

# DISEÑO BASES de DATOS

**Estructura de contenidos**

	Pág.
Introducción .....	3
Mapa de contenido .....	4
Desarrollo de contenidos .....	5
Generalidades .....	5
1. Modelos de datos basados en archivos .....	5
1.1. Sistema de archivos en red .....	6
1.2. Sistemas de archivos jerárquicos .....	6
2. Conceptos generales de bases de datos. ....	7
2.1. Sistemas de gestión de bases de datos (sgbd) .....	7
2.2. Transacciones .....	8
2.3. Arquitecturas de aplicaciones .....	9
3. Modelo entidad relación. ....	10
3.1. Elementos .....	10
3.1.1. Entidades .....	10
3.1.2. Atributos .....	11
3.1.3. Relaciones .....	12
3.2. Llaves .....	13
3.3. Construcción del diagrama entidad relación. ....	14
3.3.1. Representación de entidades y atributos. ....	14
3.3.2. Tipos de relaciones .....	15
3.3.3. Cardinalidad y participación .....	15
3.4. Ejemplo de un diagrama entidad relación. ....	16
3.5. Diagrama entidad relación extendido .....	17
3.6. Diagramas entidad relación con uml .....	19
4. Modelo relacional .....	20
4.1. Normalización .....	22
4.2. Reglas de integridad (constraints) .....	22
Glosario .....	23
Bibliografía .....	24
Control del documento .....	25

## Gerencia de proyectos software

### Introducción



Diseñar una base de datos es una tarea fundamental para la creación de un sistema de información ya que suministra la estructura sobre la cual se darán respuesta a los requerimientos funcionales que se levantaron en el inicio del proyecto.

En este recurso digital se darán al aprendiz los conceptos y técnicas para que logre llevar a cabo esta tarea de una manera estructurada y secuencial.

El diseño de bases de datos es una habilidad que se va mejorando en la medida que el analista desarrolla más proyectos y tiene la realimentación necesaria.

No siempre se llega a un diseño óptimo desde el primer intento. Una vez las bases de datos están en producción surgen temas de rendimiento, seguridad, entre otros, que pueden requerir algunos ajustes al diseño ya realizado.

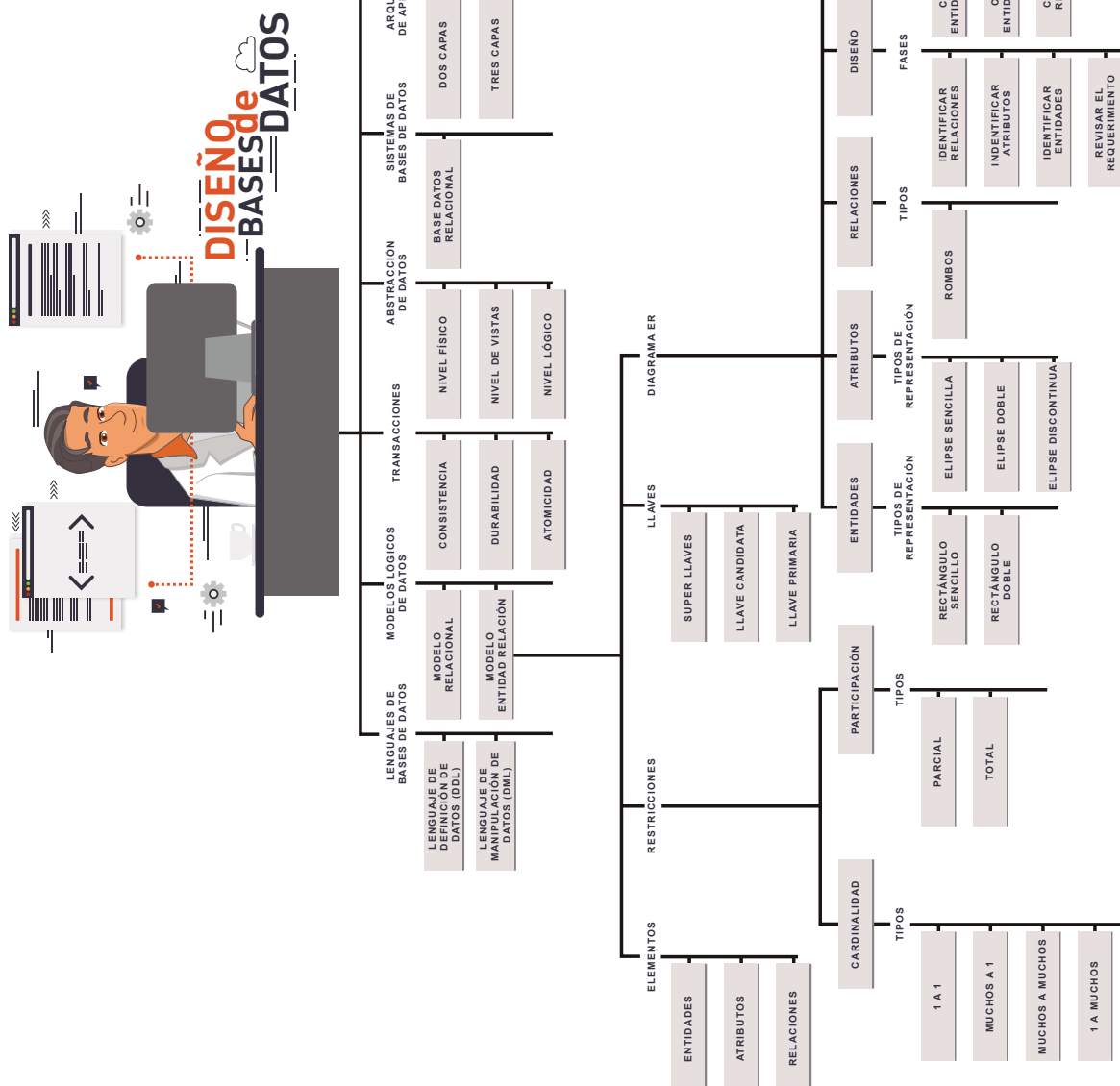
La administración de bases de datos es una tarea que ha crecido en complejidad y es común que las empresas tengan un especialista en este tema llamado DBA (Database Administrator) o Administrador de la base de datos, quién es el encargado de mantener los ambientes de producción operando sin interrupciones. También le dirá al analista o al diseñador de base de datos los inconvenientes que puedan surgir en ambientes productivos.

```

graph TD
    A[Tipos de sistemas de archivos] --> B[CONVENIENTES]
    A --> C[INCONVENIENTES]
    B --> D[Modelo Jerárquico]
    B --> E[Modelo Red]
    C --> F[Redundancia o duplicación de datos con los datos]
    C --> G[Dificultad de acceso a los datos]
    C --> H[Aislamiento de datos]
    C --> I[Problemas de integridad]
    C --> J[Problemas de atomicidad]
    C --> K[Anomalías de acceso concurrente]
    C --> L[Problemas de seguridad]
  
```

El diagrama clasifica los tipos de sistemas de archivos en dos categorías principales: convenientes e inconvenientes. Los convenientes se dividen en modelo jerárquico y modelo red. Los inconvenientes incluyen redundancia o duplicación de datos, dificultad de acceso, aislamiento de datos, problemas de integridad, problemas de atomicidad, anomalías de acceso concurrente y problemas de seguridad.

