Tema 4. Transformación del Modelo Entidad Relación al modelo relacional

Contenido

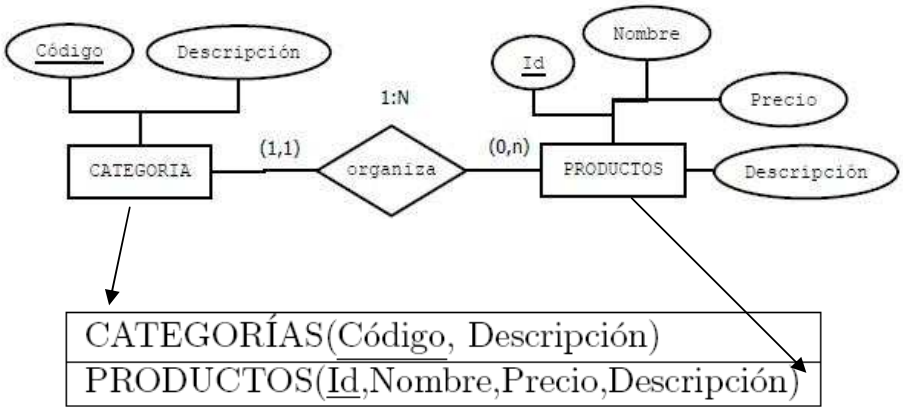
1 Transformación de las Entidades Fuertes 2
2 Transformación de Relaciones 2
2.1 Relaciones varios a varios 2
2.2 Relaciones de orden n 3
2.3 Relaciones ternarias 4
2.4 Relaciones uno a varios 7
2.5 Relaciones 1 a 1 8
2.6 Relaciones 1 a 1 con alguna cardinalidad mínima 0 8
2.7 Relaciones recursivas (otros autores las llaman cíclicas) 9
3 Transformación de Atributos de Relaciones 9
4 Transformación de Entidades Débiles 9
5 Relaciones ISA 10
6 Representación de Esquemas de Bases de Datos Relacionales 12
6.1 Grafos relacionales 12
6.2 Esquemas relacionales derivados del modelo entidad/relación 12
7 Notación que utilizaremos 14

1 Transformación de las Entidades Fuertes

En principio las entidades fuertes del modelo Entidad Relación son transformadas al modelo relacional siguiendo estas instrucciones:

- Entidades: pasan a ser tablas.
- Atributos: pasan a ser columnas o atributos de la tabla.
- Identificadores principales: pasan a ser claves primarias.
- Identificadores candidatos: pasan a ser claves candidatas.

Ejemplo:

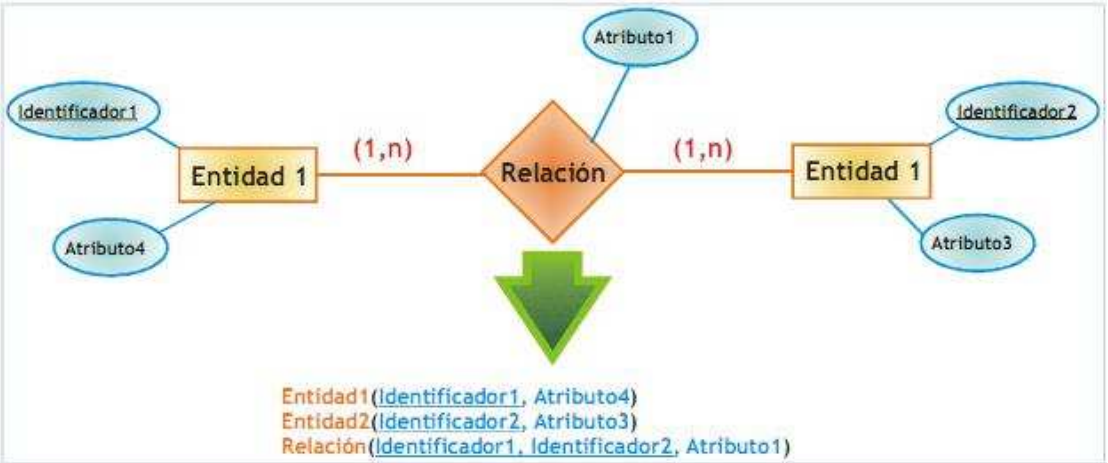


2 Transformación de Relaciones

La idea inicial es transformar cada relación del modelo conceptual en una tabla en el modelo relacional. Pero hay casos en los que esta regla tiene matices y no se cumple.

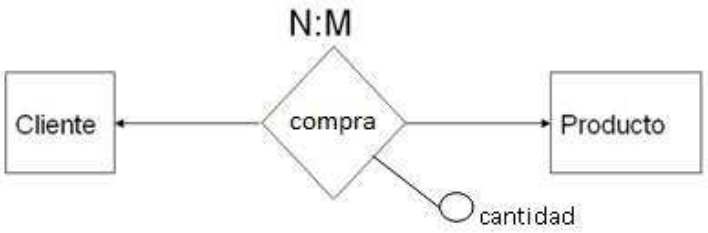
2.1 Relaciones varios a varios

En las relaciones varios a varios (n a n en la cardinalidad mayor, la cardinalidad menor no cuenta para esta situación), la relación se transforma en una tabla cuyos atributos son: los atributos de la relación y las claves de las entidades relacionadas (que pasarán a ser claves ajenas). La clave de la tabla la forman todas las claves ajenas, que son claves primarias de las tablas que forman parte de la relación.

