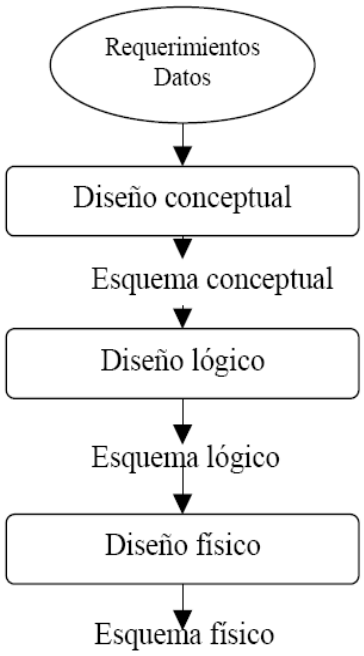
UNIDAD 3 PARTE II MODELO RELACIONAL

1 Introducción

En el proceso de diseño de una BD podemos distinguir las siguientes fases:



Diseño lógico.

Esquema lógico: descripción de la estructura de la BD en términos del modelo de datos en el que se apoya el SGDB que se va a utilizar (modelo de datos de implementación)

Modelo lógico: lenguaje usado para especificar el esquema lógico. Básicamente tres modelos se usan en diseño lógico: Relacional, Red y Jerárquico. El diseño lógico depende del modelo de datos usado, no tanto del SGDB específico elegido.

El diseño lógico consiste en transformar el esquema conceptual, que se encuentra descrito usando un cierto modelo de datos, en estructuras y transacciones descritas en términos del modelo de datos en el cuál se base el sistema de gestión de bases de datos que se vaya a utilizar.

Veremos cómo se transforma cada una de las estructuras de un diagrama Entidad-Relación en relaciones. En Unidad 4 estudiaremos la teoría de normalización, que permite refinar este primer esquema relacional obteniendo un esquema relacional adecuado desde el punto de vista de su manipulación.

Las tres reglas básicas para convertir un esquema en el modelo E/R al relacional son las siguientes:

1. Todo tipo de entidad se convierte en una relación o tabla.
2. Todo tipo de interrelación N:N se transforma en una relación o tabla.
3. Para todo tipo de relación 1:N se realiza lo que se denomina propagación de clave (regla general), o se crea una nueva relación o tabla.

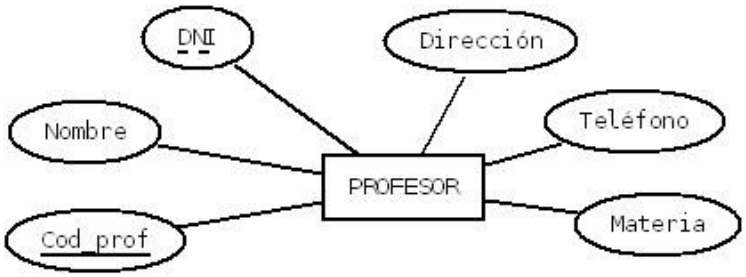
2 Transformación de entidades

2.1 Transformación de entidades fuertes y sus atributos

En principio las entidades fuertes del modelo Entidad Relación son transformadas de la siguiente manera:

- ✔ **Entidades**: Las entidades pasan a ser tablas o relaciones.
- ✔ **Atributos**: Los atributos pasan a ser columnas o atributos de la tabla.
- ✔ **Identificadores principales:** Pasan a ser las claves primarias.

Ejemplo:



La relación equivalente es la siguiente:

PROFESOR (cod_prof:dom_cod,nombre:dom_nombre, dni:dom:dni,dirección:dom_dir., teléfono:dom_telf, materia:dom_materia)

Clave Primaria {cod_prof}

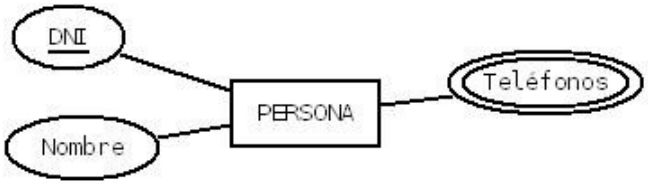
Único {DNI}

Transformación de atributos en las entidades

Los **atributos univaluados** dan lugar a un atributo de la relación.

Los **atributos multivaluados** se incluyen en la relación destacados, cada uno de ellos, entre llaves. Los problemas que puedan derivarse de la presencia de estos atributos se resolverá en la fase de normalización. Pero, vamos dan lugar a una nueva tabla cuya clave es la clave primaria de la entidad junto con el nombre del atributo.

Ejemplo:



Obtendríamos la relación:

PERSONA (dni:dom_dni, nombre:dom_nombre, {teléfono:dom_tfno}}

Clave Primaria {dni}

Como realmente quedaría

PERSONA (dni:dom_dni, nombre:dom_nombre)

Clave Primaria {dni}

TELÉFONO (dni:dom_dni, teléfono:dom_tfno)

Clave Primaria {dni, teléfono:dom_tfno}

Clave Ajena {dni} hace referencia a PERSONA

Los **atributos compuestos** se representan descomponiéndolos en sus atributos componentes.

2.2 Transformación de entidades débiles. Dependencia en existencia y en identificación

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

Toda entidad débil D (con n atributos) que depende de una entidad fuerte F (con m atributos clave) se transforma en una tabla T con n+m columnas, que se corresponden con los atributos de D y los atributos claves de F. Hay dos casos posibles:

1. Si la entidad débil D tiene una **dependencia de existencia**, Se propaga la clave, creando una clave ajena, con nulos no permitidos, en la relación de la entidad dependiente, con la característica de obligar a una modificación en cascada (ON UPDATE CASCADE) y a un borrado en cascada (ON DELETE CACADE)

2. Si la entidad débil D tiene una **dependencia de identificación**, la **clave primaria** de la tabla T estará formada por la unión de los atributos clave de la entidad débil D y de la entidad fuerte.

