GENERALITAT VALENCIANA
Conselleria d'Educació, Investigació, Cultura i Esport

GOBIERNO DE ESPAÑA
MINISTERIO DE EDUCACIÓN, CULTURA Y DEPORTE

IES ANTONIO SEQUEROS ALMORADÍ

Unión Europea
Fondo Social Europeo
El FSE invierte en tu futuro

U.T. 4: Diseño de Bases de Datos Relacionales.

I.E.S. ANTONIO SEQUEROS

ADMINISTR. DE SISTEMAS INFORMÁTICOS EN RED

GESTIÓN DE BASES DE DATOS

CURSO 2022/2023

Índice

Bibliografía

DE MIGUEL y PIATTINI (1993). *Concepción y Diseño de Bases de Datos*. Ed. Ra-Ma.
KORTH Y SILBERSCHATZ(1998). *Fundamentos de Bases de Datos*. Ed. Mc Graw-Hill.
LUQUE Y GÓMEZ(1997). *Diseño y Uso de Bases de Datos Relacionales*. Ed. Ra-Ma.

1.- METODOLOGÍAS DE DISEÑO

1.1.- INTRODUCCIÓN

Un sistema de información (SI) es un conjunto de elementos que funcionan conjuntamente con el objetivo de recoger, tratar, manipular y aportar la informaciones necesarias para el desarrollo de las actividades de una empresa u organización. Un SI puede incluir procesos manuales o automáticos.

Uno de los elementos principales de un SI es la base de datos (BD). Las BD son ejemplos típicos de grandes sistemas de software con tres características importantes:

> Hay una gran cantidad de datos que deben ser almacenados en memoria externa y que deben ser organizados de forma que los datos elementales puedan ser recuperados y actualizados fácil y eficientemente.
> Los datos guardan entre sí complejas interrelaciones. La información incluye restricciones estáticas y dinámicas, como los valores permitidos o las posibles evoluciones.
> Los datos deben ser compartidos entre diferentes usuarios y el sistema debe mantener la integridad de la información.

En los siguientes apartados se establecen los conceptos y técnicas necesarios para diseñar bases de datos. Los criterios que definen un buen diseño de la BD son:

> Independencia física
> Independencia lógica
> Manipulación de datos por no informáticos
> Eficacia del acceso a los datos
> Administración centralizada de los datos
> No redundancia de los datos
> Coherencia de los datos
> Compatibilidad de los datos
> Seguridad de los datos

1.2.- METODOLOGÍAS DE DISEÑO DE BASES DE DATOS

Existen distintas metodologías para el diseño de SI y de BD, que identifican las etapas del diseño, los subproductos que se obtienen en cada una de ellas, así como las herramientas necesarias para desarrollar esta actividad. Nosotros seguiremos 8 etapas/fases:

1.- ANÁLISIS DE REQUISITOS DE USUARIO

El objetivo de esta etapa es describir con precisión la información que va a contener la base de datos y determinar las demandas de datos que sufrirá el sistema.

Las especificaciones del sistema pueden describirse informalmente utilizando descripciones narrativas, o formalmente utilizando un modelo de datos. El modelo de datos a utilizar debe ser suficientemente 'expresivo' como para poder representar toda la información de la BD.

Al integrar en la BD las necesidades de información de distintos grupos de usuarios y de distintas aplicaciones dentro de la empresa u organización, será necesario identificar previamente todos los grupos de usuarios que interactúan con la BD. Posteriormente se obtendrá para cada uno de estos grupos la descripción detallada de sus requisitos. Cada grupo constituye una vista de la BD.

En esta etapa se identificarán los objetos o eventos del mundo real que almacenará la BD, sus propiedades y las relaciones que mantienen entre ellos. Se identificarán los requisitos de integridad de la información que darán lugar a restricciones de manipulación. Se identificarán las autorizaciones de acceso a la información por parte de los distintos grupos de usuarios. También puede ser interesante en esta etapa tener una estimación del volumen de datos a almacenar.

Por último, se identificarán las operaciones a realizar sobre la información. Esta información es relevante en el modelo relacional únicamente para el diseño del modelo físico de la BD. Por esta razón, en algunos casos no se requiere. Se describirán la naturaleza de las transacciones, la frecuencia con que se realizan, la información que utilizan y producen, así como el flujo de información entre ellas.

2.- DISEÑO DEL ESQUEMA CONCEPTUAL

En esta etapa se combinan los requisitos de los distintos grupos de usuarios del sistema para contribuir a una descripción única y coherente de toda la información a almacenar en la BD. El esquema conceptual se desarrolla en un modelo semántico y define las entidades en la BD, las relaciones entre ellas, las restricciones de integridad de la información, los grupos de usuarios y las autorizaciones de acceso de cada uno de ellos.

Este proceso se conoce a veces como integración de vistas, y en el se produce una descripción global de la BD eliminando la redundancia e inconsistencia entre las especificaciones de distintos grupos de usuarios. El diseñador debe reconocer entre las diferentes vistas los sinónimos y homónimos y aquellos tipos de datos que pertenecen a las mismas categorías.

En esta etapa se construye un modelo de datos que describe cada uno de los objetos y sus relaciones. Se identifican los atributos que forman las claves de acceso a los objetos, se decide la representación de las interrelaciones y se identifican las propiedades opcionales y fijas, para lo cual se utiliza un modelo semántico de datos, normalmente el modelo Entidad/Relación.

3.- DISEÑO LÓGICO

Durante esta etapa, el modelo conceptual se transforma en el modelo empleado por el SGBD donde se implementará el sistema. El modelo que más se utiliza hoy por hoy es el modelo relacional.

Para BD relacionales, en esta etapa se construyen las tablas a partir de los elementos del esquema conceptual, incluyendo las relaciones entre entidades y las reglas de integridad. En esta etapa se crean los usuarios y se administran las autorizaciones para acceder a la información.

El modelo E/R desarrollado en el paso anterior puede transformarse directamente en tablas, siguiendo una serie de reglas de transformación. La mayoría de modelos ofrece estas reglas para una transformación semiautomática del esquema conceptual de relaciones de la BD.

4.- NORMALIZACIÓN DE LOS ESQUEMAS RELACIONALES

En muchos casos, las tablas resultantes del proceso anterior se someten a un proceso posterior de normalización que elimina redundancias de la información no eliminadas y que repercuten en dificultades en el procesamiento de la información.

5.- DISEÑO FÍSICO

En función de las operaciones a realizar sobre la BD, se definen los mecanismos de almacenamiento y organización de la información, incluyendo la creación de índices o la agrupación en 'clusters' de los datos. En algunos casos puede ser necesario desnormalizar la información (añadiendo redundancia) para conseguir el rendimiento deseado de la aplicación.

