IES CHAN DO MONTE
C.S. de Desarrollo de Aplicaciones Multiplataforma
Módulo Base de datos

UNIDAD 2 parte 2: Modelo Relacional

Índice

1. MODELO RELACIONAL

Esta unidad trata del que actualmente es el principal modelo para las aplicaciones de procesamiento de datos: el modelo relacional, que presenta una forma muy simple y potente de representar los datos.

El modelo de datos relacional fue desarrollado por ***E.F. Codd*** para IBM a finales de los años sesenta. Propone un modelo basado en la teoría matemática de las relaciones, con el objetivo de mantener la independencia de la estructura lógica respecto al modo de almacenamiento y otras características de tipo físico. Para conseguir esta independencia, ***Codd*** introduce el concepto de **relación** (actualmente conocido como **tabla**) como estructura básica del modelo. Todos los datos de una base de datos se representan en forma de relaciones cuyo contenido varía en el tiempo.

Aunque trabajaba para IBM, esta empresa no recibió de buen grado sus teorías (de hecho continuó trabajando en su modelo en red IMS). De hecho fueron otras empresas (en especial Oracle) las que implementaron sus teorías. Pocos años después el modelo se empezó a utilizar cada vez más, hasta finalmente ser el modelo de bases de datos más popular. Hoy en día casi todas las bases de datos siguen este modelo.

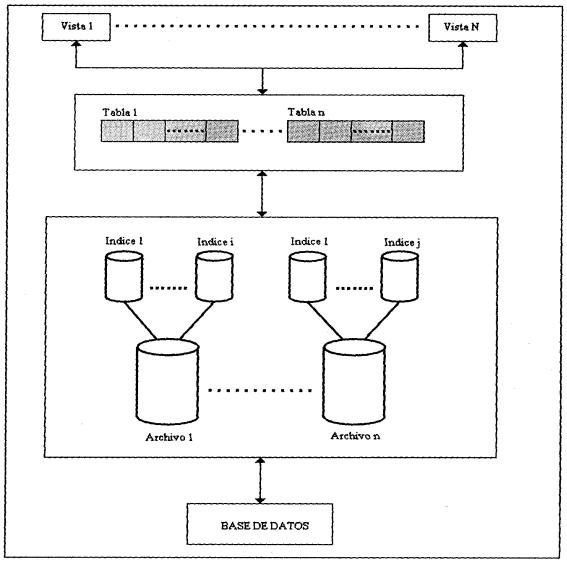
Codd perseguía estos **objetivos** con su modelo:

- **Independencia física**. La forma de almacenar los datos, no debe influir en su manipulación lógica. Si la forma de almacenar los datos cambia, los usuarios no tienen siquiera porque percibirlo y seguirán trabajando de la misma forma con la base de datos. Esto permite que los usuarios y usuarias se concentren en qué quieren consultar en la base de datos y no en cómo está realizada la misma.
- **Independencia lógica**. Las aplicaciones que utilizan la base de datos no deben ser modificadas porque se modifiquen elementos de la base de datos. Es decir, añadir, borrar y suprimir datos, no influye en las vistas de los usuarios. De una manera más precisa, gracias a esta independencia el esquema externo de la base de datos es realmente independiente del modelo lógico.
- **Flexibilidad**. La base de datos ofrece fácilmente distintas vistas en función de los usuarios y aplicaciones.
- **Uniformidad**. Las estructuras lógicas siempre tienen una única forma conceptual (las tablas).
- **Sencillez**. Facilidad de manejo (algo cuestionable, pero ciertamente verdadero si comparamos con los sistemas gestores de bases de datos anteriores a este modelo).

1.1 El modelo relacional y la arquitectura ansi/sparc

Si bien cada SGBDR implementa su arquitectura con elementos propios, los fundamentos teóricos de los tres niveles estándar en un sistema relacional de datos son los siguientes:

- **Esquema conceptual**. Se define mediante ***relaciones*** que se representan con ***tablas***. Cada entidad y relación se corresponde con una tabla bidimensional. Las columnas son los atributos y las filas las ocurrencias. En el resto de la unidad se analizan en profundidad las características de este esquema.
- **Esquema externo.** Se describe mediante ***vistas*** que se corresponden con los subesquemas de



la arquitectura estándar. Una vista es una tabla virtual que se forma a partir de las tablas del esquema conceptual, pero que no tiene correspondencia en el nivel interno.

- **Esquema interno.** Cada tabla del esquema conceptual se almacena en un ***archivo***. Para cada clave candidata se crea un índice para el posterior acceso directo a los datos del archivo.

La imagen anterior resume la arquitectura de un sistema gestor de bases de datos relacional.

Una **base de datos relacional** es una base de datos en donde todos los datos visibles al usuario están organizados estrictamente como tablas de valores, y en donde todas las operaciones de la base de datos operan sobre estas tablas

2. Estructura de las bases de datos relacionales

2.1 Relación o tabla

El elemento principal del modelo relacional de datos es la **relación**, aunque más habitualmente se le llama tabla. Cada relación se representa mediante una *tabla bidimensional* y consta de:

- **Atributos**. Referido a cada propiedad de los datos que se almacenan en la relación (nombre, dni,...).
- **Tuplas**. Referido a cada elemento de la relación. Por ejemplo si una relación almacena personas, una tupla representaría a una persona en concreto.

Puesto que una relación se representa como una tabla; podemos entender que las columnas de la tabla son los atributos; y las filas, las tuplas.

Atributo1	Atributo2	Atributo3	. . .	Atributo n	
Valor 1,1	Valor 1,2	Valor 1,3	. . .	Valor 1,n	←tupla 1
Valor 2,1	Valor 2,2	Valor 2,3	. . .	Valor 2,n	←tupla 2
. . .	. . .	. . .	. . .	. . .	
Valor m,1	Valorm,2	Valorm,3	. . .	Valorm,n	←tupla m

2.2 Tupla

Cada una de las filas de la relación. Se corresponde con la idea clásica de registro. Representa por tanto cada elemento individual de esa relación. Tiene que cumplir que:

- Cada tupla se debe corresponder con un elemento del mundo real.
- **No puede haber dos tuplas iguales** (con todos los valores iguales).

2.3 Dominio de atributo

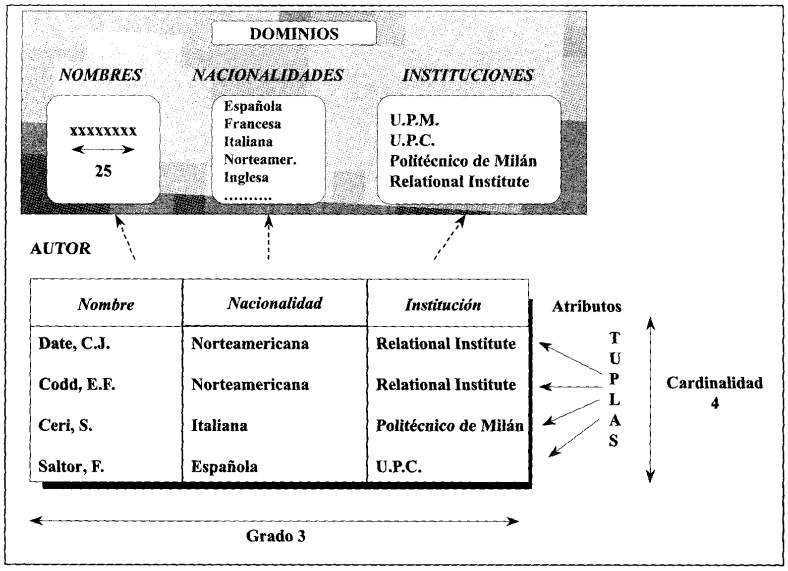
Es el *conjunto de valores que puede tomar dicho atributo*. Por ejemplo, el dominio del atributo Garantía es un número comprendido entre 3 y 18 que representa el número de meses mínimo y máximo que ofrece el concesionario como garantía del coche, Sexo puede tomar los valores "masculino" y "femenino", Color admitirá los colores válidos para el entorno de la relación, etc.