Ejemplo: Dependencia en identificación. Las relaciones correspondientes serían:

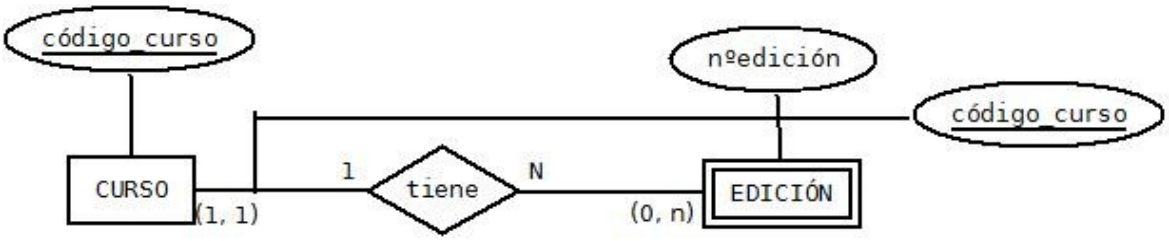
CURSO (cod_curso:dom_cod)

Clave Primaria {cod_curso}

EDICIÓN (cod_edición: dom_cod, cod_curso:dom_cod)

Clave Primaria {cod_edición, cod_curso}

Clave Ajena {cod_curso} hace referencia a CURSO



3 Transformación de relaciones

La idea inicial es transformar cada relación del modelo E/R en una tabla en el modelo relacional. Pero hay casos en los que esta regla tiene matices y no se cumple, produciéndose una **propagación de la clave** primaria.

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

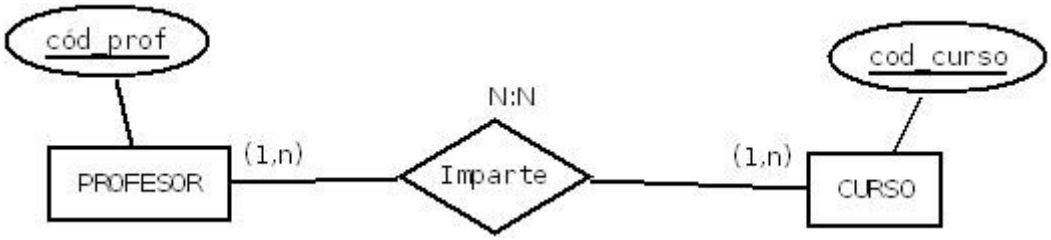
3.1 Transformación de relaciones binarias

Dependiendo del tipo de relación y de la cardinalidad que tenga, una relación binaria se podrá transformar en una nueva tabla o unirse a una tabla existente (correspondiente a una entidad) mediante la propagación de claves.

3.1.1 Transformación de relaciones con cardinalidad N:N

Se crea una nueva tabla que tendrá como clave primaria la concatenación de los dos identificadores principales de las entidades que asocia. El orden de los identificadores se elegirá según se considere más adecuado.

Además cada uno de los atributos que forman la clave primaria de esta tabla también son claves ajenas que referencian a las tablas en que se han convertido las entidades relacionadas.



Las relaciones equivalentes a esta estructura son:

PROFESOR (cod_prof:dom_cod)

Clave Primaria {cod_prof}

CURSO (cod_curso:dom_cod}

Clave Primaria {cod_curso}

IMPARTE (cod_prof:dom_cod, cod_curso:dom_cod)

Clave Primaria {cod_prof, cod_curso}

Clave Ajena {cod_prof} hace referencia a PROFESOR

Clave Ajena {cod_curso} hace referencia a CURSO

Si en la relación hubiera atributos, estos pasan a formar parte de la nueva tabla.

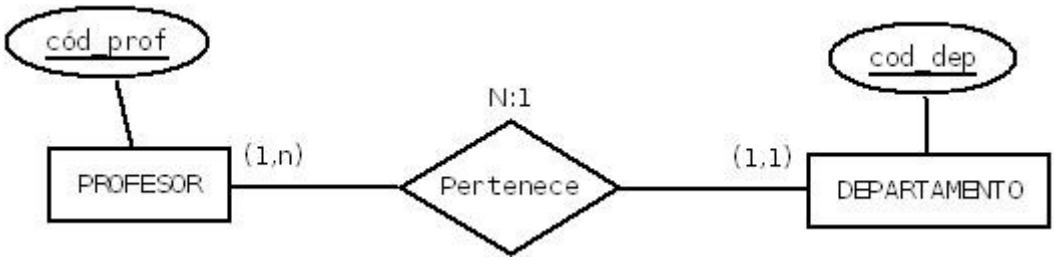
En este caso, como las dos cardinalidades máximas son n, es necesario la representación de la relación, Imparte, mediante una relación independiente.

Los casos de cardinalidad mínima 1, como en el ejemplo de arriba, es necesario escribir una expresión en SQL que represente esta restricción de existencia, que no vamos a ver en este módulo.

3.1.2 Transformación de relaciones con cardinalidad 1:N

Existen dos soluciones:

a) Propagar los identificadores principales del tipo de entidad que tiene cardinalidad máxima 1 a la que tiene N (propagación de clave). Es decir, la tabla del lado muchos incluye como clave ajena el identificador de la entidad del lado uno. Esta es la regla más habitual. Además, si en la interrelación hubiera atributos, éstos se propagan junto con la clave.



Las relaciones equivalentes a esta estructura son:

DEPARTAMENTO (cod_dep:dom_cod)

Clave Primaria {cod_dep}

PROFESOR (cod_prof:dom_cod, cod_dep:dom_cod}

Clave Primaria {cod_prof}

Clave Ajena {cod_dep} hace referencia a DEPARTAMENTO

VNN {cod_dep}

Nota: Al ser la cardinalidad mínima 1, el cod_dep en la relación PROFESOR debe tomar Valor No Nulo. La cardinalidad mínima 1 en la relación DEPARTAMENTO habría que representarla utilizando una expresión de SQL que pongo, pero que no os voy a pedir en el examen.

```
CREATE ASSERTION Rest_existencia CHECK
      NOT EXISTS (SELECT * FROM DEPARTAMENTO D
                        WHERE NOT EXISTS (SELECT * FROM PROFESOR P
                                                WHERE P.cod_dep = D.cod_dep))
```

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)

b) Transformar la interrelación en una tabla como si se tratara de una relación N:N, pero ahora la clave primaria de la tabla creada es sólo la clave primaria de la tabla del lado N. Esta solución se aplica cuando:

