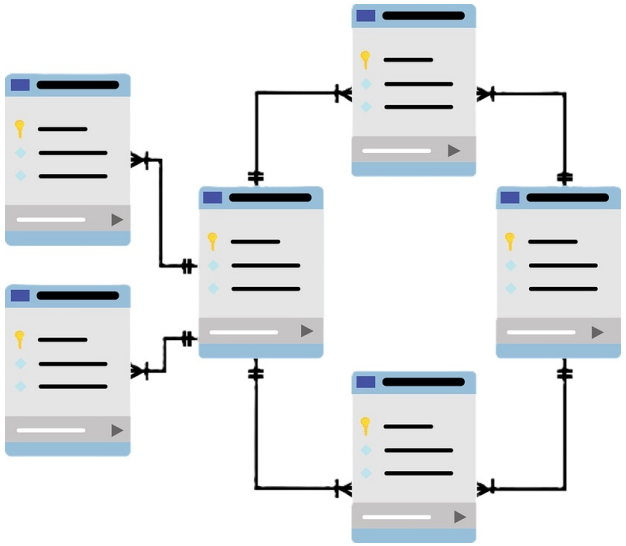


Modelado Lógico

El modelo relacional
(Implementable)

El modelo Relacional

Un Esquema Conceptual es una descripción de alto nivel de la estructura de la Base de Datos, independientemente del **SGBD** que se vaya a utilizar para manipularlo.

Diseño lógico es la etapa en la que se transforma el esquema conceptual en un esquema lógico que utilizará las estructuras de datos del modelo de base de datos en el que se basa el **SGBD** que se vaya a utilizar, como puede ser el **modelo relacional**, el modelo de red, el modelo jerárquico o el modelo orientado a objetos.

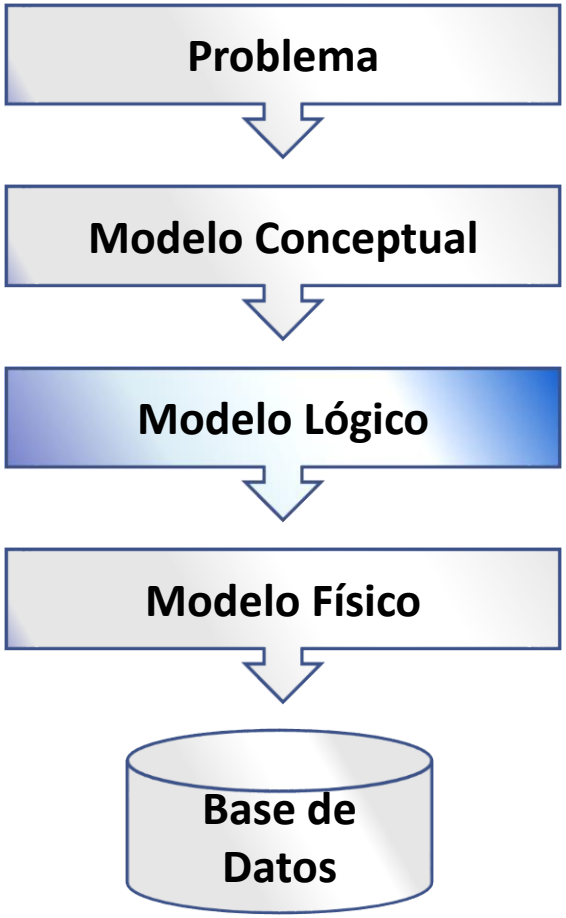
El modelo Relacional

El objetivo del diseño lógico es transformar el esquema conceptual en un modelo de datos para ser implementado en un Sistema de Gestión de Bases de Datos (**SGBD o DBMS**) determinado.

Un modelo lógico es una descripción de la estructura de la base de datos en términos de la estructura de datos que puede procesar un tipo de **SGBD**. El diseño lógico depende del tipo de SGBD que se utiliza, no depende de un producto concreto.

Un **modelo de base de datos** es un tipo de modelo de datos que determina la estructura lógica de una base de datos y de manera fundamental determina el modo de almacenar, organizar y manipular los datos

El modelo Relacional



El modelo Relacional

El **modelo relacional**, (1970 -Edgar Frank Codd) está basado en la lógica de predicados y en la teoría de conjuntos. Se consolidó como el paradigma del modelado de base de datos, y persigue una serie de objetivos:

- **Independencia física:** El modo cómo se almacenan los datos no debe influir en su manipulación lógica y, por tanto, los usuarios que acceden a esos datos no han de modificar sus programas por cambios en el almacenamiento físico

- **Independencia lógica:** Añadir, eliminar o modificar cualquier elemento de la BD no debe repercutir en los programas y/o usuarios que están accediendo a subconjuntos parciales de los mismos (vistas).

El modelo Relacional

- **Flexibilidad:** Ofrecer a cada usuario los datos de la forma más adecuada a la correspondiente aplicación.

- **Uniformidad:** Las estructuras lógicas de los datos presentan un aspecto uniforme (tablas), lo que facilita la concepción y manipulación de la BD por parte de los usuarios

- **Sencillez:** Las características anteriores, así como unos lenguajes de usuario muy sencillos, producen como resultado que el modelo relacional (MR) sea fácil de comprender y de utilizar por parte del usuario final

Características

Los avances más importantes que el MD Relacional incorpora respecto a los MD anteriores fueron:

- **Sólida fundamentación teórica**: Al estar el modelo definido con rigor matemático, el diseño y la evaluación del mismo puede realizarse por métodos sistemáticos basados en abstracciones.

- **Independencia de la interfaz de usuario:** los lenguajes relacionales, al manipular conjuntos de registros, proporcionan una gran independencia respecto a la forma en la que los datos están almacenados.

Generaciones de DBMS's

La aparición del MR representa un hito en el desarrollo de las BD, ya que ha marcado tres etapas diferentes, conocidas como generaciones de los DBMS's:

- **Prerrelacional (1º generación):** en la cual los DBMS se soportan en los modelos Codasyl (en Red) y Jerárquico
- **Relacional (2º generación):** donde los sistemas relacionales se van aproximando a su madurez y los productos basados en este modelo van desplazando poco a poco a los sistemas de primera generación
- **Postrelacional (3º generación):** aparecen otros MD, en especial los orientados al objeto, intentando abrirse un hueco en el mercado de las bases de datos e integrándose como extensiones en los DBMS's previos de la generación relacional.

DBMS's comerciales

Las ventajas citadas han contribuido a que desde mediados de los años 80, el MR sea utilizado por prácticamente la totalidad de los DBMS comerciales.