Tipos	Tipos de sistemas de archivos
CONVENIENTES	Modelo Jerárquico
	Modelo Red
INCONVENIENTES	Redundancia o duplicación de datos con los datos
	Dificultad de acceso a los datos
	Aislamiento de datos
	Problemas de integridad
	Problemas de atomicidad
	Anomalías de acceso concurrente
	Problemas de seguridad



## Desarrollo de contenidos

### Generalidades

Una base de datos es una abstracción de un modelo del mundo real que permite su implementación en un sistema informático.

Muchos de los requerimientos que se levantan en el inicio de un proyecto de desarrollo de software tienen que ver con el almacenamiento de los datos que las empresas recolectan, actualizan o consultan a lo largo de cada uno de sus procesos estratégicos, misionales y de apoyo.

Es de esta forma como surgen bases de datos de sistemas contables, financieros, logísticos, comerciales, entre otros, que representan las entidades del mundo real como clientes, facturas, inventarios, compras, etc., en un sistema computarizado.

Los **requerimientos funcionales** van a generar unos **requerimientos de datos** que tendrán que ser modelados para posteriormente ser implementados en un motor de bases de datos.

Una vez la base de datos queda diseñada e implementada en un motor de bases de datos (SGBD) como Postgres, SQL Server, Oracle, DB2, entre otras, el analista podrá utilizar las herramientas suministradas por cada una de ellas para acceder y manipular los datos de acuerdo a los requerimientos especificados.

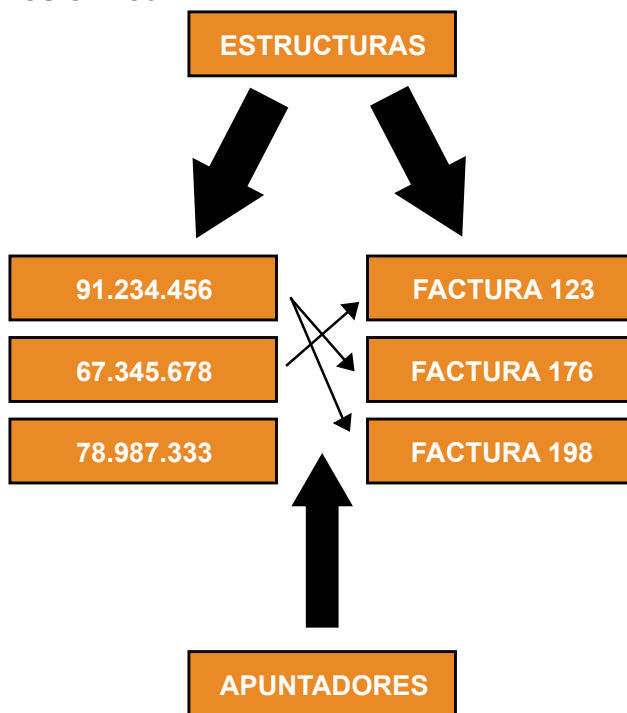
### 1. Modelos de datos basados en archivos

Los modelos basados en archivos como su nombre lo indica se apoyan en el sistema operativo para almacenar los datos en archivos digitales que posteriormente serán consultados, actualizados y compartidos por uno o varios usuarios.

Este modelo de archivos presenta los siguientes inconvenientes (SILBERSCHATZ, 2002): redundancia e inconsistencia de datos, dificultad de acceso a los datos, aislamiento de los datos, problemas de integridad, problemas de atomicidad, anomalías en el acceso concurrente, problemas de seguridad, entre otros.

Los anteriores inconvenientes eran generados porque los archivos estaban almacenados en diferentes computadores y no contaban con un software centralizado que se encargara de coordinar los distintos accesos y modificaciones a los mismos.

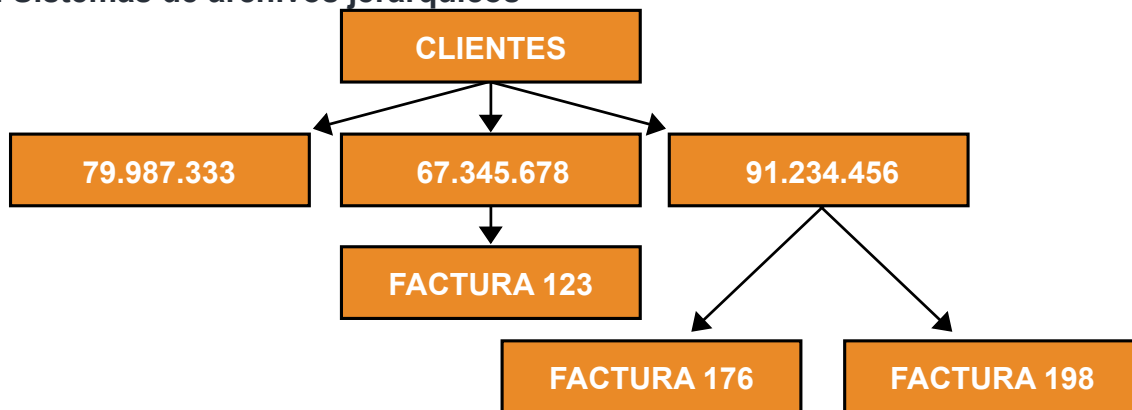
### 1.1 Sistema de archivos en red



*Figura 1.1. Sistema de archivos en red*

Fueron de los primeros sistemas de bases de datos que aparecieron y están basados en el concepto de estructuras y apuntadores (DATE, 2001).

### 1.2. Sistemas de archivos jerárquicos



*Figura 1.2. Sistema jerárquico*

En este modelo los datos son implementados como estructuras internas tipo árbol y se proporcionan operadores para manipular dichas estructuras (DATE, 2001). Dentro de los operadores están los “apuntadores de recorrido” que indican la ubicación de un dato en dicho árbol (Figura 1.2.).

## 2. Conceptos generales de bases de datos.



### 2.1 Sistemas de Gestión de Bases de Datos (SGBD)

Un SGBD (DBMS en inglés) es un sistema informático que provee los mecanismos y herramientas para almacenar, actualizar y consultar bases de datos.

