



## UD 3-1. DISEÑO LÓGICO. PARTE 1

Base de Datos  
CFGs DAW

Francisco Aldarias Raya  
[paco.aldarias@ceedcv.es](mailto:paco.aldarias@ceedcv.es)

2019/2020

Fecha 25/11/19

Versión:191125.1713

## Licencia



**Reconocimiento - NoComercial - CompartirIgual (by-nc-sa):** No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

## Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:



Importante



Atención



Interesante

## Revisiones

21/10/2019. Apartado 3.2. Restricciones de Claves. Superclaves. Apartado 4.2.1. Relaciones del modelo E/R con cardinalidad N:M. Apartado 4.2.6. Especializaciones.

22/10/2019. Apartado 3.5. Eliminado ejemplo valor no nulo. Apartado 4,2,6. La opción 1 y 2 era la misma. Se elimina la segunda. Apartado 3.3. Integridad Refencial. Representación clave ajena.

24/10/2019. Apartado 4.1.1.. Entidades fuertes, dejando VNN en lugar de valor NULO.

28/10/2019. Apartado 4.1.1. Atributos compuestos y multievaluados.

5/11/2019. Apartado 4.2.7 Ternarias.

20/11/2019. Apartado 4.2.4 Reflexivas N:M Apartado 4.2.1 Relaciones Binarias N:M con restricción de existencia.

## ÍNDICE DE CONTENIDO

<b>1.MODELO LÓGICO.....</b>	<b>4</b>
<b>2.ELEMENTOS DEL MODELO LÓGICO.....</b>	<b>4</b>
2.1 Relación.....	4
2.2 Esquema.....	5
2.3 Dominio.....	5
2.4 Cuerpo.....	5
<b>3.RESTRICCIONES.....</b>	<b>6</b>
3.1 Restricciones generales.....	6
3.2 Restricciones de Claves.....	6
3.3 Restricciones de Integridad Referencial.....	7
3.4 Restricciones de Dominio.....	9
3.5 Restricciones de valor Nulo.....	9
3.6 Restricciones de valor Único.....	10
<b>4.PASO DEL MODELO CONCEPTUAL AL MODELO LÓGICO.....</b>	<b>10</b>
4.1 Transformación de Entidades.....	11
4.1.1 Entidades fuertes.....	11
4.1.2 Entidades débiles.....	13
4.2 Transformación de Relaciones.....	15
4.2.1 Relaciones del modelo E/R con cardinalidad N:M.....	15
4.2.2 Transformación de relaciones E/R con cardinalidad 1:N.....	16
4.2.3 Transformación de relaciones E/R con cardinalidad 1:N reflexivas.....	17
4.2.4 Transformación de relaciones E/R con cardinalidad N:N reflexivas.....	18
4.2.5 Transformación de relaciones E/R con cardinalidad 1:1.....	19
4.2.6 Transformación de relaciones E/R de grado n.....	22
4.2.7 Transformación de Especializaciones.....	22
4.2.8 Relaciones ternarias.....	24
<b>5.BIBLIOGRAFÍA.....</b>	<b>25</b>

## UD03-1. DISEÑO LÓGICO. PARTE 1

### 1. MODELO LÓGICO

Una vez creado el modelo Conceptual de datos (Modelo E/R), tal y como vimos en el tema anterior, procederemos a crear el modelo lógico (también conocido como Modelo Lógico Relacional) a partir del cual, en el siguiente paso, obtendremos el modelo físico que será el que realmente utilizaremos para crear la base de datos que hemos diseñado.

El modelo lógico es independiente del Gestor de Base de Datos que vayamos a utilizar, sin embargo sí se debe conocer el tipo de gestor de base de datos que vamos a utilizar. Ya comentamos que en este curso utilizaremos bases de datos de tipo relacional son las más utilizadas.

En esta unidad vamos a ver cómo pasar de nuestro modelo conceptual (o modelo E/R) al modelo relacional (o modelo lógico), para ello vamos a comenzar con unos conceptos básicos y posteriormente veremos cómo realizar esa transformación.

Antes de comenzar, dos matizaciones importantes:

- El modelo relacional tiene una completa base matemática que permite representar, demostrar y ejecutar todas las operaciones que podemos realizar con la base de datos. Esta base matemática hace que sea un modelo fácilmente implementable con ordenadores. Sin embargo, el objetivo de este curso es más práctico que teórico, por ello no vamos a entrar en el desarrollo matemático del modelo. Para los alumnos que estén interesados, hay muchísimos libros que desarrollan completamente este modelo y es fácil encontrarlos.
- Por otro lado, vamos a ver que en el Modelo Lógico hablaremos nuevamente de Relaciones, pero ya veremos que las relaciones del modelo lógico no tienen nada que ver con las relaciones del modelo E/R. Tenedlo en cuenta desde ya!!!

### 2. ELEMENTOS DEL MODELO LÓGICO

#### 2.1 Relación

La relación es el elemento básico de este modelo.

Una relación se representa en forma de tabla donde las cabeceras de las columnas representa los diferentes atributos de la relación y cada una de las filas, que reciben el nombre de tuplas, representa la información de una ocurrencia de la relación.

DNI	Nombre	Fecha_Nacimiento	Ciclo	Curso
23456789-P	Javier Sanz	10/06/1994	AFI	1
34567890-Q	Fernando Gil	12/03/1996	ASIR	2
12345678-T	María Amor	20/08/1995	ASIR	1
13243567-E	Ana Martín	28/10/1994	DAM	1

## 2.2 Esquema

Se llama esquema de la Relación (también es conocido como intensión) al nombre de la relación junto al conjunto de atributos que define la estructura de la tabla.

El esquema de una Relación puede representarse con el nombre de la Relación junto a sus atributos entre paréntesis y separados por comas.

Alumnos (DNI, Nombre, Fecha\_Nacimiento, Ciclo, Curso)

## 2.3 Dominio

Para cada atributo también podemos indicar su dominio (recuerda que el dominio representa el conjunto de valores permitido en dicho atributo, por ejemplo si es una cadena de caracteres, si son valores numéricos, si es una fecha, etc.).

## 2.4 Cuerpo

Se llama cuerpo de la Relación (también es conocido como extensión) al conjunto de tuplas que forman la tabla.

∞ Habitualmente a las relaciones se las llama Tablas, a los atributos Campos y a las tuplas Filas.

