

# Unidad 3.

## Diseño lógico:

### El Modelo Relacional

# Contenido



1. Historia
2. Objetivos
3. Componentes y conceptos
4. Restricciones
5. Las 12 reglas de Codd
6. Paso de MER a modelo relacional
7. Esquema relacional
8. Normalización



# 1. Historia

# 1. Historia

- Edgar Frank **Codd**
  - ▣ Finales de los 60: define sus bases
  - ▣ 1970: *“Un modelo relacional de datos para grandes bancos de datos compartidos”*
    - Define las bases del Modelo Relacional de Bases de Datos
    - Hasta entonces sólo estaba estandarizado Codasyl
    - Se utilizó de forma masiva a partir de los 70
  - ▣ Se apoya en la teoría de conjuntos de Cantor y Childs
  - ▣ Indica que los datos se agrupan en relaciones (tablas)
    - Datos referidos a una misma entidad
    - Independiente del almacenamiento físico realizado

# 1. Historia

## □ Edgar Frank **Codd**

### ▣ Trabajaba para IBM

- No aceptó el modelo propuesto por Codd
- Siguió usando su IMS (modelo jerárquico)

### ▣ Sus teorías fueron implementadas por otras empresas

- ... especialmente Oracle

### ▣ El modelo relacional de Codd es el más usado hoy

- Casi todas las bases de datos existentes lo siguen

# 1. Historia

Año	Hecho
1970	Codd publica las bases del modelo relacional
1971-72	Primeros desarrollos teóricos
1973-78	Primeros prototipos de base de datos relacional. Son el <b>System R</b> de IBM. En ese sistema se desarrolla <b>Sequel</b> que con el tiempo cambiará su nombre a SQL.
1974	La Universidad de Berkeley desarrolla <b>Ingres</b> , SGBD relacional basado en cálculo relacional. Utilizaba el lenguaje <b>Quel</b> desarrollado en las universidades y muy popular en la época en ámbitos académicos.
1978	Aparece el lenguaje <b>QBE</b> ( <i>Query By Example</i> ) lenguaje de acceso relacional a los archivos <b>VSAM</b> de IBM
1979	Aparece <b>Oracle</b> , el primer SGBD comercial relacional (ganando en unas semanas al <b>System/38</b> de IBM). Implementa SQL y se convertirá en el sistema gestor de bases de datos relacionales líder del mercado. Codd revisa su modelo relacional y lanza el modelo <b>RM/T</b> como un intento de subsanar sus deficiencias.
1981	Aparece <b>Informix</b> como SGBD relacional para Unix
1983	Aparece <b>DB2</b> , el sistema gestor de bases de datos relacionales de IBM

# 1. Historia

Año	Hecho
1984	Aparece la base de datos <b>Sybase</b> que llegó a ser la segunda más popular (tras Oracle)
1986	ANSI normaliza el SQL ( <b>SQL/ANSI</b> ). SQL es ya de hecho el lenguaje principal de gestión de bases de datos relacionales.
1987	ISO también normaliza SQL. Es el <b>SQL ISO(9075)</b>
1988	La versión 6 de Oracle incorpora el lenguaje procedimental <b>PL/SQL</b>
1989	ISO revisa el estándar y publica el estándar <b>SQL Addendum</b> . <b>Microsoft</b> y <b>Sybase</b> desarrollan <b>SQL Server</b> para el sistema operativo <b>OS/2</b> de Microsoft e <b>IBM</b> . Durante años Sybase y SQL Server fueron el mismo producto.
1990	Versión dos del modelo relacional ( <b>RM/V2</b> ) realizada por Codd. Propuesta de <b>Michael Stonebraker</b> para añadir al modelo relacional capacidades de orientación a objetos.
1992	ISO publica el estándar <b>SQL 92</b> (todavía el más utilizado)

# 1. Historia

Año	Hecho
1992	ISO publica el estándar <b>SQL 92</b> (todavía el más utilizado)
1995	<p>Manifiesto de <b>Darwen</b> y <b>Date</b> en el que animan a reinterpretar el modelo relacional desde una perspectiva de objetos. Aparece el modelo objeto/relacional.</p> <p>Aparece <b>MySQL</b> una base de datos relacional de código abierto con licencia GNU que se hace muy popular entre los desarrolladores de páginas web.</p>
1996	<p>ANSI normaliza el lenguaje procedimental basado en SQL y lo llaman <b>SQL/PSM</b>. Permite técnicas propias de los lenguajes de programación estructurada.</p> <p>Aparece el SGBD abierto <b>PostgreSQL</b> como remodelación de la antigua Ingres, utilizando de forma nativa el lenguaje SQL (en lugar de Quel).</p>
1999	ISO publica un nuevo estándar que incluye características más avanzadas. Se llama <b>SQL 99</b> (también se le conoce como <b>SQL 200</b> )
2003	ISO publica el estándar <b>SQL 2003</b> . En él se añade SQL/PSM al estándar.
2006	Estándar ISO. <b>SQL 2006</b>
2008	Estándar ISO. <b>SQL 2008</b>
2011	Estándar ISO. <b>SQL 2011</b>





## 2. Objetivos

## 2. Objetivos

### □ Independencia física

- ▣ Forma de almacenar los datos independiente de éstos
  - Si cambia, los datos seguirán manipulándose igual
  - Modificaciones de almacenamiento físico transparentes
- ▣ El usuario sólo se preocupa por la información

### □ Independencia lógica

- ▣ Aplicaciones independientes de los elementos de la BD
  - Si se modifican elementos de la BD, las aplicaciones no
- ▣ Esquemas externos independientes del modelo lógico
  - Altas, bajas y modificaciones no influyen en las vistas

## 2. Objetivos

- Flexibilidad

- ▣ Se ofrecen vistas distintas según usuarios y aplicaciones

- Uniformidad

- ▣ Estructuras lógicas tienen una única forma conceptual
    - Tablas

- Sencillez

- ▣ Fácil de manejar en comparación con modelos anteriores

### 3. Componentes y conceptos

1. Relación o tabla
2. Tupla
3. Dominio
4. Grado y cardinalidad
5. Tablas: definición formal
6. Tablas: tipos
7. Claves
8. Valor nulo

# 3.1. Relación o tabla

- También llamada array o matriz
- No es lo mismo “relación” en MER que para Codd
- Elemento básico del modelo propuesto por Codd
- Elementos de una relación o tabla
  - ▣ Atributos
    - Propiedades que caracterizan a los elementos de la relación
      - Para la relación Personas: nombre, apellido1, nif, peso, ...
    - Columnas de la tabla
  - ▣ Tuplas
    - Cada uno de los elementos de la relación es una tupla
      - Para la relación Personas: Pepe Pérez de nif 1234, 76 Kg, ...
    - Filas de la tabla

# 3.1. Relación o tabla

Atributo 2 y sus valores

Relación o tabla de N atributos, con M tuplas

Atributo 1	Atributo 2	...	Atributo N
Tupla 1, Atributo 1	Tupla 1, Atributo 2	Tupla 1, Atributo...	Tupla 1, Atributo N
Tupla 2, Atributo 1	Tupla 2, Atributo 2	Tupla 2, Atributo...	Tupla 2, Atributo N
...	...	...	...
Tupla M, Atributo 1	Tupla M, Atributo 2	Tupla M, Atributo...	Tupla M, Atributo N

Tupla 2 y todos sus atributos

# 3.1. Relación o tabla

## □ Propiedades

- Cada tabla tiene un nombre distinto
- 1 atributo de la tabla toma 1 solo valor para cada tupla
  - Atributos multivaluados dan lugar a otra relación
- Cada atributo de una tabla tiene un nombre distinto
  - Sí se pueden repetir nombres de atributos en distintas tablas
- Cada tupla es única
  - No hay tuplas duplicadas
- El orden de los atributos no importa
- El orden de las tuplas no importa

## 3.2. Tupla

- Antes llamado registro
- Es cada una de las filas de la relación o tabla
  - ▣ Representa cada uno de los elementos almacenados allí
- Cada tupla se corresponde con un elemento del mundo real
- No puede haber dos tuplas iguales
  - ▣ Con los mismos valores para todos sus atributos



## 3.3. Dominio

- Conjunto de valores del mismo tipo
- Alberga todos los posibles valores para un atributo
- Dos atributos distintos pueden tener el mismo dominio
  - ▣ Se asigna un nombre al dominio para facilitarlo
- Técnicas para indicar su contenido
  - ▣ Intensión
    - Definición exacta de sus posibles valores
    - Ejemplo: dominio de `edades` de alumnos de ciclos formativos
      - `Números enteros entre el 16 y el 65`
  - ▣ Extensión
    - Proposición de algunos valores, y el resto se sobreentienden con ellos
    - Ejemplo: dominio de `localidad` de alumnos de ciclos formativos
      - `Ávila, Salamanca, Arévalo, Candeleda, ...`

## 3.3. Dominio

- Clasificación según posibles valores
  - ▣ Generales
    - Pueden tomar valores entre un mínimo y un máximo
    - Continuos
  - ▣ Restringidos
    - Pueden tomar uno de entre un conjunto de valores
    - Discretos

## 3.4. Grado y cardinalidad

- Grado de una relación
  - ▣ Número de atributos que la componen
  - ▣ Número de columnas de la tabla que la representa
  - ▣ Da una idea del tamaño de la relación
- Cardinalidad de una relación
  - ▣ Número de tuplas que posee
  - ▣ Número de filas de la tabla que la representa
  - ▣ Número de registros almacenados en ella

## 3.4. Grado y cardinalidad

### Nomenclaturas equivalentes

Relacional	Tabla	Ficheros
Relación	Tabla	Fichero
Tupla	Fila	Registro
Atributo	Columna	Campo
Grado	Número de columnas	Número de campos
Cardinalidad	Número de filas	Número de registros

## 3.5. Tablas: definición formal

- Elementos de la tabla o relación
  - ▣ Nombre: la identifica frente a otras
  - ▣ Cabecera: conjunto de sus pares “atributo-dominio”
    - $\{(A_i:D_i)\}_{i=1}^n$ , siendo  $n$  el grado de la relación
  - ▣ Cuerpo: conjunto de tuplas que la forman
    - $\{(A_i:V_{ij})\}$ , con  $V_{ij}$  el valor  $j$  del atributo  $A_i$  del dominio  $D_i$
  - ▣ Esquema: nombre y cabecera de la relación
    - $R \{(A_i:D_i)\}_{i=1}^n$
  - ▣ Estado: esquema y cuerpo de la relación

## 3.5. Tablas: definición formal

Alumnos				
Nombre	Apellido1	Edad	Grupo	Media
Pepe	Pérez	22	A	6.7
Marta	López	31	B	7.4
Juan Carlos	Lares	27	A	4.5

### □ Esquema

- Alumnos (Nombre:Nombre, Apellido1:Nombre, Edad:Edad, Grupo:Grupo, Media:Nota)

### □ Cuerpo

- { (Nombre:"Pepe",Apellido1:"Pérez",Edad:22, Grupo:'A', Media:6.7), (Nombre:"Marta",Apellido1:"López",Edad:31, Grupo:'B', Media:7.4), (Nombre:"Juan Carlos",Apellido1:"Lares",Edad:27, Grupo:'A', Media:4.5) }

## 3.6. Tablas: tipos

- Persistentes: sólo pueden ser borradas por los usuarios
  - ▣ Bases
    - Son independientes
    - Se crean a partir de su estructura y sus ejemplares
    - Contienen datos y metadatos
  - ▣ Vistas
    - Dependientes de otros elementos (tablas base, vistas o instantáneas)
    - Se crean a partir de una definición de consulta (sobre esos elementos)
    - Si cambian los datos de las tablas base, cambian también en las vistas
  - ▣ Instantáneas
    - Como las vistas, pero almacenan las tablas resultantes de la consulta
    - Modifican su contenido con refrescos periódicos del sistema
      - En un momento dado pueden contener datos obsoletos
- Temporales: las elimina automáticamente el sistema
  - ▣ Usadas por el SGBD para almacenar datos intermedios
    - Por ejemplo resultados de consultas

## 3.7. Claves

- Formadas por atributos de una ó varias relaciones
  - ▣ Claves simples:                    formadas por un solo atributo
  - ▣ Claves compuestas:            formadas por varios atributos
- Permiten identificar unívocamente un registro
  - ▣ A partir del valor de los atributos que la componen
- Tipos
  - ▣ Claves candidatas
  - ▣ Clave primaria
  - ▣ Claves alternativas
  - ▣ Claves externas, ajenas, foráneas, secundarias o migradas



## 3.7. Claves

- Clave candidata
  - ▣ Conjunto de atributos que identifica unívocamente una tupla de la relación
  - ▣ Toda relación ha de tener al menos una clave candidata
    - Puede tener más
  - ▣ Requisitos para ser clave candidata
    - Unicidad
      - No puede haber dos tuplas con valores repetidos para ese conjunto
    - Irreductibilidad
      - La clave es un conjunto mínimo de atributos
      - Si se elimina algún atributo que la compone, deja de ser clave
  - ▣ Ejemplo:
    - `nif, cod_alumno` y `(num_matricula, año)` para Alumnos

## 3.7. Claves

### □ Clave primaria

- ▣ Clave candidata elegida para identificar las tuplas de una relación
  - De entre todas se escoge la más significativa para la relación
- ▣ Ejemplo: `cod_alumno` para `Alumnos`

### □ Clave alternativa

- ▣ Clave candidata no seleccionada como clave primaria
- ▣ Ejemplo: `nif` y `(num_matricula, años)` para `Alumnos`

## 3.7. Claves

- Claves externas, ajenas, foráneas, secundarias o migradas
  - ▣ Claves primarias de una relación que son atributos de otra
    - Identifican, pues, a una tupla de la relación de origen
  - ▣ Llamadas así por proceder de otra relación

Ciclos		
Cod_Ciclo	Denominación	Nivel
DAW	Desarrollo de aplicaciones web	Superior
ASIR	Administración de sistemas informáticos en red	Superior
SMR	Sistemas microinformáticos y en red	Medio



Clave primaria de Ciclos

Clave secundaria de Asignaturas  
(migrada desde Ciclos)



Asignaturas			
Cod_Asig	Denominación	Horas_sem	Cod_Ciclo
PR	Programación	8	DAW
BD	Bases de datos	6	DAW
RL	Redes locales	7	SMR

## 3.8. Valor nulo

- Valor especial para algún atributo de una relación
  - ▣ Indica que no hay valor para él en esa tupla
- Se le puede asignar a atributos de cualquier dominio
- Denota
  - ▣ Que se desconoce el valor del atributo para esa tupla
    - Atributo opcional con valor desconocido
      - No se conoce el segundo apellido de un *alumno*
  - ▣ Que para esa tupla no existen valores para ese atributo
    - Atributo opcional sin valor para él
      - El *alumno* no tiene teléfono
  - ▣ Que la tupla no está asociada a otra de otra relación
    - Clave migrada opcional en una relación
      - La *asignatura* no es de ningún ciclo formativo

## 3.8. Valor nulo

### □ Precaución

- Significa “valor vacío”

- NO significa “ ”, ni ‘ ’, ni 0, ...

  - Cadena vacía, carácter espacio en blanco ó 0 son VALORES

### □ Para operaciones lógicas con valor “nulo”

- Se definen tres estados en vez de dos

  - Verdadero, Falso, Nulo

- El valor “nulo” se interpreta como un “quizás”

  - Ni Falso (0), ni Verdadero (1), sino Quizás (0.5)

## 3.8. Valor nulo

### Operaciones lógicas que incluyen valor NULO

<b>AND</b>	Verdadero	Falso	Nulo
Verdadero	Verdadero	Falso	Nulo
Falso	Falso	Falso	Falso
Nulo	Nulo	Falso	Nulo

<b>OR</b>	Verdadero	Falso	Nulo
Verdadero	Verdadero	Verdadero	Verdadero
Falso	Verdadero	Falso	Nulo
Nulo	Verdadero	Nulo	Nulo

<b>NOT</b>	Verdadero	Falso	Nulo
	Falso	Verdadero	Nulo

## 4. Restricciones

1. Restricciones inherentes
2. Restricciones semánticas

# 4. Restricciones

- Condiciones que han de cumplir las tuplas de la BD
- Tipos
  - ▣ Inherentes
    - Impuestas por el modelo relacional, no por el usuario
  - ▣ Semánticas
    - Impuestas por los diseñadores a partir de los requisitos
    - Se dividen en
      - Clave principal (*primary key*)
      - Unicidad (*unique*)
      - Obligatoriedad (*not null*)
      - Integridad referencial (*foreign key*)
      - Regla de validación (*check*)
      - Disparadores (*triggers*)



# 4.1. Restricciones inherentes

- Las impone el modelo relacional
- Generales para todos los modelos relacionales
- 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 la tupla
    - Dicho valor ha de ser del dominio al que pertenece

## 4.2. Restricciones semánticas

- Clave principal (*primary key*)
  - ▣ Clave primaria
- ▣ Atributo o conjunto de atributos de una tabla
  - Son identificador para esa tabla
    - Identifica unívocamente a una tupla de la misma
- ▣ Condiciones
  - Dos tuplas no pueden tener los mismos valores para ellos
  - Los atributos que la componen no pueden tener valor nulo

## 4.2. Restricciones semánticas

- Unicidad (*unique*)
  - ▣ Los atributos con esta restricción no pueden repetirse
    - No puede haber dos tuplas con el mismo valor para ellos
  - ▣ Las claves candidatas tienen restricción de unicidad
  - ▣ La restricción de “clave primaria” incluye “unicidad”
    - Al marcar atributo como *primary key*, se marca como *unique*

## 4.2. Restricciones semánticas

- Obligatoriedad (*not null*)
  - ▣ Impide dejar vacío al atributo que la lleva
    - Tiene que tener valor obligatorio, del dominio del atributo
    - No admite valor nulo (*null*)
  - ▣ Las claves primarias son obligatorias (*not null*)

## 4.2. Restricciones semánticas

- Integridad referencial (*foreign key*)
  - ▣ La poseen las claves secundarias, migradas o foráneas
  - ▣ Los atributos que las componen han de ser
    - ... valores contenidos en la clave primaria de la tabla principal
    - ... ó bien nulo (null)

Asignaturas			
Cod_Asig	Denominación	Horas_sem	Cod_Ciclo
PR	Programación	8	DAW
BD	Bases de datos	6	DAW
RL	Redes locales	7	SMR

Ciclos		
Cod_Ciclo	Denominación	Nivel
DAW	Desarrollo de aplicaciones web	Superior
ASIR	Administración de sistemas informáticos en red	Superior
SMR	Sistemas microinformáticos y en red	Medio

## 4.2. Restricciones semánticas

### □ Integridad referencial (*foreign key*)

#### ▣ Problema:

- Operaciones de modificación y borrado en tabla principal
  - Si se elimina un registro cuya clave primaria está en una migrada
  - Si se modifica una clave primaria cuyo valor estaba en una migrada
- ¿Qué hacer con registros asociados en el resto de tablas?