Para Elmasri (ELMASRI, 2004) las principales ventajas de usar un SGBD son:

- Controlar redundancia:** en los sistemas tradicionales de archivos cada grupo de usuarios mantenía una copia independiente de datos maestros como clientes, estudiantes, cuentas, etc. En un modelo de base de datos los datos sobre cada entidad quedan unificados y pueden ser consultados de forma uniforme por todos los usuarios autorizados.
- Prevenir el acceso no autorizado:** los SGBD proveen un subsistema de autorización que controla que los datos sean consultados o manipulados solamente por los usuarios o grupos de usuarios autorizados.
- Suministrar mecanismos para agilizar las consultas:** una base de datos es constantemente utilizada para realizar reportes que pueden ser desde los más sencillos hasta los más complejos. Un SGBD suministra archivos auxiliares llamados “índices” que permiten que los tiempos para consultar los datos sean mínimos apoyados en modelos matemáticos como árboles binarios para realizar las búsquedas.
- Suministrar mecanismos de respaldo y recuperación de los datos almacenados:** un SGBD suministra herramientas que permiten realizar tanto copias de seguridad como recuperación de algunas o todas las bases de datos almacenadas. Los mecanismos para estos procedimientos pueden ir desde los más simples como lo es un respaldo

fuera de línea (sin ningún usuario consultado datos) hasta los más sofisticados que hacen copias de respaldo en línea.

- e) **Suministrar varias interfaces de usuario:** las SGBD suministran varios mecanismos para que los distintos tipos de usuarios desde los más básicos hasta los más avanzados realicen sus tareas. Es así como las SGBD proveen lenguajes de consultas en modo de comandos CLI (Command Line Interface), lenguaje de consultas con interface GUI (graphical user interface), interface para programas externos tipo API (Application Program Interface), entre otras.
- f) **Controlar las restricciones de integridad:** un SGBD permite controlar que la información que se registre para cada campo cumpla con las condiciones que se diseñaron. Por ejemplo si se definió que el campo “Nota final” debe ser entre 1 y 5, el SGBD no permitirá que se ingrese un 6 o un 0.
- g) **Implementar reglas de negocio:** las SGBD modernas suministran elementos de programación que permiten inferir nuevos datos a partir de los ya existentes. Por ejemplo determinar la edad de un cliente a partir de su fecha de nacimiento, determinar si aprobó o no una materia a partir de la nota almacenada, etc.

## 2.2 Transacciones.

Una transacción es una característica suministrada por un SGBD que permite que los cambios a una base de datos sean aplicados en su totalidad y en caso que no poderse realizar, los registros involucrados queden en su estado original. (DATE, 2001)

Un ejemplo común de transacción ocurre cuando una persona realiza un pago. El programa debe crear un registro con el pago, actualizar el saldo en el dato maestro del cliente, actualizar huellas de auditoría, entre otros. Si por algún fallo el sistema no pudo completar todas las operaciones descritas deberá dejar el sistema tal y como estaba antes de iniciar la transacción.

Para Elmasri (ELMASRI, 2004) las características deseables de una transacción son:

- a) **Atomicidad:** las operaciones son realizadas o todas o ninguna.
- b) **Preservación de consistencia:** una transacción debe dejar la base de datos consistente antes y después de ejecutarse.
- c) **Aislamiento:** en el momento de ejecutarse una transacción no debe permitir que los registros involucrados sean consultados o actualizados por otros programas.
- e) **Durabilidad o persistencia:** los cambios realizados por una transacción deben ser permanentes.

Todas las características anteriores pueden ser resumidas con el acrónimo ACID (del inglés Atomicity, Consistency preservation, Isolation, Durability) .



## 2.3 Arquitecturas de aplicaciones.

En las aplicaciones basadas en bases de datos existen varios actores o participantes, entre estos tenemos:

- a) **El cliente:** o computador que interactúa directamente con el usuario.
- b) **La aplicación cliente:** es el programa que reside en el computador del usuario.
- c) **El servidor de base de datos:** computador que ejecuta el DBMS o motor de bases de datos.
- d) **El servidor de aplicaciones:** programa o conjunto de programas que se ejecutan en un servidor e interactúan entre sí para suministrar servicios a los usuarios finales.

Las primeras aplicaciones de bases de datos se ejecutaban en el modelo cliente-servidor o arquitectura de dos capas. En este modelo la aplicación alojada en el computador del cliente accede directamente a los datos almacenados en el servidor a través de algún mecanismo de red y usando estándares como ODBC o JDBC, entre otros. (SILBERSCHATZ, 2002)

Este modelo aunque efectivo tiene inconvenientes tales como alto consumo de recursos en el equipo cliente ya que los datos “viajan” desde el servidor hasta el equipo cliente donde finalmente son procesados.

Otro esquema es la arquitectura de tres capas donde aparece el concepto de “servidor de aplicaciones” que es un conjunto de programas almacenados en un servidor que puede ser el mismo de la base de datos o diferente. El servidor de aplicaciones toma los datos del servidor de bases de datos y los procesa sin que estos tengan que ser transmitidos a la máquina cliente. Solamente los resultados son enviados (Figura 2.1.).

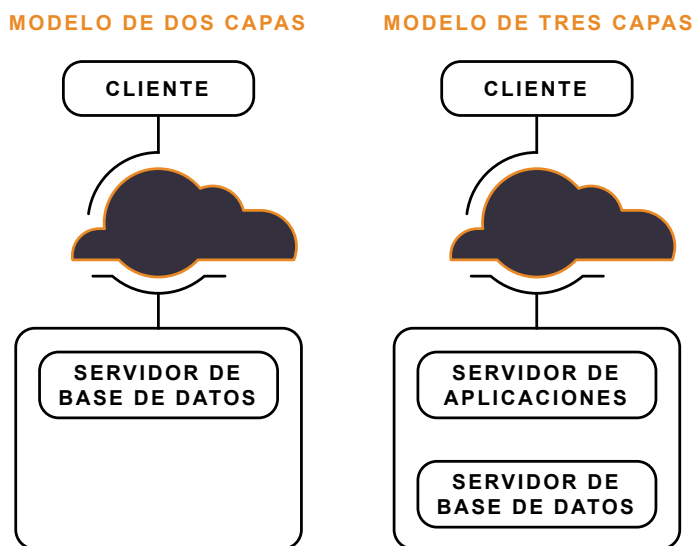
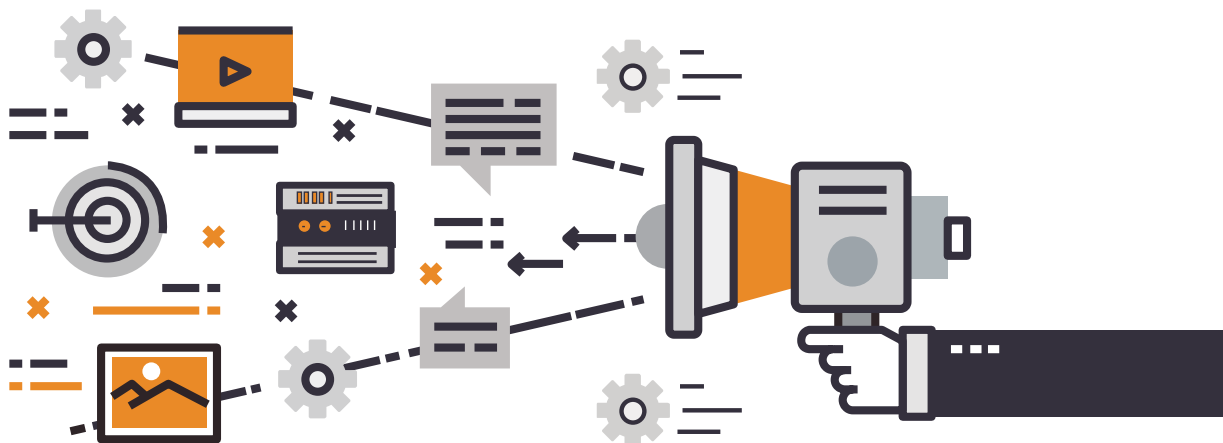