El modelo lógico, además, tiene una serie de restricciones que deben cumplir las relaciones de nuestro diseño. Estas restricciones son:

- Restricciones generales.
- Restricciones de claves.
- Restricciones de Integridad Referencial.
- Restricciones de Dominio.
- Restricciones de valor NULO.
- Restricciones de valor único.

Veamos cada una de ellas de forma detallada.

### 3. RESTRICCIONES

#### 3.1 Restricciones generales

Algunas de las restricciones generales del modelo lógico son:

- No podemos tener dos tuplas (filas) iguales. Las tuplas deben tener un valor diferente en al menos uno de sus atributos (campos).
- El orden de las tuplas (filas) es indiferente. El orden en el que se almacenen las filas de una tabla no es significativo.
- El orden de los atributos (campos) de la Relación también es indiferente, no es significativo.
- En cada tupla (fila) cada atributo (campos) solamente puede tener un valor.
- Un atributo que forme parte de la clave primaria o principal no puede tomar valores nulos en una tupla (un valor nulo es ausencia de valor, es decir que no tiene un valor asignado).

#### 3.2 Restricciones de Claves

Vamos a definir el concepto de Clave en el modelo lógico (muy similar al de clave del modelo E/R).

└ Una **Clave** es un atributo (campo) o conjunto de atributos (campos) que permite identificar de forma unívoca cada una de las tuplas de una relación (antes hablábamos de las ocurrencias de una Entidad).

Es decir, mediante la clave podemos identificar cada una de las filas de la relación. Pues bien, el modelo lógico distingue diferentes tipos de claves:

- **Superclaves:** Son atributos o conjuntos de atributos que permiten identificar una tupla (fila) dentro de una relación. Las Superclaves pueden ser no mínimas, es decir puede ser que tenga campos extras que no son necesarios para identificar las filas. Por ejemplo, en la Relación alumnos (que representa los alumnos de un centro educativo de Formación Profesional) podemos tener como superclave el DNI del alumno, pero también podría ser el DNI + Nombre del alumno (aunque obviamente añadir el Nombre del alumno no es imprescindible para identificar al alumno pues ya lo hace el DNI por sí solo). Los atributos superclaves no deben usarse.

- **Claves Candidatas:** Una clave candidata es una Superclave mínima, es decir una Superclave sin atributos extras que no sean necesarios para identificar cada tupla (fila). Puede ocurrir que en una Relación encontremos varias claves Candidatas. Por ejemplo, para los empleados de una empresa podríamos tener el número de DNI y el número de la Seguridad Social. Para un alumno como en el ejemplo anterior podríamos tener el Número de Expediente asignado en el centro donde estudia o su DNI.

- **Clave Primaria (Principal):** De entre todas las claves Candidatas encontradas debemos elegir una de ellas para que sea la clave Primaria (Principal). La clave Primaria (Principal) puede ser simple (formada por un atributo o campo o compuesta, formada por varios atributos o campos). Lo que es muy importante tener en cuenta es que ningún atributo o campo que forme parte de la clave Primaria (Principal) puede tomar valores nulos y además son únicos para esa relación.

∞ Ningún campo de la clave puede quedar vacío.

Cuando hay varias claves Candidatas, la selección de una de ellas frente al resto debe realizarse en función del contexto en el que se vaya a trabajar con nuestra base de datos. Por ejemplo, si disponemos de una relación Empleados donde tenemos las claves candidatas Numero de la Seguridad Social y DNI, si la base de datos que estamos diseñando es para la Tesorería de las Seguridad Social, posiblemente las consultas más frecuentes se realicen por el número de la seguridad social, sin embargo si es para una Asesoría Fiscal posiblemente utilizarán con mayor frecuencia el DNI. Es decir, habrá que estudiar cada caso en función del contexto donde se vaya a manejar la aplicación. También, salvo causas muy justificadas, siempre es mejor seleccionar como Clave Primaria (Principal) una clave simple frente a una compuesta.

∞ En el esquema de una relación, **la clave primaria se representará subrayada o indicándolo con CP**, para distinguirla del resto de atributos. Si la clave primaria es compuesta se subrayarán todos los atributos que la forman.

Alumnos (DNI, Nombre, Fecha\_Nacimiento, Ciclo, Curso)

CP: DNI

- **Clave Alternativa:** Las claves Alternativas serán todas aquellas claves Candidatas que no han sido seleccionadas como Clave Primaria (Principal).

Existe además otro tipo de clave, las claves foráneas, también llamadas ajenas, externas o extranjeras, que son atributos o campos de una Relación que al mismo tiempo son clave Primaria (Principal) en otra Relación. **Se representan con un subrayado discontinuo o indicándolo con C.Alternativa.**

∞ Hay que tener en cuenta que en la Relación donde ese atributo o conjunto de atributos actúa como clave foránea sí puede tener valores repetidos, donde no puede tenerlos repetidos es donde actúa como clave Primaria o Principal.

Veamos un ejemplo. Supongamos que una empresa tiene Clientes y se ha considerado que la clave Primaria de esta Relación será el CIF del cliente. Pues bien, también tenemos otra Relación llamada Facturas cuya clave Primaria es el Número de Factura y donde otro de los atributos que aparece en ella es el CIF del cliente que ha comprado los artículos facturados. Pues bien, en la Relación Clientes solo puede haber una fila o tupla por cada CIF ya que está actuando como clave principal y además el CIF es único para cada cliente. Sin embargo, en la Relación Facturas, donde actúa como clave Foránea el CIF puede aparecer repetido, ya que un cliente puede tener muchas facturas (cada factura es una tupla o fila) y en cada una de ellas perteneciente a un mismo cliente aparecerá el mismo CIF.

### 3.3 Restricciones de Integridad Referencial

Las restricciones de Integridad Referencial aparecen cuando tenemos claves ajenas ( o también llamadas foráneas), en las Relaciones. La integridad Referencial pretende determinar el comportamiento de la base de datos para asegurar la integridad y la coherencia de la información almacenada.

Las claves ajenas se representa poniendo la etiqueta C.Ajena:  $a \rightarrow B$ . Siendo a el atributo que es clave ajena, y B la tabla donde se encuentra la referencia.

Supongamos el ejemplo anterior de la Relación Clientes y la Relación Facturas en las que existe un atributo o campo, el CIF que es clave Primaria en Clientes y Foránea en Facturas. Este campo nos permite averiguar qué facturas pertenecen a un determinado cliente (a través de su CIF) y por otro lado también nos permite conocer a quién pertenece una determinada factura (también a través del CIF).