#### ▣ Soluciones:

- Prohibir la operación (*no action*)
- Transmitir la operación en cascada (*cascade*)
  - Modificar la clave migrada, o eliminar la tupla en la tabla secundaria
- Colocar nulos (*set null*)
  - ... en las claves migradas asociadas a la tupla afectada
- Colocar el valor por defecto (*default*)
  - ... en las claves migradas asociadas a la tupla afectada

## 4.2. Restricciones semánticas

- Regla de validación (*check*)
  - ▣ Condiciones lógicas que han de darse para algún campo
  - ▣ Ejemplos
    - Atributo `edad` que sea positivo y menor de 150
    - Atributo `fecha_fin` que sea mayor que `fecha_inicio`
    - ...

## 4.2. Restricciones semánticas

- Disparadores (*triggers*)
  - ▣ Pequeños programas grabados en la base de datos
  - ▣ Se asocian a un evento
    - Cuando éste tiene lugar, se ejecutan automáticamente
      - Incorporación de una nueva tupla
      - Eliminación de una tupla existente
      - Conexión por parte de un usuario
      - ...
  - ▣ Mecanismo potente para implementar restricciones



## 5. Las 12 reglas de Codd

# 5. Las 12 reglas de Codd

- Han de ser cumplidas por los SGBDR (*RDBMS*)
  - ▣ Sistemas Gestores de Bases de Datos Relacionales
- De no cumplirse, el SGBD no será relacional
- En la práctica las cumplen pocos, considerados SGBDR
- Miden la fidelidad de un sistema relacional con el modelo relacional

# 5. Las 12 reglas de Codd

## 1. Información

- ▣ Toda la información se representa en el nivel lógico
  - Mediante el uso de tablas, y sólo de tablas
- ▣ Tanto datos como metadatos han de aparecer allí
  - Los metadatos se manipularán igual que los datos

## 2. Acceso garantizado

- ▣ Todo dato es accesible mediante
  - El nombre de la tabla que lo contiene
  - El valor de su clave primaria
  - El nombre de la columna que lo alberga

## 3. Tratamiento sistemático de los valores nulos

- ▣ El SGBD debe permitir el tratamiento de valores nulos

# 5. Las 12 reglas de Codd

4. Catálogo de línea basado en el modelo relacional
  - ▣ Metadatos se almacenan y acceden igual que los datos
  - ▣ Se usará mismo modelo relacional y lenguaje de consulta
5. Sublenguaje de datos completo
  - ▣ Al menos un lenguaje para manejar y tratar los datos
    - Definición de datos y vistas
    - Manipulación de datos
    - Gestión de restricciones
6. Actualización de vistas
  - ▣ SGBD debe encargarse de mostrar última actualización
  - ▣ No son válidas vistas obsoletas

# 5. Las 12 reglas de Codd

## 7. Inserción, modificación y eliminación de alto nivel

- ▣ El lenguaje debe permitir manipular datos en conjunto
  - No sólo para consultas
  - Inserción, modificación y eliminación no irá registro a registro

## 8. Independencia física

- ▣ El esquema lógico no varía si lo hace el esquema físico
- ▣ Al usuario le preocupan los datos, no cómo se almacenan

## 9. Independencia lógica

- ▣ Cambios en tablas no implican cambios en aplicaciones

# 5. Las 12 reglas de Codd

## 10. Independencia de integridad

- ▣ Las reglas de integridad deben guardarse en la BD
  - En el diccionario de datos
- ▣ No deben “guardarse” en los programas de aplicación

## 11. Independencia de la distribución

- ▣ Manejo de los datos igual para BD centralizada o distribuida
  - Instrucciones del sublenguaje de datos funcionarán igual

## 12. No subversión

- ▣ Si el sistema tiene lenguaje de bajo nivel (registro a registro)
  - ... No puede usarse para evitar reglas anteriores
    - Especialmente regla 7

## 6. Paso de MER a modelo relacional

1. Transformación de entidades fuertes
2. Transformación de relaciones
3. Transformación de entidades débiles
4. Relaciones ISA

## 6. Paso de MER a modelo relacional

### 2.2 REGLAS DE TRANSFORMACIÓN DE UN ESQUEMA E/R A UN ESQUEMA RELACIONAL

Las tres reglas básicas empleadas para transformar un esquema conceptual E/R en un esquema relacional son:

1. Toda entidad se transforma en una relación.
2. Las interrelaciones N:M se transforman en una relación.
3. Las interrelaciones 1:N dan lugar o bien a una propagación de clave o bien a una relación.



# 6.1. Entidades fuertes

## □ Transformaciones en una **entidad fuerte**

- Entidad → Tabla
- Atributos → Columnas o atributos
- Identificador principal → Clave primaria
- Identificadores candidatos → Claves candidatas
- Atributos multivaluados → Tabla nueva
  - Clave foránea: clave primaria de la relación que tiene el atributo
  - Atributo en sí
    - Si es simple, atributo sin más
    - Si es compuesto, todos los atributos simples que lo componen
  - Clave primaria
    - Clave primaria de la relación original
    - +
    - Atributo en sí

# 6.1. Entidades fuertes

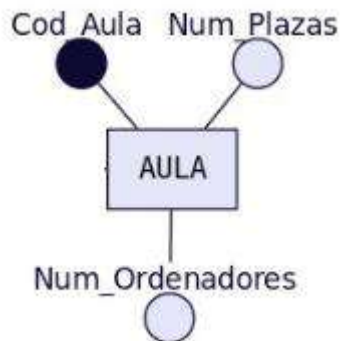
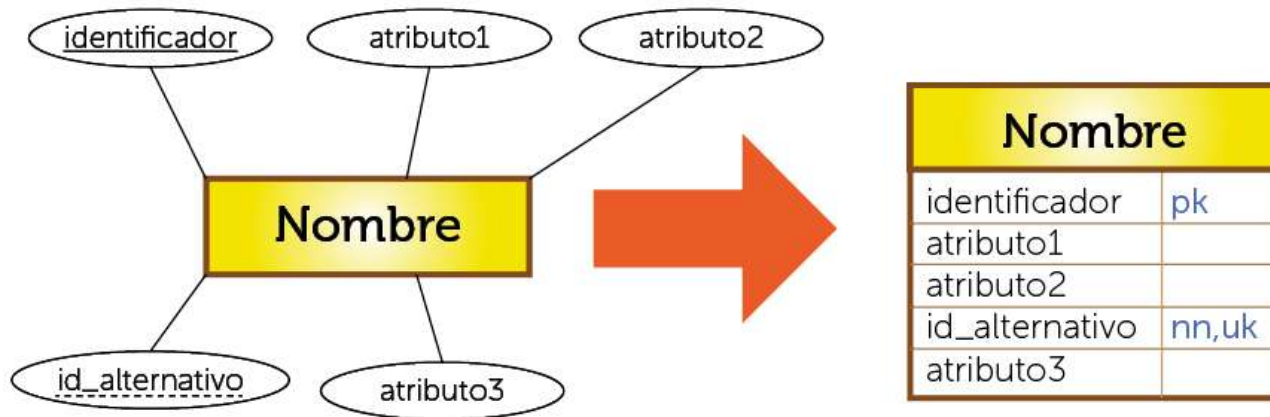
## □ Transformaciones en una **entidad fuerte**

Como norma base, 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.
- **Identificadores principales.** Pasan a ser claves primarias.
- **Identificadores alternativos.** Pasan a ser claves alternativas (tendrán restricciones **UNIQUE** y **NOT NULL**).

# 6.1. Entidades fuertes

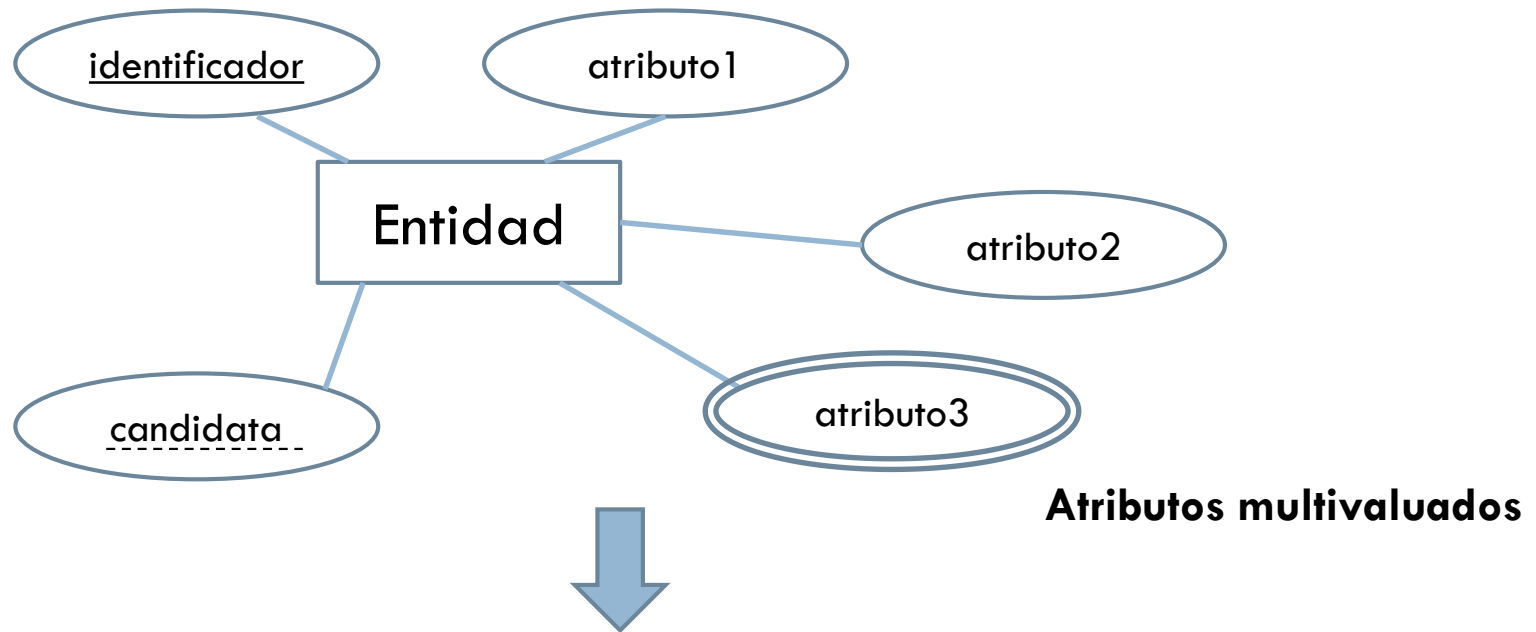
## □ Transformaciones en una **entidad fuerte**



**Cada entidad se transforma en una tabla.** El identificador (o identificadores) de la entidad pasa a ser la clave principal de la relación y aparece subrayada o con la indicación: *PK* (Primary Key). Si hay clave alternativa esta se pone en «*negrita*».

Aula (CodAula (PK), NumPlazas, NumOrdenadores)

# 6.1. Entidades fuertes



**Entidad(identificador, atributo1, atributo2, candidata)**  
**Atributo3(identificador, atributo3)**

# 6.2. Relaciones

Binarias varios a varios

## □ Relaciones binarias varios a varios

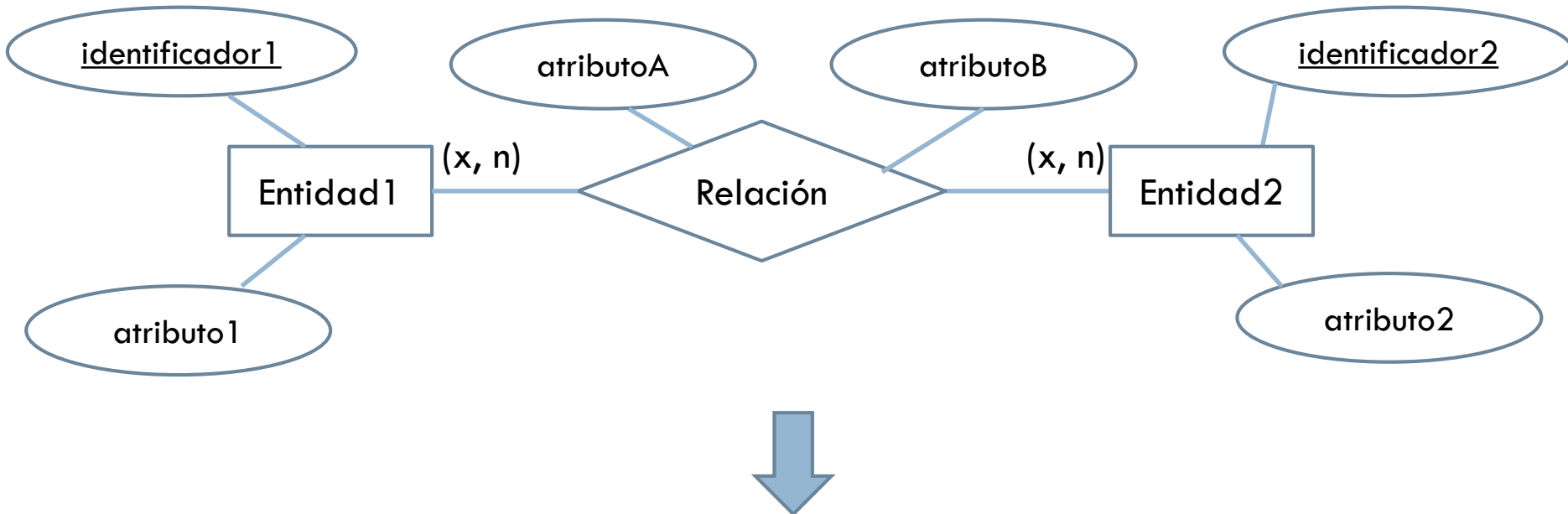
▣ Relación → Tabla

### ■ Atributos de la nueva tabla

- Clave primaria de las entidades relacionadas
  - Serán claves migradas, foráneas, secundarias, externas
  - Constituyen la clave primaria de la nueva tabla
- Atributos propios de la relación

## 6.2. Relaciones

Binarias varios a varios



Entidad1(identificador1, atributo1)

Entidad2(identificador2, atributo2)

Relacion(identificador1, identificador2, atributoA, atributoB)

## 6.2. Relaciones

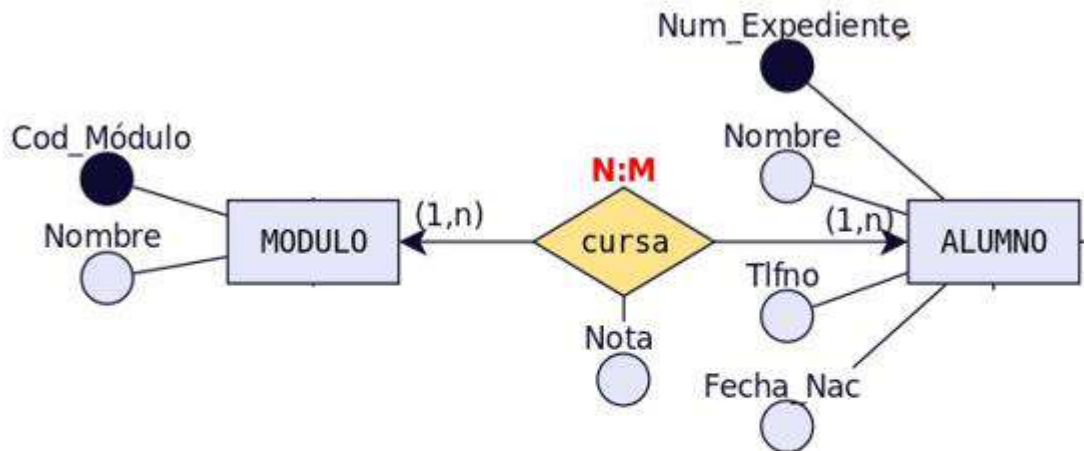
### Binarias varios a varios

Es el caso más sencillo. ***Siempre generan tabla***. Se crea una tabla que incorpora como *claves ajenas o foráneas FK (Foreign Key)* cada una de las claves de las entidades que participan en la relación. **La clave principal de esta nueva tabla está compuesta por dichos campos.** Importante: no se trata de 2 claves primarias, sino de una clave primaria compuesta por 2 campos.

Si hay atributos propios, pasan a la tabla de la relación.

Se haría exactamente igual si hubiera participaciones mínimas 0.

