y sólo si, siempre que dos o más tuplas de R coincidan en sus valores de R.x.

Representación textual de las dependencias funcionales
Formato general
R.x —> *(R.a, R.b, ... R.n)*
denotando que los atributos *R.a, R.b, ... R.n* son funcionalmente dependientes del atributo *R.x.*

EJEMPLO
Relación Alumno

matricula	*nombre*	*apellidos*	nota	*curso*

Relación *Alumno*
Alumno.matricula —> (Alumno.nota, Alumno.*curso)*
Alumno.matricula —> (Alumno.nombre, Alumno.apellidos)

A los atributos que forman parte de la clave de una relación se les denomina **atributos *primos***, *y* a los que no forman parte de la clave, **atributos *no primos.***

En los esquemas se representarán: los atributos que forman la clave primaria de las relaciones subrayados, y Los atributos que son ***claves foráneas*** de otras relaciones **en negrita**.

3.1.1 Formas normales
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:

3.2 Primera forma normal FN1

Una relación **R** satisface la primera forma normal (FN1) si, y sólo si, todos los dominios subyacentes de la relación **R** contienen valores atómicos.

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

1. 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.
2. Todos los ingresos en cualquier columna (atributo) deben ser del mismo tipo.
3. Cada columna debe tener un nombre único, el orden de las columnas en la tabla no es importante.
4. 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.

Matrícula

(*)asignatura	N(12)	Representa la identificación de las asignaturas en las que se encuentra matriculado cada uno de los alumnos.
cuenta	N(10)	Representa el número de cuenta del alumno.
calificacion	N(10)	Representa la calificación del alumno.
nombre	T(18)	
Apellido_p	T(18)	
Apellido_m	T(18)	
semestre	T(18)	Representa el semestre en el que se imparte la clase de una asignatura
aula	N(50)	Representa las aulas en las que se imparte la clase de las asignaturas.
lugar	T(18)	Representa los lugares de estudio en los que se imparte la clase correspondiente a las asignaturas.

asignatura	cuenta	nombre	Apellido_p	Apellido_m	semestre	calificacion	aula	lugar
001	001	LUIS	PEREZ	LOPEZ	sexto	7	20	ED.CENTRAL

Esquema:

Matrícula (<u>asignatura</u>, cuenta, nombre, Apellido_p, Apellido_m, curso, aula, lugar)

* Problemas de inserción, borrado y actualización de tuplas

3.3 Segunda forma normal FN2

Una relación R satisface la segunda forma normal (FN2) si, y sólo si, satisface la primera forma normal y cada atributo de la relación depende funcionalmente deforma completa de la clave primaria de esa relación.

Esquema:

Imparte (asignatura, semestre)
Alumno (cuenta, Apellido_p, Apellido_m, nombre)
Matricula (**asignatura, cuenta,** calificación, aula, lugar)

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 está definición, cada tabla que tiene un atributo único como clave, esta en segunda forma normal.

3.4 Tercera forma normal FN3

Una relación R satisface la tercera forma normal (FN3) si, y sólo si, satisface la segunda forma normal y cada atributo no primo de la relación no depende funcionalmente de forma transitiva de la clave primaria de esa relación. Es decir, no pueden existir dependencias entre los atributos que no forman parte de la clave primaría de la relación R.

Esquema:

Imparte (asignatura, semestre)
Alumno (cuenta, Apellido_p, Apellido_m, nombre)
Ubicación (aula, lugar)
Matricula (**asignatura, cuenta,** calificación, **aula**)

INVESTIGAR Y DAR UN EJEMPLO UTILIZANDO LA CUARTA Y QUINTA FORMA NORMAL

UNIDAD IV METODOLOGÍAS PARA EL DISEÑO.

***Objetivo*:** Analizará la metodología para el diseño de bases de datos.

Contenidos Teóricos:
4.1. Metodología del Diseño
4.1.1 Concepto
4.1.2 Etapas de diseño
4.1.3 Técnicas de Diseño
4.2. Conceptos

Dentro de un gran sistema de información, la base de datos también esta sometida a un ciclo de vida. **El ciclo de vida de Base de Datos (DBLC).** Contiene seis fases. Estudio inicial de la base de datos, Diseño de la base de datos, Ejecución y carga, Pruebas y evaluaciones, Operación, Mantenimiento y evaluación. Ver figura 4.1.

4 Diseño lógico y diseño de implantación del ciclo de vida de desarrollo de base de datos

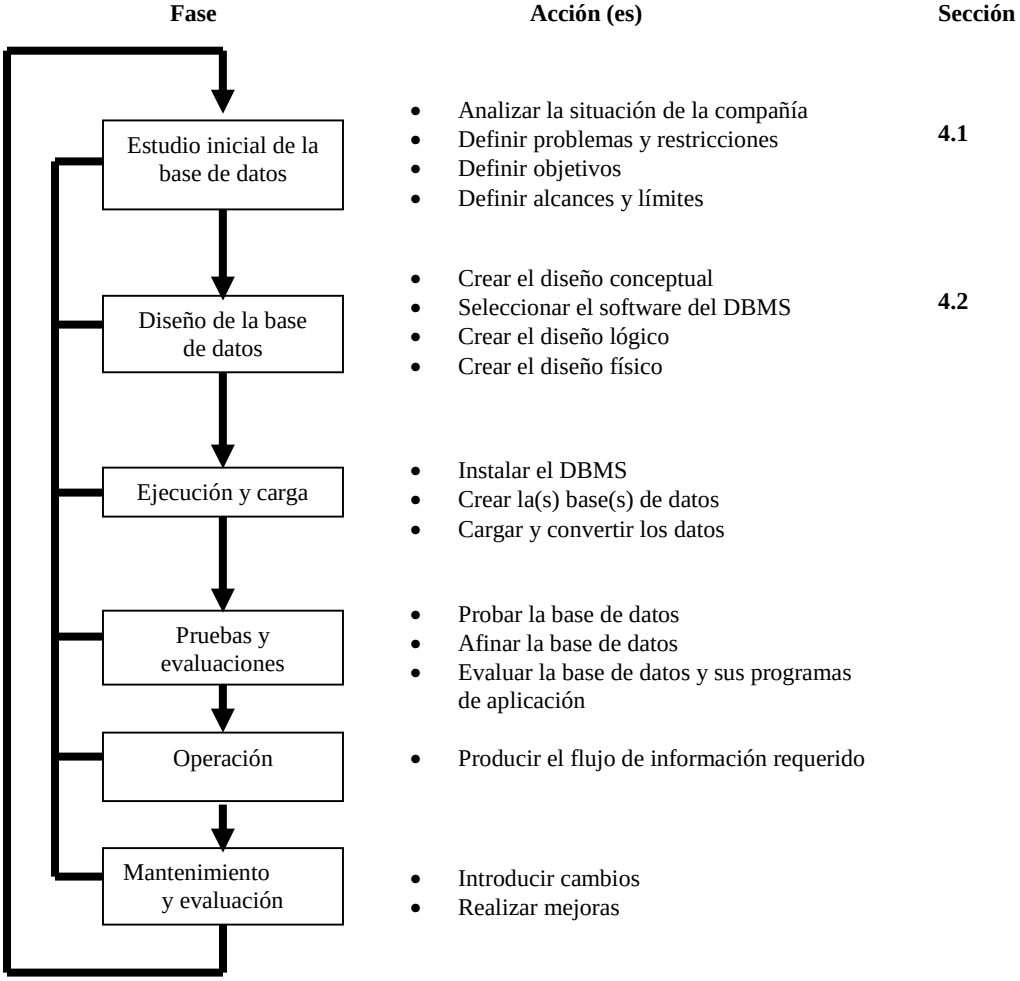


Figura 4.1.