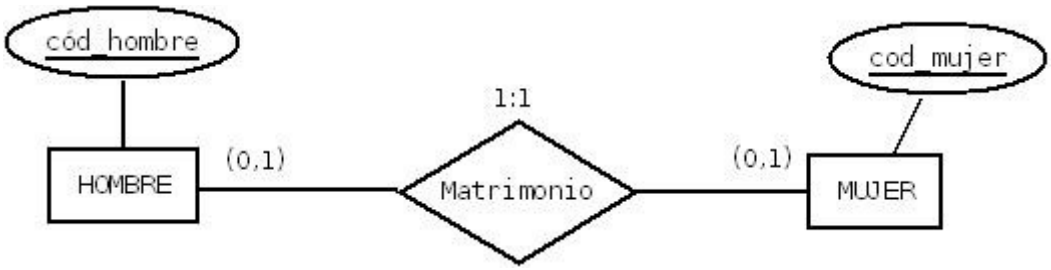
- ✔ el número de ejemplares relacionados de la entidad que propaga su clave es muy pequeño y, por tanto, existirán muchos valores nulos en la clave propagada.
- ✔ Se prevé que la interrelación en un futuro se convertirá en una de tipo N:N
- ✔ la interrelación tiene atributos propios y no es deseable propagarlos (a fin de conservar la semántica)

Nota: Nosotros usaremos como transformación el primer caso, ya que es el más adecuado.

3.1.3 Transformación de relaciones con cardinalidad 1:1

Para este tipo de relaciones hay varias posibles transformaciones:

a) Si las entidades que se asocian poseen cardinalidades (0,1), la representación de la relación R se realiza mediante la inclusión en una de las relaciones de una clave ajena que hace referencia a la otra relación y que además se define también con restricción de unicidad para representar que las cardinalidades máximas son 1.



Las relaciones equivalentes a esta estructura es:

Esquema 1:

HOMBRE (cod_hombre:dom_cod)

Clave Primaria {cod_hombre}

MUJER (cod_mujer:dom_cod, cod_hombre:dom_cod)

Clave Primaria {cod_mujer}

Único {cod_hombre}

Clave Ajena {cod_hombre} hace referencia a HOMBRE

Este esquema relacional no es el único posible, también podía haberse elegido el siguiente esquema en el que la relación Matrimonio se ha representado junto con MUJER.

Esquema 2

HOMBRE (cod_hombre:dom_cod, cod_mujer:dom_cod)

Clave Primaria {cod_hombre}

Único {cod_mujer}

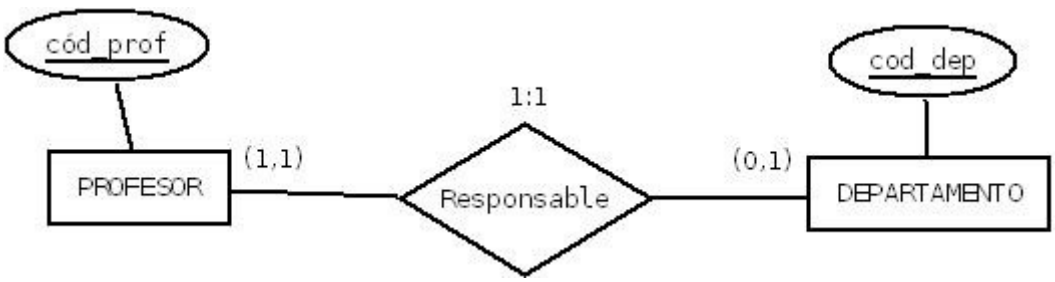
Clave Ajena {cod_mujer} hace referencia a MUJER

MUJER (cod_mujer:dom_cod)

Clave Primaria {cod_mujer}

Nota: Algunos autores representan este caso añadiendo una relación más, pero es menos adecuado ya que utiliza una relación más.

b) Si una de las entidades que participa en la interrelación posee cardinalidad (1,1) mientras que en la otra son (0,1) conviene propagar la clave de la entidad con cardinalidad (1,1) a la tabla resultante de la entidad con cardinalidad (0,1). De esta manera evitamos almacenar valores nulos, y ahorrar espacio de almacenamiento.



Las relaciones resultantes son:

PROFESOR (cod_prof:dom_cod)

Clave Primaria {cod_prof}

DEPARTAMENTO (cod_dep:dom_cod, cod_prof:dom_cod)

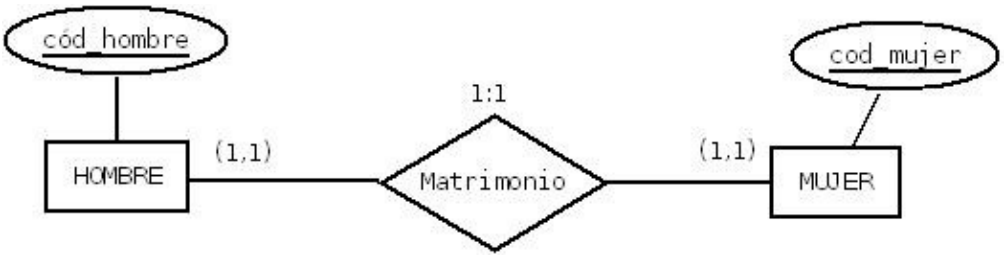
Clave Primaria {cod_dep}

Único {cod_prof}

Valor No Nulo {cod_prof}

Clave Ajena {cod_prof} hace referencia a PROFESOR

c) En el caso de tener las cardinalidades (1, 1) en ambas relaciones se puede crear una sola relación para ambas entidades, donde el IP de una de las entidades sería la clave primaria, y el IP de la otra entidad sería clave alternativa o valor único. No es lo más aconsejable.



El esquema relacional resultante sería

HOMBRE-MUJER (cod_hombre:dom_cod, cod_mujer:dom_cod)

Clave Primaria {cod_hombre}

Único {cod_mujer}

Valor No Nulo {cod_mujer}

Aunque sería preferible el siguiente esquema, colocamos 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

HOMBRE (cod_hombre:dom_cod, cod_mujer:dom_cod)

Clave Primaria {cod_hombre}

Único {cod_mujer}

Valor No Nulo {cod_mujer}

Clave Ajena {cod_mujer} hace referencia a MUJER

MUJER (cod_mujer:dom_cod)

Clave Primaria {cod_mujer}

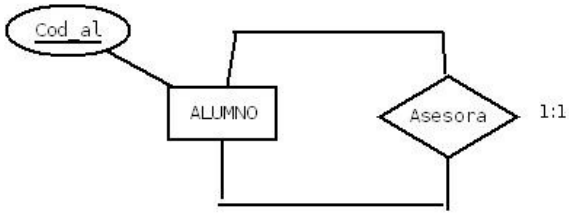
Nota: En este caso también deberíamos indicar una restricción de existencia que expresaríamos con SQL, pero que queda fuera de los contenidos de este módulo.