- Algunas de las principales empresas informáticas del mundo, son en origen, empresas de DBMS: ORACLE, Sybase, INFORMIX, ...
- Los grandes fabricantes de software tienen "su" DBMS relacional: IBM, DB2, Microsoft SQL Server, ...
- Existen varios DBMS diseñados para PCs y usuarios no expertos: Microsoft Access, etc.
- El tremendo éxito real del MR ha promovido, junto al cambio tecnológico, la aparición de los DBMS Objeto-Relacionales, y fracasan, en general, los DBMS de Objetos puros

El modelo relacional

- El modelo relacional, como todo modelo de datos, tiene que ver con tres aspectos de los datos:
  - la **estructura** de datos,
  - la **integridad** de los datos y
  - la **manipulación** de los mismos.

Componentes estáticas

Componente dinámica

- En los últimos años, se han propuesto algunas extensiones al modelo relacional para capturar mejor el significado de los datos, para disponer de los conceptos de la orientación a objetos y para disponer de capacidad deductiva.

Estructura de los Datos

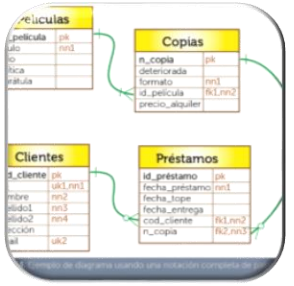
Analizamos los siguientes elementos básicos:

- **Relación o Tabla:** Es la estructura básica del modelo relacional. Se representa mediante una **tabla**.
- **Atributo**: Representa las propiedades de la relación. Se representa mediante una **columna**.
- **Dominio:** Es el conjunto válido de **valores** que toma un atributo.
- **Tupla o Registro:** Es una ocurrencia de la relación. Se representa mediante una **fila**

La relación...

El modelo relacional se basa en el **concepto matemático de relación**, **que gráficamente se representa mediante una tabla**. **Codd**, que era un experto, utilizó terminología perteneciente a las matemáticas**: de la teoría de conjuntos y de la lógica de predicados.**

En el modelo relacional, las ***relaciones o tablas*** se utilizan para almacenar información sobre los objetos que se representan en la base de datos.



Una relación es una tabla con filas y columnas.

Las relaciones o tablas

- Un DBMS **sólo necesita que el usuario pueda percibir la base de datos como un** conjunto de tablas.
- **No se aplica a la estructura física de la base de datos,** que se puede implementar con distintas estructuras de almacenamiento.

Los atributos

- Una relación se representa gráficamente como una tabla bidimensional en la que las filas corresponden a registros individua-les y las columnas corresponden a los campos o atributos de esos registros.

- Los atributos almacenan una propiedad o característica de la ENTIDAD y pueden aparecer en la relación en cualquier orden.

- Es el nombre de una columna de una relación.

Un dominio...

Un dominio es el conjunto de valores legales de uno o varios atributos.

- Los dominios constituyen una poderosa característica del modelo relacional. Cada atributo de una base de datos relacional se define sobre un dominio, pudiendo haber varios atributos definidos sobre el mismo dominio.

- Un dominio D es un conjunto de valores atómicos. Por **atómico** se entiende que cada valor del dominio es indivisible en lo tocante al modelo relacional

Un dominio...

Un dominio es el conjunto de valores legales de uno o varios atributos.

- El concepto de dominio es importante porque permite que el usuario **defina**, en un lugar común, el **significado y la fuente de los valores** que los atributos pueden tomar.

- ***Esto hace que haya más información disponible para el sistema cuando éste va a ejecutar una operación relacional***, de modo que las operaciones que son semánticamente incorrectas, se pueden evitar.

Algunos Conceptos

Tupla: Los elementos de una relación son las tuplas o filas de la tabla. Las tuplas de una relación no siguen ningún orden. ***Una tupla es una fila de una relación.*** Conceptualmente constituye un: **individuo**, **ejemplar u ocurrencia.**

Grado: **El grado de una relación es el número de atributos que contiene.** El grado de una relación no cambia con frecuencia.

Cardinalidad: **La cardinalidad de una relación es el número de tuplas que contiene.** Ya que en las relaciones se van insertando y borrando tuplas a menudo, la cardinalidad de las mismas varía constantemente

BD Relacional es un conjunto de relaciones normalizadas

Autor

Nombre	Nacionalidad	Institución
Date, C.J.	Norteamericana	Relational Institute
Codd, E.F	Norteamericana	Relational Institute
Ceri, S.	Italiana	Politecnica de Milan
De Miguel, A.	Española	UC3M