4.1 Estudio inicial de la base de datos

La figura 4.2. Ilustra los procesos interactivos e iteractivos requeridos para completar la primera fase del DBLC con éxito.

Para analizar la situación de la compañía el diseñador de la base de datos debe descubrir cuáles son los componentes operativos de la compañía, cómo funcionan y cómo interactúan.

Las siguientes cuestiones deben resolverse

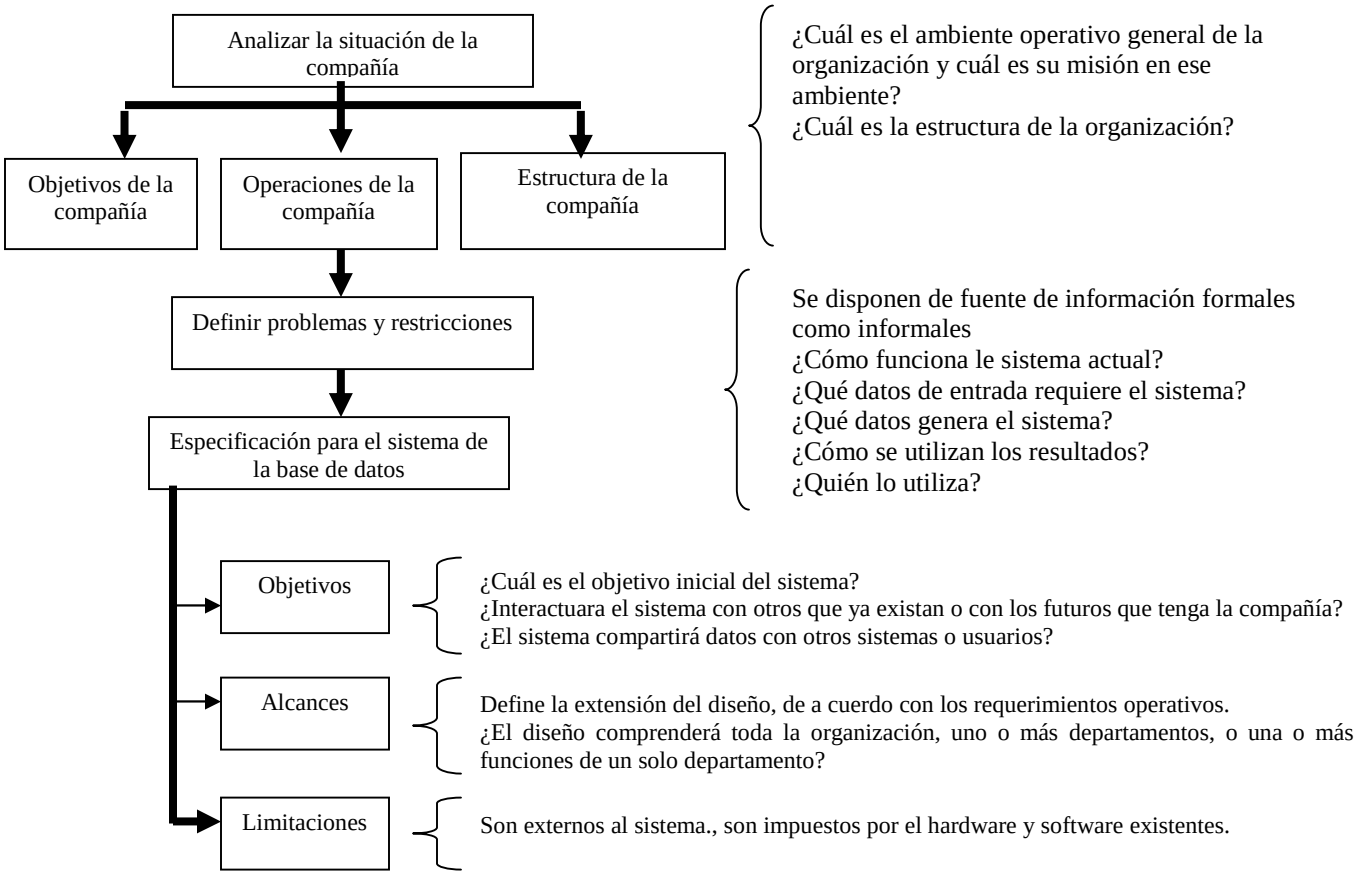


Figura 4.2 Resumen de actividades en el estudio inicial de una base de datos

Un sistema de base de datos propuesto debe diseñarse para que ayude a resolver por lo menos los problemas más importantes identificados durante el proceso de descubrimiento del problema.

4.2 Diseño de la base de datos

La segunda fase se enfoca en el diseño del modelo de base de datos que soportará las operaciones y objetivos de la compañía. Ésta es la fase del DBLC más crítica: asegúrese de que el producto satisfaga los requerimientos del usuario y del sistema. En suma, se tienen dos visualizaciones de los datos dentro del sistema: la visualización de los datos de la empresa con fuente de información y la del diseñador de la estructura de los datos, su acceso y las actividades requeridas para transformarlos en información. La figura 4.3 compara estas visualizaciones.