6.- IMPLEMENTACIÓN

Es la transformación del modelo de datos y los diseños realizados en una base de datos en funcionamiento, que opere en un determinado equipo y bajo el control de un SGBD.

7.- TEST

La etapa de test tiene el propósito de descubrir errores aparecidos en las etapas anteriores y que provocan un comportamiento del sistema diferente al previsto. Es también objetivo de la etapa de test el comprobar junto a los usuarios que el sistema satisface los objetivos establecidos durante la etapa de especificación.

8.- MANTENIMIENTO

La fase de mantenimiento incluye la corrección de errores que pudieran aparecer durante la etapa de funcionamiento del sistema. También incluye las modificaciones que el usuario pudiera solicitar a partir del trabajo diario con el sistema o de nuevas necesidades y la mejora de los interfaces de usuario.

2.- DISEÑO CONCEPTUAL. EL MODELO ENTIDAD-RELACIÓN (E/R)

En este apartado estudiaremos el modelo E/R, uno de los modelos de datos conceptuales más extendidos en las metodologías de diseño de bases de datos y en las herramientas CASE.

El modelo E/R fue propuesto por Peter P. Chen en 1976. Según Chen, 'el modelo E/R puede ser usado como una base para una vista unificada de los datos', adoptando 'el enfoque más natural del mundo real que consiste en entidades y relaciones'.

Posteriormente, otros autores han investigado y escrito sobre el modelo, proponiendo importantes aportaciones, por lo que realmente no se puede considerar que exista un único modelo E/R, sino más bien lo que podríamos llamar una familia de modelos.

El modelo E/R está centrado en dos conceptos fundamentales: el de *entidad* y el de *relación* que explicaremos más adelante .

2.1.- CONCEPTOS FUNDAMENTALES

En el modelo E/R se pueden distinguir como elementos fundamentales las entidades, los atributos, y las relaciones, además del conjunto de valores (análogo al concepto de dominio).

ENTIDAD

Se puede definir entidad como aquel objeto (real o abstracto) acerca del cual queremos almacenar información en la base de datos. Denominaremos a la estructura genérica en su sentido abstracto *tipo de entidad*, mientras que *entidad* será cada una de las ocurrencias o instancias de este tipo de entidad.

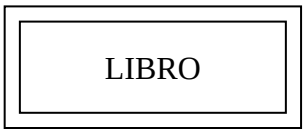
La representación gráfica de un tipo de entidad es un rectángulo etiquetado con el nombre del tipo de entidad:

Tres reglas generales que debe cumplir cualquier entidad son :

Tiene que tener existencia propia
Cada ocurrencia de un tipo debe poder distinguirse de las demás
Todas las ocurrencias de un tipo de entidad deben tener los mismos tipos de características (atributos).

Existen dos clases de entidades: *regulares*, que tienen existencia por ellas mismas, y *débiles*, cuya existencia depende de otro tipo de entidad (p. ej., FAMILIAR depende de que exista EMPLEADO, y la eliminación de EMPLEADO obliga a la eliminación de FAMILIAR).

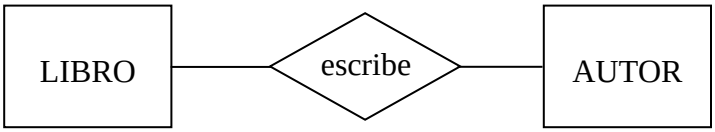
Los tipos de entidad débil se representan con dos rectángulos concéntricos con su nombre en el interior:



RELACIÓN

Se entiende por relación a aquella asociación o correspondencia existente entre entidades. Llamaremos *tipo de relación* a la estructura genérica del conjunto de relaciones existentes entre dos o más tipos de entidad.

El tipo de relación se representa mediante un rombo etiquetado con el nombre de la relación, unido mediante arcos a los tipos de entidad que asocia. P. ej.:

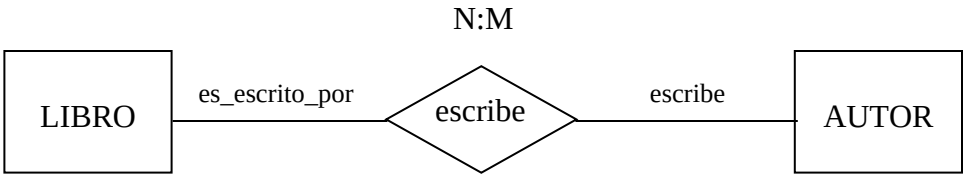


Entre dos tipos de entidades puede existir más de un tipo de relación.

Una relación se define por su *nombre* y por su *grado*. El nombre es el identificador que se le da a la propia relación, y el grado equivale al número de tipos de entidad a los que asocia o relaciona. Así por ejemplo, en el caso anterior, 'escribe' es una relación de grado 2. También pueden existir relaciones de grado 1 (o reflexivas), de grado 3, 4, …

En general, las relaciones de grado superior a 2 se suelen reducir a relaciones de grado 2, si la semántica de la relación lo permite, puesto que se facilita tanto la comprensión del diagrama como la posterior conversión a otros modelos.

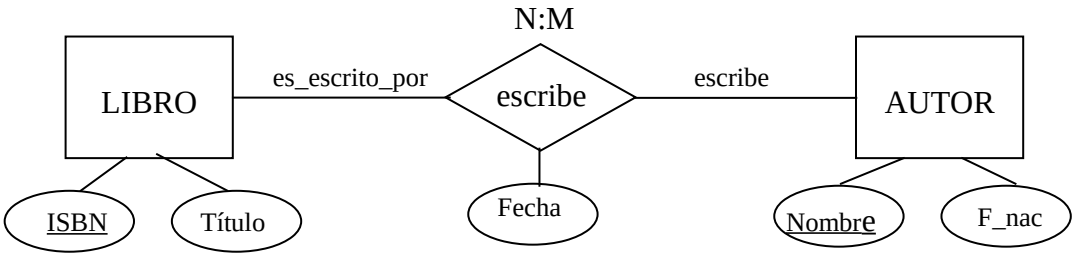
Otro elemento que caracteriza a las relaciones es el *tipo de correspondencia*, que es el número máximo de ocurrencias de cada tipo de entidad que pueden intervenir en una ocurrencia del tipo de relación que se está tratando. Gráficamente, esto se representa con alguna de estas etiquetas textuales: 1:1, 1:N, N:M. El *papel* o *rol* es la función que cada tipo de entidad realiza en el tipo de relación. Se representa con su nombre sobre el arco que une el tipo de entidad con el tipo de relación. P. ej.:



ATRIBUTO

Cada una de las propiedades o características que tiene un tipo de entidad o un tipo de relación se denomina ATRIBUTO. La representación gráfica de un atributo consiste en una elipse con el nombre del atributo en su interior.