Pues bien, la integridad referencial es la que nos dirá cómo debe actuar la base de datos en situaciones como estas:

¿Qué ocurre si se borra un cliente? No podemos borrar la tupla del cliente y ya está, pues esto causaría problemas en la integridad de la información almacenada en la base de datos. Por ejemplo si borramos un cliente que ya tiene facturas, esas facturas se quedarían sin su correspondiente cliente y cuando hagamos la pregunta de ¿a quien pertenece una determinada factura?, nos podemos encontrar que no podemos responder, pues fue eliminado. Esto es una inconsistencia de la información, se ha roto la integridad referencial, pues hacemos referencia a un elemento que ya no existe (ya que fue eliminado).

Para que esto no ocurra debemos establecer los mecanismos correspondientes mediante las restricciones de la Integridad Referencial de las siguientes formas:

- Por ejemplo, sólo se permitirá eliminar un cliente si ese cliente no tiene facturas. Esto es lo que se denomina **borrado restringido**.
- Otra opción es **borrado en cascada**. Podemos eliminar un cliente, pero ello implicará eliminar también todas las facturas que se corresponden con el CIF de ese cliente.
- La última opción, aunque en nuestro ejemplo no sería válido pero sí para otros casos, sería eliminar el cliente y poner el CIF a nulos en todas las facturas de ese cliente (hay que tener en cuenta que en nuestro ejemplo no es válido pues legalmente no podemos tener facturas que no pertenezcan a un cliente).

Supongamos otro ejemplo similar. Una persona tiene una cuenta en un banco y a partir de esa cuenta realiza una serie de pagos de facturas y recibos que se cargan mensualmente, por ejemplo la luz, el gas, la tarjeta de crédito, etc. Está claro que en la relación Cuenta tendremos un atributo que será el número de cuenta que actuará como clave primaria y por otro lado en la relación Movimientos (de la cuenta) tendremos como clave foránea el número de cuenta a la que pertenecen los movimientos.

¿Qué ocurre si se realiza una fusión bancaria, ahora tan de moda, y mi cuenta pasa a pertenecer a una nueva entidad cambiando mi número de cuenta?



Pues bien, si solamente se cambia el número de cuenta en la relación Cuenta perderé toda relación con los movimientos que se realizaron previos a dicho cambio. Además esos movimientos se quedarán sin referencia en el número de cuenta, lo cual puede provocar una inconsistencia, o lo que es peor, si mi número antiguo de cuenta ya existe en la nueva entidad, mis antiguos movimientos quedarán automáticamente asociados a la cuenta de otra persona distinta! Para evitar esto, igual que en el caso anterior, podemos:

- Por ejemplo, hacer una actualización restringida, donde sólo se permite la modificación de una clave primaria, en este caso el número de cuenta, si no tiene movimientos asociados.
- Pero lo más habitual es una actualización en cascada. La modificación de una clave primaria conlleva la actualización de ese valor en todos los movimientos asociados a ese número de cuenta que vamos a modificar. De esta forma los movimientos quedarían asociados al nuevo número asignado a la cuenta bancaria.

Existe además otra posibilidad en la integridad referencial que se llama **SET DEFAULT** (poner el valor por defecto). Cuando creamos una Relación podemos establecer un valor por defecto para un atributo. Esta opción de la integridad referencial nos permite que al eliminar o modifica una fila en la Relación donde el campo actúa como clave primaria, en el resto de Relaciones donde actúa como clave foránea se coloque en ese campo el valor por defecto establecido. Dicho así queda algo abstracto, pero simplemente se trata de poner un valor predeterminado en lugar del valor NULO. Lo veremos más detalladamente mediante ejemplos prácticos.

En resumen, las posibilidades que tenemos y la forma de representarlas son las siguientes:

- Borrado en Cascada, se indicará como B:C
- Borrado con puesta a NULO, se indicará como B:N
- Borrado con puesta al valor por defecto, se indicará como B:D
- Borrado restringido, se indicará como B:R
- Modificación en Cascada, se indicará como M:C
- Modificación con puesta a NULO, se indicará como M:N
- Modificación con puesta al valor por defecto, se indicará como M:D
- Modificación restringida, se indicará como M:R

### 3.4 Restricciones de Dominio

└ Las restricciones de Dominio exigen que los valores asignados a un atributo estén dentro del dominio definido.

Por ejemplo si para la edad se ha considerado oportuno un valor entero, que no se emplee un valor real o una fecha. Si un nombre puede tener 25 caracteres que no tenga 30, etc.

Además también podemos incluir en este apartado las restricciones de verificación o CHECK que nos permiten comprobar si un valor es válido conforme a una expresión. Por ejemplo, validar que la edad no pueda ser mayor de 125 o bien que el número de caracteres de un DNI sea de 8 dígitos más una letra, etc.

### 3.5 Restricciones de valor Nulo

└ La restricción de valor NO NULO (NULL o NOT NULL) implica la obligatoriedad o no de que un atributo tenga un valor en cada tupla o fila.

Cuando indicamos que un atributo es NO NULO (NOT NULL) estamos obligando a que ese campo tiene que ser rellenado con algún valor, no se puede quedar vacío (esta restricción en algunos gestores recibe el nombre de Requerido [Si/No]). Por el contrario, si decimos que un atributo puede ser NULO (NULL), esto implica que puede haber tuplas con ese campo sin valor, es decir, es opcional rellenarlo.

Por supuesto, vamos a recordar, que ningún atributo que sea clave principal o forme parte de ella puede ser NULL, esos campos siempre deben ser NOT NULL.

### 3.6 Restricciones de valor Único

└ Las restricciones de valor único (UNIQUE) nos permiten asegurar que en un atributo de la base de datos no se puedan introducir valores repetidos.

Si indicamos que un atributo de nuestra relación tiene la restricción de unicidad podemos estar seguros de que todos los valores que tome serán diferentes, pues en caso de que se intente introducir un valor duplicado aparecerá un error y no será admitido dicho valor en la tabla. Se representa con la etiqueta **UNICO: Nombre del atributo**.

Los valores únicos se utilizan cuando la cardinalidad máxima de una relación es 1, y sobre la clave ajena.

Por supuesto, como ya imaginaréis, todas las claves primarias simples tendrán la restricción de unicidad, pues una clave primaria nunca puede tomar valores repetidos ya que debe identificar de forma unívoca cada una de las filas de la tabla y si hubiese valores repetidos eso no se podría conseguir.

Por último, existen otros tipos de restricciones como son las Aserciones (ASSERTION) y los Disparadores (TRIGGER) que estudiaremos más adelante en el curso.