Figura 2.1. Arquitecturas de dos y tres capas.

Ejemplos de arquitectura de tres capas los tenemos con servidores de aplicaciones como Tomcat, JBOSS, SAP Netweaver, entre otros.

### 3. Modelo entidad relación.



Modelo lógico de datos creado para facilitar la representación del modelo de datos de una situación o fenómeno del mundo real mediante un esquema basado en atributos, entidades y las relaciones entre ellas. (SILBERSCHATZ, 2002)

#### 3.1. Elementos

Un modelo entidad relación permite representar un modelo de datos del mundo real a través de tres elementos: entidades, relaciones y atributos. (SILBERSCHATZ, 2002)

##### 3.1.1 Entidades

Las entidades representan un objeto o cosa que tienen una existencia en el mundo real (como una casa, un carro) o puede ser también una ente conceptual (como una empresa o un plan de cuentas.). (ELMASRI, 2004).

Las entidades se pueden clasificar en dos tipos:

- a) **Fuertes:** aquellas que tienen un atributo que las hace únicas. Como por ejemplo la cédula en una entidad “Empleado”, número de matrícula para un inmueble, entre otros. En el mundo real las entidades fuertes generalmente corresponde a los datos maestros de una empresa como clientes, empresas, activos fijos, deudores, acreedores, etc.
- b) **Débiles:** aquellas que no tienen por sí mismas un atributo que las haga únicas. Por ejemplo los pagos que hacen a un obligación financiera. En el mundo real las entidades débiles generalmente corresponden a movimientos o transacciones que se efectúan sobre las entidades fuertes como pagos de los clientes, préstamos de los acreedores, etc.

### 3.1.2 Atributos



Son los elementos que en su conjunto conforman una entidad. Los atributos reflejan una característica propia de la entidad como por ejemplo los atributos cédula, nombre, dirección y ciudad de la entidad “Empleado”

Los atributos se pueden clasificar en las siguientes categorías (ELMASRI, 2004):

- a) **Simples o atómicos:** aquellas características que son indivisibles como color, edad, estatura, etc.
- b) **Compuestos:** aquellos atributos que se pueden subdividir en partes como por ejemplo una dirección que puede estar conformada por calle, carrera, número, número de cada, número de apartamento, etc.
- c) **Mono valorados o valor único:** se presenta cuando los atributos tienen un valor único. Por ejemplo la fecha de nacimiento, cédula.
- d) **Multi valorados:** se presenta cuando un atributo puede contener más de un valor representativo, por ejemplo los números de teléfono de un empleado, o los nombres de sus hijos.
- e) **Derivados:** cuando un atributo se puede calcular a partir de otro. Por ejemplo la edad a partir de la fecha de nacimiento o el departamento a partir del municipio de residencia.

El valor que toma un atributo puede ser nulo cuando este es inexistente. Por ejemplo si un empleado está activo el valor del atributo “Fecha de retiro” será nulo.

### 3.1.3 Relaciones

Una relación es un vínculo o asociación que existe entre dos o más entidades. Por ejemplo la entidad “Empleado” y la entidad “Cargo” están relacionadas porque cada empleado “ejerce” o “desempeña” un cargo por ejemplo director, vendedor, conductor, etc. En este caso la relación puede tomar el mismo nombre del rol como “ejerce” o “se desempeña como”.

Dependiendo de cuantas entidades participen en una relación se pueden presentar los siguientes casos:

- a) **Binaria**: entre dos entidades diferentes.
- b) **Ternaria**: entre tres entidades diferentes.
- c) **n-aria**: entre n entidades diferentes.

También se da el caso cuando para un mismo conjunto de entidades existen relaciones entre sus elementos. Por ejemplo en el conjunto de entidades “Empleados” se puede presentar una relación “Es jefe de” para representar el modelo jerárquico del personal. Este caso se denomina relación recursiva. (ELMASRI, 2004).

El rol que juega una entidad dentro de una relación puede tomar un nombre que ayuda a comprender mejor esta última. Por ejemplo el rol que juega la entidad “empleado” en la relación “empleado-cargo” puede llamarse trabajador o funcionario. El nombre del rol no es obligatorio en un modelo entidad relación pero son determinantes para entender el modelo en el caso de las relaciones recursivas.



Las relaciones deben cumplir con ciertos requisitos para que el modelo conceptual de la base de datos se asemeje a la realidad que se quiere representar.

- a) **Cardinalidad o relación de cardinalidad:** representa el número de veces que la relación se puede presentar entre dos o más entidades. Por ejemplo un empleado se puede desempeñar solamente en un cargo al tiempo. En ese caso la cardinalidad para la relación “se desempeña como” entre empleados y cargos es 1. Desde el punto de vista de los cargos, un cargo puede ser desempeñado por ninguno, uno o varios empleados. Esto quiere decir que la cardinalidad es de muchos que se representa con una “M”.
- b) **Participación:** representa la cantidad mínima de relaciones que se deben presentar entre los elementos de dos entidades. También es llamada cardinalidad mínima (ELMASRI, 2004). Para el ejemplo en cuestión un empleado debe estar desempeñando de manera obligatoria un cargo.

La participación puede ser de dos tipos:

- a) **Total:** cuando es obligatorio que exista por lo menos una relación para cada elemento de un conjunto de entidades. También se denomina restricción de existencia para denotar que debe existir al menos una relación. Por ejemplo un empleado debe estar asignado a un cargo, un almacén u oficina, etc.
- b) **Parcial:** cuando es opcional que exista una relación para cada elemento de un conjunto de entidades con los elementos de otra entidad. Por ejemplo una entidad “Empleado” puede o no estar relacionado con los elementos de la entidad “Dependiente a cargo”. En otras palabras es opcional que los empleados tengan dependientes a cargo.