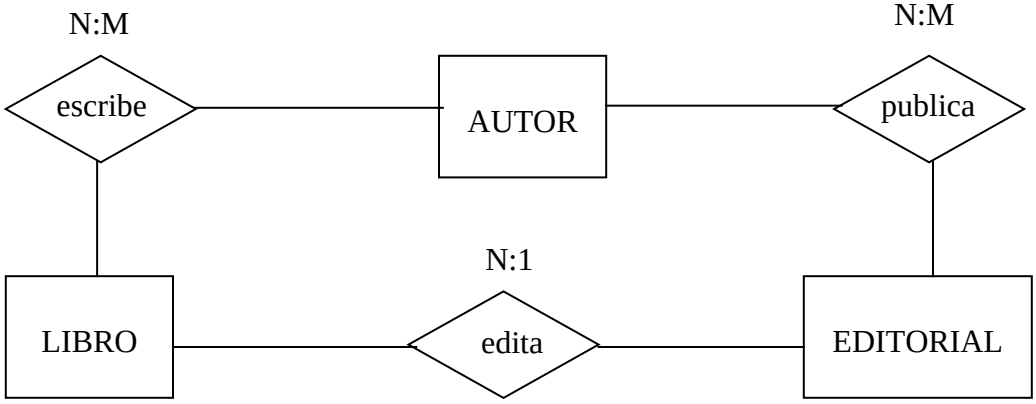
Entre todos los atributos de un tipo de entidad debemos elegir uno o varios que actúen como claves primarias. Estos atributos se representarán de la misma forma, pero con el nombre del atributo subrayado. Veamos un ejemplo:



2.2.- CONTROL DE REDUNDANCIA DE LOS DIAGRAMAS E/R

Una vez construido un esquema E/R, hay que analizar si se presentan redundancias, ya que pueden acarrear problemas a la hora de instrumentar la base de datos. Además de la existencia de atributos redundantes, como los que se derivan de otros mediante algún cálculo, hay que estudiar detenidamente los ciclos en el diagrama E/R, ya que pueden indicar la existencia de relaciones redundantes.

Un ciclo en un diagrama E/R es un grupo de tipos de entidad unidos en forma circular o cíclica a través de ciertas relaciones. P. ej.:



Es evidente que en este ejemplo podríamos eliminar la relación publica, puesto que si conocemos los libros que un autor ha escrito, y sabemos las editoriales que editan dichos libros, somos capaces de extrapolar las editoriales en las que un determinado autor publica sus libros. Por consiguiente, podemos eliminar la relación publica de nuestro diagrama por ser redundante.

Este caso no es general, es decir, es posible que existan ciclos en los que no aparezcan redundancias. Los casos más típicos son los que resultan de dividir relaciones ternarias en sus correspondientes binarias.

2.3.- MODELO E/R EXTENDIDO

Una vez visto el modelo básico de Chen, hay que profundizar un poco más en otros aspectos que aportan aún más significado y relevancia a esta herramienta, tan fundamental en la fase de diseño conceptual.

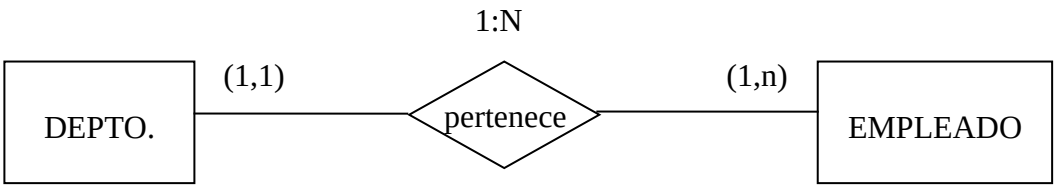
Ya hemos señalado que varios autores han considerado diversas extensiones al modelo E/R definido por Chen, dando lugar a lo que algunos han llamado modelo E/R extendido (EE/R). Trataremos de mostrar las extensiones al modelo más interesantes por su funcionalidad y uso.

CARDINALIDAD DE UN TIPO DE ENTIDAD

Se define como el número máximo y mínimo de ocurrencias de un tipo de entidad que pueden estar relacionadas con una ocurrencia del otro u otros tipos de entidad que participan en la relación. Su representación gráfica es una etiqueta del tipo (0,1), (1,1), (0,n), o (1,n), según corresponda. El significado de cada una es:

(0,1). Para una ocurrencia determinada de los otros tipos de entidades que participan en la relación, el valor mínimo de ocurrencias de la entidad que se trata es 0, y el número máximo es 1.
(1,1). Para una ocurrencia determinada de los otros tipos de entidades que participan en la relación, debe existir una y sólo una ocurrencia de la entidad que se trata.
(0,n). Para una ocurrencia determinada de los otros tipos de entidades que participan en la relación, el valor mínimo de ocurrencias de la entidad que se trata es 0, y el número máximo es n.
(1,n). Para una ocurrencia determinada de los otros tipos de entidades que participan en la relación, debe existir como mínimo una ocurrencia de la entidad que se trata, si bien no hay límite en el número de veces que dicha ocurrencia puede aparecer en la relación.

Ejemplo:



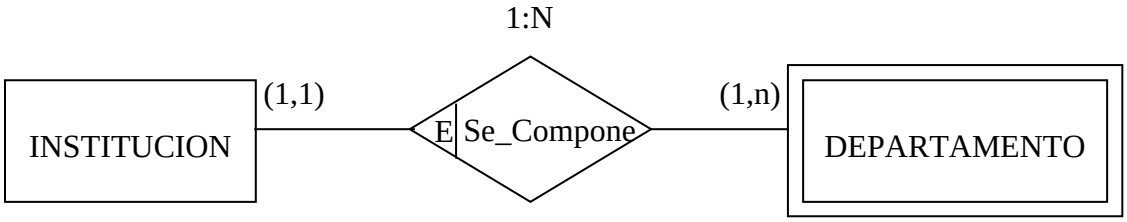
DEPENDENCIA EN EXISTENCIA Y EN IDENTIFICACIÓN

Los tipos de relación se clasifican, según el tipo de entidades que vinculan, en *regulares* si asocian tipos de entidad regulares, y *débiles* si asocian un tipo de entidad débil con un tipo de entidad regular. Un tipo de interrelación débil exige siempre que las cardinalidades del tipo de entidad regular sean (1,1).

