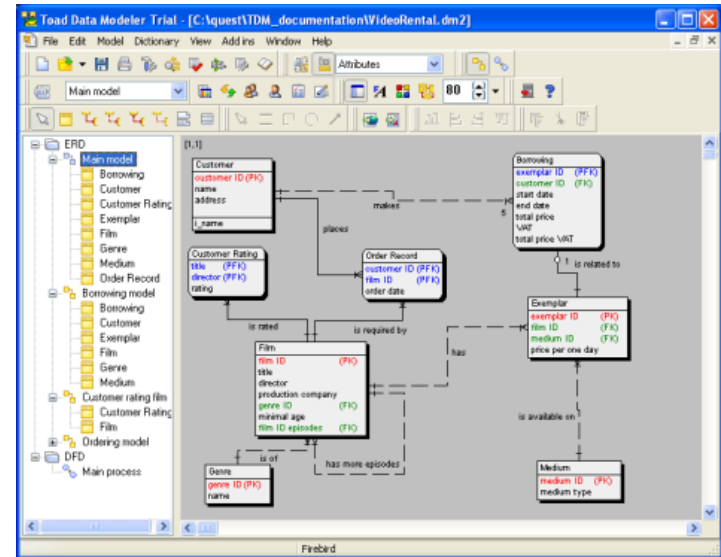


- ALUMNOS (DNIA, Teléfono, Email, Nombre, Apellidos, Dirección)
- MATRICULAN (DNIA, Código\_Curso, Fecha)
- CURSOS (Código\_Curso, Nombre, Descripción)
- TIENEN (Código\_Curso, Código\_Asignatura)
- ASIGNATURAS (Código\_Asignatura, Nombre, Dirección, DNIP)
- PROFESORES (DNIP, Nombre, Apellidos, Teléfono, Email, Dirección, Titulación)
- REALIZAN (DNIA, Código\_Examen, Fecha\_realizacion, Asistencia, Nota)
- EXAMENES (Código\_Examen, Fecha, Código\_Asignatura, DNIP)
- PREGUNTAS (CódigoP, Enunciado, Tipo, Respuesta, Código\_Examen)

## Tema 3



## Modelo Relacional



# El Modelo Relacional

- 1.- El Modelo Relacional (MR)
  - 1.1.- Las Relaciones en el MR
  - 1.2.- Conceptos necesarios para pasar del Modelo Conceptual (E/R) al Lógico (MR)
- 2.- Transformación de un diagrama E/R al MR
- 3.- Normalización
- 4.- MySQL Workbench

# El Modelo Relacional



## MODELO RELACIONAL DE DATOS

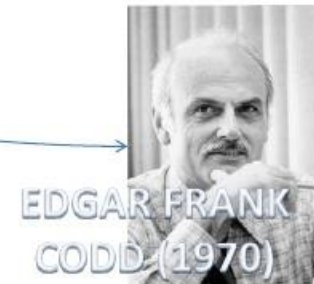
ALUMNO

Nro. Matricula	Apellidos	Nombres	Dirección	Teléfono
001-235	Caldas Flores	José Martín	Iglesia Luren	806666
002-124	Alvarado Pella	Juan Carlos	Junco al Mar	555555
003-417	Mora Maldonado	José	Cerca el cobre	777777

↓

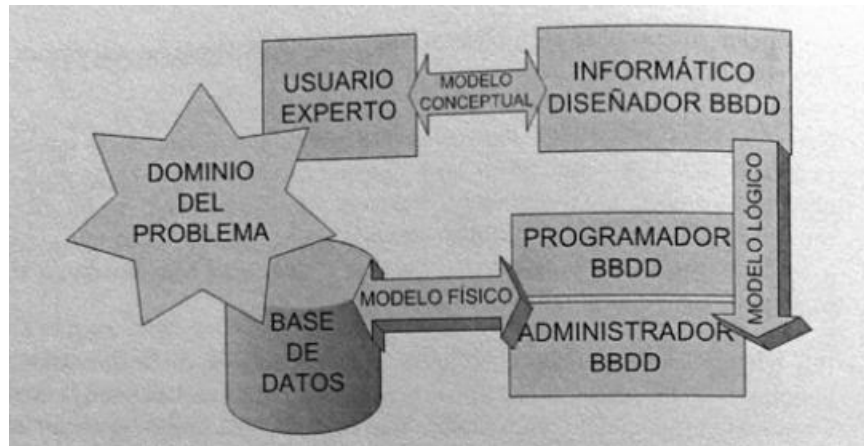
ATRIBUTOS

TUPLA

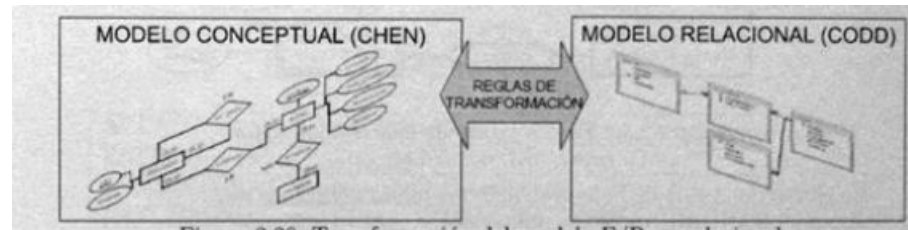


# Del Modelo E/R al Modelo Relacional

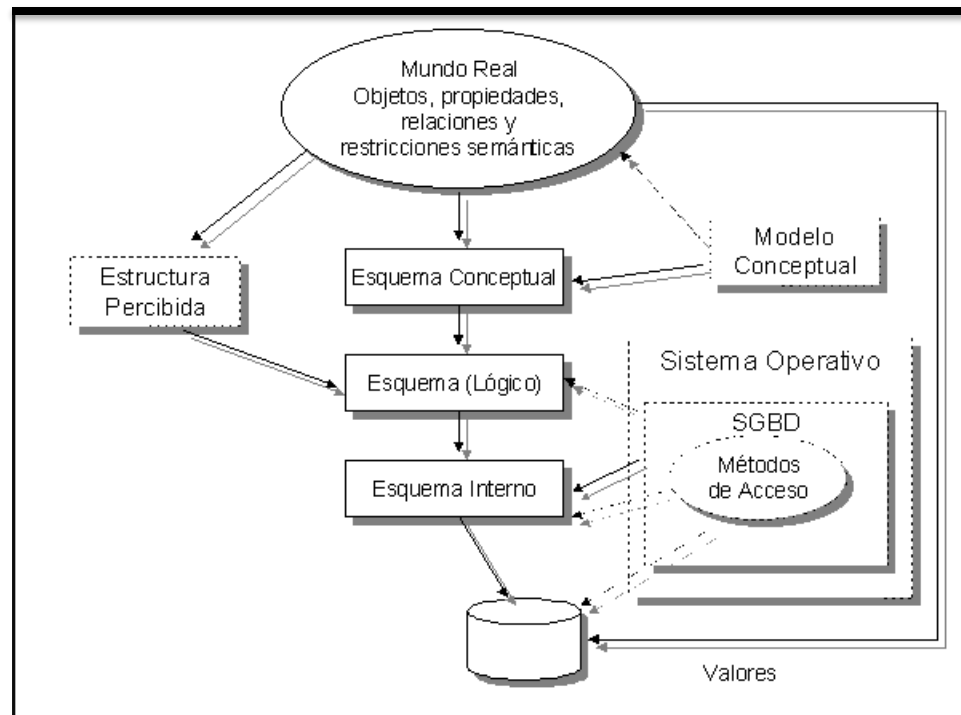
Fases



Transformación



# Fases



# FASES

- La fase de **diseño lógico** es aquella en la que se transforma el esquema conceptual (reflejado en un diagrama Entidad/Relación) en el modelo de datos **específico del SGBD seleccionado** (en nuestro caso, el **Modelo Relacional**).
  - Este paso se realiza en dos fases:
    - Transformación del MER a un modelo independiente del SGBD concreto.
    - Elaboración de las sentencias del lenguaje de definición de datos (DDL) del SGBD, que se completarán con las decisiones tomadas en el diseño físico posterior.

# FASES

- Una vez obtenido el esquema relacional independiente del SGBD, conviene realizar un refinamiento del mismo, centrándose en los siguientes aspectos:
  - ✓ Establecer las claves primarias.
  - ✓ Añadir las restricciones de integridad.
  - ✓ Establecer la posibilidad de valores nulos.
  - ✓ Normalizar el esquema obtenido.
  - ✓ Definir las reglas de inserción, actualización y borrado.
  - ✓ Revisar y validar el esquema con los usuarios.
  - ✓ Documentar la solución final.

# Del Modelo E/R al Modelo Relacional

El **Modelo E/R** Formulado por P.P. Chen en 1976

- ✓ Es un Modelo de datos que representa un esquema de base de datos mediante entidades y asociaciones.
- ✓ Describe una base de datos de una forma sencilla y global.
- ✓ Se realiza a partir de los requisitos de datos que debe cumplir una base de datos.

El **Modelo Relacional** Formulado por Edgar F. Codd En 1970 propone un nuevo modelo de datos basado en la teoría de conjuntos y en el concepto matemático de relación.

- ✓ La estructura lógica principal son tablas o relaciones
- ✓ Cada relación tiene un número fijo de columnas o atributos (esquema o intensión) y un número variable de filas o tuplas (extensión)
- ✓ Una BD relacional está compuesta por varias tablas o relaciones.

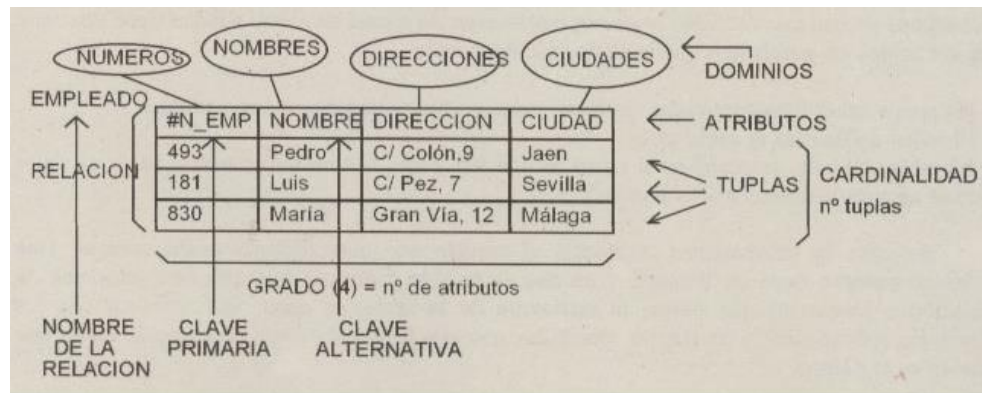


# 1.- El modelo Relacional

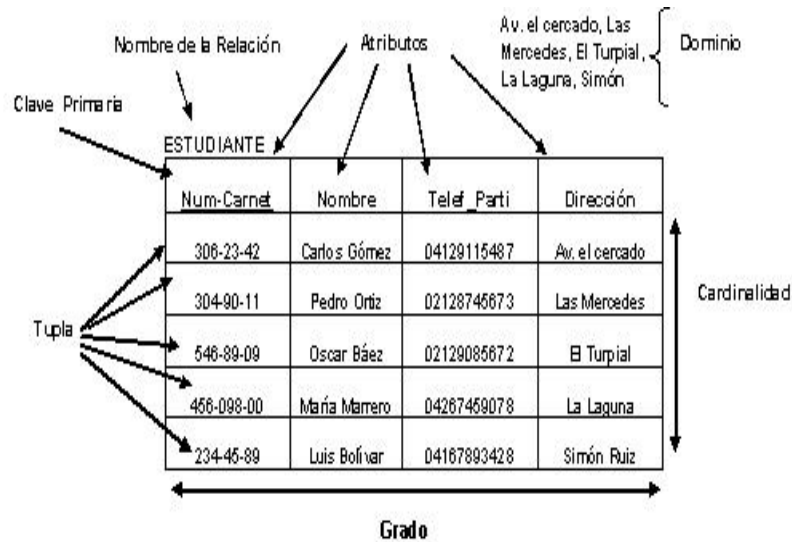
- La BD es como un conjunto de Relaciones que se pueden operar mediante el algebra relacional.
- El modelo Relacional es independiente de la forma en que se almacenan los datos y de la forma de representarlos, de forma que la BD se puede implementar en cualquier SGBD y los datos se pueden gestionar utilizando cualquier aplicación gráfica.

# 1.1.- Las Relaciones en el modelo Relacional

- Una Relación es un conjunto de atributos, cada uno perteneciente a un dominio y con un Nombre que identifica la relación.
- El conjunto de tuplas (filas) representa el **cuerpo de la relación** y el conjunto de atributos y el nombre representa el **Esquema**.
- El **Grado** de la tabla es el número de campos que posee.
- **Cardinalidad** es el número de tuplas concretas que almacena.
- **Valor**. Viene representado por la intersección entre una fila y una columna.
- **Valor Null**. Representa la ausencia de información.



# Conceptos



## Atributo

Es la propiedad que describe a cada entidad

## Campo

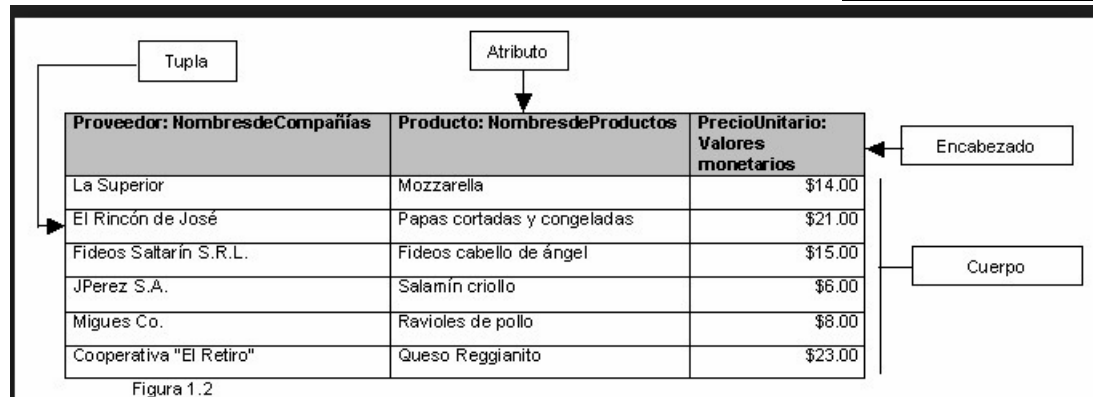
Mínima unidad de almacenamiento de información.

## Registro

Conjunto de Campos

## Dominio

Describe un conjunto de posibles valores de un atributo



# Conceptos

En el modelo relacional es frecuente llamar *tabla* a una relación, aunque para que una tabla sea considerada como una relación tiene que cumplir con algunas restricciones:

- Cada tabla debe tener su nombre único.
- No puede haber dos filas iguales. No se permiten los duplicados.
- Todos los datos en una columna deben ser del mismo tipo.

Diagram illustrating the structure of a table in a relational model. A box labeled "Columna" points to the header row of a table. A box labeled "Nombre de la tabla: Trabajo" is positioned above the table. A box labeled "Fila" points to the third row of the table.

Código	Nombre	Posición	Salario
1	Edgardo Trujillo	Gerente	19000
2	Lidimarie Fonsi	Empleada	12000
3	Jean Piaget	Empleado	13500
4	Jerome Bruner	Empleado	14000

## 1.2.- Conceptos necesarios para pasar del Modelo Conceptual (E/R) al Lógico ( MR)

- **Atributo:** Características de una entidad.
- **Dominio:** Conjunto de valores permitidos para un atributo. Es el conjunto finito de valores homogéneos (todos del mismo tipo) y atómicos (son indivisibles), que puede tomar cada atributo. Todos los dominios tienen un nombre y un tipo de datos asociado.

Existen dos tipos de dominios:

- **Dominios generales.** Son aquellos cuyos valores están comprendidos entre un máximo y un mínimo. Por ejemplo, el Código postal, que está formado por todos los números enteros positivos de 5 cifras.
- **Dominios restringidos.** Son los que pertenecen a un conjunto de valores específico. Por ejemplo, Sexo, que puede tomar los valores H o M.

## 1.2.- Conceptos necesarios para pasar del Modelo Conceptual (E/R) al Lógico ( MR)

- N° de empleado (NUMEMP), Apellido (APELLIDO), N° de departamento (NUMDEP), Salario (SALARIO).
- La **clave candidata** es el N° de empleado. El apellido se puede repetir, así que no se le considera candidata. Como solo hay una se escoge primaria el N° de empleado.
- El atributo N° de departamento es **clave ajena**; relaciona las tablas TEMPLE v TDEPART.