### 3.2. Llaves

Las llaves son atributos que permiten identificar de manera única una entidad dentro de un conjunto de entidades dado. Así la cédula puede ser la llave de un conjunto de entidades “Empleado”.

Las llaves se pueden clasificar así (SILBERSCHATZ, 2002):

- a) **Superllave:** se habla de superllave al conjunto de campos que hacen único una entidad dentro de un conjunto de entidades. En el ejemplo del empleado la cédula es una superllave porque hace único un empleado. También son super-llaves cualquier combinación como cédula-nombre, cédula-apellido, cédula-nombre-apellido, etc.
- b) **Llave:** una llave es un tipo particular de superllave que contiene el mínimo número de atributos para asegurar la unicidad de la entidad. Por ejemplo en la superllave “cédula-nombre” se podría eliminar el atributo nombre y seguir siendo un atributo único. En este caso la cédula es a la vez una llave y una superllave.

- c) **Llaves candidatas:** se definen como llaves candidatas a todas las superllaves mínimas, o aquellas que al quitarles uno atributo o más atributos dejan de serlo. Por ejemplo una tipo de entidad estudiante puede tener la cédula y el código de estudiante como llaves candidatas ya que ambas identifican de manera única.
- e) **Llave primaria:** es la llave candidata que es elegida por el diseñador de la base de datos para identificar inequívocamente una entidad dentro del conjunto.

### 3.3. Construcción del diagrama entidad relación.




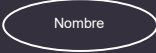
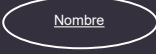


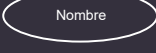
Los elementos y conceptos del modelo entidad relación se pueden representar en un diagrama que facilita su comprensión y estudio.

Antes de iniciar la diagramación se deben llevar a cabo los siguientes pasos que determinan los elementos del diagrama:

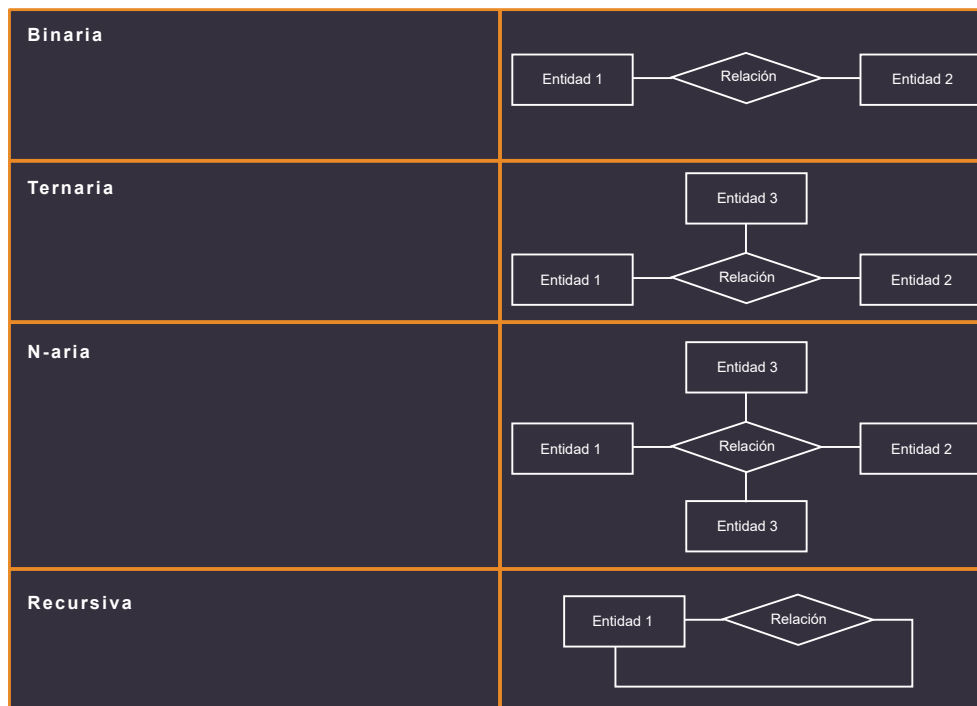
- 1) Identificar las entidades fuertes y débiles.
- 2) Identificar los atributos, llaves, atributos compuestos, derivados.
- 3) Identificar las relaciones.
- 4) Identificar las cardinalidades y participaciones de las relaciones.
- 5) Identificar las especializaciones y generalizaciones.

Luego se procede a la diagramación usando las siguientes convenciones (ELMASRI,2004):

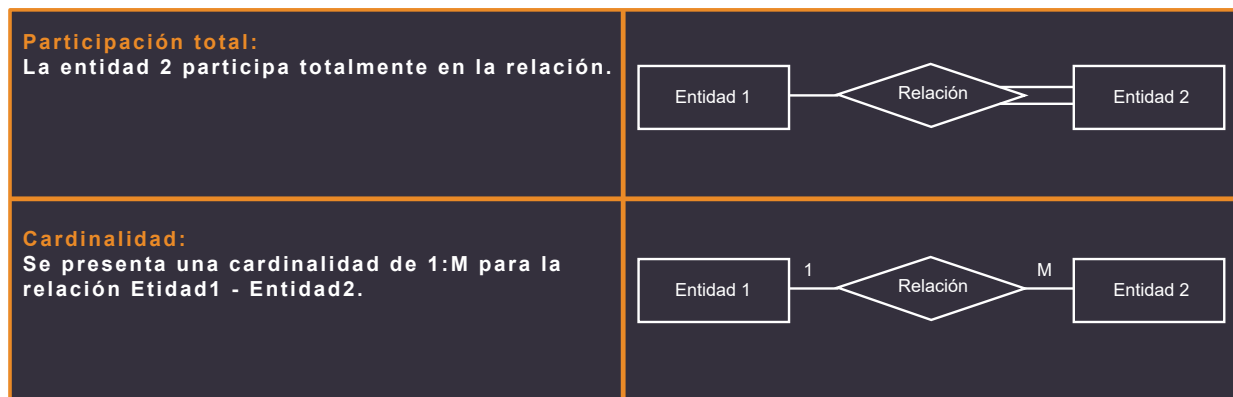
#### 3.3.1 Representación de entidades y atributos.

ELEMENTO	REPRESENTACIÓN
Entidad o entidad fuerte	
Entidad debil	
Relación	
Atributo (mono valor)	
Atributo llave	
Atributo multi valorado	
Atributo compuesto	
Atributo derivado	

### 3.3.2. Tipos de relaciones



### 3.3.3. Cardinalidad y participación:



### 3.4 . Ejemplo de un diagrama entidad relación.

El primer paso para determinar el modelo entidad relación es estudiar los requerimientos del módulo o sistema de información a desarrollar.