2.4 Grado

Indica el tamaño de una relación en base al *número de columnas* (atributos) de la misma. Todas las tuplas tienen el mismo número de atributos.

2.5 Cardinalidad

Número de tuplas de una relación o *número de filas* de una tabla.



2.6 Un esquema de relación

Un **esquema de relación R,** se compone de un nombre de relación R, de un conjunto de n atributos {Ai} y de un conjunto de n dominios (no necesariamente distintos) {Di}, donde cada atributo será definido sobre un dominio:

R (Al : D1, A2: D2, ... An : Dn)

También se acostumbra a usar el término **intensión** de una relación para el esquema de relación. La intensión es la parte de definición de las propiedades de la relación (atributos) sobre un dominio de datos y hace referencia a la estructura estática (invariante en el tiempo) de la relación.

Al conjunto de tuplas que, en un instante determinado, satisfacen el esquema de relación y se encuentran almacenadas en la base de datos, es lo que se suele llamar relación o también **extensión**. La extensión varía en el transcurso del tiempo.

ESQUEMA DE RELACIÓN (INTENSIÓN):

AUTOR (*Nombre*: *Nombres*, *Nacionalidad*: *Nacionalidades*, *Institución*: *Instituciones*)

RELACIÓN (EXTENSIÓN, ESTADO u OCURRENCIA):

AUTOR

Nombre	*Nacionalidad*	*Institución*
Date, C.J.	**Norteamericana**	**Relational Institute**
De Miguel, A.	**Española**	**UC3M**
Ceri, S.	**Italiana**	**Politécnico de Milan**

2.7 Sinónimos

Los términos vistos anteriormente tienen distintos sinónimos según la nomenclatura utilizada. A ese respecto se utilizan tres nomenclaturas:

Nomenclatura relacional		Nomenclatura tabla		Nomenclatura ficheros
Relación	=	Tabla	=	Fichero
Tupla	=	Fila	=	Registro
Atributo	=	Columna	=	Campo
Grado	=	Nº de columnas	=	Nº de campos
cardinalidad	=	Nº de filas	=	Nº de registros

2.8 Propiedades de las tablas (o relaciones)

En general, una tabla debe reunir los siguientes requisitos para ser considerada como una relación:

- Cada tabla tiene un ***nombre distinto***
- Tener un ***número fijo de atributos*** para todas las tuplas.
- Cada atributo tiene ***un único dominio*****.**
- El ***orden de las tuplas no importa***. Podemos especificar muchos ordenamientos lógicos en una relación, por ejemplo, para la relación estudiante podríamos ordenar lógicamente por Codigo_estudiante, fecha_nacimiento, nombre,etc...
- El ***orden de los atributos no importa***
- ***No*** pueden existir tuplas ni atributos repetidos.
- Cada intersección fila-columna debe contener ***un valor único*** perteneciente al dominio de la columna correspondiente.

2.9 Claves

CLAVE CANDIDATA

Conjunto de ***atributos que identifican unívocamente cada tupla*** de la relación. Es decir columnas cuyos valores no se repiten en ninguna otra tupla de esa tabla. Toda tabla en el modelo relacional debe tener al menos una clave candidata.

CLAVE PRIMARIA

Clave candidata que ***se escoge como identificador*** de las tuplas. Se elige como primaria la candidata que identifique mejor a cada tupla en el contexto de la base de datos.

Por ejemplo un campo con el DNI sería clave candidata de una tabla de clientes, si esa tabla tiene un campo de código de cliente, éste sería mejor candidato (y por lo tanto clave principal) porque es mejor identificador para ese contexto.

CLAVE ALTERNATIVA

Cualquier clave candidata que no sea primaria.

CLAVE EXTERNA, FORÁNEA, AJENA O SECUNDARIA

Son los **datos de atributos de una tabla cuyos valores están relacionados con atributos de otra tabla**. Por ejemplo en la tabla equipos tenemos estos datos:

Equipo	Num_equipo
Real Madrid	1
F.C. Barcelona	2
Athletic Bilbao	3

En la tabla anterior la clave principal es el atributo **nº equipo**. En otra tabla tenemos:

Nº Jugador	Jugador	Num_equipo
1	Muniain	3
2	Messi	2
3	Piqué	2
4	Ronaldo	1

El atributo **Nº Equipo** sirve para relacionar el Jugador con el equipo al que pertenece. Ese campo en la tabla de jugadores es una clave secundaria.

2.10 Nulos

En los lenguajes de programación se utiliza el valor nulo para reflejar que un identificador (una variable, un objeto,..) no tiene ningún contenido.

Las bases de datos relacionales permiten más posibilidades para el valor nulo (***null***), aunque su significado no cambia: ***valor vacío***. En claves secundarias indican que el registro actual ***no está relacionado*** con ninguno. En otros atributos indica que la tupla en cuestión ***carece de dicho atributo***: por ejemplo en una tabla de personas un valor nulo en el atributo teléfono indicaría que dicha persona no tiene teléfono.

Es importante indicar que ni el texto vacío ‘ ’ ni el cero significa lo mismo en un texto que el nulo.

Puesto que ese valor se utiliza continuamente, resulta imprescindible saber cómo actúa cuando se emplean operaciones lógicas sobre ese valor. Eso significa definir un tercer valor en la lógica booleana, además de los clásicos verdadero y falso. Un valor nulo no es ni verdadero ni falso.

El uso de operadores lógicos con el nulo da lugar a que:

- ***verdadero AND nulo*** da como resultado, nulo
- ***falso AND nulo*** da como resultado, falso
- ***verdadero OR nulo*** da como resultado, verdadero
- ***falso OR nulo*** da como resultado nulo
- la ***negación de nulo***, da como resultado nulo

Se utiliza un operador en todas las bases relacionales llamado es nulo ***(is null)*** que devuelve verdadero si el valor con el que se compara es nulo.

3. Restricciones

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

3.1 Inherentes

Las restricciones inherentes vienen impuestas por el propio Modelo de Datos. En el modelo relacional son aquellas que no son determinadas por los usuarios, sino que son definidas por el hecho de que la base de datos sea relacional. Las más importantes son:

- **No puede haber dos tuplas iguales**
- **El orden de las tuplas no es significativo**
- **El orden de los atributos no es significativo**
- **Cada atributo sólo puede tomar un valor en el dominio en el que está inscrito.**

3.2 Semánticas

El modelo relacional permite a los usuarios incorporar restricciones personales a los datos. Se comentan las diferentes reglas semánticas a continuación:

3.2.1 Clave principal (primary key)

Ningún valor de la clave primaria de una relación puede ser nulo o desconocido. Marca uno o más atributos como identificadores de la tabla. De esa forma en esos atributos las filas de la tabla no podrán repetir valores ni tampoco dejarlos vacíos.

3.2.2 Unicidad (unique)

Impide que los valores de los atributos marcados de esa forma, puedan repetirse. Esta restricción debe indicarse en todas las claves alternativas. Para evitar violar esta regla, se comprobará en todas las inserciones y modificaciones que el valor de todas las claves candidatas no existe en alguna tupla de esa relación.

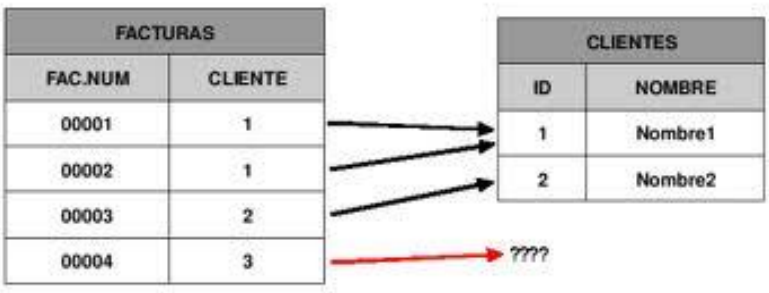
Al marcar una clave primaria se añade automáticamente sobre los atributos que forman la clave un criterio de unicidad.