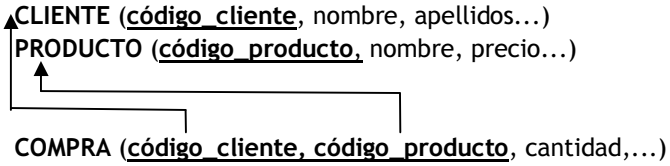


En esta figura las claves primarias se marcan con subrayado con línea continua

Ejemplo:



Algunos autores utilizan la siguiente notación: la clave primaria se subraya y la clave ajena se une con una flecha con la tabla en la que es clave primaria.



Nosotros lo representaremos de la siguiente forma:

Los atributos que forman la clave primaria llevan delante el carácter #

CLIENTE (**#código_cliente**, nombre, apellidos...)
PRODUCTO (**#código_producto**, nombre, precio...)

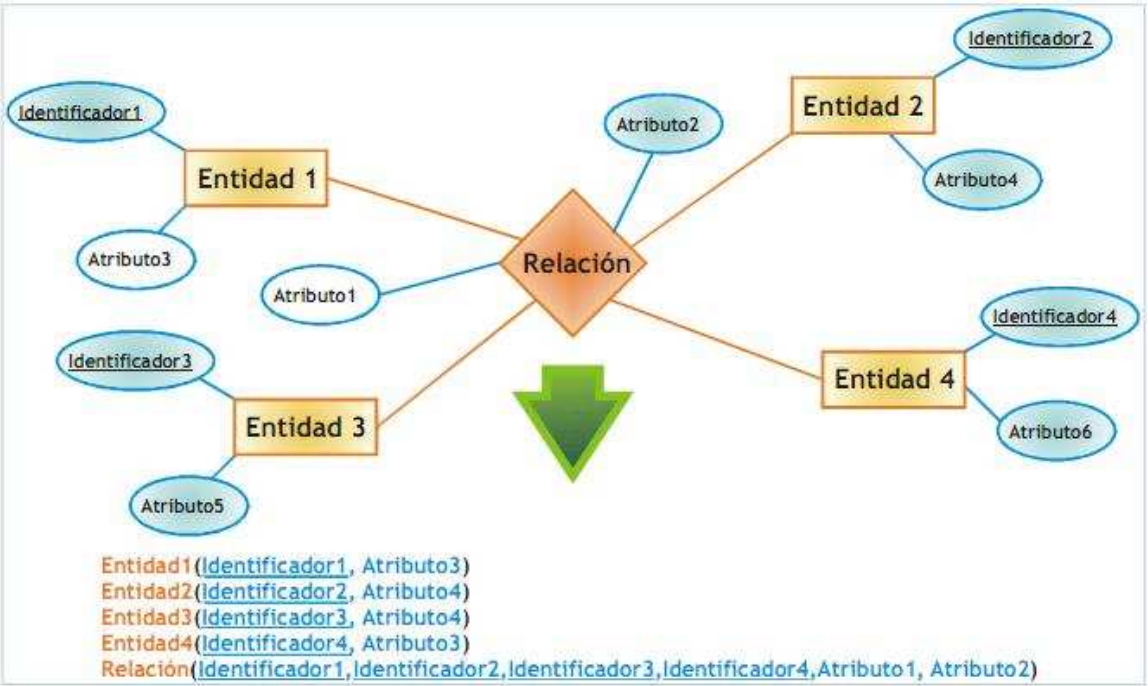
COMPRA (**#código_cliente, #código_producto**, cantidad,...)
CLIENTE **PRODUCTO**

subrayamos la clave ajena y escribimos debajo el nombre de la tabla en la que es clave primaria.

2.2 Relaciones de orden n

En el modelo Entidad/Relación hay que intentar transformar las relaciones n-arias en binarias, pero esto no siempre podremos hacerlo.

Las relaciones ternarias, cuaternarias y n-arias que unen más de dos entidades se transforman en una tabla que contiene los atributos de la relación más los identificadores de las entidades relacionadas. La clave primaria la forman todas las claves ajenas de las entidades cuya cardinalidad máxima es n, aquellas cuya cardinalidad máxima es 1 no forman parte de la clave primaria.



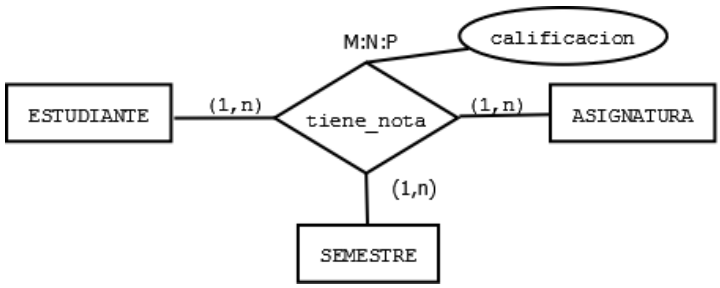
2.3 Relaciones ternarias

La transformación de una relación ternaria siempre da lugar a una nueva tabla, que tendrá como atributos las claves primarias de las tres entidades interrelacionadas y todos los atributos que tenga la relación. La clave primaria de la nueva tabla depende de la conectividad de la relación.

