Transformación de Esquemas Entidad-Interrelación a Esquemas Relacionales

Miguel Ángel Mazoteras Sáez
Ing. Técnica en Informática de Sistemas
1996/1997
E.U. de Informática Ciudad Real
Profesor: Francisco Ruiz

1. *Presentación*

Los **Sistemas de Gestión de Bases de Datos** más populares actualmente en computadores son los **relacionales**. Estos SGBD relacionales están disponibles en una amplia gama de equipos y bajo varios sistemas operativos.

Este capítulo en su primera parte introduce brevemente el modelo relacional, en su segunda parte trata de la transformación de un esquema Entidad Interrelación en un esquema relacional, es decir se partirá de un diseño conceptual en un modelo de más alto nivel (modelo Entidad-Interrelación, que en el texto se encontrará como "ER") y se harán corresponder los esquemas conceptuales con los esquemas lógicos relacionales correspondientes.

El esquema Entidad-Interrelación no es muy diferente de un esquema relacional: no incluye generalizaciones ni subconjuntos. Se necesitan realizar otras simplificaciones como la eliminación de atributos compuestos y polivalentes, el modelado de identificadores externos como internos y la eliminación de interrelaciones. Este proceso de correspondencia se analizará en posteriores apartados, en los que se trata estos aspectos de manera independiente.

En este trabajo se hará además una transformación del esquema relacional al lenguaje de datos SQL (Structured Query Language), por ser el lenguaje estándar de los Sistemas de Gestión de Bases de Datos Relacionales. Utilizaremos la sintaxis del SQL propuesto como lenguaje lógico estándar, que es una ampliación del estándar actual, ISO 9075, según las propuestas elaboradas para el SQL2, ISO(1992) (El SQL del año 92 es también llamado SQL2). En cada regla de transformación del modelo Entidad-Interrelación al modelo Relacional se incluirá además su correspondencia en SQL2 [DEMI93, cap 23].

Por último se mostrará un caso de estudio para afianzar los conceptos teóricos obtenidos del tema. Se presentará un diseño conceptual en el modelo ER (Será la representación de la realidad), partiendo de este modelo y aplicando las correspondientes reglas de transformación obtendremos el Esquema Relacional correspondiente para posteriormente representar este esquema en SQL2.

Los requisitos para poder realizar un seguimiento del trabajo sería el estudio de una introducción a las Bases de Datos para el entendimiento de los conceptos básicos de éstas. Sería necesario además un estudio del nivel conceptual de Bases de Datos, así como el pleno entendimiento del modelo Entidad-Interrelación para tener claros conceptos que se utilizarán en el trabajo y para ver el punto de partida de éste (el punto de partida como ya se ha dicho sería el esquema Entidad-Interrelación).

2. *El modelo relacional*

2.1 *Introducción*

Este modelo de datos es un modelo simple, potente y formal para representar la realidad. También ofrece una base firme para enfocar y analizar formalmente muchos problemas relacionados con la gestión de bases de datos, como el diseño de la base de datos, la redundancia, la distribución, etc. Son fundamentales para el diseño de las bases de datos relacionales el formalismo y una base matemática.

El elemento básico del modelo es la **relación**, y un **esquema de base de datos relacional** es una colección de definiciones de relaciones. El esquema de cada relación es una agregación de **atributos**; el conjunto de todos los valores que puede adoptar un atributo en particular se denomina **dominio** de ese atributo.

Una ocurrencia ó instancia de relación, llamado también **extensión de la relación** es una tabla con filas y columnas. Las **columnas** de las relaciones corresponden a los **atributos**; las **filas**, denominadas **tuplas,** son colecciones de valores tomados de cada atributo, y desempeñan la misma función que los casos individuales de entidades en el modelo ER.. El **grado** de una relación es el número de columnas; la **cardinalidad** es el número de tuplas. En el ejemplo 1 se muestra un ejemplo con la relación ESTUDIANTE:

_____ ejemplo 1 _____

Esquema: ESTUDIANTE (NOMBRE,EDAD,SEXO)
Ocurrencia: estudiante

NOMBRE	EDAD	SEXO
Juan González	19	Varòn
Antonia Sánchez	23	Hembra
Antonio Hernández	25	Varòn
Alberto Redondo	20	Varón

Ejemplo de un esquema y ocurrencia de una relación

Transformación de Esquemas Entidad-Interrelación a Esquemas Relacionales

La definición matemática de relación se desarrolla empezando por la noción de dominios. Un dominio es una colección de valores. Dados varios atributos, $A_1, A_2, ..., A_n$, con dominios D_1, D_2, ..., D_n, una ocurrencia de relación de grado n es simplemente un subconjunto del producto cartesiano $D_1 \times D_2 \ldots D_n$. Esta definición destaca una importante propiedad de las relaciones, a saber, que son conjuntos de tuplas en el sentido matemático: una relación en ningún momento puede tener tuplas duplicadas. Sin embargo, la mayoría de los sistemas relacionales no imponen esta restricción, ya que en diversas situaciones pueden ocurrir duplicados, y puede ser útil mantenerlos. En términos estrictos, tampoco importa el orden de las tuplas en la relación.

En el modelo relacional, el concepto de clave está definido de una manera muy similar al concepto de identificador en el modelo ER; una **clave** de una relación es un conjunto de atributos de la relación que identifica de manera única cada tupla de cada extensión de esa relación. Así, la única diferencia entre nuestro uso de identificadores y claves es que en el modelo relacional sólo se acepta la identificación interna.

En general, una relación puede tener más de una clave, y cada clave se denomina **clave candidata**. Por ejemplo, la relación PERSONA puede tener dos claves candidatas: la primera un número de seguro social (NSS); la segunda una clave compuesta formada por (APELLIDO, NOMBRE). Es común designar una de las claves como **clave primaria** de la relación. En el texto aquellos atributos que componen la clave primaria aparecerán subrayados.

El modelo relacional es muy sencillo debido al hecho de que todas las relaciones se definen de manera independiente; no hay conceptos como los de jerarquía, conexión o enlace entre las relaciones en el modelo. Sin embargo, las interrelaciones del modelo ER se pueden representar en el modelo relacional por una operación explícita de combinación entre atributos de tablas diferentes. Si realizamos una combinación(join), que es la operación del álgebra relacional para hacer correspondencias entre dos tablas, la correspondencia debe hacerse entre atributos comparables, es decir atributos en el mismo dominio. Mostraremos esto en el ejemplo 2.

_____ ejemplo 2 _____

ESTUDIANTE (NOMBRE, EDAD, SEXO)

NOMBRE	EDAD	SEXO
Juan González	19	Varòn
Antonia Sánchez	23	Hembra
Antonio Hernández	25	Varón
Alberto Redondo	20	Varón
Cristina Tejero	19	Hembra

CURSO (CÓDIGO, INSTRUCTOR)

CODIGO	INSTRUCTOR
C11	Piedrabuena
C12	Franca
C13	Márquez

EXAMEN (NUMERO_CURSO, NOMBRE_ESTUDIANTE, EXAMEN)

NUMERO_CURSO	NOMBRE_ESTUDIANTE	CALIFICACION
C11	Juan González	A+
C11	Antonia Sánchez	B -
C12	Antonio Hernández	A
C14	Juan González	B+

La relación EXAMEN como correspondencia de valores explícita entre las relaciones ESTUDIANTE y CURSO.

Este ejemplo muestra tres relaciones independientes, CURSO, ESTUDIANTE y EXAMEN, para la base de datos escolar. La interrelación de uno a muchos ESTUDIANTE y EXAMEN se obtendría igualando los atributos NOMBRE de ESTUDIANTE y NOMBRE_ESTUDIANTE de EXAMEN. Obsérvese en la relación EXAMEN que por cada caso de ESTUDIANTE existen cero, uno, o más casos de EXAMEN, relacionados por el mismo nombre de estudiante (es decir, el alumno ha podido realizar cero, uno o más exámenes). La interrelación uno a muchos entre la relación EXAMEN y la relación CURSO se produciría igualando los atributos comunes CÓDIGO (de CURSO) y NÚMERO_CURSO (de la relación EXAMEN). [BATI94, cap 12.1]

2.2 *Restricciones de integridad en los esquemas relacionales de bases de datos.*

En este apartado se verán las restricciones de integridad que pueden ser especificadas en un esquema relacional. Una vez especificadas se espera que tales restricciones se cumplan para cada ocurrencia o instancia de base de datos de ese esquema. Se consideran tres tipos de restricciones como parte del modelo relacional: **restricciones de clave** que especifican las claves candidatas de cada esquema de relación; los valores de las claves candidatas deben ser únicos para cada tupla en cualquier caso de ese esquema de relación.

Las **restricciones de integridad de entidades** establecen que ningún valor de clave primaria puede ser nulo ya que el valor de la clave primaria se usa para identificar las tuplas individuales de una relación; permitir valores nulos para la clave primaria implica que no se pueda identificar algunas tuplas. Por ejemplo, si dos o más tuplas tuvieran un valor nulo como clave primaria, no podríamos distinguirlas.

Las restricciones de clave y de integridad de la entidad se especifican en relaciones individuales. La **restricción de integridad referencial**, en cambio, se especifica entre dos relaciones, y se usa para mantener la congruencia entre las tuplas de las dos relaciones. Informalmente la restricción de integridad referencial establece que una tupla de una relación que haga referencia a otra relación debe referirse a una tupla existente en esa relación. Para definir la integridad referencial de manera más formal, primero se debe definir el concepto de clave ajena. Las condiciones de una clave ajena dadas a continuación, especifican una restricción de integridad referencial entre los dos esquemas de relación R1 y R2. Un conjunto de atributos, por ejemplo CA del esquema de relación R1 es una clave ajena de R1 si satisface estas dos reglas:

1. Los atributos de CA tienen el mismo dominio que los atributos de la clave primaria, PK, de otro esquema de relación R2; se dice que los atributos de CA **se refieren** a R2.

2. Un valor de CA para una tupla t1 de R1 existe como valor de PK para alguna tupla t2 de R2, o bien es nulo. En el primer caso, tenemos t1 [CA] = t2 [PK], y decimos que la tupla t1 se refiere a la tupla t2 (t[X] indica el valor del atributo X de la tupla t. X puede ser un grupo de atributos).

