DISEÑO LÓGICO RELACIONAL

Tabla de Contenidos

3.- Modelo Relacional

Con el modelo Entidad/Relación se ha obtenido una representación conceptual de la información que se desea manejar (universo del discurso) independientemente de cuál sea el sistema gestor en el que se implemente. En este bloque se dará el siguiente paso, que es convertir el modelo conceptual en un modelo lógico ya adaptado a un tipo concreto de SGBD. En este caso el tipo de SGBD que se va a utilizar es el relacional, presentado por Codd en 1970 y que sigue vigente en la actualidad.

Hasta ese momento los SGBD imperantes del mercado estaban basados en los modelos de red y jerárquico, que no habían superado el grave inconveniente que suponía la dependencia de las aplicaciones desarrolladas en ellos respecto a las estructuras de datos.

El modelo relacional representa la base de datos como una colección de relaciones. En términos informales, cada relación asemeja una tabla. El objetivo fundamental de este modelo era mantener la independencia de esta estructura lógica respecto al modo de almacenamiento y a otras características de tipo físico.

El trabajo publicado por Codd en ACM presentaba un nuevo modelo de datos que perseguía una serie de objetivos; entre ellos:

- Independencia física.
- Independencia lógica.
- Flexibilidad.
- Uniformidad.
- Sencillez.

Para conseguir esos objetivos, Codd introduce el concepto de relación –tabla- como estructura básica del modelo.

Con respecto a la parte dinámica del modelo, se propone un conjunto de operadores que se aplican a las relaciones: unos de la teoría de conjuntos, y otros introducidos específicamente para el modelo relacional. Todos ellos conforman el ***álgebra relacional***.

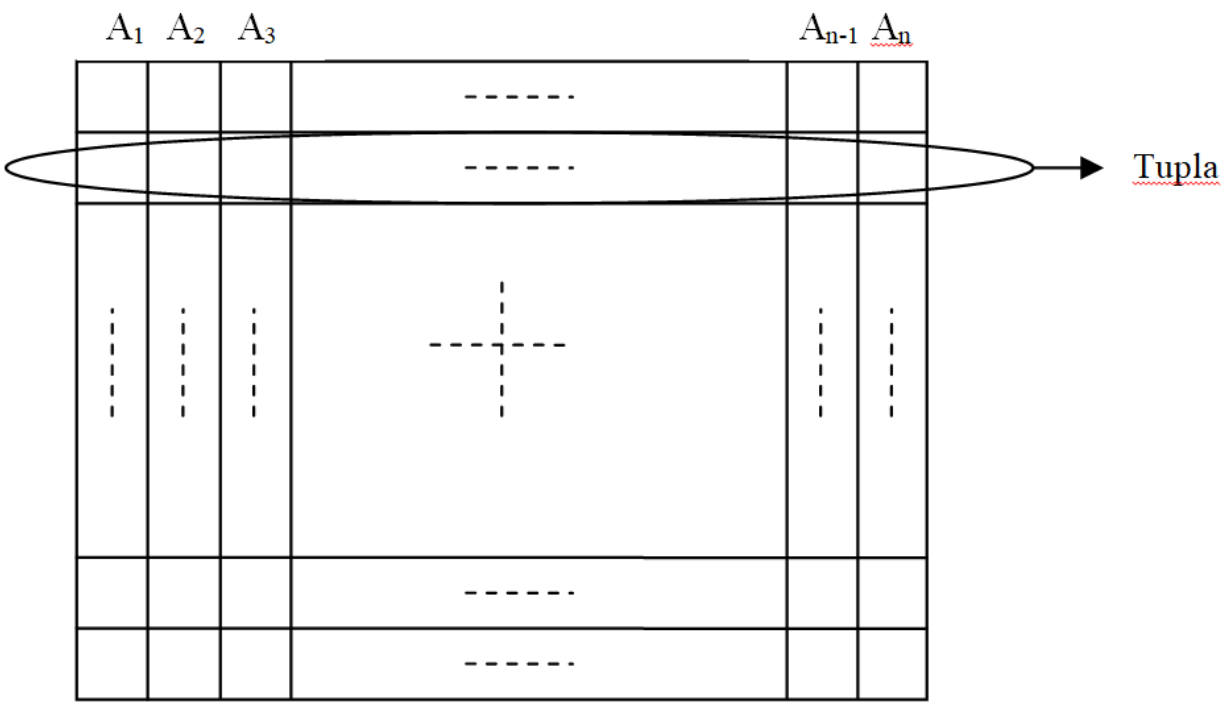
La **teoría de la normalización**, cuyas tres primeras formas normales fueron introducidas por Codd, eliminan dependencias entre atributos que originan anomalías en la actualización de la base de datos y proporciona una estructura más regular en la representación de relaciones, constituyendo el soporte para el diseño de bases de datos relacionales.
.

3.1.- Terminología relacional

Una base de datos relacional está compuesta por un conjunto de relaciones, de ahí su nombre.

El modelo relacional se ocupa de la estructura, de la integridad y de la manipulación de los datos. Cada una de estas partes tiene sus propios términos especiales. A continuación se exponen los términos relacionados con la estructura de datos.

- Una **`relación`** es un conjunto de `tuplas` con la misma estructura. Sería algo similar a una tabla con información.
- Una **`tupla`** es un conjunto de pares `atributo-valor.` Se podría asemejar a una fila de una tabla `(relación)`.



- La **cardinalidad** es el número de filas (tuplas) de una relación.
- El **grado** es el número de columnas (atributos) de una relación.
- Los **atributos** representan las propiedades de la tabla (campos en la terminología tradicional) y se identifican por un nombre que los distinguen del resto. Se corresponderían con las columnas de la tabla.

- Un **dominio** es un conjunto de valores. Todo atributo tiene asociado un dominio que indica cuales son los valores válidos que puede contener dicho atributo. En algunos casos a los dominios se les puede añadir el valor especial NULO. Un valor NULO en un atributo significa que a ese atributo no se le ha asignado ningún valor todavía. Si a un dominio **no** se le añade el valor NULO, esto significa que no pueden dejarse valores vacíos en los atributos que pertenezcan a dicho dominio.

- **Claves:**
  - **Clave candidata**: atributo o conjunto de atributos que identifican unívoca y mínimamente a cada una de las tuplas de la relación. Por la propia definición de relación, siempre hay al menos una clave candidata (el conjunto de todos los atributos). Si no se cumpliera la condición de minimalidad se eliminarían aquellos atributos que lo impidiesen. Una relación puede tener más de una clave candidata.
  - **Clave principal**: clave candidata elegida para identificar cada una de las tuplas.

- **Clave alternativa**: clave candidata no elegida como principal.
- **Clave foránea**: clave primaria exportada a otra relación, lo que permite relacionar una tabla con otra.

De manera formal, se define una `relación R` sobre un conjunto de `dominios` D_1, D_2,...D_n como una estructura con dos partes:

- **Cabecera**

  La cabecera de una relación es un conjunto de pares atributo-dominio $\{(A_1,D_1), (A_2,D_2),...., (A_n,D_n)\}$.

  La cabecera indica qué atributos tiene la relación (cómo se llama cada uno) y el dominio del que cada atributo puede obtener sus valores. La cabecera de una relación es estática, esto es, no varía con el tiempo. También se denomina *esquema de la relación*.

- **Cuerpo**

  El cuerpo de una relación es un conjunto de tuplas con pares atributo-valor de la forma: $\{(A_1,V_1), (A_2, V_2), ..., (A_n, V_n)\}$.

  El cuerpo contiene la información almacenada en la relación (la información almacenada en la tabla). El número de tuplas es variable a lo largo del tiempo, según se añada o elimine información de la relación.

EJEMPLO

PROVEEDOR

CODIGO	NOMBRE	CIUDAD
1	Salazar	Londres
2	Jaime	París
3	Bernal	París
4	Corona	Londres
5	Aldana	Atenas

Esta tabla es la relación PROVEEDOR.

Las tuplas son cada una de estas filas

- 1,Salazar,Londres
- 2,Jaime,París
- 3,Bernal,París
- 4,Corona,Londres
- 5,Aldana,Atenas

La cardinalidad es 5 y el grado es 3.