## 4. PASO DEL MODELO CONCEPTUAL AL MODELO LÓGICO

En este punto vamos a estudiar, paso a paso, cómo transformar el modelo conceptual de datos en el modelo lógico, concretamente como pasar del modelo E/R al modelo Relacional que estamos estudiando.

Para ello iremos viendo cada uno de los elementos que forman el modelo E/R y cual es su correspondencia en el Modelo Relacional.

El primer paso será transformar las Entidades del modelo E/R en sus Relaciones (Tablas) del modelo Relacional.

El segundo paso será transformar las Relaciones del modelo E/R al modelo Relacional.

Recordad que una relación en el modelo E/R no es lo mismo que una relación (tabla) en el modelo Relacional.

#### 4.1 Transformación de Entidades

Vamos a recordar que en el modelo E/R tenemos dos tipos de Entidades, las Fuertes y las Débiles. Veamos como transformar cada una de ellas.

##### 4.1.1 Entidades fuertes

▬ Cada entidad fuerte se transformará en una Relación (tabla). Cada atributo de la entidad se transformará en un atributo o campo de la Relación.

Los atributos o campos en el modelo relacional pueden nulos, si alguno de ellos es no nulo debemos indicarlo. Para ello se pondrá un asterisco detrás del nombre del atributo que pueda ser VALOR NO NULO o indicándolo con VNN.

Cada dominio de los atributos se transformará en el dominio del atributo o campo correspondiente de la Relación.

La clave primaria o principal de la Relación estará formada por la clave principal de la entidad que estamos transformando. La clave principal se indicará en el modelo Relacional subrayando los campos que la forman.

Las claves ajenas o foráneas (recordamos, aquellos atributos o campos que son clave principal en otra Relación) son representadas de diferentes maneras, algunos autores las representan en cursiva, otros subrayado punteado bajo el nombre, etc. Nosotros utilizaremos el con una flecha a la derecha del nombre. Por ejemplo si encontramos un esquema con algo así:

**Cuenta\_Bancaria** (Número, Saldo, Fecha\_Creación, Número\_Contrato , Crédito\*, DNI)

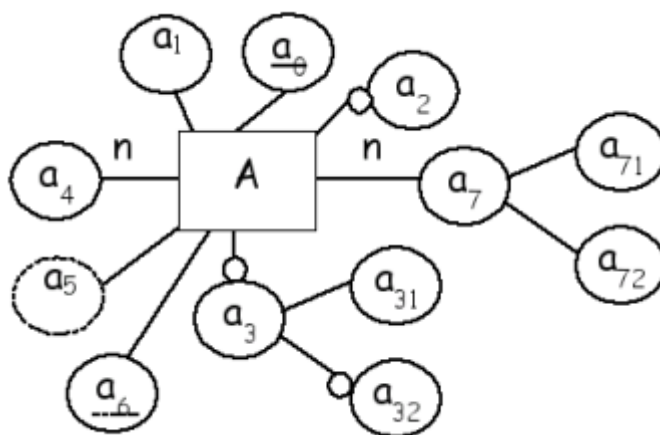
CP: Número

Clave Ajena: { DNI} → Cliente

VNN: Crédito

Podemos saber que la Relación se llama Cuenta\_Bancaria, que la clave principal o primaria es Número, Que tiene como clave ajena DNI.

Supóngase una entidad, que no sea débil ni especializada, con su conjunto de atributos



La relación equivalente es la siguiente: ´

$A(a_0, a_1, a_2, a_3.a_{31}, a_3.a_{32}, a_4, a_5, a_6, a_7.a_{71}, a_7.a_{72})$

Clave Primaria:  $\{a_0\}$

Valor No Nulo:  $\{a_2\}$

Valor No Nulo:  $\{a_3.a_{32}\}$

Único:  $\{a_6\}$

Donde:

- El atributo identificador (en el ejemplo  $a_0$ ) se convierte en la clave primaria de la relación. Si la entidad tuviera un conjunto de atributos identificadores, el conjunto de todos ellos formaríala la clave primaria de la relación.
- Las restricciones de valor no nulo (p.e.  $a_2$ ) y las restricciones de unicidad (p.e.  $a_6$ ) se representan sin problemas como se muestra en el ejemplo.
- Los atributos multivaluados (p.e.  $a_4$ ) se incluyen en la relación destacados, cada uno de ellos, entre llaves. Los problemas que puedan derivarse de la presencia de estos atributos se resolveran en la fase de normalización.
- Los atributos compuestos (p.e.  $a_3$ ) se representan descomponiendolos en sus atributos componentes. Esta descomposición podría posponerse hasta la fase de normalización, pero el realizarla directamente permite especificar algunas restricciones de valor no nulo sobre los atributos componentes. Por otra parte, si no hay problemas de ambigüedad con los nombres de los atributos, podrá utilizarse como nombre el descriptor del campo ( $a_{31}, a_{32}, \dots$ ). Obsérvese como se ha representado el atributo  $a_7$  que es un atributo compuesto multievaluado. Los atributos derivados (p.e.  $a_5$ ) se incluyen en la relación como un atributo más, ´ debiendo especificarse su

formula de derivación como restricción.

#### 4.1.2 Entidades débiles

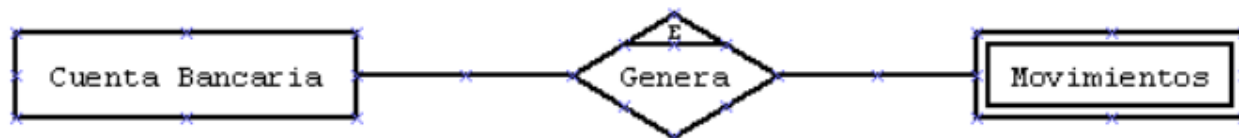
Recuerda que una entidad débil es aquella que necesita de una entidad fuerte para poder existir.

Además vimos que la relación entre la entidad Fuerte y la entidad Débil podía ser de dos tipos: de existencia o de identificación. La transformación de ambas es muy parecida, solo hay algún detalle que debemos tener en cuenta.

Realizaremos la transformación como si fuera una entidad fuerte y después analizaremos el tema de las claves principales y foráneas.

##### Dependencia de existencia

Comencemos por una entidad débil con dependencia de existencia como puede ser:



Vamos a suponer que la entidad Cuenta Bancaria tiene los atributos que hemos transformado en los campos indicados en el apartado anterior, y que la entidad Movimientos tiene otros atributos y una clave principal propia que es Referencia\_Movimiento que consiste en un número único que se asigna de forma automática a cada movimiento creado en el banco.