Conectividad M:N:P

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

Ejemplo:



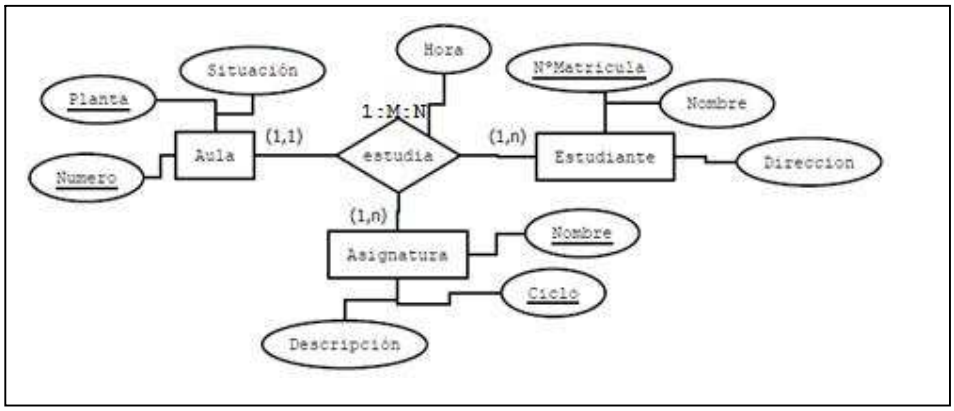
La relación anterior se transforma en:

ESTUDIANTE(#est, ...)
ASIGNATURA(#asig, ...)
SEMESTRE(#sem, ...)
TIENE-NOTA(#est, #asig, #sem, calificacion)
ESTUDIANTE ASIGNATURA SEMESTRE

Conectividad M:N:1

La tabla 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 relación etiquetados con M y con N.

Ejemplo:



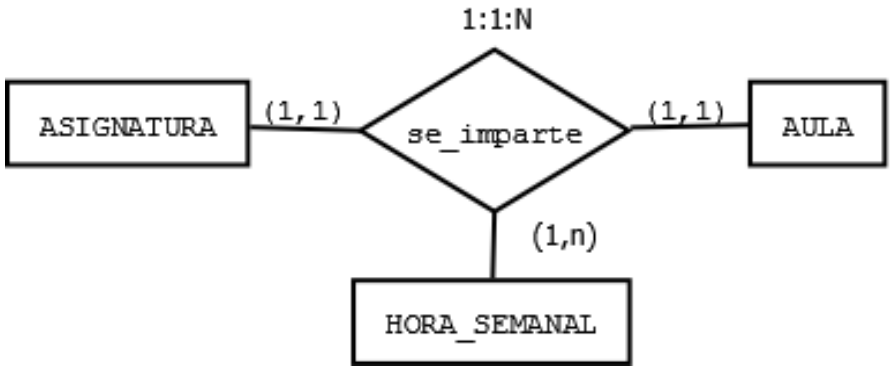
AULAS(**#Numero**, **#Planta**, Situacion)
ESTUDIANTES(**#Nºmatricula**, Nombre, Direccion)
ASIGNATURAS(**#Nombre**, **#Ciclo**, Descripcion)
ESTUDIA(Numero, Planta, **#Nºmatricula**, **#Nombre, #Ciclo**, Hora)
AULAS ESTUDIANTES ASIGNATURAS

Conectividad N:1:1

Cuando la conectividad de la relación es N:1:1, la tabla 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.

Ejemplo:

Así pues, hay dos posibles claves para la tabla que se obtiene. Son dos claves candidatas entre las cuales el diseñador deberá escoger la primaria.



Una posible transformación es la siguiente:

1) En este caso, la clave, a pesar de no incluir el atributo *asig*, identifica completamente la tabla porque para una hora-semanal y un aula determinadas hay una única asignatura de la que se hace clase a esa hora y en esa aula.

HORA-SEMANAL(#código-hora, ...)

AULA(#código-aula, ...)

ASIGNATURA(#asig, ...)

CLASE (#código-hora, #código-aula, asig, duración)
HORA_SEMANAL **AULA** **ASIGNATURA**

2) La segunda transformación posible es esta otra:

HORA-SEMANAL(#código-hora, ...)

AULA(#código-aula, ...)

ASIGNATURA(#asig, ...)

CLASE (#código-hora, código-aula, #asig, duración)
HORA_SEMANAL **AULA** **ASIGNATURA**

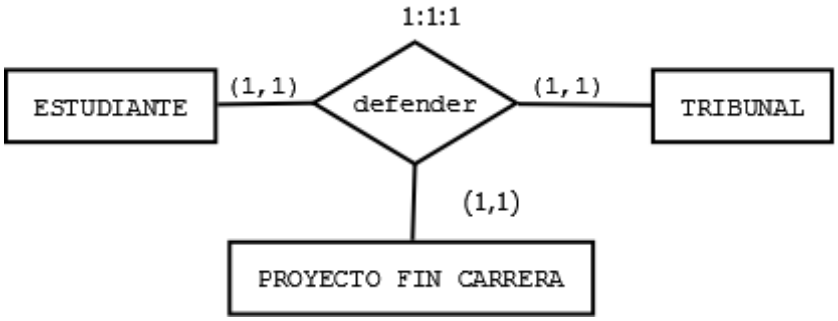
Ahora la clave incluye el atributo asig y, en cambio, no incluye el atributo código-aula. La tabla también queda completamente identificada porque, para una asignatura y hora-semanal determinadas, de aquella asignatura se da clase en una sola aula a aquella hora.

Conectividad 1:1:1

Cuando la conectividad de la relación es 1:1:1, la tabla 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.

Así pues, hay tres claves candidatas para la tabla.

Ejemplo:



TRIBUNAL(#trib, ...)

ESTUDIANTE(#est, ...)

PROYECTO-FIN-CARRERA(#pro, ...)