La cabecera sería: `{(CODIGO,NATURALES),(NOMBRE,TEXTO50),(CIUDAD,CIUDADES)}`

El dominio NATURALES es el conjunto de los números naturales, el dominio CIUDADES es el conjunto de nombres de ciudades de los proveedores y el dominio TEXTO50 es el conjunto de todas las posibles combinaciones de letras de hasta 50 caracteres de longitud.

Y el cuerpo:

`{(CODIGO,1),(NOMBRE,Salazar),(CIUDAD,Londres)}`

`{(CODIGO,2),(NOMBRE,Jaime),(CIUDAD,París)}`

`{(CODIGO,3),(NOMBRE,Bernal),(CIUDAD,París)}`

`{(CODIGO,4),(NOMBRE,Corona),(CIUDAD,Londres)}`

```
{(CODIGO,5),(NOMBRE,Aldana),(CIUDAD,Atenas)}
```

Por tanto:

- Las relaciones representan en el diseño lógico lo que las entidades en el diseño conceptual.
- Una relación tiene un nombre, una serie de atributos y un conjunto de tuplas. El nombre debe ser único, es decir, no puede haber dos relaciones con el mismo nombre en la misma base de datos.
- Los atributos representan las propiedades inherentes de las relaciones.
- Las tuplas representan los valores que toman los diferentes atributos para cada elemento de la relación.

Por otra parte:

- Las relaciones se representan mediante tablas bidimensionales.
- Las columnas de una tabla corresponden con los atributos de la relación.
- Las filas de una tabla se corresponden con las tuplas de la relación.
- Las tuplas coinciden con las ocurrencias de una entidad o de una interrelación del modelo E/R.

Se podrían establecer de forma informal las siguientes equivalencias:

- Relación.....Tabla
- Tupla.....Fila o registro
- Cardinalidad .....Número de filas
- Atributo.....Columna o campo
- Grado.....Número de columnas
- Dominio.....Conjunto de valores permitidos

3.2.- Restricciones en una base de datos relacional

En el modelo relacional, como ocurre en otros, existen estructuras u ocurrencias no permitidas o *restricciones*. Los datos almacenados han de cumplir las estructuras del modelo (por ejemplo, no tener tuplas duplicadas) y han de cumplir las restricciones de usuario para constituir una ocurrencia válida del esquema.

3.2.1.- Restricciones implícitas o inherentes basadas en el modelo

Son las restricciones derivadas de la propia naturaleza del modelo, que no tienen que ser definidas por el usuario e imponen limitaciones a la hora de modelar nuestro universo del discurso.

La base matemática de la teoría de conjuntos en la que se apoya el modelo junto al único constructor que define, la relación, implica que:

1) No se permite la existencia de tuplas repetidas (un conjunto no puede tener elementos iguales) lo que obliga a la existencia de una clave primaria o identificador (conjunto mínimo de atributos que identifican de forma unívoca las tuplas de una relación).

2) Ningún atributo que forme parte de la clave primaria puede tomar valores nulos, caso de no ser así no se podría identificar unívocamente las tuplas de la relación. A esto se conoce como **integridad de entidad**.
3) Las relaciones son tablas de dos dimensiones, por lo que no se permiten grupos repetitivos en los atributos, es decir, que cada atributo sólo puede tomar un valor del dominio.
4) El orden de las tuplas y de los atributos es irrelevante dentro de la relación.
5) Dentro de una relación no pueden existir atributos con el mismo nombre.
6) Cada relación contiene un solo tipo de registro con un número fijo de campos.

3.2.2.- Restricciones semánticas o de usuario

Las restricciones de usuario pueden considerarse como unas reglas de validación definidas sobre un conjunto de tuplas, atributos o dominios que deben ser verificadas para que éstos constituyan una ocurrencia válida del esquema.

A) **Restricción de unicidad (UNIQUE)**

Los valores de uno o varios atributos no pueden repetirse en las distintas tuplas de la relación.

B) **Restricción de obligatoriedad (NOT NULL)**

Indica que el atributo debe tomar siempre un valor y no admite valores nulos (desconocidos o inexistentes).

C) **Restricción de clave primaria (PRIMARY KEY)**

Se denomina clave candidata al conjunto de atributos mínimo capaz de identificar unívocamente cualquier tupla de la relación. Si existe más de una clave candidata, a la elegida se le denomina clave primaria (PRIMARY KEY) y al resto que no lo ha sido se les denomina claves alternativas. En cualquier caso, a todas se les aplican las restricciones de unicidad y obligatoriedad.

En la práctica se elige como clave primaria la que se considera de más importancia. Como regla general se debe preferir las claves candidatas que tengan un menor número de atributos, idealmente sólo uno. De entre estos se debe elegir el que se considere que es más significativo como identificador de cada tupla.

D) **Restricción de clave ajena (FOREIGN KEY)**

Se utiliza para indicar que un conjunto de atributos de la relación es clave primaria en otra o la misma relación. De esta forma podemos enlazar relaciones entre sí. Los atributos que son clave ajena en una relación no necesitan tener los mismos nombres que los atributos clave de la relación primaria con la que se corresponden pero sí deben estar definidos necesariamente sobre el mismo dominio.

Las claves ajenas son el mecanismo que se utiliza en el modelo relacional para crear *asociaciones* entre datos contenidos en distintas tablas. Al incluir una clave ajena en una relación se está creando un vínculo entre cada tupla de la relación y una tupla en la relación objetivo. El tipo de

asociación concreto no queda recogido en el modelo relacional, por lo que lo tiene que conocer el diseñador.

La clave ajena no necesita ser un componente de la clave primaria de la relación que la contiene.

E) **Restricción de integridad referencial**

Está vinculada directamente a la restricción de clave ajena. Indica que los valores de la clave ajena (relación hija o que referencia) se han de corresponder con los valores de la clave primaria (relación padre o referenciada) o bien ser nulos. Las relaciones padre e hija no han de ser necesariamente distintas.

El modelo relacional además de permitir enlazar relaciones entre sí dando lugar a la estructura de la base de datos, también permite definir las acciones a realizar cuando se produce un borrado o modificación de una tupla en la relación padre referenciada por una relación hija.

Las acciones pueden ser:

- **CASCADE**

  Para la actualización consiste en permitir actualizar el valor de la clave primaria de una tupla de la relación padre y actualizar todos los valores de la clave ajena de la relación hija que la referencian.

  Para el borrado consiste en permitir borrar una tupla de la relación padre y borrar todas las tuplas de la relación hija que la referencian.

- **NO ACTION / RESTRICT**

  Si la relación padre tiene tuplas asociadas en la relación hija la operación no se permitirá.

- **SET NULL**

  Para la actualización consiste en permitir actualizar el valor de la clave primaria de una tupla de la relación padre y actualizar todos los valores de la clave ajena de la relación que la referencian al valor NULL.

  Para el borrado consiste en permitir borrar una tupla de la relación padre y actualizar todos los valores de la clave ajena de la relación hija que la referencian al valor NULL.

  La clave ajena debe permitir valores nulos para que esta opción tenga sentido.

- **SET DEFAULT**

  Su funcionamiento es similar al caso anterior con la excepción de que el valor al que se ponen las claves ajenas de la relación hija es un valor por defecto que se habrá especificado en la relación para dicha clave ajena.

F) **Restricción de verificación (CHECK)**

Establece la condición que debe cumplir un atributo o varios de una relación para realizar la operación (inserción, borrado o modificación), en caso contrario la operación se rechaza. No necesita tener nombre.

G) **Restricción de aserción (ASSERTION)**

Su funcionamiento es idéntico al de las restricciones de verificación con la diferencia de que el ámbito de aplicación (la verificación) se extiende a más de una relación. Deben tener nombre.

H) **Disparadores (TRIGGER)**

Al contrario que el resto de restricciones vistas en las que se rechaza la operación si no se cumple la condición, en el caso de los disparadores la acción especificada se ejecuta siempre y cuando sí se cumpla la condición.

Los disparadores son procedimientos especiales que ayudan al diseñador de la base de datos a recoger aquellos supuestos semánticos del universo del discurso que no se han podido mantener en el modelo conceptual.

4.- Representación del modelo relacional

El modelo relacional se representa habitualmente de dos formas: textual o gráfica.