Dentro de los tipos de relación débil podemos distinguir entre dependencia en existencia y dependencia en identificación:

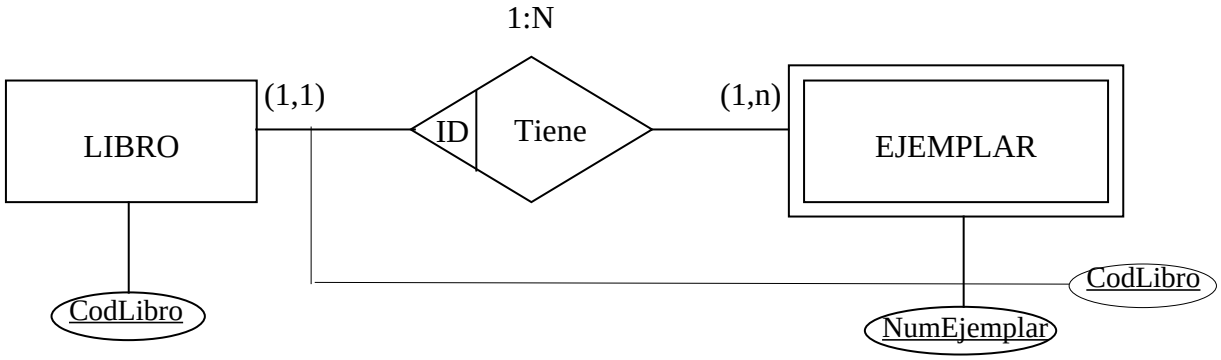
Dependencia en existencia:

Se dice que hay una dependencia en existencia cuando en una relación está vinculado un tipo de entidad regular con uno débil, de forma que las ocurrencias del tipo de entidad dependiente (tipo de entidad débil) no pueden existir sin la ocurrencia de la entidad regular de la que dependen. Si desaparece una ocurrencia de un tipo de entidad regular, todas las ocurrencias de la entidad débil que dependen en existencia de la misma desaparecen con ella. Así, por ejemplo, en la figura siguiente se observa que el tipo de relación Se_Compone que asocia el tipo de entidad regular INSTITUCIÓN con el tipo de entidad débil DEPARTAMENTO es una dependencia en existencia, ya que los datos acerca de los departamentos de una institución sólo tendrán sentido si ésta permanece en la base de datos. Se indica añadiendo la etiqueta "E" al rombo que representa la interrelación débil con dependencia en existencia.



Dependencia en Identificación:

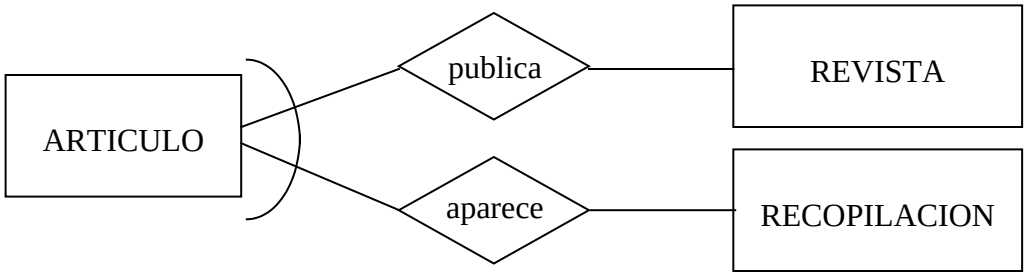
Hay una dependencia en identificación cuando, además de la dependencia en existencia, las ocurrencias del tipo de entidad débil no se pueden identificar sólo mediante sus propios atributos, sino que se tiene que añadir la clave de la ocurrencia de la entidad regular de la cual dependen. Así, por ejemplo, en la figura siguiente se observa que el tipo de relación Tiene, que asocia el tipo de entidad regular LIBRO con el tipo de entidad débil EJEMPLAR, es dependiente en identificación, ya que un ejemplar determinado, además de depender en existencia de un cierto libro, está identificado con la clave del libro (*CodLibro)* del cual depende el ejemplar, más un código propio (*NumEjemplar*).Se indica gráficamente añadiendo la etiqueta ID al rombo que representa la relación.



La clave principal de Ejemplar es *CodLibro + NumEjemplar*

RELACIONES EXCLUSIVAS

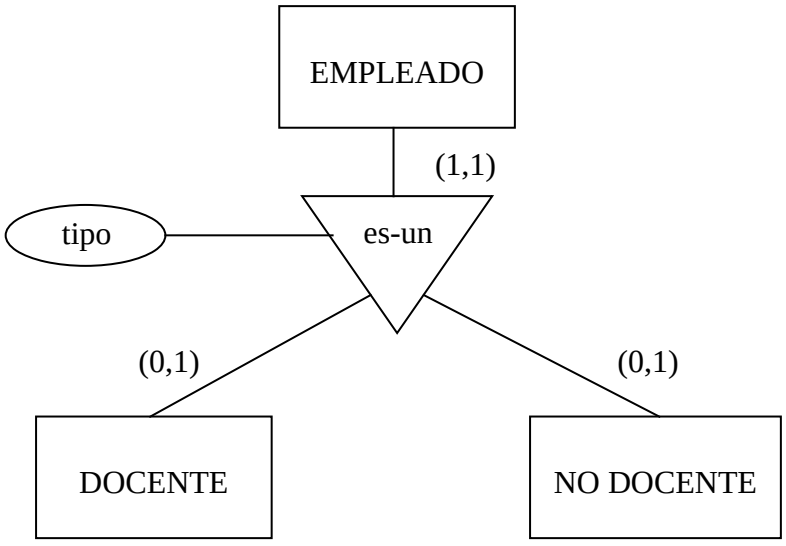
Decimos que dos o más tipos de relación son exclusivos cuando cada ocurrencia de un tipo de entidad sólo puede pertenecer a un tipo de relación. Un ejemplo, con su correspondiente forma de representación, es el siguiente:



GENERALIZACIÓN Y HERENCIA

La descomposición de tipos de entidad en varios subtipos es una necesidad muy habitual en el modelado de bases de datos. En el mundo real se pueden identificar varias jerarquías de entidades. La relación que se establece entre un supertipo y sus subtipos corresponde a la noción de 'es_un' o 'es_un_tipo_de'.