La entidad Cuenta Bancaria se transformaría en el mismo esquema relacional visto en el apartado anterior quedando:

Cuenta\_Bancaria (Número, Saldo, Fecha\_Creación, DNI, Número\_Contrato, Crédito)

CP: Número

Y por ejemplo la entidad movimientos quedaría (los campos son inventados, pues no partimos de un ejemplo concreto)

Movimientos (Referencia\_Movimiento, Tipo\_Operación, Importe, Ordenante, Número)

CP: Referencia\_movimiento

VNN: Número

C Ajena: Número → Cuenta\_Bancaria

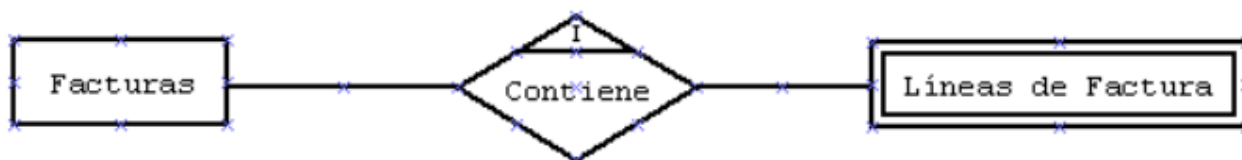
Donde podemos ver que la clave primaria o principal es Referencia\_Movimiento (tiene su propia clave principal), después tenemos otros campos y por último Número que es una clave foránea que hace referencia a la clave principal de la entidad fuerte, en este caso, ya transformada, a la relación Cuesta\_Bancaria.

Luego, lo que tenemos que tener siempre en cuenta es:

— Al transformar una entidad débil con dependencia de existencia en una relación siempre incluiremos como clave foránea o ajena: la clave principal de la entidad fuerte.

### Dependencia de Identificación

Supongamos que tenemos una parte de nuestro modelo E/R con el siguiente esquema:



Transformaremos la entidad Facturas en su correspondiente Relación o Tabla (solamente incluyo unos pocos atributos, la factura tendría muchísimos, los suficientes para entendernos).

Facturas (Número Factura, CIF\_Cliente, Fecha, Base, ImpIVA, ImpTotal)

CP: Número\_de\_Factura

En este esquema podemos ver que tenemos una relación o tabla llamada Facturas cuya clave principal es el número de factura y que tiene una clave foránea que es el CIF del cliente. Además ninguno de los campos puede ser nulo.

Transformaremos ahora la entidad Líneas de Factura, tal como si fuese una entidad fuerte y después analizaremos las claves.

Líneas\_de\_Facturas(Número Factura, Número Línea, Código Producto, Cantidad, Precio)

CP: {Número\_Factura, Número\_Línea}

Clave Ajena: Número\_Factura → Facturas

Clave Ajena: Código\_producto → Producto

VNN: Número\_Factura

El esquema nos dice que la nueva Relación se llama Líneas de Facturas y que la clave principal está formada por el Número de Factura más el Número de línea de la factura. Lo normal es que al ser una entidad con dependencia de Identificación ya aparezca la clave principal de la entidad fuerte formando parte de la clave de la entidad débil (por eso es dependencia de identificación), pero si no ocurriese, deberíamos añadir a la clave principal de la Relación que aparezca a partir de la entidad débil, la clave principal de la Relación que se deriva de la entidad fuerte. En este caso, como prácticamente siempre (si tenemos bien realizado el modelo E/R) el Número de Factura ya forma parte de la clave de la Relación Líneas de Facturas.

Además podemos apreciar que a su vez, el Número de Factura también es clave foránea, pues es clave principal en otra relación (Facturas), igual que ocurre con el Código del producto.

□ Al transformar una entidad débil con dependencia de identidad en una relación siempre incluiremos como clave principal y foránea: la clave principal de la entidad fuerte.

## 4.2 Transformación de Relaciones

Las relaciones de nuestro modelo E/R pueden tener diferentes cardinalidades y la transformación al modelo Relacional será diferente en función su cardinalidad:

### 4.2.1 Relaciones del modelo E/R con cardinalidad N:M

□ La relación con cardinalidad N:M del modelo E/R se transformará en una relación o tabla del modelo Relacional que tendrá como clave primaria la concatenación de las claves primarias de las entidades que relaciona. Además cada uno de los atributos que forman la nueva clave principal serán a su vez claves foráneas, pues son claves principales en las otras relaciones o tablas. En esa misma tabla se añadirán los atributos de la relación N:M, si los hay.

Veamos un ejemplo: Supongamos que tenemos una academia de informática y en nuestro modelo de datos aparece una parte como la siguiente:



Por un lado tendremos la tabla Alumno, por otro la tabla Cursos y además aparecerá una nueva tabla fruto de la relación N:M Está matriculado que podemos llamar Matrículas. La transformación quedaría así (los atributos son inventados, pues no aparecen en el ejemplo):

Alumno (DNI, Nombre, Fecha\_Nacimiento)

CP: DNI

Cursos (Cod\_Curso, Nombre, Tipo\_Curso, Fecha\_Comienzo, Número\_Horas, Horario)

CP: Cod\_Curso

Matrículas (DNI, Cod\_Curso, Fecha\_Matricula)

CP: {DNI, Cod\_Curso}

CA: {DNI} → Alumno

CA: {Cod\_Curso} → Curso

Podemos observar como la clave principal de la nueva tabla Matrículas está formada por la clave principal de Alumno más la clave principal de Cursos y además ambas a su vez con claves foráneas por ser claves principales en sus respectivas tablas.

Los casos de relaciones binarias M:M en los que aparecen **restricciones de existencia** se diseñan de manera análoga, siendo necesario escribir restricciones de integridad que representen las restricciones de existencia.