4.1.- Representación textual del modelo relacional

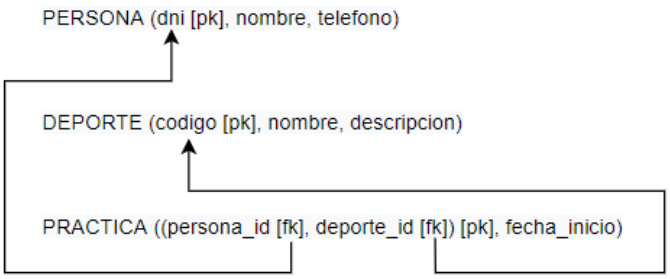
En la representación textual se da una lista con las relaciones y sus atributos. Mediante flechas se indica a qué relación corresponde una clave foránea.

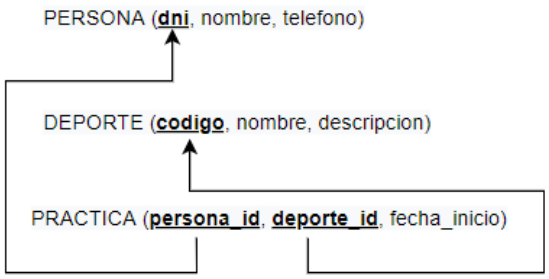
No hay unanimidad para este tipo de representación más allá del uso de flechas. Algunas de las representaciones que se pueden encontrar son:

- Resaltar en negrita y subrayado la clave principal. Resaltar con línea discontinua la clave alternativa. Utilizar flechas entre claves ajenas y primarias. Resaltar con doble subrayado las claves alternativas.
- Junto al atributo principal se escribirá [pk], si es clave ajena [fk] y si es clave alternativa [ak]. Utilizar flechas entre claves ajenas y primarias.

En cualquier caso, por claridad no especificaremos en esta representación la obligatoriedad u opcionalidad de los atributos ni tampoco las acciones asociadas a las claves ajenas en las operaciones de modificación y borrado.

EJEMPLO





4.2.- Representación gráfica del modelo relacional

Lo más habitual es utilizar la notación gráfica con la notación Crow´s Foot, que utiliza:

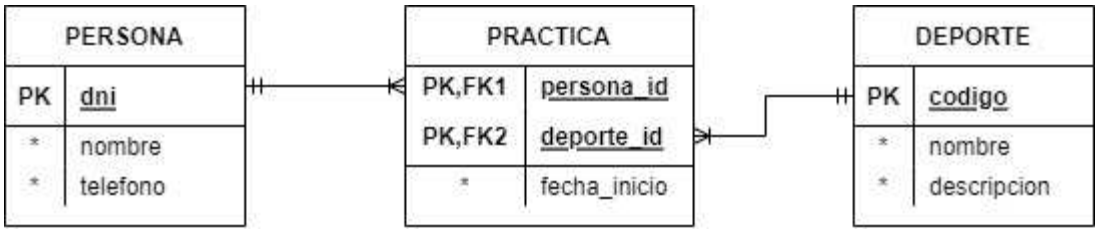
- Nodos (cajas) para representar las relaciones.
- Líneas que unen nodos para representar las modalidades de las entidades y las relaciones.

Correspondencia entre notaciones

(min, max)	Crow´s foot
(0,N) - (1,1)	
(1,N) - (0,1)	
(0,1) - (1,1)	
(1,1) - (1,1)	
(0,1) - (0,1)	

- Símbolos para indicar determinadas restricciones:
  - Clave primaria _____ PK
  - Clave alternativa _____ AK
  - Clave foránea _____ FK
  - Clave primaria ajena _____ PF
  - Clave alternativa foránea ____ AF
  - Obligatorio (NOT NULL) _____ *

EJEMPLO



3.- Conversión del E/R al Relacional

Con el modelo conceptual obtenido mediante el modelo E/R y sabiendo que el único constructor del modelo relacional es la **relación**, queda ahora aprender cómo convertir el modelo E/R en el modelo relacional equivalente.

Para ello se realizan en una serie de pasos bastante mecánicos a través de los cuales se van convirtiendo los distintos elementos de un modelo a sus equivalentes en el otro. Las reglas de transformación que se aplican en cada momento dependen del tipo de objeto del esquema conceptual que se deba transformar y en el caso de las relaciones, de las cardinalidades y modalidades de las entidades participantes.

Hay que hacer notar, sin embargo, que hay elementos que no se pueden transformar directamente y que hay que documentar en otra forma, por ejemplo con un documento anexo.

3.1.- Conversiones previas

Una restricción inherente al modelo relacional es que no pueden existir grupos repetitivos. Por tanto, para que los objetos obtenidos en el modelo conceptual satisfagan esta restricción hay que eliminar las siguientes anomalías:

3.1.1.- Eliminación de atributos compuestos

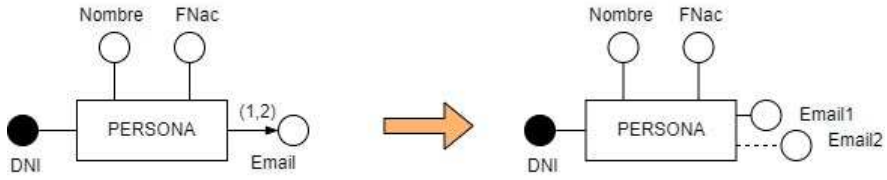
Todo atributo compuesto que provenga de una entidad o interrelación debe ser descompuesto en varios atributos simples, respetando los dominios que tenían en el atributo compuesto (que queda eliminado).

3.1.2.- Eliminación de atributos multivaluados

La eliminación del atributo multivaluado dará lugar a varias transformaciones dependiendo de los supuestos semánticos. Se distinguen los siguientes casos.

- **Primer caso**

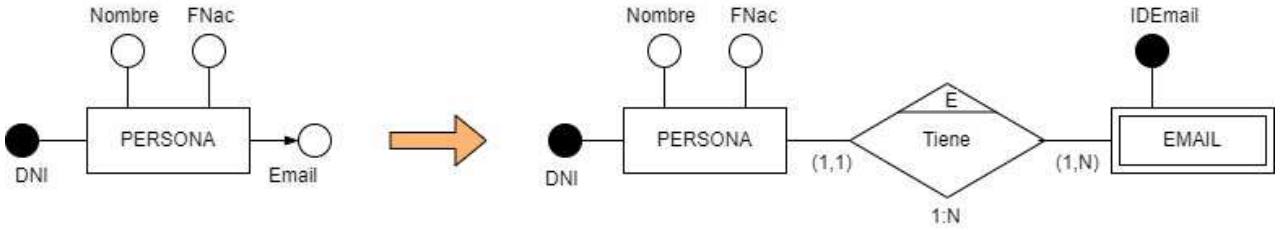
  Si el atributo multivaluado puede tomar pocos valores y son conocidos para la mayoría de los ejemplares, es posible descomponerlo en varios atributos simples.



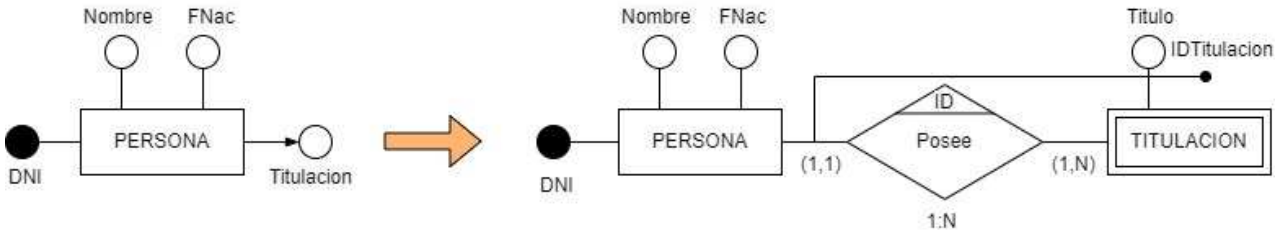
- **Segundo caso**

  Si el atributo multivaluado puede tomar muchos valores y son desconocidos para la mayoría de los ejemplares. En este caso se crea una interrelación y una entidad. Hay dos alternativas que dependen del contexto del problema:

1) Se crea una nueva entidad débil en existencia con una relación de uno a muchos. La entidad débil tendrá un solo atributo univaluado que podrá identificar de manera única cualquier ejemplar de la entidad creada.



