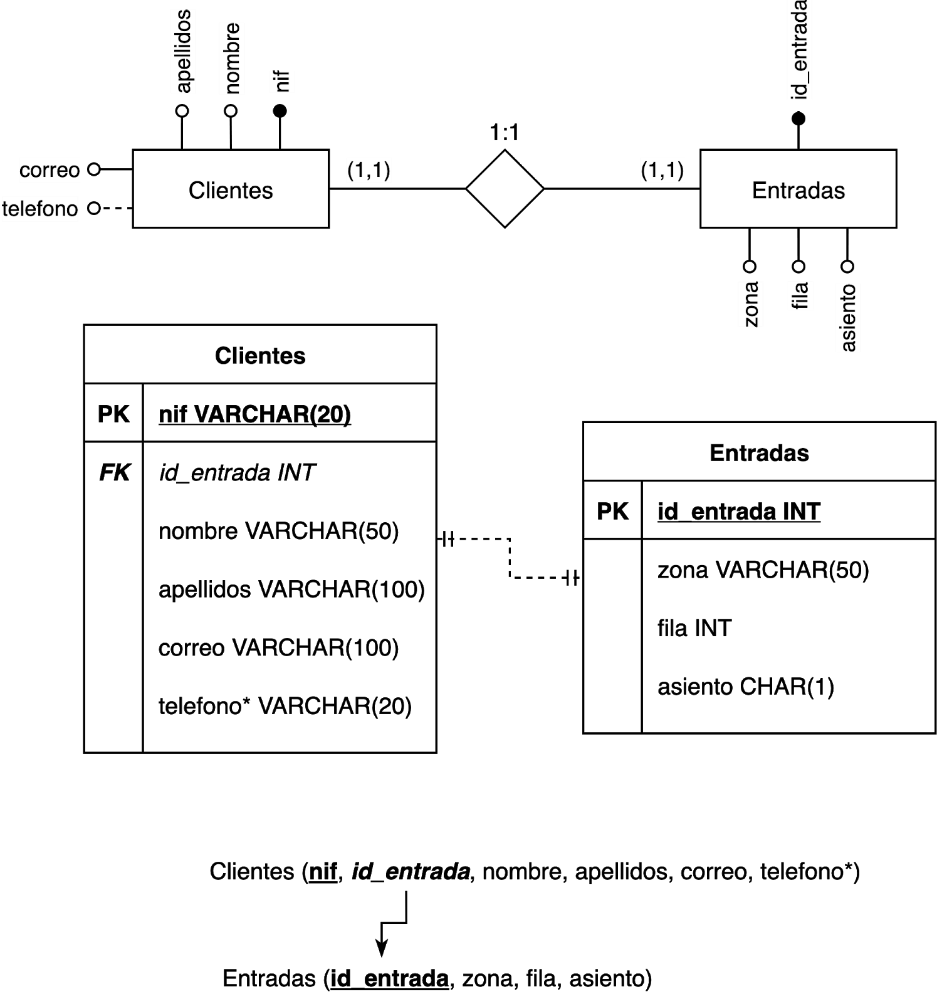
Soluciones Ejercicios propuestos

**1. Entradas unipersonales**

Tienda online para venta de entradas. Una persona solo puede comprar una entrada, que queda asignada a su NIF. Hay un stock máximo de entradas para evitar vender más entradas del aforo.

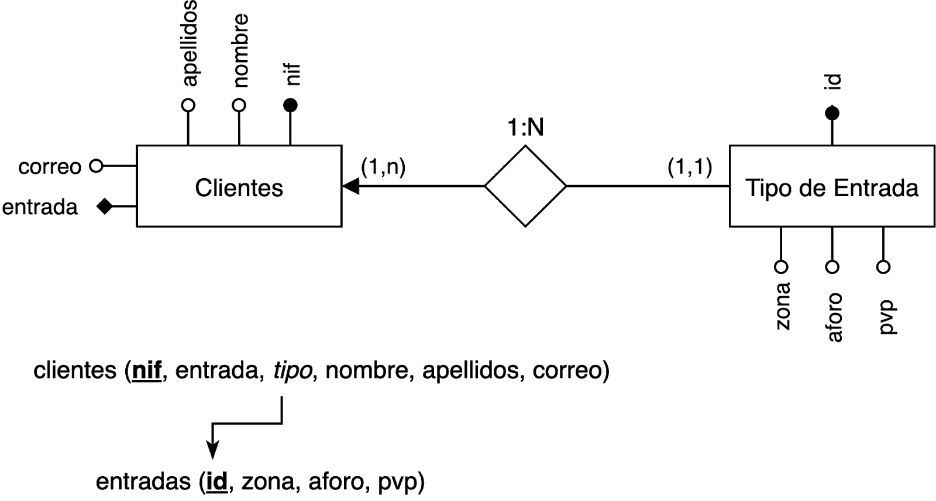
1.1 Un único tipo de entrada (todas las entradas son iguales: mismo precio y características)

[](https://app.diagrams.net/?page-id=v_R40HqxKa3s5sNUvY34&scale=auto#G1wgE3xaUUlWbZJwmSMBVIsTVVsQUXNPjx)

La FK id\_entrada debe debe ser UNIQUE,para que una entrada solo pueda estar asignada a un único cliente.

Alternativa: Llegaría una única tabla de clientes: se registra al cliente y se genera automáticamente su entrada. la PK puede ser o el NIF o el código de la entrada, el que no lo sea deberá ser un identificador alternativo (UNIQUE)

1.2 Varios tipos de entradas, cada uno con su stock máximo

[](https://app.diagrams.net/?page-id=v_R40HqxKa3s5sNUvY34&scale=auto#G1y5KrB3WbwzjuSbd7leAxdzmJH_abng4i)

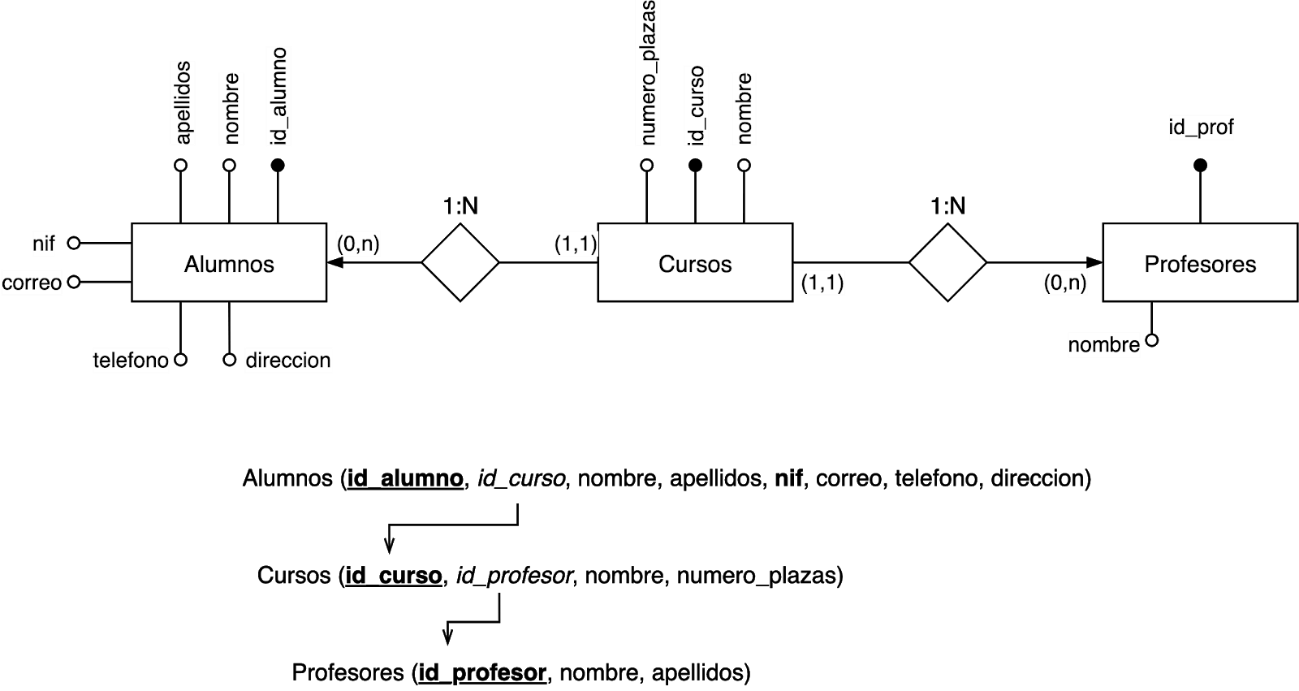
Nota: entrada será un atributo identificador alternativo (UNIQUE). Los representaremos con ese símbolo ya que este programa no tiene la representación que estamos usando en clase (el círculo solo con una mitad blanca y la otra negra).

También podría ser entrada la PK y nif UNIQUE (ya que si el NIF no se especificase como UNIQUE, un cliente podría comprar varias entradas registrándose varias veces).

[Implementación](https://drive.google.com/file/d/10Y2OozKb2L1GOrhFHtP0fqeC_B39ofzY)

**2. Academia restrictiva**

Una academia imparte cursos. Esta academia tiene la particularidad de que todos los cursos se imparten en el mismo horario y la asistencia a clase es obligatoria, con lo cual un alumno no puede matricularse en dos cursos simultáneamente. Además, cada curso tiene un límite de alumnos matriculados dependiente de la cantidad de puestos en cada aula. Se mantiene además un registro de profesores, que pueden serlo solo de uno de los cursos (por los mismos motivos antes explicados).

[](https://app.diagrams.net/?page-id=JjIfhJkld6MGYiJLekvE&scale=auto#G1xzniS3yUol3FS3IYC8WzpJuZLepF5zvV)

\* Se ha optado por considerar que un profesor puede tener sustitutos, por lo que la relación entre cursos y profesores es 1:N. También sería correcto considerarla 1:1, ciñendose al enunciado.

**3. Proveedores**

Tenemos que diseñar una base de datos sobre proveedores y disponemos de la siguiente información:

- De cada proveedor conocemos su nombre, dirección, ciudad, provincia y un código de proveedor que será único para cada uno de ellos.

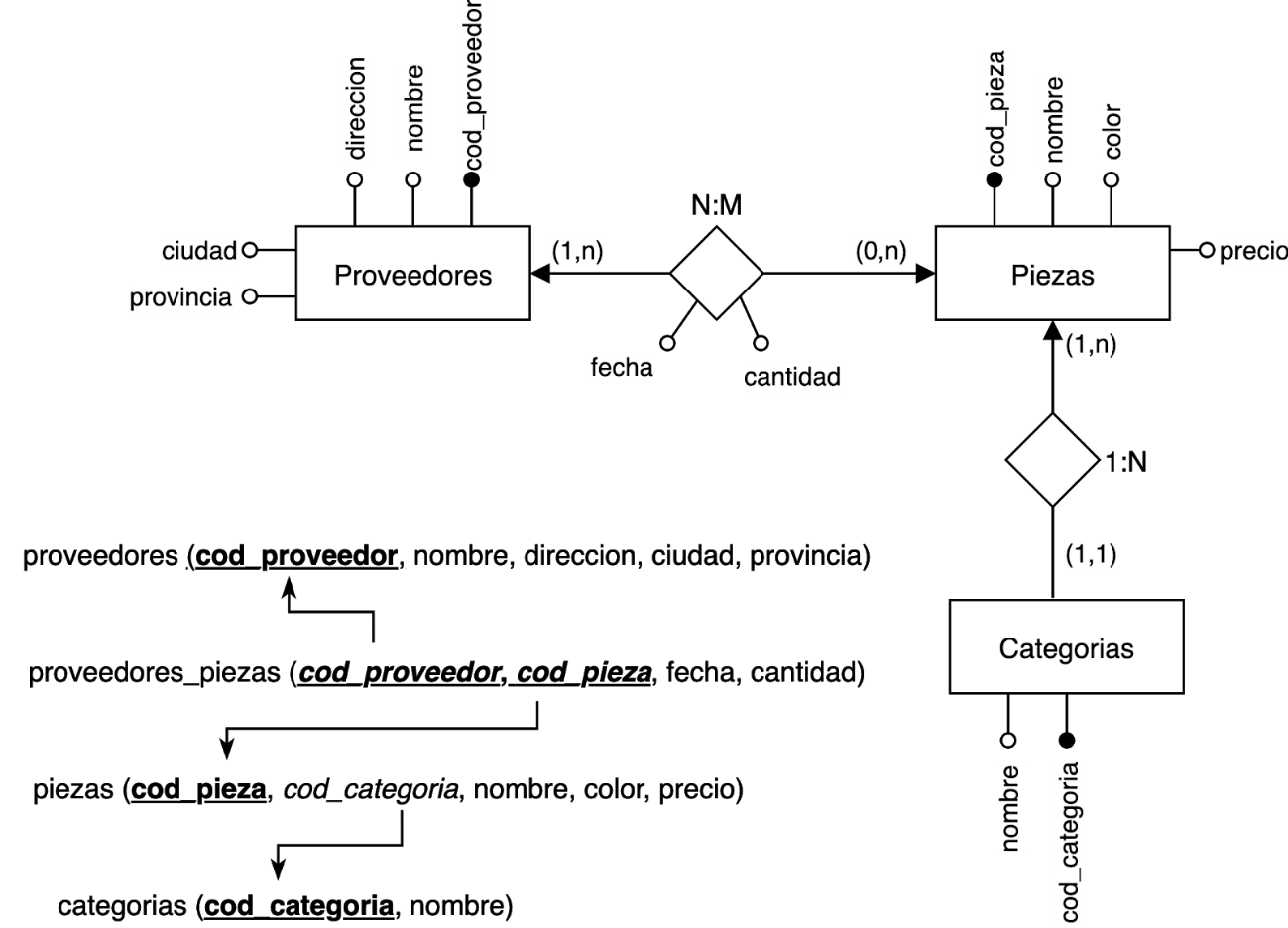
- Nos interesa llevar un control de las piezas que nos suministra cada proveedor. Es importante conocer la cantidad de las diferentes piezas que nos suministra y en qué fecha lo hace. Tenga en cuenta que un mismo proveedor nos puede suministrar una pieza con el mismo código en diferentes fechas. El diseño de la base de datos debe permitir almacenar un histórico con todas las fechas y las cantidades que nos ha proporcionado un proveedor.

- Una misma pieza puede ser suministrada por diferentes proveedores.

- De cada pieza conocemos un código que será único, nombre, color, precio y categoría.

- Pueden existir varias categorías y para cada categoría hay un nombre y un código de categoría único.

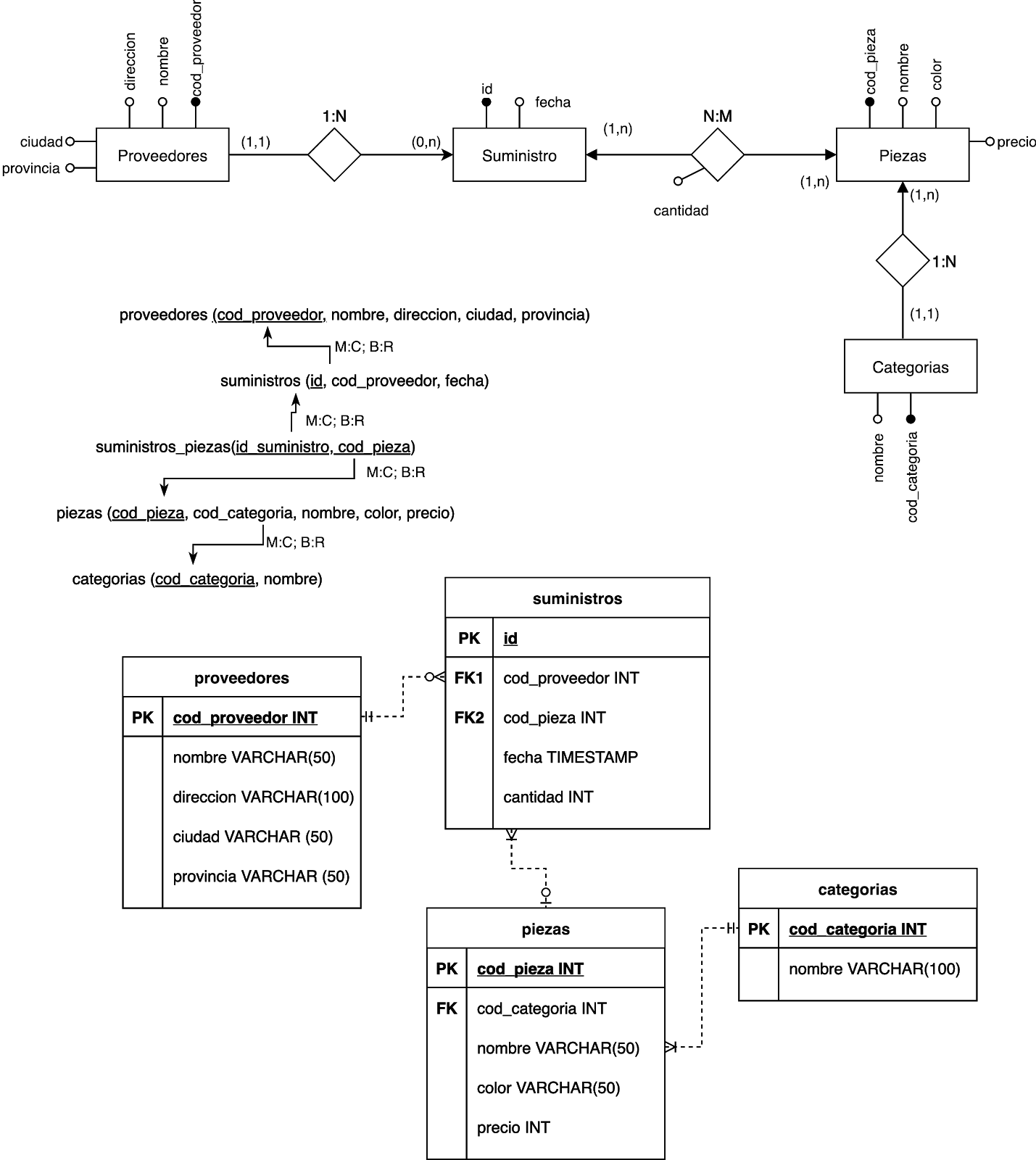
- Una pieza sólo puede pertenecer a una categoría.

Primera aproximación:  
  
[](https://app.diagrams.net/?page-id=nx1EQ1f-Dj29M_mlmttL&scale=auto#G18sqTrQd44jQwQ84ZRpEDAr208Otz0euo)

Haciéndolo de este modo, nos encontramos con que la PK de proveedores\_piezas impide que una misma pieza se pueda comprar a un mismo proveedor más de una vez, incumpliendo este supuesto del enunciado:

“Tenga en cuenta que un mismo proveedor nos puede suministrar una pieza con el mismo código en diferentes fechas.”  
  
Una primera solución sería incluir la fecha como parte de la PK. Al hacer esto estaríamos convirtiendo la interrelación N:M “proveedores\_piezas” en una entidad con doble identificación débil con respecto a proveedores y a piezas:  
  
[](https://app.diagrams.net/?page-id=nx1EQ1f-Dj29M_mlmttL&scale=auto#G1f0dtfmz9UNpkiUq9XhwDvHzF8V-vbuDw)

Vemos que la modificación en el grafo relacional (nivel lógico) es pequeña, pero el diagrama E-R a nivel conceptual cambia bastante. He añadido éste para que se entienda la coherencia entre ambos; sin embargo, es perfectamente razonable hacer el arreglo directamente a nivel lógico y simplemente señalarlo.  
  
Esta solución sería correcta y cumple con las especificaciones pedidas. Sin embargo, sería razonable detectar otra limitación que podríamos considerar: con la actual PK de la tabla de suministros no es posible que una misma pieza se compre a un mismo proveedor en la misma fecha. Añadiendo simplemente un id a la entidad suministros (que podría también llamarse “pedidos”, como se propone en la modificación del ejercicio 4) se soluciona ese problema y se simplifica mucho la solución final.

[](https://app.diagrams.net/?page-id=nx1EQ1f-Dj29M_mlmttL&scale=auto#G1PRRJGf44eZ0H98k8UTGXPT75rRFusvjt)

**5. Cadena editorial**

- La editorial tiene varias sucursales, con su domicilio, teléfono y un código de sucursal.

- Cada sucursal tiene varios empleados, de los cuales tendremos su nombre, apellidos, NIF y teléfono. Un empleado trabaja en una única sucursal.