3.1.4 Transformación de relaciones reflexivas

Las relaciones reflexivas 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 conveniente indicar el rol en el nombre del atributo.

Veamos varios casos:

a) Relación reflexiva 1:1



La relación equivalente a esta estructura es:

ALUMNO (cod_al:dom_cod, asesor:dom_cod)

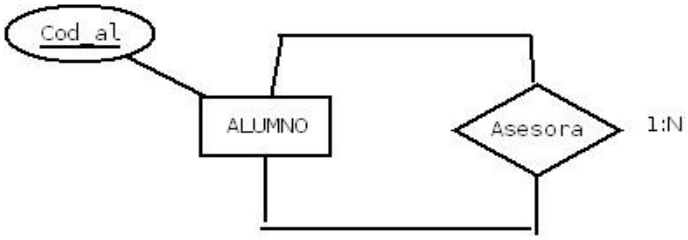
Clave Primaria {cod_al}

Único {asesor}

Clave Ajena {asesor} hace referencia a ALUMNO

Nota: Si en alguna de las entidades hubiese una cardinalidad mínima 1 se representaría con la restricción de valor no nulo, pero no entraremos en tanto detalle.

b) Relación reflexiva 1:N



La relación equivalente a esta estructura es:

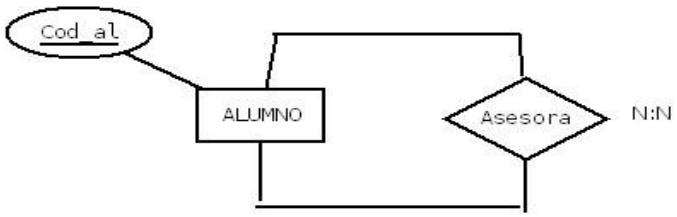
ALUMNO (cod_al:dom_cod, asesor:dom_cod)

Clave Primaria {cod_al}

Clave Ajena {asesor} hace referencia a ALUMNO

Nota: Si en alguna de las entidades hubiese una cardinalidad mínima 1 se representaría con la restricción de valor no nulo, pero no entraremos en tanto detalle.

c) Relación reflexiva N:N



Las relaciones equivalentes a esta estructura son:

ALUMNO (cod_al:dom_cod)

Clave Primaria {cod_al}

ASESORA (asesor:dom_cod, asesorado:dom_cod)

Clave Primaria {asesor, asesorado}

Clave Ajena {asesor} hace referencia a ALUMNO

Clave Ajena {asesorado} hace referencia a ALUMNO

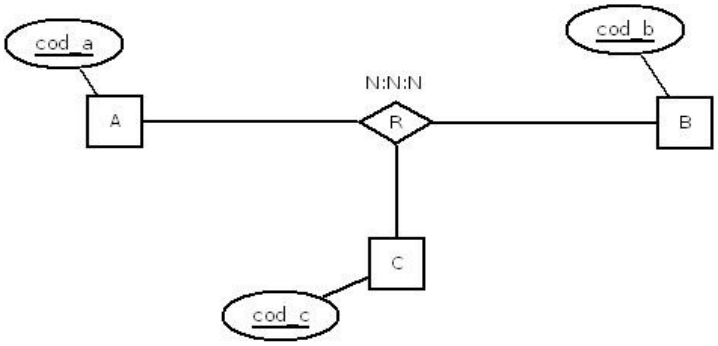
Nota: la restricción de existencia (cardinalidad mínima 1) haría necesario la inclusión de una expresión en SQL.

3.2 Transformación de relaciones ternarias

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

3.2.1 Caso N:M:P

Se representan igual que las relaciones N:N, es decir, creando una nueva tabla cuya clave primaria será la concatenación de las claves primarias de los tipos de entidades participantes.



A (cod_a:dom_cod_a)

Clave Primaria {cod_a}

B (cod_b:dom_cod_b)

Clave Primaria {cod_b}

C (cod_c:dom_cod_c)

Clave Primaria {cod_c}

R (cod_a:dom_cod_a, cod_b:dom_cod_b, cod_c:dom_cod_c)

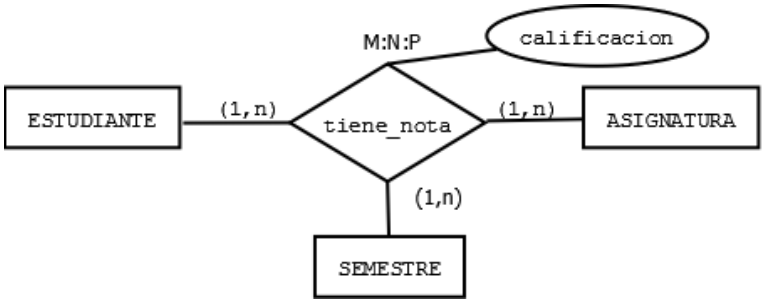
Clave Primaria {cod_a, cod_b,cod_c}

Clave Ajena { cod_a} hace referencia a A

Clave Ajena { cod_b} hace referencia a B

Clave Ajena { cod_c} hace referencia a C

Veamos otro ejemplo con relaciones más reales, la relación para ver la calificación de los estudiante en diferentes semestres de diferentes asignaturas.