TDEPART	CLAVE PRIMARIA		
	NUMDEPT	NOMDEPT	PRESUPUESTO
	D1	Marketing	1.000
	D2	Desarrollo	1.200
	D3	Investigación	5.000

TEMPLE	CLAVE PRIMARIA		CLAVE AJENA	
	NUMEMP	APELLIDO	NUMDEP	SALARIO
	E1	López	D1	1.500
	E2	Fernández	D2	1.600
	E3	Martínez	D3	1.800
	E4	Sánchez	D2	2.000

## 1.2.- Conceptos necesarios para pasar del Modelo Conceptual (E/R) al Lógico ( MR)

- **Esquema de la base de datos**
- Una base de datos relacional es un conjunto de relaciones normalizadas. Para representar un esquema de una base de datos se debe dar el nombre de sus relaciones, los atributos de estas, los dominios sobre los que se definen estos atributos, las claves primarias y las claves ajenas.
- En el esquema los nombres de las relaciones aparecen seguidos de los nombres de los atributos encerrados entre paréntesis. Las claves primarias son los atributos subrayados, y las claves ajenas se representan mediante diagramas referenciales.

TDEPART  
TEMPLE  
TEMPLE  $\xrightarrow{\text{NUMDEP}}$  TEDEPART:

(NUMDEPT, NOMDEPT, PRESUPUESTO)  
(NUMEMP, APELLIDO, NUMDEP, SALARIO)  
Departamento al que pertenece el empleado

## 1.2.- Conceptos necesarios para pasar del Modelo Conceptual (E/R) al Lógico ( MR)

- Al realizar un Modelo de datos de un Sistema, se han de reflejar determinadas condiciones o **políticas de negocio** que se deben cumplir. Hay dos grupos:
  - **Restricciones Inherentes:** Se derivan de la propia definición del modelo, no dependen del diseñador.
  - **Restricciones de Usuario o Semánticas:** Son impuestas por el diseñador en función de los requisitos del sistema a modelar.



# 1.2.- Conceptos necesarios (1)

## Restricciones Inherentes

- ✓ **No** pueden haber **dos tuplas iguales**. No pueden haber dos ocurrencias o filas de una relación con el mismo valor en todos los campos.
- ✓ El **orden** de las **tuplas** **no** es **significativo**.
- ✓ El orden de los **atributos** (columnas) no es significativo.
- ✓ Cada **atributo** solo puede tomar un **único valor** del dominio. No se admiten los grupos repetitivos, es decir, dos valores para un mismo campo.
- ✓ Se debe cumplir la regla de la **integridad de la entidad**: "Ningún atributo que forme parte de la clave primaria puede tomar un valor desconocido o nulo".

## 1.2.- Conceptos necesarios (2)

- ✓ **Clave:** Conjunto de atributos que identifican de forma única una ocurrencia de entidad, pueden ser simples o compuestas. Tipos:
  - ✓ **Clave Candidata:** Conjunto mínimo de atributos que identifican unívoca y mínimamente cada tupla en una relación. Pueden haber varias.
  - ✓ **Clave Primaria:** Es la clave Candidata elegida.
  - ✓ **Clave Alternativa:** Claves candidatas que no han sido elegidas como primaria.
  - ✓ **Clave Foránea o Ajena:** Conjunto de atributos de una relación cuyos valores han de coincidir con los valores de la clave primaria de otra relación (ambos deben estar definidos sobre los mismos dominios).

# Restricciones Semánticas o de Usuario

➤ **Las Restricciones de Usuario o de semántica** son las Condiciones que deben cumplir los datos:

- ✓ Restricciones **de clave** (identifican de forma única una entidad)
- ✓ Restricciones **de valor único** (UNIQUE)
- ✓ Restricciones de **integridad referencial(\*Nota1)**. Se da cuando una tabla tiene una referencia a un valor de otra tabla (la PK de ésta). En este caso la restricción exige que exista el valor referenciado en la otra tabla.
- ✓ Restricciones **de dominio**. Exige que el valor que puede tomar un campo esté dentro del dominio definido.

# Restricciones Semánticas o de Usuario

- ✓ Restricciones **de verificación** (CHECK). Permite comprobar si el valor de un atributo es válido conforme a una expresión. Cada vez que se realice una inserción o una actualización de datos se comprueba si los valores cumplen la condición. Rechaza la operación si no se cumple.
- ✓ Restricciones de **valor nulo**. Si un atributo se admite como NULL es que es opcional.
- ✓ Disparadores o **triggers**. Son procedimientos que se ejecutan para hacer una tarea concreta en el momento de insertar, borrar o modificar información en una tabla.

# Restricciones Semánticas o de Usuario

Restricciones genéricas adicionales o **aserciones** (ASSERT). Permite validar cualquiera de los atributos de una o varias tablas, en este caso en lugar de afectar a una relación como CHECK, puede afectar a dos o mas relaciones. La condición se establece sobre elementos de distintas relaciones. Pueden implicar a subconsultas en la condición. La definición de una aserción debe tener un nombre. Tiene vida por si misma.

# Integridad Referencial

- Integridad referencial o restricción de clave ajena (**FOREIGN KEY**). Se utiliza para enlazar relaciones, mediante claves ajenas, de una base de datos. La integridad referencial indica que los valores de la clave ajena en la relación hijo se corresponden con los de la clave primaria en la relación padre.
- Además de definir las claves ajenas hay que tener en cuenta las operaciones de borrado y actualización que se realizan sobre las tuplas de la relación referenciada.
- Las posibilidades son las siguientes:

# Integridad Referencial

- **Borrado y/o modificación en cascada (CASCADE).** El borrado o modificación de una tupla en la relación padre (relación con la clave primaria) ocasiona un borrado o modificación de las tuplas relacionadas en la relación hija (relación que contiene la clave ajena). En el caso de empleados y departamentos, si se borra un departamento de la tabla TDEPART se borrarán los empleados que pertenecen a ese departamento. Igualmente ocurrirá si se modifica el NUMDEPT de la tabla TDEPART esa modificación se arrastra a los empleados que pertenezcan a ese departamento.
- **Borrado y/o modificación restringido (RESTRICT).** En este caso no es posible realizar el borrado o la modificación de las tuplas de la relación padre si existen tuplas relacionadas en la relación hija. Es decir, no podría borrar un departamento que tiene empleados.

# Integridad Referencial

- **Borrado y/o modificación con puesta a nulos (SET NULL).** Esta restricción permite poner la clave ajena en la tabla referenciada a NULL si se produce el borrado o modificación en la tabla primaria o padre. Así pues, si se borra un departamento, a los empleados de ese departamento se les asignara NULL en el atributo NUMDEPT.
- **Borrado y/o modificación con puesta a valor por defecto (SET DEFAULT).** En este caso, el valor que se pone en las claves ajenas de la tabla referenciada es un valor por defecto que se habrá especificado en la creación de la tabla



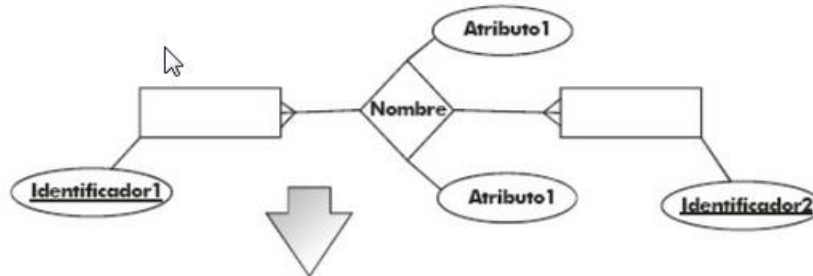
## 2.- Transformación de un diagrama E/R al Modelo Relacional

- El **grafo relacional** es la representación de un sistema mediante un conjunto de relaciones vinculadas entre sí por una o varias claves ajenas, a partir del cual podremos crear la B.D. en nuestro Sistema Informático con la ayuda del SGBD elegido.
- Lo que se pretende es pasar de describir conceptualmente el mundo mediante entidades y relaciones, a describirlo lógicamente mediante tablas.
- Las reglas básicas para transformar un esquema conceptual E-R a un esquema relacional son las siguientes:
  - Toda **entidad** se transforma en una tabla.
  - Todo **atributo** se transforma en columna dentro de una tabla.
  - El **identificador único** de la entidad se convierte en clave primaria.
  - Toda **relación N:M** se transforma en una tabla que tendrá como clave primaria la concatenación de los atributos clave de las entidades que asocia.

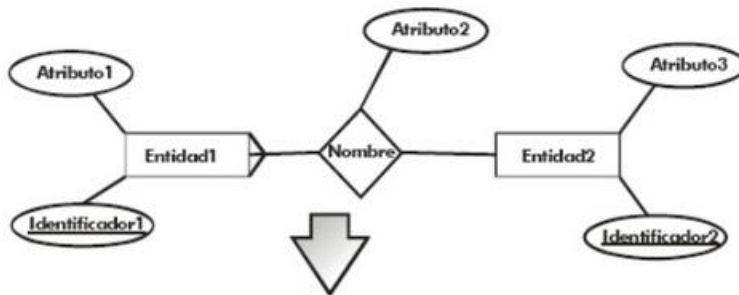
## 2.- Transformación de un diagrama E/R al Modelo Relacional

- **Claves Principales:** Los atributo/s que forman la clave principal se subrayan.
- **Claves Secundarias:** doble subrayado.
- **Atributos opcionales:** \*
- **Claves Ajenas:** trazo discontinuo
- **Opciones de Borrado(B/D) y Modificación(M/U):** se indican con las letras:
  - M:C /UC=> Modificación en cascada
  - M:N /UN=> Modificación con puesta a NULL
  - M:D /UD=> Modificación con valor por defecto
  - M:R /UR=> Modificación Restringida

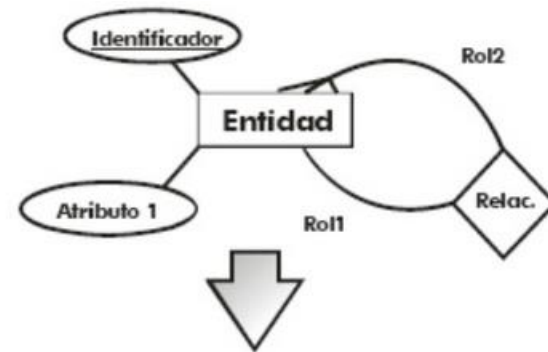
# Reglas de Transformación. Resumen



Nombre(Identificador1, Identificador2, Atributo1, Atributo2)



Entidad2(Identificador2, Atributo3)  
Entidad1(Identificador1, Atributo1, Identificador2, Atributo2)



Entidad(Identificador, Atributo1, Identificador Rol 1)

## 2.- Transformación del MER al Modelo relacional

- El paso del modelo conceptual, basado en el modelo Entidad/Relación, al modelo relacional se basa en una serie de reglas que, seguidas de forma precisa y sistemática, garantizan la finalización del proceso de forma completa y fiable **sin pérdida semántica**.
- Los pasos a seguir para la obtención del modelo lógico a partir del modelo conceptual son los siguientes:
  - Transformación de atributos.
  - Transformación de entidades.
  - Transformación de relaciones.
  - Transformación de jerarquías.
  - Normalización.

## 2.- Transformación de un diagrama E/R al Modelo Relacional

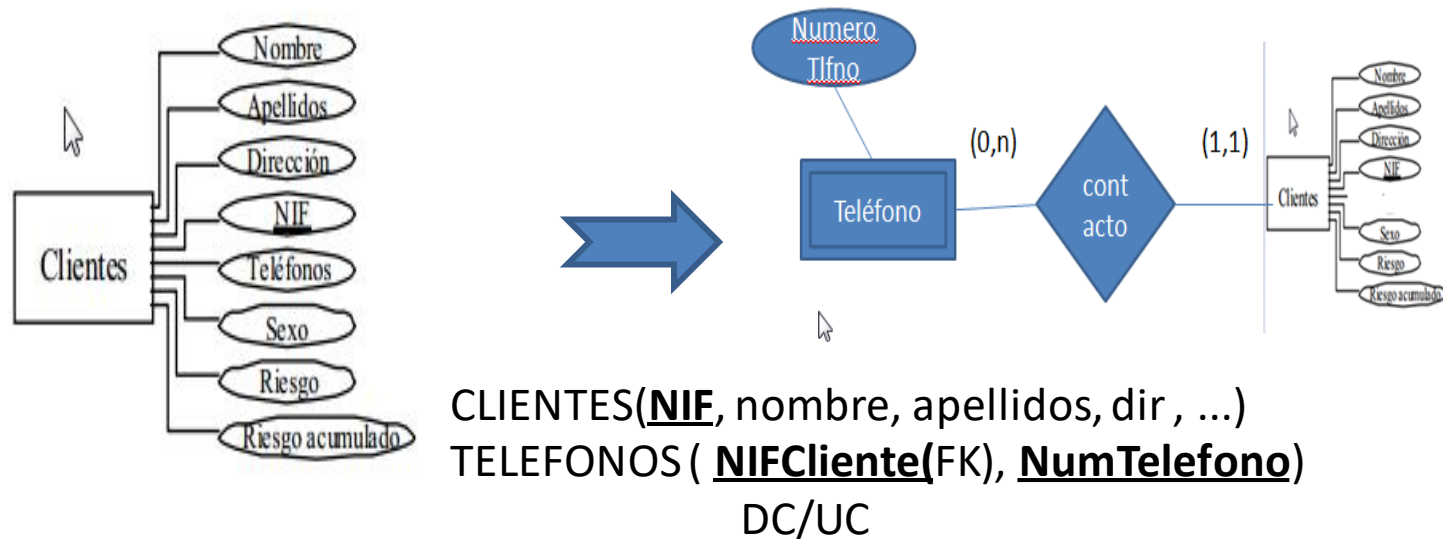
### ✓ Utilización de atributos atómicos.

El modelo relacional impone una condición sobre las características que debe poseer cada columna de una tabla relacional. Todas las columnas poseen datos simples, también llamados atómicos, de manera que cada uno de ellos representa una información unitaria y que no puede descomponerse en bloques de información más pequeños, (al menos desde el punto de vista del S.G.B.D.). Por ejemplo Teléfonos que es un atributo múltiple, pues consiste en un atributo Teléfono, duplicado un número indeterminado de veces, no tiene representación correspondiente en el modelo relacional, por lo que es necesario algún tipo de transformación que convierta atributos múltiples en atributos atómicos.

Un atributo compuesto es aquél que puede dividirse en bloques de información más pequeños, pero a diferencia de los múltiples, estos bloques pequeños no son homogéneos, sino diferentes entre sí. Por ejemplo el Domicilio de algunas entidades. Normalmente no es necesario distinguir entre calle, número, portal, planta, etc. Si en nuestro diseño deseamos distinguir entre todos esos componentes del Domicilio entonces la única solución es crear atributos atómicos para cada uno de ellos.

# Utilización de atributos atómicos

Esta transformación consiste en crear una entidad débil llamada Teléfonos, y relacionarla con Clientes a través de una relación débil. Esta nueva entidad poseerá un único atributo atómico Número de Teléfono. El atributo múltiple se convierte en una entidad, ahora ya con atributos atómicos, de manera que cada Cliente se relacionará con 0, 1 ó muchas instancias de esta nueva entidad.



## 2.- Transformación de un diagrama E/R al modelo Relacional

### ✓ Conversión de entidades y relaciones a tablas.

Una vez preparados los atributos de las entidades y relaciones, la conversión del diagrama E-R al modelo relacional pasa por dos etapas: una en la que se convierten las entidades, y otra en la que se convierten las relaciones. Pero, las tablas que se van obteniendo no adoptan su forma definitiva hasta que se ha acabado el proceso.

### ✓ Conversión de entidades a tablas.

Existe una correspondencia directa entre el concepto de entidad del diagrama E-R (una vez eliminados los atributos múltiples y los compuestos), y el concepto de tabla relacional.

Cientes

Nombre	Apellidos	Dirección	<u>NIE</u>	Sexo	Riesgo	Riesgo acumulado
--------	-----------	-----------	------------	------	--------	------------------

## 2.1- Transformación de Entidades, Atributos y Dominios

- En el modelo relacional, los atributos de las entidades y relaciones del MER se transforman en atributos del modelo relacional, teniendo en cuenta los casos:
  - ❑ Atributos **compuestos**: Se descomponen en atributos simples. Ejemplo: atributo Dirección.
  - ❑ Atributos **multivaluados**: Dan lugar a una nueva relación cuya clave primaria es la concatenación de la clave primaria de la entidad en la que se sitúa el atributo multivaluado más el nombre del atributo multivaluado. En ocasiones, si el atributo multivaluado no permite repeticiones, es suficiente éste como clave primaria.
  - ❑ Atributos **Obligatorios**: Restricción **NOT NULL**.