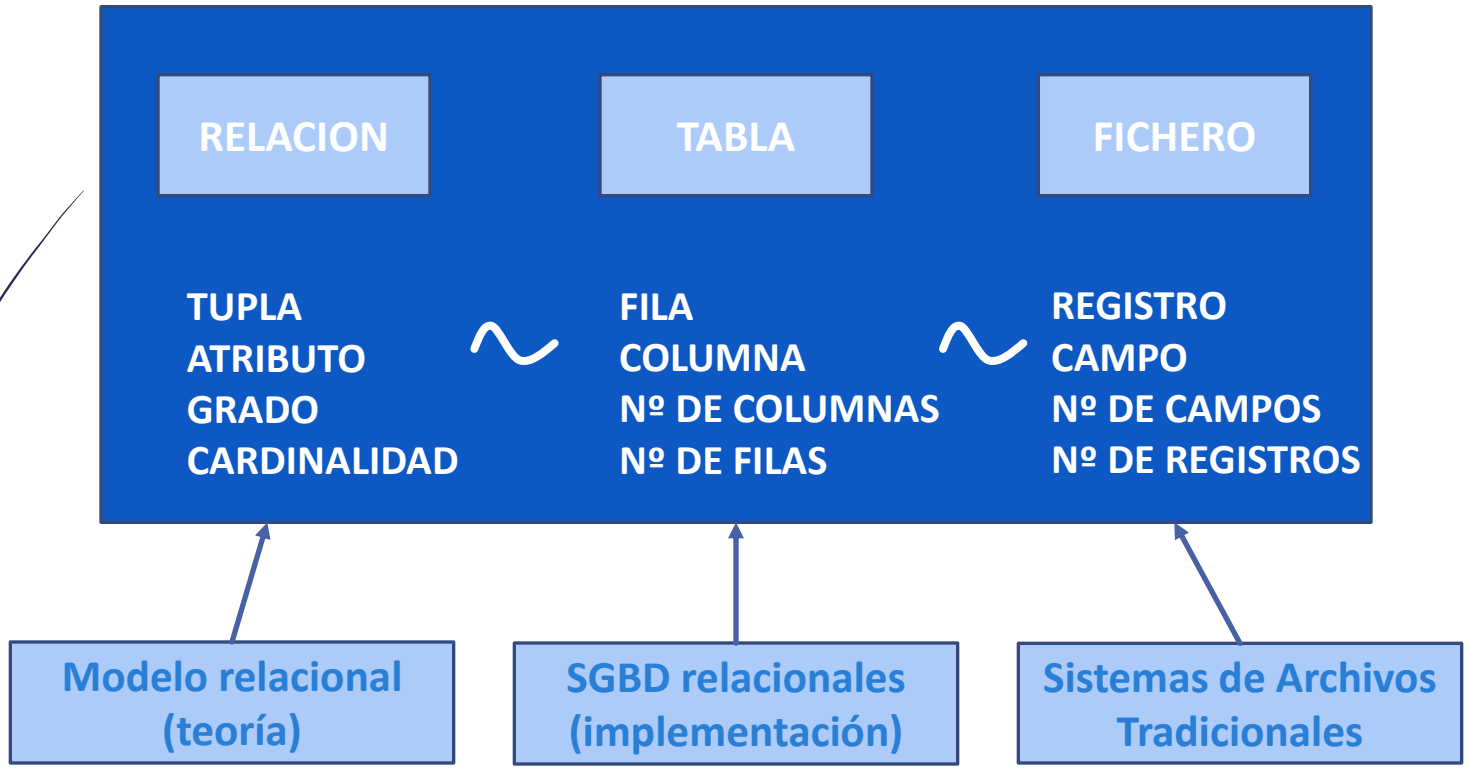
Atributos

Tuplas

Cardinalidad 4

Grado 3

En relación a la terminología



Propiedades de las Relaciones

- Cada relación tiene un nombre y éste es distinto del nombre de todas las demás.
- Los valores de los atributos son atómicos: en cada tupla, cada atributo toma un solo valor. Se dice que las relaciones están normalizadas.
- En una relación no hay dos atributos que se llamen igual.
- El orden de los atributos no importa: los atributos no están ordenados, lo importante es mantener la correspondencia entre atributos y valores
- Cada tupla es distinta de las demás: no hay tuplas duplicadas
- El orden de las tuplas no importa: las tuplas no están ordenadas

Revisión de Conceptos

Claves Candidatas: El atributo o conjunto de atributos K de la relación R es una clave candidata para K si y sólo si satisface las siguientes propiedades:

Unicidad: nunca hay dos tuplas en la relación R con el mismo valor de K.

Irreductibilidad: ningún subconjunto de K tiene la propiedad de unicidad, es decir, no se pueden eliminar componentes de K sin destruir la unicidad

Clave Primaria (PK): La clave primaria de un relación es aquella clave candidata que se escoge para identificar sus tuplas de modo único. Ya que una relación no tiene tuplas duplicadas, siempre hay una clave candidata y, por lo tanto, la relación siempre tiene clave primaria.

Clave Ajena

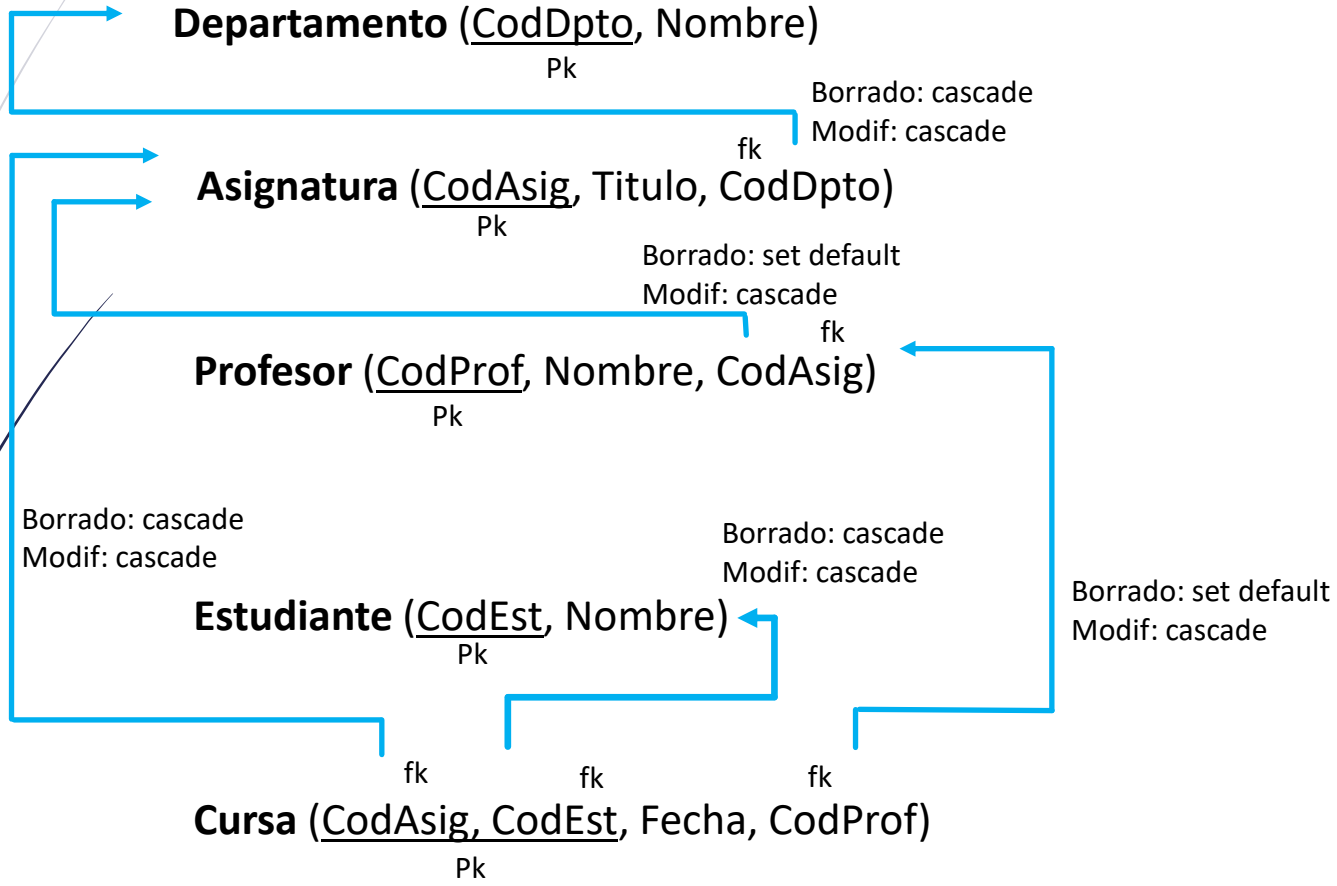
Una base de datos relacional es una colección de relaciones, pero es necesario, además ***ASOCIAR unas relaciones con otras***, de modo de completar su noción básica de "conjunto de datos relacionados".