Para este ejemplo se expone el siguiente requerimiento (Figura 3.4):




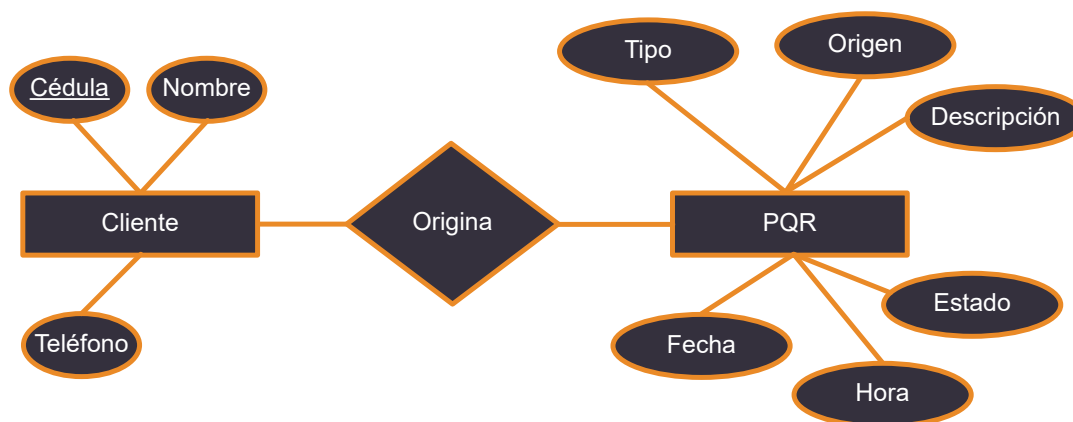
Logotipo de la empresa		EMPRESA XYZ S.A. ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE No 001	Formato XYZ-GC-001
ID	Nombre	Descripción	Prioridad
1	Módulo de servicio al cliente para una empresa de servicios públicos.	<p>Se requiere un módulo para atender las peticiones, quejas y recursos (PQR's) que interponen los clientes en nuestras oficinas, telefónicamente o por correo electrónico.</p> <p>El sistema deberá permitir la consulta, registro, actualización e impresión de cada PQR. El formulario de PQR deberá contener:</p> <ol style="list-style-type: none"> <li>1. Tipo: petición, queja, recurso, otros servicios.</li> <li>2. Fecha y hora: deberá traer la fecha y hora del sistema al momento de grabar el servicio.</li> <li>3. Cédula o identificación del cliente.</li> <li>4. Descripción de la PQR.</li> <li>5. Origen de la PQR: oficina, telefónico, correo electrónico, carta.</li> <li>6. Teléfono del cliente.</li> <li>7. Estado: solicitada, resuelta, anulada.</li> </ol> <p>El sistema deberá proveer un programa para que las PQR's no solucionados sean asignados al área que corresponda. Ejemplo los PQR's por exceso de consumo deberán ser asignados al área técnica.</p> <p>El sistema deberá proveer una relación con todos los PQR's permitiendo filtrar por fecha, estado, cédula, nombre y origen. Esta relación puede ser vista por pantalla, enviada a impresora, o enviada a una hoja electrónica.</p>	Alta
	Controles y restricciones	<p>El sistema no debe permitir que las PQR's sean borradas en ningún caso.</p> <p>Se deberán migrar las PQR's del sistema anterior al nuevo.</p> <p>Solo el supervisor puede modificar las PQR's una vez grabadas.</p> <p>El formato impreso deberá ceñirse a los estándares de la empresa.</p> <p>El formulario deberá ser con interfaz web.</p>	
	Criterios de aceptación	<p>El software cumple con los requisitos funcionales solicitados.</p> <p>EL software realiza los controles solicitados.</p> <p>El software realizó la migración de la información del sistema anterior.</p>	
	Fecha de especificación	12/05/2017	
<div>    </div> <div> <p>Pedro Pérez Gerente de servicio al cliente</p> <p>Pablo Pereira Jefe de servicio al cliente</p> <p>Firma (s) Demás usuarios involucrados en la especificación</p> </div>			

Figura 3.4. Requerimiento de ejemplo



A continuación un posible diagrama entidad relación para el requerimiento (Figura 3.5):



*Figura 3.5. Diagrama ER de ejemplo*

A partir del diagrama podemos hacer los siguientes comentarios:

1. Se identifican dos entidades: cliente y PQR (Petición, Queja, Recurso).

Las entidades Cliente y PQR se identifican directamente del requerimiento.

2. La relación entre un cliente y una PQR está dada porque un cliente “origina” una PQR.
3. Un cliente puede crear muchas PQR's. Por tanto la cardinalidad de un cliente respecto a las PQR's que puede originar es de uno a muchos o en la notación estándar “1: M”.
4. Una PQR solo puede estar relacionada con un único cliente. Además pueden existir muchas PQR's para un mismo cliente. Por tanto la cardinalidad desde la entidad PQR hacia la entidad Cliente es de muchos a uno o en la notación estándar “M: 1”.
5. Los atributos de la entidad Cliente son: cédula, nombre y teléfono. La cédula por estar subrayada es la llave primaria de la entidad.
6. Los atributos de la entidad PQR son: tipo, origen, descripción, estado, fecha, hora. Esta entidad se considera una entidad débil ya que por sí sola no tiene un atributo único o llave. Por lo anterior el diseñador de la base de datos deberá asignar una llave que puede ser un número consecutivo, un número de PQR, entre otros.

### 3.5. Diagrama entidad relación extendido

Existen ciertas características de las bases de datos que no pueden ser representadas con el modelo entidad relación estándar entre estas tenemos: la especialización, la generalización y la herencia de atributos.

### 3.5.1. Especialización y generalización.

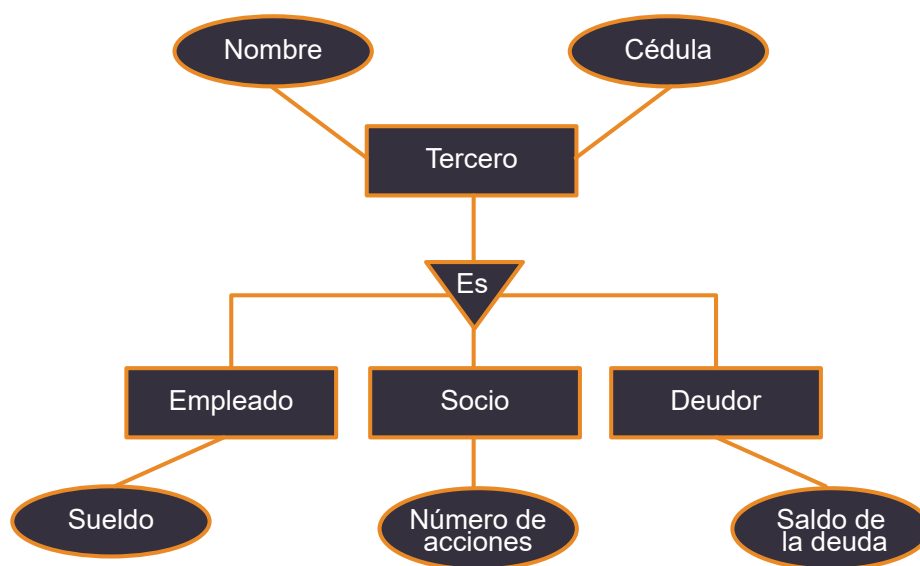
La especialización es una técnica que disminuye la redundancia de datos en un modelo entidad relación. Mediante un proceso de identificación de sub-grupos de atributos se puede optimizar el almacenamiento de los datos (SILBERSCHATZ, 2002).