La interrelación tiene_nota se transforma de la siguiente manera:

ESTUDIANTE (cod_estudiante:dom_cod)

Clave Primaria {cod_estudiante}

ASIGNATURA (cod_asignatura:dom_cod)

Clave Primaria {cod_asignatura}

SEMESTRE (cod_semestre)

Clave Primaria {cod_semestre}

TIENE-NOTA (estudiante, asignatura, semestre, calificación)

Clave Primaria { estudiante, asignatura, semestre}

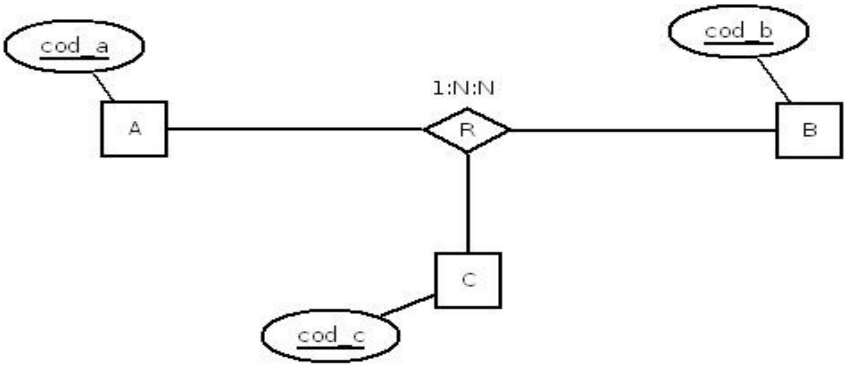
Clave Ajena {estudiante} hace referencia a ESTUDIANTE

Clave Ajena {asignatura} hace referencia a ASIGNATURA

Clave Ajena {semestre} hace referencia a SEMESTRE

3.2.2 Caso M:N:1

Igual que en la anterior, se crea una nueva tabla con las claves primarias de las entidades que están relacionadas. Ahora la clave primaria es la concatenación de las claves primarias de las tablas que corresponden a los lados N. en el ejemplo he tomado la C, como la de lado 1



R (cod_a:dom_cod_a, cod_b:dom_cod_b, cod_c:dom_cod_c)

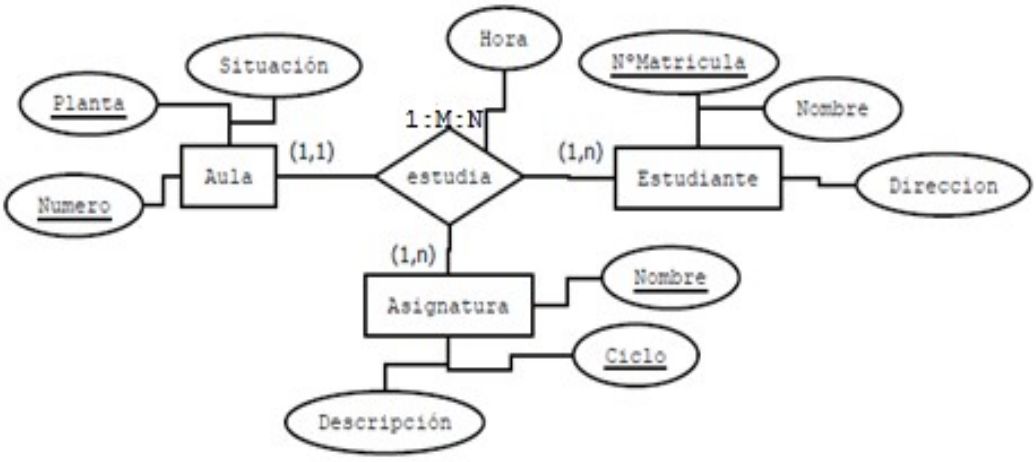
Clave Primaria {cod_a, cod_b}

Clave Ajena { cod_a} hace referencia a A

Clave Ajena { cod_b} hace referencia a B

Clave Ajena { cod_c} hace referencia a C

Veamos otro ejemplo:



En este caso la transformación quedaría así (no pongo los dominios en este caso)

AULA (Numero, Planta, Situación)

CP {Número, Planta}

ESTUDIANTE (NºMatricula, Nombre, Dirección)

CP {NºMatricula}

ASIGNATURA(Nombre, Ciclo, Descripción)

CP {Nombre, Ciclo}

ESTUDIA (Numero, Planta, NºMatricula, Nombre, Ciclo, Hora)

CP {NºMatricula, Nombre, Ciclo)

VNN {Numero, Planta}

Clave Ajena {Numero, Planta} hace referencia a AULA

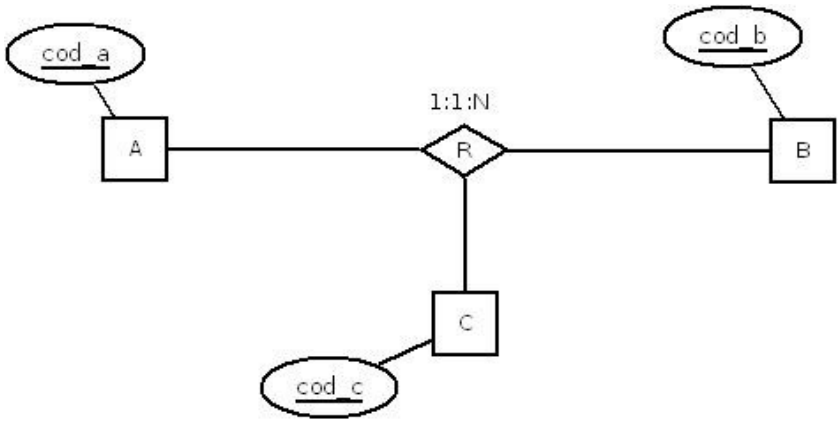
Clave Ajena {NºMatrícula} hace referencia a ESTUDIANTE

Clave Ajena {Nombre, Ciclo} hace referencia a ASIGNATURA

3.2.3 Caso 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 losa dos entidades que están conectadas con 1. Así pues, hay dos posibles claves para la relación que se obtiene. Son dos claves candidatas entre las cuales el diseñador deberá escoger la primaria.

Ejemplo:



R (cod_a:dom_cod_a, cod_b:dom_cod_b, cod_c:dom_cod_c)

Clave Primaria {cod_a, cod_b}

Único {cod_a, cod_c }

Valor No Nulo {cod_c}

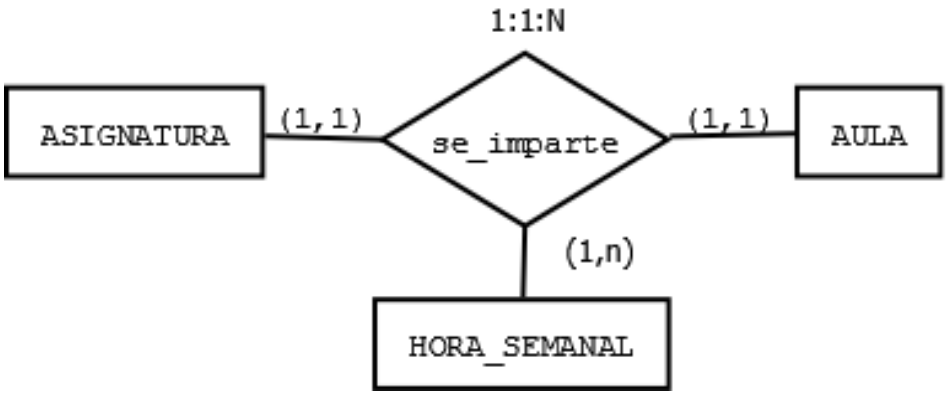
Clave Ajena { cod_a} hace referencia a A

Clave Ajena { cod_b} hace referencia a B

Clave Ajena { cod_c} hace referencia a C

Debido a que dos entidades tienen cardinalidades máxima 1, existen dos claves candidatas para la relación R.