2) Esta opción es parecida a la anterior, pero un valor del atributo multivaluado puede pertenecer a más de un ejemplar de la entidad a la que describe, por lo que la dependencia es en identificación.



3.2.- Transformación de dominios

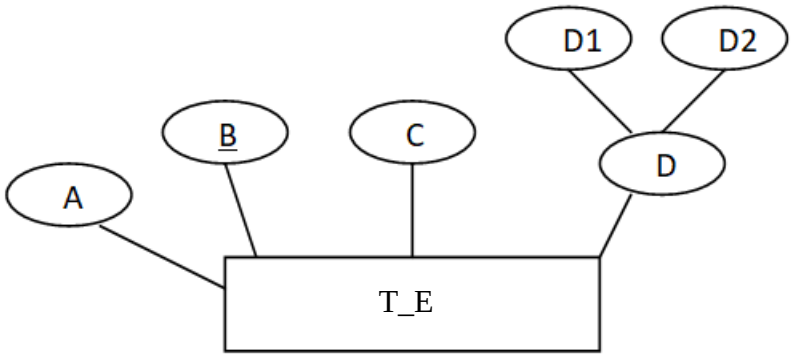
Un dominio es un conjunto de valores atómicos del mismo tipo y finito. Todo dominio tiene asociado un nombre y un formato que puede definirse por extensión (especificando la lista de valores permitida) o por intensión (especificando un tipo de datos).

Un dominio en el esquema conceptual será transformado en el mismo dominio en el modelo relacional. En el esquema conceptual no representaremos los dominios que más tarde deben ser reflejados en el esquema relacional para mantener la legibilidad en el diseño. Un dominio puede estar asignado a uno o varios atributos de distintas relaciones.

En los esquemas a continuación se representarán los atributos que forman parte de la **clave primaria subrayados**, los que forman la **clave alterna con doble subrayado**, y los atributos que son **claves foráneas en negrita**.

3.3.- Transformación de entidades

Para cada **tipo de entidad** del modelo ER extendido (sea fuerte o débil), haremos una tabla en el modelo relacional y colocaremos como atributos de la tabla todos los atributos simples o componentes simples de los atributos compuestos. Entre todos ellos, elegiremos una clave primaria.



T_E (A,B,C,D1,D2)

3.4.- Transformación de entidades débiles

Cada **entidad débil** generará una tabla que incluirá todos sus atributos, añadiéndose a ésta los atributos que son clave primaria de la entidad fuerte con la que esté relacionada. Estos atributos añadidos se constituyen como **clave foránea** que referencia a la entidad fuerte. Seguidamente, se escogerá una clave primaria para la tabla creada, que será la propia de la entidad débil en el caso de las **relaciones de existencia**, o la unión de la fuerte y la débil en el caso de las **relaciones de identificación**.

```
T_EDIdentificación (A,C,D1,D2,CP,B)  donde B es clave parcial de
                                     la entidad débil
T_EDExistencia (A,C,D1,D2,CP,B) donde  B es la clave primaria de
                                     la entidad débil
```

3.5.- Transformación de relaciones binarias

Cualquier relación binaria en el esquema conceptual se transformará en una de estas dos formas:

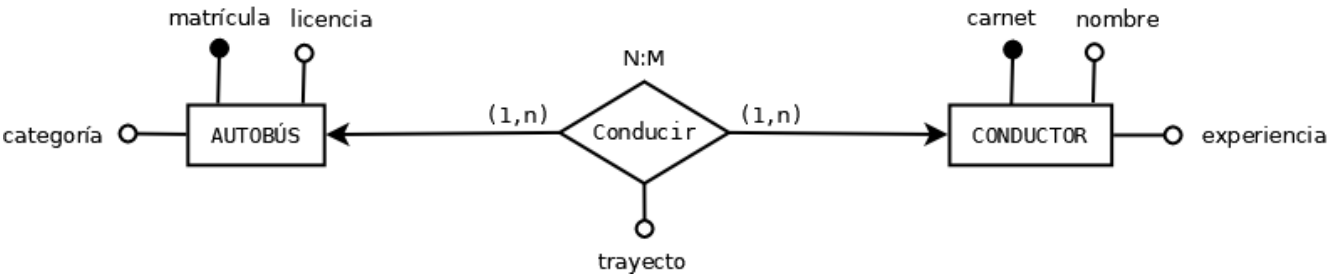
1. Creación de relación (tabla)
2. Propagación de clave

La elección de una u otra dependerá del tipo de la cardinalidad de la relación y de las modalidades de las entidades participantes (es decir, si participan parcialmente -cardinalidad mínima 0-, o totalmente -cardinalidad mínima 1).

3.5.1.- Relaciones N:M

Se transforma en una relación con los siguientes atributos: los identificadores (clave primaria) de las entidades asociadas y los atributos propios de la relación si los tuviera.

La unión de los identificadores será la **clave primaria** de la relación creada. Cada uno de estos identificadores es una clave ajena o foránea que referencia a la tabla de la que proviene.



AUTOBÚS(matrícula, licencia, categoría)

CONDUCTOR(carnet, nombre, experiencia)

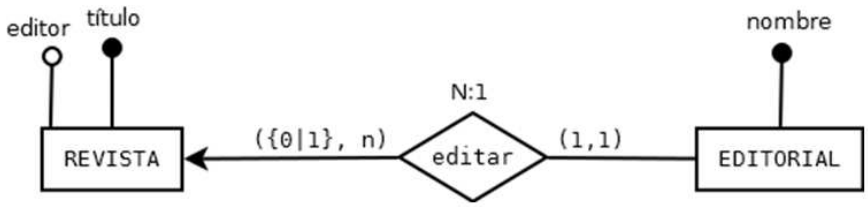
CONDUCIR(**carnet, matrícula**, trayecto)

3.5.2.- Relaciones 1:N

Se puede modelizar de varias formas según como sea la cardinalidad mínima de la entidad con cardinalidad máxima 1.

- **Primer caso**

  Si **ambos participan de forma total**, o el tipo de entidad que interviene con **cardinalidad máxima muchos participa de forma parcial**, entonces cada tipo de entidad se transforma en una tabla, y el identificador del tipo de entidad que participa con cardinalidad máxima uno pasa a formar parte de la tabla correspondiente al tipo de entidad que participa con cardinalidad máxima muchos. Este atributo será definido como clave foránea. Si el tipo de interrelación tuviera atributos asociados, estos atributos pasan a formar parte de la tabla correspondiente al tipo de entidad que participa con cardinalidad máxima muchos.

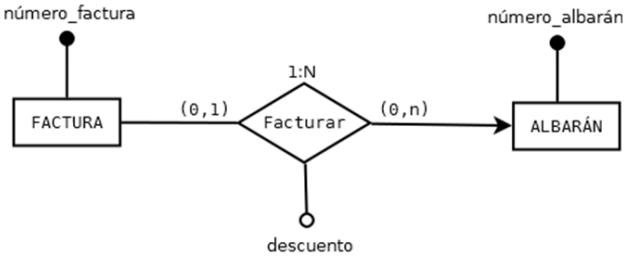


REVISTA(título, editor, **nombre**)

EDITORIAL(nombre)

- **Segundo caso**

Si **ambos tipos participan de forma parcial**, o únicamente **el tipo de entidad que interviene con cardinalidad máxima uno participa de forma parcial**, creamos una tabla para el tipo de interrelación, formada por los identificadores de los tipos de entidad que intervienen en la interrelación y por todos los atributos de la interrelación. La **clave principal** de esta tabla será el atributo identificador correspondiente al tipo de entidad que interviene con cardinalidad máxima muchos, y será necesario definir como claves foráneas los atributos identificadores correspondientes a los dos tipos de entidad.