3.2.3 Obligatoriedad (not null)

Prohíbe que el atributo marcado de esta forma quede vacío (es decir impide que pueda contener el valor nulo, null)

3.2.4 Integridad referencial (foreign key)

Sirve para indicar una **clave externa** (también llamada secundaria y foránea) sobre uno o más atributos. Los atributos marcados de esta forma **sólo podrán contener valores que estén relacionados con la clave principal de la tabla que relacionan** (llamada tabla principal). Dichos atributos sí podrán contener valores nulos.



Es decir, si tenemos una tabla de *facturas* en la que cada fila es una factura, existirá un atributo *cliente* que indicará el *código del cliente* y que estará relacionado con una tabla de *clientes*, en la que dicho atributo es la clave principal. De hecho no se podrá incluir un código que no esté en la tabla clientes, ya que en ese caso no sabríamos a quien cobrarle esa factura; eso es lo que prohíbe la integridad referencial.

Eso causa problemas en las operaciones de borrado y modificación de registros; ya que si se ejecutan esas operaciones sobre la tabla principal (si se modifica o borra un cliente) quedarán filas en la tabla secundaria

con la clave externa haciendo referencia a un valor que ya no existe en la tabla principal.

Para solventar esta situación se puede hacer uso de estas opciones:

- **Prohibir la operación (*no action*)**.
- **Transmitir la operación en cascada** (***cascade***). Es decir si se modifica o borra un cliente; también se modificarán o barrarán los alquileres relacionados con él.
- **Colocar nulos** (***set null***) Las referencias al cliente en la tabla de alquileres se colocan como nulos (es decir, alquileres sin cliente).
- **Usar el valor por defecto (*default*)**. Se colocan un valor por defecto en las claves externas relacionadas. Este valor se indica al crear la tabla (opción default).

3.2.5 Regla de validación (check)

Condición lógica que debe de cumplir un dato concreto para darlo por válido. Por ejemplo restringir el campo sueldo para que siempre sea mayor de 1000, sería una regla de validación. También por ejemplo que la fecha de inicio sea mayor que la fecha final.

3.2.6 Disparadores o triggers

Se trata de **pequeños programas grabados en la base de datos que se ejecutan automáticamente cuando se cumple una determinada condición**. Sirven para realizar una serie de acciones cuando ocurre un determinado evento (cuando se añade una tupla, cuando se borra un dato, cuando un usuario abre una conexión…)

Los **triggers** permiten realizar restricciones muy potentes; pero son las más difíciles de crear.

4. Transformación de un esquema E/R a un esquema relacional

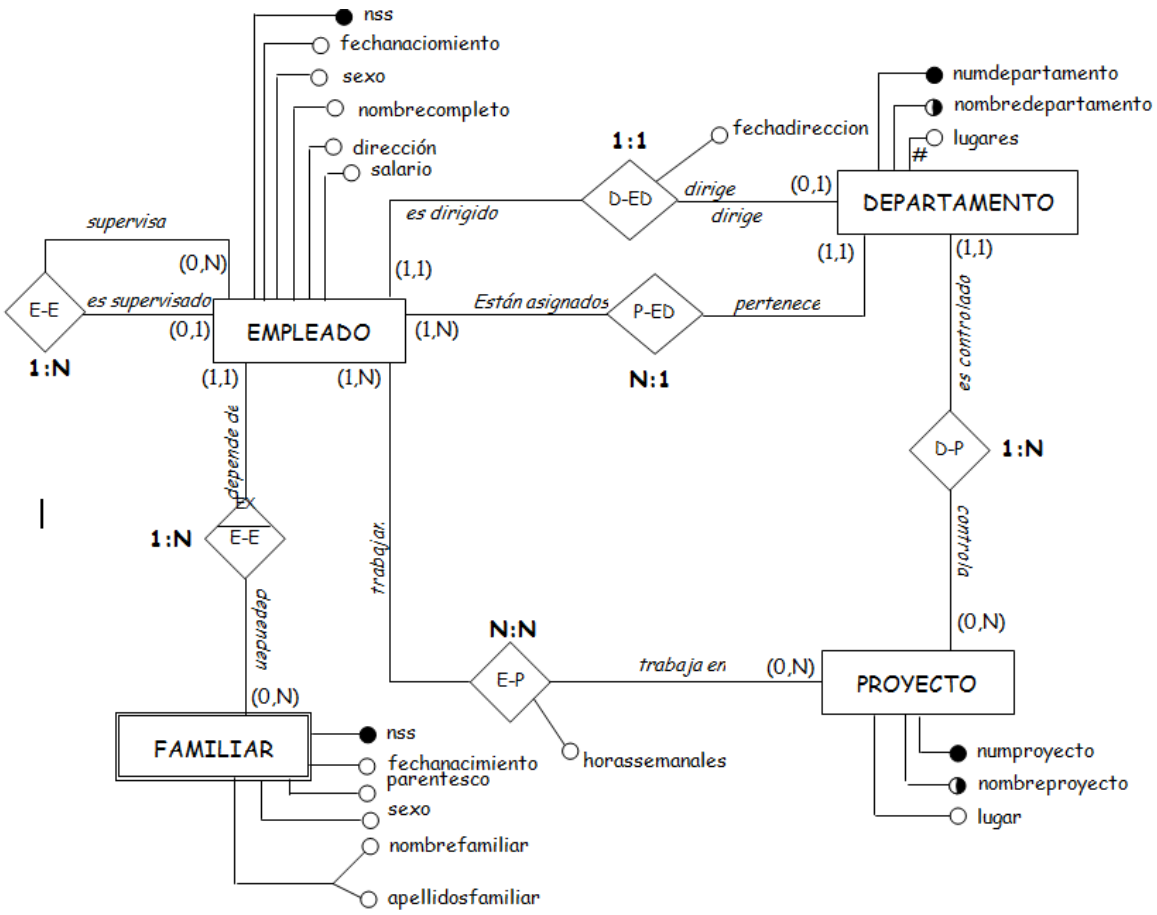
Una vez obtenido el esquema conceptual mediante el modelo E/R, hay que abordar **el diseño lógico** que generaba el esquema de la base de datos en el modelo lógico elegido (relacional, jerárquico, de redes u orientado a objetos).

Los pasos que se ven a aplicar en el diseño lógico son los siguientes:

- **Transformar el diagrama E/R** (esquema conceptual) **al modelo relacional** (esquema lógico)
- Aplicar **reglas de normalización.** (Para diseñar relaciones bien estructuradas)

Metodología para realizar la conversión

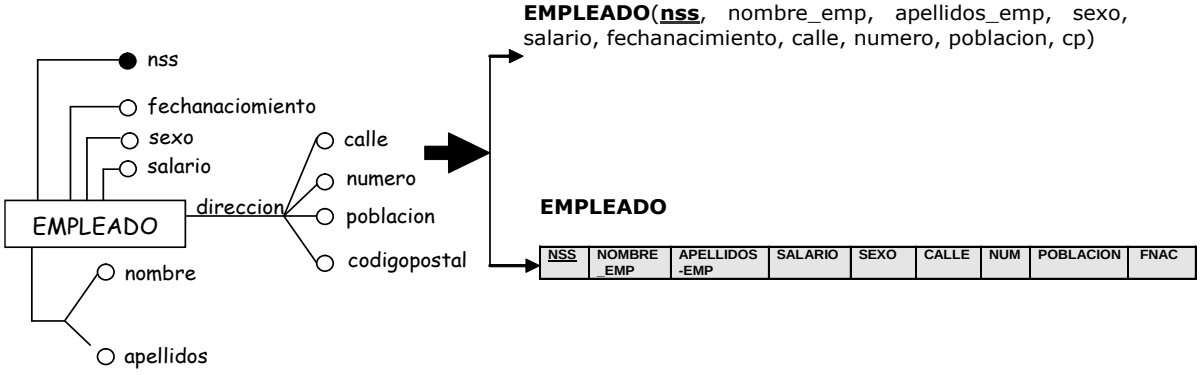
Vamos a trabajar con el siguiente diagrama entidad/relación:



4.1 Transformación de las entidades fuertes

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

- **Entidades**. Las entidades pasan a ser tablas
- **Atributos**. Los atributos pasan a ser columnas o atributos de la tabla.
  - Los atributos compuestos no se pueden representar con el modelo relacional, así que se descompondrán en atributos simples.
- **Identificadores principales**. Pasan a ser claves primarias
- **Identificadores candidatos**. Pasan a ser claves candidatas.