Para la nueva tabla DEFENDER, tenemos las tres posibilidades siguientes:

Primera opción:

DEFENDER(#trib, #est, pro, fecha-defensa)
TRIBUNAL **ESTUDIANTE** **PROYECTO**

Segunda opción:

DEFENDER(#trib, est, #pro, fecha-defensa)
TRIBUNAL **ESTUDIANTE** **PROYECTO**

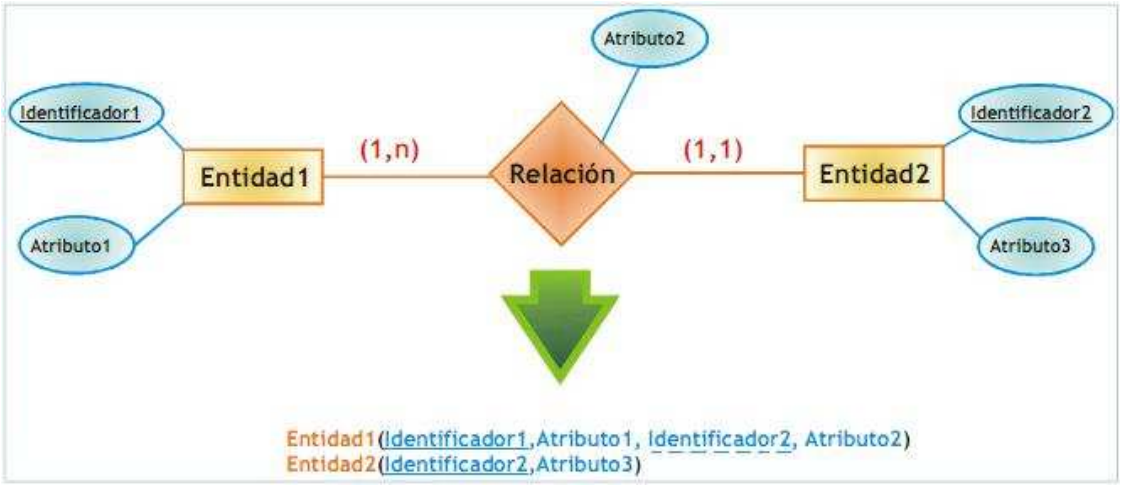
Tercera opción:

DEFENDER(trib, #est, #pro, fecha-defensa)
TRIBUNAL **ESTUDIANTE** **PROYECTO**

En los tres casos, es posible comprobar que la clave identifica completamente la tabla si se tiene en cuenta la conectividad de la relación DEFENDER.

2.4 Relaciones uno a varios

Hay dos soluciones:



En esta figura las claves ajenas se marcan con subrayado con línea discontinua

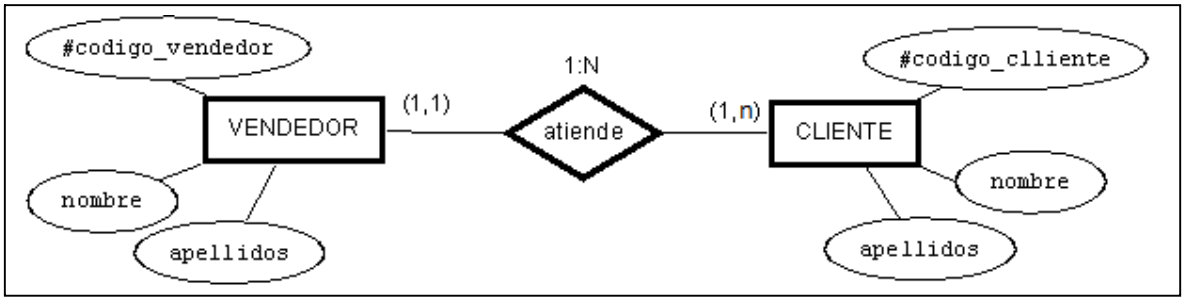
- **Propagar la clave** del tipo de entidad que tiene de cardinalidad máxima **1** a la que tiene cardinalidad máxima **N**, desapareciendo el nombre de la relación. Esta clave será ajena. Admitirá nulos si la cardinalidad del lado 1 de la relación es (0,1). Si existen atributos en la relación, éstos también se propagarán.

  Así en el dibujo, el **Identificador2** en la tabla **Entidad1** pasa a ser una clave ajena. En el caso de que la cardinalidad mínima de la relación sea de **cero** (puede haber ejemplares de la Entidad1 sin relacionar con ejemplares de la Entidad2), se deberá permitir valores nulos en la clave ajena (en el ejemplo sería el **identificador2** en la **Entidad1**). En otro caso no se podrán permitir y en el campo que sea clave ajena habrá que indicarlo explícitamente poniendo una restricción NOT NULL (ya que siempre habrá un valor relacionado).

  Si se van a permitir nulos, en la clave ajena hay que poner una restricción que permita un **borrado con puesta a NULL** (ON DELETE SET NULL).

- **Transformar la relación en una tabla.** Se hace como si se tratara de una relación **N:M**. Sin embargo en este caso, la clave primaria de la tabla creada es sólo la clave primaria de la tabla a la que le corresponde la cardinalidad N. Lo normal es el primer caso pero puede ser apropiado en los casos siguientes:
  - Cuando el número de valores interrelacionados de la entidad que propaga su clave es muy pequeño y por tanto existirían muchos valores nulos en la clave propagada.
  - Cuando se prevé que dicha relación en un futuro se convertirá en una de tipo N:M
  - Cuando la relación tiene atributos propios y no deseamos propagarlos

Ejemplo:

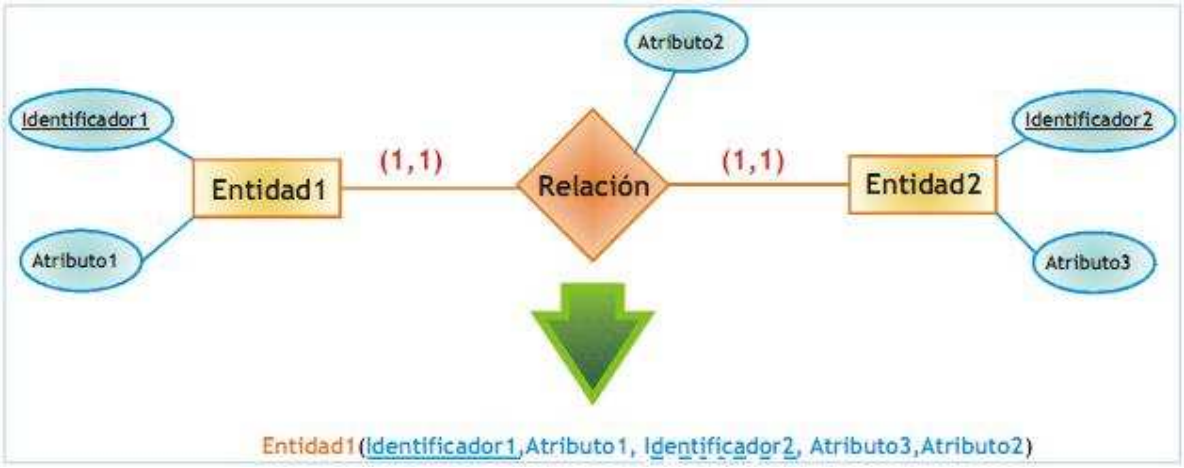


VENDEDOR (**#código_vendedor**, nombre, apellidos, ...)
CLIENTE (**#código_cliente**, nombre, apellidos, código_vendedor)
VENDEDOR

2.5 Relaciones 1 a 1

En el caso de las relaciones entre dos entidades con todas las cardinalidades a 1; hay dos posibilidades:

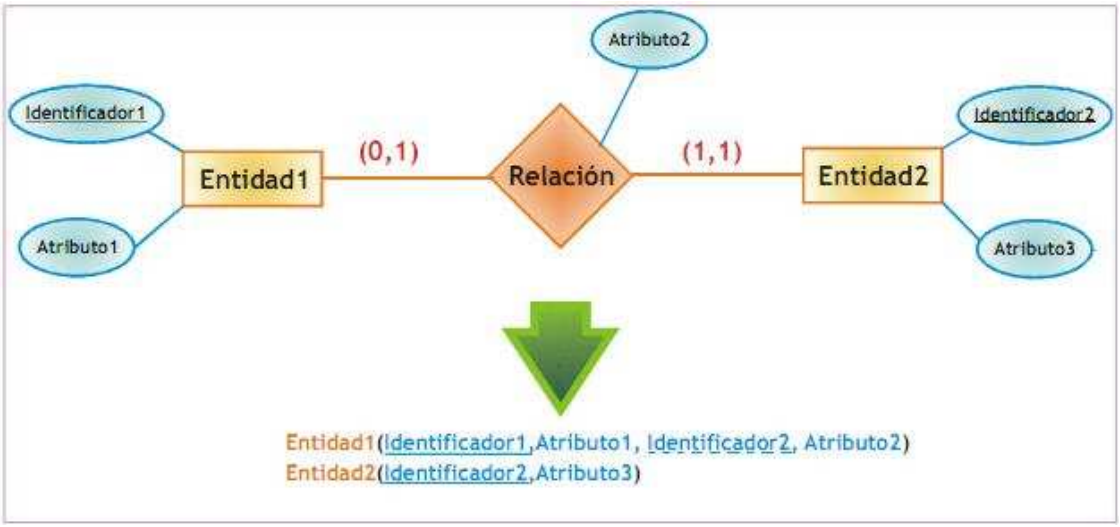
- Colocar la clave de una de las entidades como clave ajena de la otra tabla (da igual cuál), teniendo en cuenta que dicha clave será clave alternativa además de ser clave ajena.
- Generar una única tabla con todos los atributos de ambas entidades colocando como clave principal cualquiera de las claves de las dos entidades. La otra clave será marcada como clave alternativa. El nombre de la tabla sería el de la entidad más importante desde el punto de vista conceptual.



2.6 Relaciones 1 a 1 con alguna cardinalidad mínima 0

Se trata de relaciones entre dos entidades con cardinalidad máxima de 1 en ambas direcciones, pero en una de ellas la cardinalidad mínima es 0. En este caso la solución difiere respecto a la anterior solución. No conviene generar una única tabla ya que habría numerosos valores nulos en la tabla (debido a que hay ejemplares que no se relacionan en las dos tablas).

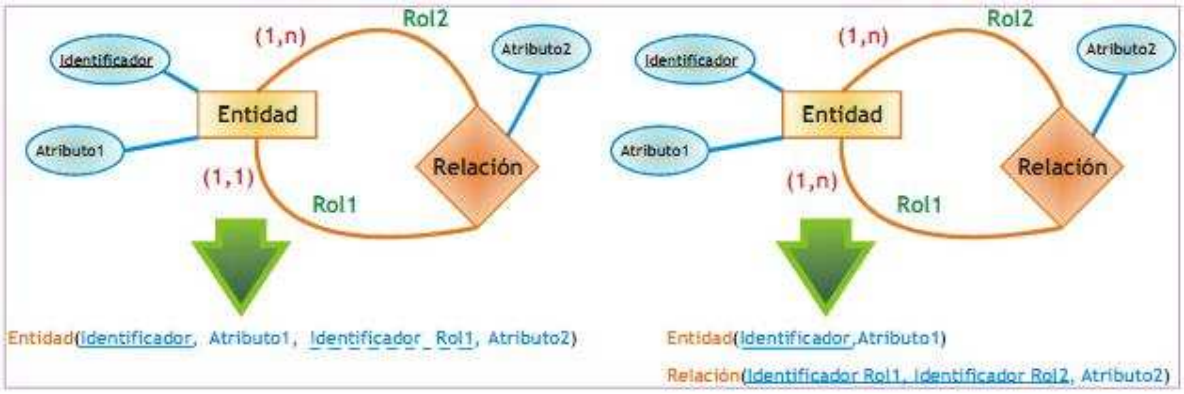
La solución sería generar dos tablas, una para cada entidad. En la tabla con cardinalidad mínima 0, se coloca como clave ajena o secundaria la clave principal de la otra (dicha clave sería clave alternativa de esa tabla).