En una base de datos de muchas relaciones, habrá normalmente muchas restricciones de integridad referencial que correspondan a las interrelaciones de las entidades representadas por las relaciones. Veremos ahora estas restricciones en el ejemplo 2:

NOMBRE es una clave para ESTUDIANTE, CÓDIGO es una clave para CURSO, y el par (NUMERO_CURSO, NOMBRE_ESTUDIANTE) es una clave para EXAMEN. Estas constituyen las **restricciones de clave** en el esquema de base de datos compuesto por las tres relaciones.

Para el ejemplo vemos ahora las **restricciones de integridad de entidades** : debido a las restricciones de ESTUDIANTE, CÓDIGO no puede ser nulo en ninguna tupla de CURSO, y el par (NUMERO_CURSO, NOMBRE_ESTUDIANTE) deben tener valores no nulos en cualquier tupla de EXAMEN.

Para el caso de la **restricción de integridad referencial**: La relación EXAMEN tiene la clave primaria (NUMERO_CURSO, NOMBRE_ESTUDIANTE). El atributo NOMBRE_ESTUDIANTE se refiere al estudiante que hizo el examen, por tanto, se puede designar NOMBRE_ESTUDIANTE como clave ajena de EXAMEN, refiriéndose a la relación ESTUDIANTE. Esto significa que un valor de NOMBRE_ESTUDIANTE en cualquier tupla t1 de la relación EXAMEN debe coincidir con un valor de la clave primaria de ESTUDIANTE (el atributo NOMBRE) en alguna tupla t2 de la relación ESTUDIANTE, o bien ser nulo. Sin embargo, un valor nulo de NOMBRE_ESTUDIANTE no está permitido porque viola la restricción de integridad de entidades de la relación EXAMEN. [BATI94, cap 12.1.1]

3. *Correspondencia de esquemas del modelo ER al modelo Relacional*

En este apartado se presenta una metodología para el diseño lógico que tiene como objetivo el modelo relacional. Nuestro punto de partida será un esquema conceptual ER al cual le haremos corresponder un esquema relacional. Este esquema relacional consistirá en un conjunto de definiciones de relaciones, en el cual cada relación tiene una clave primaria. Las relaciones producidas por la transformación de esquemas corresponden a entidades o bien a interrelaciones.

Antes de hacer esta transformacíón habrá que preparar el esquema Entidad Relación para que su correspondencia con el modelo relacional sea sencilla. Esta preparación consiste en:

1.- Eliminación de identificadores externos (este paso se asocia también con la eliminación de algunas interrelaciones).

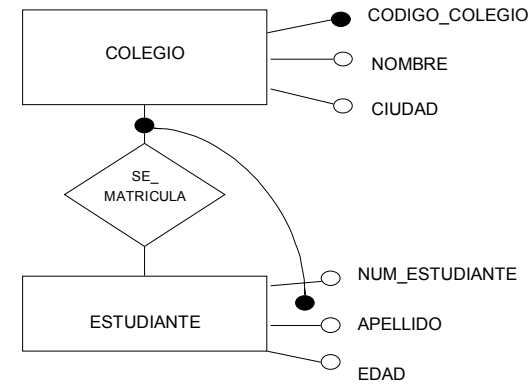
2.- Eliminación de atributos compuestos y multivaluados del esquema.

3.1 *Eliminaciones en el esquema del Modelo ER*

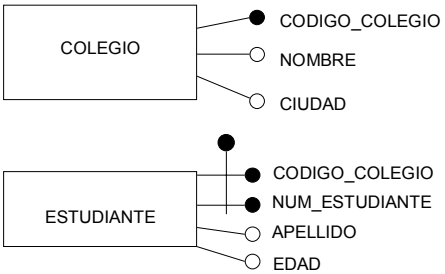
3.1.1 *Eliminación de identificadores externos*

Al no poder usar identificadores externos en el modelo relacional, se deben transformar estos en identificadores internos. Supongamos que la clave primaria de una entidad E1 es un identificador externo o mixto, y sea la entidad E2 la que suministra la identificación externa a través de la interrelación R a E1; supongamos además, que E1 tiene un identificador interno como su clave primaria. Para eliminar el identificador externo de E1, se debe importar a la entidad E1 la clave primaria de E2. Después de esta operación, puede eliminarse la interrelación R, ya que el vínculo entre E1 y E2 se modela al insertar en E1 la clave primaria de E2. Este proceso descrito no sería posible si E2 a su vez estuviera identificada externamente por alguna otra entidad E3. Este proceso debe realizarse empezando por las entidades que tienen como clave primaria un identificador interno (denominadas entidades fuertes) y luego continuando con las entidades vecinas. Los identificadores pueden propagarse según se necesite para la identificación externa. Así por ejemplo supongamos que E3 tiene un identificador interno como clave primaria. Los identificadores irán entonces de E3 a E2 y de E2 a E1. La selección de una sola clave primaria es muy útil porque obliga a concretar el identificador que deberá ser propagado a otras entidades cuando se haga la correspondencia en las relaciones. Veamos esto en el ejemplo 3:

ejemplo 3



(a) Esquema inicial



(b) Esquema final

Eliminación de un identificador externo.

Se representa la identificación mixta de la entidad ESTUDIANTE (que tiene un identificador externo suministrado por la entidad COLEGIO) al incluir CODIGO_COLEGIO en la entidad ESTUDIANTE. De esta manera, la interrelación SE_MATRICULA queda modelada automáticamente en virtud de la clave primaria (CODIGO_COLEGIO, NUM_ESTUDIANTE) de la entidad ESTUDIANTE, y se puede eliminar del esquema. El resultado se muestra en ejemplo 3. Otro aspecto a tratar sería la Integridad Referencial, es decir si la entidad ESTUDIANTE tiene como descriptor CODIGO_COLEGIO (clave primaria de la entidad COLEGIO), entonces dicho descriptor debe concordar con un valor de la clave primaria de COLEGIO, o ser nulo.
[BATI94, cap12.2.1]

3.1.2 *Eliminación de atributos compuestos*

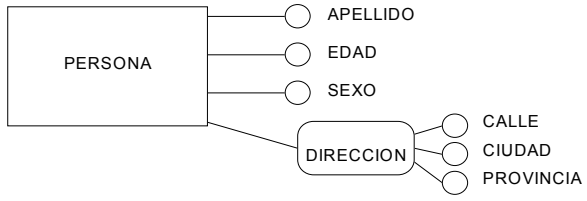
El modelo relacional en su forma básica contiene sólo atributos simples, por ello los atributos compuestos deben ser modelados en términos de atributos simples. Con cada atributo compuesto se tienen básicamente dos alternativas:

1.- Eliminar el atributo compuesto considerando todos sus componentes como atributos individuales.

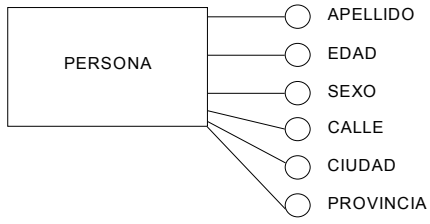
2.- Eliminar los componentes individuales y considerar el atributo compuesto entero como un solo atributo.

Estas dos alternativas las mostraremos con el ejemplo 4:

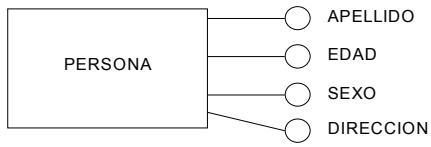
ejemplo 4



(a) Esquema con un atributo compuesto



(b) Atributo compuesto reducido a sus componentes



(c) Atributo compuesto considerado como atributo simple

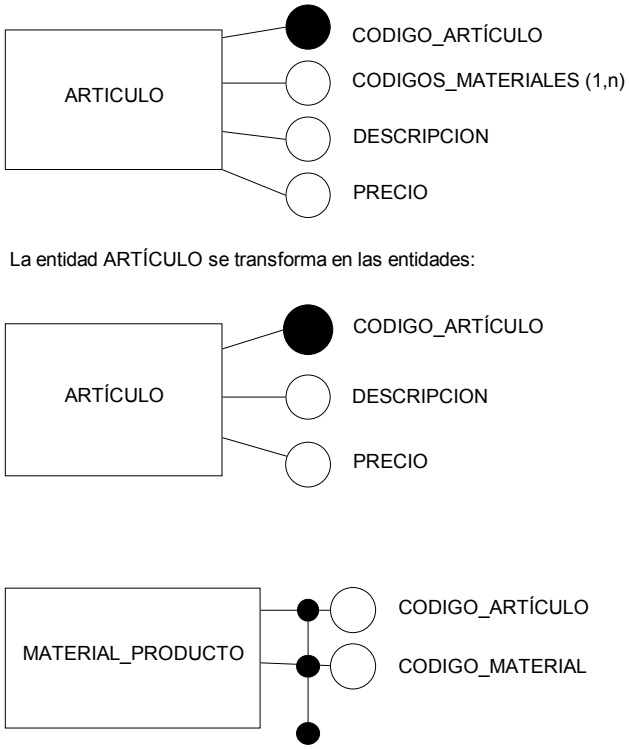
Eliminación de un atributo compuesto.

En el primer caso se pierde el concepto de que CALLE, CIUDAD y ESTADO son atributos de algún modo relacionados. En el segundo caso se no se distinguirá que DIRECCIÓN estará compuesto por más de un atributo, pero sólo se pierde saber su estuctura interna. [BATI94, cap 12.2.2]

3.1.3 *Eliminación de atributos multivaluados.*

Los atributos multivaluados requieren la introducción de entidades nuevas; cada atributo multivaluado distinto requiere una entidad en la cual pueda estar representado como atributo sencillo (también llamado monovaluado). La nueva entidad contiene el atributo multivaluado más el identificador de la entidad original; el identificador de la nueva entidad es el conjunto de todos sus atributos. Esto se ve en el ejemplo 5:

_____ ejemplo 5 _____



Eliminación de un atributo multivaluado de una entidad.

Si el atributo multivaluado pertenece a una interrelación R entre las entidades E1 y E2, se necesita crear una entidad nueva NE para representarlo. La nueva entidad NE incluye uno o dos atributos tomados de E1, E2, o ambos, dependiendo del tipo de interrelación:

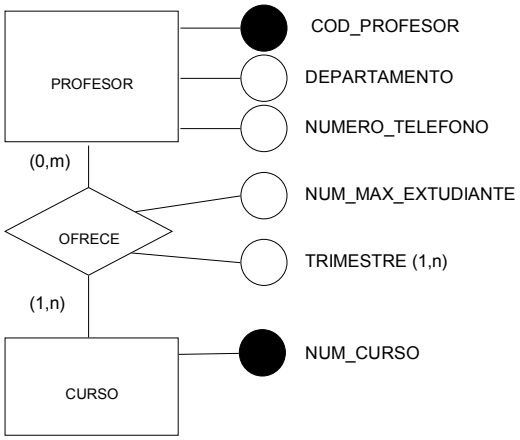
1.- Si la interrelación es de uno a uno, R(E1(1,1):E2(1,1)), entonces NE incluye la clave primaria de E1 o bien de E2.