Orden de los atributos en las claves compuestas: Se deben poner a la izquierda todos los atributos que forman la clave.



```
Alumno (NumExpediente (PK), Nombre, Tlfno, FechaNac)
Alumno_Módulo (NumExpediente (FK), CodMódulo (FK)) (PK), Nota)
Módulo (CodMódulo (PK), Nombre)
```

## 6.2. Relaciones

n-arias

### □ Relaciones **n-arias**

(incluye ternarias, cuaternarias, n-arias)

▣ Relación → Tabla

#### ■ Atributos de la nueva tabla

- Clave primaria de las entidades relacionadas
  - Serán claves migradas, foráneas, secundarias, externas
  - Constituyen la clave primaria de la nueva tabla
- Atributos propios de la relación

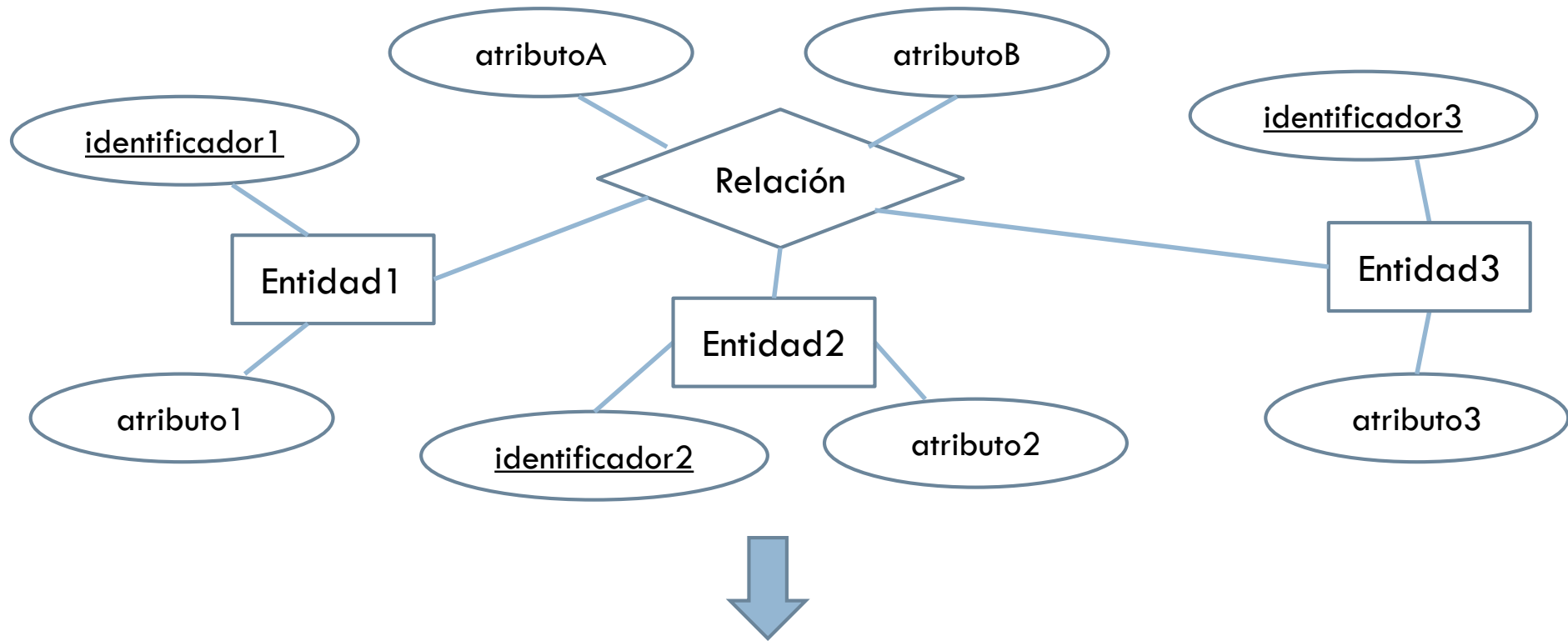
**Siempre generan tabla.** Las claves principales de las entidades que participan en la relación pasan a la nueva tabla como claves foráneas. Y **solo las de los lados «n» forman la principal.**

Si hay atributos propios de la relación, estos se incluyen en esa tabla.



## 6.2. Relaciones

n-arias



Entidad1(identificador1, atributo1)

Entidad2(identificador2, atributo2)

Entidad3(identificador3, atributo3)

Relacion(identificador1, identificador2, identificador3, atributoA, atributoB)

## 6.2. Relaciones

Binarias uno a varios

### □ Relaciones binarias uno a varios

Por lo general, **NO** generan tabla:

En la transformación de relaciones 1:N existen dos soluciones:

- **Transformar la relación en una tabla.** Se hace como si se tratara de una relación N:M. Esta solución se realiza cuando se prevé que en un futuro la relación se convertirá en N:M y cuando la relación tiene atributos propios. También se crea una nueva tabla cuando la cardinalidad es opcional, es decir, (0,1) y (0,M). La clave de esta tabla es la de la entidad del lado *muchos*.
- **Propagar la clave.** Este caso se aplica cuando la cardinalidad es obligatoria, es decir, cuando tenemos cardinalidad (1,1) y (0,M) o (1,M). Se propaga el atributo principal de la entidad que tiene de cardinalidad máxima 1 a la que tiene de cardinalidad máxima N, desapareciendo el nombre de la relación. Si existen atributos propios en la relación, éstos también se propagarán.

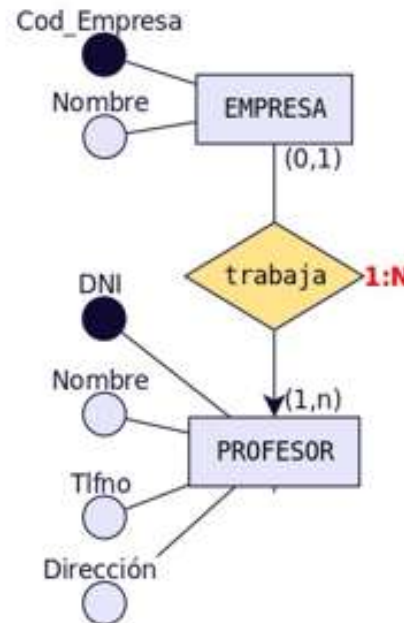
## 6.2. Relaciones

### Binarias uno a varios

Caso 1: Si la entidad del lado «1» presenta participación (0,1), entonces **se crea una nueva tabla** para la relación que incorpora como **claves ajenas** las claves de ambas entidades. La **clave principal** de la relación será sólo la clave de la entidad del lado «N».

No habitual

Según algunos autores,  
**NO generan tabla**



*Nota:* En el paso a tablas de la relación 1:N entre PROFESOR (1,n) y EMPRESA (0,1). Como en el lado «1» encontramos participación mínima 0, se generará una nueva tabla.

<https://jorgesanchez.net/manuales/gbd/disenio-logico-relacional.html>

Profesor(DNI (PK), Nombre, Tlfno, Dirección)

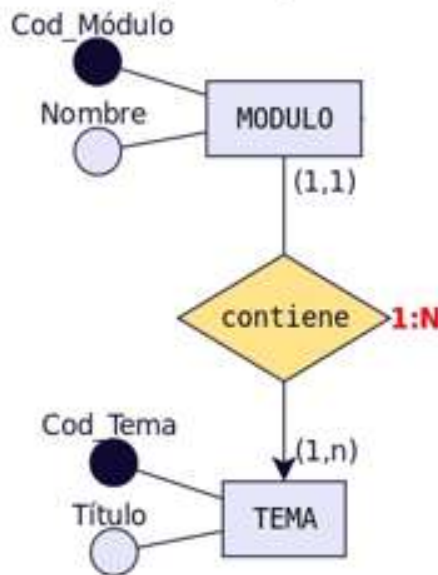
Profesor Empresa(DNI (FK1) (PK), CodEmpresa (FK2), Nota)

Empresa(CodEmpresa (PK), Nombre)

## 6.2. Relaciones

### Binarias uno a varios

Caso 2: Para el resto de situaciones, la **entidad del lado «N» recibe como clave ajena la clave de la entidad del lado «1»**. Los atributos propios de la relación pasan a la tabla donde se ha incorporado la clave ajena.



*Nota: En el paso a tablas de la relación 1:N entre MÓDULO (1,1) y TEMA (1,n). Como no hay participación mínima «0» en el lado 1, no genera tabla y la clave principal del lado «1» pasa como foránea al lado «n».*

```
Tema(CodTema(PK), Título, CodModulo(FK))
```

```
Módulo(CodMódulo(PK), Nombre)
```

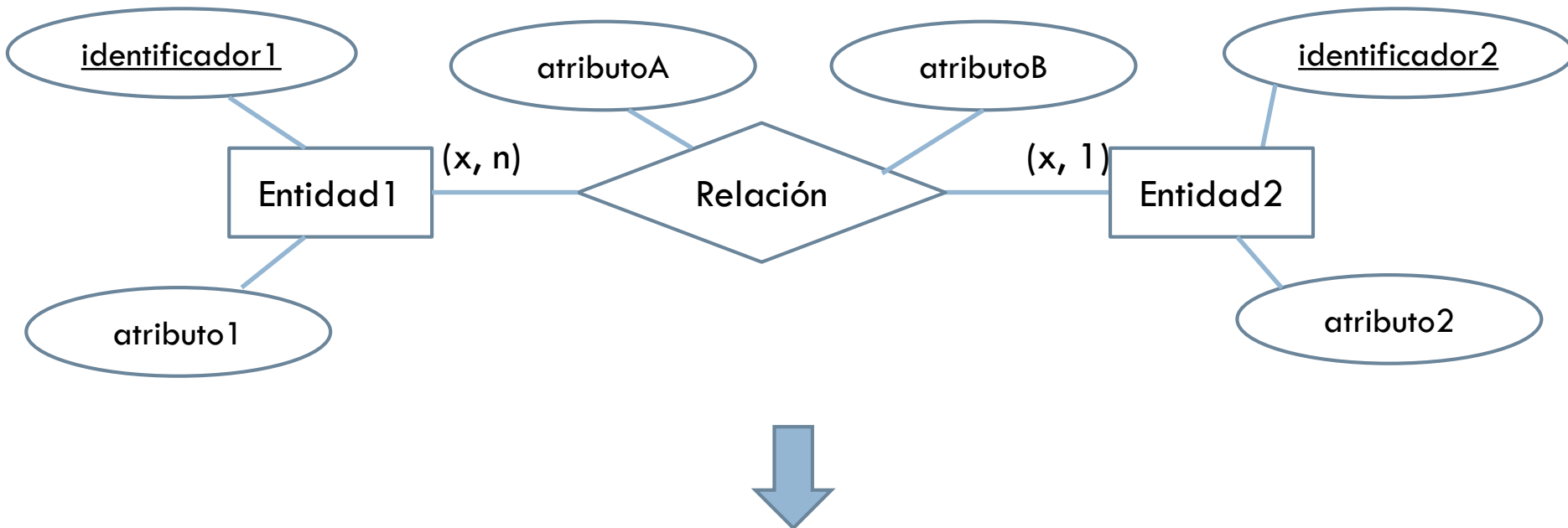
## 6.2. Relaciones

### Binarias uno a varios

- Relaciones binarias uno a varios (2° caso)
  - ▣ No requieren tabla para la relación
  - ▣ Se migra la clave del lado 1 al lado N
    - En la tabla de la entidad del lado N de la relación...
      - Nueva clave externa: clave primaria de la entidad del lado 1
  - ▣ Se incluyen en lado N los atributos propios de la relación
  - ▣ Si la cardinalidad mínima del lado de 1 es 0...
    - Se debe permitir nulo en el valor de la clave secundaria
  - ▣ Si la cardinalidad mínima del lado 1 es 1...
    - No se debe permitir nulo en el valor de la clave secundaria

## 6.2. Relaciones

Binarias uno a varios



Entidad1(identificador1, atributo1, identificador2, atributoA, atributoB)

Entidad2(identificador2, atributo2)

## 6.2. Relaciones

### Binarias uno a uno

#### □ Relaciones binarias uno a uno

En la transformación de relaciones 1:1 se tienen en cuenta las cardinalidades de las entidades que participan. 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 a la tabla resultante de la otra. En este caso, también se pueden añadir los atributos de una entidad a otra, resultando una única tabla con todos los atributos de las entidades y de la relación, si los hubiera, eligiendo como clave primaria una de las dos.

## 6.2. Relaciones

Binarias uno a uno

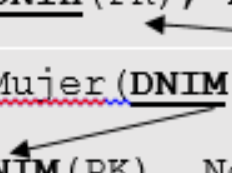
Por lo general, **no generan tabla**, pero se dan 2 casos:

**Genera tabla**, pero prácticamente  
NO se da este 1er. caso

Caso 1: Si las dos entidades participan con participación (0,1), entonces **se crea una nueva tabla** para la relación.

*Nota: Una situación donde puede darse este caso es en HOMBRE (0,1) se casa con MUJER (0,1). Es similar al caso 1 del apartado anterior en relaciones 1:N, aunque en este caso debemos establecer una restricción de valor único para FK2.*

```
Hombre(DNIM(PK), Nombre, Tlfno, Dirección)
Hombre_Mujer(DNIM(FK1) (PK), DNIM(FK2) (UQ))
Mujer(DNIM(PK), Nombre, Tlfno, Dirección)
```



**No habitual**

Según algunos autores,

**NO generan tabla**

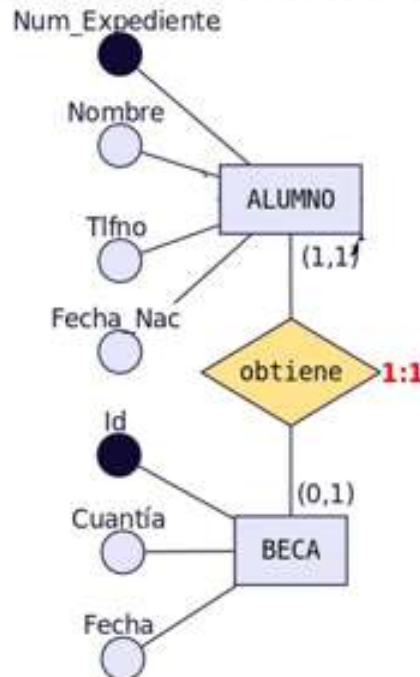
<https://jorgesanchez.net/manuales/gbd/disenio-logico-relacional.html>



## 6.2. Relaciones

### Binarias uno a uno

Caso 2: Si alguna entidad, pero no las dos, participa con participación mínima 0 (0,1), entonces se pone la **clave ajena en dicha entidad**, para evitar en lo posible, los valores nulos.



*Nota: En el paso a tablas de la relación 1:1 entre ALUMNO (1,1) y BECA (0,1). Como BECA tiene participación mínima 0, incorporamos en ella, como clave foránea, la clave de ALUMNO. Esta forma de proceder también es válida para el caso 3, pudiendo acoger la clave foránea cualquiera de las entidades.*

Beca(ID(PK), Cuantía, Fecha, NumExpediente(FK))

Alumno(NumExpediente(PK), Nombre, Tlfno, FechaNac)

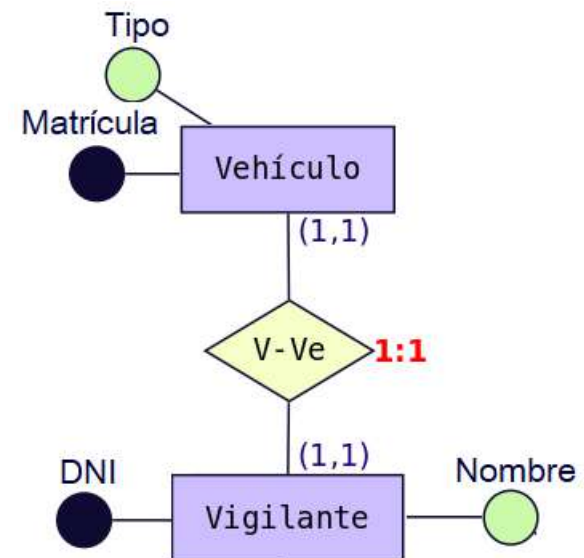
## 6.2. Relaciones

### Binarias uno a uno

Caso 2 (bis): Si tenemos una relación 1:1 y ninguna tiene participación mínima 0, elegimos la **clave principal de UNA DE ELLAS y la introducimos como clave ajena en la otra tabla**. Se elegirá una u otra forma en función de cómo se quiera organizar la información para facilitar las consultas. Los atributos propios de la relación pasan a la tabla donde se introduce la clave ajena.

```
Vehículo (Matrícula (PK), Tipo, DNI (FK))  
Vigilante (DNI (PK), Nombre)
```

**Hay otras 2 posibles soluciones,  
válidas también  
(ver diapositiva 68)**



## 6.2. Relaciones

Binarias uno a uno

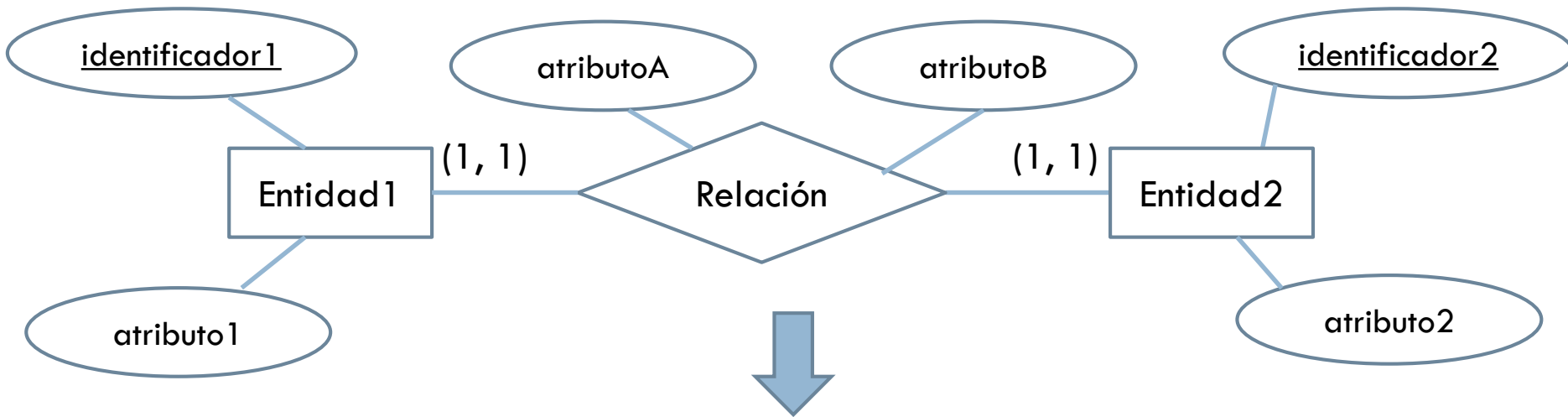
### □ Relaciones binarias uno a uno

#### ▣ Dos alternativas de transformación

1. Migrar clave de una entidad a otra
  - Se puede elegir el sentido, es indiferente
  - La nueva clave externa será también clave alternativa
2. Generar una sola tabla con los atributos de las 2 entidades
  - Clave principal: cualquiera de las claves de las entidades
  - Clave alternativa: la clave de la otra entidad (no seleccionada)
  - Nombre de tabla: el de la entidad “más importante” de las dos

## 6.2. Relaciones

Binarias uno a uno



Solución 1    Entidad1 (identificador1, atributo1, identificador2, atributoA, atributoB)  
Entidad2(identificador2, atributo2)

Solución 2    Entidad1 (identificador1, atributo1)  
Entidad2(identificador2, atributo2, identificador1, atributoA, atributoB)

Solución 3    Entidad1 (identificador1, atributo1, identificador2, atributo2, atributoA, atributoB)