Considérese el caso de una entidad “Tercero” como se conoce comúnmente a las personas desde el punto de vista contable. Al analizar las entidades se puede encontrar que existen los siguientes sub-grupos: empleados, socios, deudores, entre otros.

Las entidades “especializadas” heredan los atributos de la entidad origen. De este modo los atributos principales de un tercero como son cédula y nombre son heredados a las entidades “empleado”, “socio” y “deudor”.

Las nuevas entidades especializadas contienen atributos propios de su subclase. Para el ejemplo anterior la entidad “empleado” tendrá un atributo “sueldo”, la entidad “socio” tendrá un atributo “número de acciones” y la entidad “deudor” tendrá un atributo “saldo de la deuda”.

La especialización permite modelar estos casos mediante la siguiente simbología (Figura 3.6):



*Figura 3.6. Ejemplo de especialización de entidades*

En el proceso de especialización se toma una entidad y se buscan los subgrupos o subcategorías de atributos. El proceso de generalización es inverso, parte de analizar varias entidades para encontrar los atributos comunes. Luego se crea una entidad o super-entidad con los atributos comunes. Por ejemplo las entidades “camión”, “ambulancia” y “bus” comparten atributos en común como son “placa”, “modelo”, entre otros, que se usan para crear una super-entidad llamada “vehículo”.

Tanto la especialización como la generalización se representan en un modelo entidad relación con el mismo símbolo. La diferencia radica en la forma como fueron concebidas o conceptualizadas (Figura 3.7.).

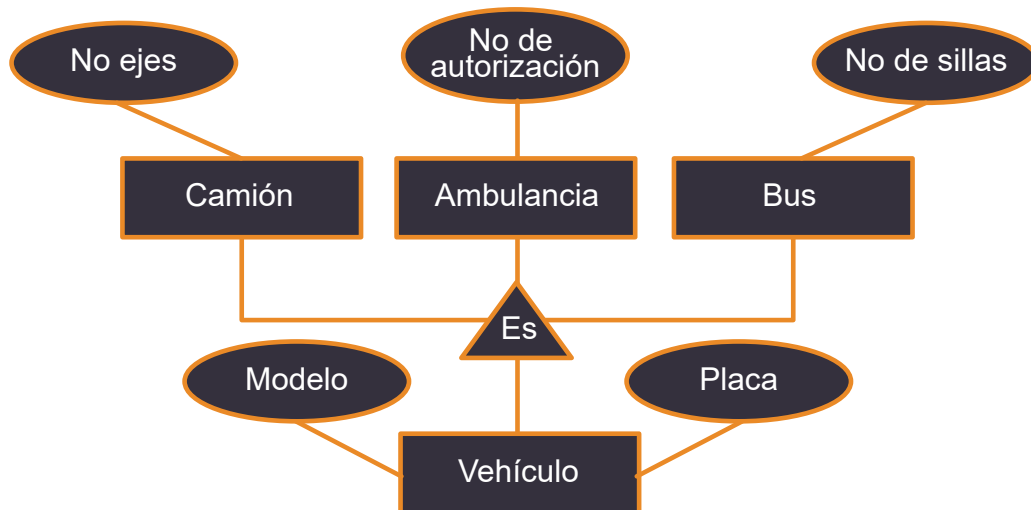


Figura 3.7. Ejemplo de generalización

### 3.6. Diagramas entidad relación con UML

Los diagramas entidad relación también se pueden representar usando el lenguaje de modelado universal o UML. (SILBERSCHATZ, 2002).

	Modelo tradicional	UML
Entidades y atributos		
Relaciones		
Cardinalidad y participación		

#### 4. Modelo relacional

El modelo relacional es un modelo lógico que utiliza un esquema basado en tablas, registros y campos para representar los datos del mundo real. Fue introducido por Ted Codd en los años setenta (ELMASRI, 2004).

Los primeros manejadores de bases de datos relacionales aparecieron en la década de los ochentas y hoy en día son populares los productos de IBM (DB2 e Informix), Oracle DBMS, Microsoft SQL Server. También son populares los manejadores de código libre como Postgresql, Mysql, entre otros.

Este modelo utiliza el concepto de tabla para representar un conjunto de entidades como por ejemplo empleados, clientes, productos, vehículos, entre otros. También utiliza el concepto de tabla para representar el conjunto de relaciones que existen entre las entidades. Por ejemplo puede existir una relación entre la entidad “empleado” y la entidad “Cargo desempeñado” la cual se representa también a través de tablas. (SILBERSCHATZ, 2002)

Las tablas representan un conjunto de entidades y un registro o fila representa una entidad en particular. Los atributos o características de cada entidad, como lo son la cédula y el nombre de un empleado, son representados por campos o columnas (Figura 4.1.).

Empleados					
Cedula	Nombres	Apellidos	Sexo	Grupo sanguíneo	Factor RH
78.234.789	Carlos Alberto	Pérez	M	O	+
98.654.876	Juan Carlos	Ruiz	M	O	-
1098.764.789	Antonio	Rodriguez	M	A	+
1095.764.889	Paula Andrea	Jimenez	F	B	-
72.876.567	María Alejandra	Arévalo	F	AB	+

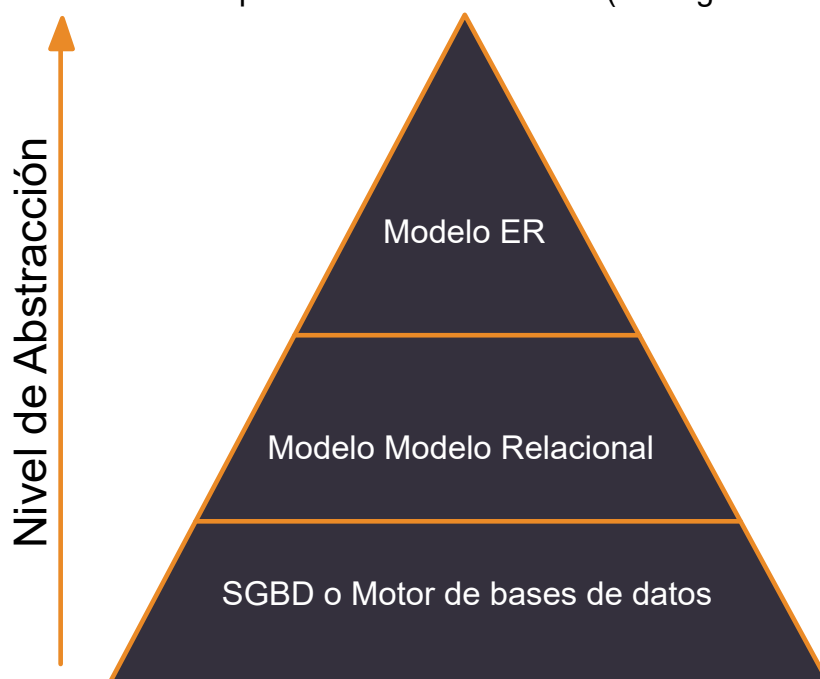
  

Pagos			
Cedula	No. de recibido	Fecha de pago	Valor pagado
72.876.567	8989	2017-05-27	1.200.00
98.654.876	7676	2017-05-27	800.000
78.234.789	7878	2017-05-27	800.00
98.654.876	6500	2017-05-27	800.00