## 2.1- Transformación de Entidades, Atributos y Dominios

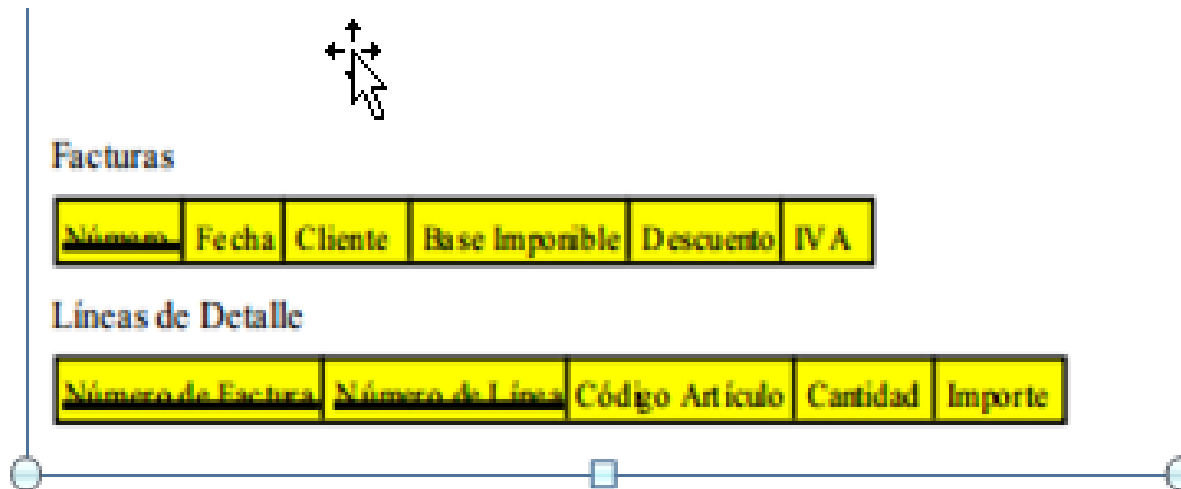
- ❑ **Atributos Opcionales:** Atributos que pueden tomar valor NULL
- ❑ **Atributos Identificador Principal:** Atributo/s que forman la clave primaria **PRIMARY KEY**.
- ❑ **Atributos Identificador Alternativo:** Atributo/s con la restricción **UNIQUE**.
- ❑ **Atributos Derivados:** Atributo/s que se obtienen como resultado de algún cálculo sobre otros atributos.
- ❑ **Dominios:** Se transforman en los dominios del modelo Relacional.

## 2.2- Transformación de Interrelaciones

- ✓ **Conversión de relaciones binarias a tablas.**
  - **Conversión de relaciones es-un y relaciones débiles.**
- La conversión de relaciones es-un y relaciones binarias débiles en general, no conlleva realmente la aplicación de ninguna regla, ya que la propia conversión de la entidad débil en tabla convierte automática e implícitamente la relación también.
- Supongamos por ejemplo, el diagrama que nos permite representar las facturas propias de cualquier negocio. Dado que el número de líneas de detalle de una factura es indeterminado, es necesario crear una relación débil que relacione cada factura con los detalles que en ella se facturan:

# Conversión de relaciones es-un y relaciones débiles

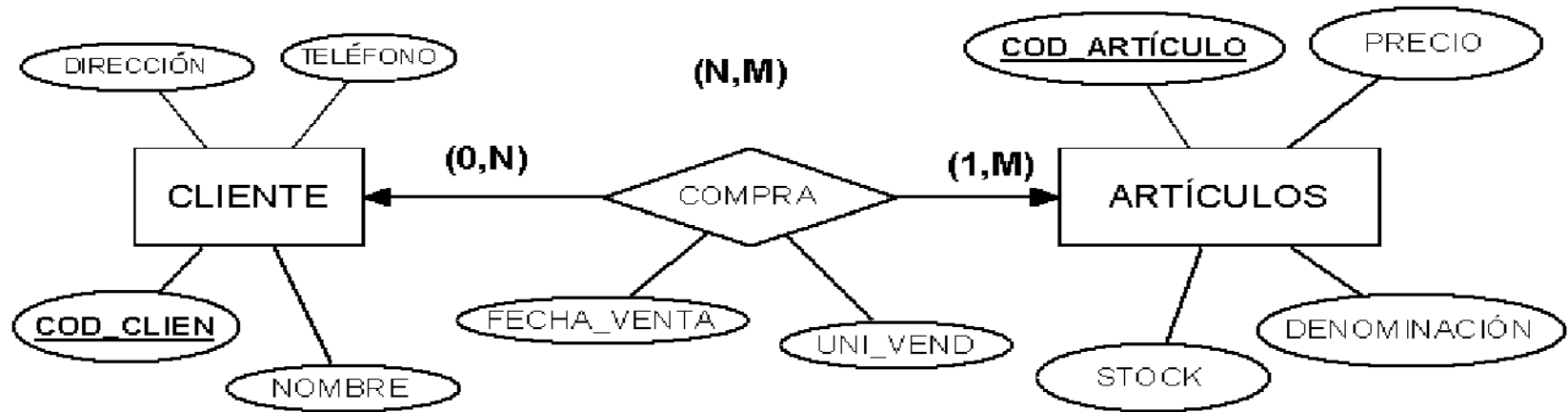
Al convertir las entidades Facturas y Líneas de Detalle en sus tablas correspondientes, se observa que la tabla de Líneas de Detalle hereda los atributos que forman la clave de Facturas. Se deberá tener la opción de borrado en cascada, es decir, no podrá existir ninguna ocurrencia de la entidad hija que no esté relacionada con una entidad padre.



## 2.2.1.- Transformación de Interrelaciones N:M

- Estas interrelaciones dan lugar a una relación cuya clave será la concatenación de los Identificadores Principales de ambas entidades. Por tanto, los atributos que forman la clave primaria de la relación serán claves ajenas respecto a las relaciones en las que estos atributos son claves primarias. Si la interrelación tiene atributos, éstos pasarán a la nueva tabla generada.
- En la práctica, es necesario especificar que ocurre en los casos de borrado y modificación de la clave primaria de referencia, indicando la acción correspondiente:
  - Restricción.
  - Asignación de valor nulo o valor por defecto.
  - Operación *en cascada*.
  - Ejecución de un procedimiento de usuario.

## 2.2.1.- Transformación de Interrelaciones N:M



- CLIENTE (COD\_CLIEN, NOMBRE, DIRECCION, TELEFONO)
- ARTICULOS (COD\_ARTICULO, PRECIO, STOCK, DENOMINACION)
- COMPRA (COD\_CLIEN (FK), COD\_ARTICULO(FK), UNI\_VEND, FECHA\_VENTA)

DC:UC

DC:UC

# Código de creación de estas tres tablas:

## **CREATE TABLE CLIENTE (**

```
COD_CLIEN NUMBER(6) NOT NULL PRIMARY KEY,  
NOMBRE VARCHAR2(15) NOT NULL ,  
DIRECCION VARCHAR2(15) NOT NULL ,  
TELEFONO NUMBER(10) NOT NULL );
```

## **CREATE TABLE ARTICULOS(**

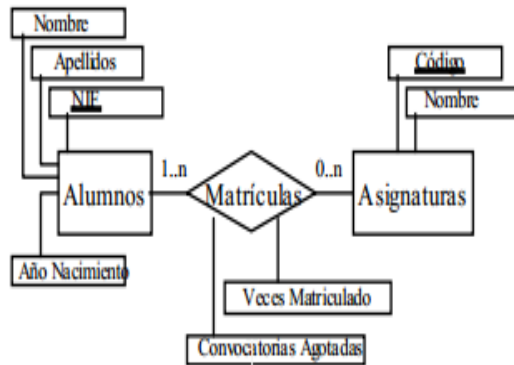
```
COD_ARTICULO NUMBER(6) NOT NULL PRIMARY KEY,  
PRECIO NUMBER (6,2) NOT NULL ,  
STOCK NUMBER (4) NOT NULL ,  
DENOMINACION VARCHAR2(15) NOT NULL );
```

## **CREATE TABLE COMPRA(**

```
COD_CLIEN NUMBER(6) NOT NULL,  
COD_ARTICULO NUMBER(6) NOT NULL,  
UNI_VEND NUMBER (4) NOT NULL ,  
FECHA_VENTA DATE NOT NULL ,  
CONSTRAINT PK_COMPRA PRIMARY KEY (COD_CLIEN,COD_ARTICULO),  
CONSTRAINT FK_CLIEN FOREIGN KEY (COD_CLIEN)  
REFERENCES CLIENTE (COD_CLIEN)  
ON DELETE CASCADE ON UPDATE CASCADE,  
CONSTRAINT FK_ARTIC FOREIGN KEY (COD_ARTICULO)  
REFERENCES ARTICULOS (COD_ARTICULO)  
ON DELETE CASCADE ON UPDATE CASCADE);
```

## 2.2.1.- Transformación de Interrelaciones N:M

Ejemplos:



Alumnos

Nombre	Apellidos	NIE	Año Nacimiento
--------	-----------	-----	----------------

Asignaturas

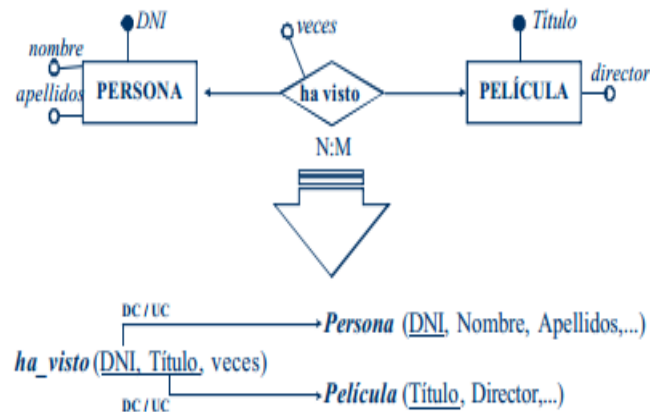
Código	Nombre
--------	--------

Matriculas

NIE del Alumno	Código de Asignatura	Veces Matriculado	Convocatorias Agotadas
----------------	----------------------	-------------------	------------------------

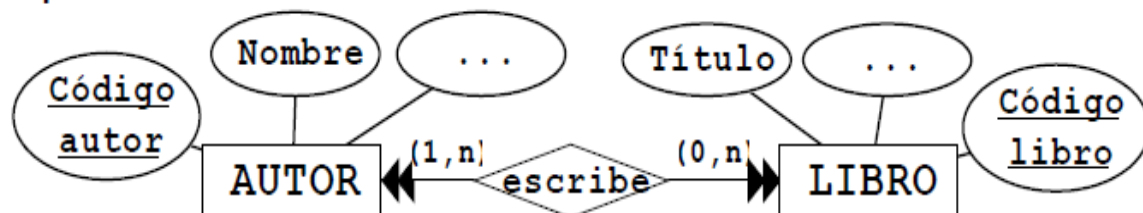
### Interrelaciones N:M

Ejemplo:



# Ejemplo transformación de una Relación N:M

- Ejemplo. Transformación de la relación `escribe`:



`AUTOR`(#Código autor, Nombre, Fecha de nacimiento, ...)

`LIBRO`(#Código libro, Título, Año publicación, ...)

`escribe`(#Cód autor, #Código libro)

## Claves externas:

`Cód autor` referencia a `AUTOR`.

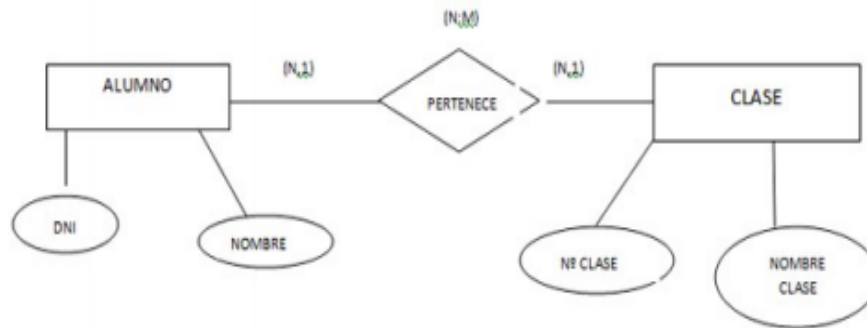
`Código libro` referencia a `LIBRO`.

## Restricciones:

Las claves externas `Cód autor` y `Código libro` no pueden ser nulas.



# Ejemplo transformación de una Relación N:M



Para este modelo de entidad-relación el paso a tablas quedaría de la siguiente forma:

Tabla alumno	DNI(clave primaria)	nombre
--------------	---------------------	--------

Tabla clase	Nº clase (clave primaria)	Nombre clase
-------------	---------------------------	--------------

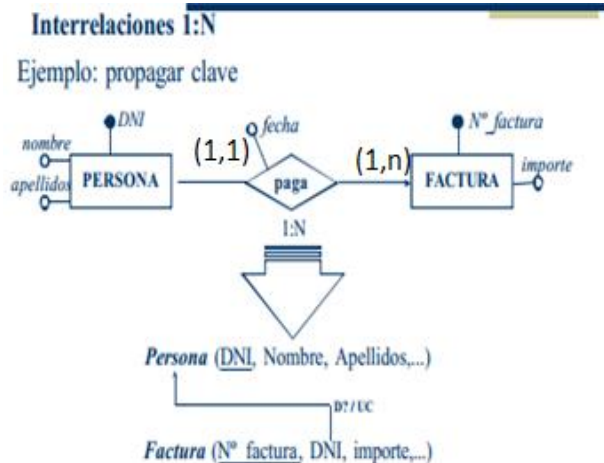
Tabla pertenece	DNI (clave foránea)	Nº clase (clave foránea)
-----------------	---------------------	--------------------------

Clave primaria

## 2.2.2.- Transformación de Interrelaciones 1:N

Cuando la relación que se desea convertir es del tipo 1:N hay dos posibles soluciones:

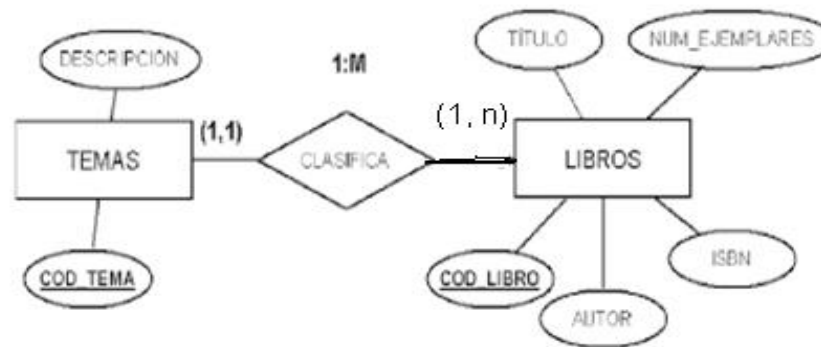
- 1/ **Propagar la clave** de la entidad que actúa con participación máxima **1** a la tabla correspondiente a la entidad que actúa con participación máxima **N**, así como los atributos propios de la relación. Esto se aplica cuando la cardinalidad es obligatoria, es decir, cuando tenemos cardinalidad (1,1) y (1,N)



Inconveniente: la propagación de la clave de una entidad a otra genera la aparición de una *clave externa en la entidad de destino*. Esto obliga a especificar el mecanismo de eliminación y modificación en función de la semántica del problema.

## 2.2.2.- Transformación de Interrelaciones 1:N

Clasificación de los libros en temas.



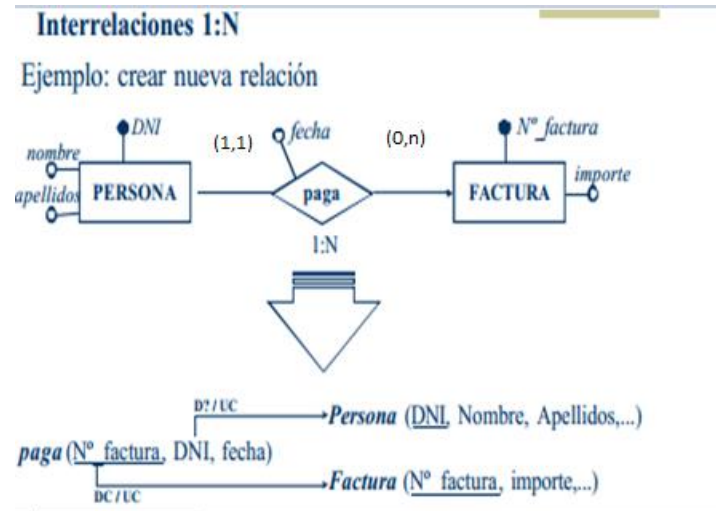
- TEMAS (COD\_TEMA, DESCRIPCION)
- LIBROS (COD\_LIBRO, AUTOR, ISBN, TITULO, NUM\_EJEMPLARES, COD\_TEMA(FK))

## 2.2.2.- Transformación de Interrelaciones 1:N

2/ Transformarla en una **nueva relación** que tendrá como atributos las claves de ambas entidades y los atributos propios de la nueva relación. Su **clave** será la clave de la entidad que interviene en la relación con **N** ocurrencias.