Este tipo de relación especial se representa a través de un triángulo invertido con la base paralela al rectángulo que representa al supertipo, y conectado a los subtipos. Esta relación tiene la característica de que toda ocurrencia de un subtipo es una ocurrencia del supertipo, aunque no sucede lo contrario, con lo que las cardinalidades serán siempre (1,1) en el supertipo, y (0,1) o (1,1) en el subtipo. El atributo del supertipo que actúa como discriminante se liga al triángulo a través de una elipse. Por ejemplo:

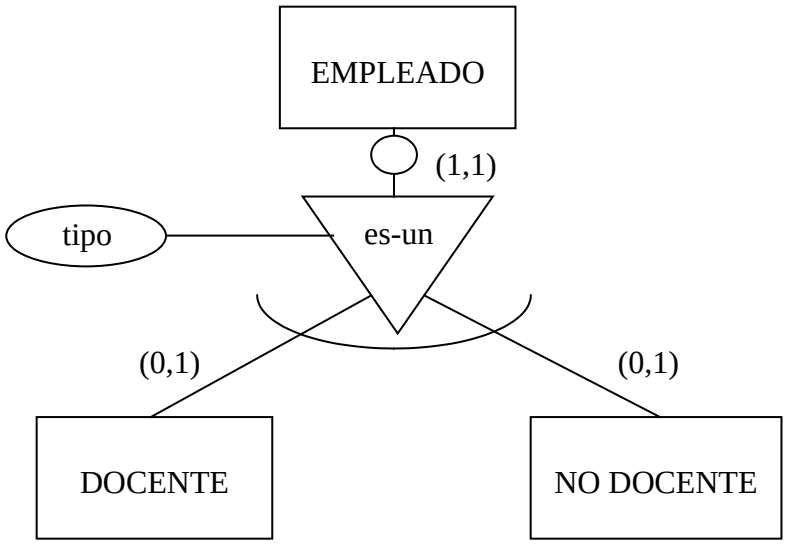


Una característica importante en estas relaciones es la herencia, puesto que cualquier atributo del supertipo pasa a ser un atributo de los subtipos.

Adicionalmente, existen dos tipos de elementos para la representación que se utilizan para indicar:

Arco. Indica la exclusividad de los subtipos, es decir, que una entidad del supertipo sólo puede ser de uno sólo de los subtipos.
Elipse vacía. Indica la obligatoriedad del supertipo de pertenecer a alguno de los subtipos.

Un ejemplo completo de estos diagramas es:

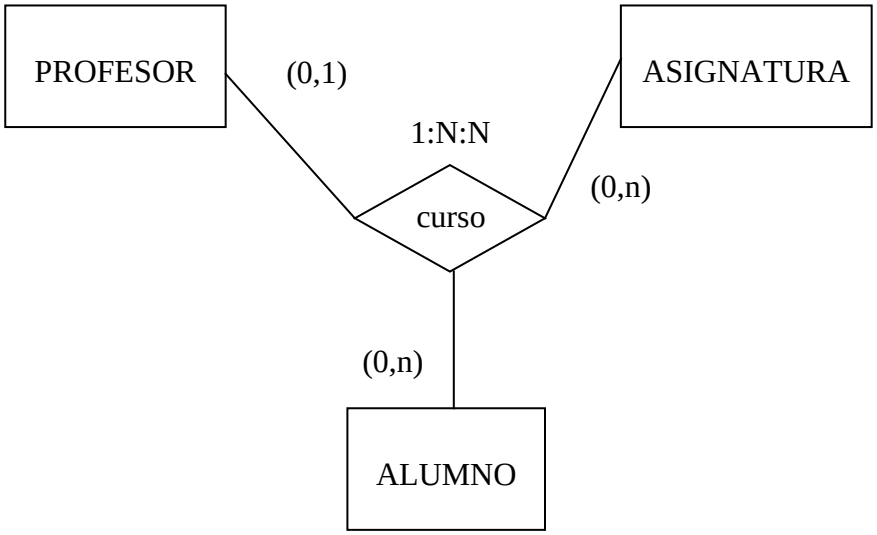


En este caso, se indica que un empleado debe ser de tipo docente o no docente, y además no puede pertenecer a los dos subtipos a la vez.

RELACIONES TERNARIAS (DE GRADO 3)

Como ya se ha mencionado con anterioridad, lo habitual es encontrar relaciones binarias en los diagramas E/R. Sin embargo, esto no significa que no puedan existir relaciones de orden superior, aunque siempre es conveniente tratar de reducirlas (en la medida en que la semántica de la relación lo permita) a relaciones binarias. Vamos a analizar cómo se representan las relaciones ternarias, y cómo se deben interpretar las cardinalidades de estos tipos de relaciones.

Supongamos la relación ternaria mostrada en la figura que aparece en la página siguiente. En esta relación de ejemplo aparecen las entidades Profesor, Asignatura y Alumno. La relación se llama curso, y representa las ternas que se pueden dar durante un curso académico, relacionando los profesores que imparten ciertas asignaturas, con los alumnos matriculados en dichas asignaturas, y por ende, los profesores que tienen asignados ciertos alumnos.



Es evidente que esta relación se podría descomponer en tres relaciones binarias, de modo que los profesores estuviesen relacionados con los alumnos a los que les imparten clase, los alumnos estuviesen relacionados con las asignaturas en las que están matriculados, y las asignaturas estuviesen relacionadas con los profesores que las imparten.

La información que se podría extraer de ambos diagramas E/R sería la misma, sólo que quizás se encontraría de forma más compacta en la representación tabular correspondiente al diagrama con la relación ternaria. No obstante, es mucho más fácil de interpretar el diagrama E/R con relaciones binarias.

Volviendo al diagrama con la relación ternaria, hay que indicar que todos los elementos que pueden aparecer en torno a la relación tienen el mismo significado que en las relaciones binarias, excepto la cardinalidad de las relaciones.

Como podrá observarse fácilmente, la cardinalidad de una relación binaria hace referencia al número de apariciones que puedo tener (máximo y mínimo) de una entidad respecto a la aparición de un elemento de la otra entidad que participa en la relación. Si se pretende extender este significado al caso de relaciones de orden n, habrá que especificar la cardinalidad máxima y mínima de elementos de una entidad respecto a los n-1 elementos de las n-1 entidades restantes. Es decir, hay que fijar la aparición de elementos de n-1 entidades para encontrar la cardinalidad máxima y mínima de los elementos de la entidad restante.

Por tanto, en nuestro ejemplo, la cardinalidad (0,n) correspondiente a la entidad alumno indica que para una pareja fija de profesor/asignatura puedo tener en la relación un mínimo de 0 alumnos y un máximo sin determinar.

3.- DISEÑO LÓGICO. NORMALIZACIÓN.

En el diseño lógico se deben coordinar exigencias casi siempre encontradas, como son eliminar redundancias, conseguir la máxima simplicidad, y evitar cargas suplementarias de programación, obteniendo una estructura lógica adecuada que venga a establecer el debido equilibrio entre las exigencias de los usuarios y la eficiencia.

Pero básicamente, el proceso de diseño lógico debe utilizar como entrada un modelo de datos conceptual, el cual se deberá convertir al modelo de la base de datos para el que se está diseñando, esto es, el diseño lógico consiste en una transformación de modelos. Adicionalmente, será necesario comprobar que el modelo lógico resultante garantiza las propiedades de eliminación de redundancias, fácil accesibilidad a los datos, etc. Para ello, en el caso de bases de datos relacionales, recurriremos a la teoría de la normalización, que nos permitirá obtener las mejores relaciones posibles para un esquema lógico.