FACTURA(<u>número_factura</u>)

ALBARÁN(<u>número_albarán</u>)

FACTURAR(**<u>número_albarán</u>**, **número_factura**, descuento)

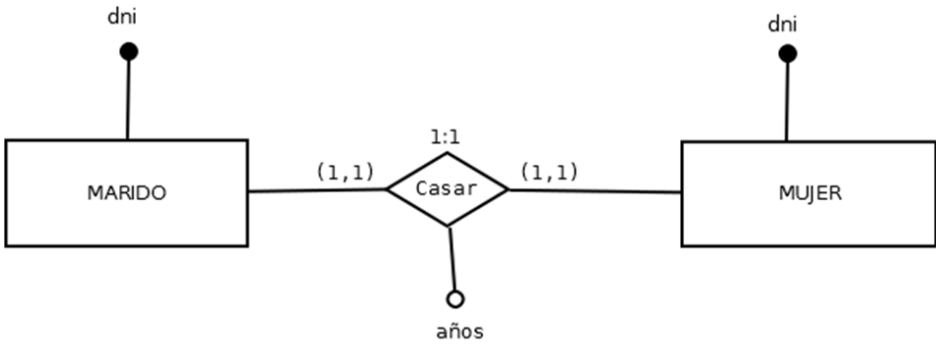
3.5.3.- Relaciones 1:1

Se puede transformar de varias formas, dependiendo de si al menos una de las entidades participa con modalidad (1,1) o si por el contrario, ambas participan con modalidad (0,1).

En toda transformación de una relación con cardinalidad 1:1 se creará un índice sin repetidos sobre la clave ajena propagada.

- **Primer caso**

Los **dos tipos de entidad participan de forma completa**, es decir, ambos tienen como cardinalidad mínima 1. En este caso, migraremos la clave de una cualquiera de las entidades a la otra.



Solución 1:

MARIDO(<u>dni</u>)

MUJER(<u>dni</u>, **dni_marido**)

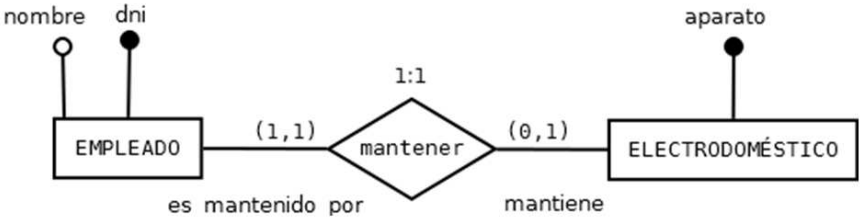
Solución 2:

MARIDO(<u>dni</u>, **dni_mujer**)

MUJER(<u>dni</u>)

- **Segundo caso**

Cuando **alguno de los tipos participa de forma parcial**, el identificador de la entidad que participa de forma total pasa como atributo de la tabla correspondiente a la transformación del otro tipo de entidad. Este atributo será definido como **clave foránea y alterna**, no pudiendo tomar valores nulos para las diferentes tuplas de la tabla.



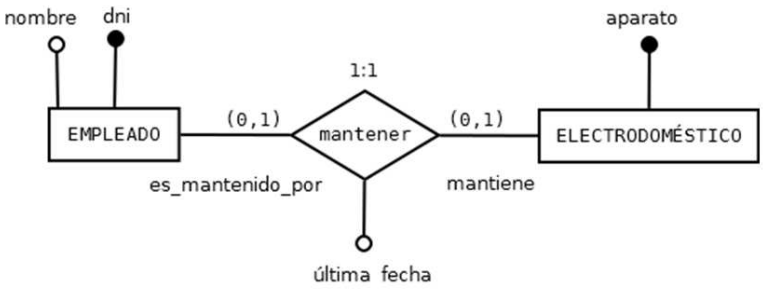
EMPLEADO(<u>dni</u>, nombre)

ELECTRODOMÉSTICO(<u>aparato</u>, **dni**)

- **Tercer caso**

Si **ambos tipos de entidad participan de forma parcial**, cada uno de ellos se transforma en una tabla y, además, se construye una nueva tabla para la interrelación, cuyos atributos serán los

identificadores de los dos tipos de entidad, que serán definidos como claves foráneas. La clave principal de la tabla generada será el identificador de uno de los tipos de entidad y, necesariamente, se definirá como **clave alterna** al identificador del otro tipo de entidad.



EMPLEADO(dni, nombre)

MANTENER(dni,aparato, última_fecha)

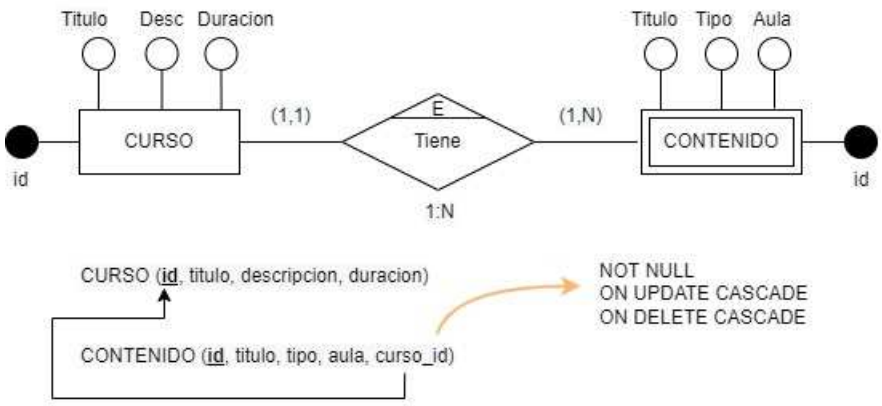
ELECTRODOMÉSTICO(aparato)

3.5.4.- Relaciones débiles

La transformación de las interrelaciones débiles se hace del mismo modo que las fuertes. Al haber transformado la entidad débil, la información necesaria referente a la debilidad de la relación (ya sea de existencia o identificación) ya se habrá migrado.

Además, podemos completar el modelo con la siguiente información:

La interrelación que provenga de una entidad débil con dependencia en existencia o identificación se modeliza como propagación de clave y la clave ajena no admitirá valores nulos. Además, también tendrá asociada la acción CASCADE tanto para la modificación como para el borrado.

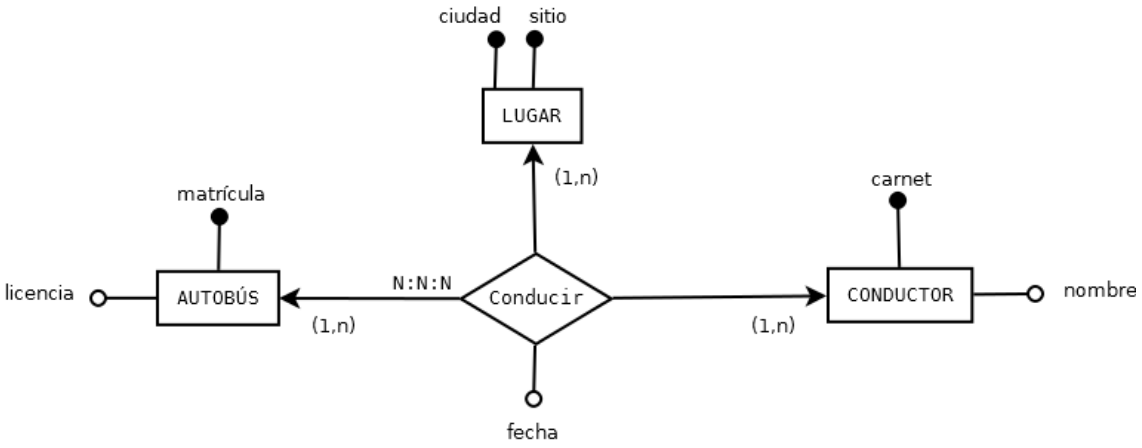


3.6.- Transformación de relaciones ternarias

En este caso, cada tipo de entidad se transforma en una tabla y el tipo de interrelación se transforma en una nueva tabla cuyos atributos serán los atributos de la relación, más la unión de los atributos claves de las entidades. Para ver cuál es la clave primaria de la relación o tabla, tendremos que ver el tipo de interrelación.

3.6.1.- Conectividad M:N:P

La relación que se obtiene de su transformación tiene como clave primaria todos los atributos que forman las claves primarias de las tres entidades interrelacionadas.