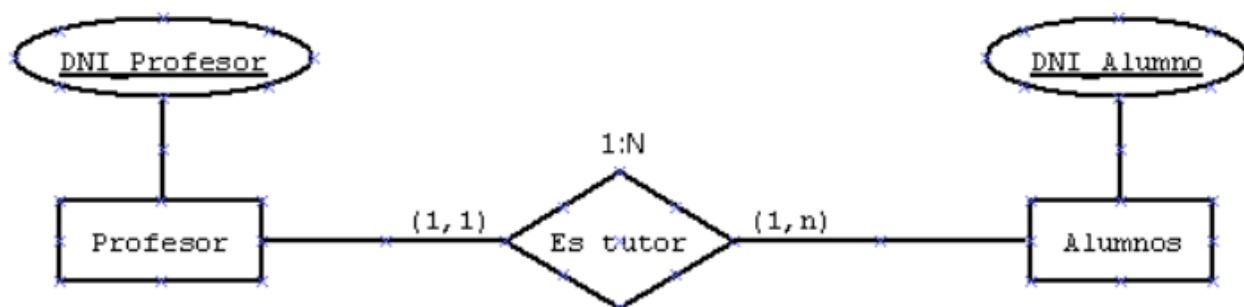
#### 4.2.2 Transformación de relaciones E/R con cardinalidad 1:N

Las relaciones del modelo E/R con cardinalidad 1:N no se transformarán en una nueva tabla. Lo que se hará será añadir a la tabla generada a partir de la entidad que actúa como N, un nuevo atributo que será la clave principal de la tabla generada a partir de la entidad que actúa como 1, donde pasará a actuar como clave foránea. Si la relación 1:N tuviera algún atributo adicional, éste también sería añadido a la misma tabla.

— Las relaciones del modelo E/R con cardinalidad 1:N no crean tabla. Añadirán a la tabla que actúa con cardinalidad N como clave ajena, la clave principal de la entidad que actúa con cardinalidad 1. En esa misma tabla se añadirán los atributos de la relación 1:N, si los hay.

Veamos un ejemplo:





La transformación de este fragmento dará lugar a dos tablas, la tabla Profesor y la tabla Alumnos, quedando de la siguiente forma:

Profesor ( DNI\_Profesor, Nombre, Fecha\_Incorporación)

CP: DNI\_Profesor

Alumnos( DNI\_Alumno, Fecha\_Nacimiento\* , Correo, Teléfono, DNI\_Profesor\_Tutor)

CP: DNI\_Alumno

CAj: DNI\_Profesor\_Tutor → Profesor

VNN: DNI\_Profesor\_Tutor

VNN: Fecha\_Nacimiento

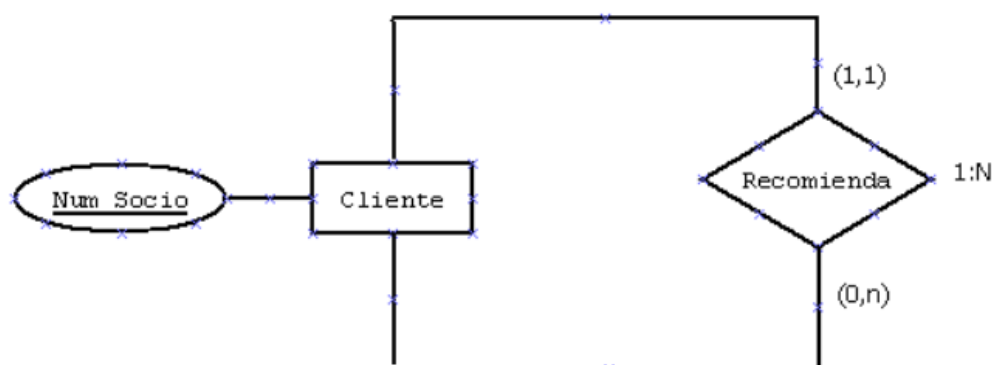
Podemos apreciar que en la transformación, el DNI\_Profesor ha sido añadido a la tabla Alumnos con el nombre DNI\_Profesor\_Tutor (se ha cambiado el nombre para que sea más significativo lo que representa ese campo) y se ha marcado como clave foránea.

Esta transformación nos permitirá en todo momento conocer los alumnos tutelados a partir de un profesor, para ello bastará con ver el DNI del profesor y buscar todos los alumnos que tengan ese valor en el atributo DNI\_Profesor\_Tutor. Por el contrario si queremos saber quien es el tutor de un alumno, bastará con tomar el valor del campo DNI\_Profesor\_Tutor y ver a qué profesor pertenece en la tabla Profesor.

Para que haya un Alumno tiene que haber un Profesor entonces la clave ajena para que se valor mínimo 1, debemos por VNN la clave Ajena.

#### 4.2.3 Transformación de relaciones E/R con cardinalidad 1:N reflexivas

Supongamos que estamos modelando la pertenencia a un club exclusivo en el que los clientes solo pueden ser admitidos si vienen recomendados por otros clientes. Tendríamos algo así:



Pues bien, la transformación será la misma, pero ahora el lugar donde se debe añadir el nuevo atributo será en la misma tabla generada a partir de la entidad (lógico, pues sólo disponemos de una entidad). Por ello cambiaremos el nombre del atributo para que sepamos a qué corresponde.

Cliente (Num\_Socio, Nombre, Recomendado\_Por\_Num\_Socio)

CP: Num\_socio

Clave Ajena: Recomendado\_por\_Num\_Socio → Cliente

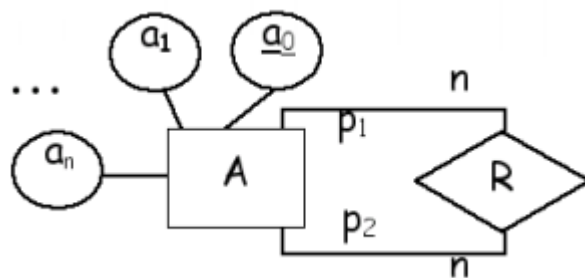
VNN: Recomendado\_por\_Num\_Socio

Podemos ver que el nuevo campo al que hemos llamado Recomendado\_Por\_Num\_Socio representará al socio que realizó la recomendación de ese cliente.

Además, también vendría bien pensar qué ocurriría si en lugar de tener una participación (1,1) como hemos puesto, fuese de (0,1). Eso querría decir que podría haber socios que no han sido recomendados por otros socios, y aunque la transformación sería idéntica, deberíamos tener en cuenta que el nuevo campo podría admitir valores nulos en aquellos cliente que no fueron recomendados por nadie (piensa en ello y ponte varios ejemplos para que veas lo que puede influir la participación a la hora de decidir que campos pueden ser NULOS o no).