*Figura 4.1. Ejemplo modelo relacional.*

El modelo relacional es el modelo de datos más usado actualmente (SILBERCHATZ, 2002) y es el paso anterior en la implementación de un esquema de base de datos directamente en un SGBD.

El nivel de abstracción del modelo relacional es menor, es decir, contiene más detalles que el modelo entidad relación que se verá más adelante (ver Figura 4.2.).



*Figura 4.2. Nivel de abstracción de los modelos*

Características del modelo relacional:

- a) Los renglones, líneas o filas, poseen datos acerca de una entidad.
- b) Las columnas contienen datos acerca de los atributos de la entidad.
- c) Cada posición en la tabla almacena un valor simple y atómico de un atributo.
- d) Todos los valores almacenados en una columna son del mismo tipo, es decir, están definidos sobre el mismo dominio.
- e) Cada columna o atributo debe poseer un nombre único.
- f) El orden de las columnas no es relevante.
- g) El orden de las filas no es relevante.
- h) No pueden existir dos filas idénticas en la tabla.

A partir de un modelo entidad relación se puede llegar a un modelo relacional usando diferentes técnicas una de ellas es usar el algoritmo de mapeo ER a Relacional propuesto por Elmasri (ELMASRI, 2004). Como su nombre lo indica este algoritmo utiliza las similitudes entre los dos modelos para convertir entidades y relaciones en tablas y atributos en campos.

## 4.1 Normalización

La normalización es un proceso mediante el cual se reduce la redundancia de datos a través del análisis de las relaciones entre las tablas hasta llegar a un nivel óptimo de agrupamiento de campos. (ELMASRI, 2004).

Cuando un dato está almacenado en diferentes tablas se dice que hay redundancia de datos. Por ejemplo. Las tablas “empleados” y “pagos” pueden contener los campos “cédula” y “nombre empleado”. Pero usando el modelo relacional se podría eliminar el campo “nombre empleado” de la tabla “pagos” ya que el campo puede ser consultado de la tabla “empleados” a través de relación que hay entre las mismas.

El proceso consiste en aplicar reglas de normalización sobre las tablas de una base de datos, cada regla se denomina “Forma Normal”. Si una tabla cumple la primera regla, se dice que está en la “primera forma normal” y si cumple la regla N está en forma normal N. Aunque son posibles otros niveles de normalización, la tercera forma normal se considera el máximo nivel necesario para la mayor parte de las aplicaciones.

## 4.2. Reglas de integridad (Constraints)

Estas reglas nos permiten asegurar la consistencia o integridad del modelo de datos a través de controles que son ejercidos cada vez que la base de datos es actualizada o consultada.

Las reglas de integridad se pueden agrupar en los siguientes tipos (ELMASRI, 2004):

1. Las reglas de integridad de modelo. Son las restricciones generales en un modelo de datos. Estás son genéricas y en el caso del modelo relacional son:

- a) **Unicidad de la clave primaria:** toda llave primaria no debe admitir valores repetidos.
- b) **Integridad de entidad de la llave primaria:** los atributos de la llave primaria no pueden tener valores nulos.
- c) **Integridad referencial:** definición de las políticas de inserción y eliminación para las filas que cuyos campos estén siendo referenciados.
- d) **Integridad de dominio:** los valores almacenados en un atributo deben ser del dominio declarado para dicho atributo.

2. Las restricciones de integridad semánticas:

Son también conocidas como reglas de negocio, es decir, condiciones muy particulares que deben cumplir los datos que son impuestas por las políticas de la empresa. Por ejemplo se puede restringir que un ítem de una factura se grave sólo si el campo “número de fórmula médica” está diligenciado.

## Glosario

**ACID:** del inglés Atomicity, Consistency preservation, Isolation, Durability. Características de las transacciones realizadas por una SGBD.

**Ambiente productivo:** se dice del conjunto de recursos informáticos que están soportando las operaciones reales que llevan a cabo las empresas.

**Ambiente de desarrollo:** se dice del conjunto de recursos informáticos que soportan el desarrollo de software operando con datos de prueba.

**DBA:** acrónimo de Database Administrator. Persona encargada de administrar las bases de datos de una empresa o entidad.

**JDBC:** acrónimo de Java Database Connectivity. Estándar de conexión a bases de datos de aplicaciones desarrollado en Java.

**ODBC:** acrónimo de Open DataBase Connectivity. Estándar para conexión que los programas se conecten a motores de bases de datos.

**PQR:** acrónimo de Petición, Queja o Recurso.

**SGBD:** acrónimo de Sistema Gestor de Bases de Datos.

## Bibliografía

Date, C. (2001). *Introducción a los sistemas de bases de datos*. México: Pearson Educación.

Elmasri, R., Navathe, S. (2004). *Fundamentals of Database Systems*. Boston: Addison Wesley.

Silberschatz, A., Korth, H., Sudarshan, S. (2002). *Fundamentos de bases de datos*. Madrid: McGrawHill.



## Control del documento

CONSTRUCCIÓN  
OBJETO DE  
APRENDIZAJE



### DISEÑO DE BASES DE DATOS - ADSI

Centro Industrial de Mantenimiento Integral - CIMI  
Regional Santander

**Líder línea de producción:** Santiago Lozada Garcés

**Asesores pedagógicos:** Rosa Elvia Quintero Guasca  
Claudia Milena Hernández Naranjo

**Experto temático:** Julio César Hernández-V1  
**Experto temático:** Nelson Mauricio Silva Maldonado -V2

**Diseño multimedia:** Jesús Antonio Vecino Valero

**Programador:** Francisco José Lizcano Reyes

**Producción de audio:** Víctor Hugo Tabares Carreño



Este material puede ser distribuido, copiado y exhibido por terceros sin se muestra en los créditos. No se puede obtener ningún beneficio comercial y las obras derivadas tienen que estar bajo los mismos términos de la licencia que el trabajo original.