3.1.- TRANSFORMACIÓN DE DIAGRAMAS E/R EN RELACIONES

El paso de un esquema en el modelo E/R al relacional está basado en los tres principios siguientes:

Todo tipo de entidad se convierte en una relación.
Todo tipo de relación N:M se transforma en una relación.
Todo tipo de relación 1:N se traduce en el fenómeno de propagación de clave o se crea una nueva relación.

En el proceso de transformación de esquemas se pierde semántica, pero esto no va a afectar para nada la integridad de la base de datos, ya que se definirán, por ejemplo, restricciones de integridad referencial para asegurar la conservación de la misma.

TRANSFORMACIÓN DE ENTIDADES

Cada tipo de entidad se debe convertir a una relación, es decir, será necesario crear una tabla para cada entidad que aparezca en el diagrama E/R. Esto, sin embargo, tiene alguna restricción para el caso particular en que las relaciones entre entidades sea del tipo 1:1.

TRANSFORMACIÓN DE ATRIBUTOS DE ENTIDADES

Cada atributo de una entidad se debe transformar en una columna en la relación a la que ha dado lugar la entidad:

El o los atributos principales de una entidad (es decir, los que actúan como identificador único para los elementos de la relación, subrayados en el esquema) pasan a ser la clave primaria de la relación.
El resto de atributos pasan a ser columnas de la tabla, pudiendo tomar valores nulos, a no ser que se indique lo contrario.

TRANSFORMACIÓN DE RELACIONES

Dependiendo del tipo de relación (cardinalidad) de que se trate, existen diversas formas de transformarlas:

Relaciones N:M. Se transforman en una relación que tendrá como clave primaria la concatenación de los atributos principales de cada una de las entidades que relacionan. Estos atributos deben ser clave ajena respecto a cada una de las tablas donde ese atributo es clave primaria. Habrá que considerar lo que ocurre en los casos en los que se borre o modifique la clave primaria referenciada. Será necesario crear las restricciones de cardinalidad adecuadas a través de la sentencia CHECK de SQL.

Relaciones 1:N. Existen dos soluciones para transformarlas. Habrá que considerar en ambos casos las referencias ajenas, y las actuaciones en caso de borrado y modificación.

a) Propagar el atributo principal de la entidad que tiene cardinalidad máxima 1 a la que tiene N, y hacer desaparecer la relación como tal.

b) Transformarla en una relación como si fuese una de tipo N:M. Esta acción sólo es recomendable en los casos en que: 1) cuando es posible que aparezcan muchos nulos porque existen pocos elementos relacionados, 2) cuando se prevé que dicha relación pasará en un futuro a ser de tipo N:M, y 3) cuando la relación tiene atributos propios.

Relaciones 1:1. Puesto que se trata de una particularización de cualquiera de los dos casos anteriores, se puede del mismo modo aplicar cualquiera de las dos reglas definidas. No obstante, es recomendable seguir las siguientes reglas: 1) si la relación es (0,1)(0,1), es mejor crear una relación para evitar el tener muchos nulos como propagación de alguna de las claves a la otra relación (si se prevé que puede haber muchos nulos); 2) si la relación es (0,1) (1,1), es mejor propagar la clave de la entidad (1,1) a la (0,1); 3) si la relación es (1,1)(1,1) la propagación es indiferente, y se hará atendiendo a los criterios de frecuencia de acceso (consulta, modificación, inserción, etc.) a cada una de las tablas en cuestión.

TRANSFORMACIÓN DE ATRIBUTOS DE RELACIONES

Si la relación se transforma en una tabla, todos sus atributos pasan a ser columnas de la tabla. En el caso en que alguno de los atributos de la relación sea principal, deberá ser incluido como parte de la clave primaria en dicha tabla.

TRANSFORMACIÓN DE RELACIONES EXCLUSIVAS

Para soportar relaciones exclusivas debemos definir las restricciones en cada caso. Por ejemplo, en el caso en que exista una exclusividad en la edición de un libro por parte de una editorial o de una universidad, estas dos relaciones se resuelven mediante la propagación de las claves, llevando las claves primarias de editorial y universidad a libro. Se utilizará entonces la cláusula CHECK de SQL para introducir las restricciones.

Ejemplo:

```
CREATE TABLE LIBRO
( Cod_libro char(10),
Titulo, char(100),
…
Cod_editorial char(20),
Cod_univers char(20),
PRIMARY KEY(Cod_libro),
FOREIGN KEY (Cod_editorial) REFERENCES Editorial ON UPDATE CASCADE,
FOREIGN KEY (Cod_univers) REFERENCES Univers ON UPDATE CASCADE,
CHECK ( (Cod_editorial IS NULL AND Cod_univers IS NOT NULL) OR
(Cod_univers IS NULL AND Cod_editorial IS NOT NULL) ) );
```

TRANSFORMACIÓN DE TIPOS Y SUBTIPOS

Los tipos y subtipos no son objetos que se puedan representar en el modelo relacional estándar. Existen, pues, varias posibilidades para su transformación:

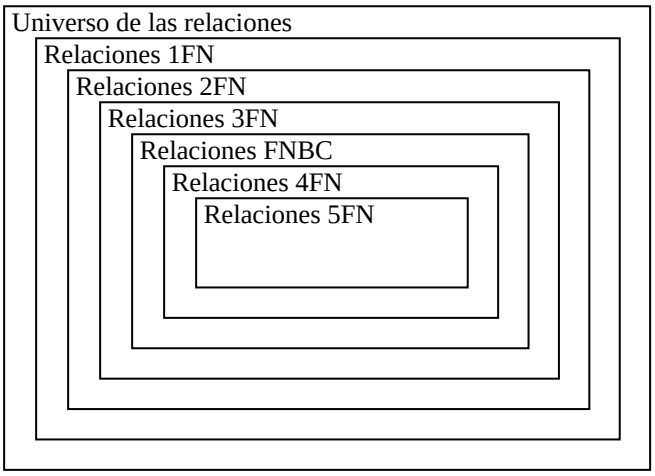
1. Englobar todos los atributos de una entidad y sus subtipos en una sola relación (tabla), añadiendo el atributo que permite distinguir los subtipos. También habrá que especificar las restricciones semánticas adecuadas.
2. Crear una relación (tabla) para el supertipo, y tantas relaciones (tablas) como subtipos existan (entidades débiles por identificación). Esta es la mejor opción desde el punto de vista semántico, pero es menos eficiente que la opción anterior.
3. Crear sólo relaciones (tablas) para los subtipos, añadiendo en cada una de ellas los atributos pertenecientes al supertipo.

3.2.- NORMALIZACIÓN. FORMAS NORMALES

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

Se ha definido un gran número de formas normales (FN). En principio, Codd definió la 1FN, 2FN y 3FN, con la idea de que era más deseable que una relación estuviese en 2FN que en 1FN, y a su vez, era mejor que estuviese en 3FN que en 2FN. Se introdujo también la idea de un procedimiento, el *procedimiento de normalización*, con el cual una cierta relación en una determinada FN puede convertirse en un conjunto de relaciones más deseables (o sea, en una FN superior). Además, este procedimiento es reversible, lo que garantiza que no se pierde información en cada paso del proceso.