Así, de nuestro ejemplo obtendríamos las siguientes tablas:

EMPLEADO (**NSS**, NOMBREEMPLEADO, APELLIDOSEMPLEADO, SEXO, SALARIO, CALLE, NUMERO, CP, POBLACION, FECHANACIMIENTO)

DEPARTAMENTO (**NUMDEPARTAMENTO**, NOMBREDEPARTAMENTO)

PROYECTO (**NUMEROPROYECTO**, NOMBREPROYECTO, LUGAR)

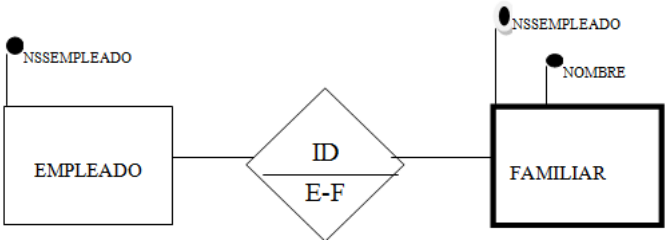
4.2 Transformación de las entidades débiles

Toda entidad débil da lugar a una relación, que incorpora los mismos elementos que una entidad fuerte y además debe:

- Incluir como **clave foránea** los atributos que forman la clave primaria de la tabla fuerte de la que dependen.
- En ocasiones el identificador de la entidad débil tiene como parte de su identificador al identificador de la entidad fuerte (por ejemplo si para identificar líneas de factura utilizamos el número de línea y el número de factura, clave de la entidad factura). En esos casos no hace falta añadir de nuevo como clave externa el identificador de la entidad fuerte.

- **La clave primaria** de la entidad débil estará formada:
  - En caso de tratarse de una **debilidad por existencia**: por los atributos que forman la clave primaria de W.

    FAMILIAR (**NSSFAMILIAR**, FECHANACIMIENTO, NOMBREFAMILIAR, APELLIDOSFAMILIAR, PARENTESCO, SEXO ,***NSSEMPLEADO***)
  - En caso de tratarse de una **debilidad por identificación**: por los atributos de la clave foránea más los atributos que formen la clave parcial de la entidad débil.



FAMILIAR (**NSSEMPLEADO, NOMBRE** FECHANACIMIENTO, NOMBREFAMILIAR, APELLIDOSFAMILIAR, PARENTESCO, SEXO ,)

4.3 Relaciones N:M

Se crea una nueva relación (tabla) para el tipo de relación que incluye:

- Todos los **atributos clave** de las relaciones que participan en el tipo de relación
- Todos los **atributos propios** del tipo de la relación

En general la clave primaria de la relación tipo estará formada por los ***atributos clave de las dos relaciones*** que participan en tipo de la relación.

En la relación E-P (TRABAJA-EN) entre EMPLEADO Y PROYECTO:

EMPLEADOPROYECTO (**NSSEMPLEADO**, **NUMEROPROYECTO**, NUMHORAS)

4.4 Relación binaria 1:N

A.-En la mayor parte de los casos **se *propaga la clave***, esto es, se propaga el atributo principal de la entidad que tiene cardinalidad máxima 1 a la que tiene cardinalidad máxima N. Si existen atributos propios de la relación, estos también se propagan.

En nuestro ejemplo:

- En la relación: P-ED (TRABAJA-EN) entre DEPARTAMENTO y EMPLEADO se añade en la relación EMPLEADO la clave foránea de DEPARTAMENTO

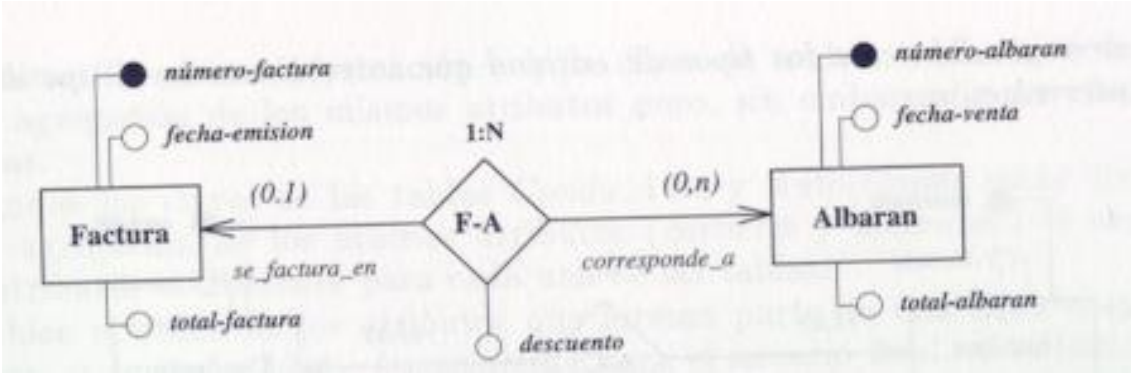
  EMPLEADO (**NSS**, NOMBREEMPLEADO, APELIDOSEMPLEADO, SEXO, SALARIO, CALLE, NUMERO, CP, POBLACION, FECHANACIMIENTO, ***NUMDEPARTAMENTO***)

- En la relación D-P (CONTROLA): Se añade en la relación PROYECTO la clave foránea de DEPARTAMENTO.

  PROYECTO (**NUMEROPROYECTO**, NOMBREPROYECTO, LUGAR, ***NUMDEPARTAMENTO***)

B.-En algunos casos es conveniente ***transformar la relación en una tabla***. Esta solución se toma cuando la cardinalidad es opcional **(0,1) y (0,N)** y se prevé que en la mayor parte de los ***casos no se produce esta relación***, lo cual llevaría a tener **muchos nulos** en el/los atributos propagados.

- Esta tabla **estará formada por los identificadores de los tipos de entidad** que intervienen en el tipo de interrelación y por todos **los atributos** asociados al tipo de interrelación.
- La **clave principal** de esta tabla será el atributo identificador correspondiente al tipo de entidad que interviene con cardinalidad máxima N, y será necesario definir como claves foráneas los atributos identificadores correspondientes a los dos tipos de entidad.



Si aplicamos a este esquema conceptual esta regla, se obtendría el esquema relacional:

Factura (**número-factura**, fecha-emisión, total-factura)
Albaran (**número-albarán**, fecha-venta, total-albarán)
Fact-Alba (**número-albarán**, ***número-factura***, descuento)

Se puede observar que la regla **propagar la clave puede ser también aplicada a este caso**. El que se aplique una u otra regla va a depender, principalmente, del número de instancias del tipo de interrelación. En este caso, el esquema relacional quedaría:

Factura (**número-factura**, fecha-emisión, total-factura)
Albaran (**número-albarán**, fecha-venta, total-albarán, ***número-factura***, descuento)

Como se puede apreciar, ambos esquemas representan fielmente el problema representado en el esquema conceptual.

4.5 Relación binaria 1:1

Se puede considerar un caso particular de la anterior, y por tanto, las soluciones anteriores son válidas también en este caso.