En el caso de que en ambos extremos nos encontremos con cardinalidades (0,1), entonces la solución es la misma, pero la clave que se copia en la tabla para ser clave secundaria, debe de ser tomada de la entidad que se relacione más con la otra (la que esté más cerca de tener la cardinalidad 1 a 1 en el otro extremo). Dicha clave ajena, en este caso, no será clave alternativa (pero sí tendría restricción de unicidad). En algunas ocasiones puede ser conveniente transformar la relación 1:1 en una sola tabla.

2.7 Relaciones recursivas (otros autores las llaman cíclicas)

Las relaciones recursivas se tratan de la misma forma que las otras, sólo que un mismo atributo puede figurar dos veces en una tabla como resultado de la transformación (por eso es interesante indicar el rol en el nombre del atributo).



3 Transformación de Atributos de Relaciones

Si la relación se transforma en una tabla, todos sus atributos pasan a ser columnas de la tabla. En caso de que la relación se transforme mediante propagación de clave, sus atributos migran junto a la clave a la tabla que corresponda.

4 Transformación de Entidades Débiles

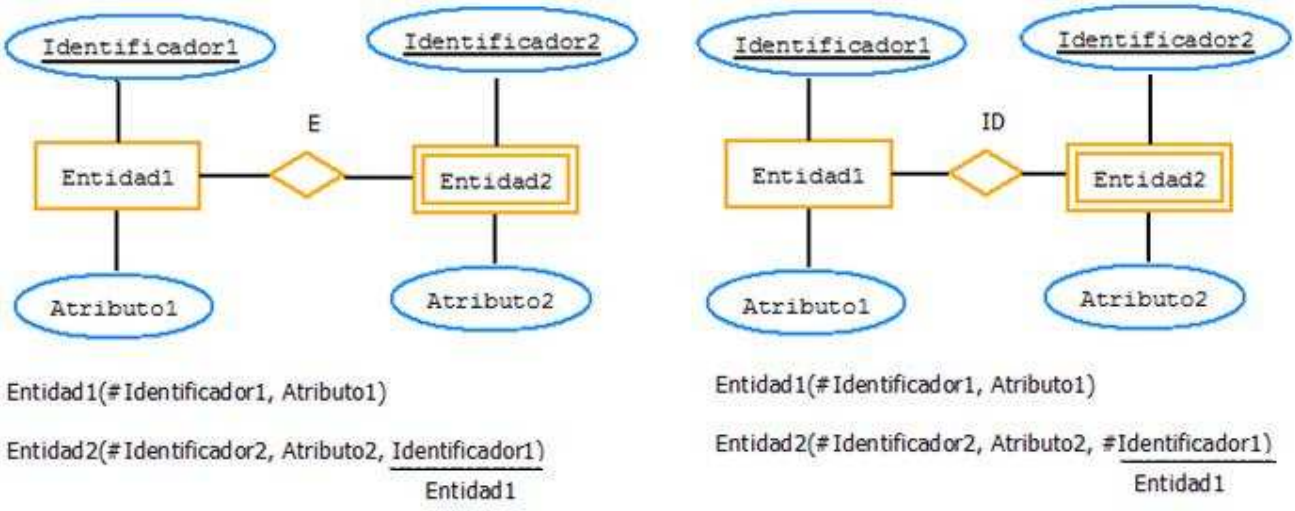
Se trata de condiciones de obligado cumplimiento por las tuplas de la base de datos. Las hay de varios tipos.

Toda entidad débil incorpora una relación implícita con una entidad fuerte. Esta relación no necesita incorporarse como tabla en el modelo relacional (al tratarse de una relación n a 1), bastará con añadir como atributo y clave foránea en la entidad débil, el identificador de la entidad fuerte.

Se propaga la clave, creando una clave ajena, con nulos no permitidos, en la tabla (tabla) de la entidad dependiente, con la característica de obligar a una modificación en cascada (ON UPDATE CASCADE,

aunque esta opción no existe en Oracle) y a un borrado en cascada (ON DELETE CASCADE) si no está restringido el borrado o la actualización (razones legales o de regla de negocio).