Es recomendable cuando:

- ✓ La relación tiene atributos propios.
- ✓ La cardinalidad es opcional, e.d., la cardinalidad mínima es 0 en, al menos, uno de los extremos.
- ✓ Cuando se prevé que en un futuro la relación se convertirá en N:M



## 2.2.2.- Transformación de Interrelaciones 1:N

- En un supermercado hay productos organizados en categorías (frutas, prod. Limpieza, carnes, pastelería, etc.), Cada producto pertenece a una sola categoría y pueden haber categorías que todavía no tengan productos, pero no pueden haber productos sin categoría

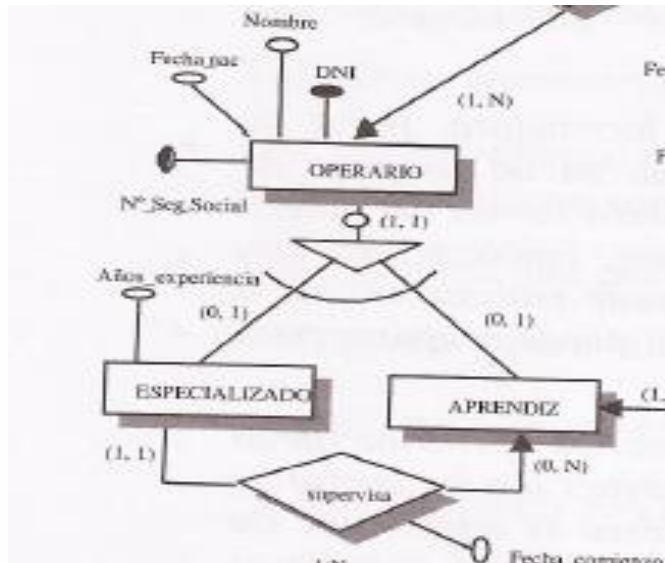


CATEGORIAS(CodCategoría, nombre, sección, estantería)

PRODUCTOS(CodProducto, nombre, precio)

ORGANIZA(CodProducto(FK), CodCategoría(FK))

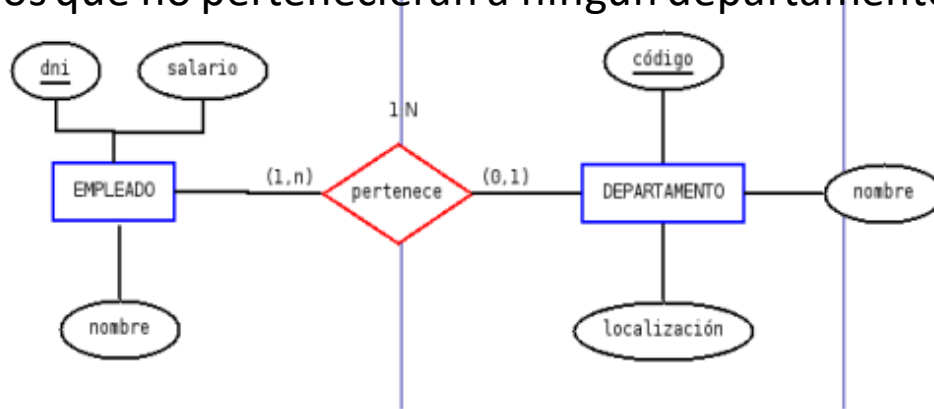
# Ejemplo Relación de 1:N



- En este caso, se crea una nueva relación para la interrelación supervisa, en la que la clave primaria constará de la clave primaria de la entidad del lado N.
- SUPERVISA(**DNIAprendiz(FK)**, DNIEspecializado(FK), FechaComienzo)

# Ejemplo Relación de 1:N

Un empleado pertenece a un único departamento, y un departamento tiene uno o más empleados. Imaginemos que pudiera darse el caso de que hubiera empleados que no pertenecieran a ningún departamento.



- EMPLEADO(dni, nombre, salario)
- DEPARTAMENTO(código, nombre, localización)
- PERTENECE(dni empleado(FK), código\_departamento(FK))

# Transformación de relaciones

- Hay que revisar *cuidadosamente las cardinalidades mínima y máxima* de cada una de las entidades que participan en la relación, tomando la decisión en función de las mismas.
- ● Si la cardinalidad mínima es 1 **no pueden admitirse valores** nulos en la clave externa (propagada), hay que definir explícitamente la restricción.



## 2.2.3.- Transformación de Interrelaciones 1:1

Existen dos soluciones:

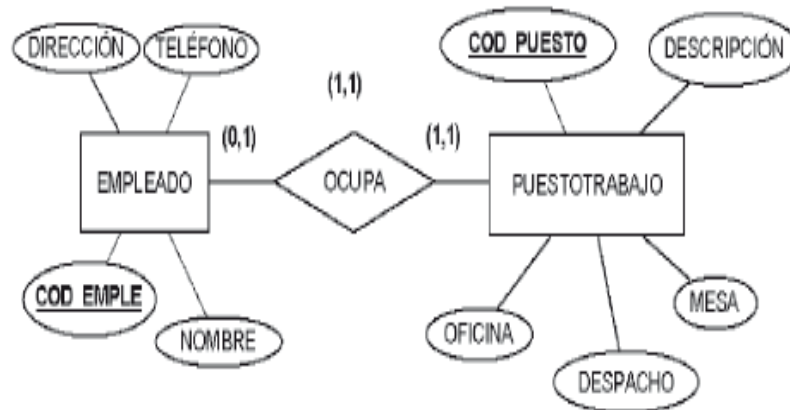
- **Transformar la relación en una tabla.** Si las entidades poseen cardinalidades (0,1), la relación se convierte en una tabla.
- **Propagar la clave.** Si una de las entidades posee cardinalidad (0,1) y la otra (1,1), conviene propagar la clave de la entidad con cardinalidad (1,1) a la tabla resultante de la entidad de cardinalidad (0,1). Si ambas entidades poseen cardinalidades (1,1), se puede propagar la clave de cualquiera de ellas. En este caso, también se añadirán los atributos de la relación a la entidad receptora si los hubiera.

# Conversión de relaciones uno a uno

- Opciones:
  - a) Transformación en una tabla.** Es recomendable cuando ambas entidades tienen cardinalidad mínima 0.
  - b) Propagación de la clave** (En este caso, el criterio *habitual es propagar en* aquel sentido que recoja una mayor cantidad de **semántica** posible, evitando los valores nulos) :
    - ✓ Si ambas entidades tienen cardinalidad (1,1), teniendo en cuenta los accesos mas frecuentes o prioritarios a los datos.
    - ✓ En las relaciones con cardinalidades (1,1) y (0,1), de la entidad con cardinalidad mínima 1 a la de cardinalidad mínima 0.  
Si la relación tiene atributos, se propagan con la clave.

## Conversión de relaciones uno a uno.

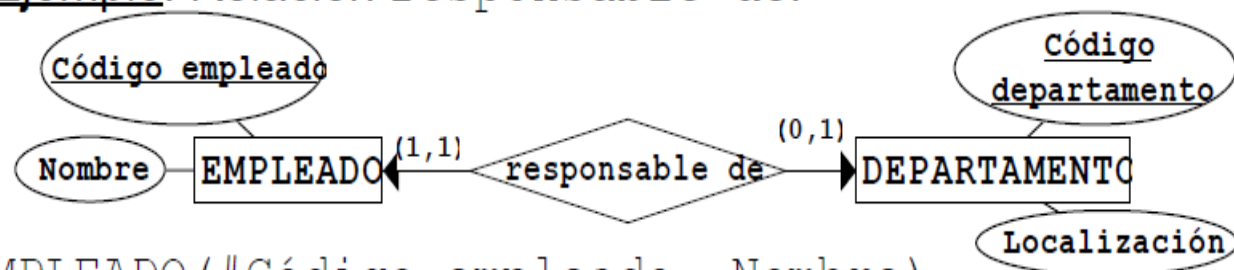
**EMPLEADO-OCUPA-PUESTOTRABAJO.** Un empleado ocupa un solo puesto de trabajo, y ese puesto de trabajo es ocupado por un solo empleado o por ninguno.



- En este caso, la clave se propaga desde la entidad PUESTOTRABAJO, con cardinalidad (1,1), a la entidad EMPLEADO, con cardinalidad (0,1).
- PUESTOTRABAJO (COD PUESTO, DESCRIPCION, OFICINA, DESPACHO, MESA)
- EMPLEADO(COD EMPL, NOMBRE, DIRECCION, TELEFONO, COD\_PUESTO(FK))

# Ejemplo de Relación 1:1

- Ejemplo. Relación responsable de:



EMPLEADO(#Código empleado, Nombre)

DEPARTAMENTO(#Código departamento, Nombre,  
Código empleado)

Clave externa:

Código empleado referencia a EMPLEADO.

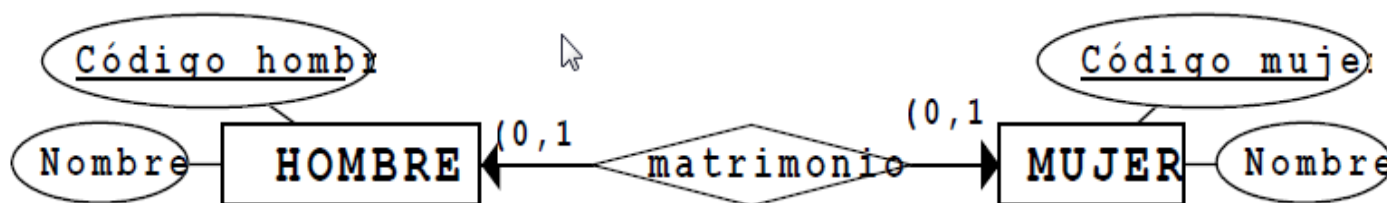
Restricciones:

Clave externa no nula.

# Ejemplo de Relación 1:1

## Relaciones de grado 1:1

- Ejemplo. Relación `matrimonio`:



`MUJER(#Código mujer, Nombre)`

`HOMBRE(#Código hombre, Nombre)`

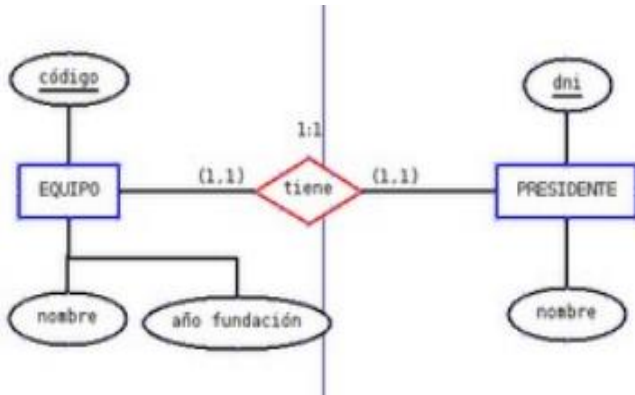
`matrimonio(#Código hombre, #Código mujer)`

### Claves externas:



`Código hombre` referencia a HOMBRE.

`Código mujer` referencia a MUJER.

# Ejemplo de Relación 1:1



En este ejemplo, podemos propagar la clave de cualquier tabla a la tabla resultante de la otra. Es decir, tenemos dos opciones, o mover la clave de PRESIDENTE a EQUIPO o mover la clave de EQUIPO a PRESIDENTE.

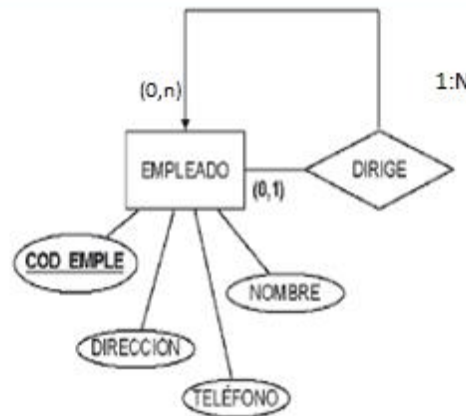
- EQUIPO(códigoEq, nombre, año\_fundación)  DC, UC
- PRESIDENTE(dni, nombre, código\_equipo(FK))
- PRESIDENTE(dni, nombre)
- EQUIPO(códigoEq, nombre, año\_fundación, dni\_presi\*(FK))  DN, UC

# Relaciones reflexivas o recursivas

- Son relaciones en las que participa un tipo de entidad. Para convertir una relación reflexiva a tabla hay que tener en cuenta sobre todo la cardinalidad. Lo normal es que toda relación reflexiva se convierta en dos tablas, una para la entidad y otra para la relación. Se pueden presentar los siguientes casos:
  - ❑ Si la relación es **1:1**, la clave de la entidad se repite, con lo que la tabla resultante tendrá dos veces ese atributo, una como clave primaria y otra como clave ajena de ella misma. No se crea la segunda tabla.
  - ❑ Si la relación es **1:M**, podemos tener dos casos:
    - a) Caso de que la entidad muchos sea siempre obligatoria se procede como en el caso 1:1.
    - b) Si no es obligatoria, se crea una nueva tabla cuya clave será la de la entidad del lado muchos, y además se propaga la clave a la nueva tabla como clave ajena.
  - ❑ Si es **N:M**, se trata igual que en las relaciones binarias. La tabla resultante de la relación contendrá dos veces la clave primaria de la entidad del lado muchos, mas los atributos de la relación si los hubiera. La clave de esta nueva tabla será la combinación de las dos.

# Relaciones reflexivas o recursivas

- Ejemplo, relación EMPLEADO-DIRIGE-EMPLEADO. Un empleado puede dirigir a muchos empleados o a ninguno. Y un empleado es dirigido por un director o por ninguno si él es el que dirige. En este caso no hay obligatoriedad en la entidad muchos.



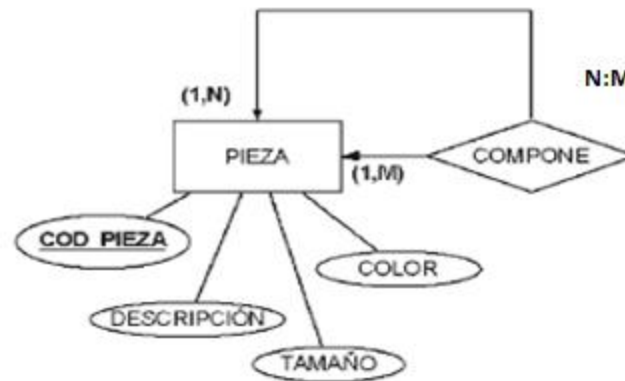
EMPLEADO (COD\_EMPL, DIRECCION, TELEFONO, NOMBRE)

DIRIGE (COD\_EMPL(FK), COD\_DIREC(FK))



# Relaciones reflexivas o recursivas

- Ejemplo, una pieza se compone de muchas piezas, que a su vez están compuestas de otras piezas, es decir, PIEZA-COMPONE-PIEZA.



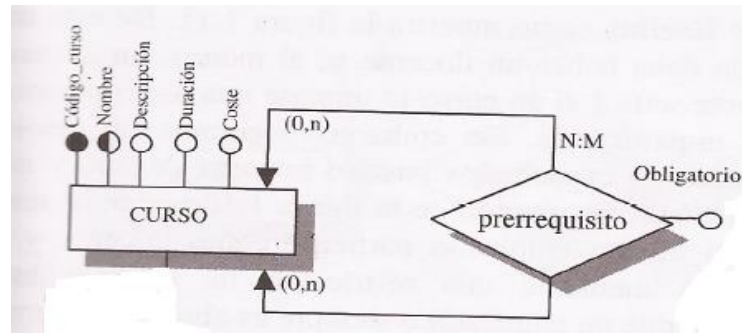
En este caso, obtenemos la segunda tabla **COMPONE\_PIEZA** en la que aparecerá repetido el código de pieza y formará la clave de la tabla. El atributo **COD\_PIEZ\_COM** representa la pieza que se compone de otras, y **COD\_PIEZA** tiene el papel de pieza que compone a otras piezas. Ambos atributos son claves ajenas a la tabla **PIEZA**.

**PIEZA** (**COD\_PIEZA**, DESCRIPCION, TAMANO, COLOR)

**COMPONE\_PIEZA** (**COD\_PIEZ\_COM**(FK), **COD\_PIEZA**(FK))

# Relaciones reflexivas o recursivas

- Ejemplo, un curso puede tener como prerequisite el haber realizado o no otro u otros cursos y a su vez el puede ser prerequisite para la realización de otro/s cursos o no.



En este caso, obtenemos la segunda tabla PRERREQUISITO en la que aparecerá repetido el código de Curso y formará la clave de la tabla. El atributo Tiene\_Pre representa el curso que tiene Prerequisite/s, y Es\_Pre tiene el papel de curso que es prerequisite para otro/s. Ambos atributos son claves ajenas a la tabla PRERREQUISITO .