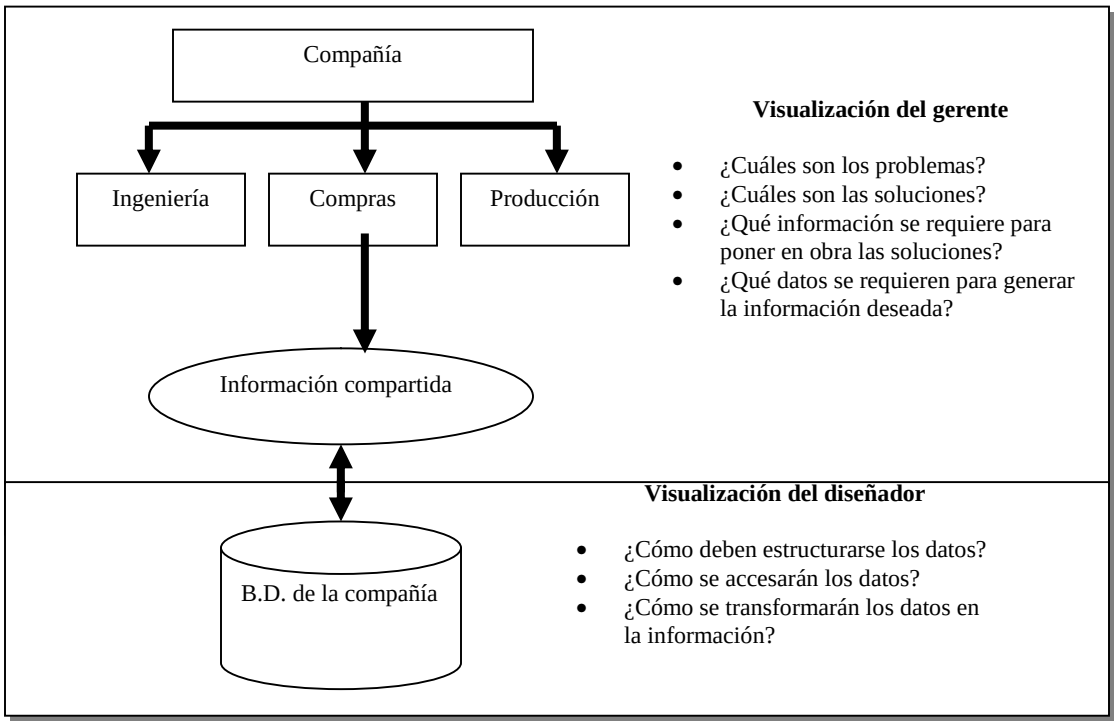
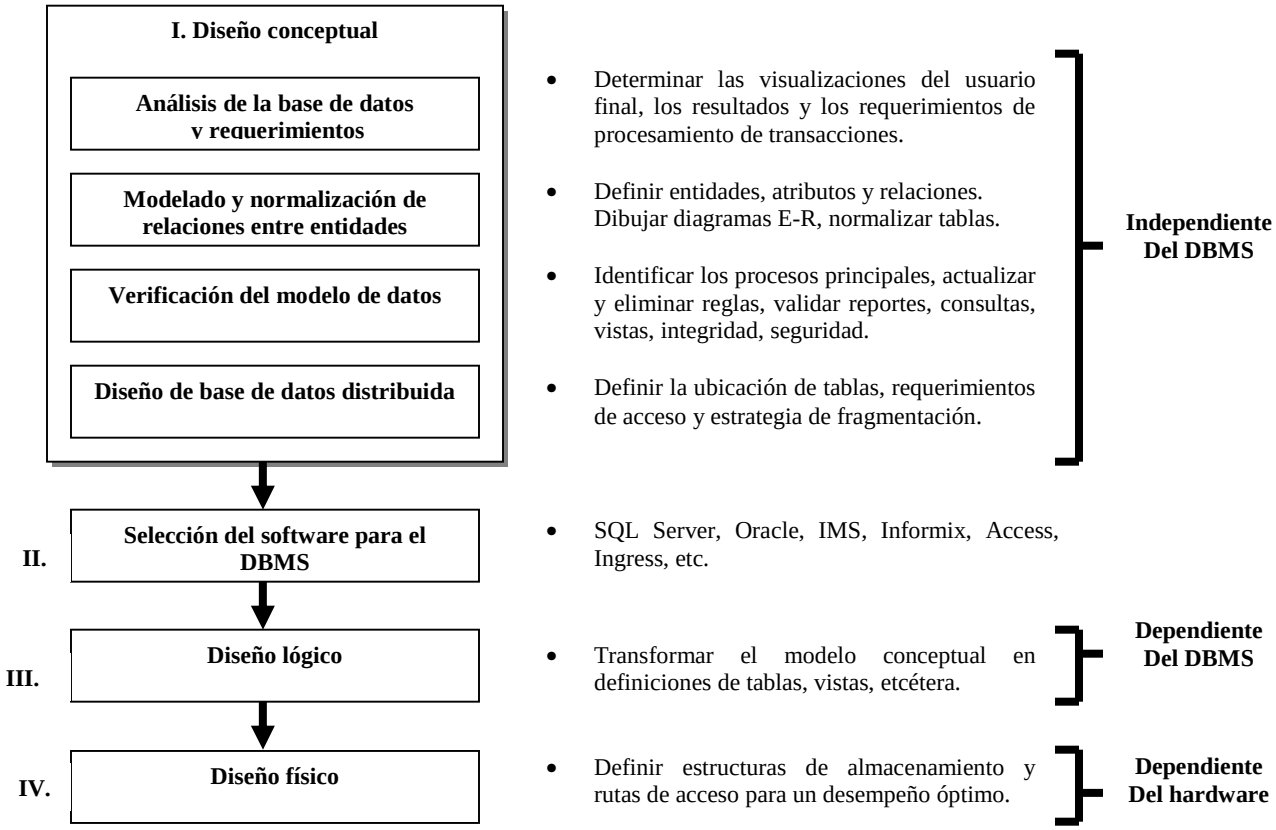


Figura 4.3 Dos visualizaciones de los datos: gerente de la compañía y diseñador

La figura 4.4 Flujo de procedimiento en el diseño de una base de datos.



4.2.1 Diseño conceptual

En la etapa de diseño conceptual, se utiliza el modelado de datos para crear estructuras de base de datos abstractas que representen objetos del mundo real de la manera más real posible. El modelo conceptual debe dar cuerpo a un claro entendimiento de la empresa y a sus áreas funcionales. A este nivel de abstracción, el tipo de hardware o de modelo de base de datos, o ambos a ser utilizados, es posible que aún no puedan ser identificados. Por consiguiente, el diseño debe ser independiente del software y del hardware, de modo que el sistema pueda ser configurado más adelante con cualquier hardware y plataforma de software que se elija.
Tome en cuenta la siguiente **regla de datos mínima:**

Todo lo que se requiere está allí, y todo lo que está allí se requiere.

La figura 4.5. Resume las interacciones en el proceso de modelado E-R. la figura 4.6. Resume el arreglo de herramientas de diseño y fuentes de información que el diseñador puede utilizar para producir el modelo conceptual