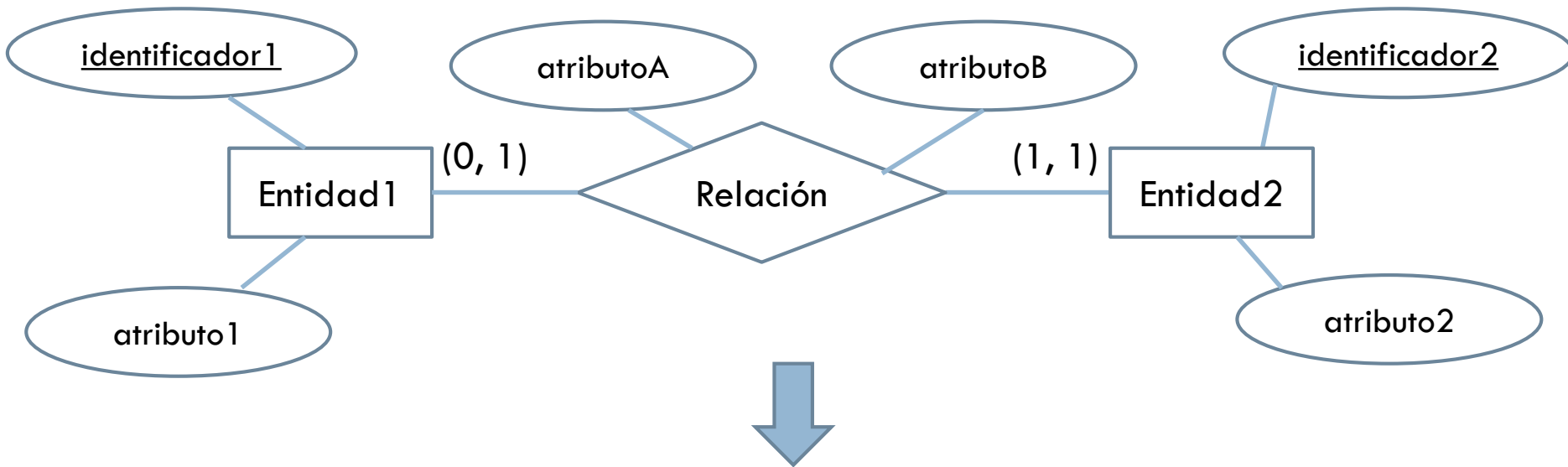
## 6.2. Relaciones

### Binarias uno a uno

- Relaciones binarias cero a uno
  - Cardinalidad máxima en ambas = 1
  - Cardinalidad mínima en UNA de las entidades = 0
- ▣ Parecido al caso anterior, pero:
  - NO se escoge como solución crear una sola entidad
    - Dicha entidad tendría muchos campos nulos
  - NO se escoge como solución 2 tablas y migrar del lado 0 al 1
    - La clave migrada sería nula cuando no hubiese relación
  - Sí se escoge como solución 2 tablas y migrar del lado 1 al 0
    - En la tabla con cardinalidad mínima 0
      - Clave externa = Clave primaria que tiene la tabla del lado 1

## 6.2. Relaciones

Binarias uno a uno



Entidad1 (identificador1, atributo1, identificador2, atributoA, atributoB)

Entidad2(identificador2, atributo2)

## 6.2. Relaciones

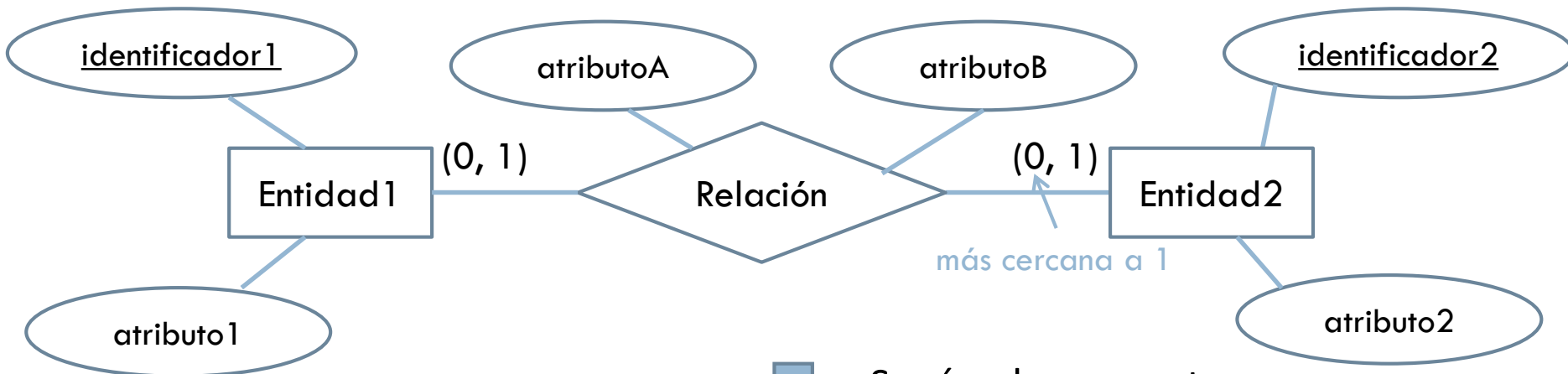
Binarias uno a uno

- Relaciones binarias cero a cero
  - Cardinalidad máxima en ambas = 1
  - Cardinalidad mínima en AMBAS entidades = 0
- ▣ Parecido a los casos anteriores
- ▣ Como ninguno tiene cardinalidad mínima 1...
  - Migra clave primaria de entidad que más próxima esté a ello
    - La que “más veces intervenga en la relación”
    - La que “menos opcional sea”
  - La otra entidad recibe esa clave como clave externa
    - NO será clave alternativa
    - Sí tendrá restricción de unicidad

## 6.2. Relaciones

Binarias uno a uno

<https://jorgesanchez.net/manuales/gbd/disenio-logico-relacional.html>



Según algunos autores,

**NO generan tabla**

Entidad1 (identificador1, atributo1, identificador2\*, atributoA, atributoB)

Entidad2 (identificador2, atributo2)

*La razón de este movimiento es cuanto más cerca esté del valor “uno” la cardinalidad mínima, menos huecos vacíos dejaremos en las tablas (base de datos, por lo tanto, más eficiente).*

*En la clave secundaria no se indica ninguna restricción NOT NULL, ya que no todos los datos se relacionan.*

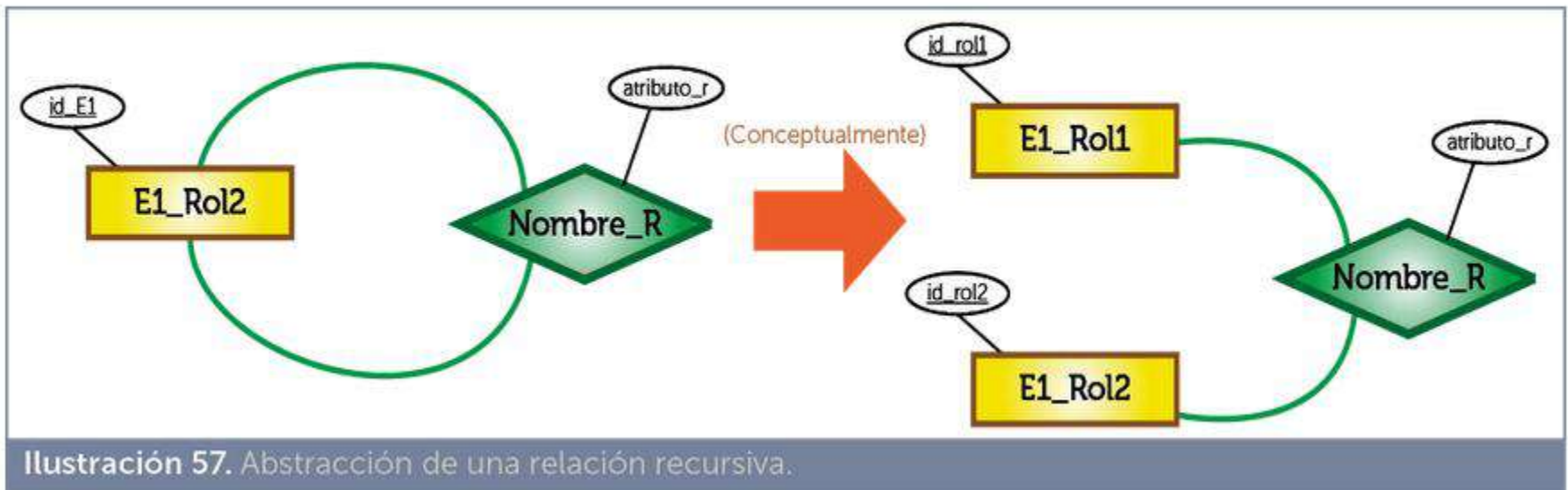


## 6.2. Relaciones

Reflexivas

### □ Relaciones reflexivas o recursivas

Es fácil confundirse en este tipo de relaciones por lo que hay que imaginarse esta situación:



## 6.2. Relaciones

### Reflexivas

#### □ Relaciones reflexivas o recursivas

Son relaciones binarias en las que participa un tipo de entidad. En el proceso de 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 (No se crea la segunda tabla)**
  - 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 la **relación 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, más los **atributos de la relación si los hubiera**. La clave de esta nueva tabla será la **combinación de las dos**.

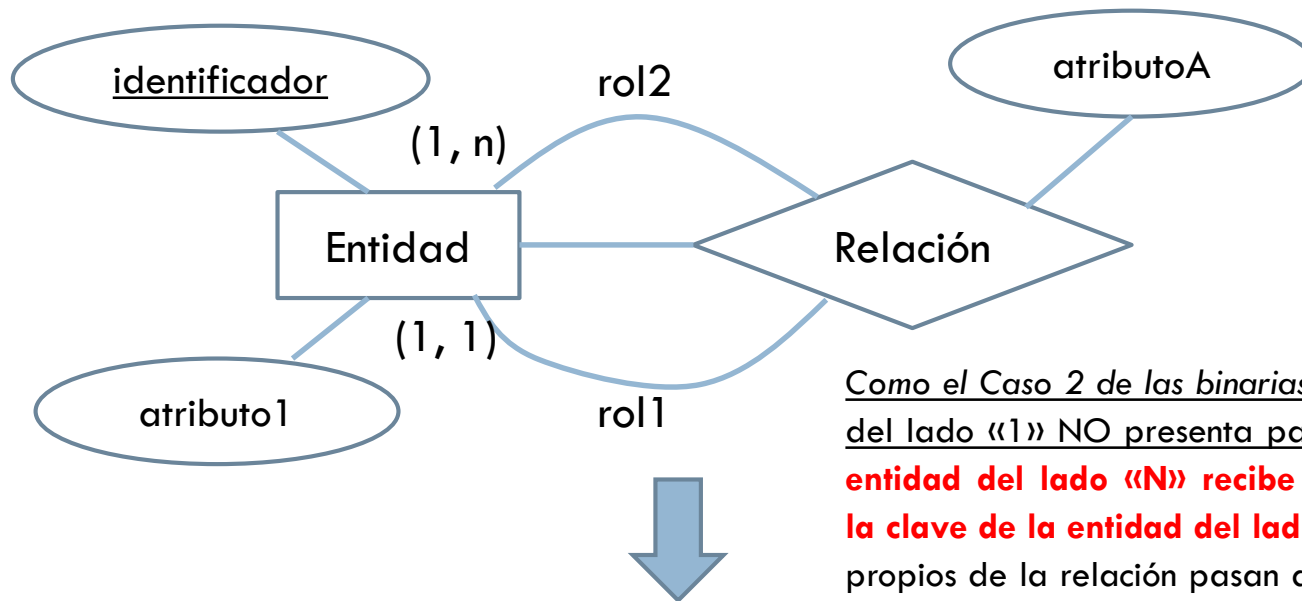
## 6.2. Relaciones

### Reflexivas

#### □ Relaciones **reflexivas o recursivas**

##### ▣ Mismo tratamiento que las binarias

- En vez de dos entidades se manejan dos roles



Como el Caso 2 de las binarias 1:N: Si la entidad del lado «1» NO presenta participación (0,1), la entidad del lado «N» recibe como clave ajena la clave de la entidad del lado «1». Los atributos propios de la relación pasan a la tabla donde se ha incorporado la clave ajena.

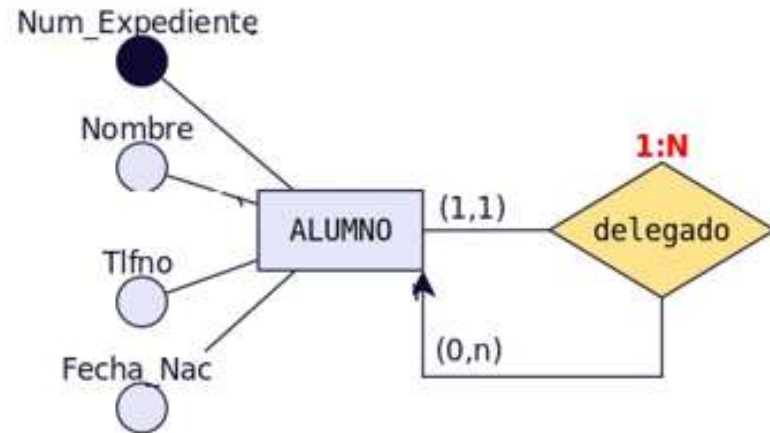
Entidad(identificador rol2, atributo1, identificador rol1, atributoA)

## 6.2. Relaciones

### Reflexivas

#### □ Relaciones reflexivas o recursivas

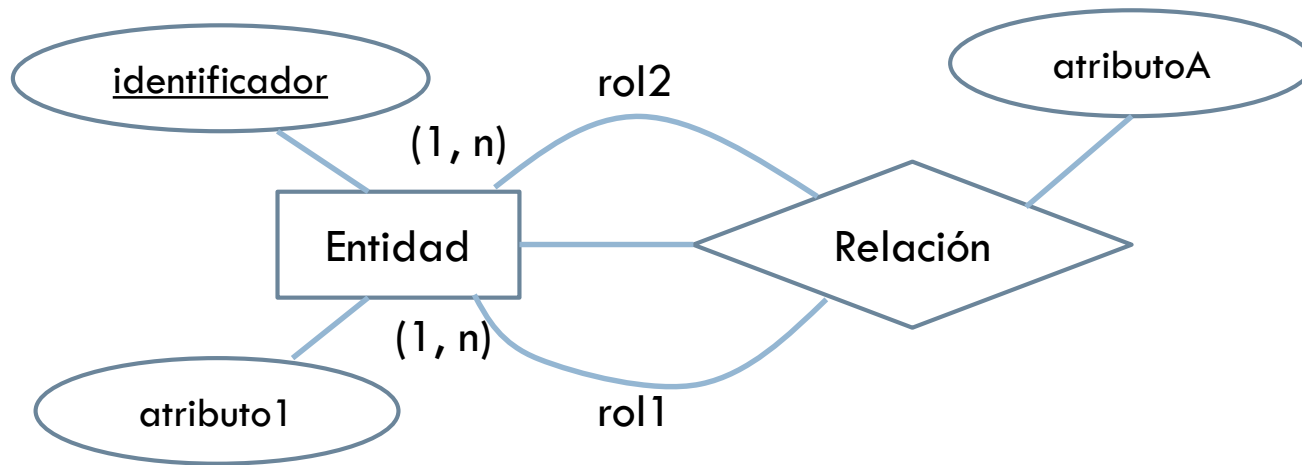
En el paso a tablas de la relación reflexiva de ALUMNO, como **no tiene participación mínima «0» en el lado 1, no genera tabla**. La **clave principal** de ALUMNOS, volverá a aparecer en ALUMNOS como clave foránea, igual que en cualquier relación 1:N. Ahora bien, como no puede haber dos campos con el mismo nombre en la misma tabla, deberemos cambiar un poco el nombre de la clave principal, para que haga referencia al papel que ocupa como clave foránea.



Alumno (NumExpediente (PK), Nombre, Tlfno, FechaNac, NumExpedDeleg (FK))

## 6.2. Relaciones

Reflexivas

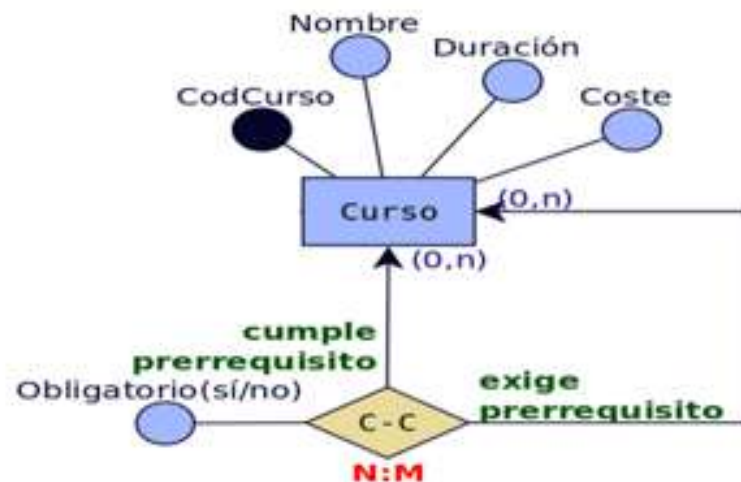


Entidad(identificador, atributo1)

Relación(identificador rol1, identificador rol 2, atributoA)