Se denomina **Clave Ajena** de una relación **R2** a un conjunto no vacío de atributos cuyos valores han de coincidir con los valores de una clave primaria de una relación **R1.**

Una clave ajena es un atributo o un conjunto de atributos de una relación cuyos valores **coinciden con los valores de la clave primaria de alguna otra relación** (puede ser la misma).

Las claves ajenas representan relaciones entre datos.

Clave Ajena


Departamento (CodDpto, Nombre)
Pk
Borrado: cascade
Modif: cascade
fk
Asignatura (CodAsig, Titulo, CodDpto)
Pk
Borrado: set default
Modif: cascade
fk
Profesor (CodProf, Nombre, CodAsig)
Pk
Borrado: cascade
Modif: cascade
Borrado: cascade
Modif: cascade
Borrado: set default
Modif: cascade
Estudiante (CodEst, Nombre)
Pk
fk
fk
fk
Cursa (CodAsig, CodEst, Fecha, CodProf)
Pk

Clave Ajena

equipo		
id_equipo	int	PK
nombre	Varchar(50)	not null

jugador		
id_jugador	int	PK
nombre	Varchar(50)	not null
id_equipo	int	Fk, not null

jugador		
Id_jugador	nombre	Id_equipo
1	Messi, L	1
2	Rodrygo	2
3	Suarez, L	1
4	Mbappé, K	2
5	De Paul, R	3
6	Alvarez, J	3
7	Vinicius, J	2

equipo	
Id_equipo	nombre
1	Inter Miami
2	Real Madrid
3	Atlético Madrid

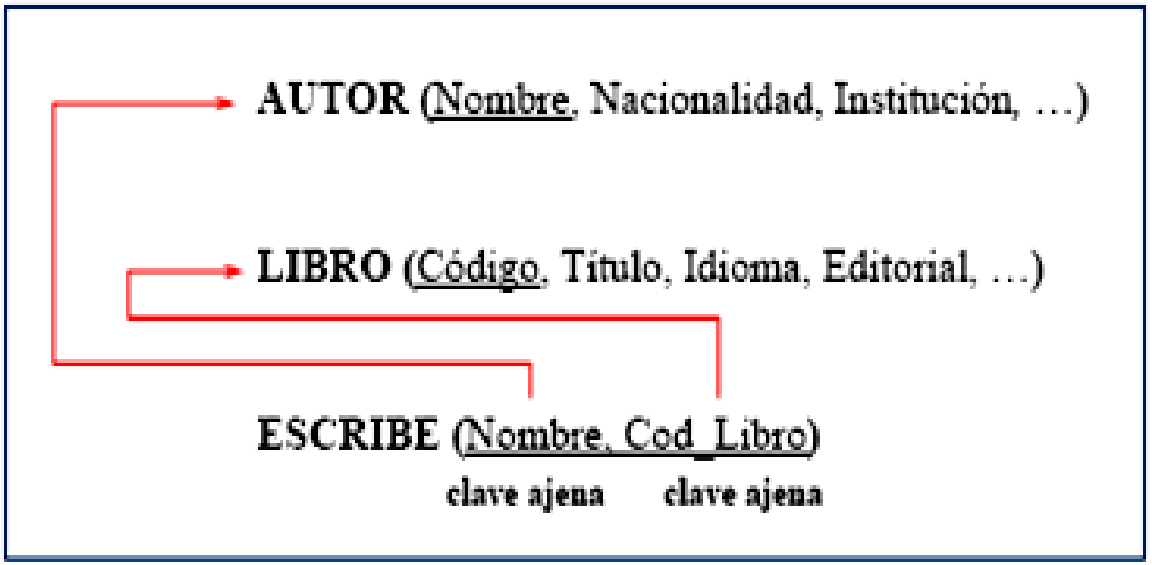
Clave Foranea - Foreign key

Clave Primaria - Primary Key

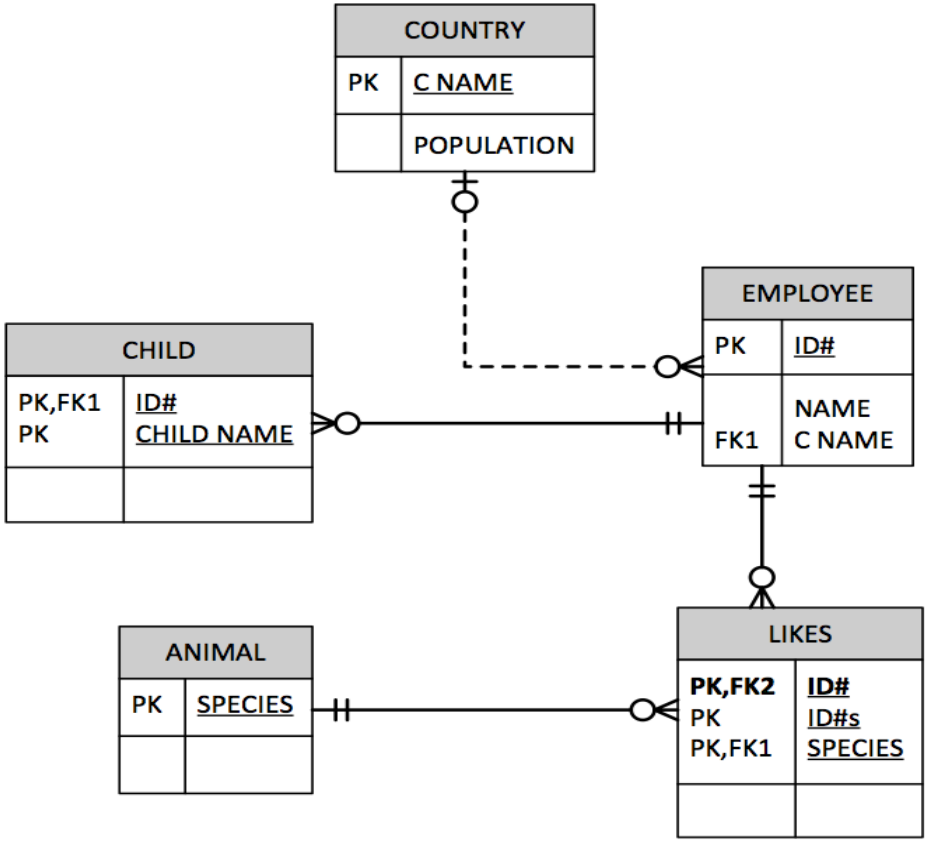
Esquema de una BD Relacional

- Una base de datos relacional es un conjunto de relaciones normalizadas
- Un esquema de base de datos relacional S es un conjunto de esquemas de relaciones S = R1, R2, ..., Rn y un conjunto de restricciones de integridad.
- Para representar el esquema de una base de datos relacional se debe dar el nombre de sus relaciones, los atributos de éstas, los dominios sobre los que se definen estos atributos, las claves primarias y las claves ajenas.