- En cada sucursal se publican varias revistas, de las que almacenaremos su título, número de registro, periodicidad y tipo.

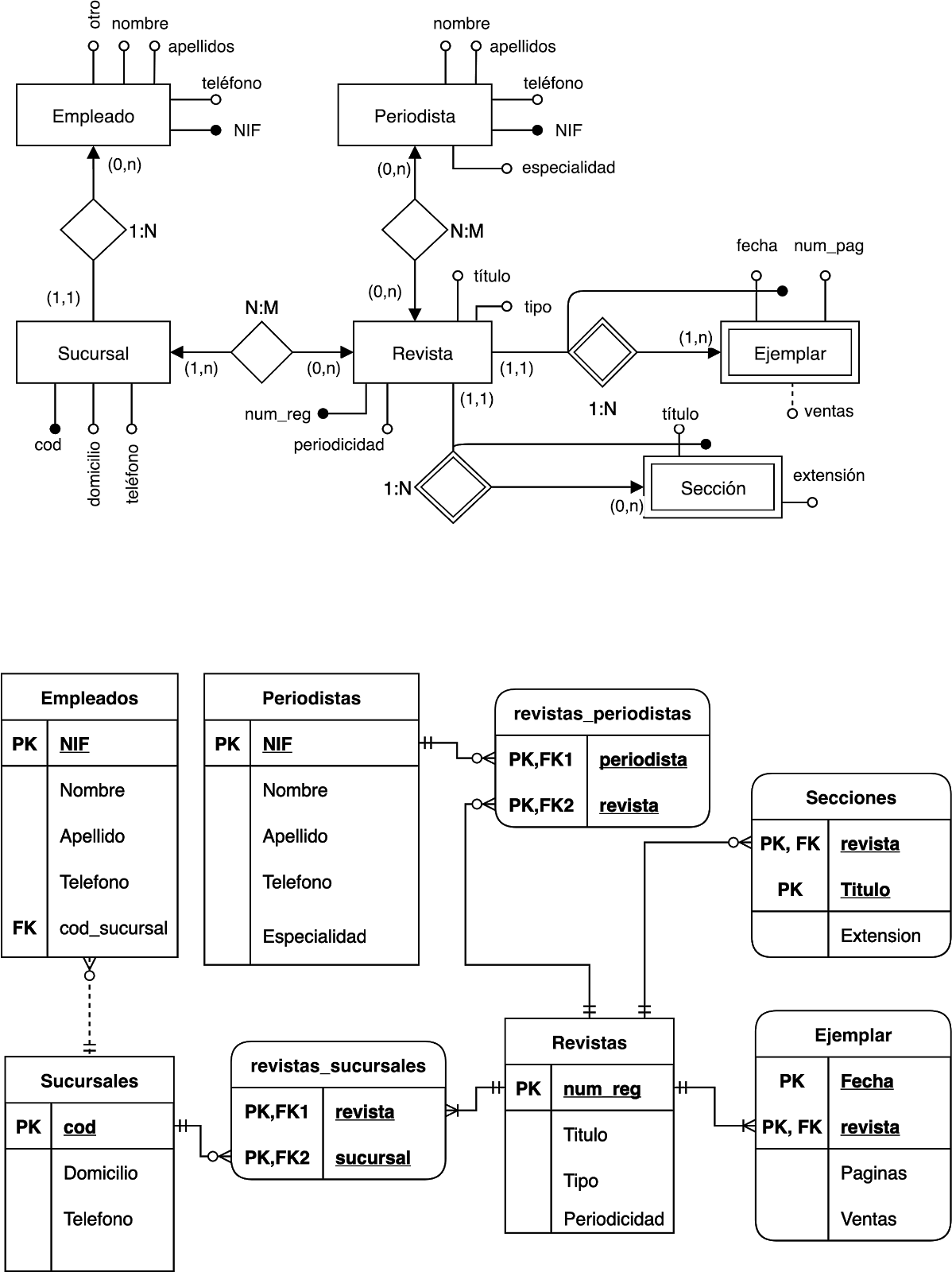
- Una revista puede ser publicada por varias sucursales.

- La editorial tiene periodistas (que no trabajan en las sucursales) que pueden escribir artículos para varias revistas. Almacenaremos los mismos datos que para los empleados, añadiendo su especialidad.

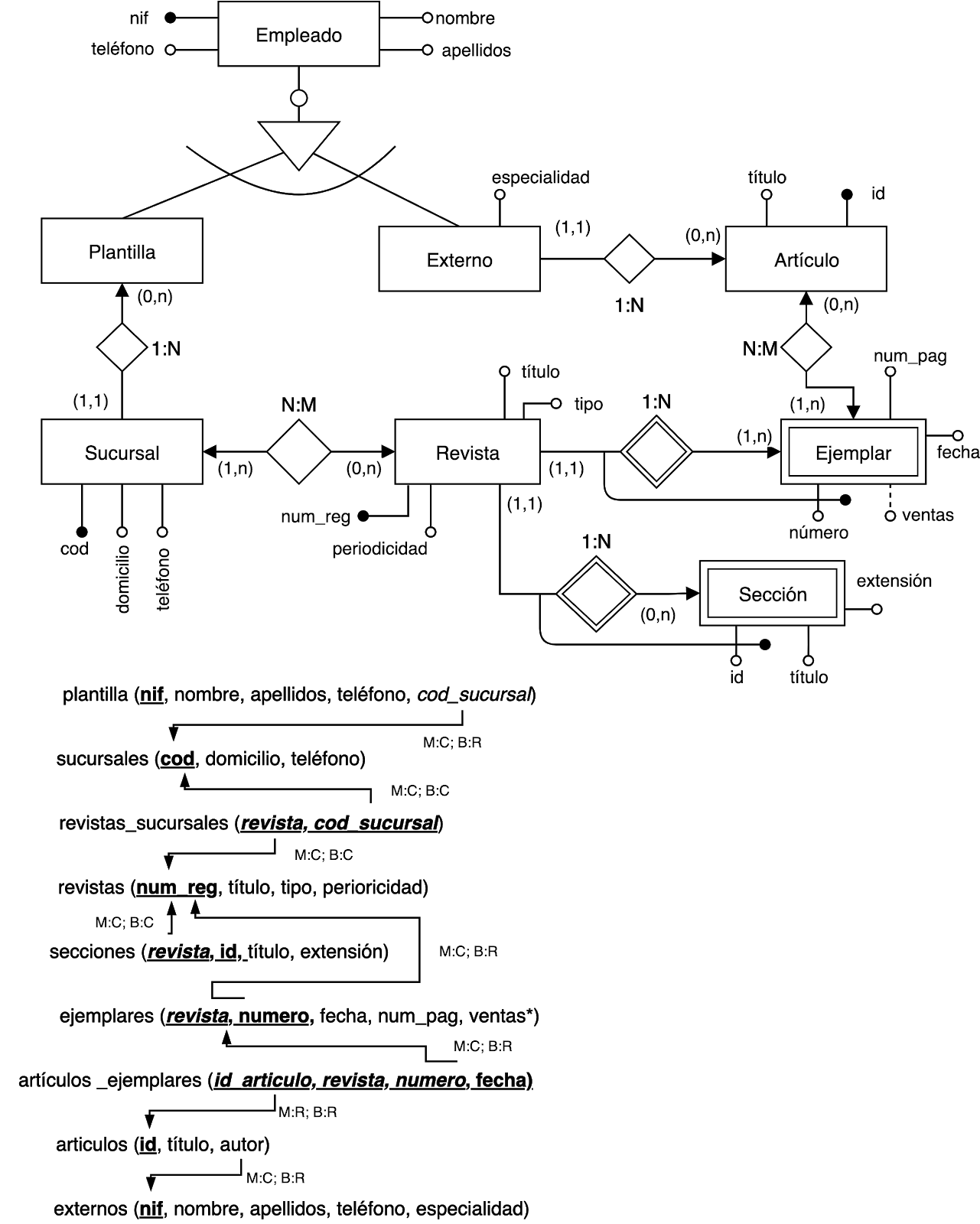
- También es necesario guardar las secciones fijas que tiene cada revista, que constan de un título y una extensión.

- Para cada revista, almacenaremos información de cada ejemplar, que incluirá la fecha, número de páginas y el número de ejemplares vendidos.

Una solución básica sería la siguiente:

[](https://app.diagrams.net/?page-id=_yk-xvrtyksXhZcdGm7a&scale=auto#G1dBEqL1AM9XwkP1USGuc81EchqVSsBVtc)

Otra solución:  
  
Decisión de diseño: Dado que se indica que los periodistas pueden escribir artículos para varias revistas, se puede considerar que la relación entre periodistas y revistas es a través de un registro de sus artículos publicados en estas. Además, es razonable asumir que un artículo se publica en un ejemplar en concreto de cada revista en la que se publica. Consideraremos por tanto la entidad “artículo” relacionada con uno o varios ejemplares (entendiendo que el mismo artículo puede ser publicado en varias revistas) y con un periodista (asumiremos que un artículo tiene un solo autor, aunque podría también considerarse que pueda tener más).  
  
Además, en esta solución se incluye una relación de generalización/especialización (que no guarda relación con la decisión de diseño antes tomada; podría haberse puesto en la solución anterior). La conversión a tablas de esta parte no cambiaría, por tratarse de una especialización total sin solapamiento en que solo hay interrelaciones de las entidades hijas.

[](https://app.diagrams.net/?page-id=_yk-xvrtyksXhZcdGm7a&scale=auto#G1IbgCEq6f44Q570h7SuNY7AFu2nN2VuX_)

**5c. Cadena editorial (variante)**

- La editorial tiene varias sucursales, con su domicilio, teléfono y un código de sucursal.

- Cada sucursal tiene varios empleados, de los cuales tendremos su nombre, apellidos, NIF y teléfono. Un empleado trabaja en una única sucursal. Algunos empleados son periodistas que pueden escribir artículos.

- En cada sucursal se publican varias revistas, de las que almacenaremos su título, número de registro, periodicidad y tipo.

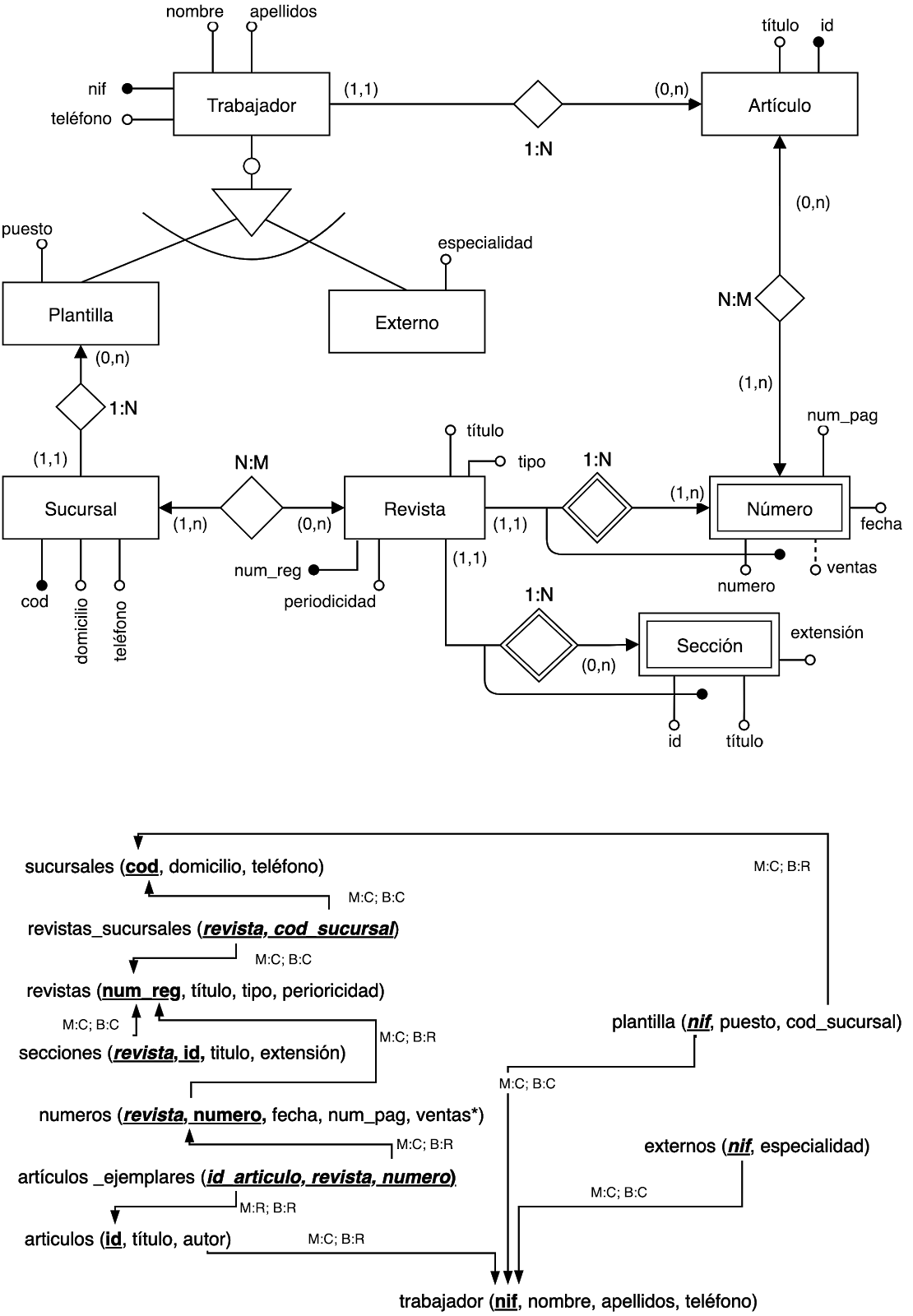
- Una revista puede ser publicada por varias sucursales.

- La editorial tiene un registro de periodistas que no están en plantilla que pueden escribir artículos para varias revistas. Almacenaremos los mismos datos que para los empleados, añadiendo su especialidad.

- Cada artículo se publica en un número concreto de cada revista en la que se publica, pero el mismo artículo puede publicarse en más de una revista.

- También es necesario guardar las secciones fijas que tiene cada revista, que constan de un título y una extensión.

- Para cada revista, almacenaremos información de cada número de la revista, que incluirá la fecha, número de páginas y el número de ejemplares vendidos.

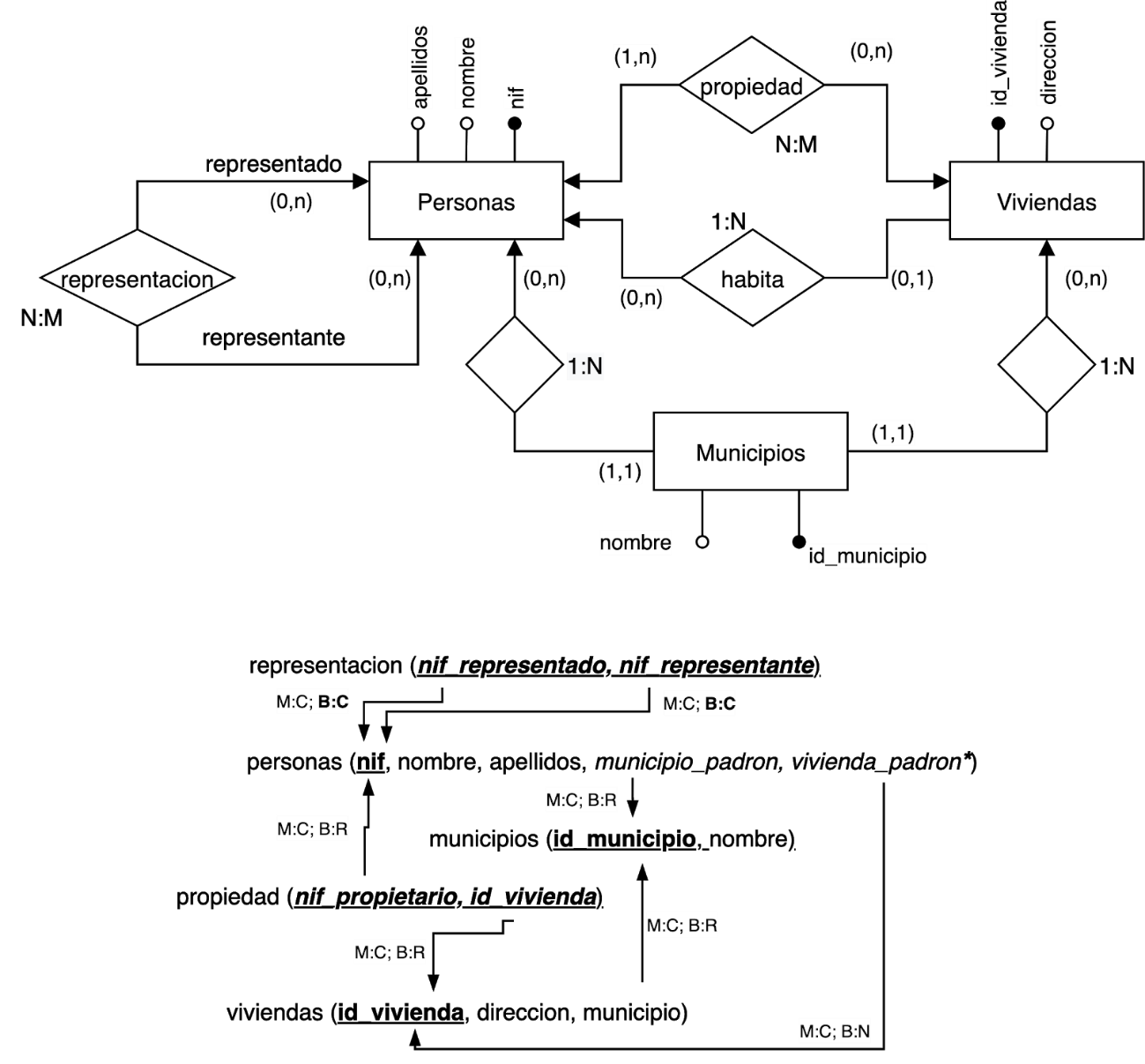
[](https://app.diagrams.net/?page-id=_yk-xvrtyksXhZcdGm7a&scale=auto#G1rGfZjJbAGsOU0t6qMpR5x12UGZ0hWhj9)

**6. Habitantes y municipios**

Supongamos el siguiente universo de discurso sobre municipios, viviendas y personas. Cada persona solo puede habitar una vivienda y estar empadrona en un municipio, pero puede ser propietaria de varias viviendas.  
Algunas personas pueden estar representadas por otras personas, que son las que realizan las gestiones administrativas relativas a sus viviendas.

Solución 1:  
- Una persona puede estar empadronada en un municipio pero habitar en un domicilio de otro.

- Se puede registrar a personas sin hogar (por tanto no habitan en ninguna vivienda).