Veamos otro ejemplo:



Primera transformación:

ASIGNATURA (asignatura)

Clave Primaria {asignatura}

AULA (código_aula)

Clave Primaria {código_aula}

HORA_SEMANAL (código_hora)

Clave Primaria {código_hora}

SE_IMPARTE (código_hora, código_aula, asignatura, duración)

Clave Primaria {código_hora, código_aula}

Único {código_hora, asignatura}

Clave Ajena {código_aula} hace referencia a AULA

Clave Ajena {código_hora} hace referencia a HORA_SEMANAL

Clave Ajena {asignatura} hace referencia a ASIGNATURA

En este caso, la clave, a pesar de no incluir el atributo asignatura, identifica completamente la relación porque para una hora_semanal y un aula determinada hay un única asignatura.

Segunda transformación:

ASIGNATURA (asignatura)

Clave Primaria {asignatura}

AULA (código_aula)

Clave Primaria {código_aula}

HORA_SEMANAL (código_hora)

Clave Primaria {código_hora}

SE_IMPARTE (código_hora, código_aula, asignatura, duración)

Clave Primaria {código_hora, asignatura}

Único {código_hora, código_aula}

Clave Ajena {código_aula} hace referencia a AULA

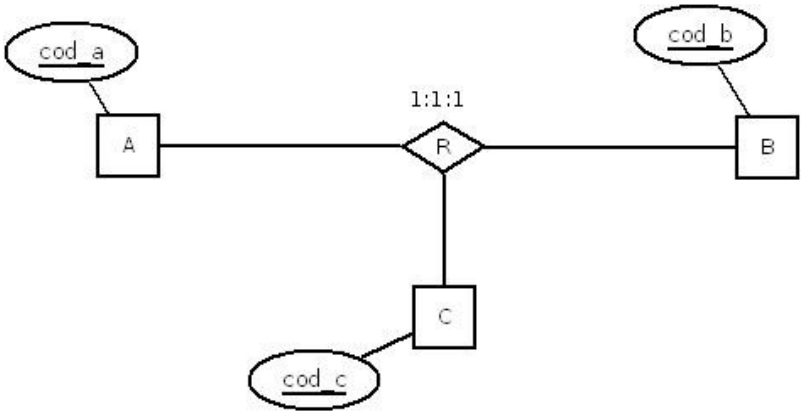
Clave Ajena {código_hora} hace referencia a HORA_SEMANAL

Clave Ajena {asignatura} hace referencia a ASIGNATURA

Ahora la clave incluye el atributo asignatura y, en cambio, no incluye el atributo código_aula. La relación también queda completamente identificada porque, para una signatura y hora-semana determinadas, de esa asignatura se da clase en una sola aula a aquella hora.

3.2.4 Caso 1:1:1

Cuando la conectividad de la interrelación es 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. Así pues, hay tres claves candidatas para la relación.



R (cod_a:dom_cod_a, cod_b:dom_cod_b, cod_c:dom_cod_c)

Clave Primaria {cod_a, cod_b}

Único {cod_a, cod_c }

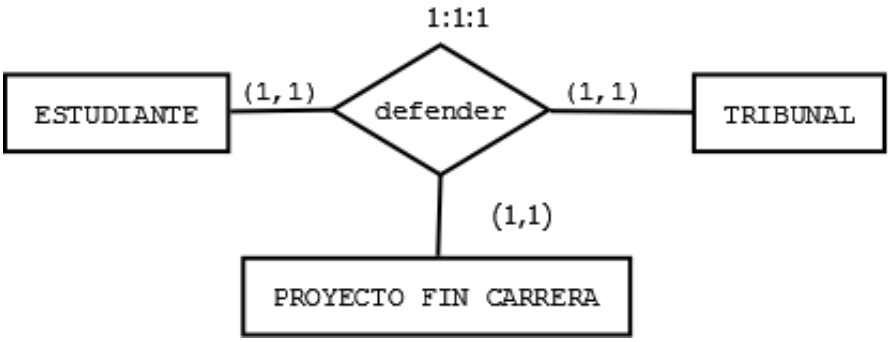
Único {cod_b, cod_c }

Clave Ajena { cod_a} hace referencia a A

Clave Ajena { cod_b} hace referencia a B

Clave Ajena { cod_c} hace referencia a C

Veamos otro ejemplo:



La transformación sería para las tres entidades:

ESTUDIANTE (cod_estudiante)

Clave Primaria {cod_estudiante}

TRIBUNAL (cod_tribunal}

Clave Primaria {cod_tribunal}

PROYECTO_FIN_CARRERA (cod_proyecto)

Clave Primaria (cod_proyecto)

Para la nueva relación DEFENDER tenemos tres posibilidades:

Primera opción:

DEFENDER (cod_estudiante, cod_tribunal, cod_proyecto, fecha_defensa)

Clave Primaria {cod_estudiante, cod_tribunal}

Único { cód_proyecto, cod_estudiante}

Único { cód_proyecto, cod_tribunal}

Clave Ajena {cod_estudiante} hace referencia a ESTUDIANTE

Clave Ajena {cod_proyecto} hacer referencia a PROYECTO

Clave Ajena {co_tribunal} hace referencia a TRIBUNAL

Segunda opción:

DEFENDER (cod_estudiante, cod_tribunal, cod_proyecto, fecha_defensa)