Las soluciones aplicables son:

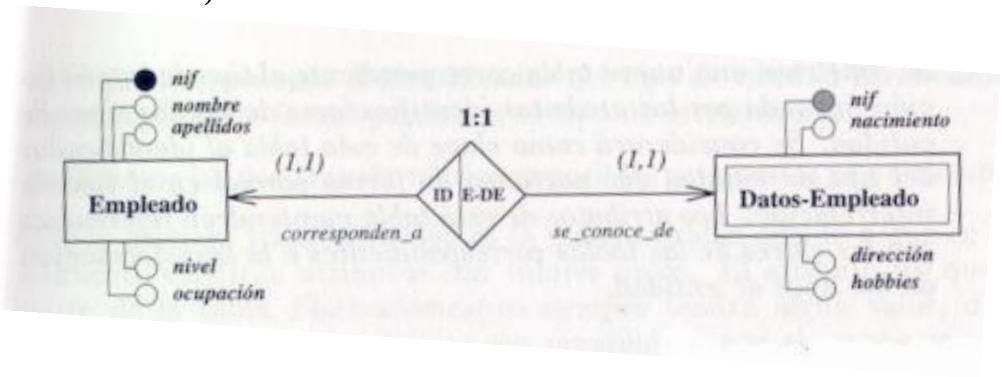
a) **Propagar una de las claves**
b) **Crear una nueva relación**
c) **Propagar las claves de las dos entidades (mutuamente).**
d) **Fusionar ambas entidades (relacionadas) en una sola relación**

El proceso de transformación de los tipos de relación binarias en los cuales los tipos de entidad participan con **cardinalidad máxima uno (1:1)** va a depender del valor de la **cardinalidad mínima** con la cual participa cada tipo de entidad en el tipo de relación. En base a este valor se pueden presentar los siguientes casos:

- Los **dos tipos de entidad** participan de forma **completa** en el tipo de relación. Es decir, los dos tipos de entidad participan con cardinal mínima uno. Caso (1,1) Y (1:1).
- **Uno de los dos tipos de entidad** participa de forma **parcial en el tipo relación**; es decir, con cardinalidad mínima cero. Caso (0,1) y (1:1).
- **Los dos tipos de entidad** participan de forma **parcial** en el tipo de relación. Caso (0,1) y (0,1)

A. **Si en un tipo de interrelación binaria los dos tipos de entidad participan de forma completa**; es decir, ambos tipos de entidad participan con las cardinalidades mínima y máxima igual a uno, entonces:

- Si los dos tipos de entidad **tienen el mismo identificador**: ***Se fusionan ambas entidades en una sola relación***:
  - Los dos tipos de entidad se transforman en **una única tabla** formada por la agregación de los atributos de los dos tipos de entidad.
  - La **clave primaria** de la tabla es el **identificador** de los tipos de entidad (el mismo en ambas).



- En este caso, se deriva el siguiente esquema relacional:

  Empleado (**nif**, nombre, apellidos, nivel, ocupación, dirección, nacimiento, hobbies)

- Puede ser también necesario considerar, por motivos de procesamiento, más conveniente la existencia de tablas independientes para cada tipo de entidad. En este caso el esquema relacional resultante quedará de la forma:

  Empleado (**nif**, nombre, apellidos, nivel, ocupación)

  Datos-Empleados(**nif**, dirección, nacimiento, hobbies)

- Si los tipos de entidad **tienen diferente identificador**: **Se propagan las claves de las dos entidades mutuamente** y :
  - Cada tipo de entidad se transforma **en una tabla**.
  - Cada tabla tendrá como **clave principal** el identificador de cada uno de los tipos de entidad de los cuales se deriva.
  - Cada tabla tendrá como **clave foránea** el identificador del otro tipo de entidad con el cual está relacionado.

B. **Si en un tipo de interrelación binaria alguno de los tipos de entidad participa de forma parcial**, entonces, cada tipo de entidad se transforma en una tabla y se procede a **propagar una de las claves:**

- Cada tipo de **entidad** se transforma en **una tabla**
- Si una de las entidades posee cardinalidad (0,1), y la otra (1,1) conviene propagar **la clave de la entidad con cardinalidad (1,1)** a la otra tabla resultante de la entidad de cardinalidad (0,1).
- Si hay atributo de la relación también se propaga junto al atributo identificador.

Ej: **Tipo de relación- D-ED (DIRIGE).** Se elige la relación DEPARTAMENTO, con grado de participación total para incluir la clave foránea de EMPLEADO y el atributo Fecha de Inicio del Empleado como director del departamento.

EMPLEADO (**NSSEMPLEADO**, NOMBREEMPLEADO, APELLIDOSEMPLEADO, SEXO, SALARIO, CALLE, NUMERO, CP, POBLACION, FECHANACIMIENTO)

DEPARTAMENTO (**NUMDEPARTAMENTO**, NOMBREDEPARTAMENTO, ***NSSEMPLEADO***, FECHADIRECCION)

Esta **transformación es la más favorable** debido a que en **ningún momento existirán atributos con valores nulos**. El atributo NSSque forma parte de la tabla EMPLEADO siempre tendrá algún valor, debido a que todos los departamentos son dirigidos por al menos uno y como máximo un empleado.

- Si **se hubiera realizado el proceso inverso**, es decir, el identificador del tipo de entidad DEPARTAMENTO hubiera pasado a formar parte de la tabla correspondiente al tipo de entidad EMPLEADO, se hubiera obtenido un esquema relacional de la forma:

EMPLEADO (**NSS,** NOMBREEMPLEADO, APELLIDOSEMPLEADO, SEXO, SALARIO, CALLE, NUMERO, CP, POBLACION, FECHANACIMIENTO,NUMDEPARTAMENTO, FECHADIRECCION)

DEPARTAMENTO (**NUMDEPARTAMENTO**, NOMBREDEPARTAMENTO)

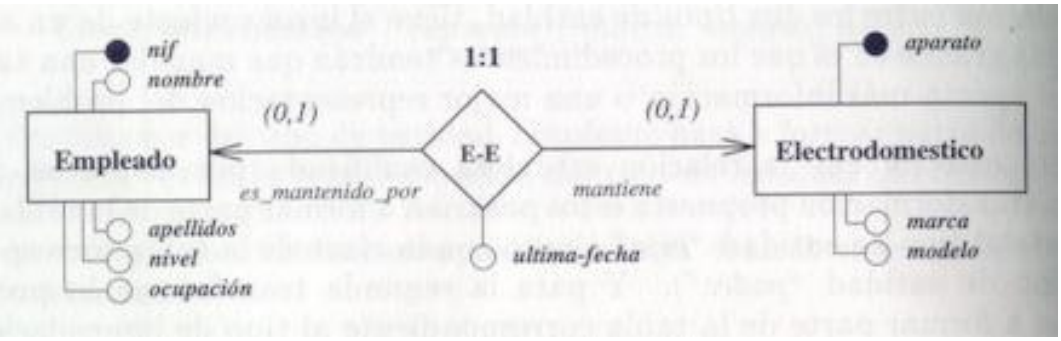
Se puede observar que en la tabla EMPLEADO, los atributos NUMDEPARTAMENTO y FECHADIRECCION **tomarán valores nulos** para todas aquellas tuplas de empleados que no dirijan ningún departamento.

C. **Si en un tipo de relación binaria ambos tipos de entidad participan de forma parcial**, cada una de las entidades se transforma en una tabla y se procede de alguna de las dos formas siguientes:

- **Crear una nueva relación**:

Si una las dos entidades poseen cardinalidad (0,1), se puede proceder a crea una nueva relación para evitar tener valores nulos en las claves foráneas.

- Se construye una **nueva tabla** correspondiente al tipo de relación y cuyos atributos serán los **identificadores de los dos tipos de entidad**.
- La **clave primaria** de la tabla generada será el identificador de uno de los tipos de entidad y, necesariamente, se definirá como clave alternativa al identificador del otro tipo de entidad.



Empleado (**nif**, nombre, apellidos, nivel, ocupación)

Electrodoméstico (**aparato**, marca, modelo)

Empl-Elect(**nif**, ***aparato***, ultima-fecha)

- **Se propagan las claves de las dos entidades mutuamente:**
  - Los identificadores de cada uno de los tipos de entidad pasan a formar parte como atributos de las tablas correspondientes al otro tipo de entidad. Estos atributos actuarán como claves foráneas en estas tablas.