- No se contempla que una persona no pueda representarse a sí misma.[](https://app.diagrams.net/?page-id=vVEcXn3-v2QgTs4LzTO1&scale=auto#G1eAEx539g4C8FZy-zh9WnVJNGhEmkR6Wt)

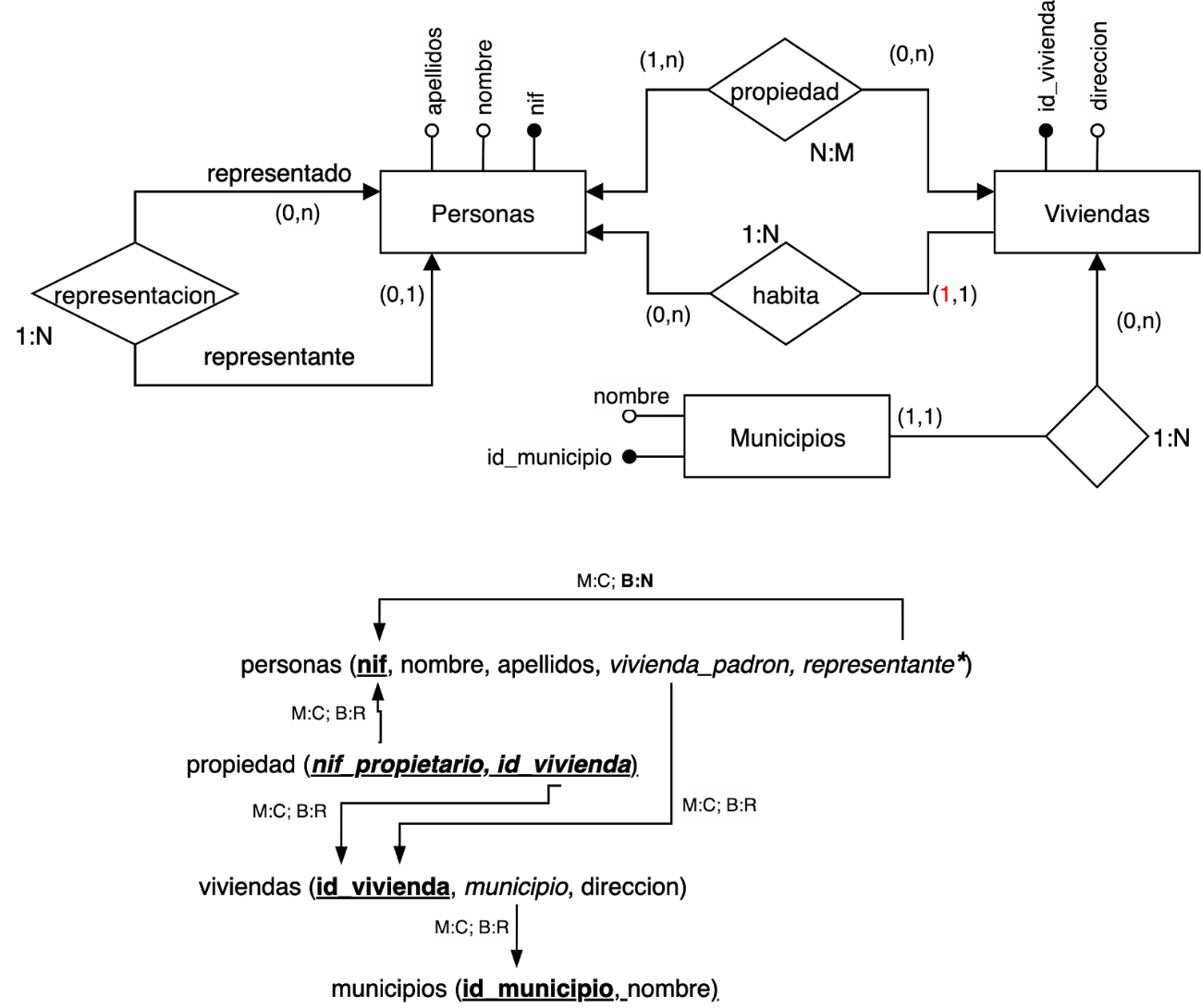
Otra solución:

- Se asume que una persona está empadronada en el municipio en que habita, por tanto, estará empadronada en el municipio en que está la vivienda en que habita.

- Se asume que toda persona habita en algún domicilio y solo en uno. [Sin esta condición, la anterior no tendría sentido].

- Una persona puede ser representada por otra pero solo por otra.

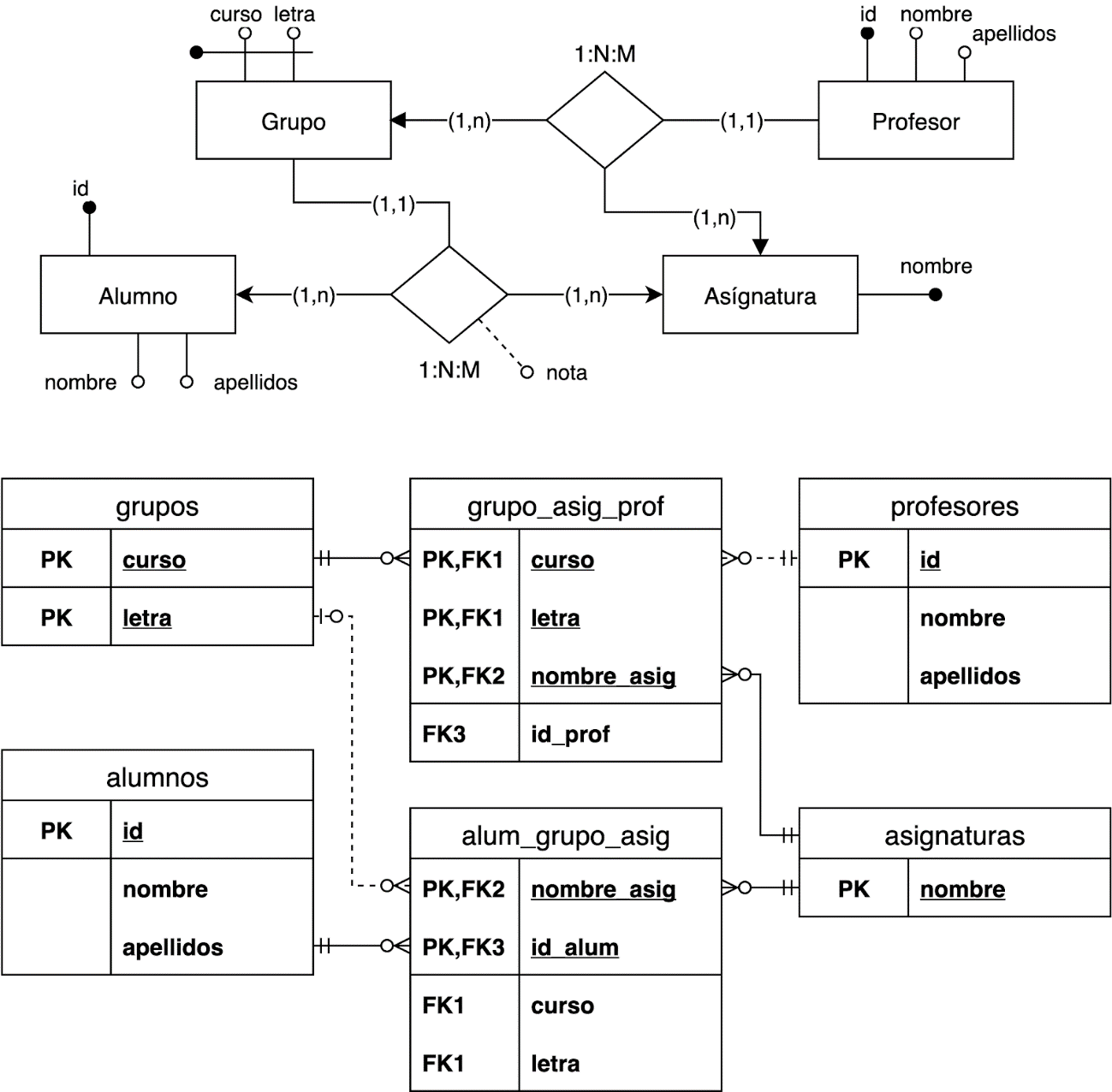
- No se contempla que una persona no pueda representarse a sí misma.

[](https://app.diagrams.net/?page-id=vVEcXn3-v2QgTs4LzTO1&scale=auto#G1kgkFw3rsLgZKKMiL1zF-PSGSsJbaMVrF)

# **Alumnos, asignaturas y profesores**

Diseñar una BD para una carrera universitaria que contenga información sobre los alumnos, las asignaturas y los profesores, teniendo en cuenta que:

* Una asignatura puede estar impartida por muchos profesores ya que puede existir más de un grupo de alumnos por curso (a cada grupo le podría impartir clase un profesor distinto).
* Un profesor puede dar clases de muchas asignaturas.
* Se necesita tener constancia de las asignaturas que imparten los profesores independientemente de si tienen o no alumnos matriculados en su grupo.
* No existen asignaturas distintas con el mismo nombre.
* Un alumno que está matriculado en alguna asignatura, está asignado a un grupo determinado.

[](https://www.draw.io/?page-id=e7e014a7-5840-1c2e-5031-d8a46d1fe8dd&scale=auto#G1Cl9qYKiRoKDbttJ3CtOKlEMFd8PRVi7b)

profesores (id, nombre, apellidos)

grupos (curso, letra)

grupo\_asig\_prof (curso, letra, nombre\_asig, id\_prof)

asignaturas (nombre)

alum\_grupo\_asig (id\_alumno, nombre\_asig, curso, letra)

alumnos (id, nombre, apellidos)

[Implementación](https://drive.google.com/file/d/1FP7U7Q0Lluca_58VQHhpOGpgHa63oDeF/view?usp=sharing)

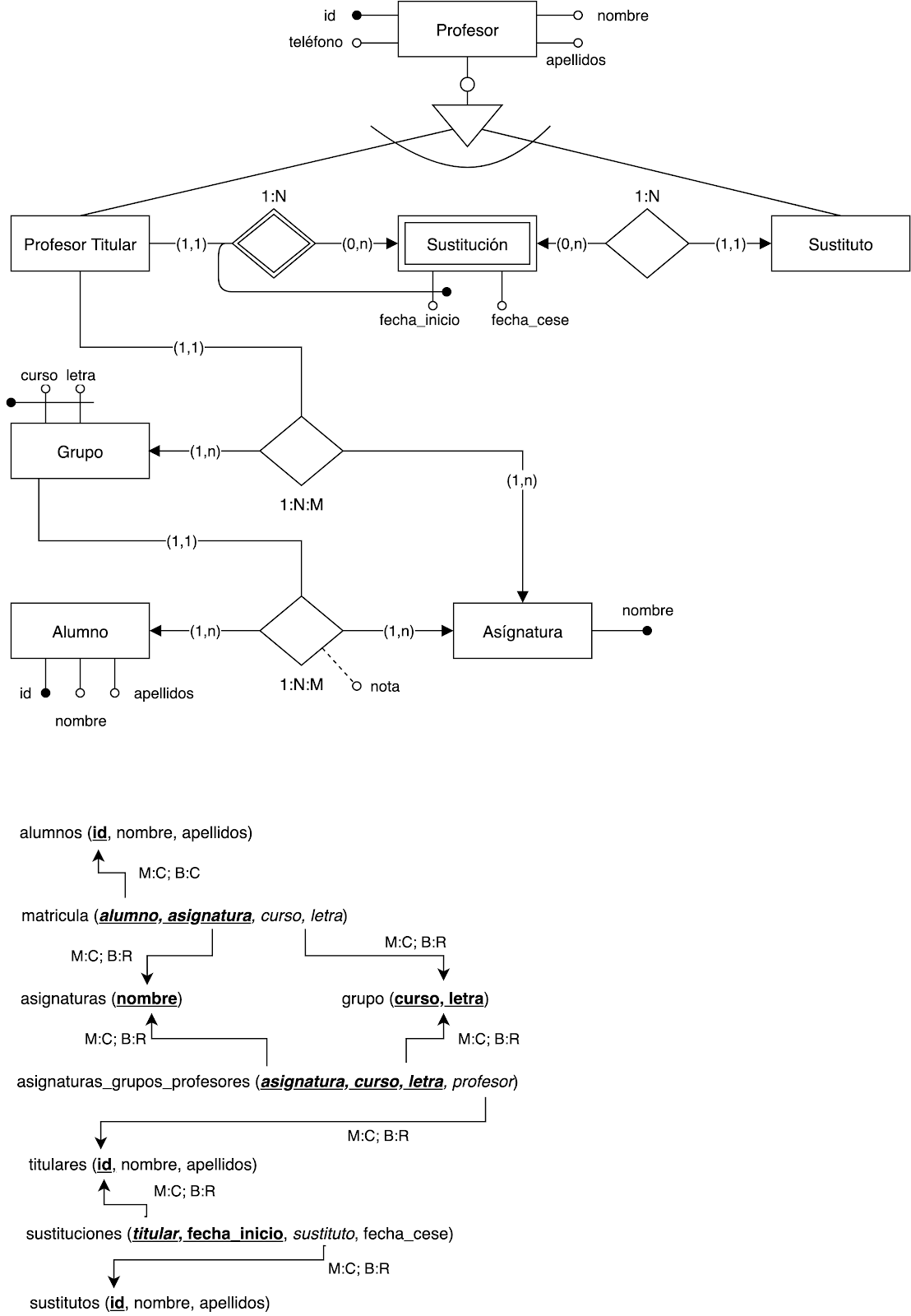
# **Alumnos, asignaturas y profesores (con sustituciones)**

Diseñar una BD para una carrera universitaria que contenga información sobre los alumnos, las asignaturas y los profesores, teniendo en cuenta que:

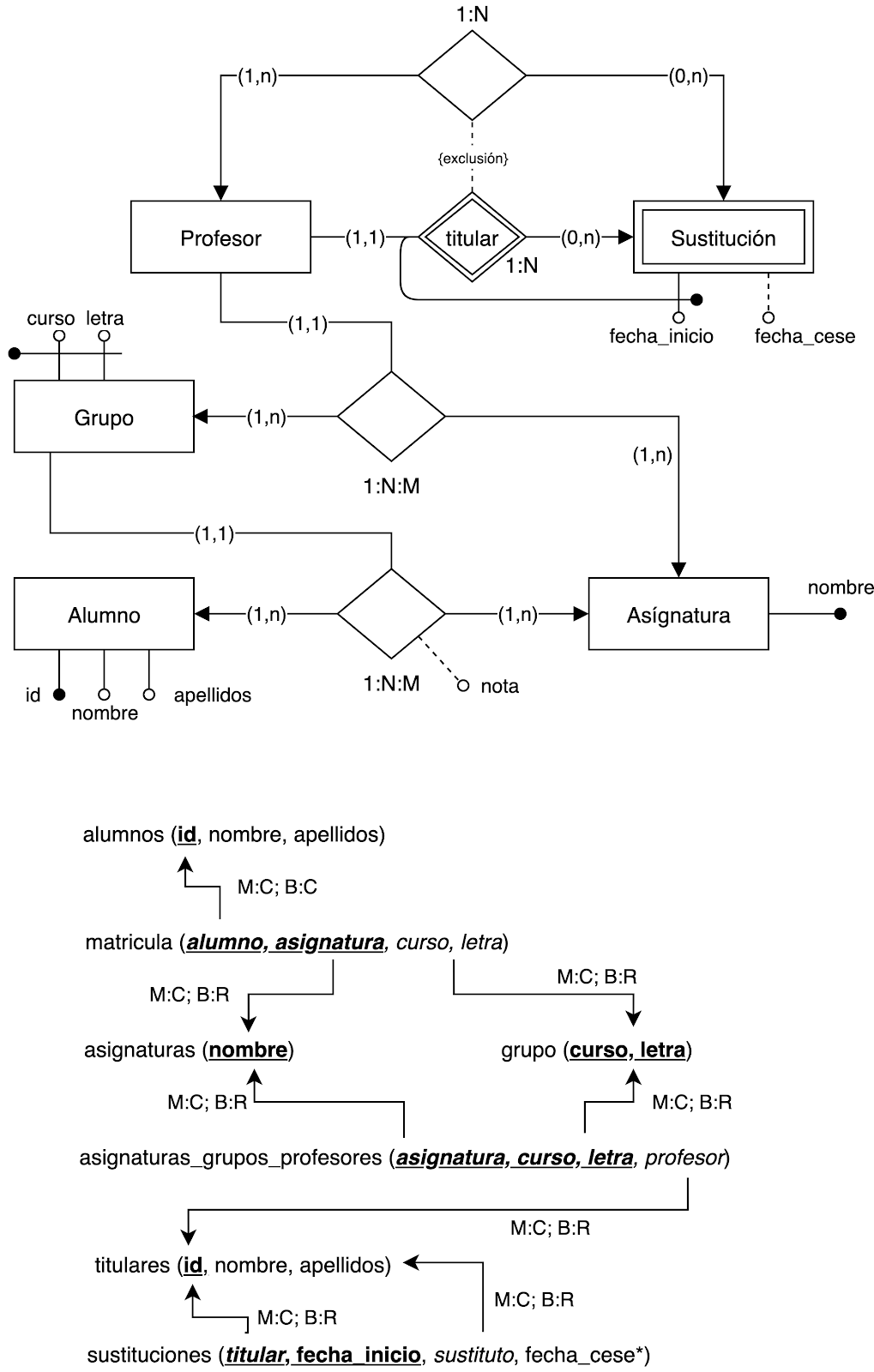
* Una asignatura puede ser impartida por muchos profesores ya que puede existir más de un grupo de alumnos por curso (a cada grupo le podría impartir clase un profesor distinto).
* Un profesor puede dar clases de muchas asignaturas.
* Se necesita tener constancia de las asignaturas que imparten los profesores independientemente de si tienen o no alumnos matriculados en su grupo.
* No existen asignaturas distintas con el mismo nombre.
* Un alumno que está matriculado en alguna asignatura, está asignado a un grupo determinado.

7b: Un profesor puede estar de baja en determinados periodos y ser sustituido por otro profesor sustituto. Un sustituto sustituye el horario completo del profesor sustituido.

Si asumimos que un profesor titular no puede actuar en calidad de sustituto.

[](https://app.diagrams.net/?page-id=e7e014a7-5840-1c2e-5031-d8a46d1fe8dd&scale=auto#G17Nss3kSqKZWP6mI_vMki1xDbvUdMcf8K)

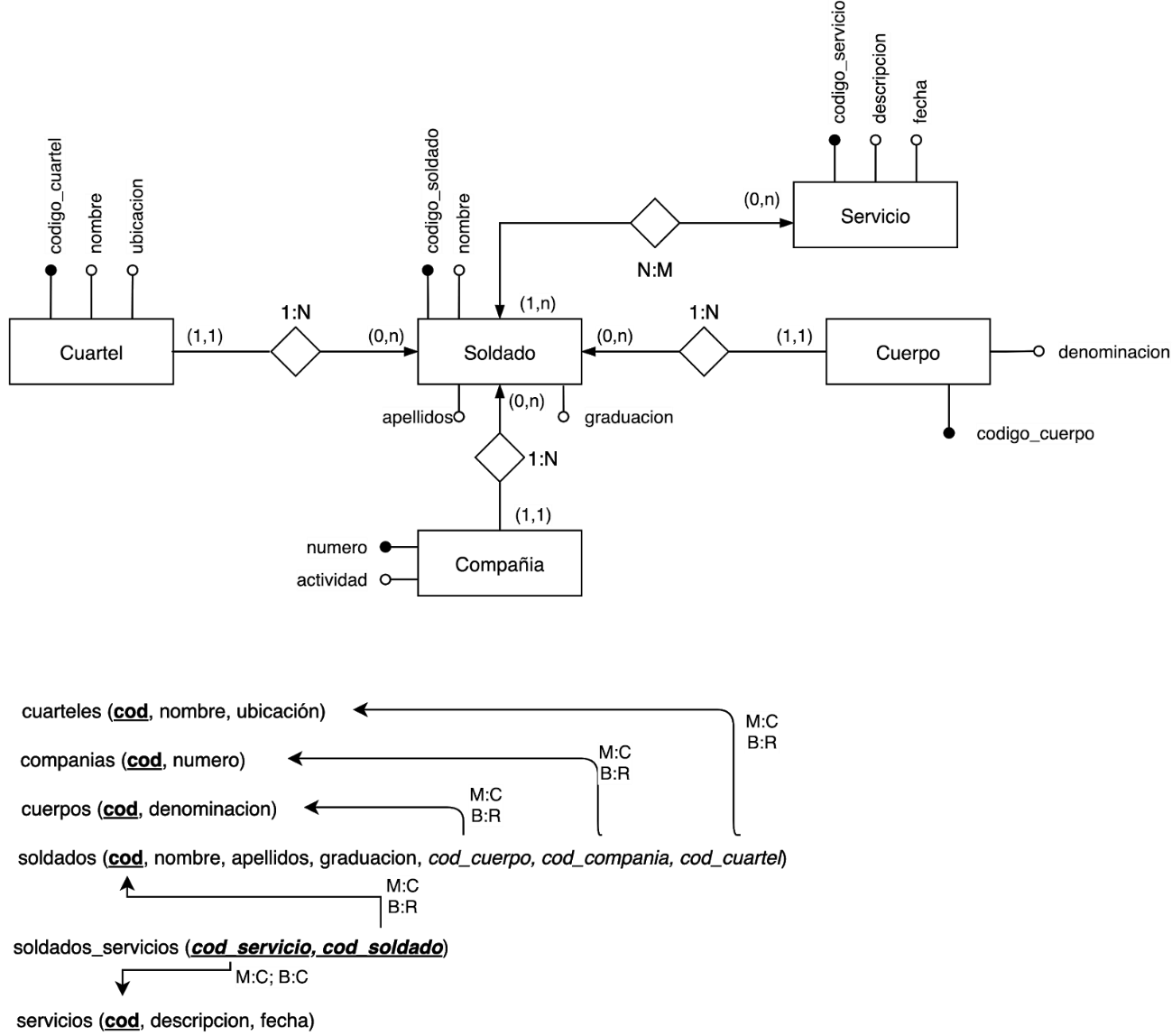
- La relación de generalización no sería necesaria. De hecho, convertirla en tabla no aportaría nada en este caso.  
  