2.- Si la interrelación entre E1 y E2 es de uno a muchos, R(E1(1,n):E2(1,1)), NE incluye la clave primaria de E1..

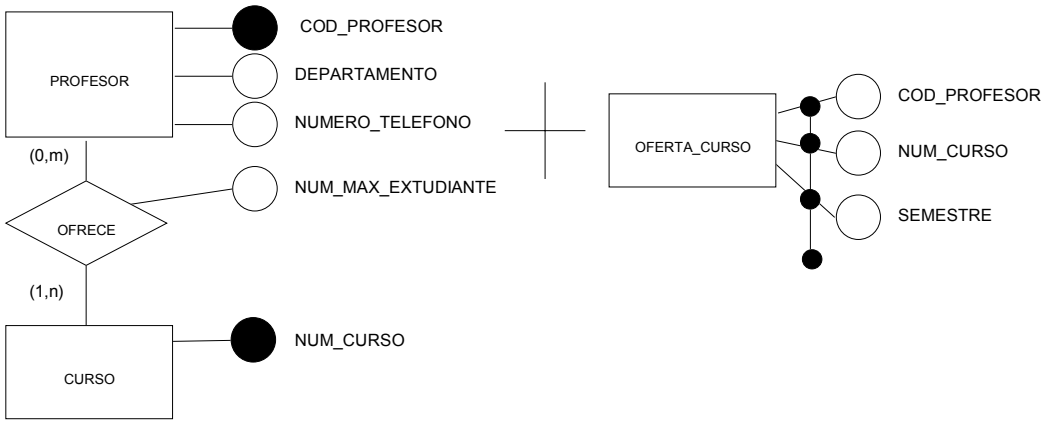
3.- Si la interrelación entre E1 y E2 es de muchos a muchos, R(E1(1,n):E2(1,n)), NE incluye las claves primarias tanto de E1 como de E2.

La clave primaria de NE está constituida por todos sus atributos: esto incluye aquellos que se tomaron de E1 y E2, y el atributo multivaluado. Cualquier atributo no multivaluado que tenga R seguirá siendo atributo de R. Esto se verá en el ejemplo 6:

ejemplo 6



Este esquema se transforma en:



Eliminación de un atributo polivalente de una interrelación creando una entidad aparte.

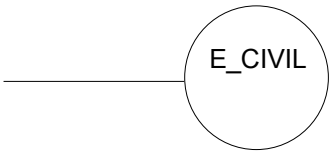
En el ejemplo cada profesor ofrece muchos cursos. La interrelación OFRECE tiene dos atributos: NUM_MAX_ESTUDIANTES, que es sencillo (ó monovalente), y TRIMESTRE, que es multivaluado (ó polivalente). Para resolver TRIMESTRE, se crea una nueva entidad, OFERTA_CURSO, con los siguientes atributos: COD_PROFESOR de PROFESOR, NUM_CURSO de CURSO, y el atributo sencillo TRIMESTRE. La clave primaria de la nueva entidad OFERTA_CURSO contiene todos sus atributos. [BATI94, cap 12.2.2]

3.2 *Transformación de dominios.*

En el modelo relacional estándar un dominio es un objeto más. Es propio de la estructura del modelo que, como tal, tendrá su definición concreta en el lenguaje de definición de datos (LDD) que se elija. En el ejemplo 7; creamos el dominio de los estados civiles, que es un conjunto de valores de tipo carácter, de longitud 1, y que puede tomar los valores ‘S’, ‘C’, ‘V’ o ‘D’:

ejemplo 7

Modelo Entidad Relación :



Modelo Relacional :

DOMINIO E_CIVIL CHAR(1) IN ('S','C','V','D')

En el SQL2 expresamos el dominio de la siguiente forma :

CREATE DOMAIN Estados_Civiles AS CHAR(1)

CHECK (VALUE IN ('S','C','V','D'))

[DEMI93, cap 23.3]

3.3 *Transformación de entidades.*

Cada tipo de **entidad** se convierte en una **relación**. Esto es, el modelo lógico estándar posee el objeto RELACIÓN mediante el cual representamos las entidades. La relación se llamará igual que el tipo de entidad de donde proviene.

Cada **atributo** de una entidad se transforma en una **atributo** de la relación a la que ha dado lugar la entidad, pero teniendo en cuenta que tenemos atributos identificadores principales, identificadores alternativos y el resto de atributos que no son identificadores (atributos no principales):

1.- **Atributos identificadores principales**. El (o los) atributo(s) identificador(es) principal(es) de cada tipo de entidad pasan a ser la clave primaria de la relación.

2**.- Atributos identificadores alternativos**. Respecto a los atributos identificadores alternativos SQL2 recoge por medio de la cláusula UNIQUE estos objetos, ya que son soportados directamente por el modelo Relacional. Al ser la transformación directa, no hay perdida semántica.

3**.- Atributos no identificadores**. Los atributos no principales pasan a ser atributos de la relación, los cuales tienen permitido tomar valores nulos a no ser que se indique lo contrario.

Para la definición de RELACIÓN disponemos en SQL2 de la sentencia CREATE TABLE. El concepto de identificador principal se recoge directamente por medio de la cláusula PRIMARY KEY en la descripción de la tabla. La transformación es directa y no hay pérdida semántica.

El concepto de identificador alternativo se expresa mediante UNIQUE (explicado antes). Para expresar que un atributo no puede tomar valores nulos se especifica mediante la cláusula NOT NULL.

Los atributos no identificadores se representarán en SQL2 como columnas normase dentro de la definición de la tabla.

Definición de una relación en SQL2 (Sintaxis general):

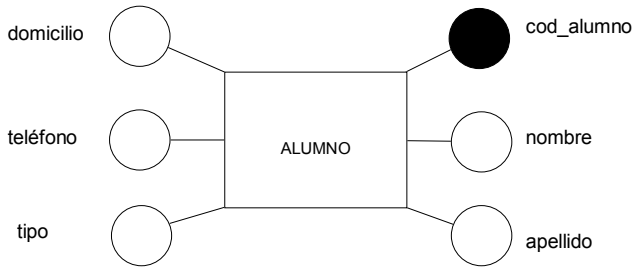
```
CREATE TABLE <nombre_tabla>
( <nombre_columna1>        <tipo_dato1>,
  <nombre_columna2 >       <tipo_dato2>,
  <nombre_columna3 >       <tipo_dato3> NOT NULL,
.....
  <nombre_columna_n>       <tipo_columna_n>,
  PRIMARY KEY (<identificador principal>),
  UNIQUE (<identificador alternativo>)   )
```

[DEMI93, cap 23.3]

Veremos esta transformación de entidades y atributos en el ejemplo 8:

_____ ejemplo 8 _____

Partimos del modelo Entidad Relación:



En el modelo Relacional quedaría:

ALUMNO (Cod_alumno , nombre , apellido , domicilio , teléfono , tipo)

(La clave primaria en el modelo relacional aparece de forma subrayada).

Una instancia de la Relación ALUMNO sería:

cod_alumno	Nombre	Apellido	Domicilio	Teléfono	Tipo
c1	María	Martínez	Ríos 14	3238060	P
c2	Pepe	Sánchez	Limón 22	3141932	P
c3	Miguel	Bermúdez	Ancha 45	3981265	I
c4	Juan	Pérez	Alta 34	5667676	A

Clave primaria

La transformación correspondiente en SQL será:

```
CREATE TABLE Alumno
( Cod_Alumno Códigos,
  Nombre Nombres,
  Apellido Nombres,
  Dirección Lugares,
  Teléfono Nos_teléfono,
   Tipo  Tipo_alumno,
  PRIMARY KEY (Cod_Alumno),
    UNIQUE (Nombre, Apellido)   )
```

3.4 *Transformación de Interrelaciones.*

Dependiendo del tipo de correspondencia de la interrelación variará la manera de realizar la transformación en el esquema relacional, por eso desglosamos esta regla en subreglas. [DEMI93, cap 23.3]

3.4.1 *Interrelaciones N:M.*

Un tipo de interrelación N:M (muchos a muchos) se transforma en una relación que tendrá como clave primaria la concatenación de los Identificadores principales (IP) de los tipos de entidad que asocia. No habrá manera de diferenciar dentro de un esquema relacional qué relaciones provienen de una entidad y cuáles de ellas proceden de la transformación de interrelaciones, por lo tanto hay cierta pérdida semántica en este punto de la transformación que sólo puede ser salvada almacenando comentarios sobre la procedencia de cada una de las tablas o mediante un convenio en la denominación de estas tablas que provienen de interrelaciones en el modelo ER.

En el caso de las interrelaciones N:M, la transformación no depende de la cardinalidad mínima de la interrelación, es decir se tratarán de igual modo los casos:
E1(0,n):E2(0,m), E1(0,n):E2(1,m) y el caso E1(1,n):(1,m).

Cada uno de los atributos que forman la clave primaria de esta relación son clave ajena respecto a cada una de las tablas donde este atributo es clave primaria, lo que se especifica en SQL2 a través de la cláusula FOREIGN KEY dentro de la sentencia de creación de la tabla. Habrá que estudiar, además, qué ocurre en los casos de borrado y modificación de la clave primaria referenciada, teniendo en cuenta que en nuestro LLS las opciones permitidas son operación restringida (en caso de no especificar la acción), puesta a nulo (SET NULL), puesta a valor por defecto (SET DEFAULT), u operación en cascada (CASCADE).