## 6.2. Relaciones

Reflexivas



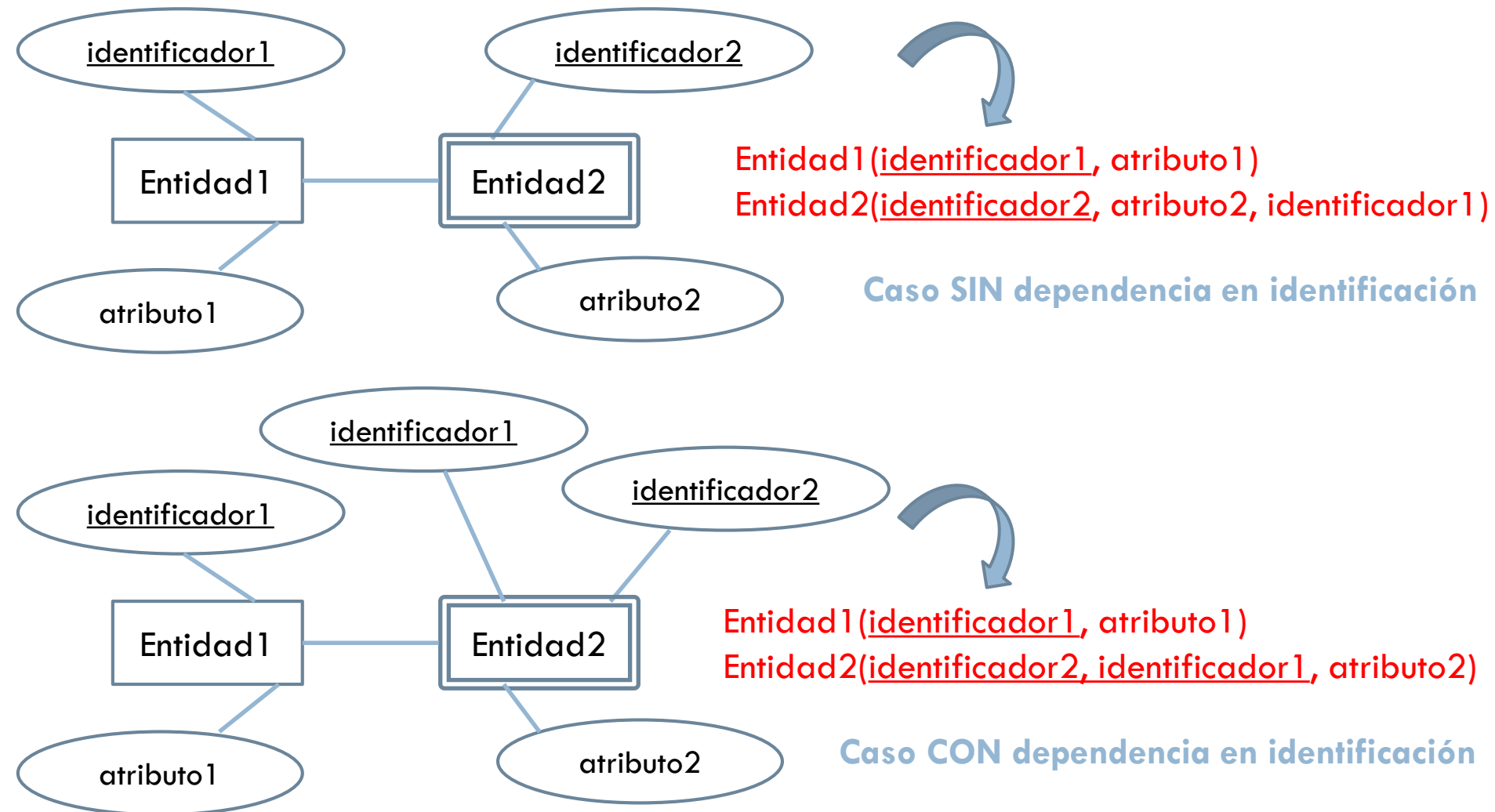
Prerrequisito (CurSolicitado (FK), CurPrevio (FK) (PK), Obligatorio)

Curso (CodCurso (PK), Nombre, Duración, Coste)

## 6.3. Entidades débiles

- En el MER no aparece relación como tal (es implícita)
  - ▣ Relación fuerte - Relación débil (1:N)
- Se trata como otra relación 1:N
  - ▣ Se migra clave primaria de la fuerte a la débil
    - Del lado 1 (entidad fuerte) al lado N (entidad débil)
    - Pasa a ser clave externa en la débil
- ¡ Cuidado ! Si hay dependencia en identificación...
  - ▣ El identificador de la débil contiene el de la fuerte
  - ▣ Eso ya estará representado en el diagrama E/R
  - ▣ No hay que “volver a migrar” de la fuerte a la débil

## 6.3. Entidades débiles

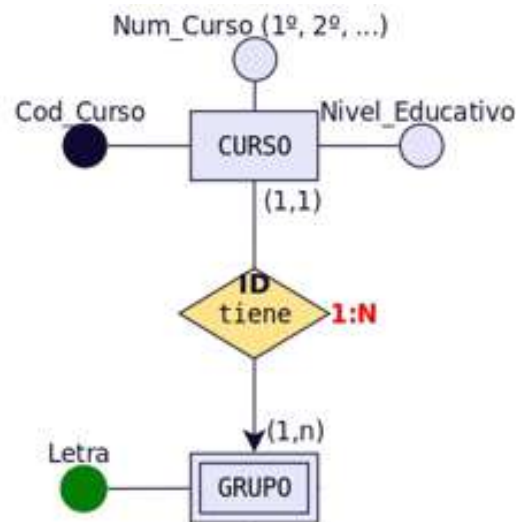




## 6.3. Entidades débiles

### Caso CON dependencia en identificación

Por lo general no generan tablas, porque suelen ser 1:1 o 1:N. Como en toda relación 1:N, la clave de la entidad fuerte debe introducirse en la tabla de la entidad débil como foránea y, además en este caso, formar parte de la clave de ésta. En las entidades débiles, la clave de la entidad fuerte debe ir la primera y, a continuación, los discriminadores de la débil.



*Paso a tablas de la relación débil en identificación entre CURSO Y GRUPO.*

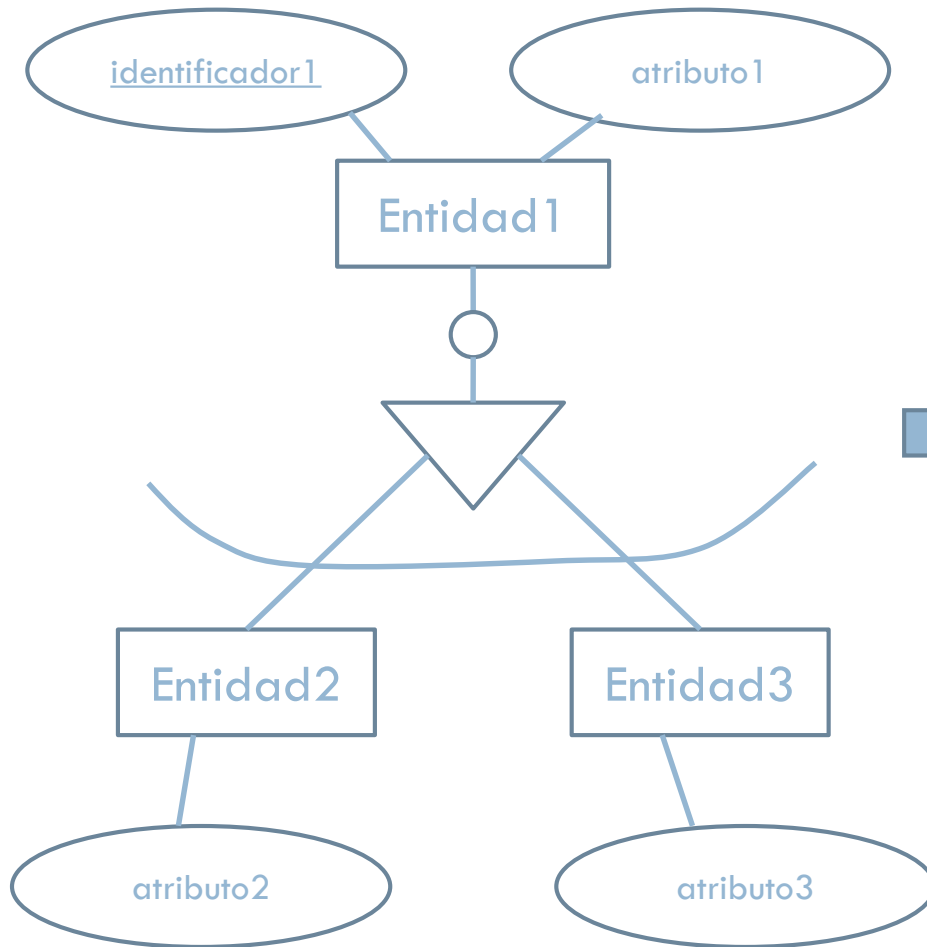
Grupo (CodCurso (FK), Letra (PK))

↓  
Curso (CodCurso (PK), NivelEducativo, NumCurso)

## 6.4. Relaciones ISA

- Superentidad → Tabla nueva en el modelo
  - ▣ Si la ISA es TOTAL, se puede omitir
    - Se pasarían sus atributos a TODAS las subentidades
    - No recomendable, por complicar el esquema interno
- Subentidades → Una tabla para cada una
- Atributos → Campos de la tabla correspondiente
- Si las subentidades NO tienen clave propia
  - ▣ Clave de la superentidad → Clave primaria de subentidades
    - También será clave secundaria
- Si las subentidades SÍ tienen clave propia
  - ▣ Clave de la superentidad → Clave secundaria de subentidades
    - También será clave alternativa

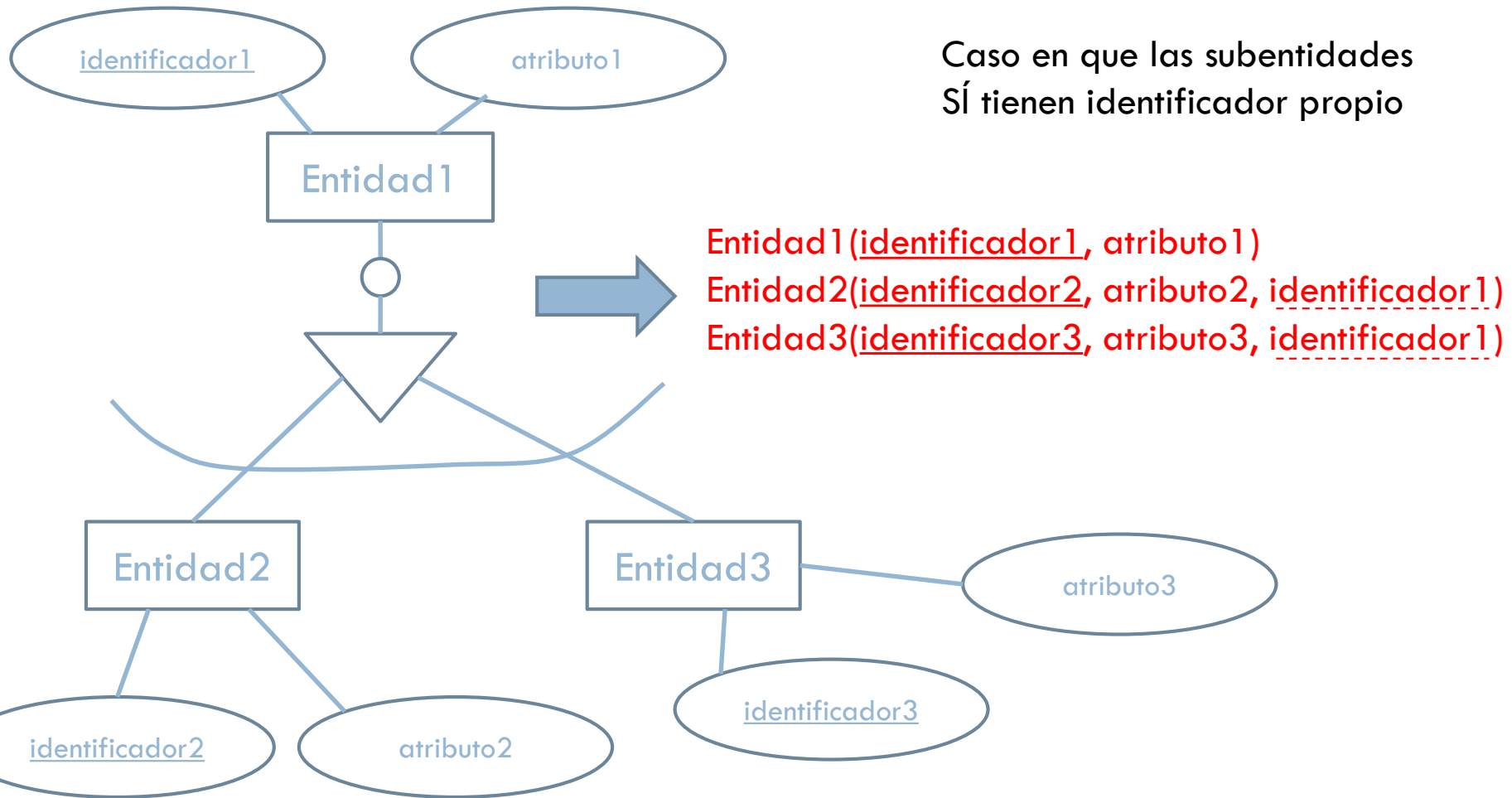
## 6.4. Relaciones ISA



Caso en que las subentidades  
NO tienen identificador propio

Entidad1 (identificador1, atributo1)  
Entidad2(identificador1, atributo2)  
Entidad3(identificador1, atributo3)

## 6.4. Relaciones ISA



## 6.4. Relaciones ISA

### □ Influencia del tipo de relación ISA

#### ▣ Exclusiva o Solapada

- No cambia el esquema relacional
- Ha de tenerse en cuenta para restricciones en esquema interno
  - Exclusivas: clave superentidad no puede repetirse en subentidades
  - Uso de disparadores o *triggers* para conseguir esas restricciones

#### ▣ Total o Parcial

- No cambia el esquema relacional
- Ha de tenerse en cuenta para posibles restricciones futuras
- Suele añadirse esa información como nota junto al esquema



## 7. Esquema relacional

# 7. Esquema relacional

- Modelo lógico de datos
  - ▣ Generado a partir del modelo conceptual
    - Esquema o diagrama Entidad/Relación
  - ▣ NO sustituye a ese modelo conceptual (diagrama E/R)
    - Hay información que no se refleja en el esquema relacional
      - Obligatoriedad en una relación (cardinalidad mínima 0 ó 1)
      - Exclusividad en una relación ISA
      - Totalidad en una relación ISA
      - ...
    - Esa información es importante para crear índices y restricciones
- Verdadero “mapa” de la BD = diagrama E/R
  - ▣ Aunque el esquema relacional es más próximo al físico

# 7. Esquema relacional

- Representación de esquemas de BDs relacionales
  - ▣ Modo formal (clásico)
  - ▣ Grafo relacional
  - ▣ Esquemas derivados del modelo entidad/relación



# 7. Esquema relacional

## □ Modo formal (clásico)

Entidad(clave, clave\_alternativa, atributo1, ..., atributoN)

## □ Problema

- No se aprecian bien las relaciones entre entidades

## □ Solución

- Completar el esquema con líneas y diagramas
  - Permiten apreciar esas relaciones
  - Más cercanos al modelo entidad/relacional del que proceden

# 7. Esquema relacional

## □ Modo formal (clásico)

Ciclos(cod ciclo, nombre, n\_cursos, plan)

Asignaturas(cod asig, nombre, cod\_ciclo, horas\_sem)

Matrículas(cod asig, cod alum, año, nota)

Alumnos(cod alum, nif, nombre, apellido)

# 7. Esquema relacional

- Grafo relacional
  - ▣ Completa el modo clásico mostrando las relaciones
  - ▣ Líneas que van desde clave primaria a clave secundaria
  - ▣ Conforman un grafo
    - Nodos: atributos de las diferentes entidades
    - Arcos: migración de claves primarias a otras tablas
      - Marcan las relaciones

# 7. Esquema relacional

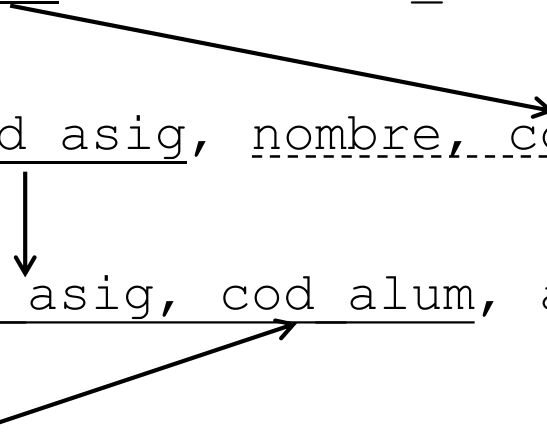
## □ Grafo relacional

Ciclos(cod ciclo, nombre, n\_cursos, plan)

Asignaturas(cod asig, nombre, cod\_ciclo, horas\_sem)

Matrículas(cod asig, cod alum, año, nota)

Alumnos(cod alum, nif, nombre, apellido)



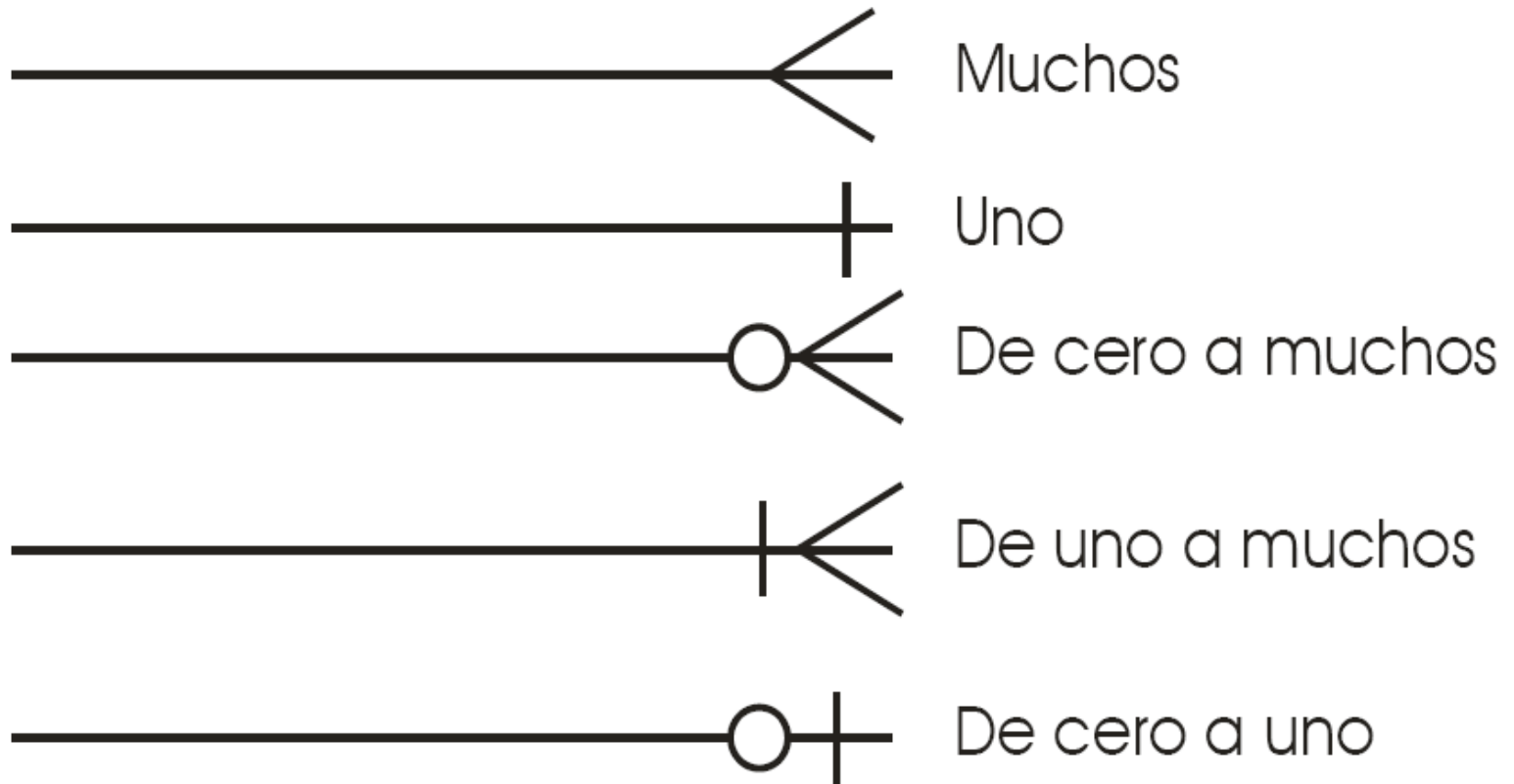
# 7. Esquema relacional

- Esquemas derivados del modelo E/R
  - ▣ Llamados también esquemas E/R relacionales
    - Variantes del modelo E/R
    - No suelen ser capaces de representar tanto como el E/R
  - ▣ Modelo de patas de gallo
    - Notación más popular en las herramientas CASE
      - Ingeniería del Software Asistida por Computadora
    - Cardinalidades representadas con símbolos
    - Mezcla modelo entidad/relación y relacional
      - Se usa incluso para crear diseños conceptuales (E/R)