    Empleado (**nif**, nombre, apellidos, nivel, ocupación, ***aparato***)

    Electrodoméstico (**aparato**, marca, modelo, ***nif***, última-fecha)

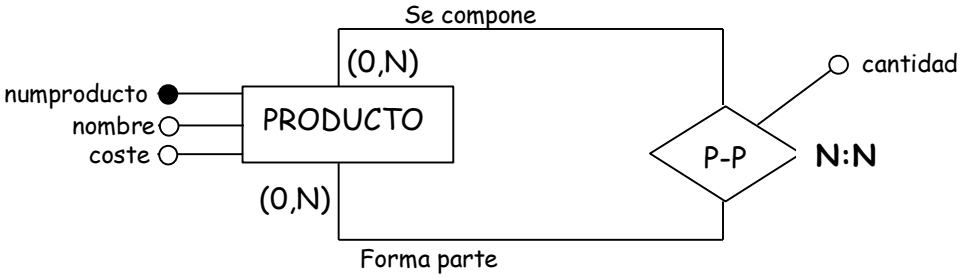
4.6 Relaciones reflexivas o recursivas

Las relaciones recursivas se tratan de la misma forma que las otras, sólo que hay que imaginar que la tabla se divide en dos, una por cada rol. Teniendo en cuenta eso, la solución es idéntica a lo ya resuelto en los casos anteriores.

En el ejemplo: la relación recursiva **E-E (**SUPERVISA), se representa añadiendo a la relación EMPLEADO un atributo que contendrá valores de clave primaria existentes en empleado

EMPLEADO (**NSS**, NOMBREEMPLEADO, APELIDOSEMPLEADO, SEXO, SALARIO, CALLE, NUMERO, CP, POBLACION, FECHANACIMIENTO, NUMDEPARTAMENTO, ***NSSSUPERVISOR***)

A continuación vemos un ejemplo de transformación de una relación reflexiva N:M, donde la tabla resultante de la relación contendrá dos veces la clave primaria de la entidad, más los atributos de la relación si los hubiera. La clave de esta nueva tabla será la combinación de las dos claves.



PRODUCTO (**NUMPRODUCTO**, NOMBRE, COSTE)

PRODUCTOCONSTA (**NUMPRODUCTO**, **NUMCONSTAPRODUCTO**, CANTIDAD)

4.7 Atributos multivaluados

El modelo relacional no permite dominios multivalorados, ya que los datos deben ser atómicos.

Para los atributos multivaluados se **creará una tabla** que contendrá:

- La **clave primaria** de la relación que contiene el atributo multivaluado.
- Un atributo que se corresponde con el **atributo multivaluado**.

La **clave primaria** de la nueva relación estará formada por la **clave primaria junto con el atributo multivaluado.** Este atributo multivalorado será clave foránea referenciado a la tabla primaria

Por ejemplo: La entidad tipo DEPARTAMENTO tiene el atributo multivaluado lugares. Se crea la siguiente relación:

DEPARTAMENTO_UBICA (**NUMDEPARTAMENTO**, **LUGAR**)

4.8 Atributos derivados

No existe una representación directa, por tanto, **se deben tratar como atributos simples**, que pasaran a ser columnas de la tabla correspondiente.

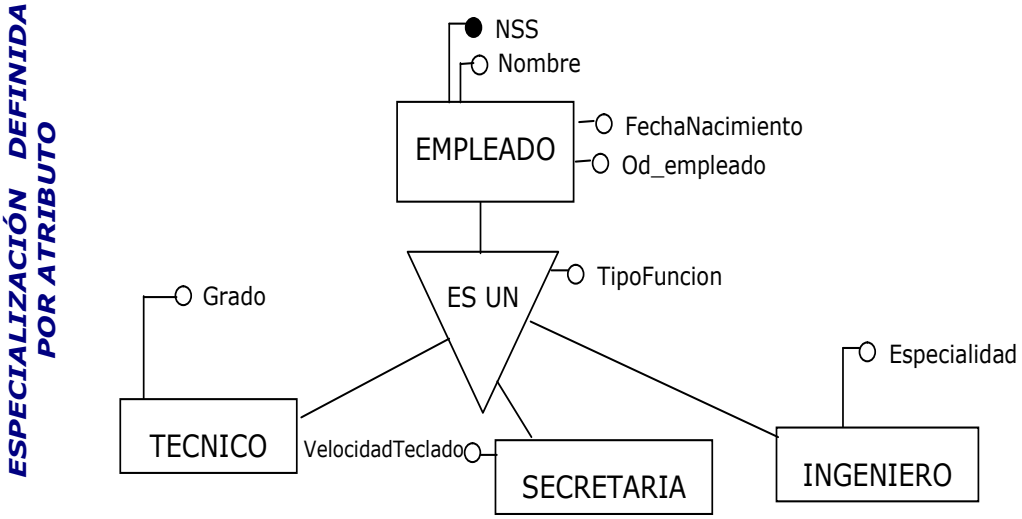
Se deberan construir **disparadores (trigger)** que calcule el valor del atributo derivado cada vez que se inserten o se borren las ocurrencias de los atributos que intervienen en el cálculo y añadir las restricciones correspondientes.

4.9 Especializaciones/Generalizaciones

Existen tres posibilidades para la transformación de jerarquías al modelo relacional:

- ***Integrar todas las entidades en una única, eliminando los subtipos,*** Esta nueva entidad contendrá todos los atributos del supertipo, todos los del subtipo, y los atributos discriminativos que permiten distinguir a qué subentidad pertenece cada atributo. Todas las relaciones se mantienen con la nueva entidad.

  Esta regla puede aplicarse a cualquier **tipo de jerarquía**. La gran ventaja es la simplicidad pues todo se reduce a una entidad. El gran inconveniente es que se generan demasiados valores nulos en los atributos opcionales propios de cada entidad.



EMPLEADO (**NSS**, NOMBRE, APELLIDOS, FECHANACIMIENTO, OD_EMPLEADO, TIPOFUNCION,VELOCIDADTECLEO,GRADO,ESPECIALIDAD).

- ***Eliminación del supertipo,*** transfiriendo los atributos del supertipo a cada uno de los subtipos. Las relaciones del supertipo se consideran para cada uno de los subtipos. La clave genérica del supertipo pasa a cada uno de los subtipos. Sólo puede ser aplicada para **jerarquías totales y**

exclusivas. Inconvenientes que presenta esta transformación:

(1) Se crea redundancia ya que los atributos del supertipo se repiten en cada uno de los subtipos.

(2) El número de relaciones aumenta, ya que si el supertipo tiene relaciones estas pasan a cada uno de los subtipos.

SECRETARIA (**NSS**, NOMBRE, APELLIDOS, FECHANACIMIENTO, OD_EMPLEADO, VELOCIDADTECLEO)

TECNICO (**NSS**, NOMBRE, APELLIDOS, FECHANACIMIENTO, OD_EMPLEADO, GRADO)

INGENIERO (**NSS**, NOMBRE, APELLIDOS, FECHANACIMIENTO, OD_EMPLEADO, ESPECIALIDAD)

- ***Se crea una relación por cada entidad participante en la jerarquía,*** (una para el supertipo y otra para cada uno de los subtipos), de tal forma que el supertipo propaga su Identificador principal a cada uno de los subtipos que pasan a identificarse por el mismo identificador como clave ajena (foránea). La relación creada para el supertipo podrá contener el atributo discriminante de la jerarquía. Es la opción **más utilizada de forma general**.

  EMPLEADO (**NSS_EMP**, NOMBRE, APELLIDOS, FECHANACIMIENTO, OD_EMPLEADO, *TIPOFUNCION*)

  SECRETARIA (**NSS**,VELOCIDADTECLEO)

  TECNICO (**NSS**,GRADO)

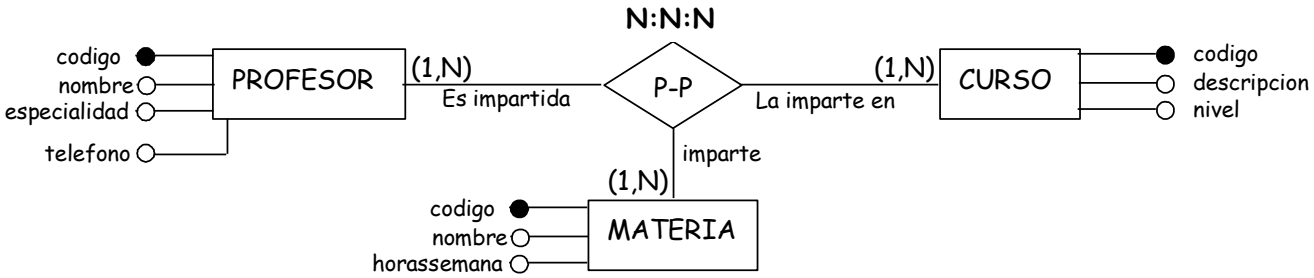
  INGENIERO (**NSS**,ESPECIALIDAD)

4.10 Relaciones N-arias

En este tipo de relaciones se *agrupan 3 o más entidades*, y para pasar al modelo de datos relacional cada entidad se convierte en una tabla. La relación se transforma en otra tabla, que va a contener los atributos propios de ella más las claves de todas las entidades. La clave de la tabla resultante será la concatenación de las claves de las entidades. Hay que tener en cuenta:

- *Si la relación es N:N:N*, la **clave de la tabla resultante es la unión de las claves de las entidades que relaciona**. Esa tabla incluirá los atributos de la relación si los hubiera.