La sintaxis general en SQL sería:

```
CREATE TABLE <nombre_tabla>
( <nombre_columna1>        <tipo_dato1>,
.....
  <nombre_columna_n>       <tipo_dato_n>,
  PRIMARY KEY (<identificador principal>),
  FOREIGN KEY (<clave externa>) REFERENCES <nombre de la tabla de la cual proviene la clave externa>
```

ON < DELETE ó UPDATE> CASCADE,

Otra característica que debemos recoger en esta transformación son las cardinalidades máxima y mínima de cada una de las entidades que participan en la interrelación, lo que se hace mediante la especificación de restricciones o aserciones. Su sintaxis en SQL2 sería:

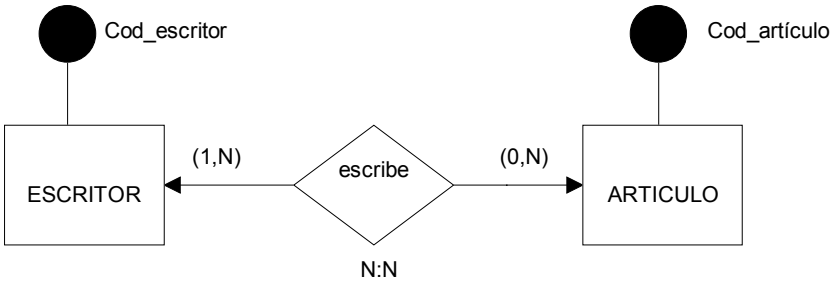
CREATE TABLE <nombre_tabla>
(<nombre_columna1> <tipo_dato1>,
.....
<nombre_columna_n> <tipo_dato_n>,
PRIMARY KEY (<identificador principal>),
FOREIGN KEY (<clave externa>) REFERENCES <nombre de la tabla de la cual proviene la clave externa>
ON < DELETE ó UPDATE> CASCADE,

CREATE ASSERTION <nombre de la restricción>
CHECK (<condición de búsqueda>)

Veremos todo esto en el ejemplo 9:

_____ ejemplo 9 _____

Partimos de la siguiente interrelación entre dos entidades:



En el modelo relacional se obtiene:

ESCRIBE (Cod_escritor, cod_artículo,.....)

ESCRITOR (Cod_escritor,.....)

ARTÍCULO (Cod_artículo,.....)

Aplicando lo antes explicado en SQL obtendríamos:

CREATE TABLE Escribe
(Cod_Libro Isbns,
Cod_escritor Códigos_A,
. . .

```
        PRIMARY KEY (Cod_artículo, Cod_escritor),
        FOREIGN KEY ( Cod_artículo) REFERENCES Artículo
                ON DELETE CASCADE
                ON UPDATE CASCADE,
        FOREIGN KEY (Cod_escritor) REFERENCES  escritor
                ON UPDATE CASCADE
```

Un ejemplo de cardinalidades máxima y mínima sería aquel en el que suponemos que cada artículo puede ser escrito por más de 1 escritor y menos de 4, que podría quedar reflejada en la siguiente aserción:

```
CREATE ASSERTION Num_Escritores_Por_Artículo
CHECK ( 1 < ( SELECT MIN (ocurrencias) FROM
        ( SELECT COUNT (*) AS Ocurrencias FROM
          ( Escritor JOIN Escribe
                ON Escritor.Cod_Escritor = Escribe.Cod_Escritor)
        GROUP BY Cod_Escritor ) )
AND ( 4 > (SELECT MAX (Ocurrencias) FROM
        ( SELECT COUNT (*) AS Ocurrencias FROM
                ( Escritor JOIN Escribe
                  ON Escritor.Cod_Escritor = Escribe.Cod_Escritor)
        GROUP BY Cod_Escritor ) ) )
```

[DEMI93, cap23.3]

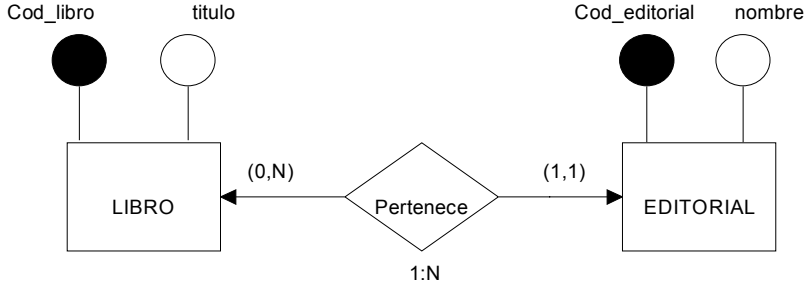
3.4.2 *Interrelaciones 1:N.*

Supongamos que tenemos dos entidades E1 y E2, relacionadas por una interrelación R de uno a muchos(1:n). Serán posibles dos casos:

1.- Caso en el que la participación sea obligatoria en el lado de "muchos", es decir, participación total; Serían los casos R (E1(x,n):E2(1,1)), siendo indiferente el valor de x.. En este caso se propaga el Identificador Principal (IP) del tipo de entidad que tiene cardinalidad máxima 1 a la que tiene n, desapareciendo el nombre de la interrelación (con lo cual hay cierta pérdida semántica). Se mostrará este caso en el ejemplo 10.

_____ ejemplo 10 _____

En el modelo ER



Su correspondencia en el modelo Relacional sería:

LIBRO (Cod_libro, titulo,...., cod_editorial)

EDITORIAL (Cod_editorial, nombre,.....)

2.- Caso en el que la participación no sea obligatoria en el lado de "muchos", es decir, participación parcial; Serían los casos R (E1(x,n):E2(0,1)), siendo indiferente el valor de x.: La interrelación se transformará en una relación como si se tratara de una interrelación N:M. Los casos en los que sería mejor transformar la interrelación en una relación serían los siguientes:

a) Cuando el número de ocurrencias interrelacionadas de la entidad que propaga su clave es muy pequeño (por el solo hecho de existir alguna ocurrencia no interrelacionada la cardinalidad mínima es cero) y cabe, por tanto, la posibilidad de que existan muchos valores nulos.

b) Cuando se prevé que dicha interrelación en un futuro se convertirá en una de tipo N:M.

c) Cuando la interrelación tiene atributos propios.

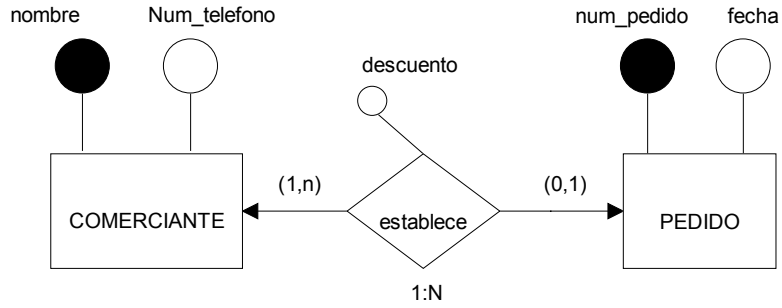
Por otro lado, la propagación de clave causa la aparición de claves ajenas, con su mecanismo de borrado y actualización correspondiente, según la semántica del problema.

Veremos este caso en el ejemplo 10:

ejemplo 10

Transformación de una interrelación 1:N en una relación:

Modelo ER:



En el modelo relacional:

COMERCIANTE (NOMBRE,NUM_TELÉFONO)

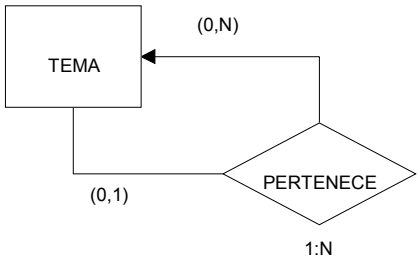
PEDIDO (NUM_PEDIDO,FECHA)

PEDIDO_VENTAS (NUM_PEDIDO,NOMBRE,DESCUENTO)

Ejemplo con interrelación reflexiva (ejemplo 11):

_____ ejemplo 11 _____

En el Modelo ER:



Modelo Relacional MR:

solución 1.-

PERTENECE (Cod_tema, cod_tema_sup)

TEMA (Cod_tema)

solución 2.-

TEMA (Cod_tema,....., Cod_tema_sup)

Si transformamos la interrelación en una solución del tipo 1, las correspondientes sentencias en *SQL2* serían**:**

```
CREATE TABLE Pertenece
(  Cod_tema Codigos,
   Cod_Tema_SUP Codigos,
   PRIMARY KEY ( Cod_tema)
   FOREIGN KEY (Cod_tema) REFERENCES  Tema
      ON DELETE CASCADE
      ON UPDATE CASCADE
   FOREING KEY ( Cod_Tema_SUP) REFERENCES Tema
      ON DELETE CASCADE
      ON UPDATE CASCADE
```

[DEMI93, cap 23.3]

3.4.3 *Interrelaciones 1 : 1*

Una interrelación de tipo 1:1 es un caso particular de una N:N o, más restrictivamente, de una 1:N, por lo que no hay una regla fija para la transformación de este tipo de interrelación en el modelo relacional estándar, pudiéndose aplicar la regla explicada para interrelaciones N:M (con lo que crearíamos una relación) o aplicar la regla de las interrelaciones 1:N (Esto es, propagar la clave correspondiente). En este último caso hay que observar que en una interrelación 1:1, la propagación de la clave puede efectuarse en ambas direcciones.

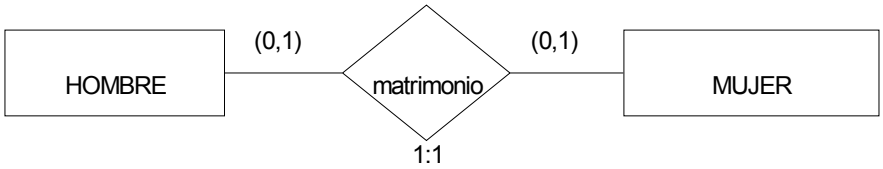
Los criterios para aplicar una u otra regla y para propagar la clave se basan en las cardinalidades mínimas, en recoger la mayor cantidad de semántica posible, evitar los valores nulos e incluso en motivos de eficiencia. Veremos esto con algunos ejemplos:

1.- Si las entidades E1 y E2 que se relacionan tienen participación parcial, es decir, de la forma R (E1(0,1):E2(0,1)), entonces la interrelación 1:1 se transformará en una relación, además de aparecer las dos relaciones que representan cada una de las entidades E1 y E2.