Este modelo no permitiría sustituciones anidadas (que un sustituto sea a su vez sustituido por otro). Para solventar esto sería necesario modelarlo del siguiente modo:  
  
**7c.** Se quiere permitir que un profesor titular pueda sustituir a otro en su horario. Puede haber sustitutos de sustitutos.

[](https://app.diagrams.net/?page-id=e7e014a7-5840-1c2e-5031-d8a46d1fe8dd&scale=auto#G1DbKaxejo1srqkYQNiWwNSJ2hzK29rw1Q)

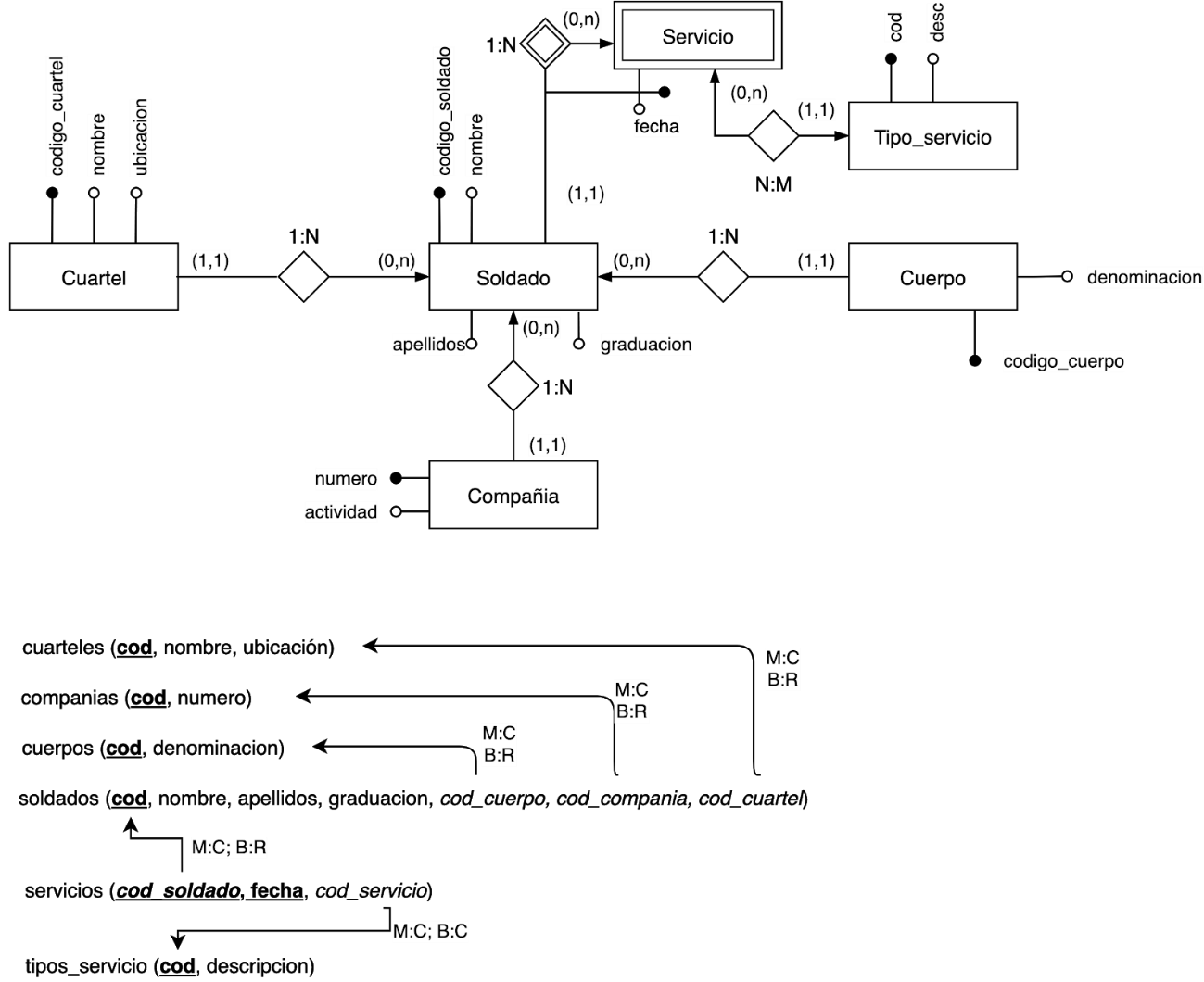
La relación de exclusividad indica que un profesor no puede sustituirse a sí mismo. Estas relaciones no pueden modelarse a nivel lógico, tendrán que ser implementadas a nivel físico.

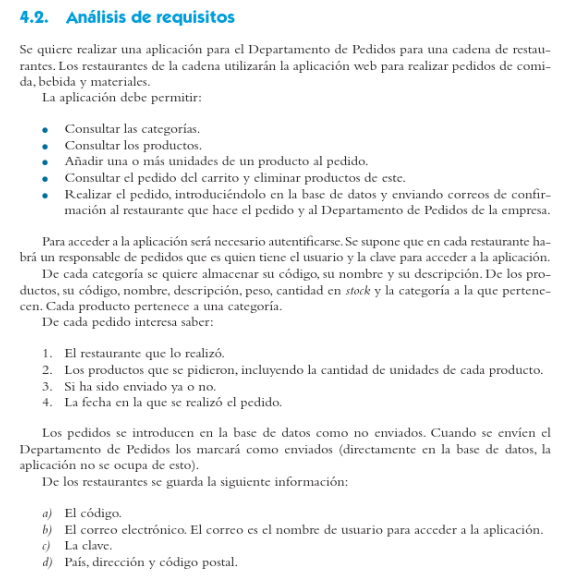
## 8. Ministerio de Defensa

El Ministerio de Defensa desea diseñar una Base de datos para llevar un cierto control de los soldados que realizan el servicio militar. Los datos significativos a tener en cuenta son: Un soldado se define por su código de soldado (único), su nombre y apellidos, y su graduación. Existen varios cuarteles, cada uno se define por su código de cuartel, nombre y ubicación. Hay que tener en cuenta que existen diferentes Cuerpos del Ejército (Infantería, Artillería, Armada, etc), y cada uno se define por un código de Cuerpo y denominación. Los soldados están agrupados en compañías, siendo significativa para cada una de éstas, el número de compañía y la actividad principal que realiza. Se desea controlar los servicios que realizan los soldados (guardias, instructores, cuarteleros, ...), y se definen por el código de servicio y descripción. Consideraciones de diseño: Un soldado pertenece a un único cuerpo y a una única compañía, durante todo el servicio militar. A una compañía pueden pertenecer soldados de diferentes cuerpos, no habiendo relación directa entre compañías y cuerpos. Los soldados de una misma compañía pueden estar destinados en diferentes cuarteles, es decir, una compañía puede estar ubicada en varios cuarteles, y en un cuartel puede haber varias compañías. Un soldado sólo esta en un cuartel. Un soldado realiza varios servicios a lo largo del servicio militar. Un mismo servicio puede ser realizado por más de un soldado (con independencia de la compañía), siendo significativa la fecha de realización.

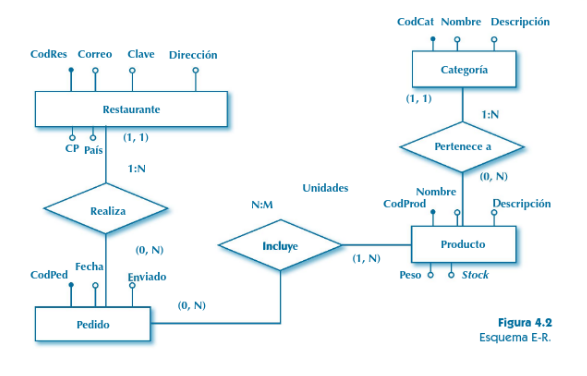
[](https://app.diagrams.net/?page-id=KpLDfOjxwIR8PgUKMCJR&scale=auto#G1wUp9JgSuLs2ZACxl4sr08nsizo7h5KAC)

Si se entiende por “servicio” el tipo de servicio, de modo que el mismo servicio pueda ser llevado a cabo por distintos soldados en distintas fechas, el atributo “fecha” pasaría a la interrelación N:M, o sea: a la tabla “soldados\_servicios (**cod\_servicio, cod\_soldado**, fecha)”. Indicando la fecha en que cada soldados lo realiza, pero esto impediría que un soldado realizase el mismo servicio varias veces en distintas fechas.

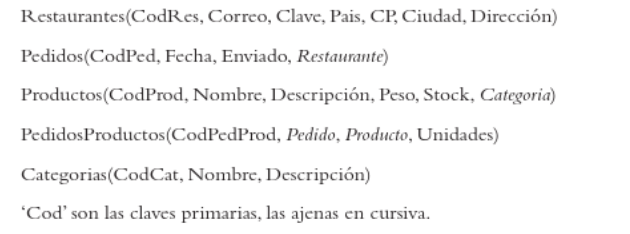
La ambigüedad en el concepto de servicio se podría solventar con esta versión, diferenciando el concepto de tipo de servicio y el del servicio concreto llevado a cabo (el enunciado con los atributos indicados no lo deja claro):  
  
[](https://app.diagrams.net/?page-id=KpLDfOjxwIR8PgUKMCJR&scale=auto#G1w8wZDE2_MFm0Iadmdd_PGxWDU8lRMoCR)La identificación con respecto a Soldado no sería necesaria (y, conceptualmente, parecería tener más sentido desde servicio, de haber alguna), pero permite a nivel lógico establecer una clave en la tabla servicios que evita que un soldado pueda realizar dos servicios concretos distintos en la misma fecha (esto no se pide, es una asunción que se sugirió en clase.  
Si no se quiere hacer esa restricción, bastaría con poner una PK *ad hoc* en servicio (un “id”) y dejarla como fuerte, sin mayor complicación.



Modelo Entidad-Relación:



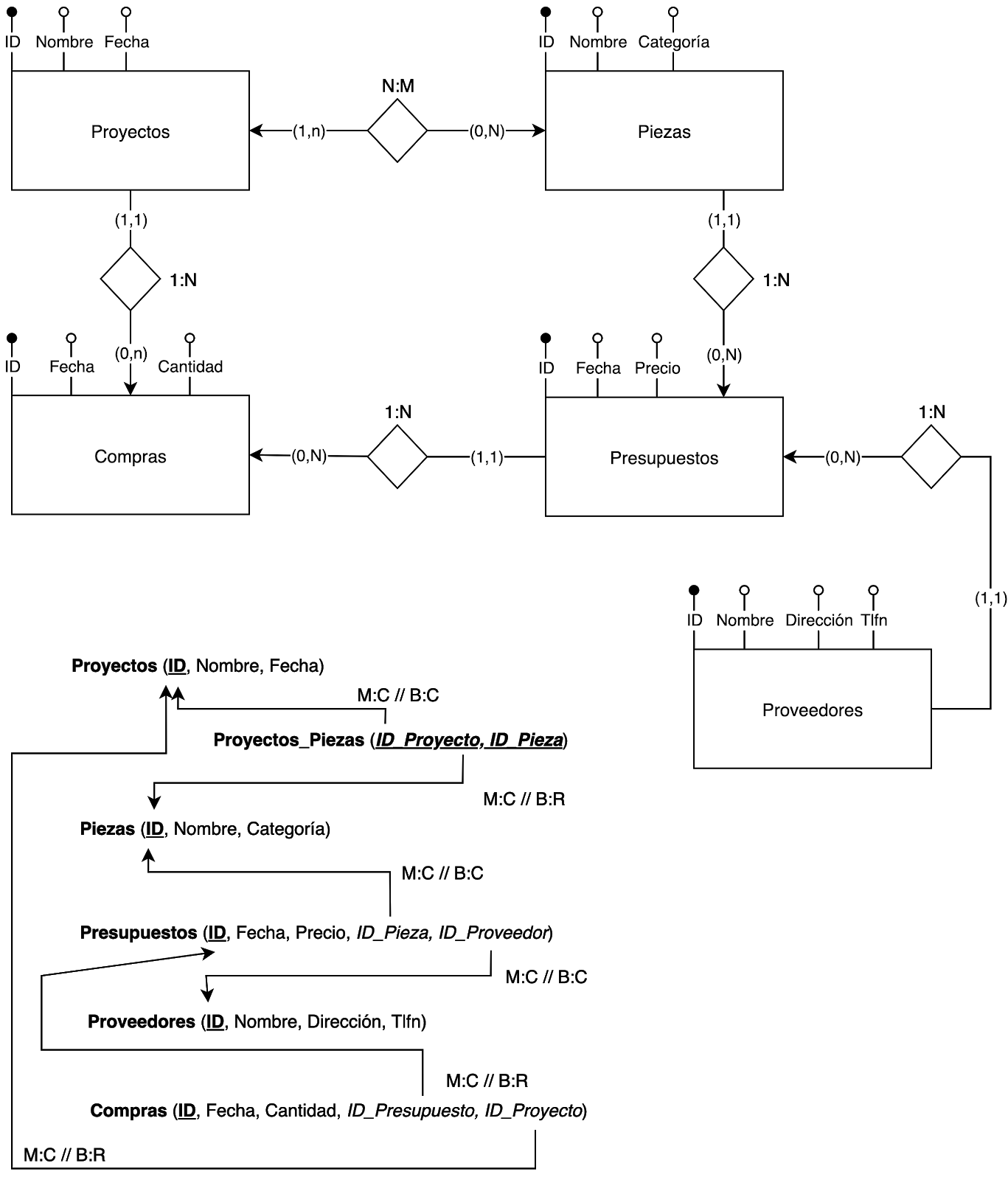
Modelo Relacional:



La tabla PedidosProductos aquí se implementa creando una PK *ad hoc* (CodPedProd); sin embargo, sería recomendable utilizar Pedido, Producto como PK compuesta, de modo que el conjunto de un pedido y un producto no pudiese repetirse, restringiendo así la posibilidad de que un se añadiese a un pedido un producto que ya forma parte de él (lo correcto sería incrementar las cantidad).

## 10. Piezas de proyectos

Mantendremos una bbdd para almacenar información de las piezas utilizadas en cada proyecto desarrollado por nuestra empresa. En la fase de diseño se almacenará el listado de piezas necesarias para cada proyecto y se pedirán a distintos proveedores presupuestos de dichas piezas, almacenando el precio de la pieza para cada proveedor y la fecha del presupuesto.  En la fase de implementación, se almacenarán la información de a qué proveedor se compró cada pieza y por qué precio final (ya que para cada proyecto separado en el tiempo, se usarán presupuestos distintos).   
  
Ejemplares distintos de una misma pieza puede ser comprada para distintos proyectos a distintos proveedores o al mismo por distintos precios.

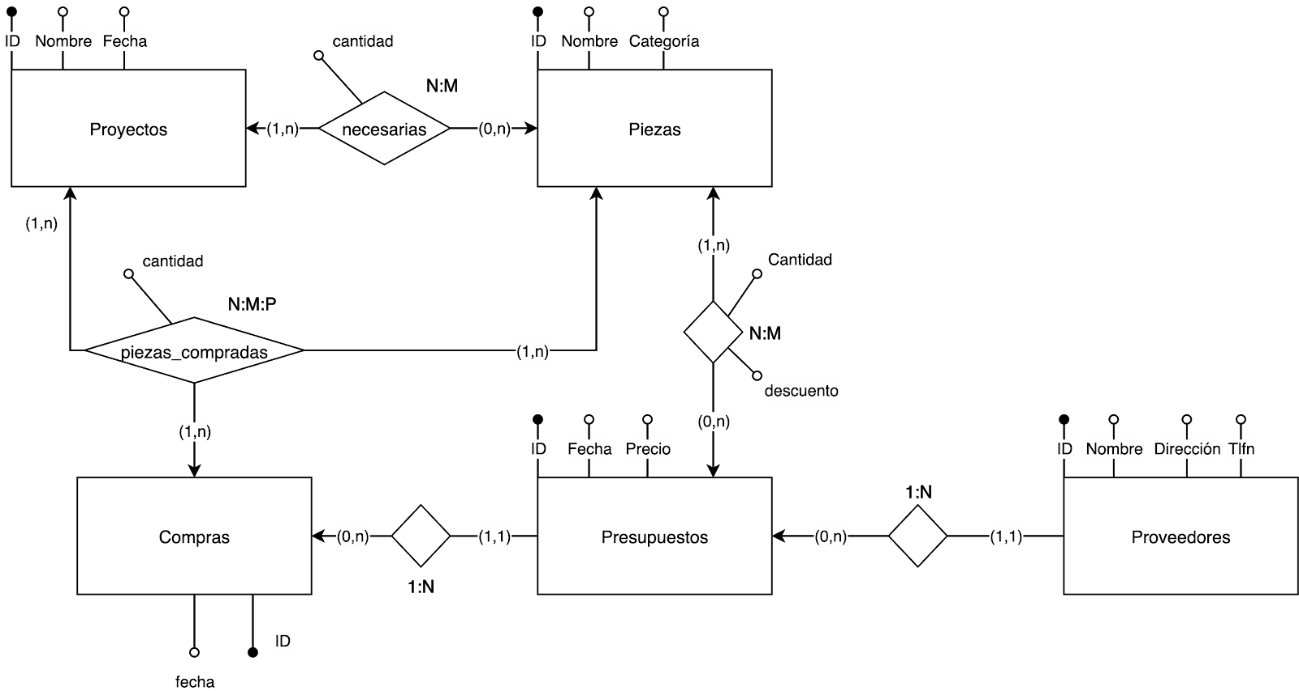
[](https://app.diagrams.net/?page-id=SyolxlnatC9VMOIqrzDf&scale=auto#G1ERv-AKWabPEXKJ0AtJg0IXbEZCkBsf34)

Solución extendida:  
  
- Se contempla que un proyecto pueda necesitar varias piezas de un tipo (una determinada cantidad de una pieza).

- Se contemplan presupuestos que incluyan un conjunto de piezas con una determinada cantidad por cada pieza.

- Cada compra sigue realizándose sobre un presupuesto previamente solicitado, pero pueden agruparse en él piezas de distintos proyectos (o una pieza que se use en varios proyectos, incrementando la cantidad presupuestada de esa pieza).

- De cada compra, se mantiene un registro de qué piezas y en qué cantidad se asignan a cada proyecto.

[](https://app.diagrams.net/?page-id=SyolxlnatC9VMOIqrzDf&scale=auto#G1swGF4d1-U4s0PVEhgFkoD-q4xylzSzPW)

# **11. Cursos de formación**

El departamento de formación de una empresa desea construir una base de datos para planificar y gestionar la formación de sus empleados.

La empresa organiza cursos internos de formación de los que se desea conocer el código de curso, el nombre, una descripción, el número de horas de duración y el coste del curso.

Un curso puede tener como prerrequisito haber realizado otro u otros previamente, y a su vez, la realización de un curso puede ser prerrequisito de otros. Un curso que es un prerrequisito de otro puede serlo de forma obligatoria o sólo recomendable.

Un mismo curso tiene diferentes ediciones, es decir, se imparte en diferentes lugares, fechas y con diferentes horarios (intensivo, de mañana o de tarde). En una misma fecha de inicio sólo puede impartirse una edición de un mismo curso.

Los cursos se imparten por personal de la propia empresa.

De los empleados se desea almacenar su código de empleado, nombre y apellidos, dirección, teléfono, NIF (Número de Identificación Fiscal), fecha de nacimiento, nacionalidad, sexo, firma y salario, así como si está o no capacitado para impartir cursos.

Un mismo empleado puede ser docente en una edición de un curso y alumno en otra edición, pero nunca puede ser ambas cosas a la vez (en una misma edición de curso o lo imparte o lo recibe).

Realiza el Modelo Entidad/Relación y el modelo relacional.

Parte 1:

La empresa organiza cursos internos de formación de los que se desea conocer el código de curso, el nombre, una descripción, el número de horas de duración y el coste del curso.

Un curso puede tener como prerrequisito haber realizado otro u otros previamente, y a su vez, la realización de un curso puede ser prerrequisito de otros. Un curso que es un prerrequisito de otro puede serlo de forma obligatoria o sólo recomendable.

Parte 2:

Un mismo curso tiene diferentes ediciones, es decir, se imparte en diferentes lugares, fechas y con diferentes horarios (intensivo, de mañana o de tarde). En una misma fecha de inicio sólo puede impartirse una edición de un mismo curso.