Con posterioridad, se definieron la forma normal de BOYCE/CODD (FNBC) y las 4FN y 5FN. El gráfico que aparece en la página siguiente muestra como las relaciones se agrupan en función de su estado de normalización:



DEPENDENCIAS FUNCIONALES

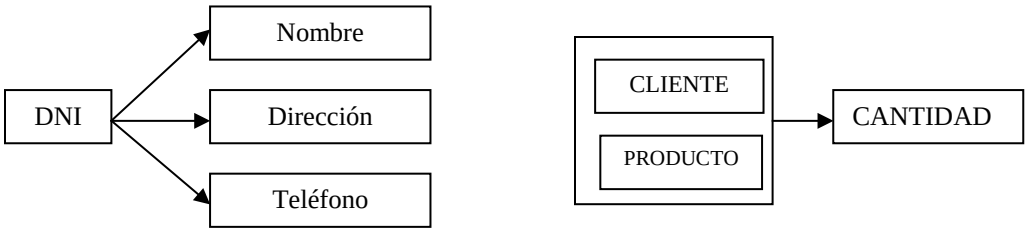
El concepto de dependencia funcional se define del siguiente modo:

Dada una relación R, el atributo Y de R depende funcionalmente del atributo X de R si y sólo si siempre que dos tuplas de R concuerden en su valor de X, deben por fuerza concordar en su valor de Y.

Definimos también el concepto de dependencia funcional completa:

Se dice que el atributo Y de la relación R es por completo dependiente funcionalmente del atributo X de R si depende funcionalmente de X y no depende funcionalmente de ningún subconjunto propio de X.

Para representar las dependencias funcionales utilizaremos los *diagramas de dependencias funcionales*. Un ejemplo de este tipo de diagramas es:



La dependencia funcional es un concepto semántico. Como consecuencia, se puede decir que estas dependencias son un tipo especial de reglas de integridad. Estas dependencias representan a su vez interrelaciones de muchos a uno.

PRIMERA, SEGUNDA Y TERCERA FORMAS NORMALES

Por simplicidad, se definen estas tres formas normales para el caso en el que las relaciones sólo tienen una clave candidata (y por tanto, solo tienen clave primaria). Para el caso en que no se cumpla esto, se recurre directamente a la FNBC, que se explica con posterioridad.

Definiciones:

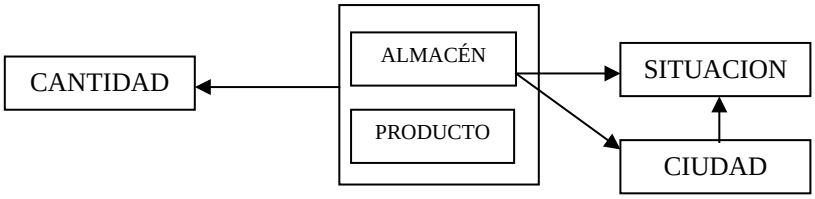
1. Una relación está en 1FN si y sólo si todos los dominios simples subyacentes contienen sólo valores atómicos (no hay atributos multivalor).
2. Una relación está en 2FN si y sólo si está en 1FN y todos los atributos no clave dependen por completo de la clave primaria.
3. Una relación está en 3FN si y sólo si está en 2FN y todos los atributos no clave dependen de manera no transitiva de la clave primaria. O dicho de otro modo, una relación está en 3FN si y sólo si los atributos no clave son:

   mutuamente independientes, y

   dependientes por completo de la clave primaria

donde no clave significa que no participa en la clave primaria, y mutuamente independientes significa que cualquiera de los atributos no depende funcionalmente de cualquier combinación de los otros.

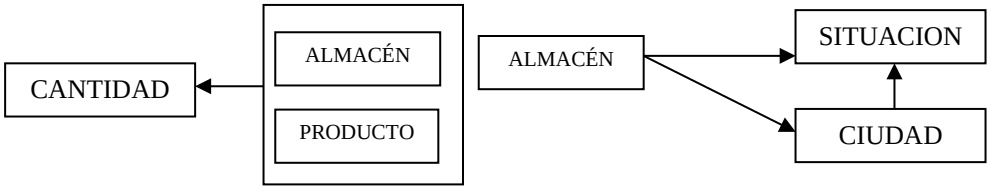
Veamos ahora un ejemplo de cómo convertir relaciones entre distintas formas normales.



En esta relación, ALMACÉN y PRODUCTO son la clave primaria de la relación, y cantidad, ciudad y situación son los atributos de la relación que dependen funcionalmente de la clave primaria (o de sus componentes) tal y como se indica en la figura.

Esta relación está en 1FN por definición, es decir, todos los atributos de la relación tienen valores atómicos (no hay tablas dentro de tablas). El problema de esta relación es que no tiene buen comportamiento respecto a las acciones de actualización (INSERT, DELETE y UPDATE), puesto que las redundancias existentes hacen que las variaciones que se deseen para un cierto numero de tuplas dejen la relación con un contenido incongruente. Por ejemplo, para modificar la situación en una tupla, habrá que modificarla en todas las tuplas donde aparezca dicha situación, puesto que de otro modo la dependencia ciudad → situación dejará de cumplirse. En general, el borrado puede eliminar información sobre ciertas dependencias, la inserción puede generar discordancias, y la modificación puede provocar el mismo efecto que una inserción incorrecta.

Es obvio que esta relación no está en 2FN puesto que no todos los atributos no clave dependen por completo de la clave primaria. De hecho, Situación y Ciudad dependen de un componente de la clave primaria. Por tanto, para pasar a 2FN habrá que descomponer la relación en otras relaciones más simples (y este será el procedimiento genérico que seguiremos para pasar de unas FN a otras superiores: descomponer las tablas, o lo que es lo mismo, realizar operaciones de proyección). En este caso, la siguiente descomposición nos permite obtener relaciones en 2FN:



La segunda de las relaciones separadas contiene relaciones transitivas, y por tanto, tampoco está en 3FN. Para obtener relaciones en 3FN hay que de nuevo dividir la relación en las siguientes relaciones:

y con esta separación ya se obtienen, definitivamente, las relaciones en 3FN.

Como detalle, hay que destacar que la última descomposición se podría haber realizado de dos formas distintas:

ALMACÉN →CIUDAD, CIUDAD → SITUACION, o
ALMACÉN → CIUDAD, ALMACÉN → SITUACION

Es evidente que en la segunda descomposición se pierde información sobre la dependencia CIUDAD → SITUACION. Por tanto, ésta sería una 'mala' descomposición. Habrá que obrar, por tanto, con cautela antes de decidir que tipo de descomposición se realiza en las relaciones con transitividad.