Clave Primaria {cod_proyecto, cod_tribunal}

Único { cód_proyecto, cod_estudiante}

Único { cód_estudiante, cod_tribunal}

Clave Ajena {cod_estudiante} hace referencia a ESTUDIANTE

Clave Ajena {cod_proyecto} hacer referencia a PROYECTO

Clave Ajena {co_tribunal} hace referencia a TRIBUNAL

Tercera opción:

DEFENDER (cod_estudiante, cod_tribunal, cod_proyecto, fecha_defensa)

Clave Primaria {cod_proyecto, cod_estudiante}

Único { cód_proyecto, cod_tribunal}

Único { cód_estudiante, cod_tribunal}

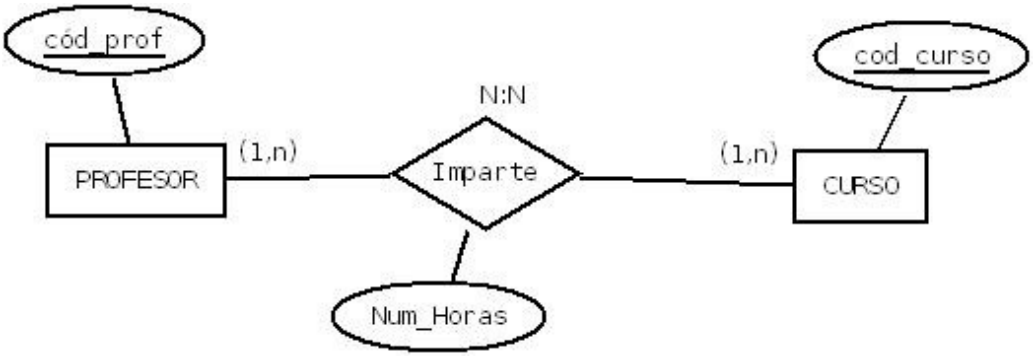
Clave Ajena {cod_estudiante} hace referencia a ESTUDIANTE

Clave Ajena {cod_proyecto} hacer referencia a PROYECTO

Clave Ajena {co_tribunal} hace referencia a TRIBUNAL

3.3 Transformación de los atributos en las interrelaciones

Si la relación se transforma en una tabla, todos sus atributos pasan a ser columnas de la tabla.



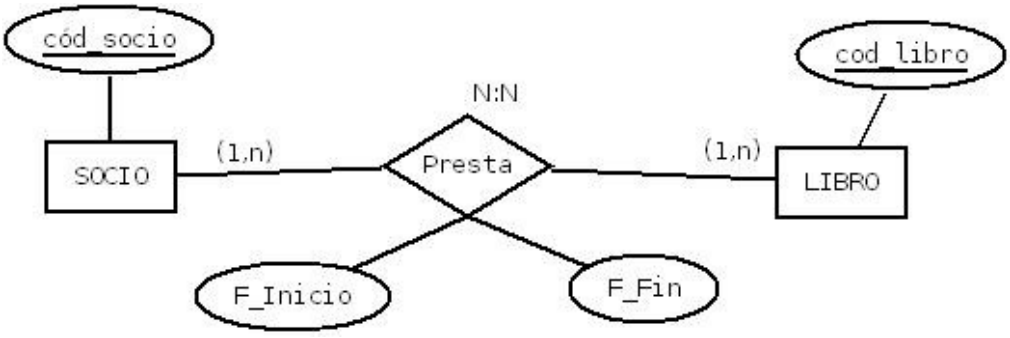
En caso de que la relación se transforme mediante propagación de clave, sus atributos migran junto a la clave a la tabla correspondiente.

3.4 Transformación 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.

3.5 Transformación de la dimensión temporal

Cuando la dimensión temporal se ha representado a través de atributos de relación de tipo FECHA, la transformación consiste en pasarlos a columnas de la tabla que corresponda, pero teniendo especial cuidado a la hora de elegir la clave primaria de la tabla resultante, dependiendo de los supuestos semánticos del entorno.



El esquema relacional resultante sería:

SOCIO (cod_socio:dom_cod)

Clave Primaria {cod_socio}

LIBRO (cod_libro:dom_cod)

Clave Primaria {cod_libro}

PRESTA (cod_socio:dom_cod, cod_libro:dom_cod, f_inicio:dom_fecha, f_fin:dom_fecha)

Clave Primaria {cod_libro, cod_socio, f_inicio}

Clave Ajena {cod_libro} hace referencia a LIBRO

Clave Ajena {cod_socio} hace referencia a SOCIO

3.6 Transformación de generalizaciones y especializaciones

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

a. Tanto las **superentidades** como las **subentidades** generarán tablas en el modelo relacional.

b. Los atributos se colocan en la tabla que se refiere a la entidad correspondiente.

c. Si las **subentidades** no tienen clave propia, se colocará como clave, la clave de su superentidad. Además esta clave heredada será clave secundaria, además de clave principal.

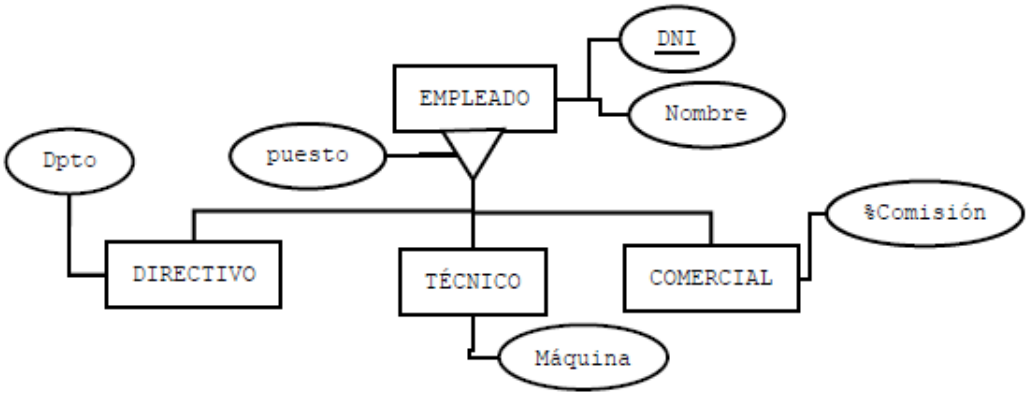
d. En el caso de que las **subentidades** tengan clave principal propia. Se colocará en las subentidades el identificador de la superentidad como clave secundaria, además será clave alternativa.

e. 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), ya que en las exclusivas no puede haber repetición de la clave de la superentidad en ninguna subentidad.

f. No varía el resultado por ser total o parcial la relación ISA, el modelo relacional no puede marcar esa posibilidad. Pero es interesante tenerlo en cuenta (se suele añadir un comentario al diseño relacional para posibles restricciones a tener en cuenta)

Otros autores proponen cuatro opciones. Cada una de ellas se adaptará mejor o peor a los distintos tipos de especialización (exlusiva/inclusiva, total/parcial).