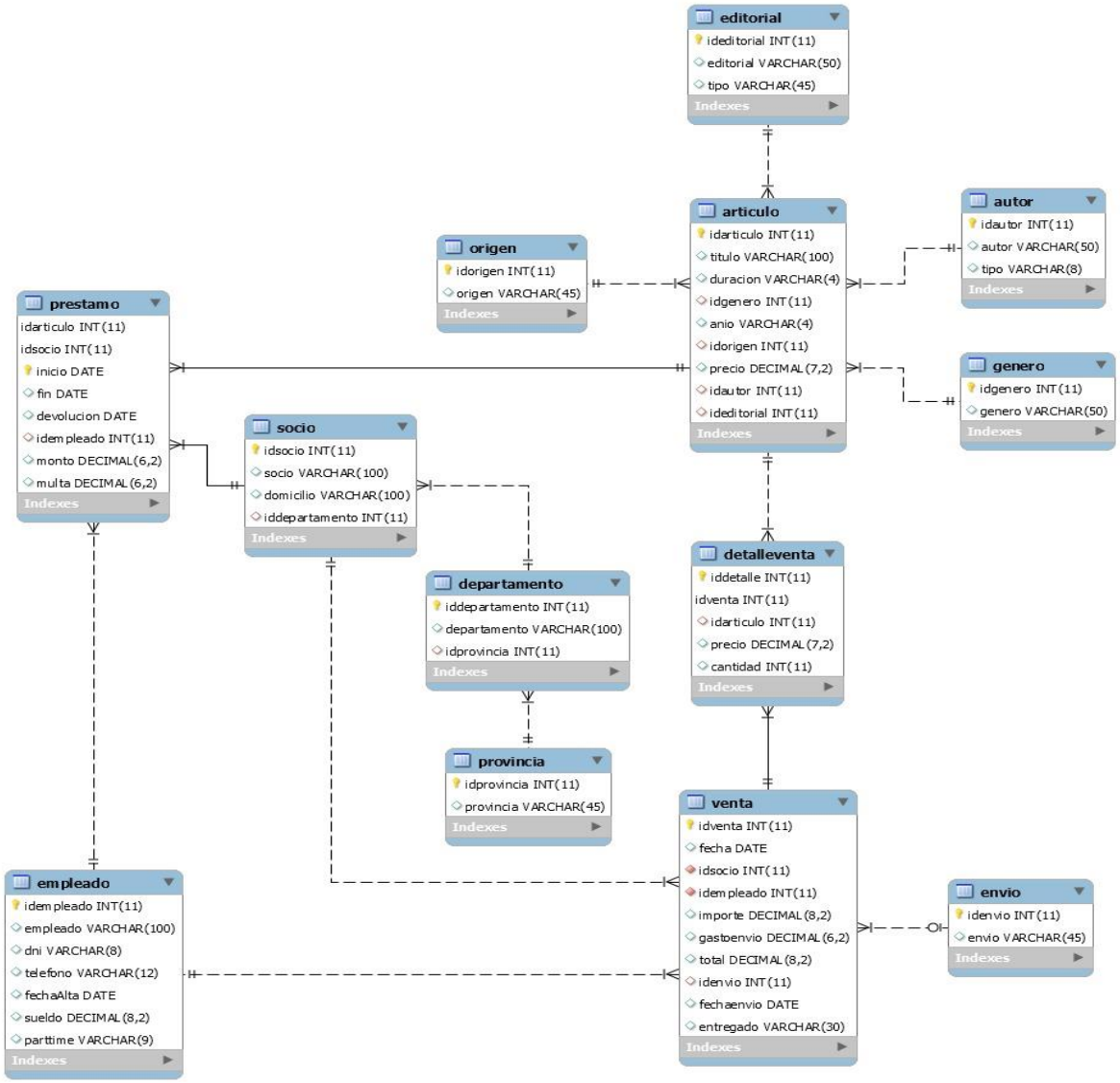
Notación


AUTOR (Nombre, Nacionalidad, Institución, …)
LIBRO (Código, Título, Idioma, Editorial, …)
ESCRIBE (Nombre, Cod_Libro)
clave ajena
clave ajena

Notación


COUNTRY
PK
C NAME
POPULATION
EMPLOYEE
PK
ID#
NAME
FK1
C NAME
CHILD
PK,FK1
PK
ID#
CHILD NAME
ANIMAL
PK
SPECIES
LIKES
PK,FK2
PK
PK,FK1
ID#
ID#s
SPECIES

Notación


editorial
ideditorial INT(11)
editorial VARCHAR(50)
tipo VARCHAR(45)
Indexes
articulo
idarticulo INT(11)
titulo VARCHAR(100)
duracion VARCHAR(4)
idgenero INT(11)
anio VARCHAR(4)
idorigen INT(11)
precio DECIMAL(7,2)
idautor INT(11)
ideditorial INT(11)
Indexes
autor
idautor INT(11)
autor VARCHAR(50)
tipo VARCHAR(8)
Indexes
origen
idorigen INT(11)
origen VARCHAR(45)
Indexes
genero
idgenero INT(11)
genero VARCHAR(50)
Indexes
prestamo
idarticulo INT(11)
idsocio INT(11)
inicio DATE
fin DATE
devolucion DATE
idempleado INT(11)
monto DECIMAL(6,2)
multa DECIMAL(6,2)
Indexes
socio
idsocio INT(11)
socio VARCHAR(100)
domicilio VARCHAR(100)
iddepartamento INT(11)
Indexes
detalleventa
iddetalle INT(11)
idventa INT(11)
idarticulo INT(11)
precio DECIMAL(7,2)
cantidad INT(11)
Indexes
departamento
iddepartamento INT(11)
departamento VARCHAR(100)
idprovincia INT(11)
Indexes
provincia
idprovincia INT(11)
provincia VARCHAR(45)
Indexes
venta
idventa INT(11)
fecha DATE
idsocio INT(11)
idempleado INT(11)
importe DECIMAL(8,2)
gastoenvio DECIMAL(6,2)
total DECIMAL(8,2)
idenvio INT(11)
fechaenvio DATE
entregado VARCHAR(30)
Indexes
envio
idenvio INT(11)
envio VARCHAR(45)
Indexes
empleado
idempleado INT(11)
empleado VARCHAR(100)
dni VARCHAR(8)
telefono VARCHAR(12)
fechaAlta DATE
sueldo DECIMAL(8,2)
parttime VARCHAR(9)
Indexes

Restricciones al Modelo

Reglas de integridad

Una vez definida la estructura de datos del modelo relacional, pasamos a estudiar las reglas de integridad que los datos almacenados en dicha estructura deben cumplir para garantizar que son correctos.