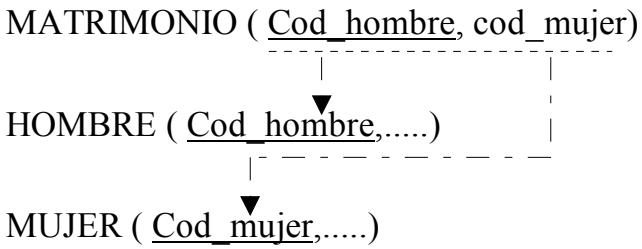
En el ejemplo 13 la interrelación MATRIMONIO entre las entidades HOMBRE y MUJER que se transforma en una relación, evitando así los valores nulos que aparecerían en caso de propagar la clave de la entidad MUJER a la tabla HOMBRE o viceversa, ya que como reflejan las cardinalidades no todos los hombres ni todas las mujeres se encuentran casados.

ejemplo13

Modelo ER:



Modelo relacional:

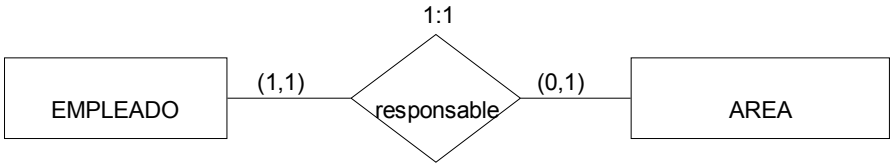


En su representación en *SQL2* se emplea lo ya explicado.

2.- Si de las dos entidades E1 y E2 que se relacionan E1 tiene participación parcial y E2 tiene participación total, es decir, de la forma R (E1(1,1):E2(0,1)), conviene entonces propagar la clave de la entidad con cardinalidades (1,1) a la relación resultante de la entidad de cardinalidades (0,1). En el siguiente ejemplo tenemos una interrelación que recoge el empleado que es responsable de un departamento, suponiendo que un empleado puede ser responsable como máximo de un departamento y que cada departamento tiene que tener un responsable (pero sólo uno), en cuyo caso propagamos la clave de EMPLEADO a la relación de ÁREA, evitando así valores nulos y captando más semántica (recogemos la cardinalidad mínima 1, que en caso de realizar la propagación en sentido contrario no podríamos captar directamente). Veamos esto en el ejemplo 14.

ejemplo 14

Modelo ER:



Modelo Relacional:

EMPLEADO (Cod_Empleado,.....)

AREA (Cod_area,....., Cod_Empleado)

clave ajena

not null

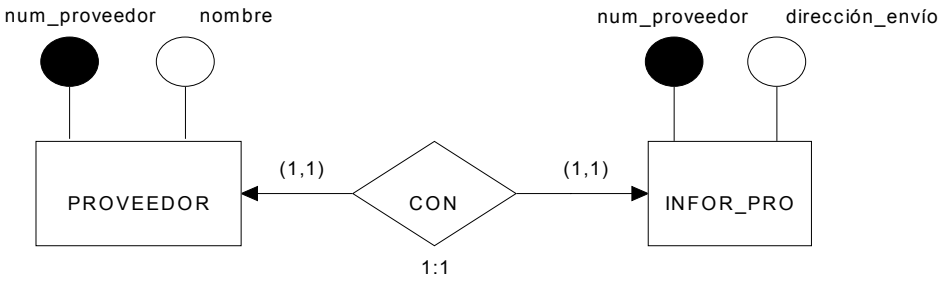
En su representación en *SQL2* se emplea lo ya explicado.

3.- Si las dos entidades E1 y E2 que se relacionan tienen participación total, es decir, de la forma R (E1(1,1):E2(1,1)), se pueden entonces plantear dos casos:

a) Caso que las dos entidades tengan las mismas claves primarias; Las dos relaciones correspondientes se integran en una relación combinando todos sus atributos e incluyendo la clave primaria sólo una vez. Este caso se ve en el ejemplo 15.

ejemplo 15

En el modelo ER:



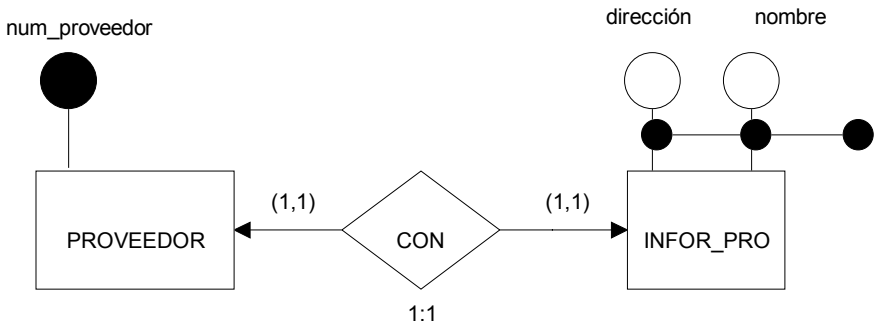
En el modelo relacional:

ENVIO_PROVEEDOR (num_proveedor,nombre,dirección_envío)

b) Si las dos entidades tienen claves primarias diferentes también se integran en una relación, combinando todos los atributos e incluyendo las claves primarias de ambas, teniendo que elegir una de las claves primarias como clave primaria de la relación resultante. Lo vemos en el ejemplo 16.

ejemplo 16

En el modelo ER:



En el modelo relacional:

ENVIO_PROVEEDOR (num_proveedor,dirección,nombre)

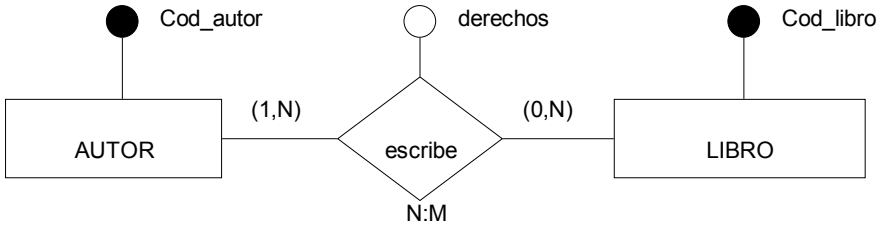
[DEMI93, cap23.3]

3.4.4 *Transformación de atributos de interrelaciones.*

Si la interrelación se transforma en una relación, todos sus atributos pasan a ser columnas de la relación. Por ejemplo, la interrelación ESCRIBE entre AUTORES y LIBROS tiene un atributo *derechos* que representa los derechos de autor que éste recibe por el libro, expresado, por ejemplo, en porcentaje; como se observa en el ejemplo 15, *derechos* pasa a ser una columna de la relación que se crea a partir de ella. La transformación es directa y no hay pérdida de semántica.

ejemplo 15

Modelo ER:



Modelo Relacional:

ESCRIBE (Cod_autor, cod_libro, derechos)

AUTOR (Cod_autor,.....)

LIBRO (Cod_libro,.....)

En caso de que la interrelación se transforme mediante propagación de clave, sus atributos migran junto a la relación que corresponda, aunque ya hemos advertido que suele ser mejor crear una nueva relación para representar la interrelación que tiene atributos.
[DEMI, cap 23.3]

3.5 *Transformación de restricciones.*

En cuanto a las restricciones, existen ciertas cláusulas en el lenguaje lógico estándar que pueden recogerlas. Un ejemplo de restricción sería la condición que deben cumplir un conjunto de atributos, expresado en SQL2 mediante la cláusula CHECK, explicada en el apartado 2.6.1. Otra restricción sería restringir a un rango determinado los valores de un dominio a través de la cláusula BETWEEN, o determinar por enumeración los valores que pueden tomar una columna en una tabla con la cláusula IN.

La sintaxis general de la cláusula BETWEEN sería:

<expresión1> [NOT] BETWEEN <expresión2> AND <expresión3>

verifica si la expresión1 tiene un valor comprendido entre los valores de expresión2 y expresión3.

La sintaxis general de la cláusula IN sería:

<expresión> [NOT] IN (<lista de valores>)

verifica si la expresión tiene un valor de los indicados en la lista de valores.

Por ejemplo, para que la fecha de inicio de un préstamo sea siempre menor que la de finalización, en la creación de la tabla escribiríamos las siguientes sentencias SQL2:

```
CREATE TABLE Préstamo
 ( Cod_Socio Códigos_Socio,
   Cod_Ejemplar Cods,
Fecha_Inicio  Fechas,
Fecha_Fin  Fechas,
PRIMARY KEY (Cod_Socio, Cod_Ejemplar, Fecha_Inicio),
FOREIGN KEY (Cod_Socio) REFERENCES Socio
     ON UPDATE CASCADE,
FOREIGN KEY (Cod_Ejemplar) REFERENCES Ejemplar
     ON UPDATE CASCADE,
CHECK ( fecha_Inicio < fecha_fin )
```

[DEMI93, cap23.3]

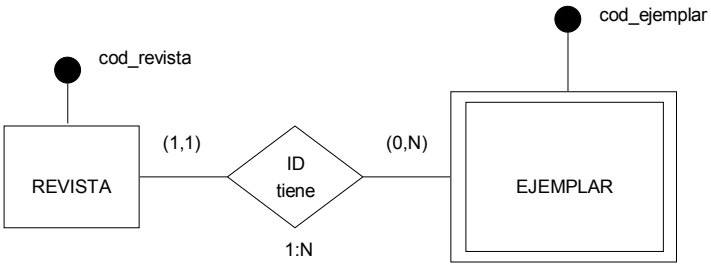
3.6 *Reglas concernientes a las extensiones del modelo Entidad Interrelación.*

3.6.1 *Transformación de dependencias en identificación y en existencia.*

La manera de transformar una interrelación de este tipo es utilizar el mecanismo de propagación de claves, creando una clave ajena, con nulos no permitidos, en la relación de la entidad dependiente, con la característica de obligar a una modificación y un borrado en cascada. Además, en el caso de dependencia en identificación la clave primaria de la relación de la entidad débil debe estar formada por la concatenación de las claves de las dos entidades participantes en la interrelación. Veremos esto en el ejemplo 16. [DEMI93, cap 23.4]

_____ ejemplo 16 _____

Modelo ER:



En el modelo relacional:

REVISTA (Cod_revista,.....)

EJEMPLAR (Cod_revista, cod_ejemplar,...)

clave ajena
not null
on delete cascade
on update cascade

En SQL2:

```
CREATE TABLE libro
        (Cod_Libro Isbns,
PRIMARY KEY (Cod_Libro) )
CREATE TABLE Ejemplar
        ( Cod_Libro Isbns,
          Cod_Ejemplar Cods,
          .....
          PRIMARY KEY ( Cod_Libro, Cod_Ejemplar)
          FOREIGN KEY ( Cod_Libro ) REFERENCES Libro
                ON DELETE CASCADE
                ON UPDATE CASCADE )
```

3.6.2 *Transformación de interrelaciones exclusivas.*

Supongamos que tenemos una entidad E1 (clase genérica), y que esta entidad E1 podrá ser o bien una entidad E2 ó una entidad E3 (E1 será una generalización de E2 y E3): podríamos realizar la transformación de diferentes formas:

1.- En esta primera forma se crea una relación a partir de la entidad E1, con atributos los de E1, y un atributo mas que nos distinga si E1 es del tipo E2 ó E3. El identificador principal de E1 pasará a ser la clave primaria de la relación.