Ejemplo:



a. Se puede crear **una tabla para la superclase o supertipo 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)

Clave Primaria (DNI)

DIRECTIVOS (DNI, dpto)
Clave Primaria (DNI)
TÉCNICOS (DNI, Máquina)
Clave Primaria (DNI)
COMERCIALES (DNI, %comisión)
Clave Primaria (DNI)

b. 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)
Clave Primaria (DNI)
TÉCNICOS (DNI, nombre, Máquina)
Clave Primaria (DNI)
COMERCIALES (DNI, nombre, %comisión)
Clave Primaria (DNI)

c. 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, Máquina, %Comisión, Tipo)
Clave Primaria (DNI)

d. 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, Máquina, %Comisión, EsDirectivo, ESTécnico, EsComercial))
Clave Primaria (DNI)

4 Representación de esquemas de bases de datos relacionales

En la Unidad 3 Parte 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.

4.1 Grafos relacionales

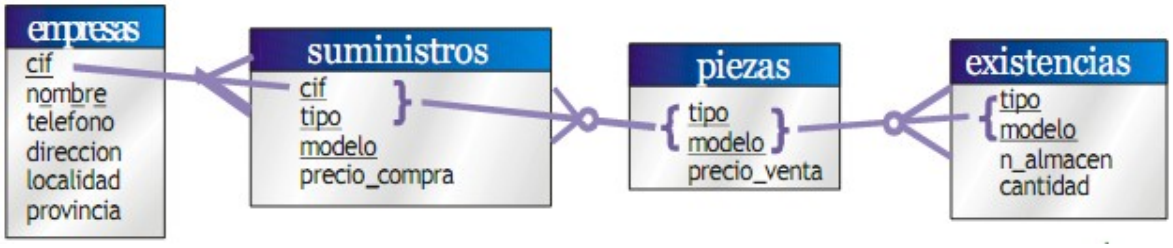
Es un esquema relacional en el que hay líneas que enlazan las claves principales con las claves secundarias para representar mejor las relaciones. A veces se representa en forma de nodos de grafos y otras se complementa el clásico.

4.2 Esquemas relacionales

Algunos autores los denominan 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.

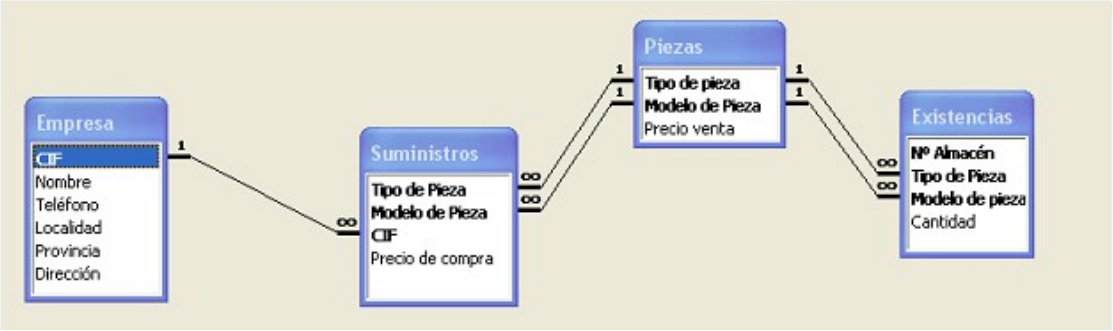
Uno de los más utilizados actualmente es éste:



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.

o:



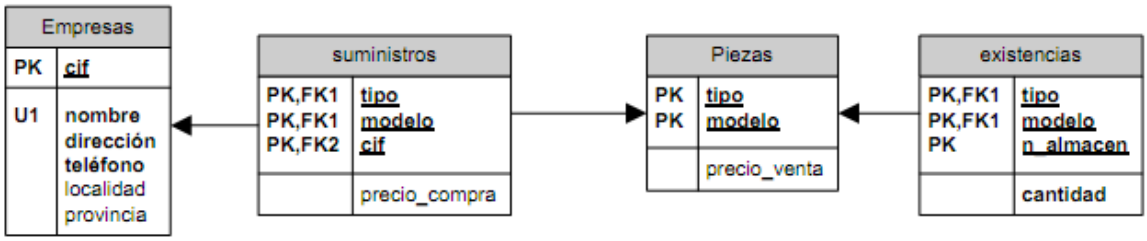
Es otra forma muy clara de representar relaciones y cardinalidades (aunque tiene problemas para representar relaciones de dos o más atributos).

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
°	Nombre °	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).

5 Notación

Existen muchas notaciones para representar las relaciones. Os muestro dos, en los ejemplos he utilizado una derivada de las segunda.

PRIMERA:

- Pondremos el símbolo # delante de los atributos que formen la **clave primaria**.
- Clave **ajena**: subrayada y escribiremos debajo el nombre de la tabla a la que se refiere y 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.
- Para indicar que un atributo es **clave alternativa** utilizaremos subrayado discontinuo.
- Si tiene valor único o no admite valores nulos utilizaremos lo mismo que en la segunda notación.

SEGUNDA:

a. TABLA_X (AX1, AX2, …, AXm) para indicar el nombre de la tabla y los atributos.
b. [PK|CP] {AX1, …, AXn} para indicar cuál es la clave primaria
c. [FK|CAj] {AXo, …, AXp} Ref a TABLA_Y indica si hay claves ajenas
   - AXo AYo
   - AXp AYp
   - D:C U:C (borrado o actualización en cascada)
   - D: NULL (borrado con puesta a nulos)
d. [AK|CAlt] {AXq, …, Ar} claves alternativas.
e. **VNN**{AXs,…, AXt} no admiten valores nulos.
f. **UNIQUE**{AXu,…, AXz} no admiten valores repetidos.