CURSO (Código\_curso, Nombre, Descripción, Dirección, Coste)

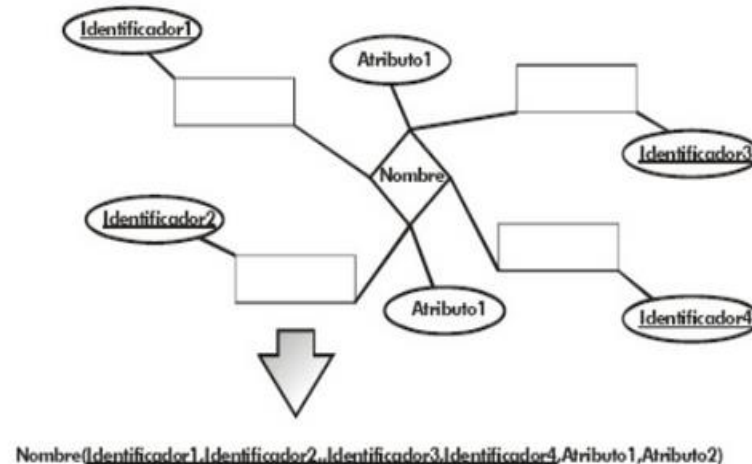
PRERREQUISITO (Tiene\_Pre(FK), Es\_Pre(FK), Obligatorio)

B:C,M:C B:C,M:C

## 2.- Transformación de un diagrama E/R al Modelo Relacional

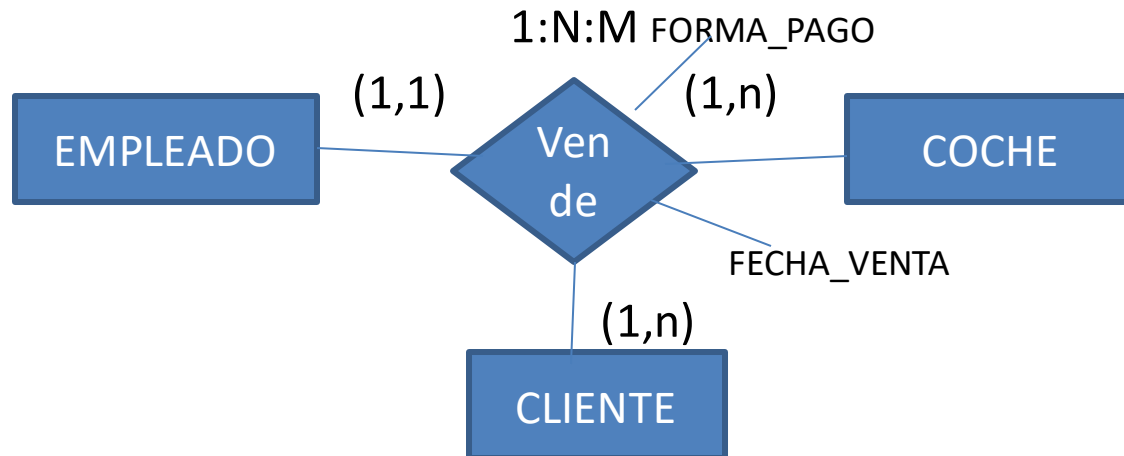
### ✓ Conversión de relaciones no binarias a tablas.

Las relaciones ternarias, cuaternarias y *n-arias* que unen más de dos relaciones se transforman en una tabla que contiene los atributos de la relación más los identificadores de las entidades relacionadas. La clave la forman todas las claves externas:



# Conversión de relaciones no binarias a tablas

- Suponemos el caso de una tienda de venta de coches, en la que un empleado 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.



CLIENTES (CD\_CLIENTE, NOMBRE, TLF)

EMPLEADO (COD\_EMPLEADO, NOMBRE, TLF, SALARIO, FECHA\_ALTA)

COCHES (COD\_COCHE, MATRICULA, MODELO, PRECIO)

VENTA (COD\_COCHE(FK), CD\_CLIENTE(FK), COD\_EMPLEADO(FK),  
FORMA\_PAGO, FECHA\_VENTA)

## 2.- Transformación de un diagrama E/R al Modelo Relacional

- El modelo relacional no dispone de mecanismos para la representación de las relaciones jerárquicas, por lo tanto, se tienen que eliminar. Para ello hay que tener en cuenta:
  - La especialización que los subtipos tienen respecto a los supertipos, es decir, los atributos diferentes que tengan asociados cada uno de los subtipos, que son los que se diferencian con el resto de atributos de los otros subtipos.
  - El tipo de especialización que representa el tipo de relación jerárquica: *total o parcial exclusiva*, y *total o parcial solapada*.
  - Otros tipos de relación que mantengan tanto los subtipos como el supertipo.
  - La forma en la que se va a acceder a la información que representan tanto el supertipo como el subtipo.

# Generalizaciones:

Para pasar estas relaciones al modelo relacional se aplicará una de las siguientes reglas:

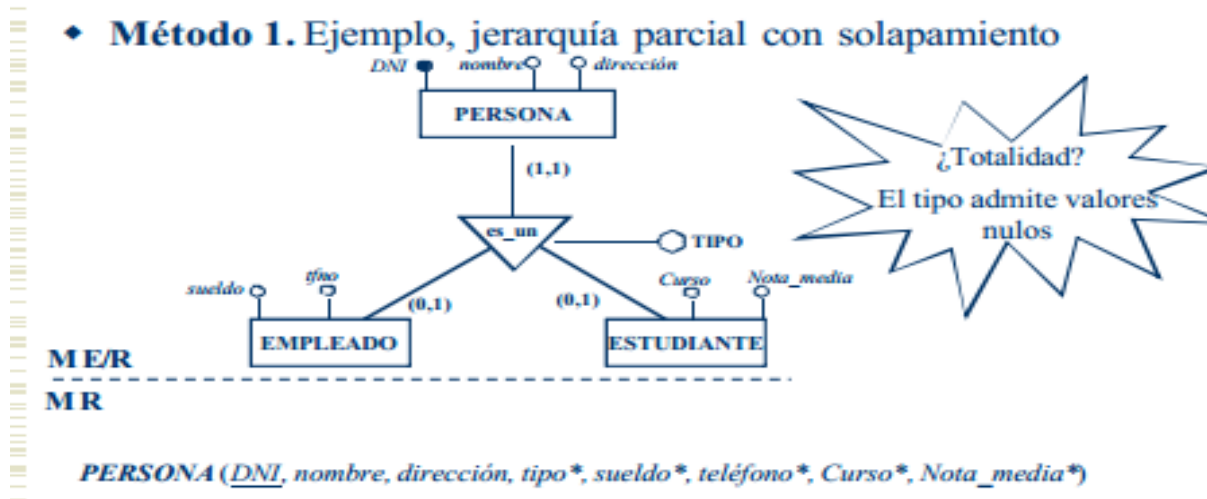
- **A/ Integrar todas las entidades en una única eliminando a los subtipos.** Esta nueva entidad contendrá todos los atributos del supertipo, todos los de los subtipos, y los atributos discriminativos para distinguir a qué subentidad pertenece cada atributo.
- Todas las relaciones se mantienen con la 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.
- El atributo tipo sería obligatorio para indicar el tipo de hijo.

# A/ Eliminación de los subtipos

## • Inconvenientes:

- ✓ Proliferación de valores nulos en los atributos procedentes de los subtipos.
- ✓ Cuando se pretende consultar un subtipo concreto, se accede a información no requerida (de los restantes subtipos).

Se recomienda cuando los subtipos tienen pocos atributos



Los atributos y relaciones de cada subtipo se transfieren al supertipo.

# A/ Eliminación de los subtipos



Persona (Cédula, Nombre, Apellido, Dirección, **Tipo**, Salario, Costo\_Hora, Carrera)

Donde **Tipo** puede ser 0 para la subclase Empleado, 1 para la subclase Profesor o 2 para la subclase Estudiante

<12453334, 'Pedro', 'Perez', 'Av. 8', 0, 2000, **NULL**, **NULL**>



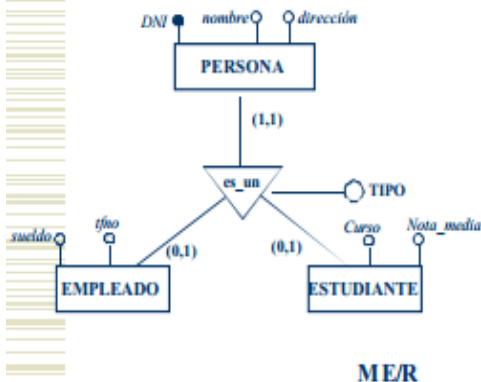
# Generalizaciones

- **B/ 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. Solo puede ser aplicada para **jerarquías totales y exclusivas**. Inconvenientes que presenta esta transformación:
  - Se crea redundancia en la información, pues los atributos del supertipo se repiten en cada uno de los subtipos.
  - El número de relaciones aumenta, pues si el supertipo tiene relaciones, estas pasan a cada uno de los subtipos.

# B/ Eliminación del Supertipo

- Los atributos del supertipo se añaden a la tabla de cada uno de los subtipos.
- Las relaciones en las que aparece el supertipo se aplican a cada uno de los subtipos.

## ♦ Método 3. Ejemplo, jerarquía parcial con solapamiento



*EMP(DNI, nombre, dir, sueldo, tfno)*

*EST(DNI, nombre, dir, curso, nota\_media)*

MR

## ● Inconvenientes:

–Se introduce redundancia.

–Se incrementa el número de relaciones.

–Se pierde semántica, de forma que hay que acceder a varias tablas para recuperar la información común.

● Se recomienda cuando el supertipo tiene pocos atributos y participa en pocas relaciones. Se suele utilizar en generalizaciones totales y disjuntas.

mantienen y cada subtipo se identificara con la clave aiena del supertipo.

## **C/ Supertipo y Subtipos**

El supertipo mantendra una relación 1:1 con cada subtipo. Este tipo valdrá para generalizaciones totales, parciales, inclusivas o exclusivas.

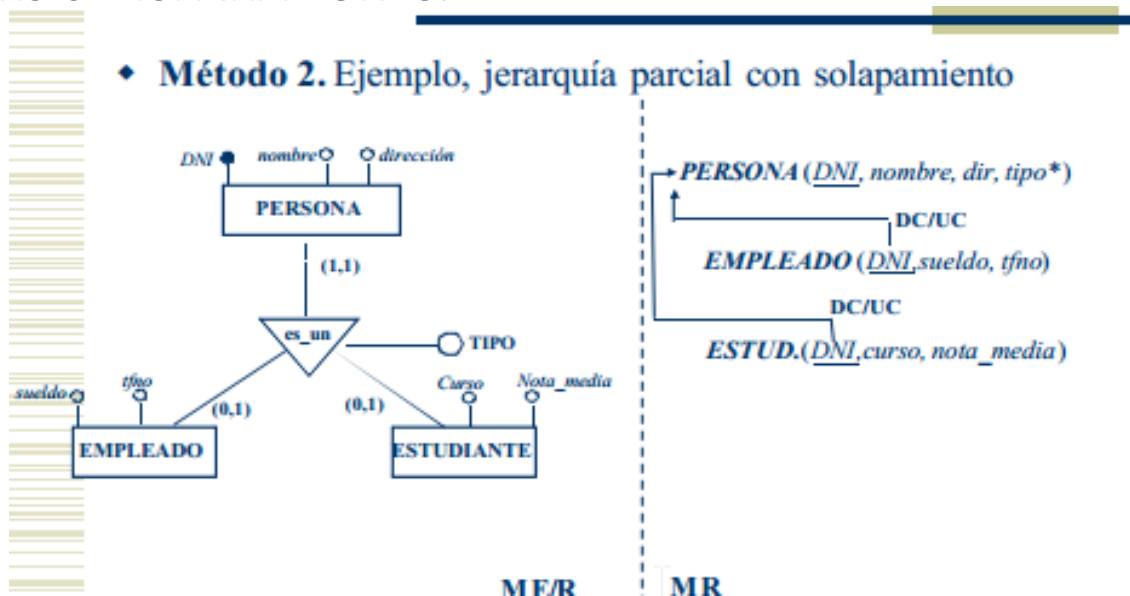
# C/ Supertipo y Subtipos

## ✓ Generalizaciones:

Utilizar una relación para representar al supertipo y tantas relaciones como subtipos haya (con la clave de la entidad padre). Habrá que añadir un atributo que indique el tipo de entidad al que se hace referencia. Se utiliza cuando:

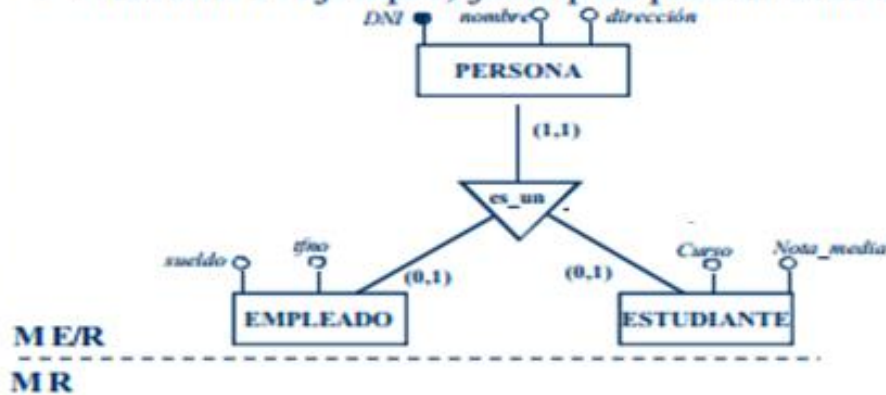
- los subtipos tienen atributos dispares y/o interrelaciones diferentes
- Incorporar mayor semántica en el grafo relacional

Será necesario implementar las restricciones semánticas necesarias a través de CHECKS o DISPARADORES.



# D/ Eliminación de los subtipos. Utilización de Booleanos

- Método 1. Ejemplo, jerarquía parcial con solapamiento



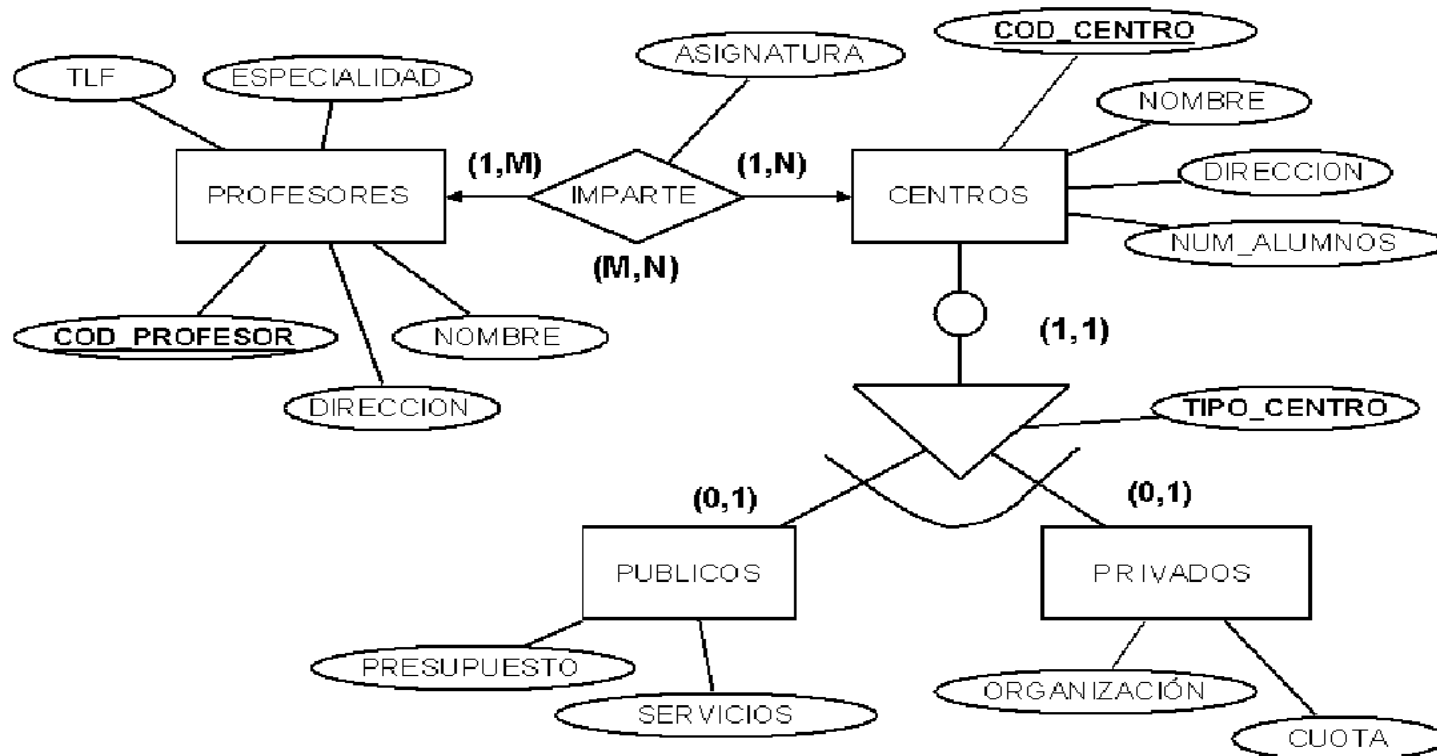
PERSONA(DNI, nombre, dirección, Es\_Empleado, sueldo\*, telefono\*, Es\_Estudiante, curso\*, notaMedia\*)