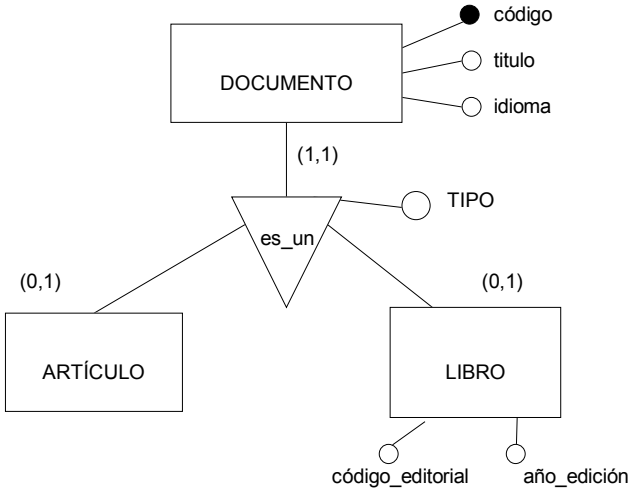
2.- En la segunda forma se crea una relación a partir de la entidad E1 con los atributos de E1 y clave primaria el identificador principal de E1. Además se crearán dos relaciones a partir de la entidades E2 y E3; Estas tendrán como clave primaria el identificador principal de E1 además de sus atributos correspondientes.

3.- Con la tercera forma no se crea una relación a partir de la entidad E1. Se crean relaciones a partir de las entidades E2 y E3; cada una de estas relaciones tendrán como clave primaria el identificador principal de la entidad E1. Tendrán además los demás atributos no identificadores de E1, y los atributos propios de E2 y E3.

Veremos estos diferentes casos en el ejemplo 17.

ejemplo 17

En el Modelo Entidad Relación:



Modelo Relacional:

a.- Resolviéndolo como en el primer caso:

DOCUMENTO (codigo,titulo,idioma,...., tipo)

b.- Resolviéndolo como en el caso segundo:

DOCUMENTO (codigo,titulo,idioma,....,)

ARTÍCULO (código,...)

LIBRO (código,año_editorial,año_edición,...)

c.- Resolviéndolo como en el tercer caso:

ARTÍCULO (código,titulo,idioma,...)

LIBRO (código,título,idioma,...)

[DEMI, cap 23.4]

3.6.3 *Transformación de tipos y subtipos.*

En lo que respecta a los tipos y subtipos, no son objetos que se puedan representar explícitamente en el modelo relacional estándar. Ante una entidad y sus subtipos se pueden dar varias soluciones en el modelo relacional, con la consiguiente pérdida de semántica dependiendo de la estrategia elegida. Destacamos tres principalmente:

1.- Englobar todos los atributos de la entidad y sus subtipos en una relación. En general, adoptaremos esta solución cuando los subtipos se diferencien en muy pocos atributos y las interrelaciones que los asocian con el resto de las entidades del esquema sean las mismas para todos los subtipos. Por ejemplo, la diferencia que existe entre un libro y un artículo podemos considerarla como mínima desde el punto de vista de ser documentos de la biblioteca (ver ejemplo 17.a). Por este motivo, la solución adecuada en este caso sería la creación de una sola tabla que contenga todos los atributos del supertipo y los de los subtipos, añadiendo un atributo adicional que indique el tipo de documento que es (el atributo discriminante de la jerarquía).

También habrá que especificar las restricciones semánticas correspondientes; si lo hacemos para el ejemplo 17:

```
CHECK ( (Tipo_documento = ' ARTICULO'
            AND  Año_Edición IS NULL
            AND Código_Editorial IS NULL )
      OR
            ( Tipo_documento = 'LIBRO'
            AND Año_Edición IS NOT NULL
            AND Código_Editorial IS NOT NULL) )
```

Hay que observar que el atributo discriminante de la jerarquía podrá admitir valores nulos en el caso de que la jerarquía sea parcial y que deberá declararse como NOT NULL si la jerarquía es total.

2.- Crear una relación para el supertipo y tantas relaciones como subtipos haya, con sus atributos correspondientes. Esta es la solución adecuada cuando existen muchos atributos distintos entre los subtipos y se quieren mantener de todas maneras los atributos comunes a todos ellos en una relación. Este caso lo vemos en el ejemplo 17.b.

3.- Considerar relaciones distintas para cada subtipo, que contengan además los atributos comunes (ver ejemplo 17.c). Se eligiría esta opción cuando se dieran las mismas condiciones que en el caso anterior (muchos atributos distintos) y los accesos realizados sobre los datos de los distintos subtipos siempre afectan a atributos comunes.

Podemos, por tanto, elegir entre tres estrategias distintas para la transformación de un tipo y sus subtipos al modelo relacional. Sin embargo, desde un punto de vista exclusivamente semántico la opción 2ª es la mejor. Por otra parte, desde el punto de vista de la eficiencia tenemos que tener en cuenta que:

Con la opción 1º el acceso a una fila que refleje toda la información de una determinada entidad es mucho más rápido (no hace falta combinar varias relaciones)

La opción 2ª es la menos eficiente, aunque, como ya hemos señalado, es la mejor desde un punto de vista exclusivamente semántico.

Con la opción tercera aumentamos la eficiencia ante determinadas consultas (por ejemplo, las que afecten a todos los atributos de un subtipo) pero la disminuimos ante otras (como las que conciernen a los atributos comunes de los distintos subtipos) e introducimos redundancias. Esta solución es en la que se pierde más semántica.

Elegiremos una estrategia u otra dependiendo de que sea la semántica o la eficiencia la que sea mas importante para el usuario en un momento determinado.

Por lo que se refiere a la totalidad/parcialidad de la jerarquía y al solapamiento/disyunción de los subtipos, pueden ser soportados por medio de restricciones en el propio esquema y por procedimientos que recojan la semántica asociada a la dinámica de estos casos. [DEMI93, cap 23.4]

3.6.4 *Transformación de la dimensión temporal.*

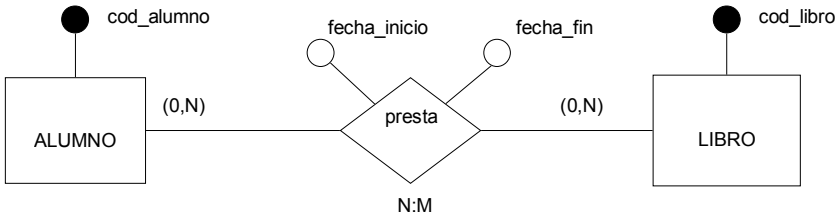
En el caso de que en el esquema E/R aparezca el tiempo como una entidad más, la transformación en el esquema relacional estándar no constituye mayor problema, ya que se tratará como otra entidad cualquiera, y por lo tanto se creará una relación más del tipo:

TIEMPO (FECHA_INICIO,HORA_INICIO,HORA_FIN,MINUTOS_I,...)

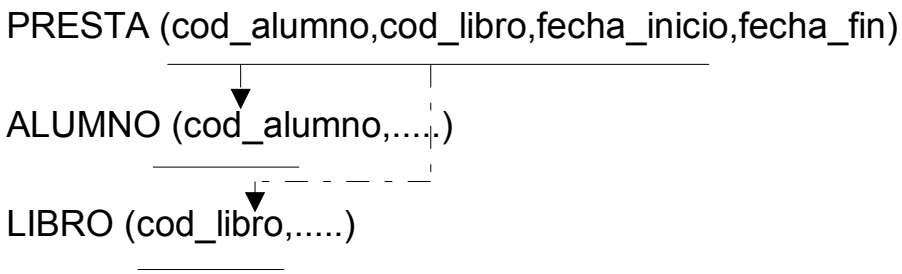
Sin embargo, cuando la dimensión temporal la hemos recogido en el esquema E/R a través de atributos de interrelación de tipo FECHA, la transformación en el Modelo Lógico Estándar consiste en pasarlos a atributos de la relación que corresponda. Sobre este punto debemos tener cuidado a la hora de elegir la clave primaria de la relación resultante, dependiendo de los supuestos semánticos del entorno. Veámoslo en el ejemplo 18.

_____ ejemplo 18 _____

Modelo ER:



Modelo Relacional:



Por ejemplo en esta figura podemos comprobar que la clave primaria de la relación obtenida de la interrelación " un socio toma prestado un libro de la biblioteca durante un periodo de tiempo determinado " no es sólo la concatenación del atributo identificador principal de ALUMNO y el de LIBRO, sino que si se supone que un alumno puede tomar prestado un mismo libro en distintos periodos de tiempo, es necesario, a fin de formar la clave primaria, añadir a los códigos de ALUMNO y de LIBRO el atributo fecha_inicio. La transformación, por lo tanto, es directa, pero debemos tener en cuenta lo que acabamos de señalar respecto a la clave primaria a fin de salvaguardar la integridad de la base de datos.