  Por ejemplo: Para la relación ternaria entre Profesor, Curso y Materia, en la que un profesor imparte en varios cursos varias asignaturas, y además las asignaturas son impartidas por varios profesores en varios cursos el paso al esquema relacional quedaría:



PROFESOR (**CODIGO**, NOMBRE, ESPECIALIDAD, TELEFONO)

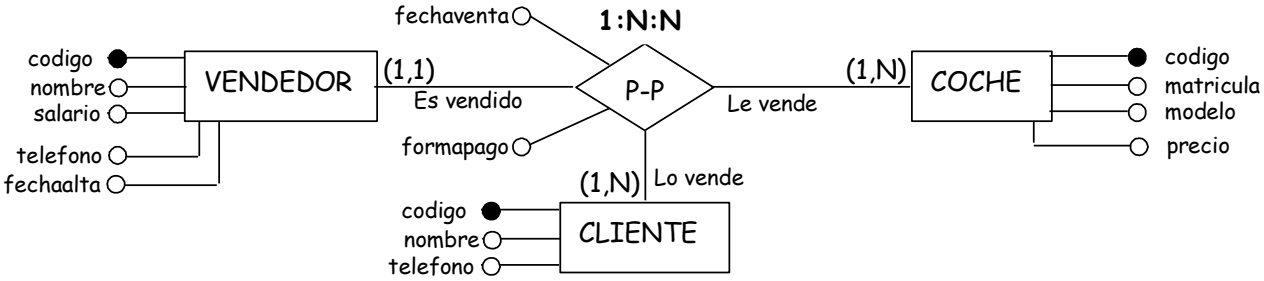
CURSO (**CODIGO**, DESCRIPCION, NIVEL)

MATERIA (**CODIGO**, NOMBRE, HORASSEMANA)

IMPARTE (**COD_CURSO**, **COD_PROFESOR**, **COD_MATERIA**)

- ***Si la relación es 1:N:N*****,** la clave de la entidad que participa con **cardinalidad máxima 1 no pasa a formar parte de la clave de la tabla resultante,** pero forma parte de la relación como un atributo más.

  Por ejemplo: Supongamos el caso de un concesionario de coches en el que un vendedor vende muchos coches a muchos clientes, y los coches son vendidos por un solo empleado. En la venta hay que tener en cuenta la forma de pago y la fecha de venta.



VENDEDOR (**CODIGO**, NOMBRE, SALARIO, TELEFONO, FECHAALTA)

CLIENTE (**CODIGO,** NOMBRE, TELEFONO)

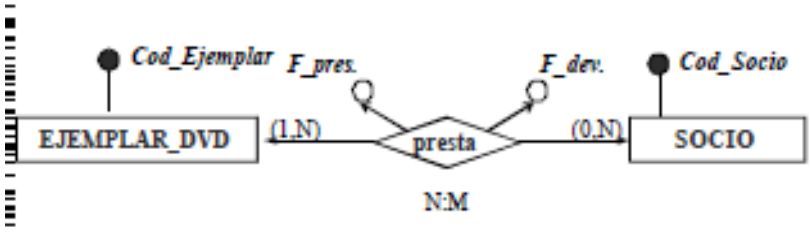
COCHE (**CODIGO**, MATRICULA, MODELO, PRECIO)

VENTA (**COD_CLIENTE, COD_COCHE**, ***COD_VENDEDOR***, FECHAVENTA, FORMAPAGO)

4.11 Transformación de la dimensión temporal

En el caso de **atributos con una dimensión temporal** (generalmente atributos que denotan fechas, horas, intervalos de tiempo), tanto si son multivalorados como univaluados, es necesario estudiar la semántica del universo de discurso con el fin de determinar cuáles van a ser los atributos que formen la clave primaria de la relacion a la que da lugar la interrelacion.

- Ejemplo1: **Base de datos histórica: consideramos el pasado**



Podemos comprobar que la clave primaria de la relación obtenido de la interrelacción "un socio toma prestado un libro de biblioteca durante un período determinado de tiempo " no es sólo la concatenación del Cod_Ejemplar y de Cod_Socio, sino se supone que un socio puede alquilar un ejemplar de DVD en diferentes períodos de tiempo (por tanto, F_pres y F_dev son atributos multivalorados), es necesario, a fin de formar la clave primaria, añadir a los códigos de SOCIO y de EJEMPLAR_DVD el atributo F_pres.

SOCIO(Cod_Socio,.....)

EJEMPLAR_DVD(Cod_Ejmeplar.....)

PRESTA(Cod_Socio, Cod_Ejemplar, F_pres, F_dev)

-Para el caso en que **la fecha del préstamo se registre con dos valores: fecha y hora**, y dado que en un instante de tiempo, un socio solo se puede llevar prestado un ejemplar de DVD, Socio no formaría parte de la clave. Entonces la tranformación quedaría:

SOCIO(Cod_Socio,.....)
EJEMPLAR_DVD(Cod_Ejmeplar.....)
PRESTA(**Cod_Ejemplar, F_pres**, Cod_Socio, F_dev)

- Ejemplo2: **Base de datos actual: no consideramos el pasado**:

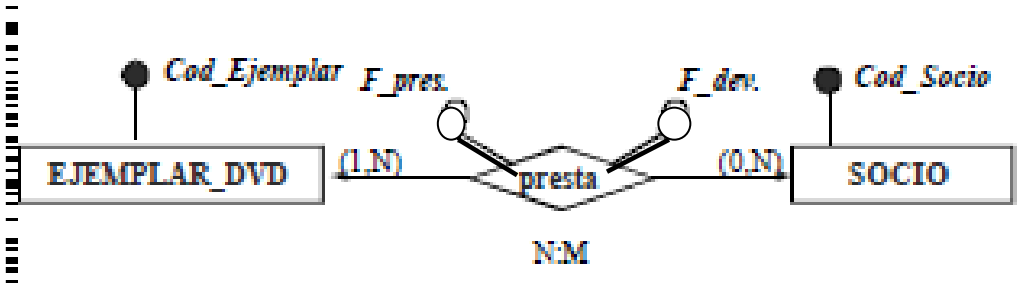


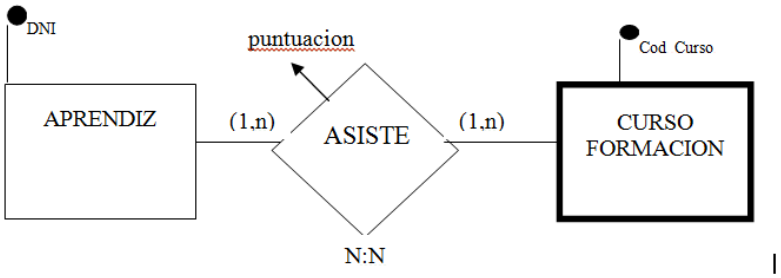
Fig.21. Equivalencia de entidades N:N y atributos monovaluados en la interrelacion

Si ahora consideramos la siguiente restricción, que **solo se almacena para cada socio, la fecha del último préstamo de ejemplar de DVD**, entonces la transformación quedaría:

SOCIO(Cod_Socio,.....)
EJEMPLAR_DVD(Cod_Ejmeplar.....)
PRESTA(**Cod_Ejemplar, Cod_Socio,** F_pres, F_dev)

Ejemplo 3: En el caso de que la interrelacion contenga un **atributo multivalorado que no denote dimensión temporal**, la **clave primaria de la relación deberá incluir también este atributo**.