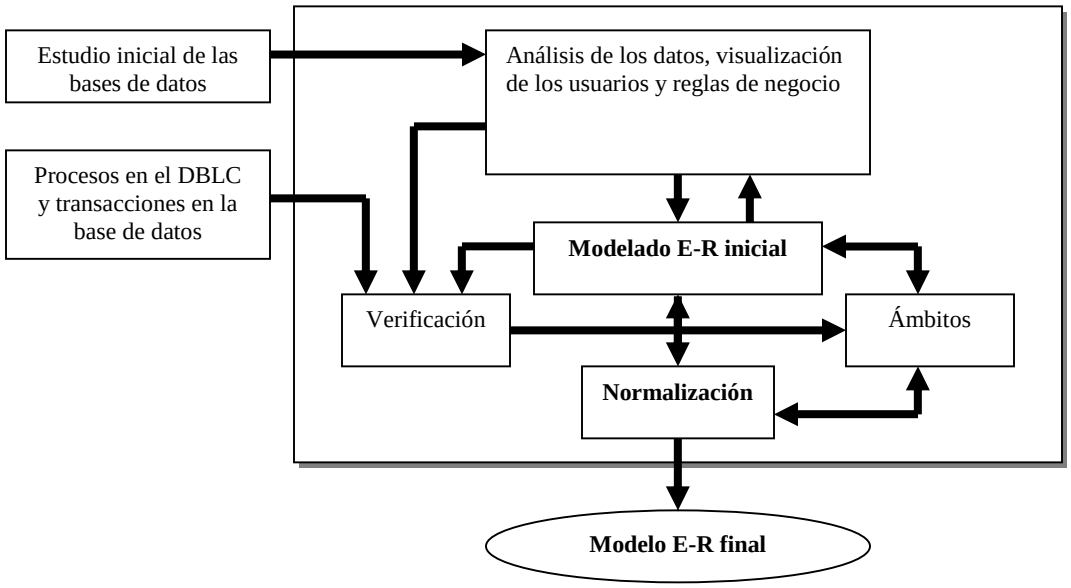


Figura 4.5 El modelo E-R es un proceso iteractivo basado en muchas actividades

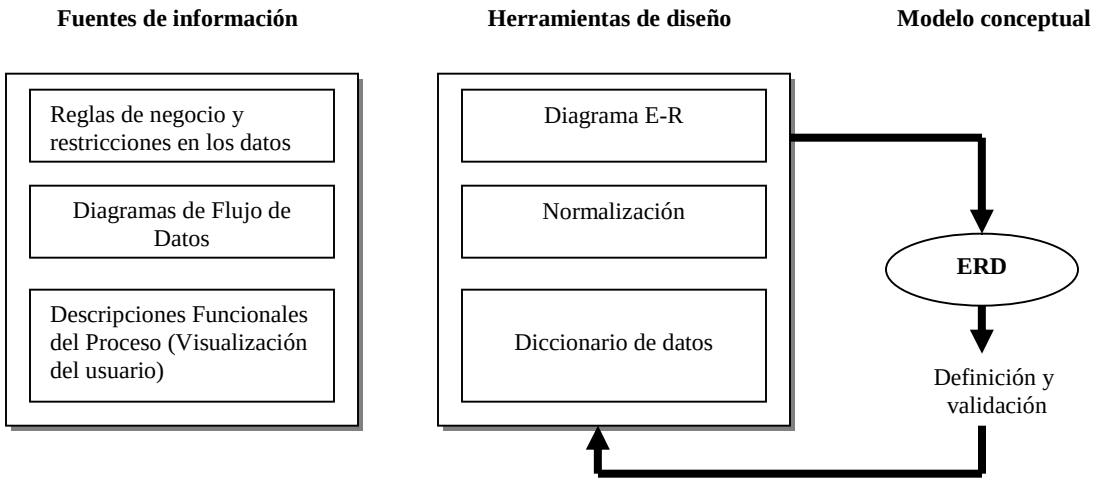


Figura 4.6 Herramientas de diseño conceptual y fuentes de información

Todos los objetos (entidades, atributos, relaciones, vistas, etcétera) están definidos en el diccionario de datos, el cual se utiliza junto con el proceso de normalización para eliminar las anomalías de datos y los problemas de redundancia. Durante este proceso de modelado E-R, el diseñador debe:

- Definir entidades, atributos, claves principales y claves foráneas (las claves foráneas sirven como base para las relaciones entre las entidades).
- Tomar decisiones sobre la adición de atributos nuevos de la clave principal para satisfacer los requerimientos del usuario final o de procesamiento, o ambos.
- Tomar decisiones sobre el tratamiento de atributos de valores multivaluados.
- Tomar decisiones sobre la adición de atributos derivados para satisfacer los requerimientos de procesamiento.
- Tomar decisiones sobre las localizaciones de clave foránea en relaciones 1:1.
- Evitar relaciones ternarias innecesarias.
- Dibujar el diagrama E-R correspondiente.
- Normalizar el modelo de datos.
- Incluir todas las definiciones de elemento de datos en el diccionario de datos.
- Tomar decisiones sobre convenciones de nominación estándar.

Proceso de verificación

Tome en cuenta que el proceso de verificación requiere la verificación continua de las transacciones de negocio, así como también los requerimientos del sistema y del usuario. La secuencia de verificación debe repetirse por cada uno de los módulos del sistema. La figura 4.7 ilustra la naturaleza iteractiva del proceso.

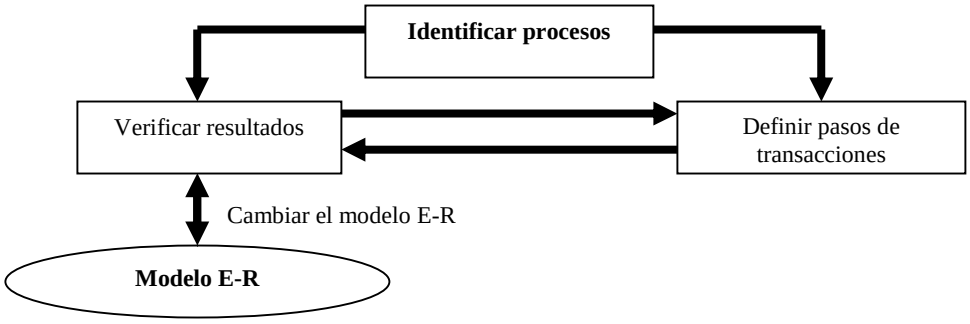


Figura 4.7 Proceso iteractivo de verificación de modelo E-R

RESUMEN

Los datos constituyen la materia prima con la que se crea la información. La transformación de los datos en información se produce cuando el código de programación opera en los datos, con lo que se producen aplicaciones.

Un sistema de información se diseña para facilitar la transformación de los datos en información y para manejar tanto los datos como la información. La base de datos es, por lo tanto, una parte muy importante del sistema de información. El análisis de sistemas es el proceso que establece la necesidad y la extensión de un sistema de información. El desarrollo de sistemas es el proceso de crear un sistema de información.

El Ciclo de Vida del Desarrollo de Sistemas (SDLC) rastrea el historial (ciclo de vida) de una aplicación dentro del sistema de información. El SDLC se divide en cinco fases: planificación, análisis, diseños de sistemas detallados, ejecución y mantenimiento. El SDLC es un proceso iterativo y no uno secuencial.

El Ciclo de Vida de Base de Datos (DBLC) describe el historial de la base de datos dentro del sistema de información. El DBLC se compone de seis fases: el estudio inicial de la base de datos, el diseño de la base de datos, ejecución y carga, pruebas y evaluaciones, operación, mantenimiento y evolución. Al igual que el SDLC, el DBLC es iterativo y no secuencial.

Como la base de datos es el componente más básico del sistema de información, el proceso de su diseño y ejecución es de particular interés. El proceso de diseño y ejecución pasa por una serie de etapas bien definidas:
1. El análisis de los datos y la recopilación de los requerimientos ayudan a determinar las visualizaciones de los usuarios y las necesidades de datos.

2. El producto final de la Etapa 1 produce la definición de entidades, atributos y relaciones pertinentes, el que conduce a su vez al modelo de relaciones entre entidades. Después, los componentes del modelo E-R dan lugar a un conjunto de tablas normalizadas.

3. El modelo conceptual se verifica mediante la identificación de sus procesos principales y mediante la definición de la reglas INSERT, UPDATE y DELETE apropiadas. Además, las visualizaciones del usuario y las restricciones en los datos apropiadas se revisan durante el proceso de verificación.

4. Después de la verificación, se analiza un diseño de base de datos distribuida para evaluar la ubicación de las tablas, las estrategias de fragmentación de la base de datos y los requerimientos de acceso.

5. Idealmente, la selección del software para la base datos se realiza junto con el desarrollo del modelo conceptual (Etapas 1 a 4). En el mundo práctico, menos perfecto, podría descubrirse que el software para la base de datos ya está en su lugar y el diseño conceptual es parcialmente dictado por la existencia de ese software.

6. Dados el modelo conceptual y el software para la base de datos, el diseño lógico transforma el esquema conceptual en la definición específica del DBMS de tablas, vistas, etcétera. El diseño lógico es, por lo tanto, una función de las características del DBMS seleccionado, aunque es independiente del hardware.

7. El diseño físico define estructuras de almacenamiento y rutas de acceso específicas a los datos y, por consiguiente, es independiente del hardware.

8. La ejecución constituye la etapa final, en la que se cargan los datos, se crean las tablas y visualizaciones. Si es necesario, los formatos de datos de atributo se transforman en formatos que puedan ser accesados con eficiencia por el DBMS seleccionado. La codificación y pruebas de las aplicaciones se completan durante esta etapa.

La parte conceptual del diseño puede someterse a algunas variaciones con base en dos filosofías de diseño.