<12453334, 'Pedro Perez', 'Av. 8', true, 2000, 911232323, true, 'ASIR1', 5>

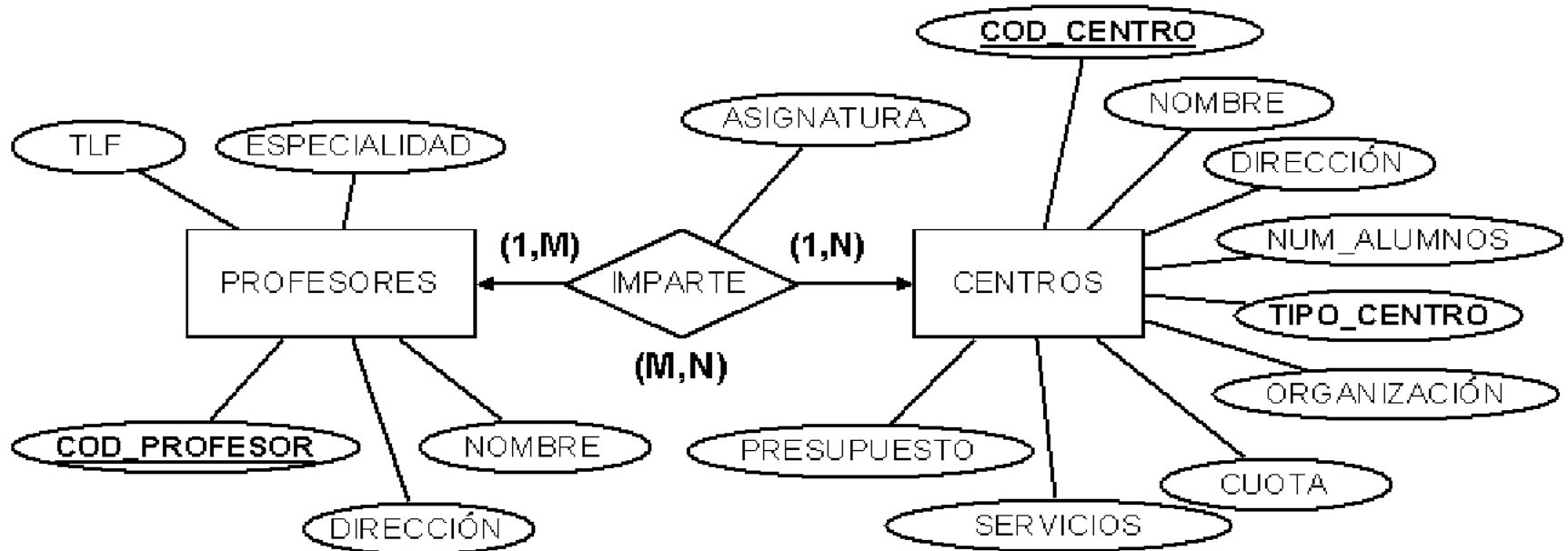
<12453334, 'Pedro Perez', 'Av. 8', true, 2000, 911232323, **false, NULL, NULL**>

# Ejemplo Generalización

- Consideramos los profesores que imparten clases en dos tipos de centros educativos: públicos y privados. Un profesor puede impartir clase en varios centros, ya sean públicos o privados. Consideramos la asignatura como atributo de la relación entre profesor imparte en centro. Los centros educativos solo pueden ser de estos dos tipos. Un centro público no puede ser a la vez privado. Los atributos específicos para los centros públicos son el presupuesto y los servicios, para los privados la organización y la cuota. La jerarquía a representar es total y exclusiva



# Opción I: Eliminar Subtipos

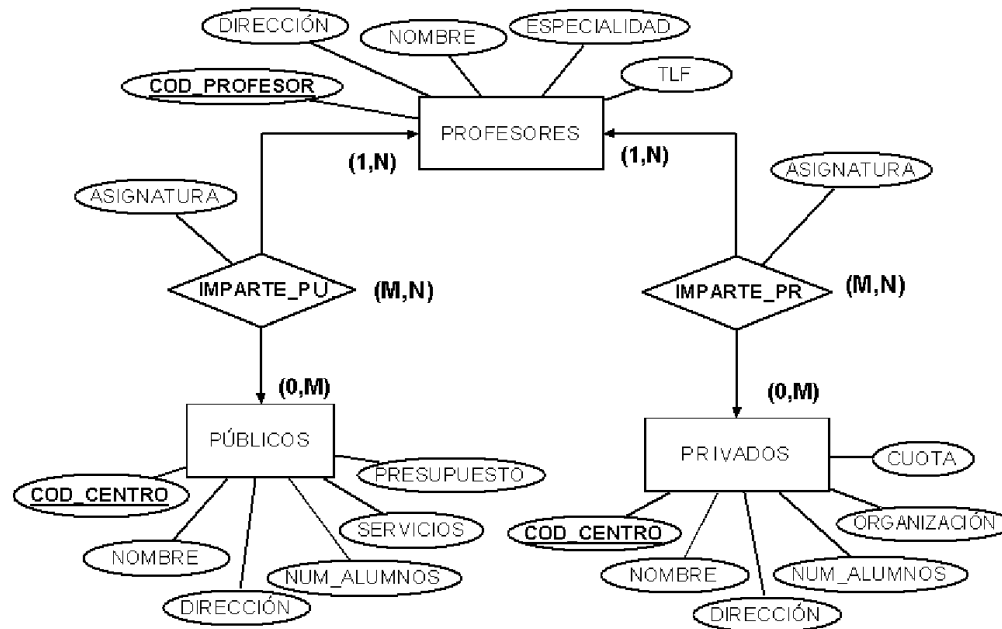


PROFESORES (COD\_PROFESOR, DIRECCION, NOMBRE, TLF, ESPECIALIDAD)

CENTROS (COD\_CENTRO, NOMBRE, DIRECCION, NUM\_ALUMNOS, TIPO\_CENTRO, ORGANIZACIÓN\*, CUOTA\*, SERVICIOS\*, PRESUPUESTO\*)

IMPARTE (COD\_PROFESOR(FK), COD\_CENTRO(FK), ASIGNATURA)

# Opción II: Eliminar Supertipo



PROFESORES (COD\_PROFESOR, DIRECCION, NOMBRE, TLF, ESPECIALIDAD)

PUBLICOS (COD\_CENTRO, NOMBRE, DIRECCION, NUM\_ALUMNOS, SERVICIOS, PRESUPUESTO)

PRIVADOS (COD\_CENTRO, NOMBRE, DIRECCION, NUM\_ALUMNOS, ORGANIZACION, CUOTA)

IMPARTE\_PR (COD\_PROFESOR(FK), COD\_CENTRO(FK), ASIGNATURA)

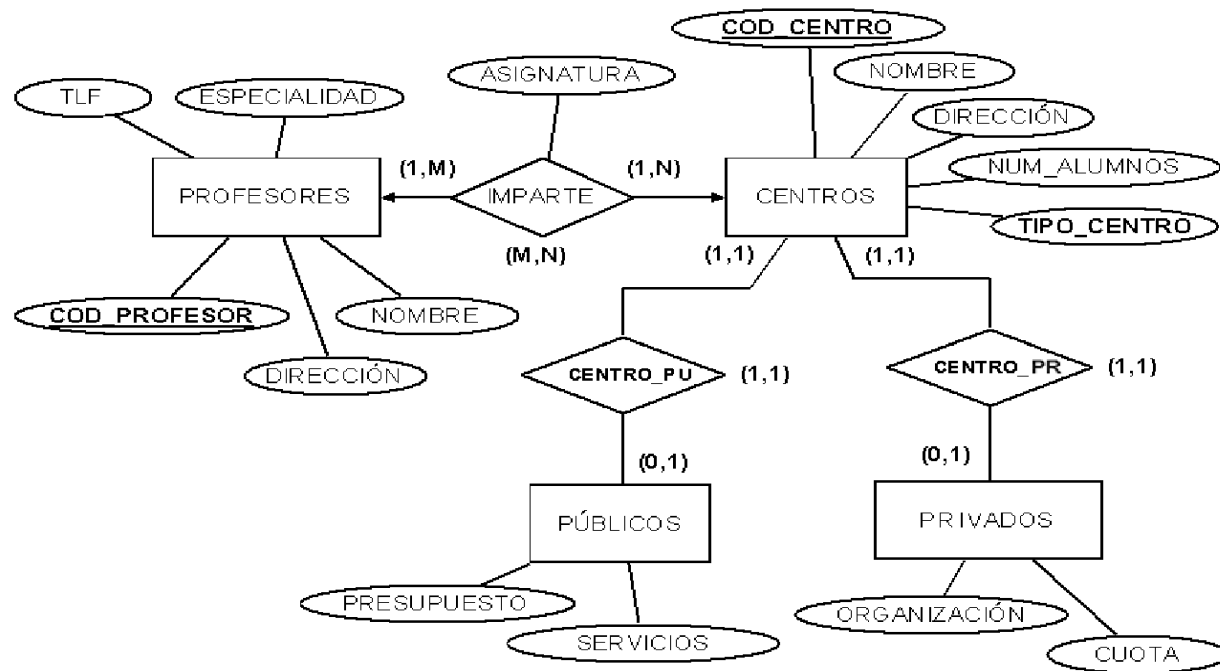
IMPARTE\_PU (COD\_PROFESOR(FK), COD\_CENTRO(FK), ASIGNATURA)



# Opción II: Eliminar Supertipo

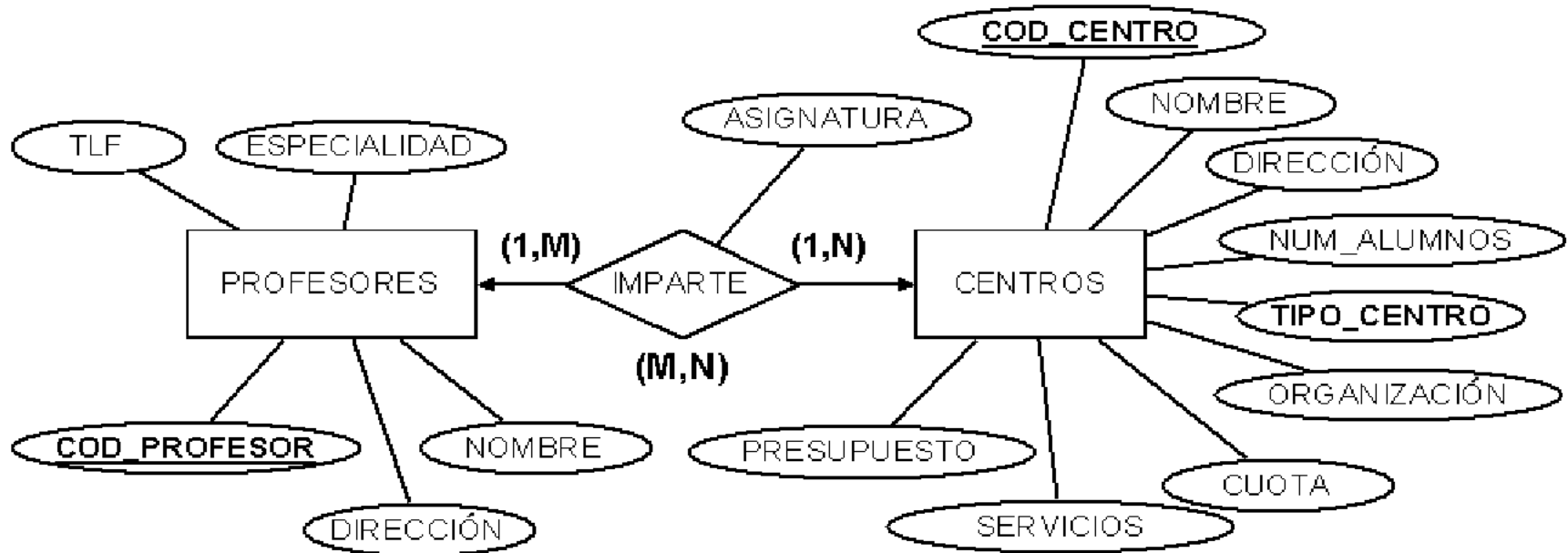
- Las tablas PUBLICOS y PRIVADOS tiene la misma clave, y no existe ningún condicionante lógico que impida que un centro este en las dos tablas, algo que va en contra del enunciado del problema. Para controlar esta situación habrá que recurrir a los programas que manejen las tablas o a la creación de triggers asociados a cada una de las tablas.
- Podríamos pensar que las tablas IMPARTE\_PR e IMPARTE\_PU, son iguales y podrían agruparse en una sola, sin embargo, habría problemas con el control de la integridad puesto que habría que definir la clave de esta tabla como clave ajena de las tablas PUBLICOS y PRIVADOS, y esto no seria correcto.

# Opción III: Insertar una relación 1:1 entre el supertipo y cada uno de los subtipos



- PROFESORES (COD\_PROFESOR, DIRECCION, NOMBRE, TLF, ESPECIALIDAD)
- CENTROS (COD\_CENTRO, NOMBRE, DIRECCION, NUM\_ALUMNOS, TIPO\_CENTRO)
- PUBLICOS (COD\_CENTRO(FK), SERVICIOS, PRESUPUESTO)
- PRIVADOS (COD\_CENTRO(FK), ORGANIZACION, CUOTA)
- IMPARTE (COD\_PROFESOR(FK), COD\_CENTRO(FK), ASIGNATURA)

# Opción IV: Eliminar Subtipos. Utilización de Booleanos en el Supertipo



PROFESORES (COD\_PROFESOR, DIRECCION, NOMBRE, TLF, ESPECIALIDAD)

CENTROS (COD\_CENTRO, NOMBRE, DIRECCION, NUM\_ALUMNOS, Es\_Privado, ORGANIZACIÓN\*, CUOTA\*, Es\_Publico, SERVICIOS\*, PRESUPUESTO\*)

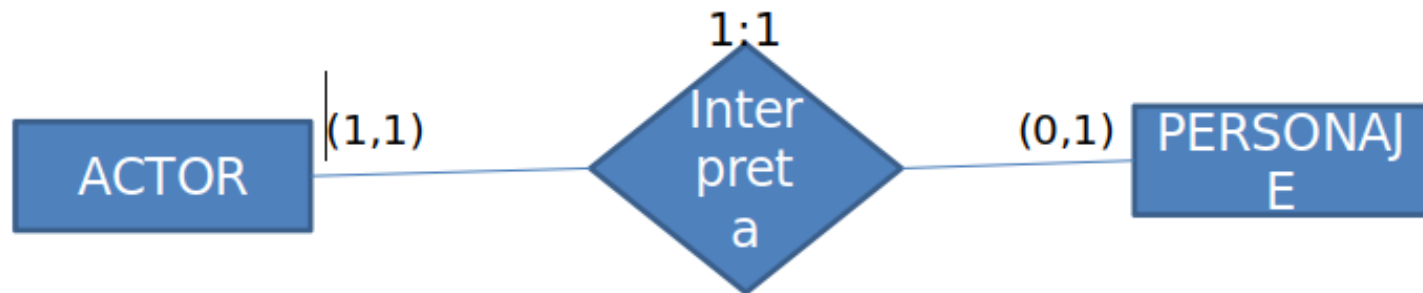
IMPARTE (COD\_PROFESOR(FK), COD\_CENTRO(FK), ASIGNATURA)

# ACTIVIDAD 1

- Realizar el Modelo E/R y El Relacional de los siguientes supuestos:
  - 1/ Un Actor (CodActor, Nombre) puede interpretar a un Personaje (CodPersonaje, Nombre, Pelicula), un Personaje sólo es interpretado por un único Actor.

# ACTIVIDAD 1

- Realizar el Modelo E/R y El Relacional de los siguientes supuestos:  
1/ Un Actor puede interpretar a un Personaje, un Personaje sólo es interpretado por un único Actor



ACTORES (CodActor, Nombre)  
Clave Foránea

PERSONAJES(CodPersonaje, Nombre, Película, CodActor\*)

DN:UC

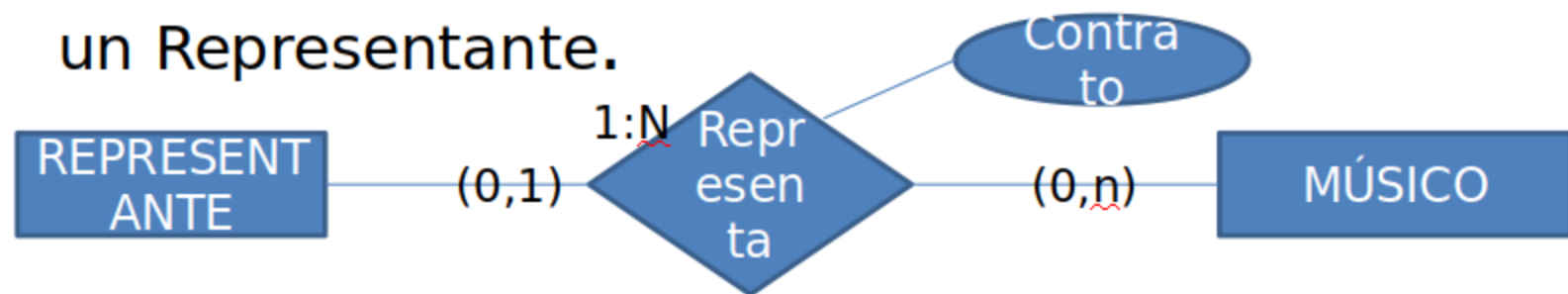


# ACTIVIDAD 2

2/ Un Representante (NumLicencia, Nombre) puede gestionar la carrera de varios Músicos (Codigo,NombreArtistico,NombreReal), sin embargo, un Músico sólo puede tener un Representante, se generará un Contrato.