- Las **RESTRICCIONES INHERENTES** vienen impuestas por el propio Modelo de Datos.

    1. Intrínsecas
    2. Integridad de entidades

- Las **RESTRICCIONES SEMANTICAS**.

Restricciones Inherentes Intrínsecas

En el caso del MR, una relación tiene unas propiedades intrínsecas y que se derivan de la misma **definición matemática de relación**, ya que, al ser un conjunto:

No puede haber dos tuplas iguales, obligatoriedad de la clave primaria

- El orden de las tuplas no es significativo.
- El orden de los atributos no es significativo

Cada atributo sólo puede tomar un **único valor del dominio subyacente**; no se admiten grupos repetitivos (ni otro tipo de estructuras) como valores de los atributos de una tupla

Restricciones Inherentes de Integridad de Entidades

"Ningún atributo que forme parte de la clave primaria de una relación puede tomar un valor nulo"; es decir, un valor desconocido o inexistente.

Nulos → Cuando en una tupla un atributo es desconocido, se dice que es *nulo*.

Un **nulo** no representa el valor cero ni la cadena vacía, éstos son valores que tienen significado.

El **nulo implica ausencia de información**, bien porque al insertar la tupla se desconocía el valor del atributo, o bien porque para dicha tupla el atributo no tiene sentido

Restricciones Semánticas

- Son definidas por el usuario.
- El modelo ofrece a los usuarios/diseñadores facilidades para que puedan reflejar en el esquema, lo más fielmente posible, la semántica del mundo real.
- Los tipos de restricciones semánticas permitidos en el MR son:

  Clave Primaria (**PRIMARY KEY**),
  Unicidad (**UNIQUE**),
  Obligatoriedad (**NOT NULL**),
  Integridad Referencial (**FOREIGN KEY**),
  Verificación (**CHECK**)

Restricciones Semánticas

1. **(PRIMARY KEY):** Permite declarar un atributo o un conjunto de atributos como **clave primaria** de una relación.
   - => sus valores no se podrán repetir ni se admitirán los nulos (o valores “ausentes”).
   - Debemos distinguir entre la restricción inherente de obligatoriedad de la clave primaria y la restricción semántica que le permite al usuario indicar qué atributos forman parte de la clave primaria.

2. **(UNIQUE):**
   - Los valores de un conjunto de atributos (uno o más) no pueden repetirse en una relación.
   - Esta restricción permite la definición de **claves candidatas (alternativas).**

Restricciones Semánticas

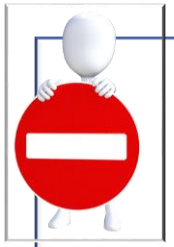
3. **Obligatoriedad (NOT NULL):** El conjunto de atributos no admite valores nulos.
4. **Integridad Referencial (FOREING KEY):** Si una relación R2 (relación que referencia) tiene un descriptor (subconjunto de atributos) CA que referencia a una clave candidata CC de la relación R1 (relación referenciada), todo valor de dicho descriptor CA debe coincidir con un valor de CC o ser nulo.
   - La condición puede expresarse como R2.CA = R1.CC
   - El descriptor CA es, por tanto, una **clave ajena** de la relación R2.
   - Las relaciones R1 y R2 no son necesariamente distintas.
   - La clave ajena puede ser también parte (o la totalidad) de la clave primaria de R2.
   - CA puede admitir nulos o tener restricción de obligatoriedad (NOT NULL).

Integridad Referencial (Modos de Borrado y Modificación)

Además de definir las claves ajenas, hay que determinar las consecuencias que pueden tener ciertas operaciones (borrado y modificación) realizadas sobre tuplas de la relación referenciada:

NO ACTION: rechazar la operación de borrado o modificación. (Restringir)

CASCADE: Propagar la modificación (o borrado) de las tuplas de la tabla que referencia.



SET NULL: poner valor nulo en la clave ajena de la tabla que referencia.

SET DEFAULT: poner un valor por defecto en la clave ajena de la tabla que referencia.

Restricciones Semánticas

5 Restricciones Semánticas de Rechazo: El usuario formula una condición mediante un predicado definido sobre un conjunto de atributos, tuplas o dominios, que debe ser verificado en toda operación de actualización para que el nuevo estado constituya una ocurrencia válida del esquema. **Verificación** (**CHECK**):

Comprueba, en toda operación de actualización, si el predicado es **cierto o falso** y, en el segundo caso, rechaza la operación. La restricción de verificación se define sobre **un único elemento** (dentro de un CREATE TABLE) y ***puede o no*** tener nombre.

CHECK N_HORAS > 30 en CURSO_DOCTORADO

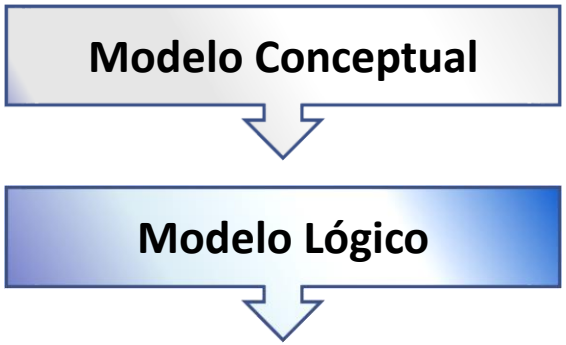
Reglas de Negocio

- Además de las dos reglas de integridad anteriores, los usuarios o los administradores de la base de datos pueden imponer ciertas restricciones específicas sobre los datos, denominadas *reglas de negocio*.