Ascendente contra descendente. El diseño ascendente primero identifica los elementos de datos (atributos) y luego, los agrupa en conjuntos (entidades). El diseño descendente primero identifica las entidades y luego, define los atributos de cada entidad. La selección de uno u otro método depende del alcance del problema de datos y de las preferencias personales. Sin embargo, como una regla general, el método descendente es superior en un ambiente de datos complejo.

Centralizado contra descentralizado. Con el método centralizado las bases de datos relativamente simples son fáciles de crear cuando las operaciones de la compañía se prestan para una "vista general" de la base de datos. El método de diseño descentralizado puede ser más apropiado cuando las operaciones de la compañía se esparcen a través de sitios operativos múltiples (cada conjunto de datos de un sitio es un subconjunto de todas las operaciones rde la compañía) o cuando la base de datos se compone de un número de entidades muy grande que se somete a relaciones muy complejas.

UNIDAD V ANALIZAR LA IMPLEMENTACIÓN DE BASES DE DATOS A TRAVÉS DE LOS DIVERSOS MODELOS DE BASES DE DATOS.

***Objetivo*:** Diseño e implementación de base de datos utilizando SQL.

Contenidos Teóricos:
5.1 Diseño empleando el modelo relacional
5.2 Criterios del diseño
5.3 Procedimiento del diseño
5.4 Diccionario de Datos
5.5 Algebra Relacional
5.6 Implementación
5.7 Procesamiento de transacciones
5.8 Lenguaje de Datos
5.9 Lenguaje de Manipulación de datos

EL ALUMNO FOTOCOPIARA EL CAPÍTULO 7, LABORATORIO UNIVERSITARIO: DISEÑO CONCEPTUAL DEL LIBRO PETER ROB/ CARLOS CORONEL, "SISTEMAS DE BASES DE DATOS", THOMSON.

5.8 Lenguajes de bases de datos

EL ALUMNO INVESTIGARA HISTORIA DE SQL

Una vez finalizado el diseño de una base de datos y escogido un DBMS para su implementación, el primer paso consiste en especificar el esquema conceptual y el esquema interno de la base de datos.

Un sistema de bases de datos proporciona un **lenguaje de definición de datos (DDL)** para especificar el esquema conceptual de la base de datos y un **lenguaje de manipulación de datos (DML)** para expresar las consultas a la base de datos y las modificaciones. En la practica, los lenguajes de definición y manipulación de datos no son lenguajes separados., simplemente forman partes de un único lenguaje de bases de datos, como es SQL.

SQL	**DDL**	**CREATE**	**Crea** tablas, campos e índices
		ALTER	**Modifica** tablas, agregando o cambiando la definición de los campos
		DROP	**Elimina** tablas e índices
	DML	**SELECT**	Consulta registros
		INSERT	Inserción de información en la B.D
		UPDATE	Modifica valores de los campos y registros
		DELETE	Elimina registros de una tabla de una B.D

El **DDL. lenguaje de definición de datos**(Data Definition Language) es aquel que permite describir un esquema de base de datos. Las definiciones resultantes conformaran al DICCIONARIO DE DATOS.

Un **DICCIONARIO DE DATOS** es un archivo que contiene metadatos que se consulta antes de leer o modificar datos reales en el sistema de base de datos.
Permiten crear, las bases de datos, las tablas, definir índices y reglas de integridad. Igualmente modificar y borrar lo antes definido.

Ejemplo

CREATE TABLE *prestamo*
(numero- prestamo **integer,** importe **integer)**

5.9 Lenguaje de Manipulación de Datos

A diferencia del anterior este tiene estrecha relación con las operaciones que los usuarios realizan sobre los datos almacenados.

El DML (Data Manipulation Language) nos sirve para manejar la información contenida en la base de datos. Este manejo consiste básicamente en la inserción, consulta, eliminación y modificación de la información.

El DML aplicado a nivel físico será utilizado para realizar procesos que permitan un acceso más eficiente a la información; en el nivel de visión tendrá como finalidad mostrar al usuario los datos en una forma clara y sencilla.

Existen dos tipos de DML:

DE PROCEDIMIENTOS.- Requieren que el usuario especifique que datos se necesitan y cómo obtener esos datos.

Esto quiere decir que el usuario debe especificar todas las operaciones de acceso a datos llamando a los procedimientos necesarios para obtener la información requerida. Estos lenguajes acceden a un registro, lo procesan y basándose en los resultados obtenidos, acceden a otro registro, que también deben procesar. Así se va accediendo a registros y se van procesando hasta que se obtienen los datos deseados. Las sentencias de un LMD procedural deben estar embebidas en un lenguaje de alto nivel, ya que se necesitan sus estructuras (bucles, condicionales, etc.) para obtener y procesar cada registro individual. A este lenguaje se le denomina *lenguaje anfitrión*.

SIN PROCEDIMIENTOS.- Requieren que el usuario especifique que datos se necesitan sin especificar cómo obtener esos datos.

Los **DML** de procedimientos son mucho más eficientes en lo que respecta a sus capacidades de manejo y control de la información, pero su complejidad es mayor.

SQL es el lenguaje de consulta más ampliamente usado y es un lenguaje no procedimental.

EJEMPLO

SELECT saldo **FROM** *prestamo.,*
INSERT INTO *prestamo* (numero-prestamo)
VALUES (‘3’) .,
UPDATE *prestamo* **SET** *saldo*=10.,
DELETE importe **FROM** *prestamo.,*

Programas de aplicación: Son programas que se usan para interaccionar con la base de datos. los programas de aplicación se escriben usualmente en un lenguaje anfitrión, tal como, c, c++, java, basic, entre otros.

Los programas de aplicación normalmente acceden a bases de datos mediante

- Extensiones de lenguaje que permiten embeber SQL.
- Interfaces de programación de aplicaciones (p.e. ODBC/JDBC) que permiten enviar consultas SQL a una base de datos.

5.9.1 Componentes del SQL

El lenguaje SQL está compuesto por comandos, cláusulas, operadores y funciones de agregado. Estos elementos se combinan en las instrucciones para crear, actualizar y manipular las bases de datos.

5.9.2 Comandos

Existen dos tipos de comandos SQL:

- **DDL** que permiten crear y definir nuevas bases de datos, campos e índices.
- **DML** que permiten generar consultas para ordenar, filtrar y extraer datos de la base de datos.

Comandos DDL

CREATE: Utilizado para crear nuevas tablas, campos e índices
DROP: Empleado para eliminar tablas e índices
ALTER: Utilizado para modificar las tablas agregando campos o cambiando la definición de los campos

Comandos DML

SELECT: Utilizado para consultar registros de la base de datos que satisfagan un criterio determinado
INSERT: Utilizado para cargar lotes de datos en la base de datos en una única operación
UPDATE: Utilizado para modificar los valores de los campos y registros especificados
DELETE: Utilizado para eliminar registros de una tabla de una base de datos

5.9.2.1 Cláusulas

FROM	WHERE	GROUP BY	HAVING	ORDER BY

5.9.2.2 Operadores Lógicos

AND	OR	NOT

5.9.2.3 Operadores de Comparación

< Menor que	> Mayor que	<> Distinto de	<= Menor ó Igual que	>= Mayor ó Igual que

= Igual que	BETWEEN: Utilizado para especificar un intervalo de valores.	LIKE: Utilizado en la comparación de un modelo

In: Utilizado para especificar registros de una base de datos

5.9.2.4 Funciones de Agregado

Las funciones de agregado se usan dentro de una cláusula SELECT en grupos de registros para devolver un único valor que se aplica a un grupo de registros.