# ACTIVIDAD 2

- 2/ Un Representante gestiona la carrera de varios músicos, sin embargo, un Músico sólo puede tener un Representante.



REPRESENTANTES (NumLicencia, Nombre)

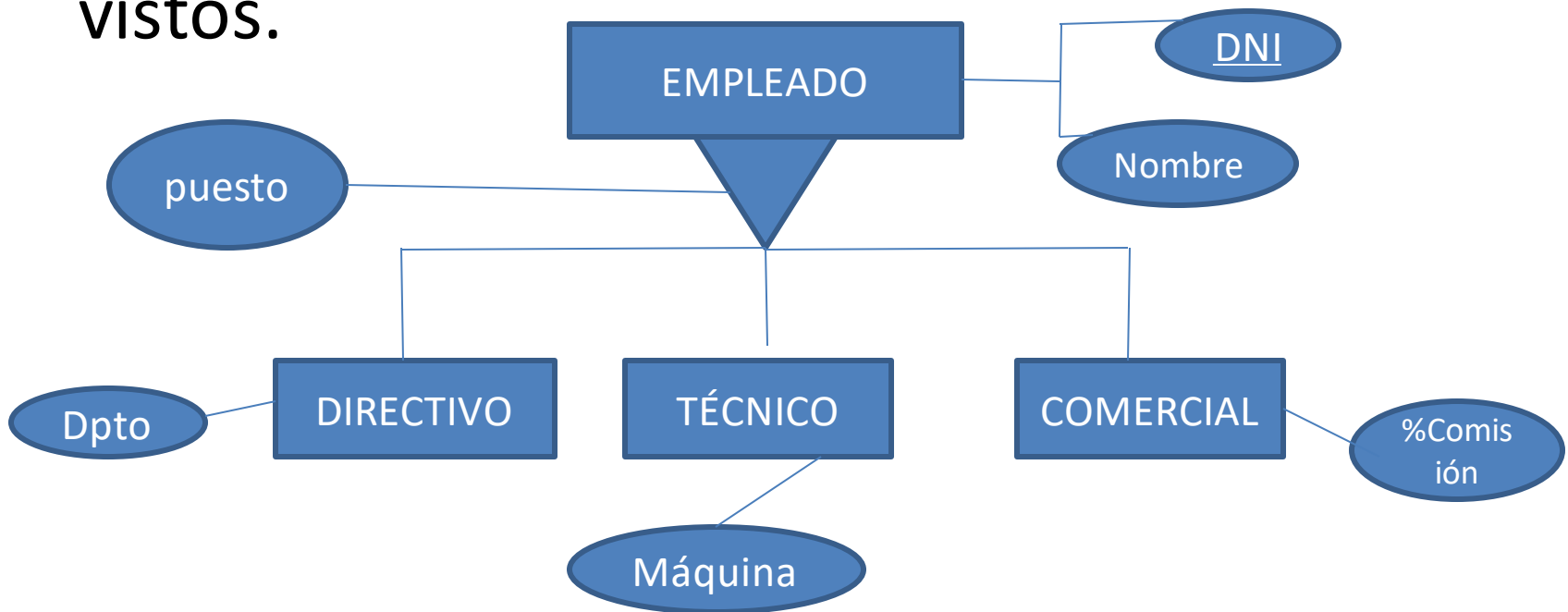
MUSICOS(Codigo, NombreArtistico, NombreReal)

REPRESENTA(CodigoMusico, NumLicenciaRepresentante, Contrato)

DC:UC                      DC:UC

# ACTIVIDAD 3

3/ Dado el siguiente diagrama E/R, pasar al Modelo Relacional, utilizando los 4 métodos vistos.





# ACTIVIDAD :



## 3.1/ Eliminar los subtipos:

Se incorporan todos los atributos de las subclases en la superclase y se añade un campo tipo, que contiene el tipo de subclase a la que pertenece cada tupla.  
(Especializaciones exclusivas)

EMPLEADOS(DNI, Nombre, Puesto/Tipo, Dpto\*, Maquinas\*, Comision\*)

**NOTA:** Si la generalización es exclusiva Puesto será obligatorio, si es inclusiva será optativo.

## 3.2/ Eliminar los subtipos.

Supertipo con atributos para distinguir los tipos de subclases.

EMPLEADOS(DNI, Nombre, Dpto\*, Maquinas\*, Comision\*, EsDirectivo, EsTecnico, EsComercial)

# ACTIVIDAD 3

## 3.3/ Eliminar el Supertipo:

Se crea una tabla para cada subclase con todos los atributos de la clase padre.

DIRECTIVOS (DNI, Nombre, Dpto)

TECNICOS (DNI, Nombre, Maquina)

COMERCIALES (DNI, Nombre, Comision)



## 3.4/ Supertipo y Subtipos:

Se crea una tabla para la Superclase y otras para cada subclase con el identificador Principal de la clase padre y sus atributos.

EMPLEADOS(DNI, Nombre, Puesto)

DIRECTIVOS (DNI(FK), Dpto)

TECNICOS (DNI(FK), Maquina)

COMERCIALES (DNI(FK), Comision)

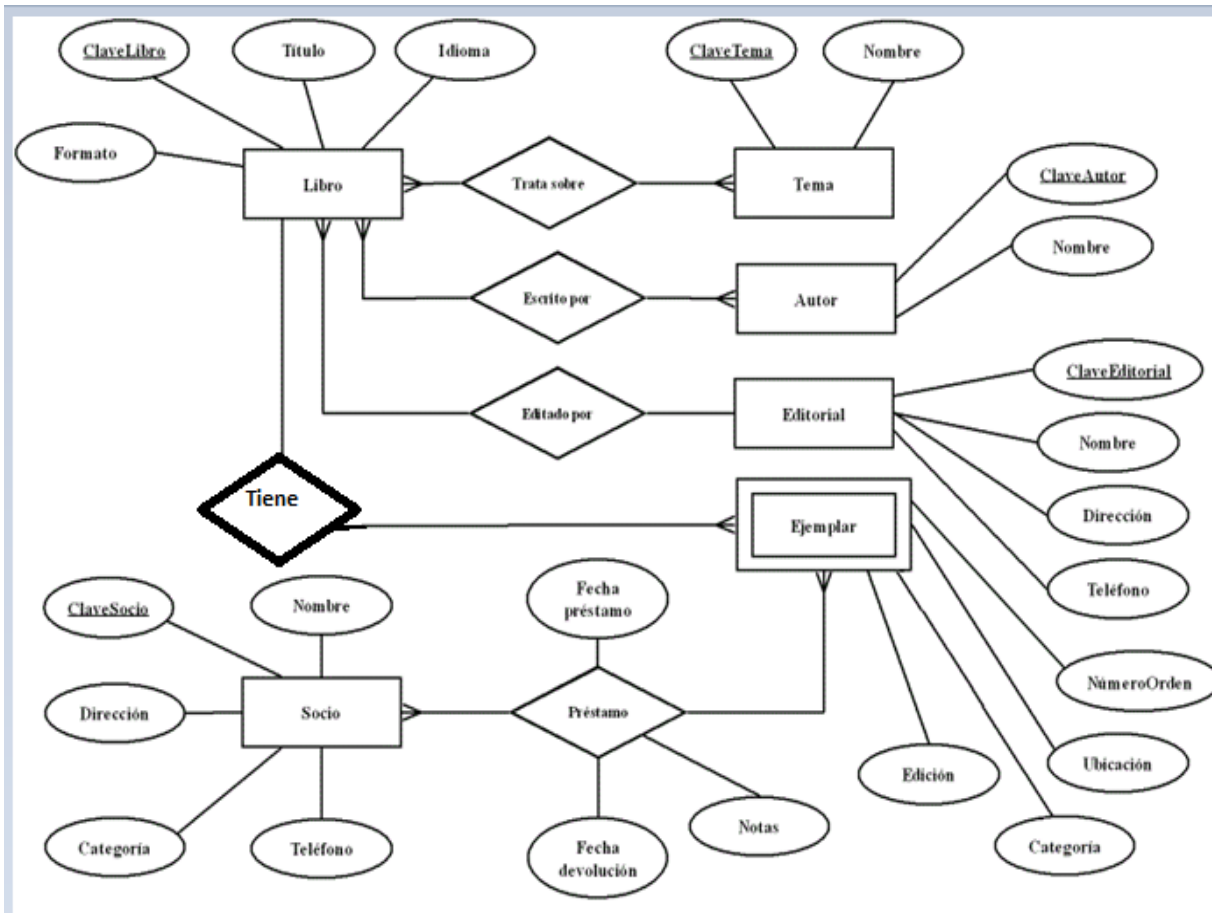
En todos los casos DC:UC

# ACTIVIDAD 4

- Se trata de gestionar una biblioteca, controlar libros, socios y préstamos. Adicionalmente se necesita un control de los ejemplares de cada libro, su ubicación y su estado, con vistas a su retirada o restitución, para esto último se necesita información sobre editoriales a las que se deben pedir los libros.
- El sistema debe proporcionar también un método de búsqueda para libros por parte de los socios, por tema, autor o título.
- Además, se debe conservar un archivo histórico de préstamos, con las fechas de préstamo y devolución, así como una nota que el responsable de la biblioteca quiera hacer constar, por ejemplo, sobre el estado del ejemplar después de su devolución.

# ACTIVIDAD 4

4/



# ACTIVIDAD 4

5/**Libros**(IdLibro,Titulo, IdEditorial(FK), Idioma,Formato)

**Temas**(IdTema, Nombre)

**TrataSobre**(IdLibro, IdTema)

**Autores**(IdAutor,Nombre)

**EscritoPor**(IdLibro(FK),IdAutor(FK))

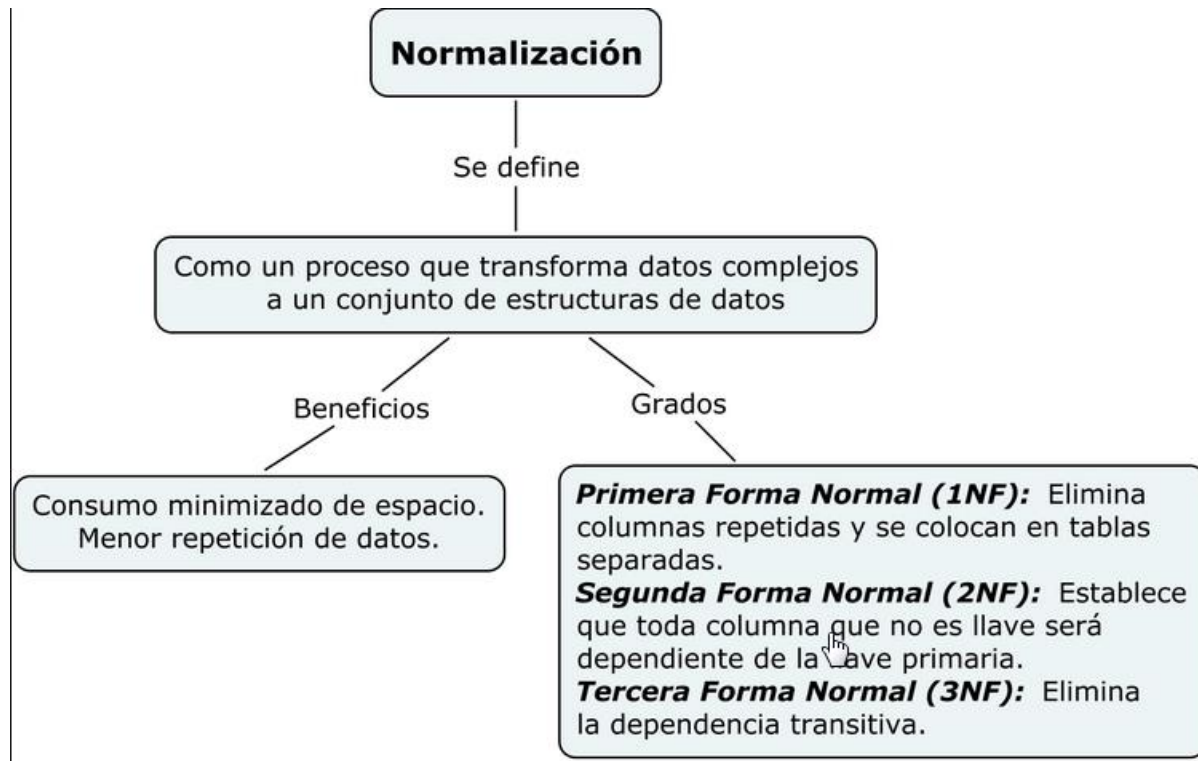
**Editoriales**(IdEditorial, Nombre, Dirección, Teléfono)

**Ejemplares**(IdEjemplar, IdLibro (FK), NumOrden,  
Edición, Categoría, Ubicación)

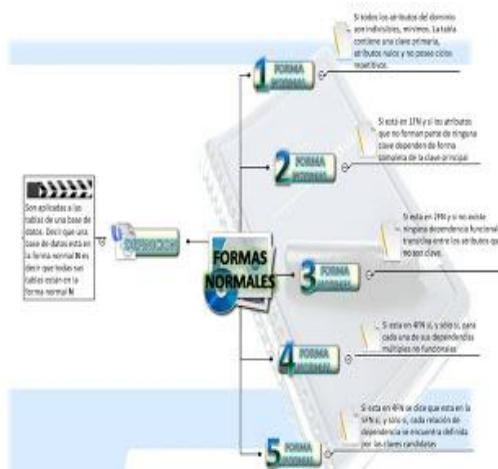
**Prestamos**(IdEjemplar(FK),IdSocio(FK),FechaPretamo,  
FechaDevolucion, Notas)

**Socio**(IdSocio, Nombre, Categoría, Dirección, Teléfono)

# 3.- Normalización



# 3.- Normalización



El proceso de normalización de bases de datos consiste en aplicar una serie de reglas a las relaciones obtenidas tras el paso del modelo entidad-relación al modelo relacional.

Las bases de datos relacionales se normalizan para:

- Evitar la redundancia de los datos.
- Evitar problemas de actualización de los datos en las tablas.
- Proteger la integridad de los datos.

## Formas Normales:

Las formas normales son aplicadas a las tablas de una base de datos. Decir que una base de datos está en la forma normal N es decir que todas sus tablas están en la forma normal.

Existen 5 formas normales.

# PRIMERA FORMA NORMAL

- Una tabla está en Primera Forma Normal si:
- Todos los atributos son atómicos. Un atributo es atómico si los elementos del dominio son simples e indivisibles.
- La tabla contiene una clave primaria única.
- La clave primaria no contiene atributos nulos.
- No debe existir variación en el número de columnas.
- Los Campos no clave deben identificarse por la clave (Dependencia Funcional)
- Debe Existir una independencia del orden tanto de las filas como de las columnas, es decir, si los datos cambian de orden no deben cambiar sus significados
- Esta forma normal elimina los valores repetidos dentro de una Base de Datos.



# PRIMERA FORMA NORMAL

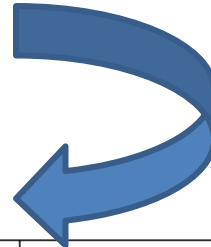
- Para que una base de datos cumpla la primera forma normal, cada columna debe ser atómica, es decir, no está permitido que en una tabla haya atributos que puedan tomar más de un valor. Tampoco pueden existir tuplas idénticas.
- Para aplicar la primera forma bastará con dividir cada columna no atómica en tantas columnas atómicas como sea necesario.

nombre	teléfono
John Smith	45 35 45 12 35 46 78 98
Carmen Aguilar	55 25 12 45 54 36 11 28

nombre	teléfono
John Smith	45 35 45 12
John Smith	35 46 78 98
Carmen Aguilar	55 25 12 45
Carmen Aguilar	54 36 11 28

# PRIMERA FORMA NORMAL

MATRICULA	CALIFS
331678	CB-001 10, MA-001 9
337890	FS-001 7, HD-002 8
337777	CB-002 7, CS-056 8
446789	MA-031 7, CB-072 8, HD-002 9



MATRICULA	CLAVE	CALIF
331678	CB-001	10
331678	MA-001	9
337890	FS-001	7
337890	HD-002	8
337777	CB-002	7
337777	CS-056	8
446789	MA-031	7
446789	CV-072	8
446789	HD-002	9

# SEGUNDA FORMA NORMAL

- Un diseño se encuentra en 2FN si está en 1FN y además cada atributo que no forma parte de la clave tiene dependencia completa de la clave principal, es decir, que no existen dependencias parciales. (Todos los atributos que no son clave principal deben depender únicamente de la clave principal). Esta regla significa que en una relación sólo se debe almacenar información sobre un tipo de entidad, y se traduce en que los atributos que no aporten información directa sobre la clave principal deben almacenarse en una relación separada. Por ejemplo:
- COMPRAS(CodProducto,CodProveedor, NombreProd, Cantidad, FechaCompra)  
CodProducto -> NombreProd, por tanto al no ser dependencia funcional completa no está en 2FN.
- Lo primero que necesitamos para aplicar esta forma normal es identificar las claves candidatas.