Parte 3:

Los cursos se imparten por personal de la propia empresa.

De los empleados se desea almacenar su código de empleado, nombre y apellidos, dirección, teléfono, NIF (Número de Identificación Fiscal), fecha de nacimiento, nacionalidad, sexo, firma y salario, así como si está o no capacitado para impartir cursos.

Un mismo empleado puede ser docente en una edición de un curso y alumno en otra edición, pero nunca puede ser ambas cosas a la vez (en una misma edición de curso o lo imparte o lo recibe).

Parte 3.1

Los cursos se imparten por personal de la propia empresa.

De los empleados se desea almacenar su código de empleado, nombre y apellidos, dirección, teléfono, NIF (Número de Identificación Fiscal), fecha de nacimiento, nacionalidad, sexo, firma y salario, así como si está o no capacitado para impartir cursos.

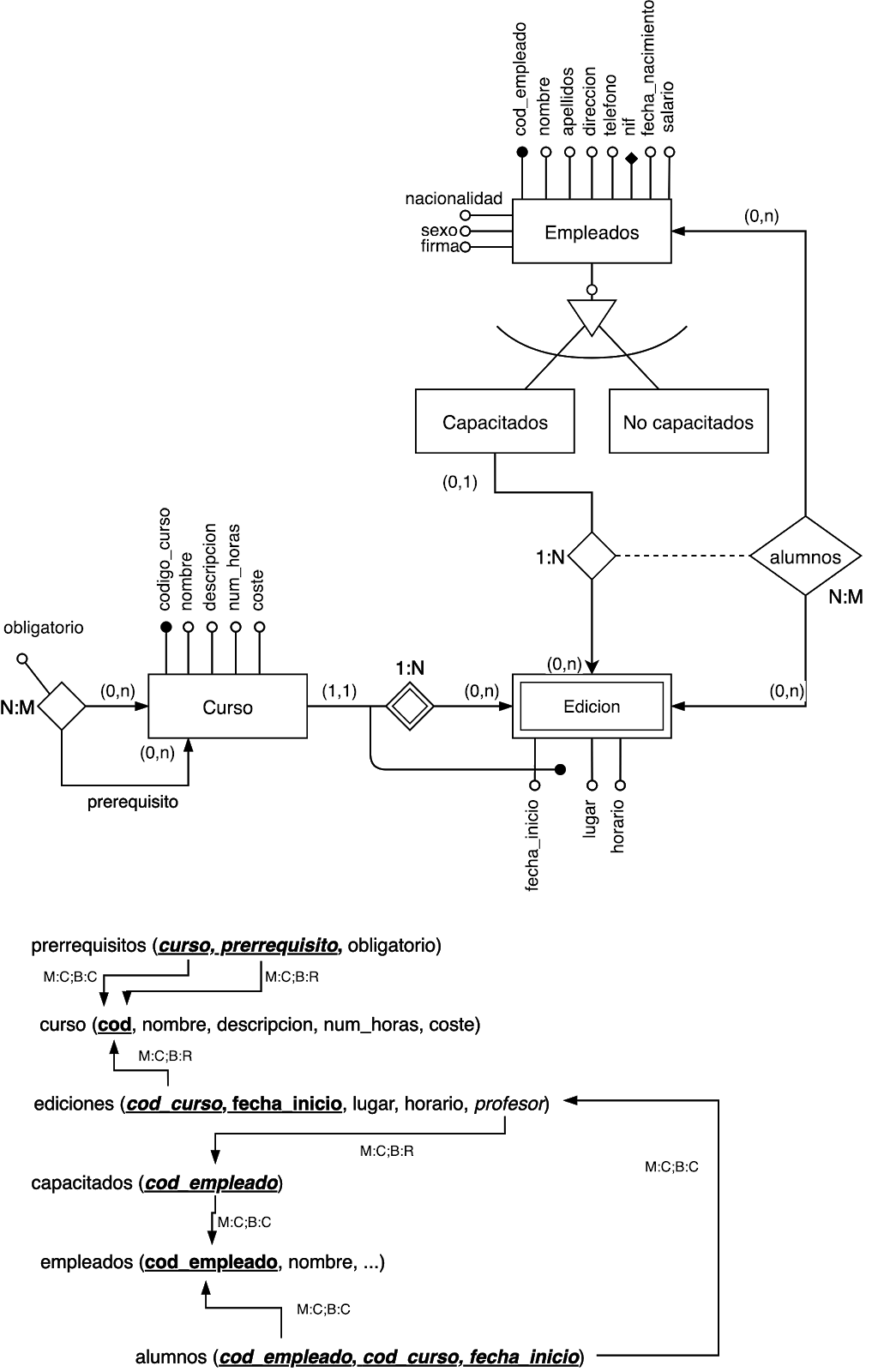
Un mismo empleado puede ser docente en una edición de un curso y alumno en otra edición.

Parte 3.2

Solo los empleados capacitados pueden participar en una edición como docentes. (Restricción Implicita en el enunciado)

Parte 3.3

Un mismo empleado puede ser docente en una edición de un curso y alumno en otra edición, pero nunca puede ser ambas cosas a la vez (en una misma edición de curso o lo imparte o lo recibe).

[](https://app.diagrams.net/?page-id=oFuOFetKs2MZVvZwmoiE&scale=auto#G1R5_RLfKxmNErpF1DC80uBymqmqkAMnxt)

|  |
| --- |
| **DROP** **DATABASE** **IF** **EXISTS** formacion; **CREATE** **DATABASE** formacion CHARACTER **SET** utf8mb4; **USE** formacion;  **CREATE** **TABLE** empleado (   **id** INTEGER **UNSIGNED** AUTO\_INCREMENT PRIMARY **KEY**,   nombre VARCHAR(100) **NOT** NULL,   apellido1 VARCHAR(100) **NOT** NULL,   apellido2 VARCHAR(100),   nif CHAR(9) **NOT** NULL **UNIQUE**,   telefono CHAR(9) **NOT** NULL **UNIQUE**,   direccion VARCHAR(200) **NOT** NULL,   tipo ENUM('Capacitado', 'No capacitado') **NOT** NULL );  **CREATE** **TABLE** capacitado (   **id** INTEGER **UNSIGNED** PRIMARY **KEY**,   **CONSTRAINT** fk\_capacitado FOREIGN **KEY** (**id**) **REFERENCES** empleado(**id**) );  **CREATE** **TABLE** curso (   **id** INTEGER **UNSIGNED** AUTO\_INCREMENT PRIMARY **KEY**,   nombre VARCHAR(100) **NOT** NULL **UNIQUE**,   descripcion VARCHAR(512) **NOT** NULL,   duracion SMALLINT **UNSIGNED** **NOT** NULL,   coste FLOAT(6,2) **NOT** NULL );  **CREATE** **TABLE** prerrequisitos (   id\_curso INTEGER **UNSIGNED** **NOT** NULL,   id\_curso\_prerrequisito INTEGER **UNSIGNED** **NOT** NULL,   es\_obligatorio BOOLEAN **NOT** NULL,   PRIMARY **KEY** (id\_curso, id\_curso\_prerrequisito),   FOREIGN **KEY** (id\_curso) **REFERENCES** curso(**id**),   FOREIGN **KEY** (id\_curso\_prerrequisito) **REFERENCES** curso(**id**)     );    **CREATE** **TABLE** edicion (   id\_curso INTEGER **UNSIGNED** **NOT** NULL,   fecha DATE **NOT** NULL,   horario ENUM('Intensivo', 'Mañana', 'Tarde') **NOT** NULL,   lugar VARCHAR(100) **NOT** NULL,   profesor INTEGER **UNSIGNED** **NOT** NULL,   PRIMARY **KEY** (id\_curso, fecha, horario, lugar),   FOREIGN **KEY** (id\_curso) **REFERENCES** curso(**id**),   FOREIGN **KEY** (profesor) **REFERENCES** capacitado(**id**)  -- !!! );  **CREATE** **TABLE** empleado\_recibe\_formacion (   id\_empleado INTEGER **UNSIGNED** **NOT** NULL,   id\_curso INTEGER **UNSIGNED** **NOT** NULL,   fecha DATE **NOT** NULL,   horario ENUM('Intensivo', 'Mañana', 'Tarde') **NOT** NULL,   lugar VARCHAR(100) **NOT** NULL,   PRIMARY **KEY** (id\_empleado, id\_curso, fecha, horario, lugar),   FOREIGN **KEY** (id\_empleado) **REFERENCES** empleado(**id**),   FOREIGN **KEY** (id\_curso, fecha, horario, lugar)          **REFERENCES** edicion(id\_curso, fecha, horario, lugar) ); |

## 11. Cursos de formación

El departamento de formación de una empresa desea construir una base de datos para planificar y gestionar la formación de sus empleados.

La empresa organiza cursos internos de formación de los que se desea conocer el código de curso, el nombre, una descripción, el número de horas de duración y el coste del curso.

Un curso puede tener como prerrequisito haber realizado otro u otros previamente, y a su vez, la realización de un curso puede ser prerrequisito de otros. Un curso que es un prerrequisito de otro puede serlo de forma obligatoria o sólo recomendable.

Un mismo curso tiene diferentes ediciones, es decir, se imparte en diferentes lugares, fechas y con diferentes horarios (intensivo, de mañana o de tarde). En una misma fecha de inicio sólo puede impartirse una edición de un mismo curso.

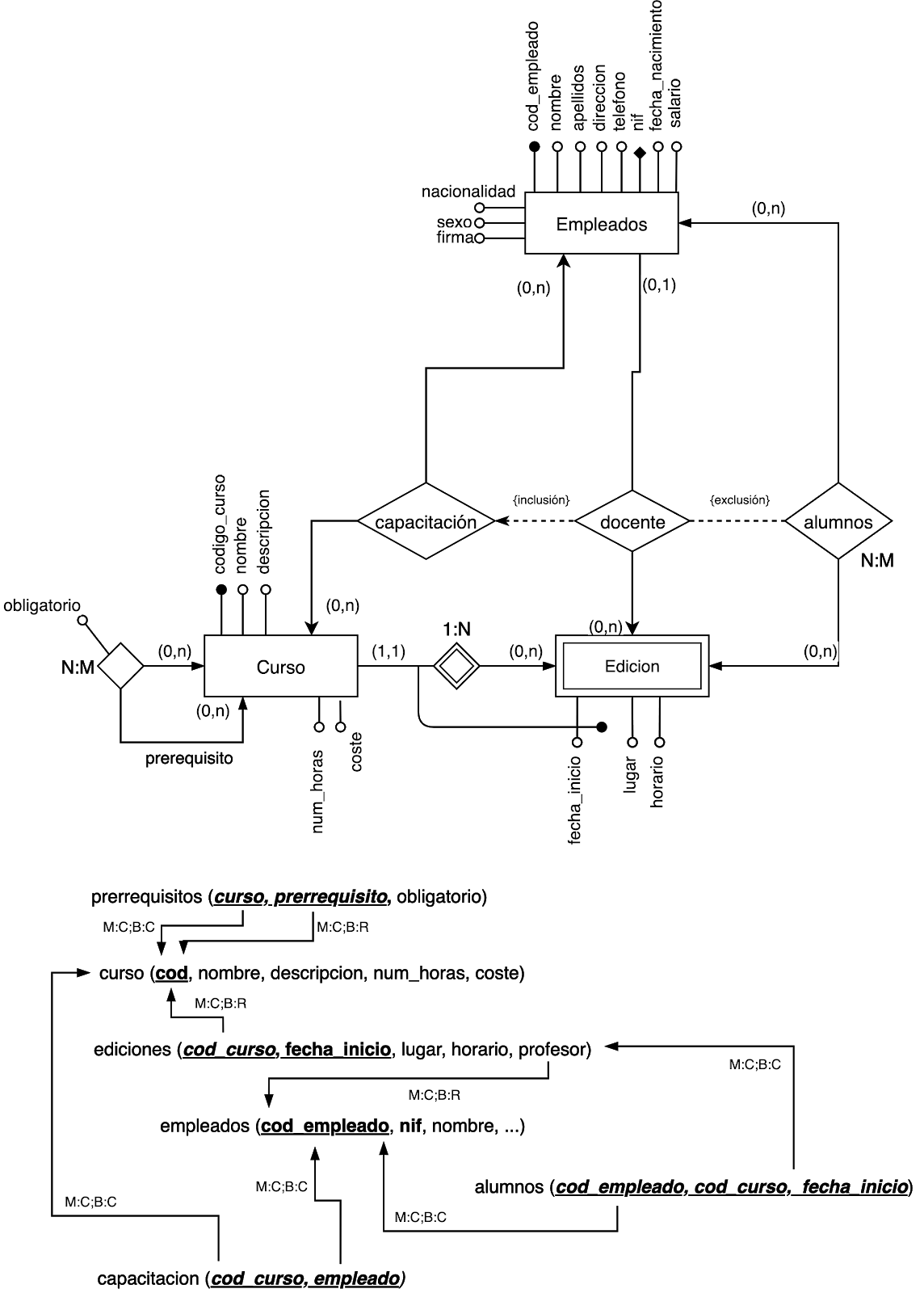
Los cursos se imparten por personal de la propia empresa.

De los empleados se desea almacenar su código de empleado, nombre y apellidos, dirección, teléfono, NIF (Número de Identificación Fiscal), fecha de nacimiento, nacionalidad, sexo, firma y salario, así como si está o no capacitado para impartir cursos.

Un mismo empleado puede ser docente en una edición de un curso y alumno en otra edición, pero nunca puede ser ambas cosas a la vez (en una misma edición de curso o lo imparte o lo recibe).

## 12.  Cursos de formación modificado

Supongamos que la capacitación es para cursos específicos. Es decir: queremos saber para cada empleado si está o no capacitado para impartir cada curso.

[](https://app.diagrams.net/?page-id=oFuOFetKs2MZVvZwmoiE&scale=auto#G1H_YRqyXayTO3BHIk7Ocus4ixnPQt20qB)

# **Residuos**

Se desea abordar la problemática ambiental de los residuos tóxicos y peligrosos cuya incorrecta gestión produce daños de gran importancia en el medio ambiente y en la salud del ser humano. La información a contemplar es la que corresponde desde que es producido el residuo por un centro o empresa productora hasta que el residuo se encuentra en lugar seguro, donde los residuos reciben un tratamiento especial como puede ser la incineración, almacenamiento en depósitos de seguridad, etc. En el sistema de información se desea considerar la información de los productores de residuos, los residuos, las empresas que transportan los residuos hasta los lugares seguros y el traslado de los residuos teniendo en cuenta el tipo de transporte, el envase, etc.

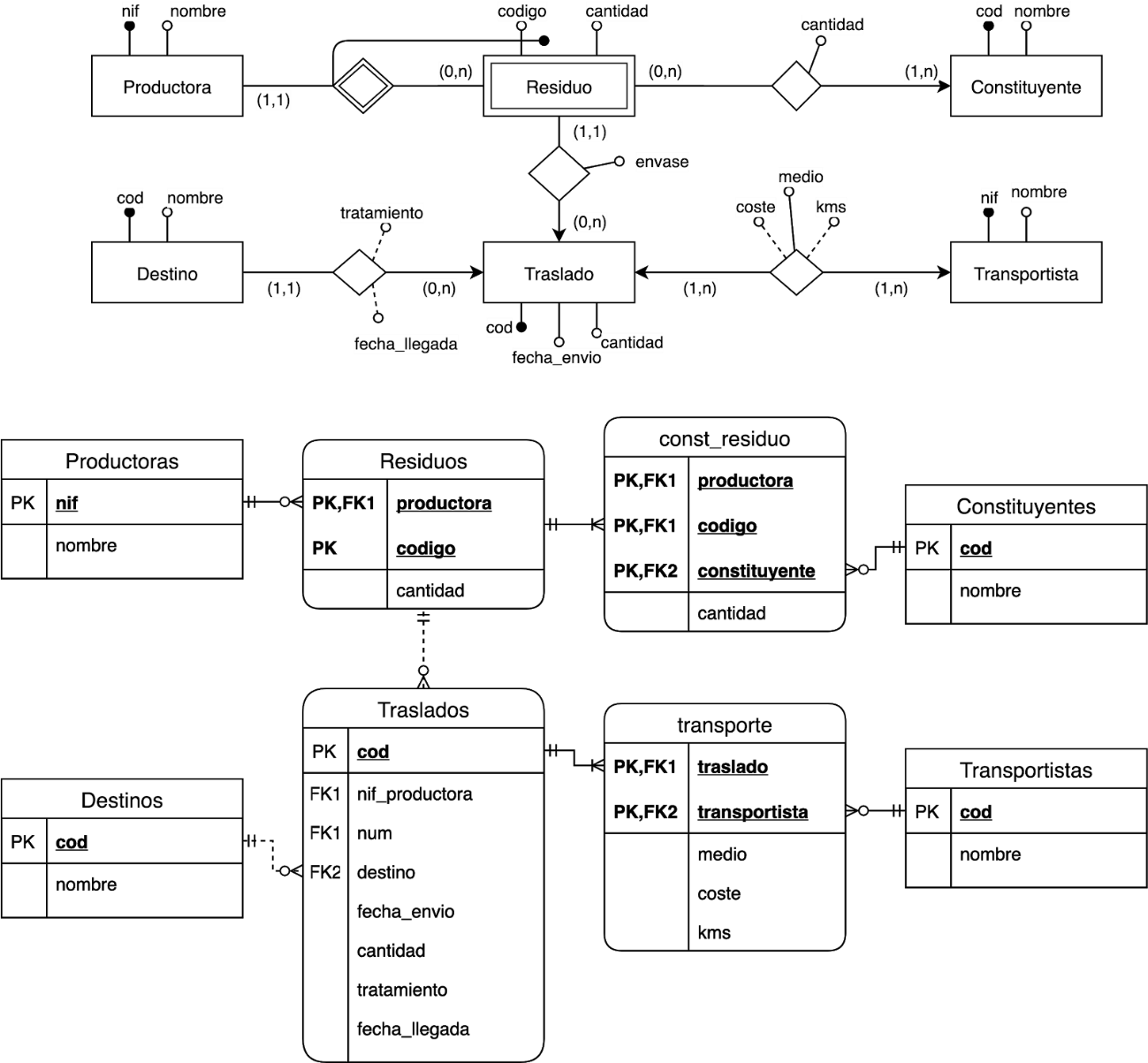
Se consideran además los siguientes supuestos semánticos en el problema:

* Una empresa productora produce un número amplio de residuos constituidos por un número variable de constituyentes químicos.
* Más de una empresa productora puede producir residuos con igual número de constituyentes químicos y con las mismas o distintas cantidades de los mismos.
* Las empresas productoras asignan un código único a los residuos que producen, lo que les permite diferenciar distintas producciones de los mismos productos. Además, más de una empresa puede asignar el mismo código a los residuos que produce.
* Los residuos pueden ser trasladados en su totalidad (cantidad total del mismo o en partes, o no ser trasladados nunca).
* En cada traslado de residuos, la cantidad que se traslada de los mismos es enviada a un único destino.
* En un misma fecha las empresas productoras pueden ordenar más de un traslador de un mismo o distintos residuo (cantidades parciales del mismo) a un mismo o distinto destino.
* En cada traslado puede intervenir más de una empresa transportista usando el mismo o distinto transporte, por lo que resulta interesante conocer tanto el medio de transporte utilizado como los kilómetros realizados, así como el coste del trabajo.
* El residuo se traslada en un tipo de envase determinado por la empresa productora y que no varía a lo largo de su traslado.
* Es interesante conocer la fecha de llegada a destino y el tratamiento a que se someten los residuos una vez alcanzado el mismo.
* Por seguridad se considera que en un traslado sólo puede trasladarse un residuo de una empresa productora.

Asunciones:

- Ya que “más de una empresa puede asignar el mismo código a los residuos que produce”, el código no puede ser el atributo identificador, así que: o usamos otro id *ad hoc* o establecemos la entidad residuo como dependiente en identificación de empresa (ya que el código sí es unívoco dentro de los residuos de una misma empresa).

- Como “en cada traslado puede intervenir más de una empresa transportista usando el mismo o distinto transporte”, se asume que la fecha de llegada también podría ser distinta para cada transportista y medio, no solo para cada traslado.