AVG: Utilizada para calcular el promedio de los valores de un campo determinado.
COUNT: Utilizada para devolver el número de registros de la selección.

SUM: Utilizada para devolver la suma de todos los valores de un campo determinado.
MAX: Utilizada para devolver el valor más alto de un campo especificado.
MIN: Utilizada para devolver el valor más bajo de un campo especificado.

5.9.2.5 Consultas con Predicado

El predicado se incluye entre la cláusula y el primer nombre del campo a recuperar, los posibles predicados son:
ALL: Devuelve todos los campos de la tabla.
TOP: Devuelve un determinado número de registros de la tabla.
DISTINCT: Omite los registros cuyos campos seleccionados coincidan totalmente.
DISTINCTROW: Omite los registros duplicados basándose en la totalidad del registro y no sólo en los campos seleccionados.

UNIDAD VI ADMINISTRACIÓN Y SEGURIDAD.

Objetivo: Identificar los conceptos sobre administración y seguridad en las bases de datos.

Contenidos Teóricos:
6.1 Gestión de base de datos
6.2 Funciones del ADB (administrador de base de datos)
6.3 Objetivos del ADB
6.4 Integridad de la base de datos
6.5 Seguridad de la base de datos
6.6 Recuperación de la base de datos

6.1 Gestión de base de datos

El alcance de la actividad de la **Administración de Datos** es la organización completa (empresa, institución u otro organismo), mientras que el alcance de la **Administración de Bases de Datos** queda restringido a una Base de Datos en particular y a los sistemas que los procesan. La Administración de la Base de Datos opera dentro de un marco proporcionado por la Administración de Datos facilitándose de esta manera el desarrollo y el uso de una Base de Datos y sus aplicaciones. **Las siglas DBA suelen utilizarse para designar tanto la función Administración de Base de Datos como al titulo del puesto administrador de Base de Datos.**

La responsabilidad general del DBA es facilitar el desarrollo y el uso de la Base de Datos dentro de las guías de acción definidas por la administración de los datos.

6.2 Funciones del administrador de bases de datos (DATE)

DEFINIR EL ESQUEMA CONCEPTUAL: es tarea del administrador de datos decidir con exactitud cual es la información que debe mantenerse en la base de datos, es decir, identificar las entidades que interesan a la empresa y la información que debe registrarse acerca de esas entidades. Este proceso por lo general se denomina diseño lógico –a veces conceptual- de bases de datos. Cuando el administrador de datos decide el contenido de la base de datos en un nivel abstracto, el DBA crea a continuación el esquema conceptual correspondiente, empleando el DDL conceptual. El DBMS utilizará la versión objeto (compilada) de ese esquema para responder a las solicitudes de acceso. La versión fuente sin compilar servirá como documento de referencia para los usuarios del sistema.

DEFINIR EL ESQUEMA INTERNO: el DBA debe decidir también como se representará la información en la base de datos almacenada. A este proceso suele llamársele diseño físico de la base de datos. Una vez hecho esto el DBA deberá crear la definición de estructura de almacenamiento correspondiente (es decir el esquema interno) valiéndose del DDL interno. Además deberá definir la correspondencia pertinente entre los esquemas interno y conceptual. En la práctica, ya sea el DDL conceptual o bien el DDL interno incluirán seguramente los medios para definir dicha correspondencia, pero las dos funciones (crear el esquema, definir la correspondencia) deberán poder separarse con nitidez.

VINCULARSE CON LOS USUARIOS: el DBA debe encargarse de la comunicación con los usuarios, garantizar la disponibilidad de los datos que requieren y escribir - o ayudar a los usuarios a escribir- los esquemas externos necesarios, empleando el DDL externo aplicable. Además, será preciso definir la correspondencia entre cualquier esquema externo y el esquema conceptual. En la práctica, el DDL externo incluirá con toda probabilidad los medios para especificar dicha correspondencia, pero en este caso también el esquema y la correspondencia deberán poder separarse con claridad. Cada esquema externo y la correspondencia asociada existirán en ambas versiones fuentes y objeto.

DEFINIR LAS VERIFICACIONES DE SEGURIDAD E INTEGRIDAD: las verificaciones de seguridad y de integridad pueden considerarse parte del esquema conceptual. El DDL conceptual incluirá los medios para especificar dichas verificaciones.

DEFINIR PROCEDIMIENTOS DE RESPALDO Y RECUPERACION: en caso de que sufra daño cualquier porción de la base de datos – por causa de un error humano, digamos, o una falla en el equipo o en el sistema que lo apoya – resulta esencial poder reparar los datos implicados con un mínimo de retraso y afectando lo menos posible el resto del sistema. En teoría, por ejemplo la disponibilidad de los datos no dañados no debería verse afectada. El DBA debe definir y poner en práctica un plan de recuperación adecuado que incluya, por ejemplo una descarga o "vaciado" periódico de la base de datos en un medio de almacenamiento de respaldo, y procedimientos para cargar otra vez la base de datos a partir de vaciado más reciente cuando sea necesario.

SUPERVISAR EL DESEMPEÑO Y RESPONDER A CAMBIOS EN LOS REQUERIMIENTOS: es responsabilidad del DBA organizar el sistema de modo que se obtenga el desempeño que sea "mejor para la empresa", y realizar los ajustes apropiados cuando cambien los requerimientos.

INVESTIGAR LAS FUNCIONES DEL ADMINISTRADOR DE BASE DE DATOS SEGÚN KORTH)

6.3 Objetivos del DBA

El DBA es responsable primordialmente de:

- Administrar la estructura de la Base de Datos
- Administrar la actividad de los datos
- Administrar el Sistema Manejador de Base de Datos
- Establecer el Diccionario de Datos
- Asegurar la confiabilidad de la Base de Datos
- Confirmar la seguridad de la Base de Datos.

Administración de la estructura de la base de datos: La administración de la estructura de la Base de Datos incluye participar en el diseño inicial de la misma y su puesta en práctica así como controlar, y administrar sus requerimientos, ayudando a evaluar alternativas, incluyendo los DBMS a utilizar y ayudando en el diseño general de BD. En los casos de grandes aplicaciones de tipo organizacional, el DBA es un gerente que supervisa el trabajo del personal de diseño de la BD.

Implicaciones por la modificación de esquemas: Las solicitudes de modificación son inevitables una vez que el sistema ha entrado en operación, pueden aparecer solicitudes de nuevos requerimientos o estos pueden resultar de una comprensión inadecuada de los mismos. En cualquier caso, deberán efectuarse modificaciones en relación con toda la comunidad de la BD.

Una administración eficaz de la BD debe incluir procedimientos y políticas mediante las cuales los usuarios puedan registrar sus necesidades de modificaciones, y así la comunidad podrá analizar y discutir los impactos de dichas modificaciones, determinándose entonces la puesta o no en practica de tales alteraciones.

En razón del tamaño y complejidad de una BD y de sus aplicaciones, las modificaciones pudieran tener resultados inesperados. El DBA debe estar preparado para reparar la BD y reunir suficiente información para diagnosticar y corregir el problema provocado por la falla. Después de un cambio la BD es más vulnerable a fallas.

Documentación: la responsabilidad final de un DBA en la administración de la estructura de una BD es la DOCUMENTACIÓN. Es de suma importancia saber que modificaciones han sido efectuadas, como fueron realizadas y cuando fueron establecidas. Una modificación sobre la estructura de la BD pudiera ocasionar un error que no apareciera a corto plazo.