# 7. Esquema relacional

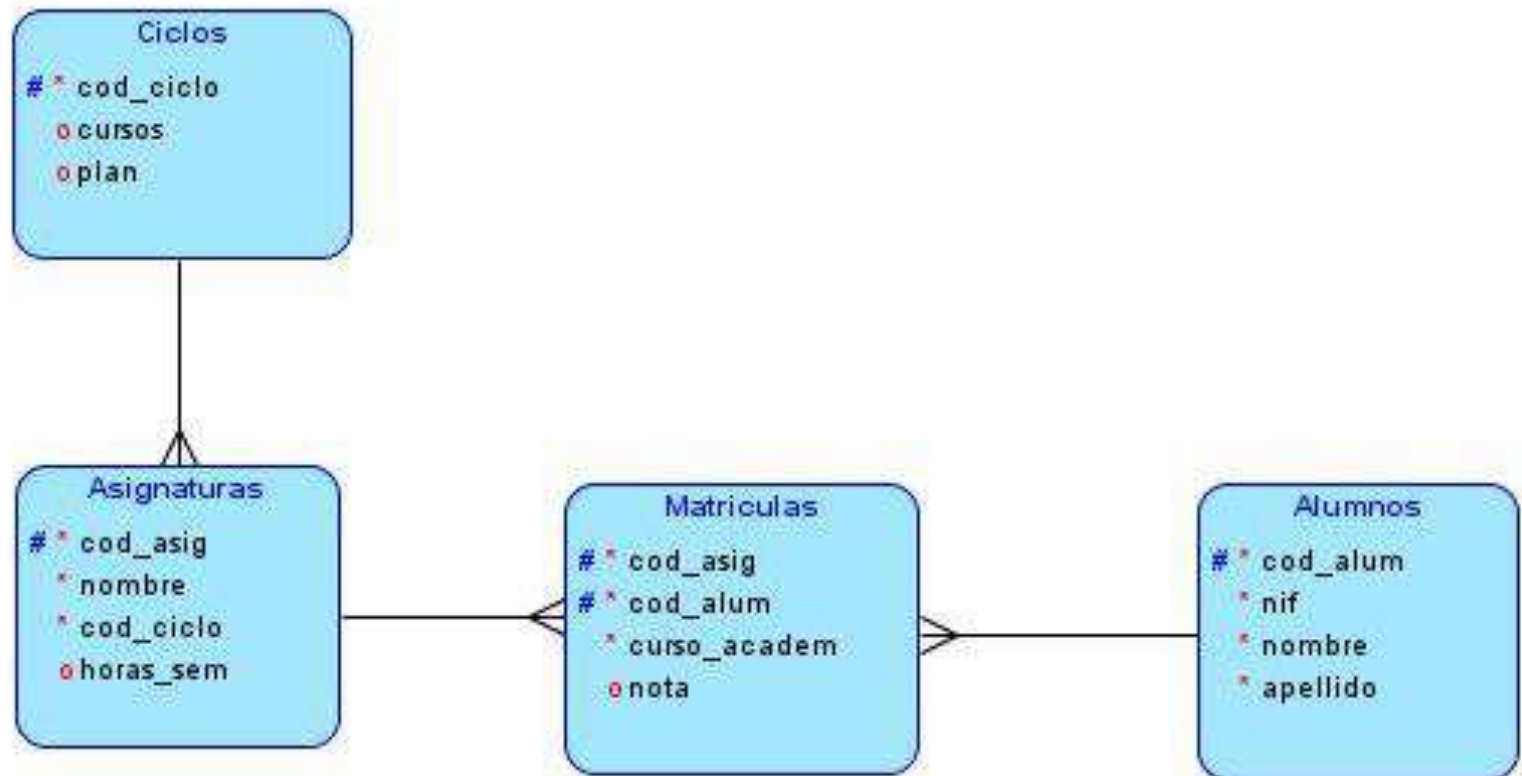
## □ Esquemas derivados del modelo E/R

### ▣ Modelo de patas de gallo



# 7. Esquema relacional

- Esquemas derivados del modelo E/R
  - ▣ Modelo de patas de gallo

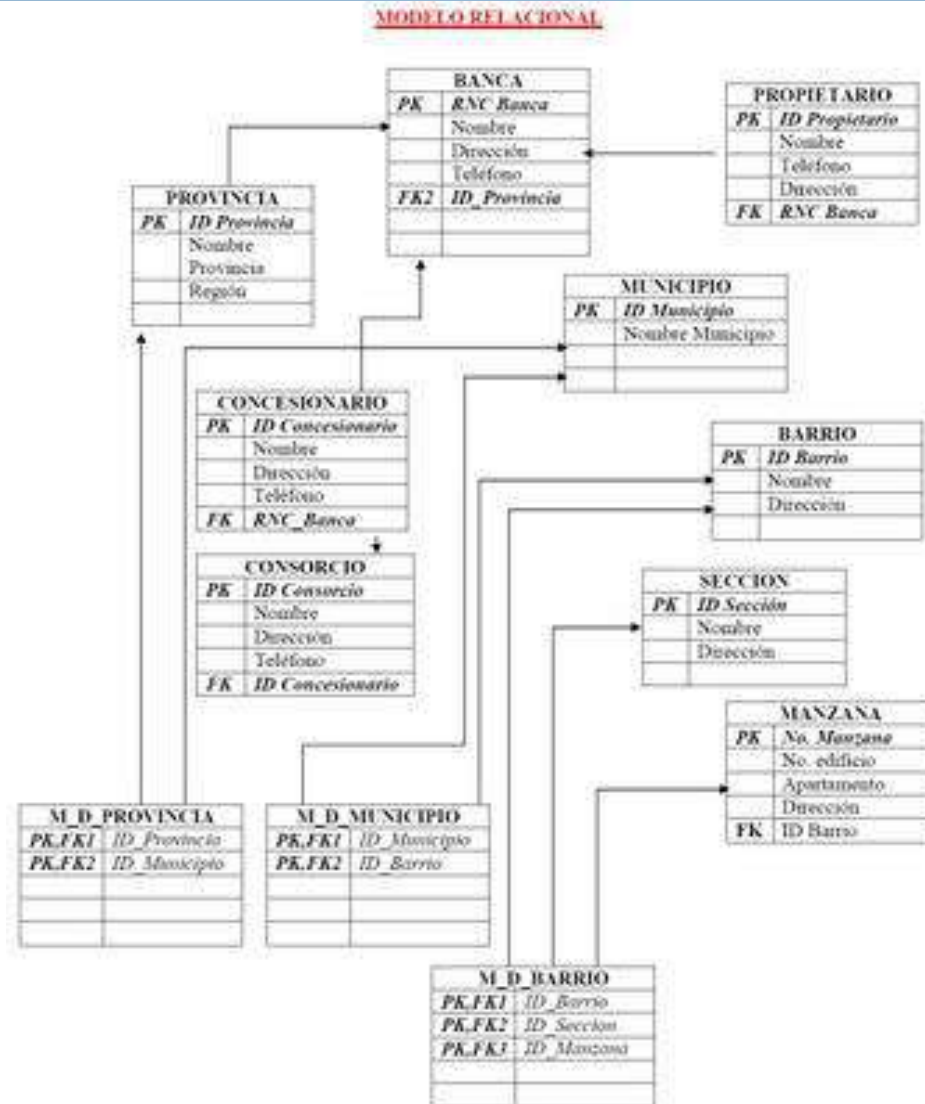


# 7. Esquema relacional

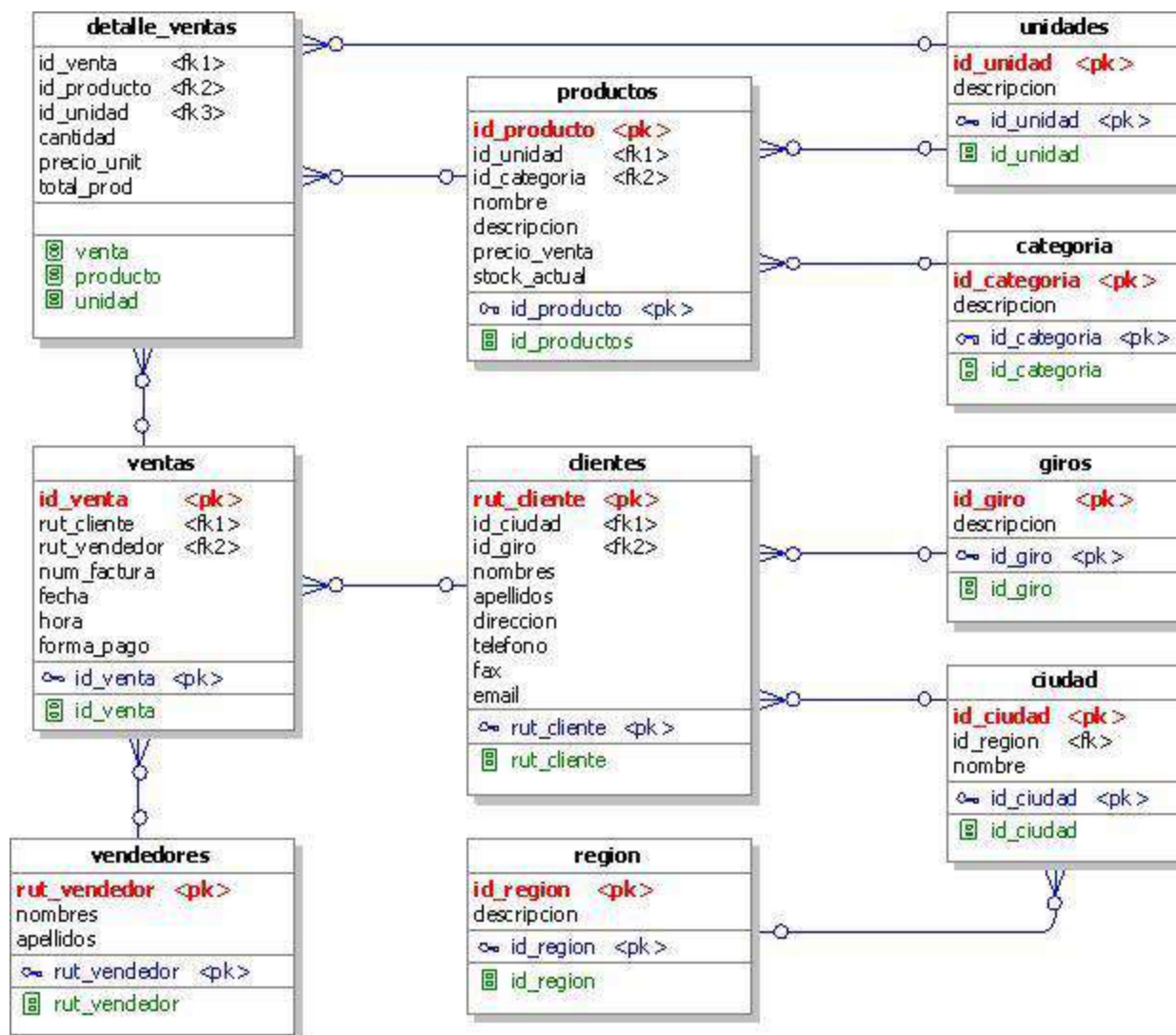
- Esquemas derivados del modelo E/R
  - ▣ A veces los esquemas muestran información muy completa
    - Cardinalidades
    - Opcionalidad
    - Tipo de datos de los atributos
    - Restricciones
      - Clave primaria
      - Clave migrada
      - Obligatoriedad (no nulo)
      - Unicidad
      - ...
  - ▣ Las herramientas CASE facilitan mucho la tarea

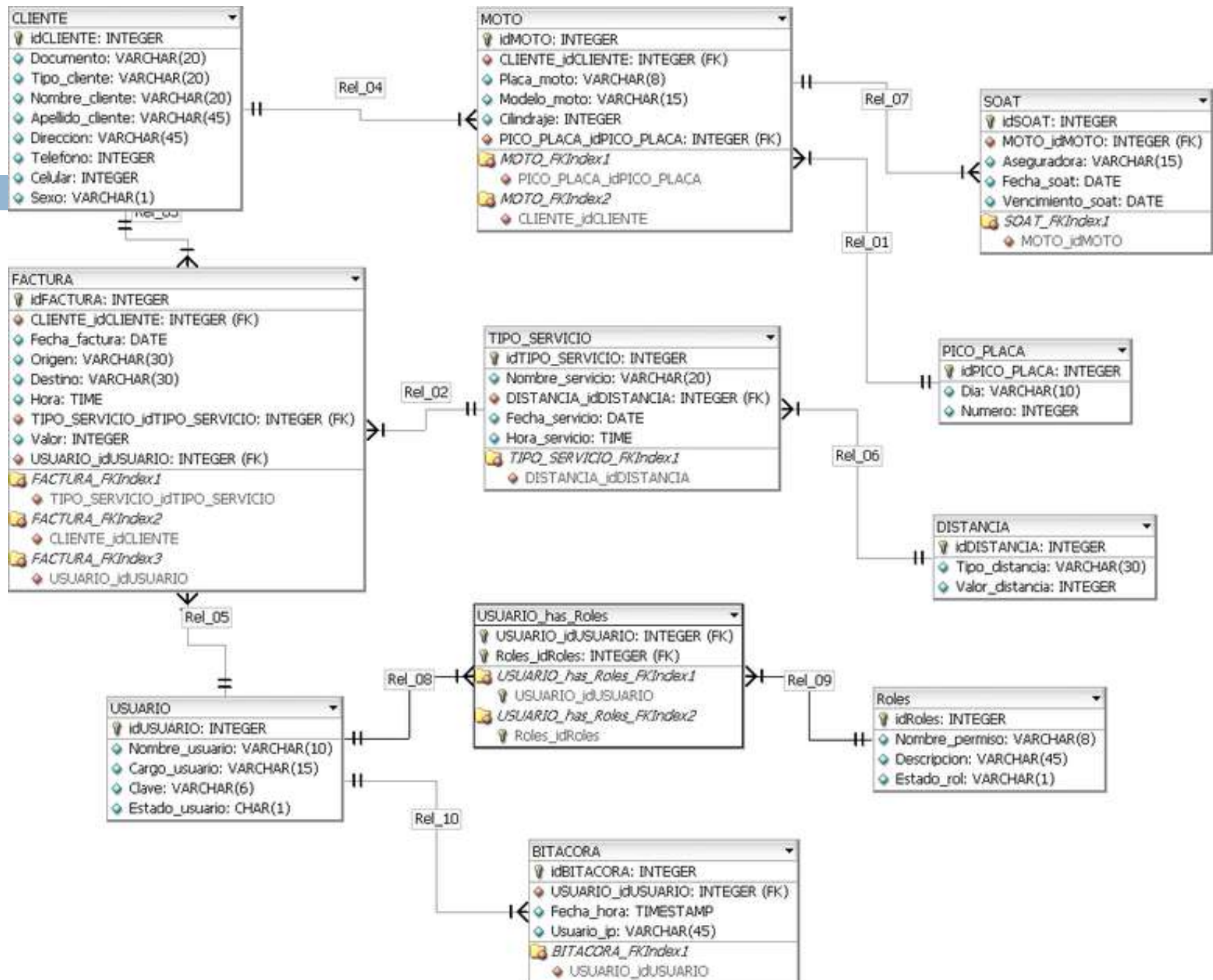


# 7. Esquema relacional

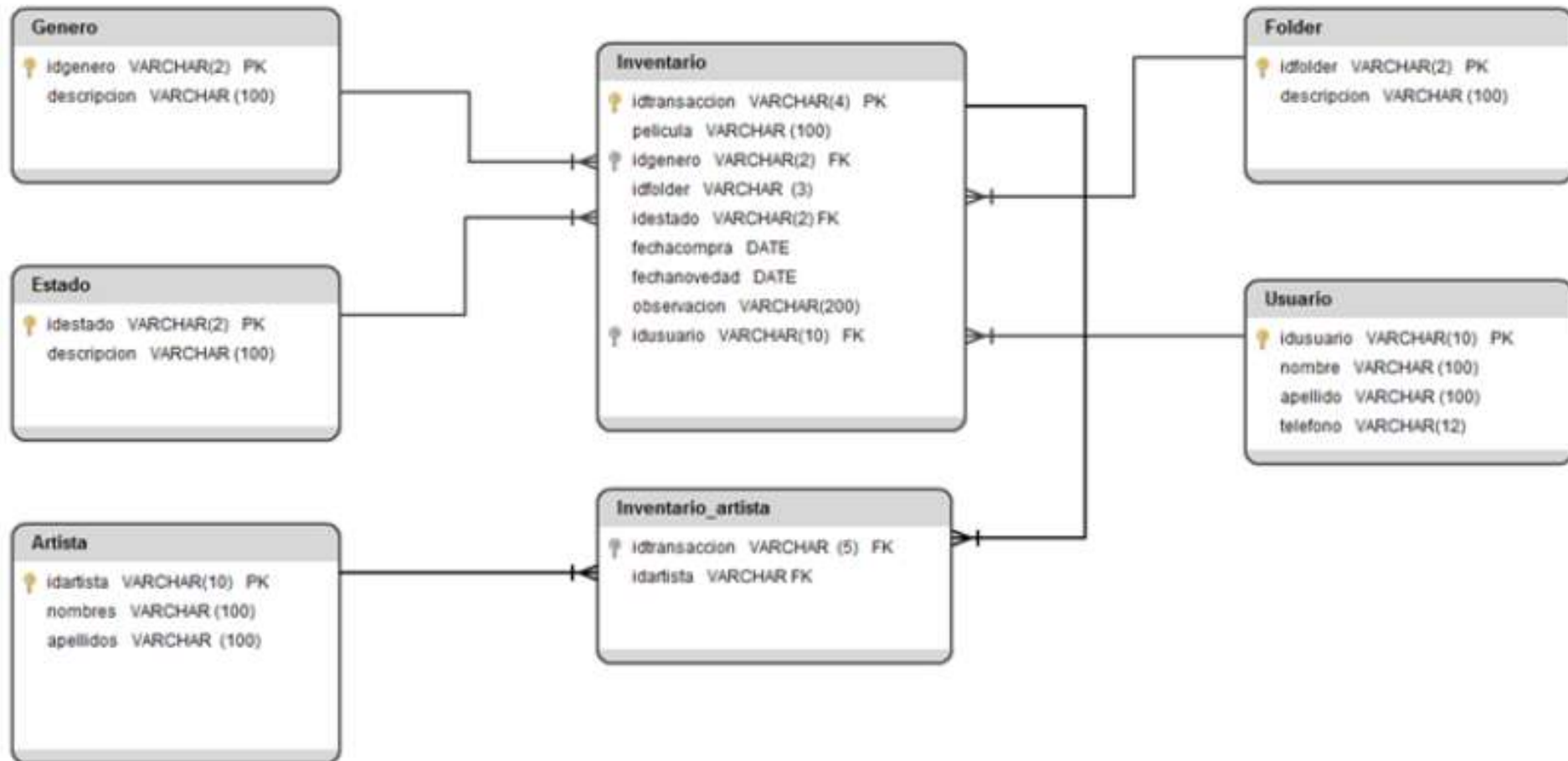


# 7. Esquema relacional





# 7. Esquema relacional



## 8. Normalización

1. Concepto de normalización
2. Dependencias funcionales
3. Primera forma normal (1FN)
4. Segunda forma normal (2FN)
5. Tercera forma normal (3FN)
6. Forma normal de Boyce-Codd (FNBC)
7. Cuarta forma normal (4FN)
8. Quinta forma normal (5FN)
9. Forma normal de dominio clave (FNDC)