[](https://app.diagrams.net/?page-id=C5RBs43oDa-KdzZeNtuy&scale=auto#G148RKwyFgjiwoz1G2va4-uK_dRatxCG8k)

La fecha de llegada podría considerarse en la interrelación con el transportista si se entiende que un traslado se puede distribuir entre dos transportistas, llegando cada parte del traslado en dos fechas distintas. Sin embargo, parece más verosímil la solución propuesta, entendiendo que los varios transportistas que intervienen en un traslado, lo hacen por fases (por ejemplo, por que un transportista realice el traslado hasta un puerto, donde embarque y otro transportista lo lleve a un destino).

**14. Cupones de supermercado**

Se desea crear una BBDD para almacenar información de las compras realizadas en un supermercado. Para calcular el precio de cada compra habrá que tener en consideración cuándo se aplican descuentos en los productos comprados, ya que los clientes podrán canjear cupones de descuento.

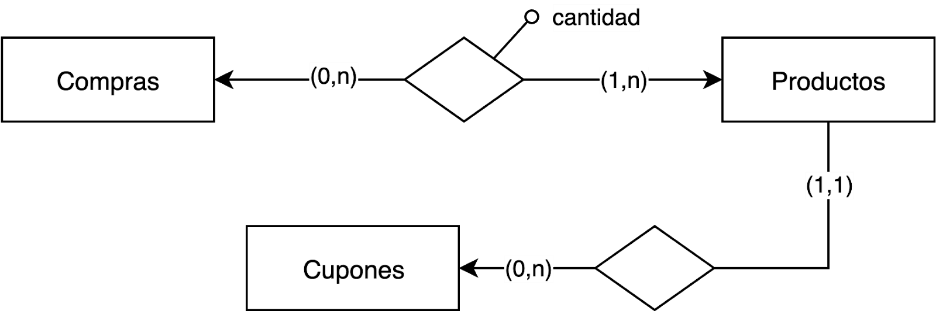
Cada cupón es válido para un determinado producto, de modo que el descuento se aplicará al canjearlo en compras que incluyan ese producto. No podrá utilizarse más de un cupón en cada producto comprado. En caso de comprar varios productos, el descuento se aplicará sobre todas las unidades compradas.

Asunción:

Un cupón se usa una única vez. No es un código con un número determinado de usos.

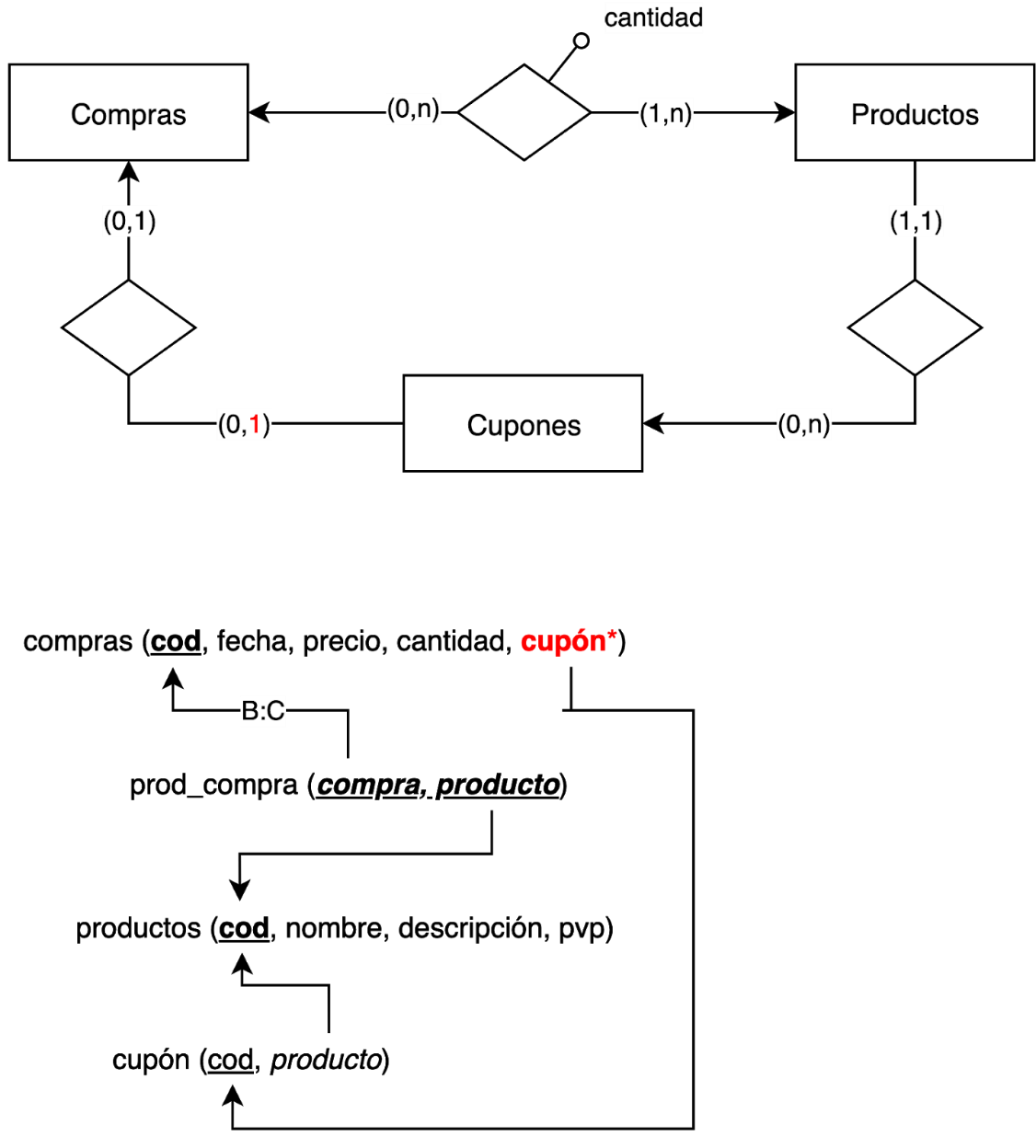
Desarrollo:

Es fácil establecer la interrelación entre compras y productos (productos que componen una compra) y entre cupones y producto (producto para el que es válido cada cupón):

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1ImxxzfEeToceQGlWv6bVIrwZr3-O1B2S)

La dificultad está en almacenar la información de los cupones que se utilizaron en cada compra.

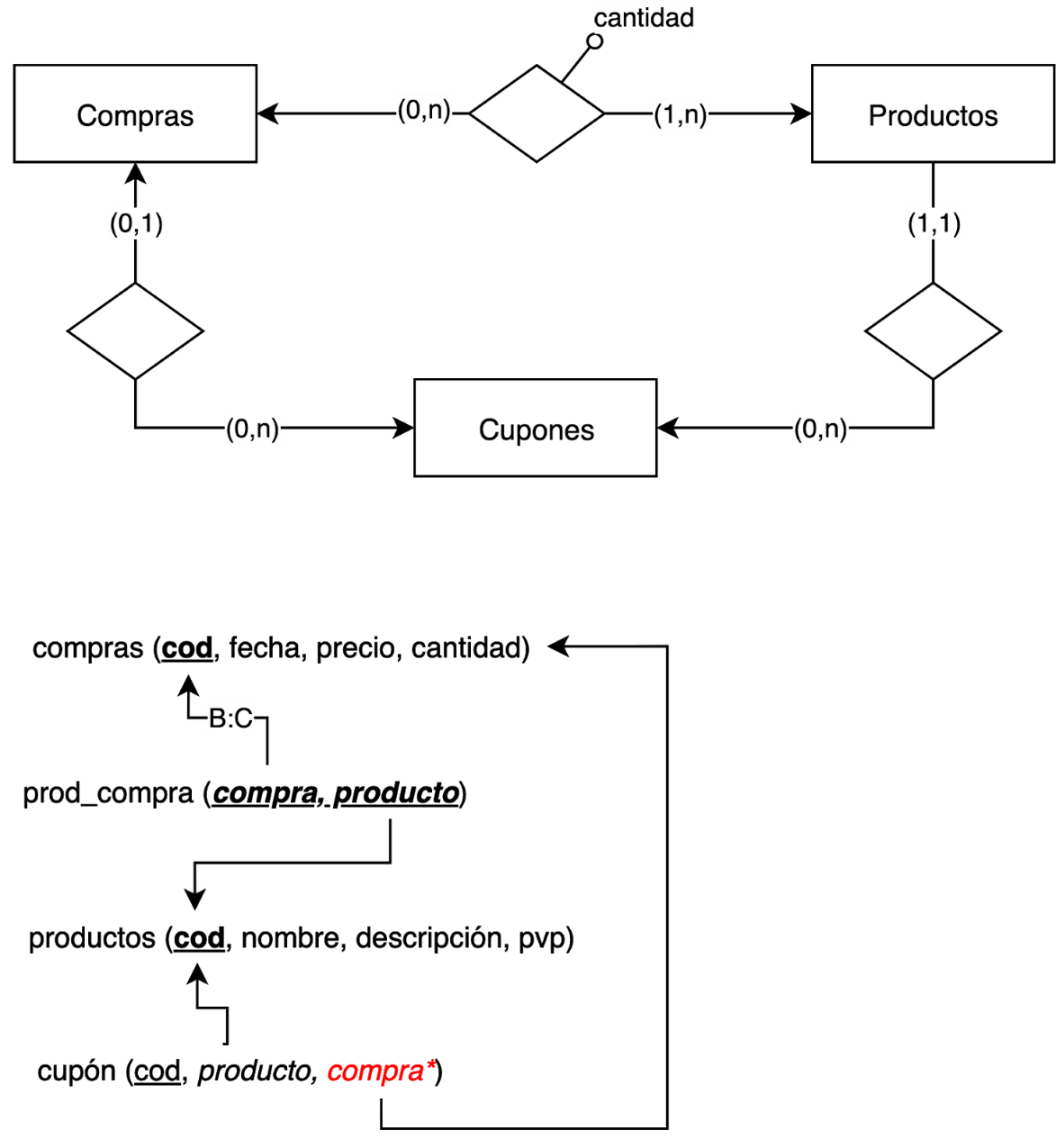
Podemos asociar la compra con los cupones que van en ella. Por ejemplo, si hay un único cupón como máximo por compra:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1o0rr7acTMX0xYphDdrprp1BBC7aeB-sd)

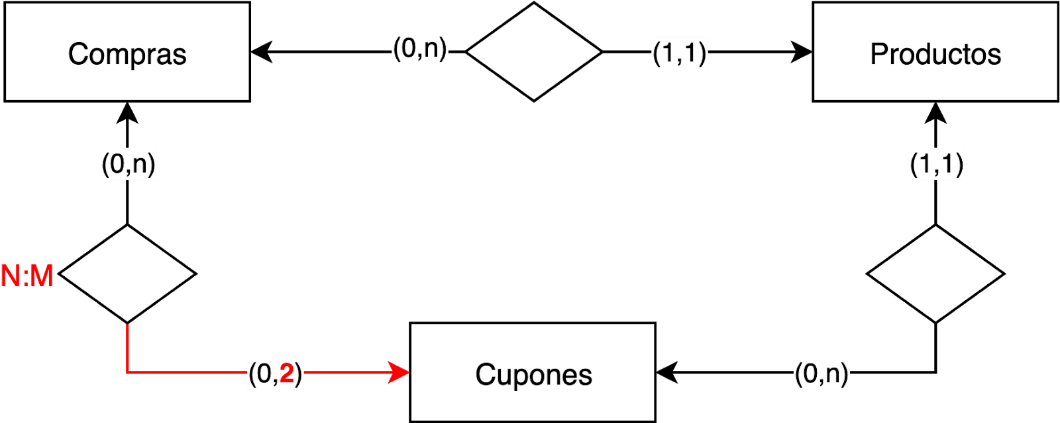
Por ser la relación entre cupones y compras 1:1, puede convertirse pasando la cualquiera de las dos a la otra como clave foranea, pero esa clave foranea debería ser también UNIQUE.

El bucle no es redundante, ya que la interrelación entre cupones y productos es la validez de éstos, mientras que entre compras y cupones es su uso. Por otra parte, un cupón usado en una compra, sabemos sobre qué producto se aplica comprobando que el producto esté en la compra.

Si no hay límite de cupones utilizados en cada compra el modelo podría ser este:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1NZA-zGwyCJZI55ZYzM57YJ0yh3_QCJZ7)

Y en el caso de que se estipulase un máximo concreto de cupones por compra, se podría estipular en el máximo de la cardinalidad, aunque esa restricción solo podría implementarse a nivel físico (el modelo relacional no cambiaría):

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1PEz37p3yyaJgNuic0KiP8EKb_sofV1-o)

## Variación: si se estipula un máximo de cupones por cada producto en una compra.

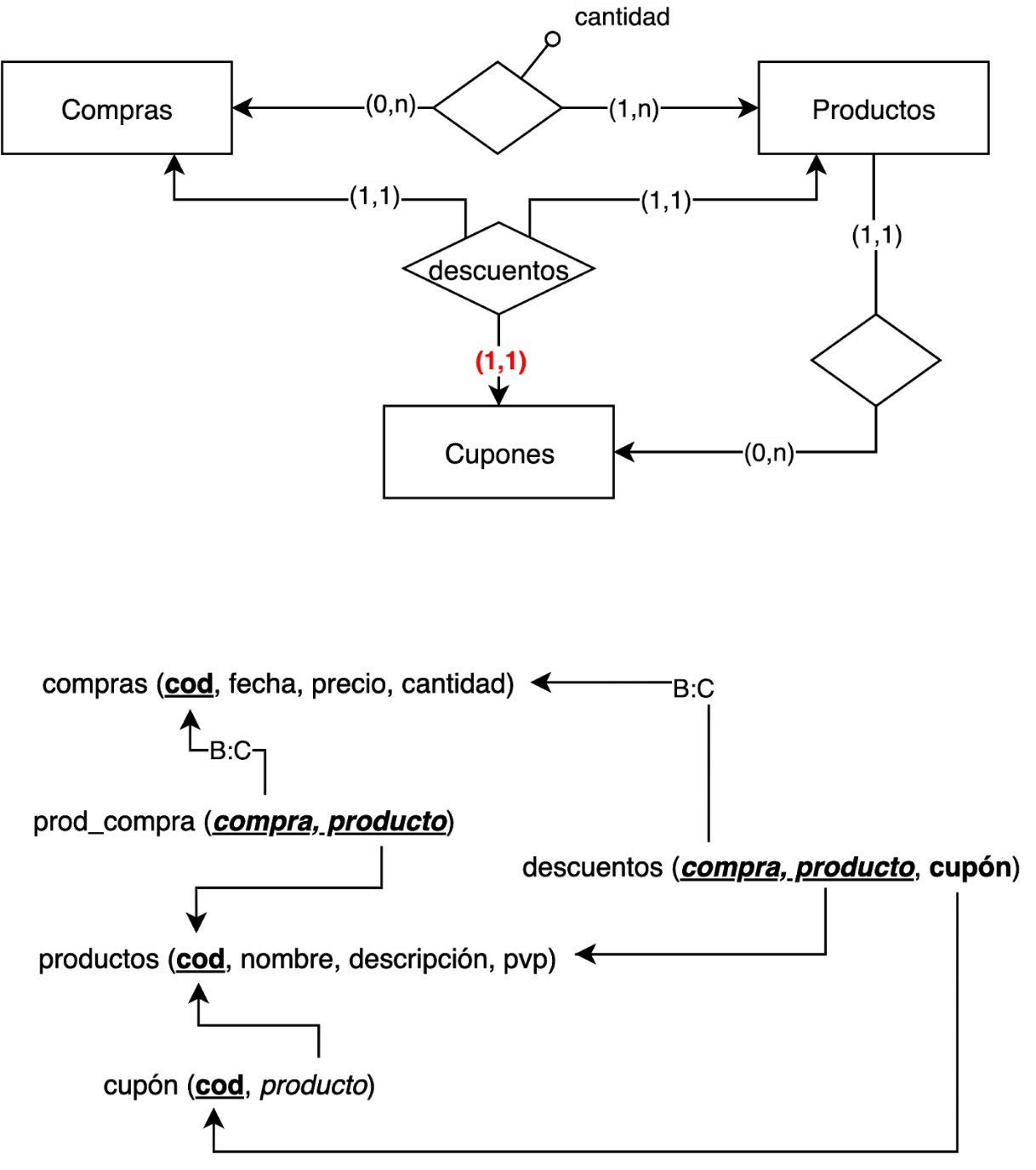
Ahora bien, supongamos que lo que queremos limitar es que no se pueda utilizar más de un cupón por cada producto comprado, no por cada compra en conjunto. El modelo anterior no permite especificar eso (dos cupones distintos para el mismo producto podrían usarse en la misma compra). Esta condición se refiere a la concurrencia de las 3 entidades, de modo que solo se puede restringir mediante una interrelación ternaria:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1kgzBRjbrjxhKrhBCmjGnxRm1QB4cwuIN)

Al introducirla, la interrelación binaria entre compras y cupones sería redundante y puede eliminarse, pero no está tan claro que la interrelación entre compras y productos también lo sea.  
  
La ternaria que acabamos de introducir es una 1:1:1 ya que registra el uso de los cupones. Entendida así, la relación entre compras y productos sigue siendo necesaria para establecer la interrelación N:M entre compras y productos.  
  
Una interrelación 1:1:1 puede convertirse de muchos modos, así que lo mejor será considerar aquel que más nos interesa a nivel relacional: para poder restringir que un producto en una compra use un único cupón, necesitaremos que la tabla que registra el uso de cupones esté identificada por compras y productos:  
  
descuentos (*compra, producto,* ***cupón***)

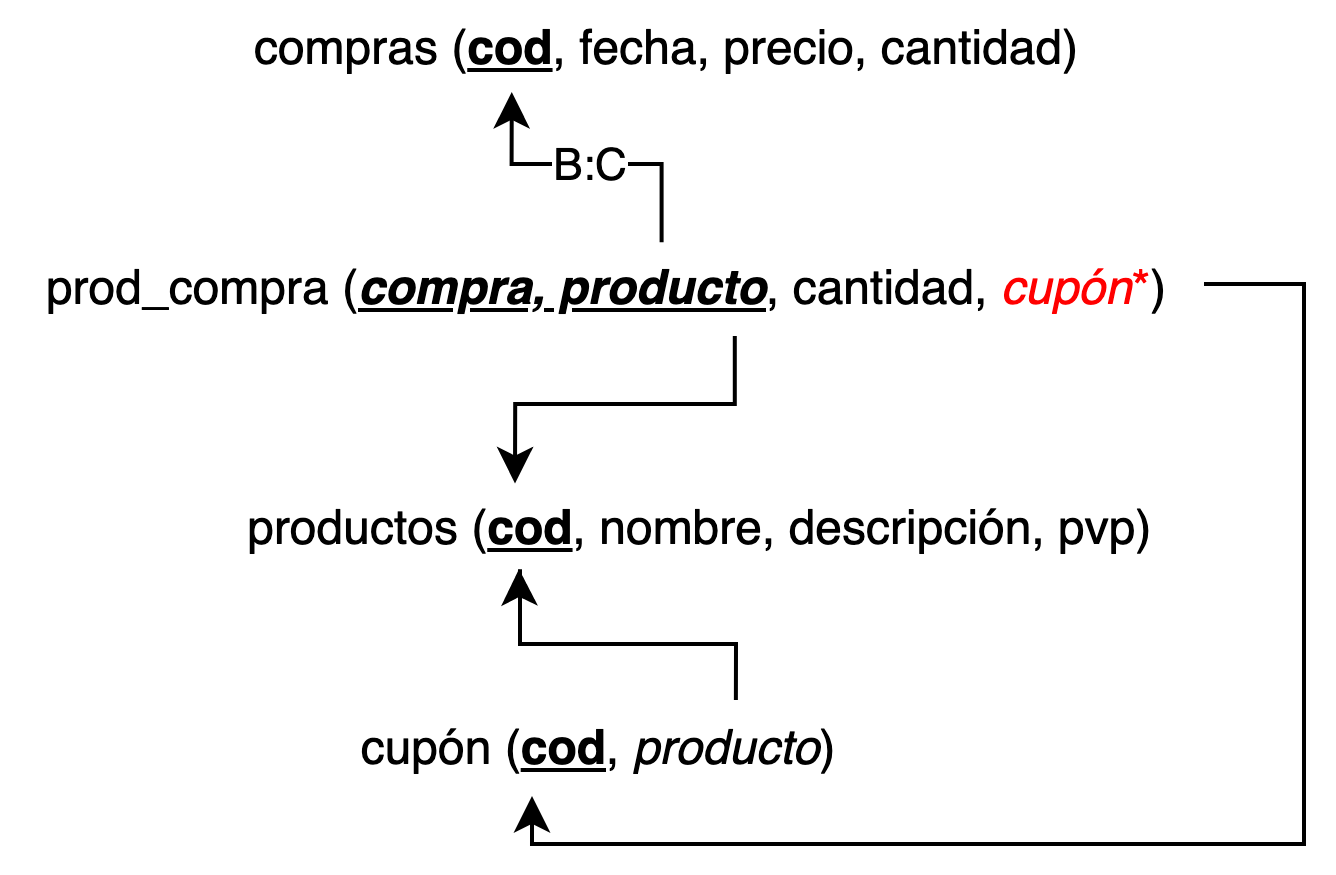
de ese modo, si se intenta insertar un segundo cupón para el mismo producto en la misma compra, saltaría un error por estarse repitiendo la clave primaria.