Administración de la actividad de datos: aunque el DBA protege los datos, no los procesa. El DBA no es usuario del sistema, en consecuencia, *no administra valores de datos;* el DBA *administra actividad de datos.* Dado que la BD es un recurso compartido, el DBA debe proporcionar estándares, guías de acción, procedimientos de control y la documentación necesaria para garantizar que los usuarios trabajan en forma cooperativa y complementaria al procesar datos en la BD.

Como es de suponerse, existe una gran actividad al interior de un DBMS. La concurrencia de múltiples usuarios requieren de estandarizar los procesos de operación; el DBA es responsable de tales especificaciones y de asegurarse que estas lleguen a quienes concierne. Todo el ámbito de la BD se rige por estándares, desde la forma como se capture la información (tipo, longitud, formato), como es procesada y presentada. El nivel de estandarización alcanza hasta los aspectos más internos de la BD; como sé accesa a un archivo, como se determinan los índices primarios y auxiliares, la foliación de los registros y demás.

Una administración de BD efectiva deberá disponer siempre de este tipo de estándares; entre las funciones del DBA se encuentra la de revisarlos periódicamente para determinar su operatividad, y en su caso ajustarlos, ampliarlos o cancelarlos. Es también su responsabilidad el que estos se cumplan.

Cuando se definen estándares sobre la estructura de la BD, estos deben registrarse en una sección del diccionario de datos a la que todos aquellos usuarios relacionados con ese tipo de proceso pueden acceder.

Entre las alternativas mas utilizadas por el DBA para tratar de resolver o minimizar este problema se encuentran las siguientes:

a) Restringir el acceso a los procedimientos para ciertos usuarios.
b) Restringir al acceso a los datos para ciertos usuarios procedimientos y/o datos.
c) Evitar la coincidencia de horarios para usuarios que comparten.

El DBA es el responsable de la publicación y mantenimiento de la documentación en relación con la actividad de los datos, incluyendo los estándares de la BD, los derechos de recuperación y de acceso a la BD, los estándares para la recuperación de caídas y el cumplimiento de las políticas establecidas. Los productos DBMS más populares que se encuentran en el mercado proporcionan servicios de utilerías para ayudar al DBA en la administración de los datos y su actividad. Algunos sistemas registran en forma automática los nombres de los

usuarios y de las aplicaciones a las que tienen acceso así como a otros objetos de la BD. Incorpora también utilerías que permitan definir en el diccionario de datos las restricciones para que determinadas aplicaciones o módulos de ellas solo tengan acceso a segmentos específicos de la BD.

6.4 Integridad de la base de datos

Un control de integridad o restricciones es aquel que nos permite definir con precisión el rango de valores validos para un elemento y/o las operaciones que serán consideraciones validas en la relación de tale elementos.

- El objetivo primordial de un control de integridad es la reducción de la inconsistencia en la BD.

Las restricciones de integridad normalmente se aplican en tres niveles:

- **UN ATRIBUTO SIMPLE:** Se define un dominio del atributo que es totalmente independiente del resto del entorno de la Base de Datos.
- **UN ATRIBUTO DEPENDIENTE DE OTRO:** Se definen subconjuntos de dominios posibles para un atributo X según el valor que previamente a sido asignado al atributo W.
- **RELACIONES ENTRE TUPLAS DE UNA O VARIAS TABLAS:** Se especifican valores posibles para registros completos según los valores acumulados registros previos o por valores existentes en registros de otras tablas.

La implementación de la cardinalidad resultante en el modelo será una de las restricciones importantes que el sistema debe considerar.

La programación de todas estas restricciones regularmente corre a cuenta de un programador especializado (que pudiera ser el DBA), mediante la adición de módulos al sistema; lo anterior dado que los DBMS comúnmente no incorporan facilidades para su implementación.

6.5 Seguridad de la base de datos

Existen múltiples riesgos para la seguridad de la información durante la operación, implantación y tiempos muertos en el sistema.

Riesgos en la implantación: Cuando se esta instalando o actualizando un sistema, los principales factores de riesgo son aquellos relacionados con el ajuste de formatos, dominios y otros parámetros que pueden verse afectados por la conversión del sistema; ya sea manual-automatizado o automatizado-automatizado.

Cuando el sistema que se implanta ha de recibir nueva información, es importante el establecimiento de códigos que permitan validar la captura para minimizar los riesgos de información no confiable.

Riesgos en la operación: Mientras el sistema se encuentra en uso, se dice que las operaciones se realizan en línea; es decir, la información se afecta por medio de los procedimientos definidos en el sistema.

La protección más común para reducir estos riesgos consiste en el establecimientos de claves de operación (Password) tanto para accesar a la aplicación como a las diversas operaciones que esta desempeña.

Las claves pueden asignarse:

- Genérico
- Por niveles de seguridad
- Por tipos de acceso a los datos.

La selección de las claves de acceso debe llevarse a cabo utilizando los siguientes criterios:

- No información que pueda asociarse al usuario.
- Fácil de recordar, difícil de adivinar.
- Debe utilizar un parámetro variable o algoritmo

Algunos sistemas que manejan claves fijas pueden incluir controles sobre el usuario que lo obliguen a modificar su clave de acceso con cierta regularidad.

Es importante que el código que mantiene la tabla de claves de usuarios en el sistema se encuentre encriptado.

Riesgos en tiempos muertos: Cuando el sistema no se encuentra en operación la información esta expuesta a ser alterada *fuera de línea;* es decir, sin utilizar los programas de aplicación diseñados para este fin.

Algunas de las técnicas más utilizadas para evitar y en algunos casos sólo para ejecutar modificaciones fuera de línea son:

- **Encriptamiento:** Consiste en convertir la información de la BD a un formato que resulte ilegible sino se dispone del algoritmo de conversión.
- **Aplicación de totales de control:** Consiste en generar registros ficticios que son agregados a la BD y que permitirán detectar la inserción, eliminación o modificación de datos en la gran mayoría de los casos.
- **Dígitos de control:** son caracteres que se anexan a las claves o a los datos que serán manejados con el objeto de autentificar su validez.

Su aplicación se extiende a procesos en línea y protección fuera de línea.

6.6 Recuperación de la base de datos

6.6. 1 Respaldo

En la práctica el empleo de volver a leer la entrada de una transacción o la restauración mediante imágenes posteriores dependen de la disponibilidad de una copia de respaldo de una versión anterior de la base de datos. Si las versiones se utilizan como copias de respaldos pueden construirse volviendo a posicionare la señal actual de nivel máximo de otra manera, los respaldos se crean por copiado. Es posible generar periódicamente copias de respaldo y conservar una serie de versiones anteriores. Cada copia de respaldo estará identificada por tiempo y fecha y por la última transacción incluida. Una copia de respaldo debe generarse mientras la base de datos esta en reposo, ya que las actualizaciones durante el copiado pueden provocar que la copia se inconsistente. En la base de datos muy grandes puede no presentarse nunca un periodo de descanso lo suficientemente largo para realizar una copia de respaldo.