En el ejemplo siguiente, la interrelación ASISTE entre APRENDIZ y CURSOFORMACION contiene un atributo multivalorado Puntuacion que refleja las distintas calificaciones obtenidas por un aprendiz en cada uno de los cursos a los que ha asistido. La transformación al modelo relacional de esta interrelación da lugar a una relación ASISTE cuya clave primaria contiene la concatenación de DNI, Cod_Curso y puntuacion; sin embargo, puesto que un aprendiz podría obtener iguales calificaciones en un determinado curso, sería necesario añadir un atributo que las distinga, por ejemplo, Fecha_evaluación y que también formaría parte de la clave primaria.



APRENDIZ(DNI,...)
CURSOFORMACION(Cod_Curso,.....)
ASISTE(DNI, Cod_Curso, Fecha_Evaluacion, Puntuacion)

4.12 Pérdida de semántica en la transformación al modelo relacional

Hay algunas restricciones que deben ser controladas con mecanismos externos al modelo relacional, ya que muchos SGBD no implementan el modelo relacional completo. Algunas de las restricciones que deben contemplarse en la transformación al modelo relacional mediante checks, aserciones o disparadores son:

- **Cardinalidades mínimas de 1** en relaciones N:N y 1:N
- **Cardinalidades máximas conocidas** en relaciones N:N, 1:N y relaciones ternarias
- **Exclusividad** en las generalizaciones
- **Inserciones y borrados en las generalizaciones**
- **Restricciones** que no figuran en el enunciado original pero que se consideran convenientes (por ejemplo: restricciones que implican comparación de fechas)

Restricciones de atributos derivados

- Los atributos derivados son aquellos cuyo valor se calcula a partir de otros atributos de entidades o relaciones o del recuento de sus ocurrencias. **Tienen representación gráfica en el modelo Entidad-Relación y pueden incluirse o no en el modelo relacional en función del uso que van a tener** o por cuestiones de eficiencia; en el caso de incluirlos, se tratan como atributos normales dando lugar a un atributo de la relación a la que pertenecen.
- En el momento de la creación de la base de datos, **debe crearse la restricción de integridad que garantice que siempre está actualizado el valor de este atributo**. Esto, cuando el cálculo está basado en información de otras relaciones supone la creación de una rutina que actualice el valor del atributo cada vez que se inserta, modifica o elimina valores en las tablas que dan lugar al cálculo. Este tipo de rutinas se conocen con el nombre de disparadores o triggers.

Restricciones en la transformación de jerarquías

- Las restricciones aplicadas cuando transformamos jerarquías totales y/o exclusivas, no se pueden plasmar en el modelo relacional. En la siguiente tabla vemos qué deberíamos hacer en función de la solución adoptada en la transformación.

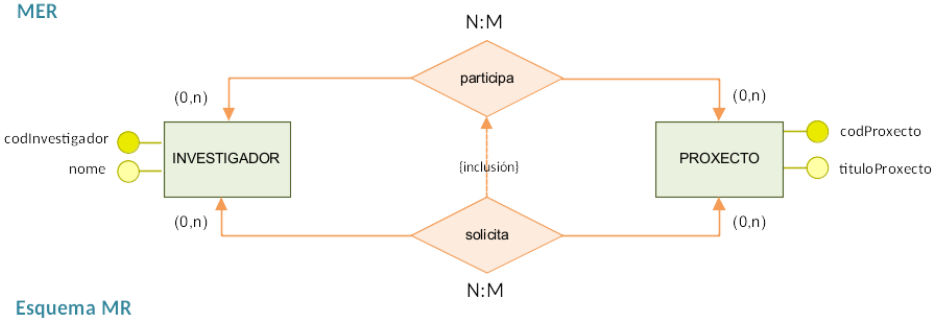
Solución de transformación	Restricción de jerarquía total	Restricción de jerarquía exclusiva
Integrar todas las entidades en una única, eliminando los subtipos, un atributo discriminante para saber a qué subtipo pertenece cada tupla y los atributos propios de los subtipos.	El atributo discriminante no puede tomar el valor nulo.	Solo los atributos de un subtipo pueden tener valores distintos de nulo para cada tupla.
Se crea una relación por cada entidad participante en la jerarquía.	No existe ninguna tupla en la tabla supertipo que no esté relacionada con una tupla de una tabla subtipo.	No permitir insertar nuevas tuplas en la tabla supertipo, sino que se insertan automáticamente cada vez que se inserta una tupla en una tabla subtipo. De esta manera, cando se intenta insertar una tupla en una tabla subtipo con la misma clave primaria de otra que ya existe en otra tabla subtipo, provocará un error de clave duplicada.
Crear una relación para cada subtipo.	No es necesario.	No permitir insertar una tupla en una tabla subtipo con una clave primaria que ya existe en alguna de las otras tablas subtipo.

- Este tipo de restricciones se resuelven en la creación da base de datos utilizando aserciones que son procedimientos que permiten validar cualquier atributo de una o varias.

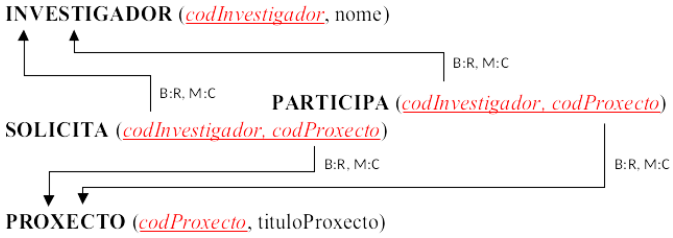
Restricciones asociadas a la Inclusividad y exclusividad

- Las restricciones de exclusividad, inclusividad, exclusión e inclusión de los diagramas Entidad-Relación no pueden ser plasmadas en el grafo relacional, pero deben ser documentadas con el fin de que en el momento de crear la base de datos en el SXBD se realicen las comprobaciones necesarias creando módulos programados en la base de datos para tal fin.
- Por ejemplo, la restricción de inclusión del siguiente diagrama Entidad-Relación representa que para que un investigador pueda solicitar un proyecto, tuvo que participar con anterioridad en otro, que no tiene por qué ser el que solicitó. Habrá que comprobar, antes de insertar una fila en la tabla SOLICITA, que exista alguna fila en la tabla PARTICIPA para ese investigador; en caso contrario no se permitirá la inserción.

MER



Esquema MR



5. Representación de esquemas de bases de datos

La manera formal es:

EMPLEADO (**NSS**, NOMBREEMPLEADO, APELIDOSEMPLEADO, SEXO, SALARIO, CALLE, NUMERO, CP, POBLACION, FECHANACIMIENTO, NUMDEPARTAMENTO, NSSSUPERVISOR)

DEPARTAMENTO (**NUM DEPARTAMENTO**, NOMBREDEPARTAMENTO, NSSEMPLEADO, FECHADIRECCION)

PROYECTO (**NUMEROPROYECTO**, NOMBREPROYECTO, LUGAR, NUMDEPARTAMENTO)

FAMILIAR (**NSSFAMILIAR**, NSSEMPLEADO, FECHANACIMIENTO, NOMBREFAMILIAR, APELLIDOSFAMILIAR, PARENTESCO, SEXO)

EMPLEADOPROYECTO (**NSSEMPLEADO**, **NUMEROPROYECTO**, NUMHORAS)

DEPARTAMENTO_UBICA (**NUMDEPARTAMENTO**, **LUGAR**)

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

Podríamos completar este esquema añadiendo los siguientes símbolos:

Símbolo	Ejemplo	Significado
Subrayado	Codigo	Clave principal
Subrayado doble	NSSEmpleado *NumDepartamento*	Clave alternativa Clave foránea
*	Nombre *	No admite valores nulos (restricción **NOT NULL**)

5.1 Grafo Relacional

Es un esquema relacional en el que hay líneas que enlazan las claves principales con las claves secundarias para representar mejor las relaciones