#### 4.2.4 Transformación de relaciones E/R con cardinalidad N:N reflexivas

Dada la relación siguiente que representa este tipo de relación unaria:



Las relaciones equivalentes a esta estructura son:

$A(a_0, a_1, \dots, a_n)$

Clave Primaria:  $a_0$

$R(a_0\_p1, a_0\_p2)$

Clave Primaria:  $\{a_0\_p1, a_0\_p2\}$

Clave Ajena:  $a_0\_p1$  hace referencia a A

Clave Ajena:  $a_0\_p2$  hace referencia a A

En este caso, debido a que la relación reflexiva R tiene cardinalidad máxima M:M, es necesario utilizar una relación independiente para representarla. Nótese como en este caso hay en R dos claves ajenas que se refieren a la misma relación. Si hubiera definida alguna restricción de existencia, también sería necesaria la inclusión de una restricción de integridad. La presencia de restricciones de existencia en las relaciones reflexivas es, sin embargo, poco frecuente.

#### 4.2.5 Transformación de relaciones E/R con cardinalidad 1:1

Cuando tenemos una relación con una cardinalidad 1:1 tenemos varias opciones para obtener su correspondencia en el modelo relacional.

##### Primera opción:

La primera opción es unir las tablas generadas a partir de las entidades que participan en el relación en una sola tabla, sobre todo cuando ambas tienen la misma clave primaria. Además si la relación (E/R) tiene atributos, estos se añadirán a la nueva tabla resultante de la unión de las anteriores.

Si las tablas a unir tienen claves primarias diferentes seleccionaremos aquella que al unirse no pueda tomar valores nulos. Si ninguna tomase valores nulos para alguna ocurrencia entonces ambas se convertirán en claves candidatas y podremos elegir como primaria la que más nos interese en función del contexto de nuestra base de datos.

**Segunda opción:**

La segunda opción es mantener las tablas generadas a partir de las entidades y que la clave principal de una de ellas se añada como clave foránea en la otra tabla.

∞ Debemos tener en cuenta que si la participación es (0,1) esto puede provocar la aparición de valores nulos en el atributo para algunas tuplas.

**Tercera opción:**

La tercera opción es mantener las tablas generadas a partir de las entidades y colocar cada una de las claves principales como claves foráneas en la otra tabla. Esto conlleva introducir una redundancia en nuestro modelo, pero algunas veces esta redundancia puede ser de mucha utilidad para simplificar determinadas consultas.

Lo mejor es aplicar la primera opción siempre que se pueda y no aparezcan muchos campos nulos al incorporar los campos de una tabla a la otra. En caso de que el número de nulos que pueda aparecer sea elevado, es mejor aplicar la segunda opción.

Por último, siempre que esté debidamente justificado, bien por simplicidad en las operaciones de consulta o bien por mejoras de rendimiento, aplicaríamos la tercera introduciendo una redundancia controlada en nuestro modelo.

Veamos un ejemplo:

Supongamos que tenemos un modelo E/R que representa parte de la administración pública de nuestra comunidad en la que se representan los altos cargos de la administración y están relacionados con los coches que tienen asignados para sus desplazamientos. (Suponemos que los altos cargos que tienen coche oficial solo tienen uno y que no hay ningún coche oficial que no se haya asignado a un alto cargo).



Veamos las posibles soluciones:

**Primera opción:**

Supongamos que a partir de las entidades obtenemos las siguientes tablas (los campos son inventados pues no aparecen en el modelo E/R del que partimos).

Alto\_Cargo (DNI, Nombre, Puesto, Fecha\_Posesión)

CP: DNI

Coche\_Oficial (Matrícula, Marca, Modelo, Color, Fecha\_Adquisición, Fecha\_Matriculación)

CP: Matrícula.

La primera opción une en una sola tabla las dos anteriores de la siguiente forma:

Cargo\_Coche (DNI, Nombre, Puesto, Fecha\_Posesión, Matrícula, Marca, Modelo, Color, Fecha\_Adquisición, Fecha\_Matriculación)

Creamos una nueva tabla que une las dos anteriores y elegimos como clave primaria el DNI, por ser la única de las dos claves que no contendrá valores nulos, pues al tener una participación (0,1) podemos tener Altos Cargos que no tengan un coche asignado, en esos casos los campos correspondientes al Coche Oficial tendrán valor NULO. Esta solución estaría bien siempre que el número de altos cargos que no tienen coche oficial sea pequeño y la mayoría sí lo tengan.

### Segunda opción:

Se trata de colocar la clave primaria de una de las tablas en la otra como clave ajena. Para nuestro ejemplo ¿Cuál utilizaremos? Si las participaciones fuesen (1,1) en ambos casos, sería indiferente utilizar cualquiera de las dos, nos guiaríamos por el contexto. Pero al tener una participación con (0,1) eso nos quiere decir que hay Altos Cargos que no tienen coche oficial, luego si colocásemos el campo Matrícula en la tabla Alto\_Cargo tendríamos ocurrencias en las que ese atributo tomaría valores NULOS (en todas las tuplas correspondientes a los cargos que no tienen asignado un coche), por ello es recomendable hacer lo contrario y colocar el atributo DNI como clave foránea en la tabla Coche\_Oficial que representará quien tiene asignado ese coche.

Alto\_Cargo (DNI, Nombre, Puesto, Fecha\_Posesión)

CP: DNI

Coche\_Oficial (Matrícula, Marca, Modelo, Color, Fecha\_Adquisición, Fecha\_Matriculación, DNI)

CP: Matrícula

CAJ: DNI → Alto\_Cargo

En este caso particular el elegir una u otra opción dependería del número de altos cargos que tienen coche y los que no lo tienen.

#### 4.2.6 Transformación de relaciones E/R de grado n

Si en nuestro modelo E/R aparecen relaciones de grado superior a 2 será preciso realizar un estudio detallado de la cardinalidad y la participación de cada una de las entidades que se relacionan.

Si todas las participaciones de las entidades en la relación son de tipo (1,n), la relación E/R se transformará en una nueva tabla cuya clave primaria será la unión de las claves primarias de las tres entidades que están relacionadas. Además en esa tabla se podrán integrar todos los atributos de la relación.