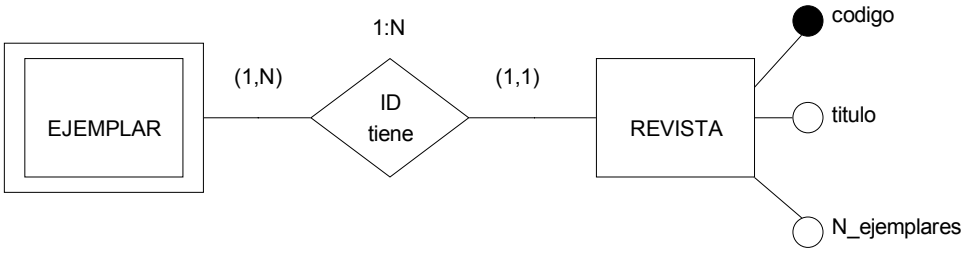
Es preciso observar que esta aparente sencillez en la transformación de algo tan complicado como es la dimensión temporal es debido a la poca semántica y precisión del Modelo E/R para tratar los aspectos temporales. En sistemas de información en los que la dimensión temporal es fundamental, como es el caso de los sistemas estadísticos, el modelo E/R no resuelve muchos de los problemas de modelización relativos a la dimensión temporal. [DEMI93, cap 23.4]

3.6.5 *Transformación de Atributos Derivados.*

No existe para los atributos derivados una representación directa y concreta en el Modelo Lógico Estándar, sino que se pueden tratar como atributos normales, que pasarán a ser columnas de la relación que corresponda (ejemplo 19).

ejemplo 19

Modelo Entidad/Relación:



Modelo Relacional:

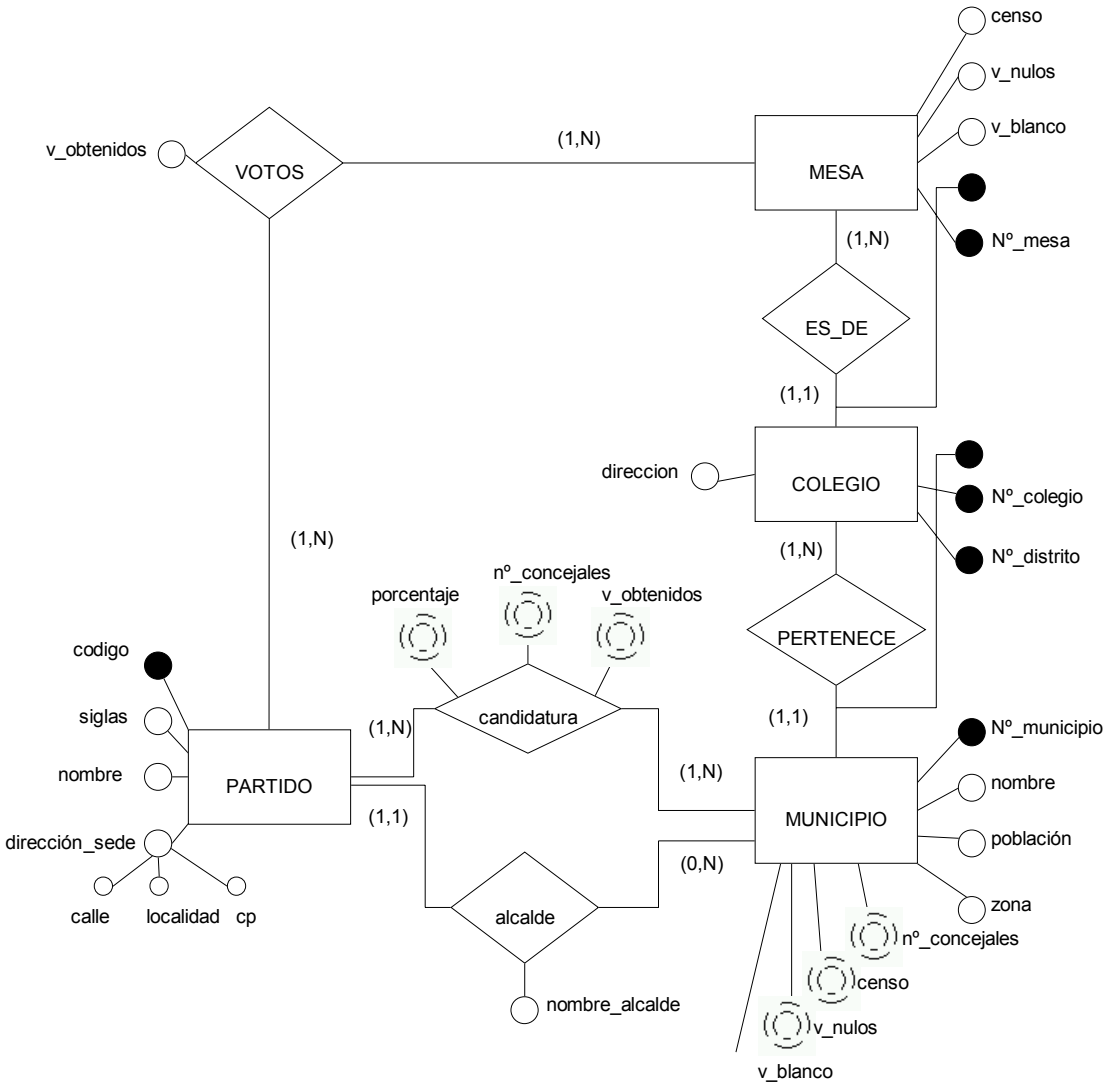
REVISTA (código, titulo, N_ejemplares)

En este caso es preciso construir un procedimiento que calcule el valor del atributo derivado cada vez que se inserten o borren las ocurrencias de los atributos que intervienen en el cálculo de éste y añadir las restricciones correspondientes. Por ejemplo, en el caso de la figura anterior, cada vez que se inserte o borre una tupla en la tabla EJEMPLAR, el procedimiento debe actualizar el atributo N_ejemplares de la tabla LIBRO. Otra solución es no almacenar las columnas que provengan de atributos derivados, creando procedimientos disparadores que calculen los valores de éstas cada vez que se recuperan, lo que en ocasiones puede resultar menos eficiente, pero puede evitar problemas de integridad y de semántica. [DEMI93, cap 23.4]

4. *Ejemplo*

A partir del esquema conceptual en el modelo Entidad/Interrelación extendido obtendremos el esquema lógico relacional por transformación y criterios utilizados en cada caso. Se dibujará además el grafo relacional correspondiente.

Diagrama Entidad Interrelación:



Obtención del esquema lógico relacional:

Se procederá a la transformación desde el esquema Entidad-Interrelación siguiendo los pasos expuestos anteriormente.

Eliminación de identificadores externos

La eliminación de identificadores externos, como ya explicamos se produce importando la clave primaria de una entidad a otra, para posteriormente poder eliminar la interrelación entre ambas entidades. Estas operaciones de propagación de clave no las llevaremos a cabo en este paso, pero si dentro del paso transformación de interrelaciones, donde se llevan a cabo las propagaciones de clave oportunas.

Eliminación de atributos compuestos

En el ejemplo nos encontramos con el atributo compuesto *dirección* (dentro de la entidad PARTIDO), que se compone de los atributos calle, localidad y cp. Obtaremos por la segunda opción de las dos explicadas, es decir, consideraremos el atributo compuesto *dirección (calle, localidad y cp)* como un atributo simple dirección, con lo cual tendremos que tener en cuenta el tamaño de la cadena de caracteres correspondiente.

Eliminación de atributos multivaluados

En este ejemplo no se encuentran este tipo de atributos.

Transformación de dominios

Los atributos del tipo *códigos*: DOMINIO CODIGOS CHAR(3)

Los atributos del tipo *números:* DOMINIO NÚMEROS INTEGER(8)

Los atributos del tipo *direcciones:* (incluye *calle, localidad y cp)*

DOMINIO DIRECCIONES CHAR(30)

Los atributos del tipo *nombres*: DOMINIO NOMBRES CHAR(20)

Los atributos del tipo *concejales:* DOMINIO CONCEJALES INTEGER(2)

Los atributos del tipo *siglas:* DOMINIO SIGLAS CHAR(10)

El atributo del tipo *porcentaje:* DOMINIO PORCENTAJE CHAR(3)

Transformación de entidades:

* La entidad PARTIDO se transforma en:

Partido (código, siglas,nombre,dirección_sede)

* La entidad MUNICIPIO se transforma en:

Municipio (nº-municipio, nombre,población, nº_concejales, zona, censo, v_nulos, v_blanco)

* La entidad COLEGIO:

Colegio (nºmunicipio, nºdistrito, nºcolegio, dirección)

* La entidad MESA:

Mesa (nºmunicipio, nºdistrito, nºcolegio, nºmesa, censo, v_nulos, v_blanco)

Transformación de Interrelaciones:

* La interrelación ES_DE: COLEGIO(1,1) : MESA(1,N). El identificador principal de Colegio se propaga a Mesa:

Mesa (nºmunicipio, nºdistrito, nºcolegio, nºmesa, censo, v_nulos, v_blanco)

La Integridad Referencial: Mesa.(nºmunicipio,nº_distrito,nº_colegio) ⟶ Colegio
Modo de borrado: CASCADA
Modo de modificación: CASCADA

* La interrelación PERTENECE : MUNICIPIO(1,1) : COLEGIO(1,N). El IP (identificador principal) de Municipio se propaga a Colegio:

Colegio (nºmunicipio, nºdistrito, nºcolegio, dirección)

Integridad Referencial: Colegio.nºmunicipio ⟶ Municipio
Modo de borrado: CASCADA
Modo de modificación: CASCADA

* La interrelación CANDIDATURA: MUNICIPIO(1,N) : PARTIDO(1,N). Se crea una nueva Relación con clave primaria los IP de Municipio y con los atributos (v_obtenidos, siglas, nombre) del tipo de interrelación:

Candidatura (nºmunicipio, código, siglas, nombre, v_obtenidos)

Integridades Referenciales: Candidatura.nº (de municipio)⟶ Municipio
Modo de borrado : Restringido
Modo de Modificación : cascada

y Candidatura.codigo (de partido)⟶ Partido
Modo de borrado : Restringido
Modo de modificación : Cascada

* Interrelación ALCALDE : MUNICIPIO (0,N) : PARTIDO (1,1). El IP de Partido se propaga a Municipio, junto con los atributos (nombre:alcalde) del tipo de interrelación:

Municipio (nºmunicipio, nombre,población, nº_concejales, zona, censo, v_nulos, v_blanco, código, nombre_alcalde)

Integridad Referencial: Municipio.Codigo (de partido)⟶ Partido
Modo de borrado : Restringido
Modo de Modificación : Cascada

* Interrelación VOTOS: MESA(1,N) : PARTIDO (1,N). Se crea una nueva relación con clave primaria los IP de Mesa y Partido y con los atributos (votos_emitidos) del tipo de interrelación:

Votos (nºmunicipio, nº_distrito, nº_colegio, nº_mesa, codigo, v_obtenidos)

Integridades Referenciales : Votos.(nºmunicipio, nº_distrito, nº_colegio, nº_mesa) → mesa

Modo de borrado : restringido
Modo de Modificación : cascada

y Votos.codigo (de partido) → partido
Modo de borrado : restringido
Modo de modificación : cascada

ACLARACIONES SOBRE LAS INTEGRIDADES REFERENCIALES: En los modos de borrado no se pueden borrar Partidos o Municipios si existen datos electorales de ellos, es decir, candidaturas y/o Votos. Se pueden borrar Municipios de los cuales sólo se tienen datos de sus colegios y mesas. En los modos de modificación si se cambia el IP se deben modificar en cascada todas las referencias hacia el.