6.6.2 Recuperación

A continuación se presentaran una serie de pasos que podrán seguirse cuando se presente una falla en el sistema y se desee recuperar la información:

1. **Detección del error,** el proceso de recuperación se inicia al detectar la existencia de un error. Se considerarán fallas de sistemas detectadas por falta de acción del sistema o por verificaciones irrecuperables de redundancia y salida incorrecta observada por un usuario.
2. **Determinación de la fuente del error**, para decidir cual es la mejor acción correctora es necesario determinar la extensión del daño. Desde luego este esfuerzo es muy relacionado con la determinación del tiempo y la causa del error. Después de una caída o cuando el procesamiento sea interrumpido debido a una señal de error, es necesario determinar tantos aquellas áreas del archivo de datos que sean sospechosas como cuál fue la transacción que no se concluyo.
3. **Ubicación de errores secundarios,** cuando se ha detectado un error que provocó una modificación inadecuada a un archivo un rastreo a través de las listas de actividad encontrara aquellas transacciones que emplearon el bloque correcto. Entonces es posible volver a introducir automáticamente el bloque correcto de las transacciones afectadas y producir resultados correctos.
4. **Aplicación de correcciones,** si la extensión del daño es limitada, puede utilizarse un proceso de volver a enrollar. Las porciones dañadas del archivo se restauran aplicando primero aquellas imágenes anteriores a los bloques en error reemplazando después de las transacciones incompletas. La salida proveniente de estas transacciones se suprime de ser posible, para evitar duplicar resultados que previamente se hayan enviado a los usuarios.

UNIDAD VII BASES DE DATOS DESCENTRALIZADAS, CENTRALIZADAS Y DISTRIBUIDAS.

Objetivo: Comprender la razón de ser de las Bases de datos distribuidas, comparar sus características con respecto a las bases de datos centralizadas.

Contenidos Teóricos:
7.1 Nuevas áreas de aplicación de las bases de Datos
7.2 Sistemas Distribuidos
7.3 Sistemas Basados en la lógica
7.4 Sistemas Paralelo

INVESTIGAR LOS SIGUIETES TEMAS

Atributos: Describen propiedades que posee cada miembro de un conjunto de entidades.

Atributos Derivados: El valor para este tipo de atributo se puede derivar de los valores de otros atributos o entidades relacionados.

Base de Datos: Serie de datos organizados y relacionados entre si, los cuales son recolectados y explotados por el sistema de información de una empresa o negocio en particular.

Clave: Permite identificar un conjunto de atributos suficiente para distinguir las entidades entre sí. Las claves también ayudan a identificar unívocamente a las relaciones y así a distinguir las relaciones entre sí.

Clave Ajena o Foránea: Duplicación en la tabla relacionada de la clave principal. Si el campo definido como clave principal o primaria en una tabla forma parte también de otra tabla se llamará clave ajena. También llamada Foreign Key.

Clave Principal: Campo de la tabla que identifica inequívocamente un registro, no pudiendo existir dos registros con la misma clave principal. Esta clave sirve para identificar un registro en concreto. También llamada Primary Key.

DDL: Permiten crear y definir nuevas bases de datos, campos e índices.

DML: Permiten generar consultas para ordenar, filtrar y extraer datos de la base de datos.

Dominio: Es el conjunto de valores permitidos para cada atributo.

Entidad: Es una «cosa», «persona» u «objeto» en el mundo real que es distinguible de todos los demás objetos. ANSI (1977).

Ejemplar de Base de Datos (ocurrencia del esquema): Colección de información almacenada en la base de datos en un momento particular.

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

Esquema: Es el diseño completo de la base de datos. Los esquemas son raramente modificados.

Formas Normales: 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.

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.

Herencia de Atributos: Un conjunto entidad de nivel más bajo hereda todos los atributos y participaciones en asociaciones del conjunto entidad de nivel superior al que está enlazado.

Independencia Física de los Datos: Es la habilidad para modificar el esquema físico sin cambiar el esquema lógico.

Interbloqueo (DEAD LOCK): Se presenta cuando dos procesos que compitan por recursos comunes inician su ejecución tomando alguno de ellos y bloqueándolo en este momento.

Modelo Entidad-Relación: Está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre estos objetos. A demás este se utiliza habitualmente en el proceso de diseño de bases de datos.

Modelo de Datos: Se utilizará para significar una descripción conceptual del espacio del problema mediante una colección de herramientas conceptuales para describir los datos, las relaciones, la semántica y las restricciones de consistencia.

Nivel Físico: Es el nivel mas bajo de abstracción, determina como están almacenados físicamente los datos (pistas, sectores, cilindros), volúmenes, archivos, tipos de datos, punteros, etc., representa el nivel más bajo.

Nivel Lógico o Conceptual: Es aquel en el que se definen las estructuras lógicas de almacenamiento y las relaciones que se darán entre ellas. Determina la organización de los archivos. Índices, llaves, orden de campos, tipos de datos.

Nivel de Vistas: Es aquel en el que se presenta al usuario final y que puede combinaciones o relaciones entre los datos que conforman a la base de datos global.

Relación: Es la asociación que existe entre dos a más entidades.

Sistema: Un sistema es un conjunto de componentes que interaccionan entre sí para lograr un objetivo común.

Sistema de Información: puede definirse técnicamente como un conjunto de componentes interrelacionados que permiten capturar, procesar, almacenar y distribuir la información para apoyar la toma de decisiones y el control en una institución.

Sistema de Bases de datos: es una colección de archivos interrelacionados y un conjunto de programas que permitan a los usuarios acceder y modificar estos archivos.

Sistemas de Recuperación de Información: se ocupan del tratamiento de datos no estructurados (documentos). Estos sistemas proporcionan facilidades de thesaurus, búsqueda en texto libre, etc.

Sistema de Gestión de Bases de Datos (SGBD): Es una colección de programas de aplicación que proporcionan al usuario de la base de datos los medios necesarios para realizar las siguientes tareas: Definición de los datos a los distintos niveles de abstracción (físico, lógico y externo).

Valor Perdido: valor que existe pero no se tiene esa información.

Básica

- Begg, C. C. (2005). Sistemas de Bases de Datos un Enfoque Práctico para Diseño, Implementación y Gestión. Madrid: 4 ed. Pearson, Addison Wesley.
- Cuadra, D., Castro, E., Iglesias, A. M., Martínez, P., Calle, F. J., De pablo, C., y otros. (2008). Desarrollo de Bases de Datos: Casos Prácticos Desde el Análisis a la Implementación. Madrid: Alfaomega, Ra-Ma .
- Mannino, V. (2007). Administración de base de datos diseño y desarrollo de aplicaciones. México: (3ª. ed), Mc Graw Hill.
- Rob, P. &. (2004). Sistemas de Bases de Datos .(5o ed.) Thomson.
- Shamkant, B. (2007). Fundamentos de Bases de Datos .(5o ed.). Pearson.

Complementaria

- Date, C. (2001). Introducción a los Sistemas de Bases de Datos. México: (7o ed.). Prentice Hall.
- De Miguel, A.; Piattini, M.(1993). Concepción y diseño de bases de datos: Del Modelo E/R al modelo relacional. Madrid, RAMA.
- Kroenke, David. (2003). Procesamiento de bases de datos Fundamentos diseño e implemantacion. Pearson Educacion.
- Gillenson, M. L. (2006). Administración de Bases de Datos. México, D.F: Limusa.
- Silberschatz., K. &. (2002). Fundamentos de Bases de Datos . Madrid: (4o ed) Mc Graw Hil.
- Luque, I., Gómez-Nieto, M., López, E., & Cerruela, G. (2002). Bases de Datos dede Chen hasta Cood con Oracle. Alfaomega , Ra-Ma: México .