FORMA NORMAL DE BOYCE/CODD

El problema de la 3FN es que no maneja relaciones que:

Tiene varias claves candidatas,

Esas claves candidatas son compuestas, y

Las claves candidatas se traslapan (o sea, tienen por lo menos un atributo en común).

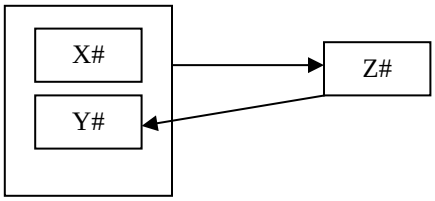
Por ello, se define la FNBC para el caso en el que existan más de una clave candidata, y que se cumplan dichas condiciones. Hay que introducir, por comodidad, el concepto de determinante, que se define como el atributo del cual depende funcionalmente algún otro atributo. Así pues, se define la FNBC como:

Una relación está en FNBC si y sólo si todo determinante es una clave candidata.

Se debe advertir que en el caso en el que no se den las condiciones anteriores, o no exista más de una clave candidata (sólo la clave primaria) la FNBC es completamente equivalente a la 3FN.

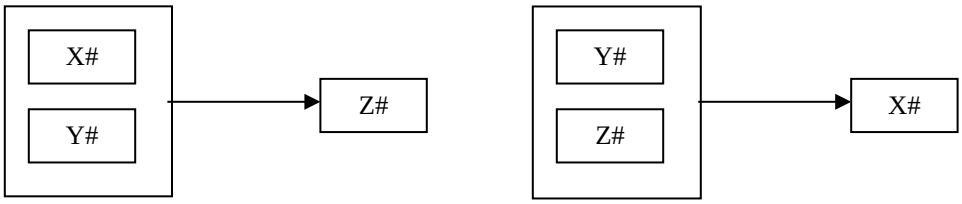
Dicho de otro modo, la definición implica que los únicos determinantes son las claves candidatas, o sea, las flechas del diagrama de dependencias funcionales sólo salen de claves candidatas.

Un problema serio, que no contempla la 3FN sería el siguiente diagrama DF:



En este caso, la relación no está en FNBC, y sería necesario descomponerla en dos relaciones ((X#, Z#), (Z#, Y#)), aunque las relaciones resultantes no podrían ser independientes entre sí.

Sin embargo, en el caso en que se diese simultáneamente cualquiera de estas dos posibles dependencias funcionales en una relación:



estaríamos ante una relación en FNBC.

CUARTA FORMA NORMAL

Existen relaciones en las que no existen dependencias entre los atributos, sin embargo la actualización de sus campos no es tarea trivial, puesto que la representación semántica de la relación así lo requiere. Por ejemplo, en una tabla en la que se haga constar los profesores que imparten ciertas asignaturas y que utilizan una serie de textos como ayuda a la enseñanza, es evidente que no habrá dependencia entre cada uno de los atributos; sin embargo, en el caso en que se desee introducir un profesor nuevo en la tabla, habrá que crear una fila nueva por cada asignatura que vaya a impartir, y a su vez una fila por cada texto que se utiliza de forma común por todos los profesores de esa asignatura.

Estamos por tanto ante una situación de patrones repetitivos en los que no existen dependencias funcionales individuales, pero que obligan a un tratamiento poco óptimos en las operaciones de actualización de la relación.

Para analizar este tipo de casos no podemos utilizar las dependencias funcionales que hemos estado manejando hasta ahora. Introducimos un nuevo concepto: *dependencia multivaluada*, que son una generalización de las dependencias funcionales. Por ejemplo, en la relación de nuestro ejemplo tenemos dos dependencias multivaluadas (ASIGNATURA → PROFESOR, y ASIGNATURA → TEXTOS), puesto que una asignatura puede estar impartida por varios profesores, y una asignatura puede utilizar varios textos como apoyo a la enseñanza.

QUINTA FORMA NORMAL

Se basa en los conceptos de Dependencia de Reunión y el teorema de Fagin.

4.- DISEÑO FÍSICO

El rendimiento de un SGBD es la última medida en el diseño de una base de datos. Un DBA puede mejorar el rendimiento ajustando algunos parámetros del SGBD, e identificando cuellos de botella y problemas de hardware. Sin embargo, el primer paso que hay que dar para obtener un buen rendimiento de la base de datos es hacer una buena toma de decisiones en cuanto al diseño físico de la misma.

4.1.- INTRODUCCIÓN AL DISEÑO FÍSICO

La descripción de la carga y los requisitos de los usuarios sobre el rendimiento de las consultas son la base para la toma de decisiones de un buen diseño físico.

La descripción de la carga del sistema incluye los siguientes elementos:

Una lista de consultas y sus frecuencias, como una fracción de todas las consultas y actualizaciones.

Una lista de actualizaciones y sus frecuencias.

Objetivos de rendimiento para cada tipo de consulta y actualización.

Conviene tener en cuenta que las actualizaciones utilizan un parámetro para seleccionar las tuplas que se deben actualizar. El uso de índices para estos campos puede resultar interesante. Sin embargo, las actualizaciones conllevan una sobrecarga de actualización de índices, con lo que actualizar campos sobre los que hay definidos índices puede hacer más lento el proceso de actualización.

Las decisiones que hay que tomar durante el diseño físico de una base de datos son las siguientes:

Qué índices crear

- Qué relaciones y campos de las relaciones se indexarán
- Tipo de índice a utilizar

Cuándo realizar cambios en el esquema conceptual para mejorar el rendimiento:

- Normalización alternativa de esquemas, cuando se pueda llegar, por ejemplo, a una descomposición FNBC o 3FN por caminos distintos.
- Desnormalización
- Particionado
- Vistas

4.2.- CRITERIOS PARA LA SELECCIÓN DE ÍNDICES

A continuación se proporcionan una serie de criterios que ayudarán a decidir que índices se deben crear y cómo se deben crear.

No hay que construir un índice a menos que alguna consulta se vaya a beneficiar de dicho índice. Cuando sea posible, seleccionar un índice que pueda mejorar la velocidad de más de una consulta.

Los atributos que aparecen en la cláusula WHERE son los candidatos a recibir índices.

Después de obtener un listado de los índices que sería conveniente crear, hay que considerar el impacto que provocarán en las sentencias de actualización:

Si el mantenimiento del índice ralentiza operaciones de actualización muy frecuentes, habrá que considerar la eliminación del índice.

Considerar que un índice puede acelerar el proceso de actualización, cuando el índice se define sobre los campos que se utilizan como criterio de selección de aquellas tuplas que se van a actualizar.