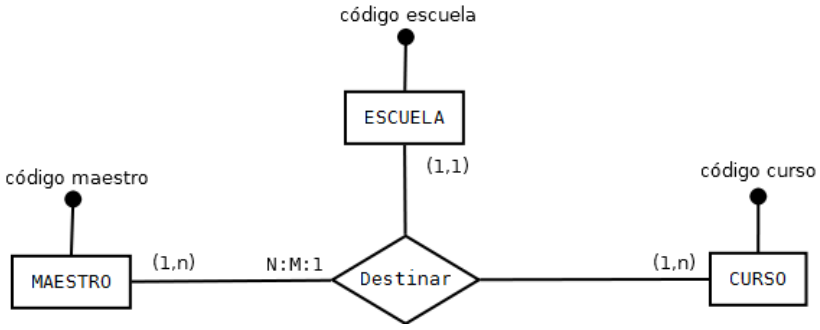


AUTOBÚS(matrícula, licencia)
CONDUCTOR(carnet, nombre)
LUGAR(ciudad, sitio)
CONDUCIR(**carnet**, **matrícula**, **ciudad**, **sitio**, fecha)

En estos ejemplos, no se va a reflejar las relaciones entre las claves externas y las tablas de la que proceden.

3.6.2.- Conectividad M:N:1

La relación que se obtiene de su transformación tiene como clave primaria todos los atributos que forman las claves primarias de las dos entidades de los lados de la interrelación etiquetados con M y con N.

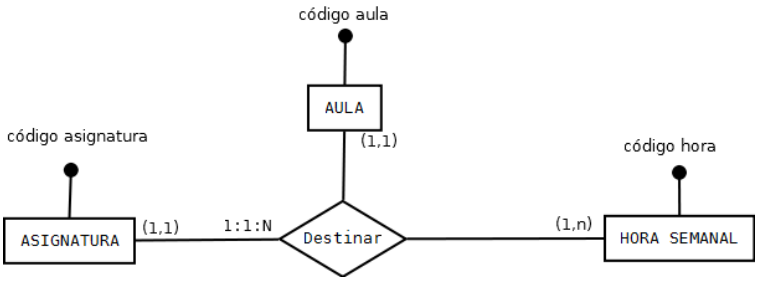


MAESTRO(código_maestro)
ESCUELA(código_escuela)

CURSO(<u>código_curso</u>)
DESTINAR(**<u>código_maestro, código_curso</u>, código_escuela**)

3.6.3.- Conectividad N:1:1

La relación que se consigue de su transformación tiene como clave primaria los atributos que forman la clave primaria de la entidad del lado N y los atributos que forman la clave primaria de cualquiera de las dos entidades que están conectadas con 1.

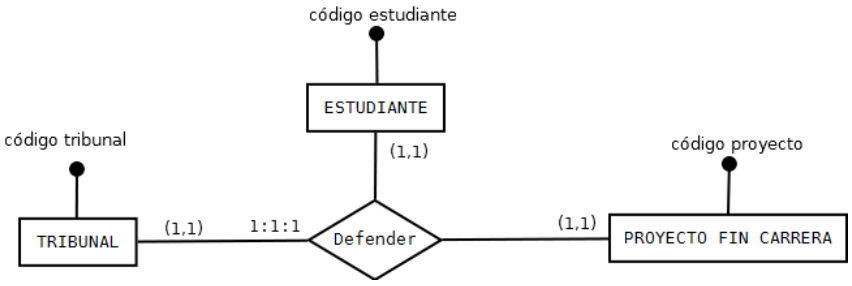


Opción 1:
AULA(<u>código_aula</u>)
ASIGNATURA(<u>código_asignatura</u>)
HORA_SEMANAL(<u>código_hora</u>)
DESTINAR(**<u>hora, código_asignatura</u>, código_aula**)

Opción 2:
AULA(<u>código_aula</u>)
ASIGNATURA(<u>código_asignatura</u>)
HORA_SEMANAL(<u>código_hora</u>)
DESTINAR(**<u>hora, código_aula</u>, código_asignatura)**

3.6.4.- Conectividad 1:1:1

La relación que se obtiene de su transformación tiene como clave primaria los atributos que forman la clave primaria de dos entidades cualesquiera de las tres interrelacionadas.



Opción 1:
TRIBUNAL(<u>código_tribunal</u>)
ESTUDIANTE(<u>código_estudiante</u>)
PROYECTO_FIN_CARRERA(<u>código_proyecto</u>)
DEFENDER(**<u>código_tribunal</u>**, **<u>código_estudiante</u>**, **código_proyecto**)

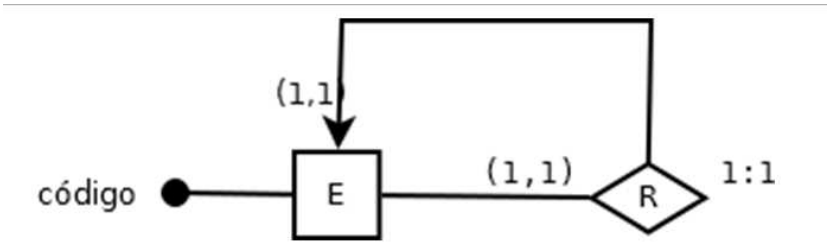
Opción 2:
TRIBUNAL(<u>código_tribunal</u>)
ESTUDIANTE(<u>código_estudiante</u>)
PROYECTO_FIN_CARRERA(<u>código_proyecto</u>)
DEFENDER(**<u>código_tribunal</u>**, **<u>código_proyecto</u>**, **código_estudiante**)

Opción 3:
TRIBUNAL(<u>código_tribunal</u>)
ESTUDIANTE(<u>código_estudiante</u>)
PROYECTO_FIN_CARRERA(<u>código_proyecto</u>)
DEFENDER(**<u>código_proyecto</u>**, **<u>código_estudiante</u>**, **código_tribunal**)

3.7.- Relaciones reflexivas

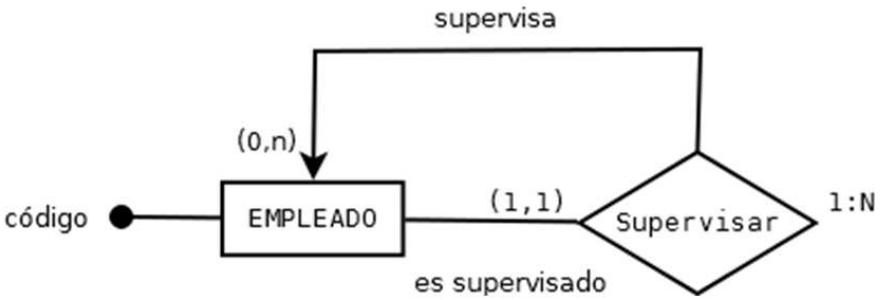
3.7.1.- Tipos de relación 1:N, 1:1

- **Cardinalidad mínima 1**: En la tabla generada para la entidad, se añade como clave foránea el identificador principal, cambiándole el nombre.



E(<u>código</u>, **código_E**)

- **Cardinalidad mínima 0:** Creamos una tabla para la interrelación y otra para la entidad. En tabla de la interrelación, colocaremos los atributos de la interrelación y el atributo clave de la entidad, que será la clave. Además, añadiremos como clave foránea la clave principal con otro nombre.



EMPLEADO(<u>código</u>)

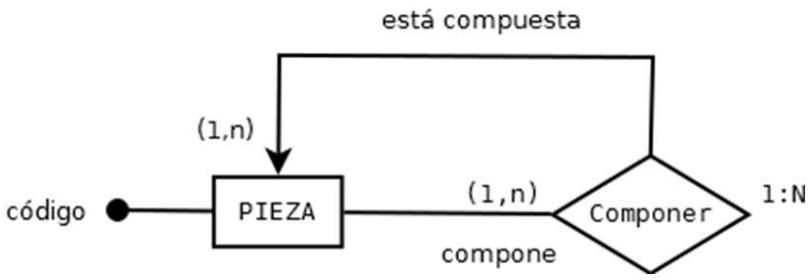
SUPERVISAR(<u>código</u>, **código_supervisor**)

3.7.2.- Tipo de relación N:M

Creamos una tabla para la interrelación, donde los atributos serían:

- ✔ Los atributos asociados al tipo de interrelación.
- ✔ El identificador del tipo de entidad desempeñando uno de los papeles en el tipo de interrelación.
- ✔ El identificador del tipo de entidad desempeñando el otro papel en el tipo de interrelación.

La clave de esta tabla será el identificador principal de la entidad por duplicado.