- ***Por ejemplo**, si en una oficina de la empresa inmobiliaria sólo puede haber hasta veinte empleados, el DBMS debe dar la posibilidad al usuario de definir una regla al respecto y debe hacerla respetar. En este caso, no debería permitir dar de alta un empleado en una oficina que ya tiene los veinte permitidos.*

- **Actualmente todo DBMS relacional permite definir este tipo de restricciones.**

Del modelo conceptual al modelo lógico

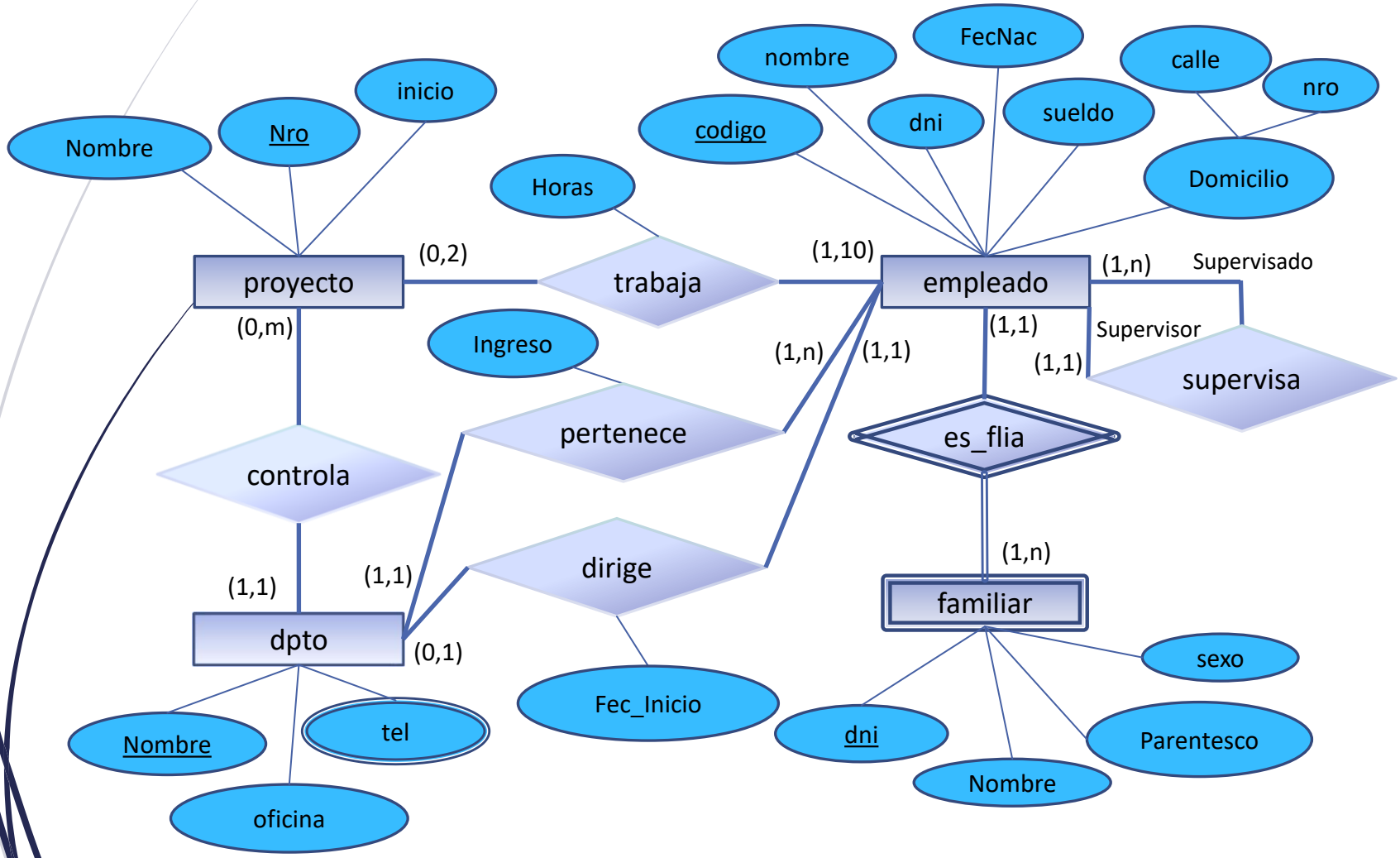


Transformación de ER a MR

- Es posible derivar (obtener) un esquema de bases de datos relacional *a partir* de un esquema conceptual creado empleando el modelo ERE.

- Método de transformación utilizaremos **Algoritmo de Siete Pasos**, puesto que *una base de datos que se ajusta a un diagrama ER puede representarse por medio de una colección de tablas*.