Por lo tanto, el modelo, convirtiéndolo, quedaría así:

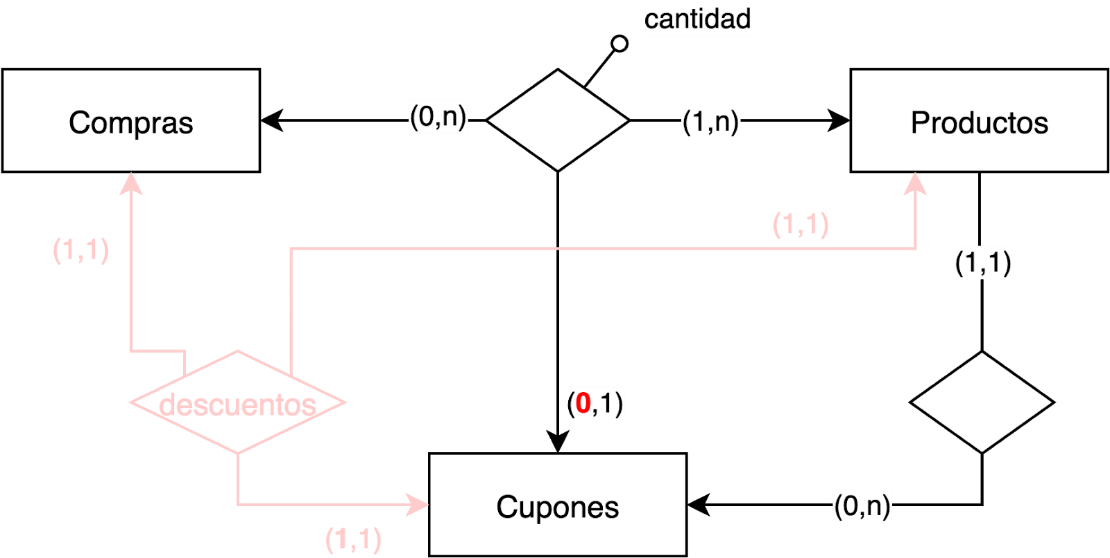
[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1EeswFtj56qgcRwravxNaVtUWDY-32q1P)

Observamos sin embargo, ahora, que la tabla que almacena el contenido de los productos por compra (prod\_compra) y los cupones utilizados (descuentos) comparten la misma clave primaria y solo se diferencian en que la segunda introduce filas por todas aquellas de la primera en las que se asigna un cupón.

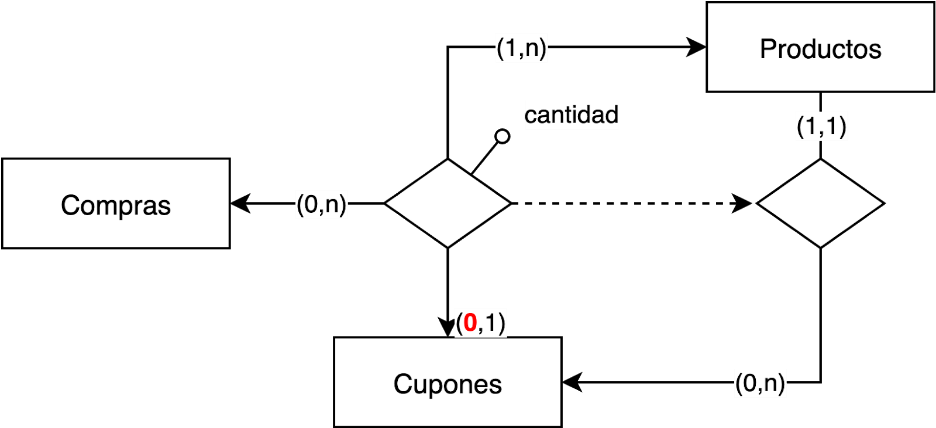
Dicho de otro modo: la tabla de compras por producto ya permitiría almacenar los cupones con simplemente añadir un atributo optativo en que se referencia el cupón utilizado cuando lo haya:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1qerh-xLWOKBZC-5Is7TCDUU60QT5GRFN)

Es reseñable que esta conversión es la equivalente a éste modelo conceptual:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1MSGZU7rLaIcwqFcRNcfpU-O1syHzMEfF)

Donde las cardinalidades se definen de un modo distinto al visto (ya se ha comentado que hay variaciones entre autores sobre el mejor modo de considerar las cardinalidades en interrelaciones superiores a 2).  
  
Además, realizando el modelo conceptual de este modo, podríamos señalar la restricción implícita de que un cupón solo se asocie en su uso con el producto para el que es válido:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1jD4iX7UN_LLCFvnJmCHtPEypvv5WaTBF)

**14. Cupones de supermercado**

Se desea crear una BBDD para almacenar información de las compras realizadas en un supermercado. Para calcular el precio de cada compra habrá que tener en consideración cuándo se aplican descuentos en los productos comprados, ya que los clientes podrán canjear cupones de descuento.

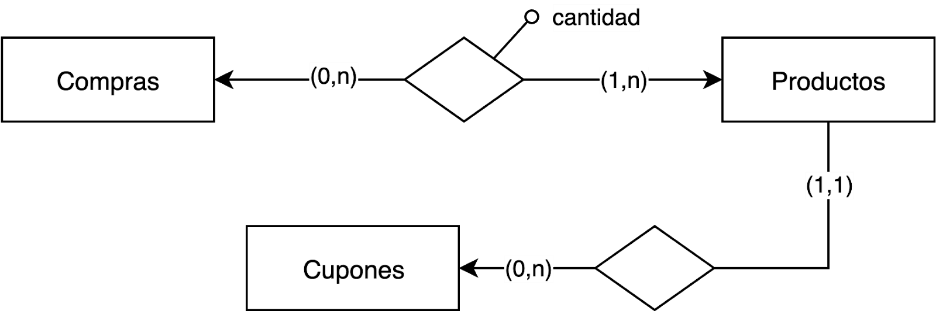
Cada cupón es válido para un determinado producto, de modo que el descuento se aplicará al canjearlo en compras que incluyan ese producto. No podrá utilizarse más de un cupón en cada producto comprado. En caso de comprar varios productos, el descuento se aplicará sobre todas las unidades compradas.

Asunción:

Un cupón se usa una única vez. No es un código con un número determinado de usos.

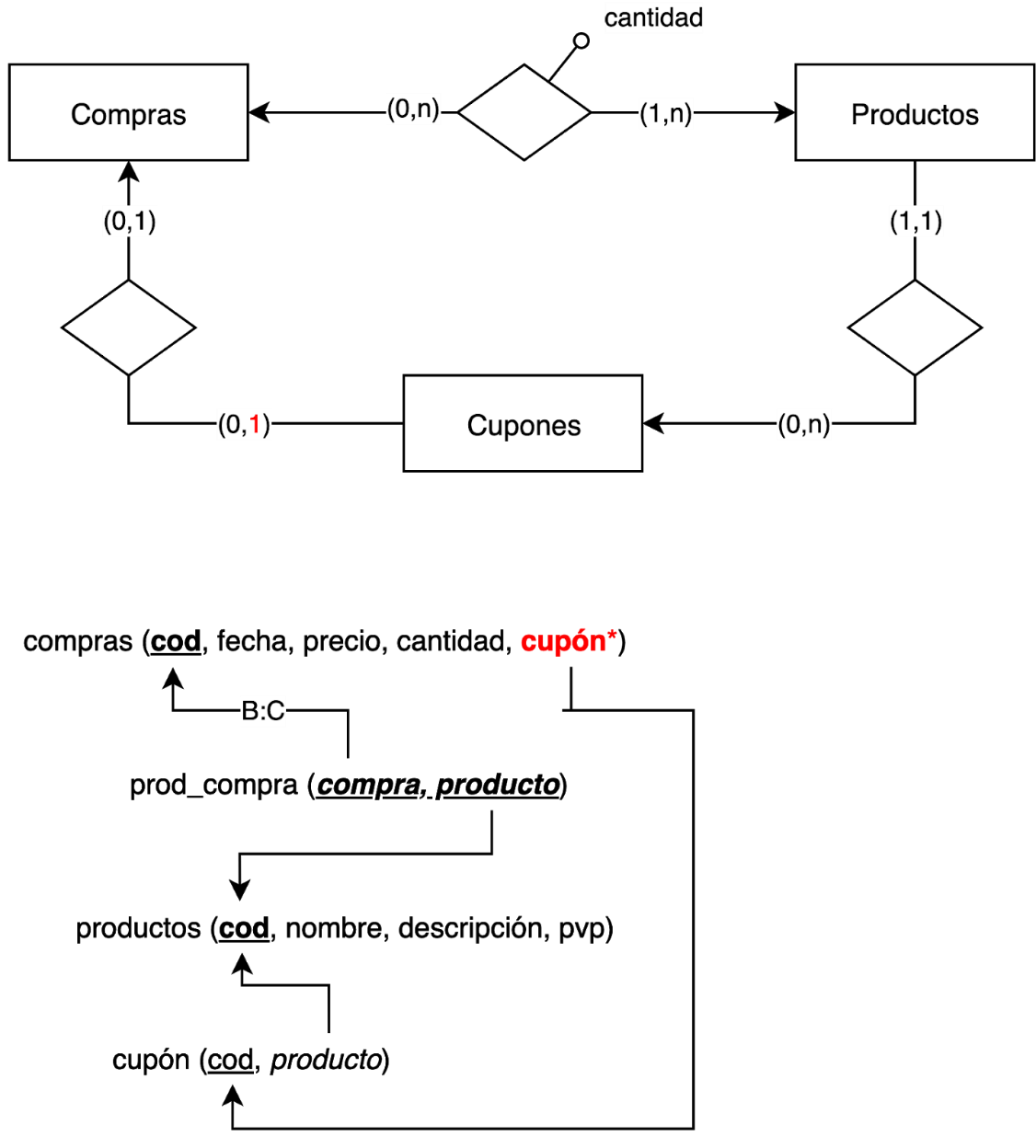
Desarrollo:

Es fácil establecer la interrelación entre compras y productos (productos que componen una compra) y entre cupones y producto (producto para el que es válido cada cupón):

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1ImxxzfEeToceQGlWv6bVIrwZr3-O1B2S)

La dificultad está en almacenar la información de los cupones que se utilizaron en cada compra.

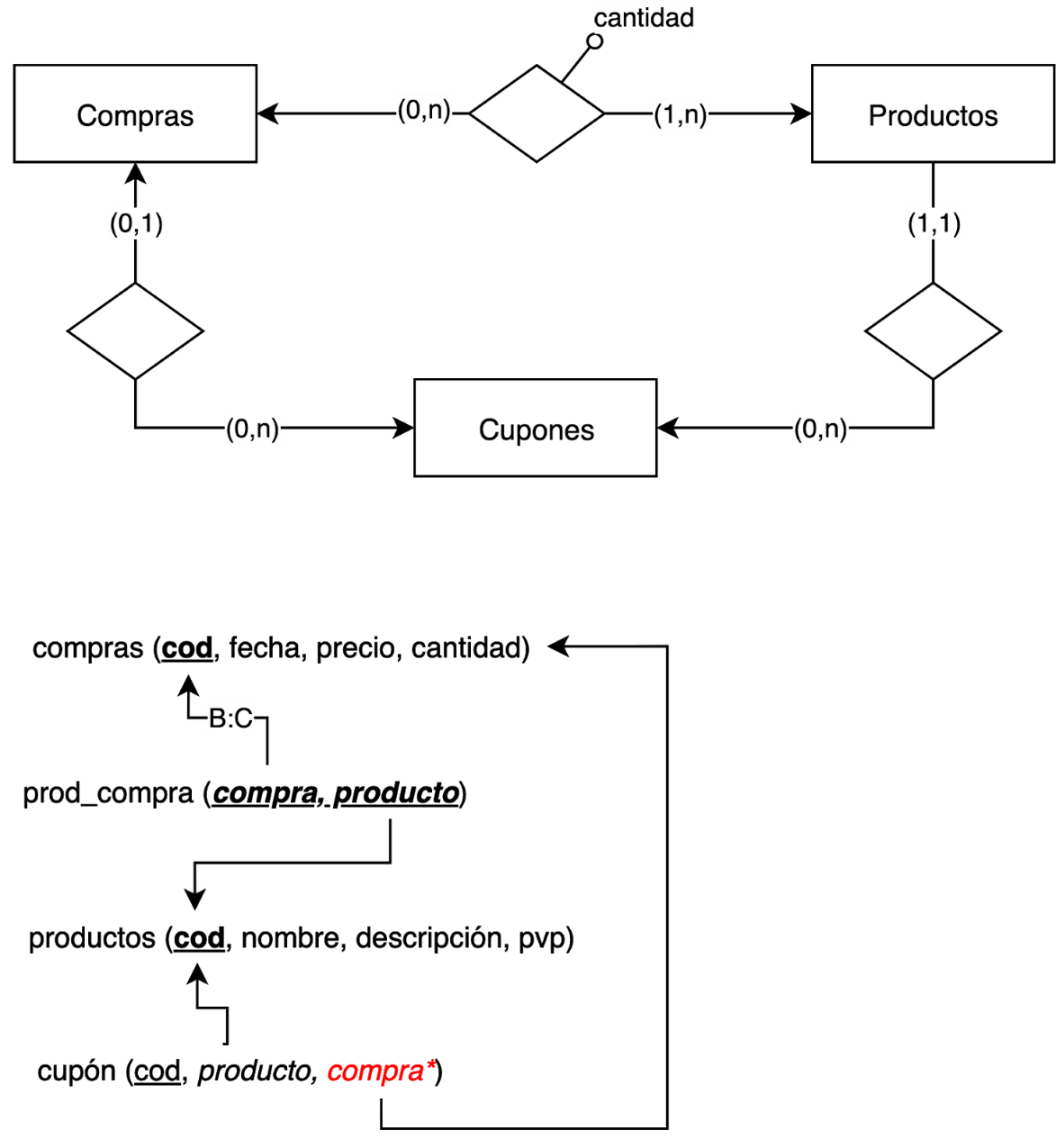
Podemos asociar la compra con los cupones que van en ella. Por ejemplo, si hay un único cupón como máximo por compra:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1o0rr7acTMX0xYphDdrprp1BBC7aeB-sd)

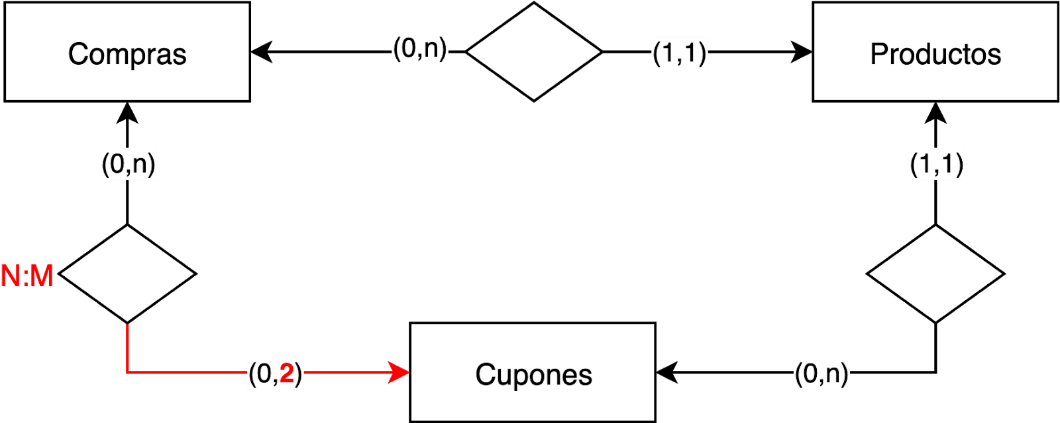
Por ser la relación entre cupones y compras 1:1, puede convertirse pasando la cualquiera de las dos a la otra como clave foranea, pero esa clave foranea debería ser también UNIQUE.

El bucle no es redundante, ya que la interrelación entre cupones y productos es la validez de éstos, mientras que entre compras y cupones es su uso. Por otra parte, un cupón usado en una compra, sabemos sobre qué producto se aplica comprobando que el producto esté en la compra.

Si no hay límite de cupones utilizados en cada compra el modelo podría ser este:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1NZA-zGwyCJZI55ZYzM57YJ0yh3_QCJZ7)

Y en el caso de que se estipulase un máximo concreto de cupones por compra, se podría estipular en el máximo de la cardinalidad, aunque esa restricción solo podría implementarse a nivel físico (el modelo relacional no cambiaría):

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1PEz37p3yyaJgNuic0KiP8EKb_sofV1-o)

## Variación: si se estipula un máximo de cupones por cada producto en una compra.

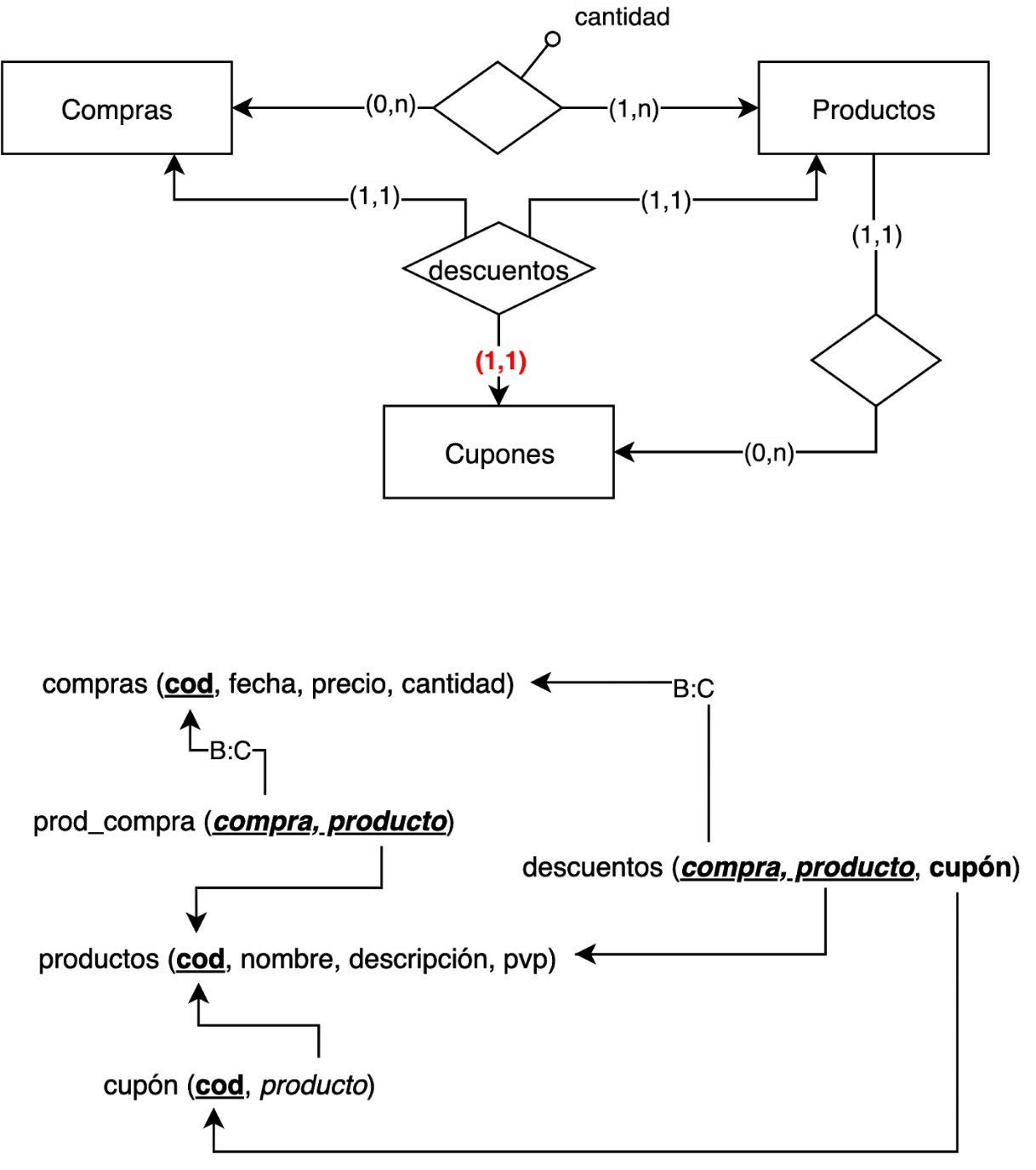
Ahora bien, supongamos que lo que queremos limitar es que no se pueda utilizar más de un cupón por cada producto comprado, no por cada compra en conjunto. El modelo anterior no permite especificar eso (dos cupones distintos para el mismo producto podrían usarse en la misma compra). Esta condición se refiere a la concurrencia de las 3 entidades, de modo que solo se puede restringir mediante una interrelación ternaria:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1kgzBRjbrjxhKrhBCmjGnxRm1QB4cwuIN)

Al introducirla, la interrelación binaria entre compras y cupones sería redundante y puede eliminarse, pero no está tan claro que la interrelación entre compras y productos también lo sea.  
  