PIEZA(<u>código</u>)

COMPONER(**<u>código, código_componente</u>**)

3.8.- Transformación de jerarquías

Las reglas de transformación de las relaciones jerárquicas vendrán dadas según el tipo de relación que estemos analizando, así como de las interrelaciones que tengan el supertipo y los subtipos. Así, podemos distinguir los siguientes casos:

Eliminación del supertipo de entidad:

En este caso, se eliminará el supertipo de entidad, migrando todos sus atributos a cada uno de los subtipos, y cada uno de los tipos de interrelación que mantuviera el supertipo de entidad serán considerados para cada uno de los subtipos.

<u>Esta regla es apropiada en las relaciones jerárquicas totales y exclusivas cuando el número de atributos transferidos sea pequeño y no existan muchos tipos de interrelación en los que participe el supertipo de entidad.</u>

Eliminación de los subtipos de entidad:

Se desestimarán los subtipos de entidad, transfiriéndose todos los atributos de los subtipos al supertipo y cada uno de los tipos de interrelación que mantuvieran los subtipos de entidad serán considerados para el supertipo. Además, el atributo cualificador del tipo de interrelación pasará a formar parte del supertipo de entidad de la siguiente forma:

- ✔ Si el tipo de interrelación es exclusivo, no formará parte de la clave.
- ✔ Si el tipo de interrelación es inclusivo, formará parte de la clave, originando redundancia de los atributos del supertipos para cada instancia de los subtipos.
- ✔ Si el tipo de interrelación es parcial, podrá tomar valores nulos para representar a entidades que no se especializan.

En principio, esta regla se puede aplicar a cualquiera de los cuatro tipos de interrelaciones jerárquicas, aunque no siempre será conveniente (en las interrelaciones exclusivas y/o parciales generará muchos nulos para los atributos transferidos de los subtipos al supertipo).

Eliminación de la jerarquía:

El tipo de interrelación jerárquica se transformará en tantos tipos de interrelación uno a uno como subtipos de entidad estén presentes, manteniéndose los tipos de interrelación en los que intervienen tanto los subtipos como el supertipo de entidad. En los tipos de interrelación generados por la transformación, los subtipos de entidad participarán:

Si el tipo de interrelación jerárquica es exclusivo, participarán con cardinalidad mínima cero.

Si el tipo de interrelación jerárquica es inclusivo, participarán con cardinalidad mínima cero o uno.

El **supertipo participará con cardinalidades mínima y máxima igual a uno**, pudiendo considerarse que los **subtipos** de entidad actúan como tipos de **entidad débiles por identificación** con respecto al supertipo.

Además, añadiremos el **atributo especificador del tipo de jerarquía** a la **superclase**. A este atributo le daremos el mismo trato que en el caso anterior, pasando a ser clave o no según si es una relación exclusiva o inclusiva.

Esta regla es la de aplicación más general, pues tiene la ventaja de que el esquema resultante preserva la representación de las relaciones existentes entre el supertipo y los sutipos de entidad. Su principal inconveniente es que el nuevo esquema conceptual generado es bastante más complejo que el original e introduce redundancia lógica.

3.9.- Transformación de la dimensión temporal

Cuando en el modelo lógico, en una interrelación existe un atributo multivaluado que representa tiempo su transformación al modelo relacional se puede hacer de dos formas:

- **Primer caso**

  Convertir el atributo multivaluado en atributo univaluado y analizar cuál es la clave primaria de la relación resultante dependiendo de los distintos supuesto semánticos.

- **Segundo caso**

  Convertir el atributo multivaluado en entidad en el modelo E/R y transformar la interrelación inicial en otra de mayor grado.

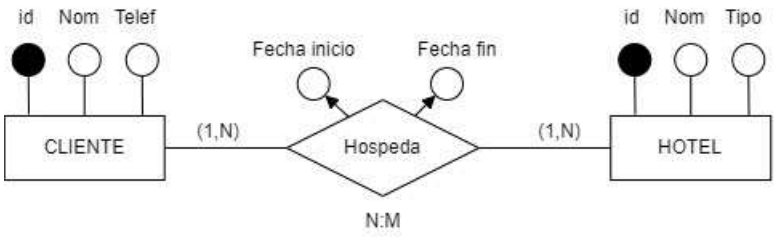
Vemos esto con un ejemplo.

Supongamos que una agencia de viajes desea crear una base de datos teniendo en cuenta los siguientes supuestos semánticos:

- En un hotel se pueden alojar varios clientes y estos se pueden hospedar en distintos hoteles a lo largo del tiempo.
- Se quiere saber en qué fechas se ha hospedado un cliente en un hotel.
- Además, se quiere imponer la restricción de que un cliente en una fecha solo puede hospedarse en un hotel.

Para el primer caso la clave de la relación Hospeda no será la suma de los identificadores de las entidades participantes y los atributos multivaluados como hemos visto en la transformación de la interrelación N:M, ya que dicha clave no sería mínima.

La clave primaria estará compuesta por el identificador de la entidad CLIENTE y el atributo multivaluado Fecha_inicio, respetando así el último supuesto semántico.

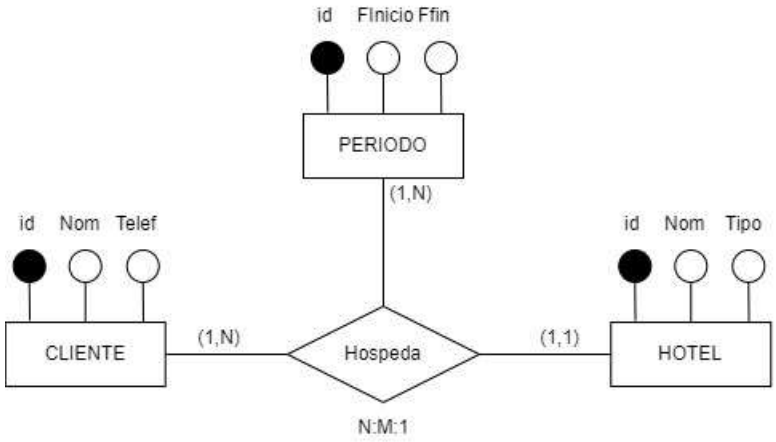


CLIENTE (**id**, nombre, telefono)

HOTEL (**id**, nombre, tipo)

HOSPEDA (**cliente_id**, **fecha_inicio**, fecha_fin, hotel_id)

Para el segundo caso el aspecto temporal se convierte en la entidad PERIODO y se aumenta el grado de la interrelación Hospeda, pasando de binaria a ternaria y se ajustan las cardinalidades de las entidades en función a los supuestos semánticos.



3.10.- Transformación de atributos derivados

Hay dos posibilidades:

- Almacenar el atributo derivado como atributo propio de la relación a la que pertenece. En tal caso se deben crear procedimientos para calcular su valor cada vez que se realicen operaciones de actualización sobre los elementos que intervienen en el cálculo de este.
- No almacenar el atributo derivado y se calcula su valor cuando se demande dicha información.

4.- El modelo relacional y los SGBD

Las siguientes restricciones del esquema E/R se deben tener en cuenta en el modelo relacional mediante checks, aserciones o disparadores:

- Cardinalidades mínimas de 1 en interrelaciones N:M y 1:N (excluyendo aquellas que se controlan con la restricción NOT NULL cuando se realiza propagación de clave).
- Cardinalidades máximas conocidas en interrelaciones binarias N:M y 1:N e interrelaciones ternarias.
- Exclusividad en las generalizaciones.
- Inserción y borrado en generalizaciones.
- Atributos derivados.
- Exclusividad entre interrelaciones.
- Atributos multivaluados obligatorios.

5.- Normalización

El modelo relacional describe los datos a través de relaciones que están enlazadas entre si por valores idénticos de campos clave: claves primarias y claves ajenas.

Sin embargo, una base de datos expresada en modelo relacional no está libre de inconsistencias de datos. Existe un proceso sistemático de eliminación de inconsistencias en bases de datos relacionales denominado *normalización*.