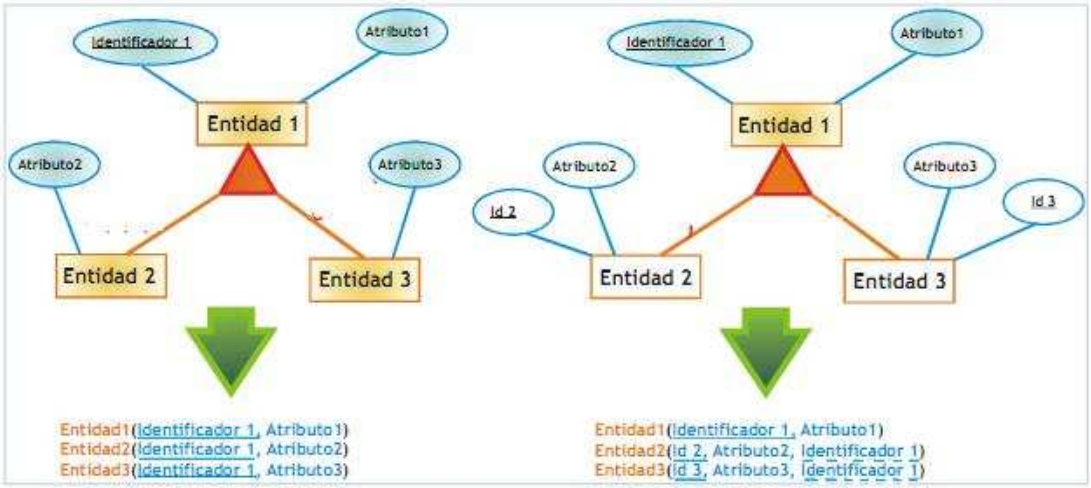
Si además hay dependencia en identificación, la clave primaria de la tabla (tabla) en la que se ha transformado la entidad débil debe estar formada por la concatenación de las claves de las dos entidades participantes en la relación.



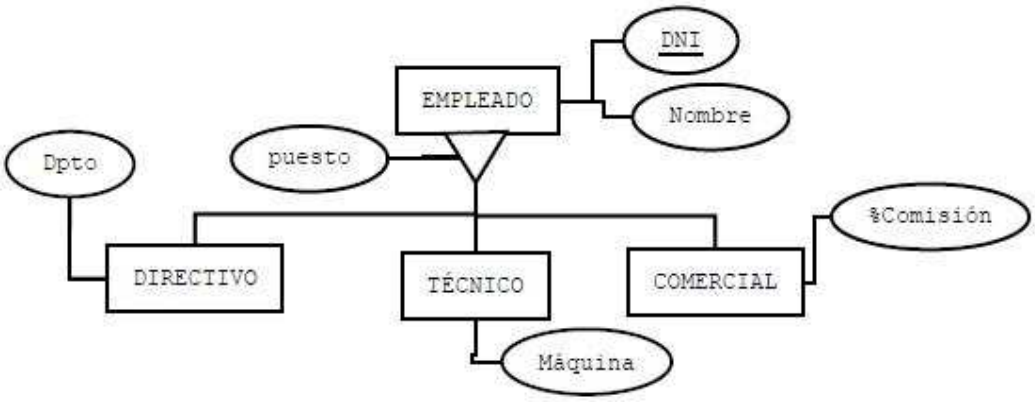
5 Relaciones ISA

En el caso de las relaciones ISA, algunos autores siguen estas normas:

- Tanto las superentidades como las subentidades generarán tablas en el modelo relacional.
- Los atributos se colocan en la tabla que se refiere a la entidad correspondiente.
- En el caso de que las subentidades no hereden el identificador de la superentidad, se colocará en las subentidades el identificador de la superentidad como clave ajena o secundaria y además será clave alternativa.
- Si la ISA es exclusiva o no, no cambia el esquema relacional, pero sí habrá que tenerlo en cuenta para las restricciones futuras en el esquema interno (casi siempre se realizan mediante triggers).



Otros autores proponen 4 opciones, cada una de ellas se adapta mejor o peor a los distintos tipos de especialización (exlusiva/inclusiva, total/parcial):



- Se puede crear una tabla para la superclase y otras tantas para cada subclase incorporando el campo clave de la superclase a las tablas de las subclases. Ésta es la solución adecuada cuando existan muchos atributos distintos entre los subtipos y se quieren mantener los atributos comunes en una tabla.

  La tabla creada para el supertipo podrá contener el atributo discriminante de la jerarquía

EMPLEADOS(DNI, Nombre)
DIRECTIVOS(DNI,Dpto)
TECNICOS(DNI,Máquinas)
COMERCIALES(DNI,Comisión)

- Se puede crear una tabla para cada subclase incorporando todos los atributos propios y los de la superclase. No crear entidad con el supertipo. Se elegiría esta opción cuando se dieran las mismas condiciones que en el caso anterior y los accesos realizados sobre los datos de los distintos subtipos siempre afectan a atributos comunes.

DIRECTIVOS(DNI,Nombre ,Dpto)
TÉCNICOS(DNI,Nombre, Máquinas)
COMERCIALES(DNI,Nombre, Comisión)

- Se puede emplear una sola tabla para la superclase, incorporando los atributos de todas las subclases y añadir, para distinguir el tipo de superclase, un campo llamado “tipo” que contendrá el tipo de subclase que representa cada tupla.

  Esta opción se adapta muy bien a las especializaciones EXCLUSIVAS

EMPLEADOS(DNI, Nombre, Dpto, Máquinas, Comisión, Tipo)

- Se puede crear una sola tabla para la superclase añadiendo varios campos que indiquen si cumple un perfil.

  De este modo se soportan las especializaciones inclusivas

EMPLEADOS(DNI, Nombre, Dpto, Máquinas, Comisión, EsDirectivo, EsTécnico, EsComercial)

6 Representación de Esquemas de Bases de Datos Relacionales

En el tema 3, ya vimos como eran los esquemas relacionales. Ejemplo:

PIEZA(**Tipo, Modelo**, Nombre, Apellido1, Apellido2)
EMPRESA(CIF, **Cod_Empresa**, Nombre, Dirección) **SUMINISTROS**(**Tipo,Modelo, Cod_Empresa**, Precio)
EXISTENCIAS(**Tipo, Modelo, N_Almacen**, Cantidad)

En ese tipo de esquemas es difícil ver las relaciones en los datos, algo que sí se ve muy bien en los esquemas entidad relación. Por ello se suelen complementar los esquemas clásicos con líneas y diagramas que representan esa información.