# 8.1. Concepto de normalización

- Esquema relacional obtenido puede presentar problemas
  - ▣ Redundancia
    - Los datos se repiten continua e innecesariamente por las tablas
    - Requiere revisión del diseño
    - Primer síntoma de problemas
    - Detección fácil
  - ▣ Ambigüedades
    - Los datos no identifican claramente al elemento que representan
      - Una fila podría tener datos de más de un ejemplar de la tabla
      - Una fila podría tener datos sin saber de qué ejemplar exactamente
    - Problema muy grave
    - Detección difícil

# 8.1. Concepto de normalización

- Esquema relacional obtenido puede presentar problemas
  - ▣ Pérdida de restricciones de integridad
    - Consecuencia de las *dependencias funcionales* existentes
    - Solución sencilla siguiendo pasos concretos
  - ▣ Anomalías en operaciones de modificación de datos
    - Una operación implica realizar varias sobre una tabla
      - Inserción de un elemento → Variación de tuplas de la tabla
      - Eliminación de un elemento → Eliminación de varias tuplas de la tabla
    - Error muy grave
      - Denota un diseño mal realizado

# 8.1. Concepto de normalización

- Para evitar esos problemas
  - ▣ Tablas han de referirse a elementos muy concretos
  - ▣ Cada fila de la tabla ha de ser un elemento del sistema
    - Reconocible inequívocamente
  - ▣ Usar modelo conceptual previo al modelo lógico
    - De no ser así, garantía casi asegurada de errores
  - ▣ Detectar posibles problemas aparecidos
  - ▣ Aplicar reglas de normalización para solventarlos
    - Suelen implicar división de una tabla en varias



# 8.1. Concepto de normalización

La **normalización** es una técnica para diseñar la estructura lógica de los datos de un sistema de información en el modelo relacional, desarrollada por E. F. Codd en 1972. Es una estrategia de diseño de abajo arriba: se parte de los atributos y éstos se van agrupando en relaciones (tablas) según su afinidad. Aquí no se utilizará la normalización como una técnica de diseño de bases de datos, sino como una etapa posterior a la correspondencia entre el esquema conceptual y el esquema lógico, que elimine las dependencias no deseadas entre atributos. Las ventajas de la normalización son las siguientes:

- Evita anomalías en inserciones, modificaciones y borrados.
- Mejora la independencia de datos.
- No establece restricciones artificiales en la estructura de los datos.

Uno de los conceptos fundamentales en la normalización es el de *dependencia funcional*. Una **dependencia funcional** es una relación entre atributos de una misma relación (tabla).

# 8.1. Concepto de normalización

## □ Formas normales

- ▣ Teorías matemáticas para depuración del modelo
- ▣ Iniciadas por Codd en 1970 con la primera forma normal
- ▣ Seguidas por otros autores como Boyce y Fagin
- ▣ Siguen un orden incremental en lo restrictivas que son
  - La segunda forma normal (2FN) cumple también la 1FN
  - La tercera forma normal (3FN) cumple también la 2FN
  - Y así sucesivamente
    - $1FN \rightarrow 2FN \rightarrow 3FN \rightarrow FNBC \rightarrow 4FN \rightarrow 5FN \rightarrow FNDC$
- ▣ Un buen diseño debería estar como mínimo en FNBC
  - Hay quien piensa que diseños en 4FN y 5FN son peores

## 8.2. Dependencias funcionales

- Dependencia funcional:  $X \rightarrow Y$ 
  - ▣ “Y depende funcionalmente de X”
  - ▣ Para cada valor de X hay un solo posible valor de Y
  - ▣  $X = \text{determinante}$ 
    - Conjunto de atributos del que depende funcionalmente Y
  - ▣  $Y = \text{implicado}$ 
    - Conjunto de atributos que dependen funcionalmente de X

## 8.2. Dependencias funcionales

□ Dependencia funcional:  $X \rightarrow Y$

Profesores			
Nif	Nombre	Apellido	Departamento
12345678A	Marta	López	Administración
87654321B	Sergio	San Victoriano	Informática
87654321B	Sergio	San Victoriano	Matemáticas

□  $\text{Nif} \rightarrow \text{Nombre}$  : dependencia funcional existente

■ Nombre depende funcionalmente de Nif

■ Para un mismo Nif sólo se puede encontrar un nombre en la tabla

□  ~~$\text{Nif} \rightarrow \text{Departamento}$~~  : dependencia funcional NO existente

■ Departamento NO depende funcionalmente de Nif

■ Para un mismo Nif puede haber varios departamentos en la tabla

## 8.2. Dependencias funcionales

- Dependencia funcional completa:  $X \Rightarrow Y$ 
  - ▣ “Y tiene dependencia funcional completa sobre X”
    - Y depende funcionalmente de X
    - No existe subconjunto de X que consiga la misma dependencia
  
- ▣  $\text{Nif, Nombre} \rightarrow \text{Apellido}$ 
  - Dependencia funcional, pero no completa
    - A partir de los atributos Nif y Nombre se obtiene sólo un Apellido
    - El determinante no es un conjunto mínimo de atributos
  
- ▣  $\text{Nif} \Rightarrow \text{Apellido}$ 
  - Dependencia funcional completa

## 8.2. Dependencias funcionales

### □ Dependencia funcional elemental

#### □ $X \rightarrow Y$ es elemental si

- Es una dependencia funcional completa ( $X \Rightarrow Y$ )
- Y (implicado) es un único atributo no contenido en X

### □ Dependencia funcional transitiva

#### □ Si $X \rightarrow Y$ y $Y \rightarrow Z$ , entonces $X \rightarrow Z$

- Profesores(Nif, Nombre, Ape, CodDepto, NomDepto)
  - $Nif \rightarrow CodDepto$  y  $CodDepto \rightarrow NomDepto$ , entonces
    - $Nif \rightarrow NomDepto$

## 8.2. Dependencias funcionales

### □ Ejemplo

Codigo_p	Nombre	Telefono	Codigo_a	Descripcion	Precio
p50	Juan	5576581	A10	tuerca	20
p32	Pedro	3349631	A20	tornillo	30
p50	Juan	5576581	A20	tornillo	32
p32	Pedro	3349631	A10	tuerca	18
p38	Jose	7755633	A49	martillo	45

La Entidad **Proveedor\_Articulo** describe los siguientes Atributos:  
Código del Proveedor, Nombre del Proveedor, Teléfono del Proveedor,  
Código del Artículo, Descripción del Artículo y Precio del Artículo

## 8.2. Dependencias funcionales

Codigo_p	Nombre	Telefono	Codigo_a	Descripcion	Precio
p50	Juan	5576581	A10	tuerca	20
p32	Pedro	3349631	A20	tornillo	30
p50	Juan	5576581	A20	tornillo	32
p32	Pedro	3349631	A10	tuerca	18
p38	Jose	7755633	A49	martillo	45

Código\_p → Nombre, Teléfono

Codigo\_a → Descripción

Código\_p, Código\_a → Precio



## 8.3. Reglas de normalización

Se dice que una relación está en una **forma normal** si satisface un cierto conjunto específico de restricciones impuestas por la regla de normalización correspondiente. La **aplicación de una regla** es una operación que toma como entrada una relación y da como resultado dos o más relaciones.

## 8.3. Primera forma normal (1FN)

- Requisitos de una tabla en 1FN
  - ▣ 1 atributo en 1 tupla no puede tomar más de un valor
    - Si tiene que tomar varios valores, será en distintas tuplas
- 1FN es inherente al modelo relacional
  - ▣ Toda tabla del modelo lo cumple
  - ▣ Es la menos restrictiva de todas
  - ▣ Todas las demás formas normales están en 1FN

## 8.3. Primera forma normal (1FN)

Profesores			
Nif	Nombre	Apellido	Departamento
12345678A	Marta	López	Administración
87654321B	Sergio	San Victoriano	Informática Matemáticas

NO está en 1FN

Profesores			
Nif	Nombre	Apellido	Departamento
12345678A	Marta	López	Administración
87654321B	Sergio	San Victoriano	Informática
87654321B	Sergio	San Victoriano	Matemáticas

SÍ está en 1FN

## 8.3. Primera forma normal (1FN)

Otro ejemplo: consideramos la tabla ALUMNO, con clave primaria COD\_ALUMNO, en la que el atributo TLF puede tomar varios valores: el movil, el de casa, el del padre, el de la madre, etc. (ver Tabla 2.3.)

<u>COD_ALUMNO</u>	NOMBRE	APELLIDO	TLF	DIRECCION
1111	PEPE	GARCÍA	678-900800 91-2233441 91-1231232	C/Las cañas 45
2222	MARÍA	SUÁREZ	91-7008001	C/Mayor 12
3333	JUAN	GIL	91-7562324 660-111222	C/La plaza
4444	FRANCISCO	MONTAYA	678-556443	C/La arboleda

Tabla 2.3. Tabla ALUMNO que no está en 1FN.

## 8.3. Primera forma normal (1FN)

Esta tabla no está en 1FN, ya que hay dos alumnos que tienen varios teléfonos. Para que esta tabla esté en 1FN se puede:

- Definir como clave primaria de la tabla COD\_ALUMNO, y el TLF, con el fin de que cada atributo tome un único valor en la tupla correspondiente. Ver Tabla 2.4.

<u>COD_ALUMNO</u>	<u>TLF</u>	NOMBRE	APELLIDO	DIRECCIÓN
1111	678-900800	PEPE	GARCÍA	C/Las cañas 45
1111	91-2233441	PEPE	GARCÍA	C/Las cañas 45
1111	91-1231232	PEPE	GARCÍA	C/Las cañas 45
2222	91-7008001	MARÍA	SUÁREZ	C/Mayor 12
3333	91-7562324	JUAN	GIL	C/La plaza
3333	660-111222	JUAN	GIL	C/La plaza
4444	678-556443	FRANCISCO	MONTOYA	C/La arboleda

Tabla 2.4. Tabla ALUMNO, primera transformación a 1FN.

## 8.3. Primera forma normal (1FN)

- O también se eliminan los grupos repetitivos (TLF) y se crea una relación (tabla) junto con la clave inicial. Ver Tablas 2.5 y 2.6.

<u>COD_ALUMNO</u>	NOMBRE	APELLIDO	DIRECCION
1111	PEPE	GARCÍA	C/Las cañas 45
2222	MARÍA	SUÁREZ	C/Mayor 12
3333	JUAN	GIL	C/La plaza
4444	FRANCISCO	MONTOYA	C/La arboleda

**Tabla 2.5.** *Tabla ALUMNO segunda transformación a 1FN.*

La segunda tabla estará formada por el TLF, y el COD\_ALUMNO. Los dos formarán la clave. Ver Tabla 2.6:

<u>COD_ALUMNO(FK)</u>	<u>TLF</u>
1111	678-900800
1111	91-2233441
1111	91-1231232
2222	91-7008001

**Tabla 2.6.** *Tabla ALUM\_TLF en 1FN.*

## 8.4. Segunda forma normal (2FN)

- Requisitos de una tabla en 2FN
  - ▣ La tabla está en 1FN
  - ▣ Cada atributo no clave depende de forma funcional completa de todas las claves
  
- Por tanto
  - ▣ Toda clave principal debe hacer dependientes a todos los atributos
  - ▣ Si hay atributos que sólo dependen de parte de la clave
    - Esos atributos y la parte indicada de la clave = Nueva tabla

## 8.4. Segunda forma normal (2FN)

### □ Ejemplo

- Situación inicial  
(NO en 2FN)

*Alumnos(nif, codAsignatura, nombre, apellido, nota)*

Alumnos				
Nif	CodAsignatura	Nombre	Apellido	Nota
85285285A	25	Pepe	Pérez	9.1
12312313B	14	Luis	Lorenzo	3.4
12312313B	25	Luis	Lorenzo	5.2
98798798B	32	María	Salinas	8.1
98798798B	14	María	Salinas	6.7

Dependencias funcionales completas

$(Nif, CodAsignatura) \Rightarrow Nota$

Dependencias funcionales (no completas)

$(Nif, CodAsignatura) \rightarrow Nombre$  (bastaría con Nif)

$(Nif, CodAsignatura) \rightarrow Apellido$  (bastaría con Nif)



## 8.4. Segunda forma normal (2FN)

### □ Ejemplo

#### ▣ Conversión a 2FN

Alumnos		
Nif	Nombre	Apellido
85285285A	Pepe	Pérez
12312313B	Luis	Lorenzo
98798798B	María	Salinas

Dependencias funcionales completas

Nif => Nombre

Nif => Apellido

Dependencias funcionales incompletas

No hay

Matrículas(nif, codAsignatura, nota)

Alumnos(nif, nombre, apellido)

Matrículas		
Nif	CodAsignatura	Nota
85285285A	25	9.1
12312313B	14	3.4
12312313B	25	5.2
98798798B	32	8.1
98798798B	14	6.7

Dependencias funcionales completas

(Nif, CodAsignatura) => Nota

Dependencias funcionales incompletas

No hay

## 8.4. Segunda forma normal (2FN)

Supongamos que tenemos una relación ALUMNO en la que representamos los datos de los alumnos y las notas en cada una de las asignaturas en que está matriculado. La clave es el número de matrícula COD\_ALUMNO y la asignatura ASIGNATURA (ver Tabla 2.7.)

<u>COD_ALUMNO</u>	NOM_ALUM	APE_ALUM	<u>ASIGNATURA</u>	NOTA	CURSO	AULA
1111	PEPE	GARCÍA	LENGUA I	5	1	15
1111	PEPE	GARCÍA	IDIOMA	5	2	16
2222	MARÍA	SUAREZ	IDIOMA	7	2	16
2222	MARÍA	SUAREZ	CIENCIAS	7	2	14
3333	JUAN	GIL	PLÁSTICA	6	1	18
3333	JUAN	GIL	MATEMÁTICAS I	6	1	12
4444	FRANCISCO	MONTOYA	LENGUA II	4	2	11
4444	FRANCISCO	MONTOYA	MATEMÁTICAS I	6	1	12
4444	FRANCISCO	MONTOYA	CIENCIAS	8	1	14

**Tabla 2.7.** Relación ALUMNO para transformar a 2FN.

Es obvio que todos los atributos no dependen de la clave completa (COD\_ALUMNO, ASIGNATURA). En primer lugar, hay que ver las dependencias funcionales de cada uno de los atributos con respecto a los atributos de la clave y el resto de atributos:

- NOM\_ALUM y APE\_ALUM, sólo dependen de COD\_ALUMNO.
- Los atributos CURSO y AULA están relacionados con la ASIGNATURA, es decir, existe una dependencia entre ASIGNATURA → CURSO, ASIGNATURA → AULA. Una asignatura pertenece a un curso y se imparte en un aula.
- El atributo NOTA depende funcionalmente de la clave, pues para que haya una nota tiene que haber una asignatura y un alumno.

## 8.4. Segunda forma normal (2FN)

Vistas las dependencias funcionales llegamos a la siguiente conclusión para que la relación ALUMNO esté en 2FN necesitamos crear tres relaciones: ALUMNO, ASIGNATURAS y NOTAS (ver Tablas 2.8, 2.9 y 2.10):

<u>COD_ALUMNO</u>	NOM_ALUM	APE_ALUM
1111	PEPE	GARCÍA
2222	MARÍA	SUAREZ
3333	JUAN	GIL
4444	FRANCISCO	MONTOYA

Tabla 2.8. Tabla ALUMNO en 2FN.

<u>ASIGNATURA</u>	CURSO	AULA
LENGUA I	1	15
IDIOMA	2	16
CIENCIAS	2	14
PLÁSTICA	1	18
MATEMÁTICAS I	1	12
LENGUA II	2	11

Tabla 2.9. Tabla ASIGNATURAS en 2FN.

<u>COD_ALUMNO (FK)</u>	<u>ASIGNATURA (FK)</u>	NOTA
1111	LENGUA I	5
1111	IDIOMA	5
2222	IDIOMA	7
2222	CIENCIAS	7
3333	PLÁSTICA	6
3333	MATEMÁTICAS I	6
4444	LENGUA II	4
4444	MATEMÁTICAS I	6
4444	CIENCIAS	8

Tabla 2.10. Tabla NOTAS en 2FN.