Supongamos que se tiene una base de datos relacional sencilla que consta de una única relación, que se muestra a continuación:

R

CodProveedor	Ciudad	País	CodPieza	Cantidad
1	Londres	Inglaterra	1	200
1	Londres	Inglaterra	2	1300
1	Londres	Inglaterra	4	345
2	Madrid	España	1	24
2	Madrid	España	6	988
3	Paris	Francia	2	200
4	Londres	Inglaterra	3	34

Se puede comprobar fácilmente que esta relación no está demasiado bien diseñada y que puede dar problemas de inconsistencia de datos por la abundancia de redundancias que aparecen. Por ejemplo, se repite muchas veces la información de que el proveedor 1 es de Londres.

Esta redundancia puede provocar graves problemas en la manipulación de los datos. Específicamente:

- **Problemas de actualización**

  Si por ejemplo, el proveedor 1 cambia de ciudad habría que actualizar TODAS las tuplas en las que aparece dicho proveedor. Si el operador se equivoca y no modifica alguna de ellas, se podrían producir dos respuestas a la pregunta ¿En qué ciudad reside el proveedor?

- **Problemas de inserción**

  Como los datos de un pedido y de un proveedor van entrelazados, no se podría añadir un proveedor hasta que no haga un pedido.

- **Problemas de eliminación**

  Si eliminamos un proveedor también se eliminan sus pedidos.

Para evitar este tipo de problemas se realiza un proceso denominado *normalización* para llevar la base de datos hasta lo que se llama una *forma normal*.

Una **forma normal** es una serie de condiciones que deben cumplir todas las relaciones de la base de datos para asegurar que se reducen las redundancias al mínimo posible.

Antes de definir las formas normales se deben definir dos conceptos fundamentales que se utilizan en las condiciones de estas formas normales: *dependencia funcional simple* y *dependencia funcional plena*.

5.1.- Dependencia funcional simple

La dependencia funcional simple, o dependencia funcional a secas, entre dos atributos se define de la siguiente forma:

> *Dados dos atributos X e Y de una relación R, se dice que Y depende funcionalmente de X si, y sólo si, en cualquier instante cada valor de X tiene asociado, en R, un único valor de Y. Se suele denotar como X->Y*

Dicho de una forma informal se dice que un atributo Y depende de otro X si el valor de Y está determinado por el valor de X pero no al revés.

5.2.- Dependencia funcional total, completa o plena

La dependencia funcional plena se define de la siguiente forma:

> *Dado un conjunto de atributos X (X_1, X_2, ..., X_n) y otro atributo Y, se dice que Y tiene dependencia funcional plena de X si Y depende de X y no se puede encontrar ningún subconjunto de X del que dependa Y. Se denota* $\boldsymbol{a \Rightarrow b}$

Dicho informalmente, si el valor de un atributo depende de una clave compuesta, debe depender de **toda** la clave y no de parte de ella.

5.3.- Primera Forma Normal

Una relación está en Primera Forma Normal (1FN) si se cumple la condición de que todos los dominios no contienen valores que sean conjuntos sino que son valores atómicos.

Dicho en otras palabras, una relación está en 1FN si no hay ningún atributo cuyo valor sea un conjunto de valores.

Es una forma normal que, por definición, toda tabla que cumpla las condiciones del modelo relacional debe cumplir, por lo que usualmente todas las bases de datos deberían estar en esta forma normal.

En el ejemplo anterior, la relación ya está en 1FN porque no hay atributos cuyos valores sean conjuntos.

Si se encontraran relaciones con atributos multivalor, habría que crear una nueva tabla por cada uno de los atributos multivalor junto con la clave primaria de la tabla en la que estaba el atributo. La clave primaria de la nueva tabla serán ambos atributos.

RELACIÓN PEDIDO NO NORMALIZADA: Existe más de un material (grupo repetitivo) por pedido.

CodPed	FecPed	CodProv	NomProv	DirProv	CodMat	DesMat	CantPed	PreUnit
0001	01/02/98	987	PEREZ	LUNA, 2	0024	CLAVOS	2000	5
					0025	PUNTAS	4000	7
					0038	CLIPS	8000	2
0003	02/02/98	999	MARQUEZ	GOYA, 3	1002	CUNAS	100	10000
					1004	CAMAS	300	23000
					2001	MESAS	500	5000
0004	04/02/92	555	LOPEZ	SOL, 10	5002	FLEXOS	50	100

ANOMALIAS: Para asignar un material a un pedido hay que comprobar si existe, y en función de esto realizar dos operaciones distintas según se trate de un alta o modificación.

SOLUCIÓN: 2 relaciones:

PEDIDO (CodPed, FecPed, CodProv, NomProv, DirProv)
PEDIDO-MATERIAL (CodPed, CodMat, DesMat, CantPed, PreUnit)

5.4.- Segunda Forma Normal

Una relación está en 2FN si, y solo si:

1.- Está en 1FN.

2.- Todo atributo que no forme parte de ninguna clave candidata debe depender de la clave en su totalidad, y no solo de una parte. Es decir, debe tener una dependencia funcional total.

Si la clave primaria de la relación no es compuesta, la relación está automáticamente en 2FN.

Dicho informalmente, esta forma normal intenta hacer que todos los atributos de una relación dependan de **toda** la clave principal. Si sólo dependen de parte, esto quiere decir que hay un problema.

Para solucionar el problema que presenta una relación que no cumple la 2FN se toman los atributos que sólo dependen de parte de la clave y se mueven a otra nueva relación que sólo contendrá los atributos parcialmente dependientes y la parte de la clave principal de la que dependen. Esta parte será la clave principal de la nueva relación. Los campos de la clave principal permanecen en la relación original también.

RELACIÓN PEDIDO-MATERIAL (CodPed, CodMat, DesMat, CantPed, PreUnit)

Los atributos DesMat (descripción del material) y PreUnit (precio unitario) no tienen una dependencia funcional total de la clave (CodPed,CodMat), sino solo de parte de ella: CodMat (código de material).

ANOMALIAS:

EN ALTAS:	No se puede introducir la descripción y el precio de un material, mientras no tengamos un pedido que lo incluya.
EN BAJAS:	Si se borra un pedido podemos perder la información de un material si solo existía en ese pedido.
EN MODIFICACIONES:	Para cambiar el precio o la descripción de un material, deberíamos cambiarlo en todos los pedidos en los que aparezca.

SOLUCIÓN: 2 relaciones:

PEDIDO-MATERIAL (CodPed, CodMat, CantPed)
MATERIAL (CodMat, DesMat, PreUnit)

5.5.- Tercera Forma Normal

Una relación está en 3FN si, y solo si:

1.- Está en 2FN.

2.- Todo atributo que no forme parte de la clave solo depende de la clave y no de otros atributos. Es decir, todo atributo que no pertenece a una clave candidata concreta no depende transitivamente de dicha clave candidata.

Si partimos de una relación en 2FN pasaremos a 3FN identificando los atributos que dependen de otro atributo distinto de una clave candidata. Con ellos formaremos una nueva relación, eliminándolos de la primera. La nueva relación tendrá como clave principal el atributo del que dependen. Este atributo en la relación antigua pasará a ser una clave ajena.

RELACIÓN PEDIDO (CodPed, FecPed, CodProv, NomProv, DirProv)

Los atributos NomProv (nombre del proveedor) y DirProv (dirección del proveedor) dependen transitivamente de la clave, es decir no dependen funcionalmente de ella sino de CodProv.

ANOMALIAS:

EN ALTAS:	No se puede introducir el nombre y dirección de un proveedor concreto mientras no exista un pedido suyo.
EN BAJAS:	Si se borra un pedido podemos perder la información de un proveedor si éste era su último pedido.
EN MODIFICACIONES:	Para cambiar el nombre y dirección de un proveedor, deberíamos cambiarlo en todos los pedidos en los que aparezca.

SOLUCIÓN: 2 relaciones:

PEDIDO (CodPed, FecPed, CodProv)
PROVEEDOR (Codprov, NomProv, DirProv)

6.- Resumen

El proceso de normalización es un proceso fácil y mecánico que permite mejorar un modelo relacional eliminando la mayoría de las redundancias para dejar sólo las imprescindibles para relacionar la información almacenada en las distintas tablas.

Dicho esto hay que reseñar asimismo que la normalización no siempre es un proceso deseable en la práctica pues a veces puede llevar a modelos en los que algunos tipos de consultas muy comunes sean difíciles o costosas de realizar. En algunos casos se prefiere una pequeña cantidad de redundancia "innecesaria" para mejorar el funcionamiento global del sistema.