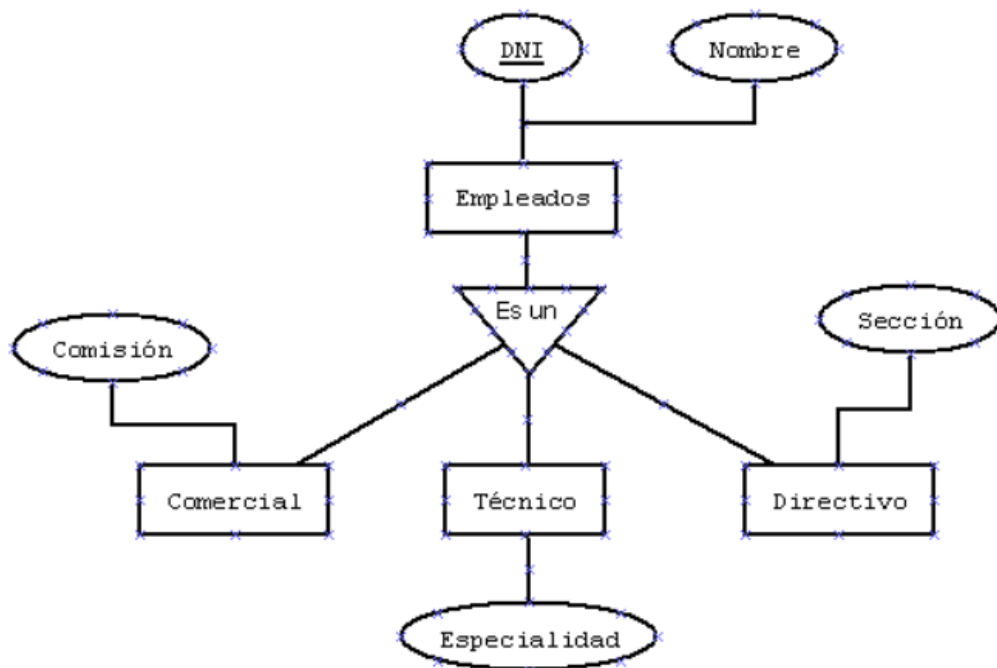
Por otro lado, en cuanto podamos tener participaciones de tipo (0,n), (0,1) o (1,1) tendremos que analizar de forma detallada el comportamiento de la relación antes de transformarlo al modelo relacional. Pensad que si una participación tiene la posibilidad de ser cero, eso implicaría tener valores nulos y las claves primaria no pueden tener valores nulos.

#### 4.2.7 Transformación de Especializaciones

Cuando tenemos especializaciones podemos optar por cuatro posibilidades de transformación. Cada tipo de transformación se adaptará mejor a cada tipo de especialización (Recuerda: total o parcial y exclusiva o inclusiva).

- Podemos optar por crear una tabla para el Supertipo y otra tabla para cada uno de los Subtipos encontrados. En este caso la clave primaria de la tabla obtenida a partir del Supertipo será añadida a cada una de las tablas obtenidas a partir de los Subtipos.
- Otra opción sería crear una tabla para cada Subtipo incorporando en cada una todos los atributos del Supertipo. NO se crearía la tabla del Supertipo pues sus campos se han incorporado en cada una de las tablas de los Subtipos. Esto no sería válido si no estamos en una especialización total ¿estás de acuerdo?.
- La tercera posibilidad es crear una sola tabla para el Supertipo y añadir en ella todos los atributos de los subtipos añadiendo además un campo (por ejemplo tipo) que nos permita diferenciar a qué tipo de especialización corresponde esa tupla. ¿Sería válido este modelo para especializaciones de tipo inclusivo?
- La cuarta opción sería crear una sola tabla para el Supertipo y añadir un campo para cada una de las especializaciones para indicar que tipos de perfiles cumple esa tupla, de esta forma se podrán representar especializaciones inclusivas.

Veamos un ejemplo:



Primera opción. Una tabla para el Supertipo y una para cada Subtipo y todas con la misma clave primaria pero siendo clave ajena. Es la más recomendable ya que indica más claramente que es una especialización.

Empleados (DNI, Nombre)

CP: DNI

Comercial (DNI, Comisión)

CP: DNI

C. Ajena: DNI → Empleados

Técnico (DNI, Especialidad)

CP: DNI

C. Ajena: DNI → Empleados

Directivo (DNI, Sección)

CP: DNI

C. Ajena: DNI → Empleados

Este diseño es correcto para

Segunda opción. Sin tabla para el Supertipo e incorporando sus atributos a las tablas de los Subtipos.

Comercial (DNI, Nombre, Comisión)

Técnico (DNI, Nombre, Especialidad)

Directivo (DNI, Nombre, Sección)

Tercera Opción. Todo en una tabla con un campo añadido para distinguir el tipo de especialización.

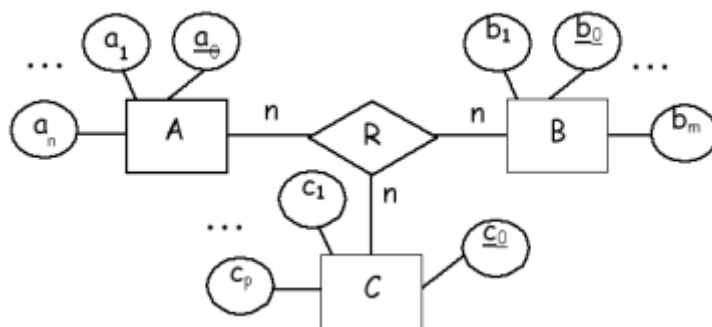
Empleados (DNI, Nombre, tipo, Comisión, Especialidad, Sección)

Cuarta Opción. Todo en una tabla con un campo añadido por cada especialización.

Empleados (DNI, Nombre, Es\_Comercial, Comisión, Es\_Técnico, Especialidad, Es\_Directivo, Sección)

#### 4.2.8 Relaciones ternarias

Relacion ternaria M:M:M sin restricciones de existencia



Las relaciones equivalentes a esta estructura son:

$A(a_0, a_1, \dots, a_n)$

Clave Primaria:  $a_0$

$B(b_0, b_1, \dots, b_m)$

Clave Primaria:  $b_0$



$C(c_0, c_1, \dots, c_p)$

Clave Primaria:  $c_0$

$R(a_0, b_0, c_0)$

Clave Primaria:  $\{a_0, b_0, c_0\}$

Clave Ajena:  $a_0$  hace referencia a A

Clave Ajena:  $b_0$  hace referencia a B

Clave Ajena:  $c_0$  hace referencia a C

Para este tipo de cardinalidad máxima, la clave primaria de la relación R debe ser el atributo compuesto por las tres claves primarias de las tres relaciones que representan a las tres entidades. (Nótese que la definición de cualquier otra clave primaria da lugar a una mala representación de la relación R)

## 5. BIBLIOGRAFÍA

- [CAS17] Bases de Datos. Iván López, M.<sup>a</sup> Jesús Castellano. John Ospino. 2017 Editorial. Garceta. 2da edición. ISBN: 978-8415452959