La ternaria que acabamos de introducir es una 1:1:1 ya que registra el uso de los cupones. Entendida así, la relación entre compras y productos sigue siendo necesaria para establecer la interrelación N:M entre compras y productos.  
  
Una interrelación 1:1:1 puede convertirse de muchos modos, así que lo mejor será considerar aquel que más nos interesa a nivel relacional: para poder restringir que un producto en una compra use un único cupón, necesitaremos que la tabla que registra el uso de cupones esté identificada por compras y productos:  
  
descuentos (*compra, producto,* ***cupón***)

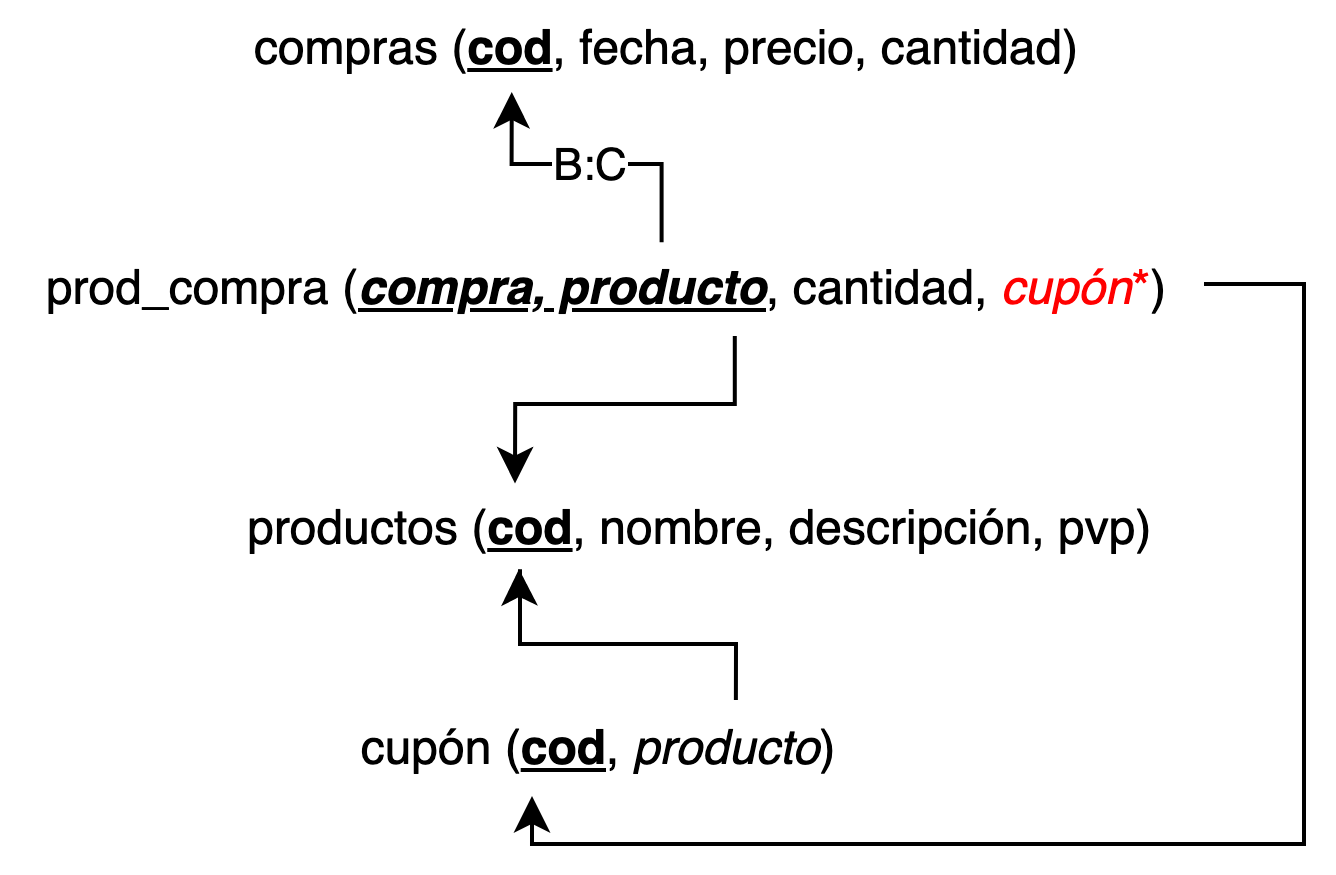
de ese modo, si se intenta insertar un segundo cupón para el mismo producto en la misma compra, saltaría un error por estarse repitiendo la clave primaria.

Por lo tanto, el modelo, convirtiéndolo, quedaría así:

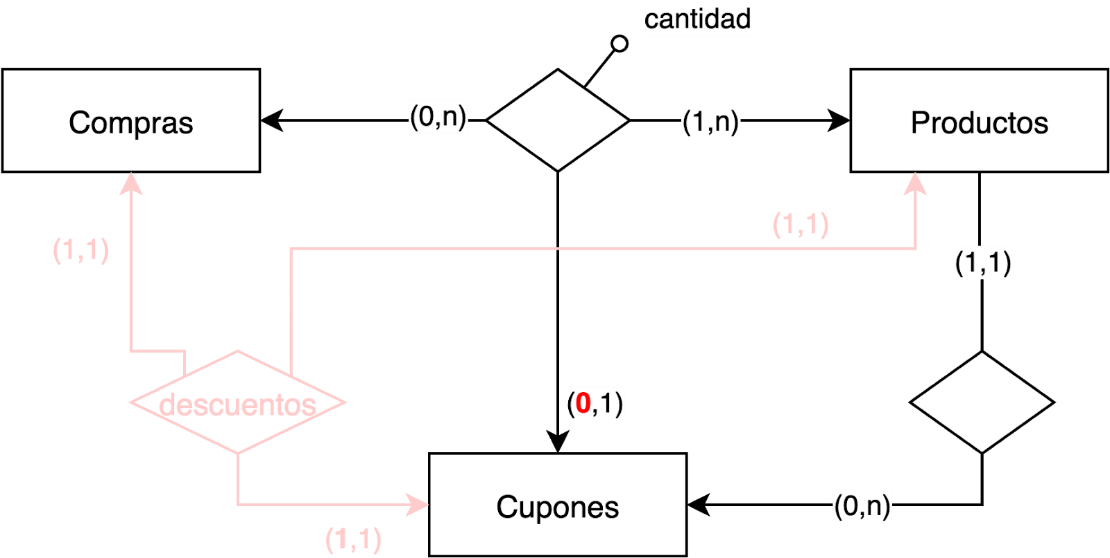
[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1EeswFtj56qgcRwravxNaVtUWDY-32q1P)

Observamos sin embargo, ahora, que la tabla que almacena el contenido de los productos por compra (prod\_compra) y los cupones utilizados (descuentos) comparten la misma clave primaria y solo se diferencian en que la segunda introduce filas por todas aquellas de la primera en las que se asigna un cupón.

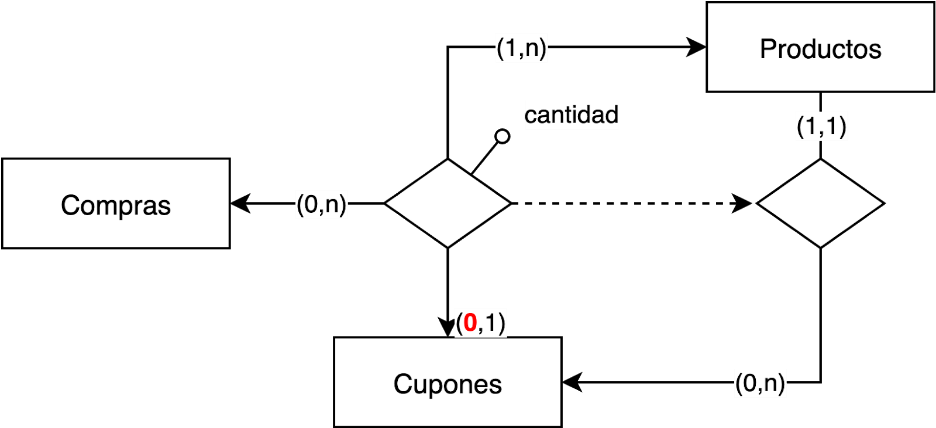
Dicho de otro modo: la tabla de compras por producto ya permitiría almacenar los cupones con simplemente añadir un atributo optativo en que se referencia el cupón utilizado cuando lo haya:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1qerh-xLWOKBZC-5Is7TCDUU60QT5GRFN)

Es reseñable que esta conversión es la equivalente a éste modelo conceptual:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1MSGZU7rLaIcwqFcRNcfpU-O1syHzMEfF)

Donde las cardinalidades se definen de un modo distinto al visto (ya se ha comentado que hay variaciones entre autores sobre el mejor modo de considerar las cardinalidades en interrelaciones superiores a 2).  
  
Además, realizando el modelo conceptual de este modo, podríamos señalar la restricción implícita de que un cupón solo se asocie en su uso con el producto para el que es válido:

[](https://app.diagrams.net/?page-id=oY_-0OO1mKY3sdKwAhEA&scale=auto#G1jD4iX7UN_LLCFvnJmCHtPEypvv5WaTBF)

Banco

Se desea diseñar una Base de Datos para un banco que contenga información sobre los clientes, las cuentas, las sucursales y las transacciones producidas. Construir el modelo E/R teniendo en cuenta las siguientes restricciones generales:

* Un cliente puede tener muchas cuentas
* Una cuenta puede tener muchos clientes
* Una cuenta sólo puede estar en una sucursal
* Una transacción viene determinada por su número de transacción, la fecha y la cantidad

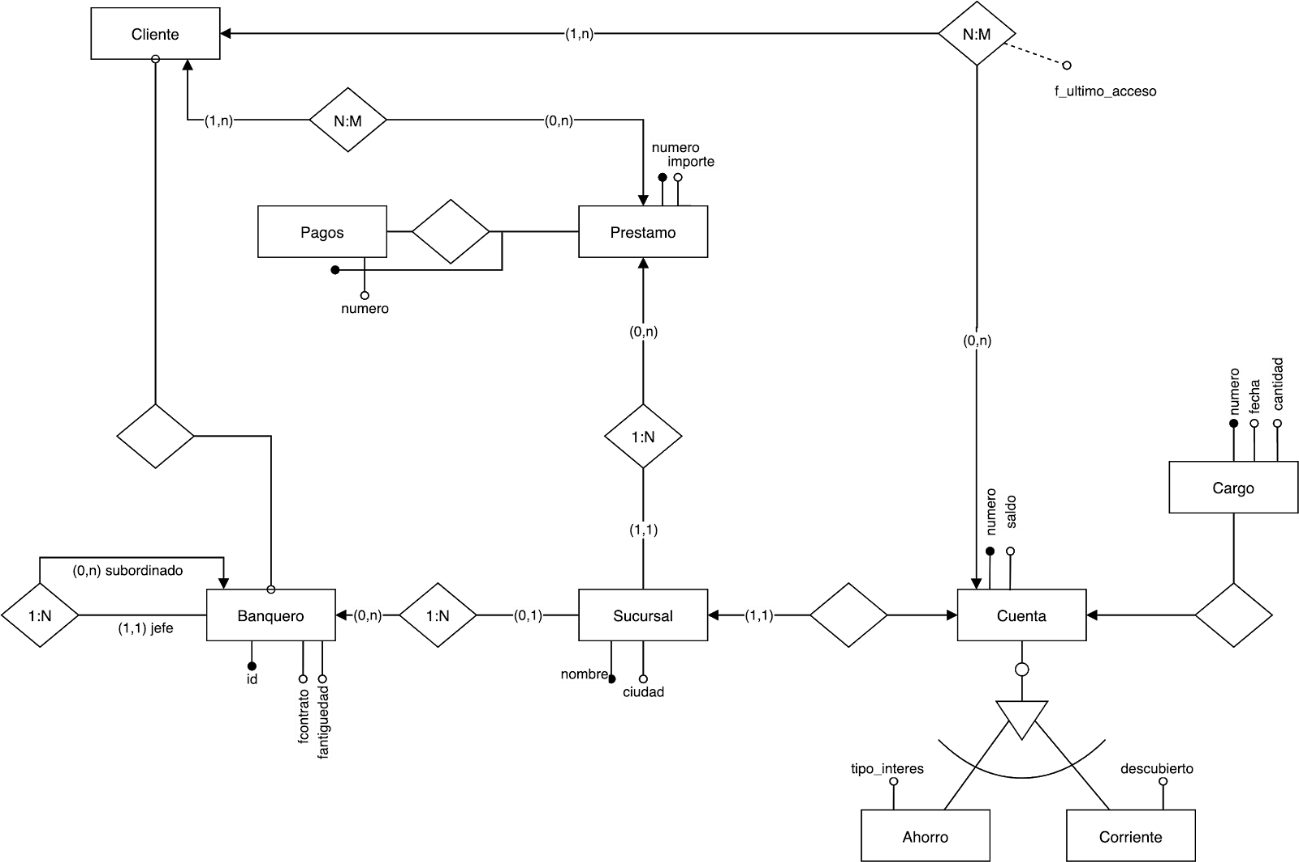
El diseño de dicha base de datos tiene que cumplir con los siguientes requisitos:

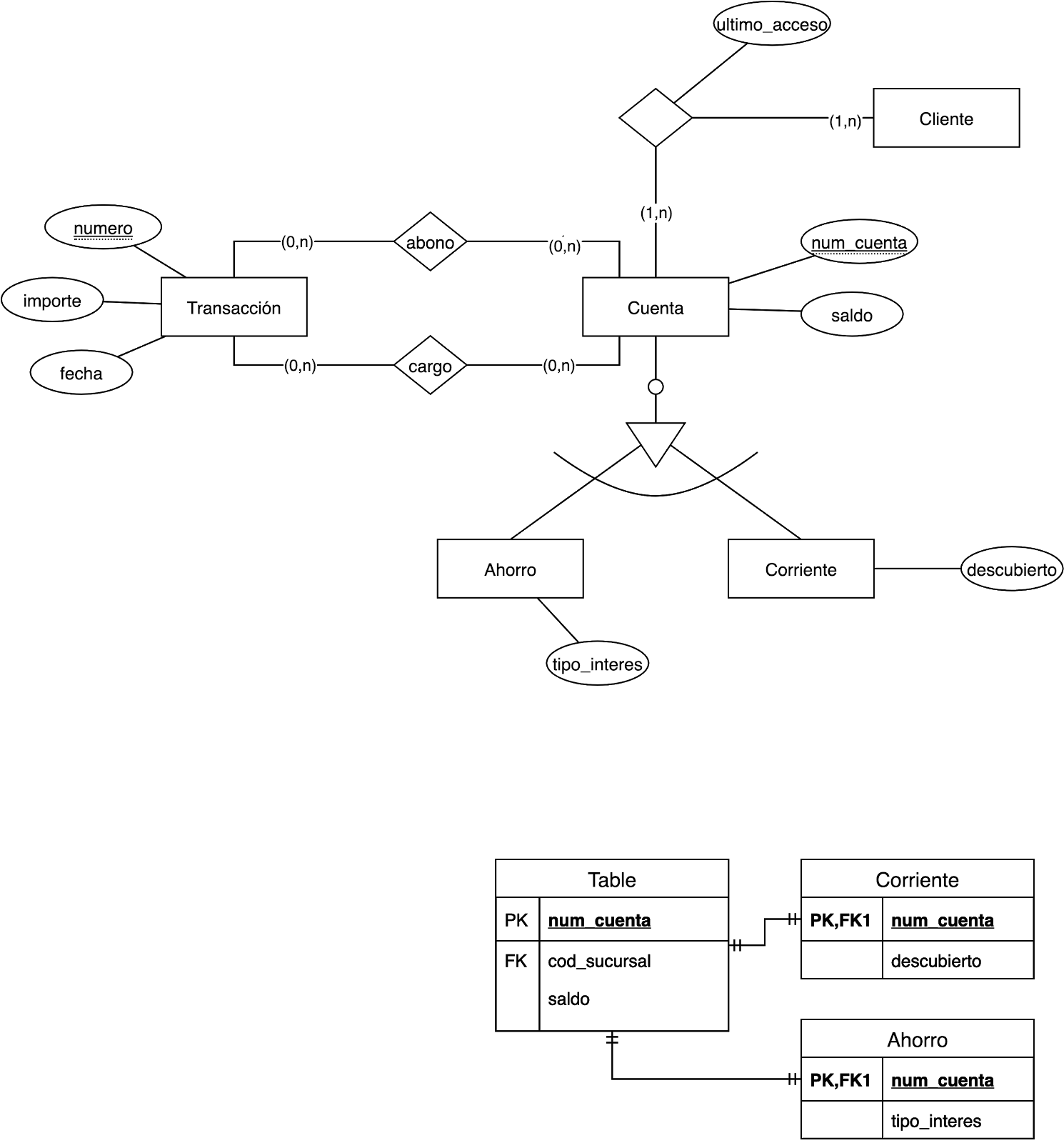
* El banco está organizado en sucursales. Cada sucursal está ubicada en una ciudad particular y se identifica por un nombre único. El banco supervisa los activos de cada sucursal.
* Los clientes del banco se identifican mediante un id-cliente. El banco almacena cada nombre de cliente, calle y ciudad donde viven. Los clientes pueden tener cuentas y pueden pedir préstamos. Un cliente puede estar asociado con un banquero particular, que puede actuar como responsable de préstamos o banquero personal para un cliente.
* Los empleados del banco se identifican mediante un id-empleado. La administración del banco almacena el nombre, número de teléfono de cada empleado, y el id-empleado del jefe del empleado, y la sucursal en la que trabaja. El banco también mantiene registro de la fecha de comienzo del contrato del empleado, así como su antigüedad.
* El banco ofrece dos tipos de cuentas: cuentas de ahorro y cuentas corrientes. Las cuentas pueden asociarse a más de un cliente y un cliente puede tener más de una cuenta. Cada cuenta está asignada a un único número de cuenta. El banco mantiene un registro del saldo de cada cuenta y la fecha más reciente en que la cuenta fue accedida por **cada** cliente que mantiene la cuenta. Además, cada cuenta de **ahorro** tiene un tipo de interés y para cada cuenta **corriente** se almacena el descubierto.
* Un préstamo tiene lugar en una sucursal particular y puede estar asociado a uno o más clientes. Un préstamo se identifica mediante un único número de préstamo. Para cada préstamo el banco mantiene registro del importe del préstamo y de los pagos del préstamo. Aunque un número de pago del préstamo no identifica de forma única un pago entre todos los préstamos del banco, un número de pago identifica un pago particular para un préstamo específico. Para cada pago se almacenan la fecha y el importe.
* El banco mantiene el registro de las transacciones, los **abonos** y **cargos** en las cuentas de ahorros y en las cuentas corrientes almacenando numero de la transacción, fecha e importe.

**Vídeos desarrollando la solución:**

<https://vimeo.com/415705243>

<https://vimeo.com/415708616><https://vimeo.com/415711989><https://vimeo.com/415716012>



Parte de abonos y cargos, detallada:  
  
[](https://app.diagrams.net/?page-id=BEEq4_s7-veDeHbeQkg6&scale=auto#G111JWZCpuOJVg6GdqywD3g9MAlsM3Ad5L)