6.1 Grafos relacionales

Es un esquema relacional en el que hay líneas que enlazan las claves principales con las claves secundarias para representar mejor las relaciones. A veces se representa en forma de nodos de grafos y otras se complementa el clásico.

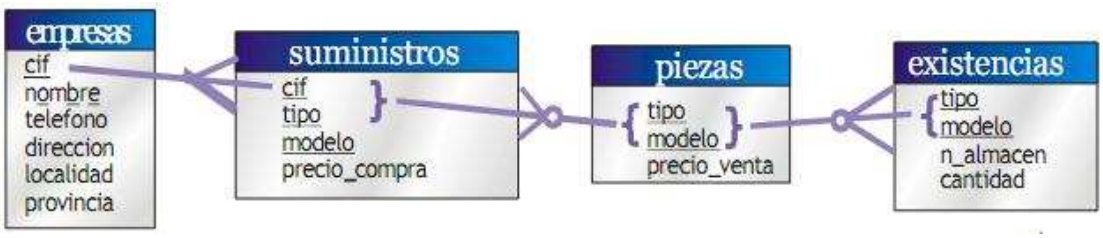
6.2 Esquemas relacionales derivados del modelo entidad/relación

Hay quien los llama esquemas entidad/relación relacionales. De hecho es una mezcla entre los esquemas relacionales y los entidad/relación. Hoy en día se utiliza mucho, en especial por las herramientas CASE de creación de diseños de bases de datos.

Las tablas se representan en forma de rectángulo que contiene una fila por cada atributo y una fila inicial para la cabecera en la que aparece el nombre de la tabla.

Después aparecen líneas que muestran la relación entre las claves y su cardinalidad.

Uno de los más utilizados actualmente es éste:



Las cardinalidades se pueden mostrar en otros formatos, pero siempre se mostrarán en este tipo de esquemas. En este caso el inicio de la línea (en la clave principal) se considera cardinalidad 1 y en el extremo podemos tener un final de línea sin símbolos (cardinalidad 1,1), acabado en varias ramas (cardinalidad 1,n) o con un círculo (cardinalidad mínima de 0).

Se ha hecho muy popular la forma de presentar esquemas relacionales del programa Microsoft Access.

Ejemplo:

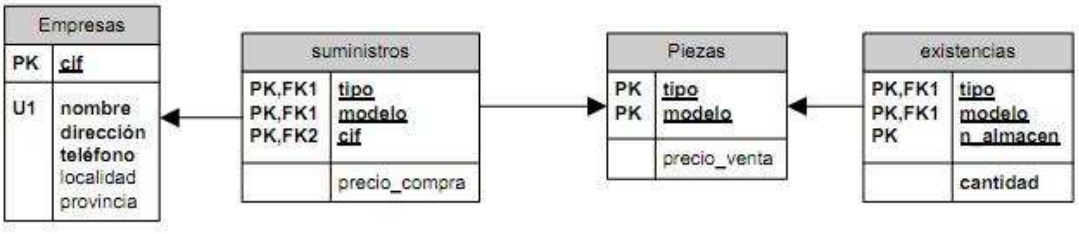
Es otra forma muy clara de representar relaciones y cardinalidades (aunque tiene problemas para representar relaciones de dos o más atributos).

Sin duda los esquemas más completos son los que reflejan no sólo las cardinalidades sino también todas las restricciones (e incluso los tipos de datos, aunque esto ya es una competencia del esquema interno). En el esquema del siguiente ejemplo los símbolos funcionan de esta forma:

Símbolo	Ejemplo	Significado
Subrayado	DNI	Clave principal
Subrayado discontinuo	Clave2	Clave alternativa
º	Nombre º	No admite valores nulos (restricción NOT NULL)
*	Nombre *	No admite duplicados (restricción UNIQUE)

Además los campos que están el final de una flecha son claves secundarias.

El programa Visio de Microsoft (y algunos otros más), representan las restricciones con letras:



En este caso los símbolos PK significan Primary Key (clave principal), FK es Foreign Key (clave secundaria, los números sirven para distinguir unas claves de otras) y UK es Unique (unicidad).

7 Notación que utilizaremos

En los ejemplos de este tema para indicar que un atributo es:

- Clave primaria se ha escrito subrayado con línea continua.
- Clave ajena se ha escrito con subrayado en línea discontinua.
- Las claves foráneas se apuntaban con una flecha a la tabla con la que se relacionaban.

Nosotros utilizaremos la notación que se indican a continuación:

- TABLA_X (AX1, AX2, …, AXm) para indicar el nombre de la tabla y los atributos
- Pondremos el símbolo # delante de los atributos que formen la **clave primaria**.
- [PK|CP] {AX1, …, AXn} para indicar cuál es la clave primaria
- Clave **ajena**: subrayada luego especificaremos si en caso de borrar el registro de la tabla padre hay que borrar, actualizar o poner a NULL los que dependen de él en la tabla hija.
- [FK|CAj] {AXo, …, AXp} Ref a TABLA_Y indica si hay claves ajenas y a que tabla apuntan.
    - AXo → AYo (campos que están relacionados en las dos tablas)
    - AXp → AYp
    - D:C U:C (borrado o actualización en cascada)
    - D: NULL (borrado con puesta a nulos)
    - D:R (borrado restringido), U:R (actualización restringida) para los registros de la tabla del campo clave del padre.
- [AK|CAlt] {AXq, …, Ar} claves alternativas.
- VNN{AXs,…, AXt} no admiten valores nulos.
- UNIQUE{AXu,…, AXz} no admiten valores repetidos.