## 8.5. Tercera forma normal (3FN)

- Requisitos de una tabla en 3FN
  - ▣ La tabla está en 2FN
  - ▣ Ningún atributo no clave depende transitivamente de las claves
- Por tanto
  - ▣ Ningún atributo depende de otro que no es clave
  - ▣ Si algún atributo depende de otro que no es clave
    - Esos atributos = Nueva tabla o relación
      - Clave = Atributo del que dependía el/los otro/s
    - En la tabla original, la clave de la otra será ahora migrada

## 8.5. Tercera forma normal (3FN)

### □ Ejemplo

- Situación inicial  
(NO en 3FN)

**Profesores**(nif, nombre, apellido, codDepto, nomDepto)

Profesores				
Nif	Nombre	Apellido	CodDepto	NomDepto
12345678A	Sandra	Ruiz	9	Latín
87654321B	Luis	Sanz	7	Informática
12121212B	Roberto	Pons	1	Matemáticas
15915915Z	Lucía	Eras	7	Informática
78787878L	Rocío	Lares	1	Matemáticas

Dependencias funcionales transitivas

Nif → CodDepto → NomDepto



Atributo NomDepto (no clave)  
depende transitivamente de Nif (clave)



Atributo NomDepto (no clave)  
depende de CodDepto (no clave)

## 8.5. Tercera forma normal (3FN)

### □ Ejemplo

#### ▣ Conversión a 3FN

Profesores(nif, nombre, apellido, codDepto)  
Departamentos(codDepto, nomDepto)

Departamentos	
CodDepto	NomDepto
9	Latín
7	Informática
1	Matemáticas

Dependencias funcionales transitivas  
No hay

Profesores			
Nif	Nombre	Apellido	CodDepto
12345678A	Sandra	Ruiz	9
87654321B	Luis	Sanz	7
12121212B	Roberto	Pons	1
15915915Z	Lucía	Eras	7
78787878L	Rocío	Lares	1

Dependencias funcionales transitivas  
No hay

## 8.5. Tercera forma normal (3FN)

Ejemplo: supongamos que tenemos una relación LIBROS en la que representamos los datos de las editoriales de los mismos (ver *Tabla 2.11.*)

<u>COD_LIBRO</u>	TÍTULO	EDITORIAL	PAÍS
12345	DISEÑO DE BD RELACIONALES	RAMA	ESPAÑA
34562	INSTALACIÓN y MANTENIMIENTO DE EQUIPOS	MCGRAW-HILL	ESPAÑA
72224	FUNDAMENTOS DE PROGRAMACIÓN	SANTILLANA	ESPAÑA
34522	BASE DE DATOS 00	ADDISON	EEUU

Tabla 2.11. Relación LIBROS para transformar a 3FN.

## 8.5. Tercera forma normal (3FN)

Veamos las dependencias con respecto a la clave:

- TÍTULO y EDITORIAL dependen directamente del código del libro.
- El PAÍS, aunque en parte también depende del libro, está más ligado a la EDITORIAL a la que pertenece el libro. Por esta última razón, la relación libros no está en 3FN, la solución se muestra en las Tablas 2.12 y 2.13:

<u>COD_LIBRO</u>	TÍTULO	EDITORIAL (FK)
12345	DISEÑO DE BD RELACIONALES	RAMA
34562	INSTALACIÓN y MANTENIMIENTO DE EQUIPOS	MCGRAW-HILL
72224	FUNDAMENTOS DE PROGRAMACIÓN	SANTILLANA
34522	BASE DE DATOS OO	ADDISON

Tabla 2.12. Tabla LIBROS en 3FN.

<u>EDITORIAL</u>	PAÍS
RAMA	ESPAÑA
MCGRAW-HILL	ESPAÑA
SANTILLANA	ESPAÑA
ADDISON	EEUU

Tabla 2.13. Tabla EDITORIAL en 3FN. ▲



## 8.5. Actividad propuesta

Dada la tabla 2.14 normalizar hasta la 3FN.

DNI	NOMBRE	APELLIDOS	DIRECCIÓN	C_POST	POBLACIÓN	PROVINCIA
413245-B	JUAN	RAMOS	C/Las cañas 59	19005	GUADALAJARA	GUADALAJARA
23456-J	PEDRO	PÉREZ	C/Pilón 12	45589	CALERUELA	TOLEDO
			C/Vitoria 3	28804	ALCALÁ DE HENARES	MADRID
34561-B	MARÍA	RODRÍGUEZ	C/El altozano	10392	BERROCALEJO	CÁCERES
			C/Sanz Vázquez 2	19004	GUADALAJARA	GUADALAJARA
222346-J	JUAN	CABELLO	C/El ensanche 3	28802	ALCALÁ DE HENARES	MADRID
			C/Los abedules 10	10300	NAVALMORAL DE LA MATA	CÁCERES

Tabla 2.14. Transformar a 3FN.

## 8.6. F.N. de Boyce-Codd (FNBC)

- Requisitos de una tabla en FNBC
  - ▣ La tabla está en 3FN
  - ▣ Todo determinante es clave candidata
  
- Por tanto
  - ▣ Existiría redundancia causada por mala selección de clave
  - ▣ Si esto ocurre, no estando en FNBC: solución
    - Descomponer en otra tabla que albergue la dependencia
      - Clave primaria: determinante de la dependencia
      - Atributo/s: implicado/s de la dependencia

## 8.6. F.N. de Boyce-Codd (FNBC)

- Ejemplo: Educación a distancia
  - ▣ Requisitos
    - Un alumno puede estar matriculado de varias asignaturas
    - Un alumno tiene un tutor para cada asignatura
    - Cada tutor sólo puede serlo de una asignatura
    - La misma asignatura puede tener varios tutores

## 8.6. F.N. de Boyce-Codd (FNBC)

### □ Ejemplo

- Diseño inicial  
(NO en FNBC)

Dependencias funcionales  
Alumno, Asignatura → Tutor  
Tutor → Asignatura



La segunda tiene determinante  
que no es clave candidata: Tutor



No está en FNBC  
(hay redundancia)

Alumnos(alumno, asignatura, tutor)

Alumnos		
Alumno	Asignatura	Tutor
85285285A	Bases de Datos	Sergio
85285285A	Programación	Pilar
98798798B	Bases de Datos	Sergio
12312312C	Bases de Datos	Luis
85285285A	Sistemas	Marisol
12312312C	Programación	Carlos

## 8.6. F.N. de Boyce-Codd (FNBC)

### □ Ejemplo

- ▣ Conversión a FNBC

Tutores(tutor, asignatura)

Alumnos(alumno, tutor)

Tutores	
Tutor	Asignatura
Sergio	Bases de datos
Pilar	Programación
Luis	Bases de datos
Marisol	Sistemas
Carlos	Programación

Alumnos	
Alumno	Tutor
85285285A	Sergio
85285285A	Pilar
98798798B	Sergio
12312312C	Luis
85285285A	Marisol
12312312C	Carlos

Está en FNBC



Todos los determinantes  
son claves candidatas (Tutor)



Dependencias funcionales  
Tutor → Asignatura

## 8.6. F.N. de Boyce-Codd (FNBC)

### □ Otro ejemplo

#### ▣ Conversión a FNBC

Supongamos que tenemos una relación EMPLEADOS en la que representamos los datos de los empleados de una fábrica, ver Tabla 2.15:

DNI	NÚM_SEG_SOC	NOMBRE	APELLIDOS	DEPARTAMENTO	PUESTO	SALARIO
413245-B	28-1234566	JUAN	RAMOS	COMPRAS	GERENTE	2.300
23456-J	28-2345686	PEDRO	PÉREZ	NÓMINAS	AUXILIAR	1.200
123123-C	19-458766	MARÍA	GIL	ALMACÉN	CONSERJE	1.530
1234556-B	45-223344	ANTONIO	SANZ	COMPRAS	GESTIÓN	2.200

**Tabla 2.15.** Relación EMPLEADOS para transformar a FNBC.

# 8.6. F.N. de Boyce-Codd (FNBC)

Observando los atributos de esta relación podemos ver que NUM\_SEG\_SOC y el grupo NOMBRE-APELLIDOS son claves candidatas (determinantes). Esta relación se transforma en dos tablas: una contendrá la clave junto con las claves candidatas (EMPLEADOS) y la otra la clave con el resto de campos (EMPLE\_TRABAJO), ver Tablas 2.16 y 2.17:

<u>DNI</u>	NUM_SEG_SOC	NOMBRE	APELLIDOS
413245-B	28-1234566	JUAN	RAMOS
23456-J	28-2345686	PEDRO	PÉREZ
123123-C	19-458766	MARÍA	GIL
1234556-B	45-223344	ANTONIO	SANZ

Tabla 2.16. Relación EMPLEADOS e FNBC.

<u>DNI (FK)</u>	DEPARTAMENTO	PUESTO	SALARIO
413245-B	COMPRAS	GERENTE	2.300
23456-J	NÓMINAS	AUXILIAR	1.200
123123-C	ALMACÉN	CONSERJE	1.530
1234556-B	COMPRAS	GESTIÓN	2.200

Tabla 2.17. Relación EMPLERTRABAJO en FNBC.

## 8.7.1. Dependencia multivaluada

- Base de la 4FN, 5FN y FNDC (diseñadas por *Fagin*)
- $X \twoheadrightarrow Y$  : “Dependencia multivaluada de X sobre Y”
  - X e Y atributos de la misma tabla
  - Y tiene conjunto de valores definidos para cualquier valor de X
- Implica la existencia de redundancia



# 8.7.1. Dependencia multivaluada

- Está en FNBC
  - ▣ No dependencias transitivas
  - ▣ No dependencias con determinante no clave
- Hay redundancia
  - ▣ Las versiones se repiten para todos los concesionarios
    - El Insignia siempre tiene Sedán y Coupé
    - El Astra siempre tiene Sedán, Ranchera y Coupé
- Modelo ->> Versión
  - ▣ Versión depende de forma multivaluada de modelo
  - ▣ No se necesita saber el concesionario
  - ▣ No hay dependencia funcional Modelo → Versión (varias versiones para un Modelo)

Concesionarios		
Modelo	Concesionario	Versión
Opel Insignia	Opel Salamanca	Sedán
Opel Insignia	Opel Salamanca	Coupé
Opel Insignia	Opel Ávila	Sedán
Opel Insignia	Opel Ávila	Coupé
Opel Astra	Opel Ávila	Sedán
Opel Astra	Opel Ávila	Ranchera
Opel Astra	Opel Ávila	Coupé

## 8.7. Cuarta forma normal (4FN)

- Requisitos de una tabla en 4FN
  - ▣ La tabla está en FNBC
  - ▣ No posee dependencias multivaluadas no triviales
    - Pueden existir dependencias multivaluadas triviales en 4FN
      - Trivial:
        - $X \twoheadrightarrow Y$  es trivial si X e Y son los únicos atributos de la tabla
- Paso a 4FN
  - ▣ Descomposición en dos tablas

## 8.7. Cuarta forma normal (4FN)

### □ Ejemplo

#### ▣ Conversión a 4FN

Modelos	
Modelo	Versión
Opel Insignia	Sedán
Opel Insignia	Coupé
Opel Astra	Sedán
Opel Astra	Ranchera
Opel Astra	Coupé

Concesionarios	
Modelo	Concesionario
Opel Insignia	Opel Salamanca
Opel Insignia	Opel Ávila
Opel Astra	Opel Ávila

Dependencia multivaluada trivial  
Modelo ->> Versión



Está en 4FN, pues son las no triviales las no permitidas.

## 8.7. Cuarta forma normal (4FN)

### □ Otro ejemplo

#### ▣ Conversión a 4FN

Consideramos la siguiente relación: GEOMETRÍA, con tres atributos con dependencias multivaluadas, los tres atributos forman la clave, ver Tabla 2.18:

<u>FIGURA</u>	<u>COLOR</u>	<u>TAMAÑO</u>
ESFERA	ROJO	GRANDE
ESFERA	VERDE	GRANDE
CUBO	BLANCO	GRANDE
CUBO	AZUL	GRANDE
PIRÁMIDE	BLANCO	MEDIANO
PIRÁMIDE	BLANCO	GRANDE
PIRÁMIDE	ROJO	GRANDE

Tabla 2.18. Relación GEOMETRÍA con dependencias multivaluadas.

## 8.7. Cuarta forma normal (4FN)

En la Tabla 2.18, FIGURA determina valores múltiples de COLOR y TAMAÑO, pero COLOR y TAMAÑO son independientes entre sí. Así pues, las dependencias son FIGURA  $\rightarrow \rightarrow$  COLOR y FIGURA  $\rightarrow \rightarrow$  TAMAÑO. Observamos que se repiten ESFERA-GRANDE, o PIRÁMIDE-BLANCO, o PIRÁMIDE-GRANDE. Para eliminar la redundancia de los datos se deben eliminar las dependencias de valores múltiples. Esto se logra construyendo dos tablas, donde cada una almacena datos para solamente uno de los atributos de valores múltiples.

Estas repeticiones hacen que la tabla no esté en 4FN. Para pasarla a 4FN, creamos las Tablas 2.19 y 2.20:

<u>FIGURA</u>	<u>COLOR</u>
ESFERA	ROJO
ESFERA	VERDE
CUBO	BLANCO
CUBO	AZUL
PIRÁMIDE	BLANCO
PIRÁMIDE	ROJO

Tabla 2.19. Tabla COLOR en 4FN.

<u>FIGURA</u>	<u>TAMAÑO</u>
ESFERA	GRANDE
CUBO	GRANDE
CUBO	MEDIANO
PIRÁMIDE	GRANDE

Tabla 2.20. Tabla TAMAÑOS en 4FN.

## 8.7. Cuarta forma normal (4FN)

### Actividades propuestas

- 3 Normaliza la Tabla 2.21. Suponemos que los estudiantes de una academia pueden inscribirse en varias especialidades y en varios cursos. Por ejemplo, el número de matrícula 12345 tiene especialidades en Sistemas y Telemática y toma los cursos de Natación y Danza.

<u>NUMATRICULA</u>	<u>ESPECIALIDAD</u>	<u>CURSO</u>
12345	SISTEMAS	NATACIÓN
12345	TELEMÁTICA	DANZA
34567	SISTEMAS	PIANO
34567	DESARROLLO	NATACIÓN
67891	TELEMÁTICA	AERÓBIC

Tabla 2.21. *Tabla ACADEMIA.*

## 8.8. Quinta forma normal (5FN)

- Requisitos de una tabla en 5FN
  - ▣ Está en 4FN
  - ▣ Cada dependencia de unión es implicada por las claves candidatas
  
- Consideraciones
  - ▣ Definida por Fagin
  - ▣ Es la más compleja y polémica
    - No está claro que mejore la base de datos
  - ▣ Derivada de restricciones semánticas especiales
    - Tablas en 4FN rara vez presentarán estos problemas

## 8.8. Quinta forma normal (5FN)

- Ejemplo: *Requisitos previos*
  - ▣ Se quiere saber qué profesores pueden dar qué materias
  - ▣ Se quiere saber en qué academias imparten
  - ▣ Un profesor puede impartir varias materias
  - ▣ Un profesor puede impartir en varias academias
  - ▣ La misma materia la pueden impartir varios profesores
    - Incluso en la misma academia



## 8.8. Quinta forma normal (5FN)

### □ Ejemplo: *Primera propuesta de solución*

Profesor	Academia	Materia
Sergio	A1	Bases de Datos
Sergio	A1	Programación
Pepe	A2	Bases de Datos
Pepe	A2	Programación
Pepe	A2	Sistemas
Pepe	A1	Programación
María	A1	Lenguajes de Marcas
María	A1	Bases de Datos
María	A2	Sistemas

## 8.8. Quinta forma normal (5FN)

### □ Ejemplo: *Requisito adicional*

- Todo profesor capacitado para dar una materia la podrá impartir en cualquier academia que imparta esa materia y para la cual trabaje.
- En la solución anterior:
  - Si ahora María se capacita para dar Programación...
    - Necesario añadir dos registros
      - Uno para la academia A1 (que imparte Programación)
      - Otro para la academia A2 (que imparte Programación)

## 8.8. Quinta forma normal (5FN)

### □ Ejemplo: *Solución sin redundancia*

Profesor	Materia
Sergio	Bases de Datos
Sergio	Programación
Pepe	Bases de Datos
Pepe	Programación
Pepe	Sistemas
María	Bases de Datos
María	Lenguajes de Marcas
María	Sistemas

Profesor	Academia
Sergio	A1
Pepe	A1
Pepe	A2
María	A1
María	A2

Academia	Materia
A1	Bases de Datos
A1	Programación
A1	Lenguajes de Marcas
A2	Bases de Datos
A2	Programación
A2	Sistemas

## 8.8. Quinta forma normal (5FN)

### □ Ejemplo: *Resultados*

- Si ahora María se capacita para impartir Programación, sólo añadiendo un registro en la primera de esas tres tablas sería suficiente

## 8.9. Forma normal de dominio clave(FNDC)

- Consideraciones previas
  - ▣ Enunciada por Fagin (1981)
  - ▣ Derivada de problemas de redundancia con algunos dominios
  - ▣ No basada en dependencias entre los datos
  - ▣ Basada en restricciones de dominio y de clave
    - Restricciones de dominio
      - Para que un atributo sólo obtenga ciertos valores
    - Restricciones de clave
      - Para que un atributo o conjunto sean clave candidata

## 8.9. Forma normal de dominio clave(FNDC)

- Enunciado: una tabla está en FNDC si...
  - ▣ Toda restricción sobre ella es consecuencia de aplicar
    - Las restricciones de dominio
    - Las restricciones de clave
- Fagin demuestra que al ocurrir esto, la tabla está también en 5FN

## 8.9. Forma normal de dominio clave(FNDC)

### □ Ejemplo: *Situación inicial*

Alumno	Nivel alcanzado	Nota
Pepe	Muy alto y eficiente	9
María	Muy alto con trabajo constante	9
Ramón	Alto con trabajo habitual	7
Marta	Medio aunque con trabajo	5

### □ Ejemplo: *Observaciones*

- Siempre que la nota es 9, aparece nivel “Muy alto”
- Siempre que la nota es 7, aparece nivel “Alto”
- Siempre que la nota es 5, aparece nivel “Medio”

## 8.9. Forma normal de dominio clave(FNDC)

### □ Ejemplo: *Restricciones de dominio y clave*

Alumno	Nivel de trabajo	Nota
Pepe	Eficiente	9
María	Trabajo constante	9
Ramón	Trabajo habitual	7
Marta	Trabajo media	5

Nivel académico	Nota mínima	Nota máxima
Muy alto	9	10
Alto	7	8,99
Medio	5	6,99
Bajo	0	4,99



# Unidad 3.

## Diseño lógico:

## El Modelo Relacional