# TERCERA FORMA NORMAL

- La tercera forma normal consiste en eliminar las dependencias transitivas. “Una relación esta en 3FN si y solo si (sii) está en 2FN y además se cumple que todos los atributos de la relación no dependen transitivamente de la clave primaria.
- Es decir, todas las columnas que no sean claves dependen de la clave completa de forma no transitiva.
- En la práctica significa que se debe eliminar cualquier relación que permita llegar a un mismo dato de dos o más formas diferentes.
- No se da la situación de que un campo dependa de la clave y otro dependa del primero.

# TERCERA FORMA NORMAL

- Tomando como ejemplo la tabla siguiente:

NUM_PROV	NOM_PROV	CIUDAD	ESTADO
S1	JUAN	ACAPULCO	2
S2	PEDRO	MERIDA	4
S3	ANA	MERIDA	4
S4	RENE	ACAPULCO	2
S5	PATRICIA	REYNOSA	6
S6	RENATA	REYNOSA	6

Se observa que el atributo ESTADO depende realmente de CIUDAD y no directamente del NUM\_PROV. Para poder normalizar la relación anterior es necesario realizar dos proyecciones y las tablas resultantes serian:

NUM_PROV	NOM_PROV	CIUDAD
S1	JUAN	ACAPULCO
S2	PEDRO	MERIDA
S3	ANA	MERIDA
S4	RENE	ACAPULCO
S5	PATRICIA	REYNOSA
S6	RENATA	REYNOSA

CIUDAD	ESTADO
ACAPULCO	2
MERIDA	4
REYNOSA	6

# TERCERA FORMA NORMAL

- Tenemos una tabla donde se almacenen datos relativos a ciudades, y una de las columnas es el país y otra el continente al que pertenecen. Por ejemplo:
- **Ciudades**(ID\_ciudad, Nombre, población, superficie, renta, país, continente)

Ciudades						
<u>ID_ciudad</u>	Nombre	población	superficie	renta	país	continente
1	Paris	6000000	15	1800	Francia	Europa
2	Lion	3500000	9	1600	Francia	Europa
3	Berlin	7500000	16	1900	Alemania	Europa
4	Pekin	19000000	36	550	China	Asia
5	Bonn	6000000	12	1900	Alemania	Europa

- Para cada aparición de un determinado país, el continente siempre es el mismo. Es decir, existe una redundancia de datos.
- Existe una relación entre país y continente, y ninguna de ellas es clave candidata. Por lo tanto, si queremos que esta tabla sea 3FN debemos separar esa columna:
- **Ciudades**(ID\_ciudad, Nombre, población, superficie, renta, nombre\_pais)  
**Países**(nombre\_pais, nombre\_continente)

# TERCERA FORMA NORMAL

## Ciudades

<u>ID_ciudad</u>	<u>Nombre</u>	<u>población</u>	<u>superficie</u>	<u>renta</u>	<u>país</u>
1	Paris	6000000	15	1800	Francia
2	Lion	3500000	9	1600	Francia
3	Berlin	7500000	16	1900	Alemania
4	Pekin	19000000	36	550	China
5	Bonn	6000000	12	1900	Alemania

## Países

<u>país</u>	<u>continente</u>
Francia	Europa
Alemania	Europa
China	Asia

Regla	Descripción
<b>Primera Forma Normal (1FN)</b>	Incluye la eliminación de todos los grupos repetidos
<b>Segunda Forma Normal (2FN)</b>	Asegura que todas las columnas que no son llave sean completamente dependientes de la llave primaria (Primary Key).
<b>Tercera Forma Normal (3FN)</b>	Elimina cualquier dependencia transitiva. Una dependencia transitiva es aquella en la cual las columnas que no son llave son dependientes de de otras columnas que tampoco son llave.

# ACTIVIDAD 2

## STARTREKFANS

- Un club de fans de la película **startrek**, ha decidido crear una página web donde se pueda consultar la información referente a todas las películas y capítulos de la saga. El dominio startrekfans.com se redirigirá a un servidor web que consulte una base de datos con la siguiente información:

[Actividad7\\_Startrekfans.doc](#)



# ACTIVIDAD 3

- **STARTREKFANS**

El club de fans de Startrek ha pensado ampliar los requisitos de la página Web para hacer una segunda versión. Esta segunda versión consiste en incluir información extra para los personajes. De esta manera, si el personaje es un humano, se indicará su fecha de nacimiento y ciudad terráquea donde nació. Si el personaje es de la raza Vulcano, se almacenará el nombre del mentor y la fecha de graduación, y si es de raza Klingon, se guardará su planeta natal y la fecha de su último combate.

- Realiza una generalización de la entidad Personaje indicando las especializaciones necesarias.
- Representa el nuevo Modelo relacional de la generalización.

# ACTIVIDAD 4

- **STARTREKFANS**

El club de fans de Startrek quiere la tercera versión de la BD de la siguiente forma:

En cada capítulo la nave que viaja a un planeta puede disponer de una nave pequeña llamada lanzadera con la que bajan a la superficie del planeta. La existencia de la lanzadera, sólo tiene sentido si existe la nave a la que pertenece. Se identificará cada lanzadera mediante un número entero y el código de la nave. Es necesario conocer la capacidad en personas de la lanzadera.

➤ Representa el nuevo Modelo relacional

# 4.- MySQL Workbench

The screenshot shows the MySQL Workbench 6.2 CE Setup Wizard window in the foreground, which has ended prematurely. The background is the MySQL Downloads page for Workbench 6.2 CE. A small error dialog box is also visible, stating that Visual C++ 2013 is required.

**MySQL Workbench 6.2 CE Setup Wizard ended prematurely**

The wizard was interrupted before MySQL Workbench 6.2 CE could be completely installed.

Your system has not been modified. To complete installation at another time, please run setup again.

Click Finish to exit the wizard.

**MySQL Resources: MySQL Workbench Prerequisites**

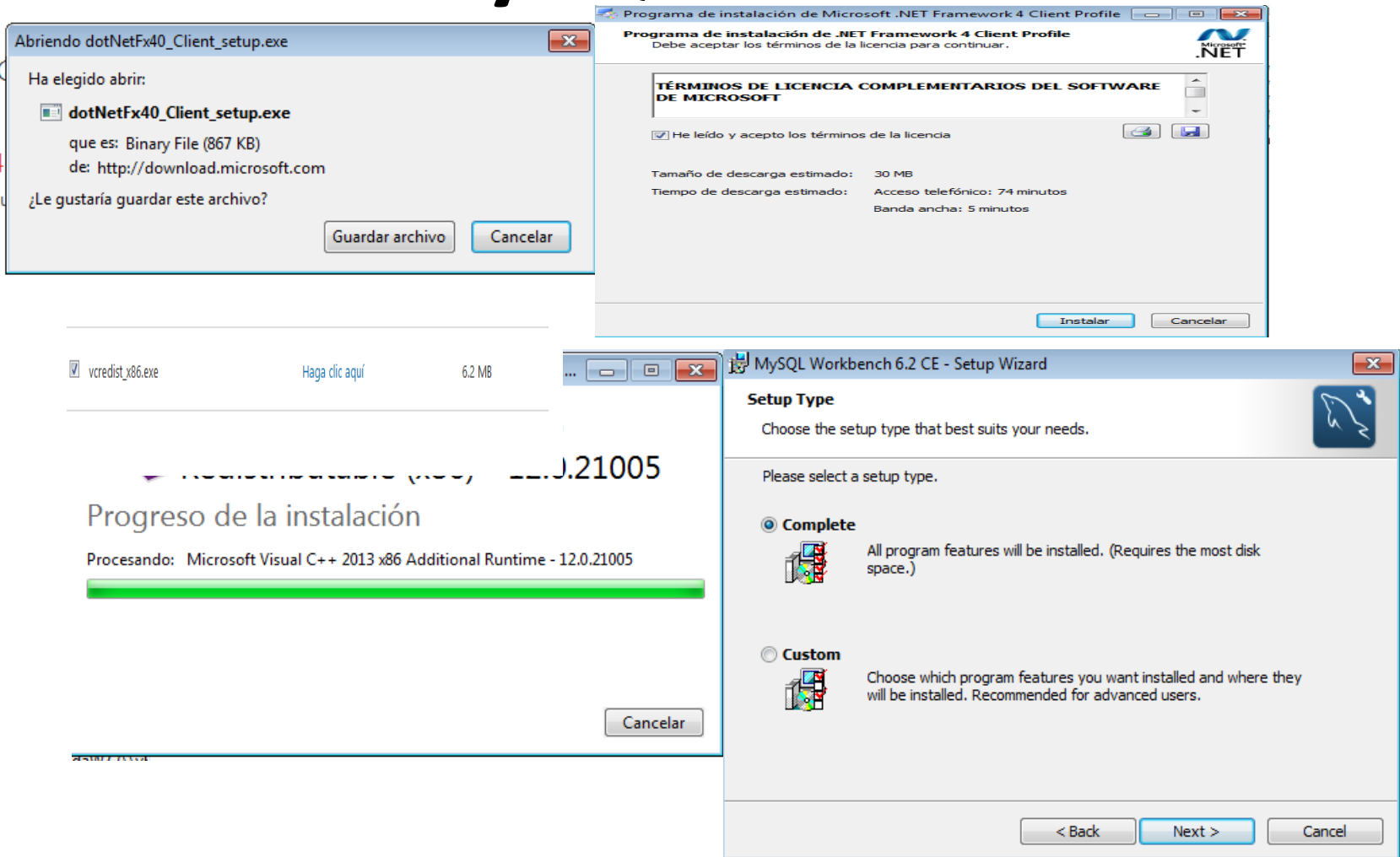
**MySQL Workbench Prerequisites**

To be able to install and run MySQL Workbench 6.2 your System needs to have libraries listed below installed. The listed items are provided as links to the corresponding download pages where you can fetch the necessary files.

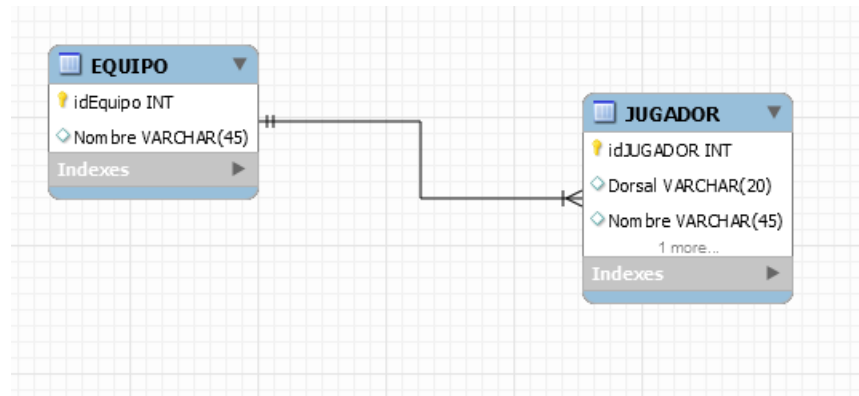
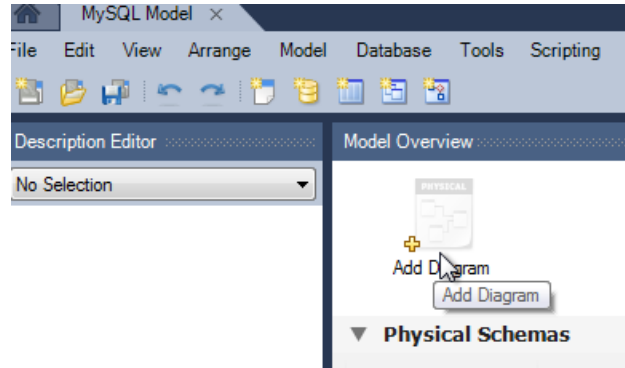
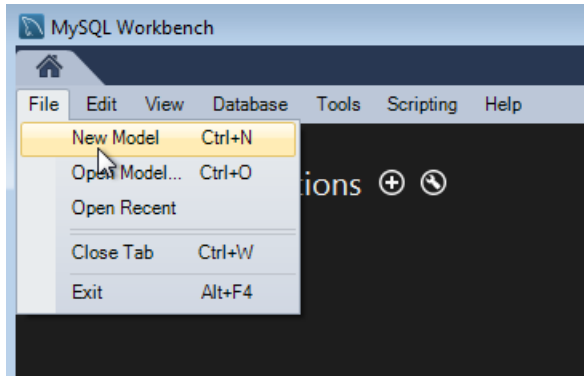
- [Microsoft .NET Framework 4 Client Profile](#)
- [Microsoft Visual C++ 2013 Redistributable Package \(x86 or x64\)](#)

MySQL Workbench 6.2 will come as win32 and win64 builds, hence the user must download the right runtime.

# 4.- MySQL Workbench

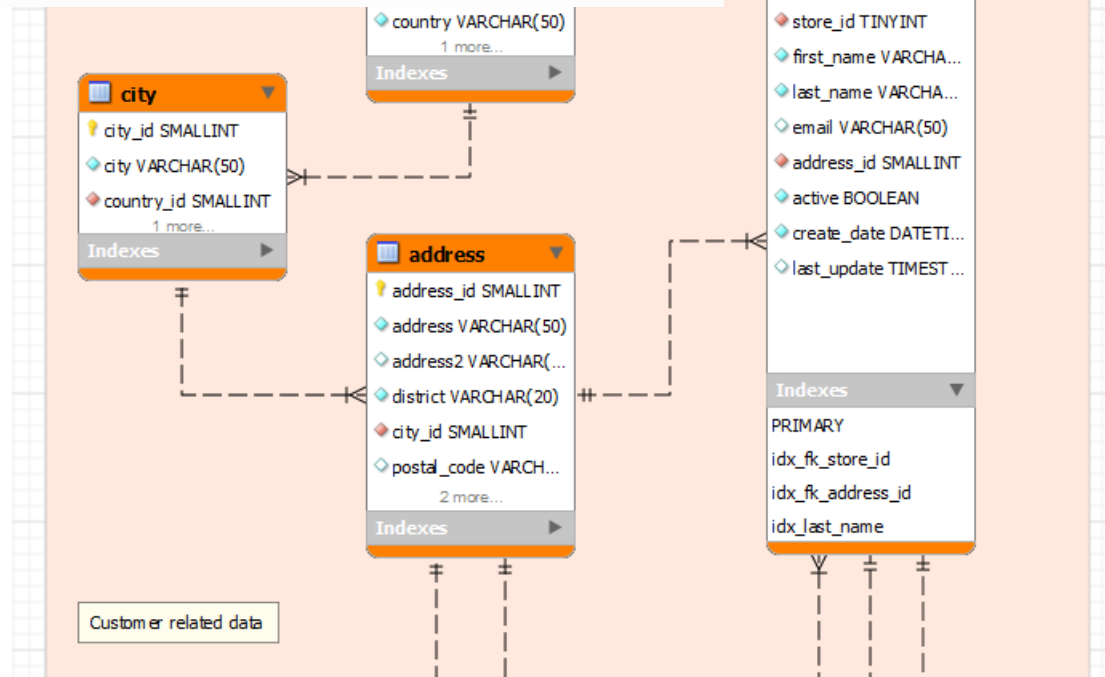


# 4.- MySQL Workbench



# 4.- MySQL Workbench

Nombre	Fecha de modifica...	Tipo	Tamaño
sakila.mwb	02/06/2009 18:07	MySQL Workbenc...	42 KB
sakila-50	28/09/2014 21:14	MySQL Workbenc...	42 KB
sakila-schema.sql	02/06/2009 18:07	Archivo SQL	3.156 KB
sakila-schema.sql	02/06/2009 18:07	Archivo SQL	23 KB



# Enlaces

- <http://uazuay.edu.ec/analisis/El%20modelo%20relacional.pdf>
- <http://ocw.uc3m.es/ingenieria-informatica/disenio-de-bases-de-datos/teoria/Tema3%28TransformacionRelacional%29.pdf>
- [http://www.unirioja.es/cu/arjaime/Temas/02.Modelo\\_E\\_R.pdf](http://www.unirioja.es/cu/arjaime/Temas/02.Modelo_E_R.pdf)
- <http://www-oei.eui.upm.es/Asignaturas/BD/BD/docbd/tema/tema2.pdf>
- <http://mysql.conclase.net/curso/?cap=004a>
- <http://ocw.uc3m.es/ingenieria-informatica/fundamentos-de-bases-de-datos/material-de-clase-1/TemaIIIbOCW.pdf>
- <http://es.slideshare.net/dante1665/modelos-relacionales-de-bases-de-datos>
- [http://www.youtube.com/watch?v=rCOEp\\_an\\_IQ](http://www.youtube.com/watch?v=rCOEp_an_IQ)