Transformación de atributos de interrelaciones

Ya han sido transformados en los pasos anteriores.

Transformación de restricciones

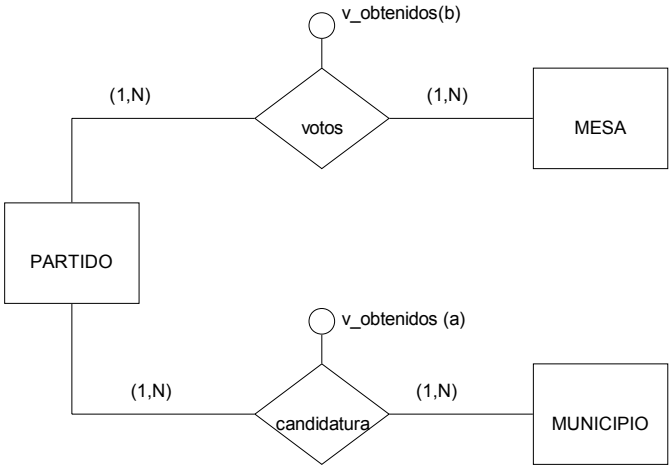
En el ejemplo tendremos las siguiente restricciones de integridad:

1.- El nº de concejales de un municipio es:

Si Población <500:	5
de 500 a 999 :	7
de 1000 a 1999:	9
de 2000 a 4999:	11
de 5000 a 9999:	13
de 10000 a 19999:	15
de 20000 a 49999:	17
mas de 49999:	17 + 2 por cada fracción

Esta restricción se hará mediante un CHECK cuando creemos la tabla correspondiente a la entidad MUNICIPIO (En SQL2)

2.- Tenemos el siguiente subesquema:



Los votos obtenidos(a) en Municipio por un partido son la suma de los obtenidos en las mesas del municipio(b). (Lo haremos en SQL2 mediante la restricción: CREATE ASSERTION) .

Esquema Relacional Final.

*** Mesa (nºmunicipio, nºdistrito, nºcolegio, nºmesa, censo, v_nulos, v_blanco)**

*** Colegio (nº municipio, nºdistrito, nºcolegio, dirección)**

*** Municipio (nºmunicipio, nombre,población, nº_concejales, zona, censo, v_nulos, v_blanco, codigo, nombre_alcalde)**

*** Partido (código, siglas,nombre,dirección_sede)**

*** Candidatura (nºmunicipio, codigo, siglas, nombre, v_obtenidos)**

*** Votos (nºmunicipio, nº_distrito, nº_colegio, nº_mesa, codigo, v_obtenidos)**

Esquema Relacional en SQL-92

* Dominios :

```
CREATE DOMAIN Codigos      CHAR(3)

CREATE DOMAIN números      INTEGER(8)

CREATE DOMAIN direcciones  CHAR(30)

CREATE DOMAIN nombres      CHAR(20)

CREATE DOMAIN concejales   INTEGER(2)

CREATE DOMAIN siglas       CHAR(10)
```

* tablas :

```
CREATE TABLE Mesa
(       nºmunicipio         códigos,
        nºdistrito          códigos,
        nºcolegio           códigos,
        nº_mesa             códigos,
        censo               números,
        v_nulos             números,
        v_blanco            números,

PRIMARY KEY (nºmunicipio, nº_distrito, nº_colegio, nº_mesa),
FOREIGN KEY (nºmunicipio, nº_distrito, nº_colegio) REFERENCES Colegio
        ON DELETE CASCADE
        ON UPDATE CASCADE  )

CREATE TABLE Colegio
(       nº municipio        Codigos,
        nº_distrito         códigos,
        nº_colegio          códigos,
        dirección           direcciones,

PRIMARY KEY ( nºmunicipio, nº_distrito, nº_colegio),
FOREIGN KEY (nºmunicipio) REFERENCES Municipio
        ON DELETE CASCADE
        ON UPDATE CASCADE  )
```

```
CREATE TABLE Municipio
(       nºmunicipio             códigos,
        nombre                  nombres,
        población               números,
        nº_concejales           concejales,
        zona                    códigos,
        censo                   números,
        v_nulos                 números,
        v_blanco                números,
        código(de partido)      códigos,
        CHECK ( (nº_concejales=5 AND población<500)
                OR (nº_concejales=7 AND población BETWEEN 500 AND 999)
                OR (nº_concejales=9 AND población BETWEEN 1000 AND 1999)
                OR (nº_concejales=11 AND población BETWEEN 2000 AND 4999)
                OR (nº_concejales=13 AND población BETWEEN 5000 AND 9999)
                OR (nº_concejales=15 AND población BETWEEN 10000 AND 19999)
                OR (nº_concejales=17 AND población BETWEEN 20000 AND 49999)
                OR (población>49999) AND ((población/50000)=((nºconcejales-17)/2)))
                )
PRIMARY KEY (nºmunicipio),
FOREIGN KEY (código) REFERENCES Partido
        ON UPDATE CASCADE )

CREATE TABLE Partido
(       código                  códigos,
        siglas                  tiposiglas,
        nombre                  nombres,
        dirección_sede          direcciones,

PRIMARY KEY (CODIGO)   )

CREATE TABLE Candidatura
(       nº municipio            códigos,
        codigo                  códigos,
        siglas                  tiposiglas,
        v_obtenidos             números,

PRIMARY KEY (nºmunicipio, codigo),
FOREIGN KEY (nºmunicipio) REFERENCES Municipio
        ON UPDATE CASCADE,
```

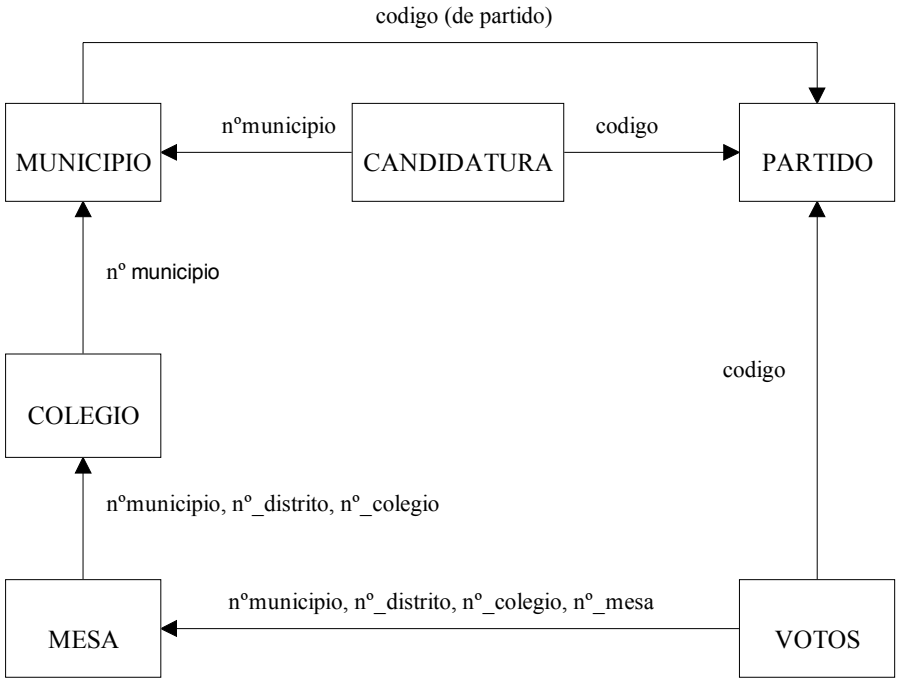
```
FOREIGN KEY (codigo) REFERENCES Partido
        ON UPDATE CASCADE   )

CREATE TABLA Votos
(       nº municipio            códigos,
        nº_distrito             códigos,
        nº_colegio              códigos,
        nº_mesa                 códigos,
        codigo                  códigos,
        siglas                  tiposiglas,
        nombre                  nombres,
        v_obtenidos(b)          números,

PRIMARY KEY (nºmunicipio, nº_distrito, nº_colegio, nº_mesa, codigo),
FOREIGN KEY (nºmunicipio, nº_distrito, nº_colegio, nº_mesa) REFERENCES Mesa
        ON UPDATE CASCADE,
FOREIGN KEY (codigo) REFERENCES Partido
        ON UPDATE CASCADE   )

CREATE ASSERTION concuerdan_votos
CHECK (SELECT  nºmunicipio, código, v_obtenidos
                FROM candidatura c
                WHERE c.v_obtenidos=(SELECT sum(v.v_obtenidos)
                                     FROM votos v
                                     WHERE (c.nºmunicipio=v.nºmunicipio) AND
                                           (c.código=v.código)
                                     GROUP BY v.nºmunicipio, v.código
                                     )
            )
```

Grafo Relacional



5. *Bibliografía*

Básica:

[BATI94] Batini, C; Ceri, S.; Navathe, S. B. (cap 12, cap relacionado 11)
Diseño Conceptual de Bases de Datos. Un enfoque de entidades-interrelaciones.
Addison-Wesley/Díaz de Santos, 1994.

[DEMI93] De Miguel, A.; Piattini, M. (cap 23, caps relacionados 15 y 24)
Concepción y Diseño de Bases de Datos
Rama, 1993.

[KORTH93] Korth, H.F.; Silberschatz, A. (caps 3 y 5)
Fundamentos de Bases de Datos (2ª Edición)
McGraw-Hill, 1993.

Complementaria:

[DATE93] Date, C. J. (caps 11, 12 y 15)
Introducción a los Sistemas de Bases de Datos. Vol 1 (5ª edición)
Addison-Wesley Iberoamericana, 1993.

[ELMA89] Elmasri, R.; Navathe, S.B. (capítulo 6)
Fundamentals of Database Systems (2nd. edit.)
Addison-Wesley, 1994.

Otros:

Beynon-Davies, Paul (capítulos 2,4,6 y 8)
Relational Database Systems
Blackwell Scientific Publications, 1991.

Brathwaite, Kenmore S. (capítulos 2, 3 y 9)
Relational Database Systems
Academic Press, 1991.

Database language SQL2, Draft International Standard. ISO/IEC DIS 9075.

Database Language SQL. ISO/IEC 9075