Pasos para la Transformación

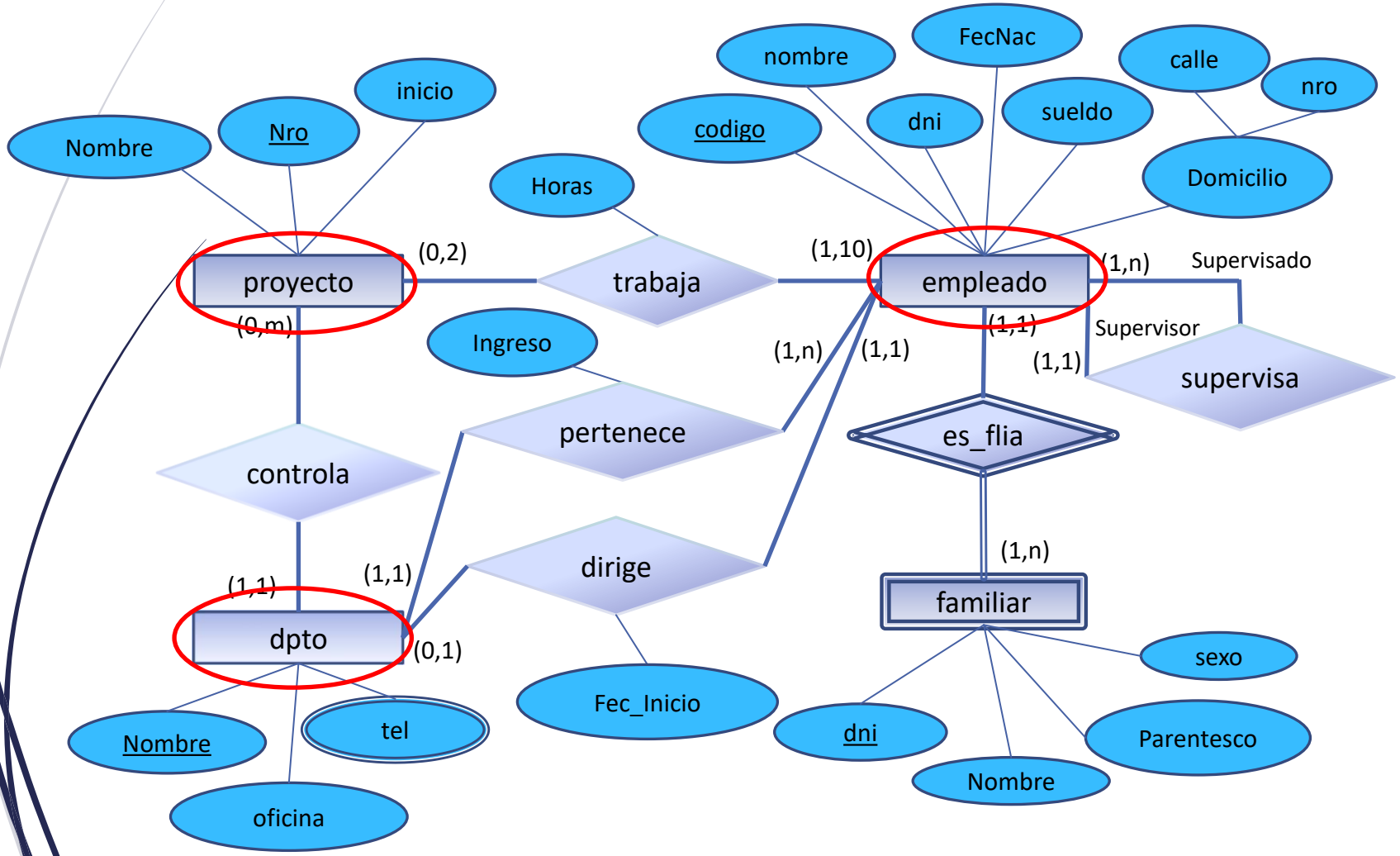


Pasos para la Transformación

1. Representación de conjuntos de Entidades **Fuertes**
2. Conversión de Entidades **Débiles**.
3. Representación de conjunto de relaciones o vínculos **1:1**
4. Representación de conjunto de relaciones o vínculos **1:N**
5. Representación de conjunto de relaciones o vínculos **M:N**
6. Conversión de atributos **multivaluados**.
7. Representación de conjunto de relaciones o vínculos n-arios R, con n>2. (ADICIONAL)

Pasos para la Transformación

Paso 1: Representación de conjuntos de Entidades **Fuertes**



Pasos para la Transformación

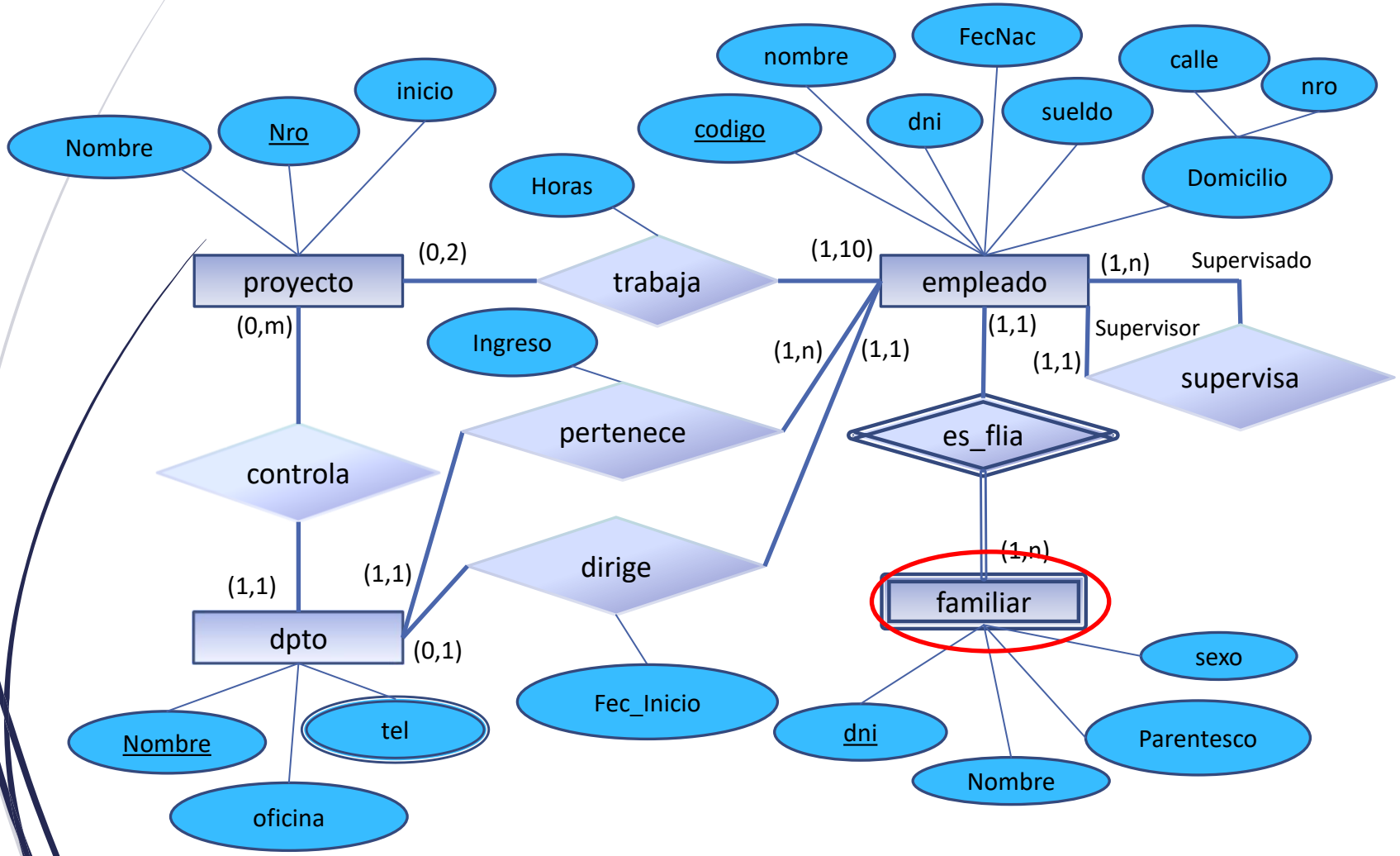
Paso 1:

- Por cada entidad E del esquema E-R, se crea una relación R que contenga todos los atributos simples de E.
- Se incluyen sólo los atributos simples de un atributo compuesto.
- Luego se elige uno de los atributos como clave primaria**.**
- En el ejemplo los tipos de entidades a representar en este primer paso son: **empleado, dpto y proyecto.**

empleado (codigo, dni, nombre, fecNac, sueldo, calle, nro)
proyecto (Nro, nombre, inicio)
dpto (nombre, oficina)

Pasos para la Transformación

Paso 2: Conversión de Entidades **Débiles**



Pasos para la Transformación

Paso 2:

- Por cada entidad débil D del esquema ER con tipo de entidades propietarias E, se crea una relación o TABLA R y se incluyen todos los atributos simples (o componentes simples de los atributos compuestos) de D como atributos de R.

- Además se incluyen como atributos de clave externa de R los atributos de clave primaria de la relación o relaciones que corresponden al tipo de entidades propietarias.

Pasos para la Transformación

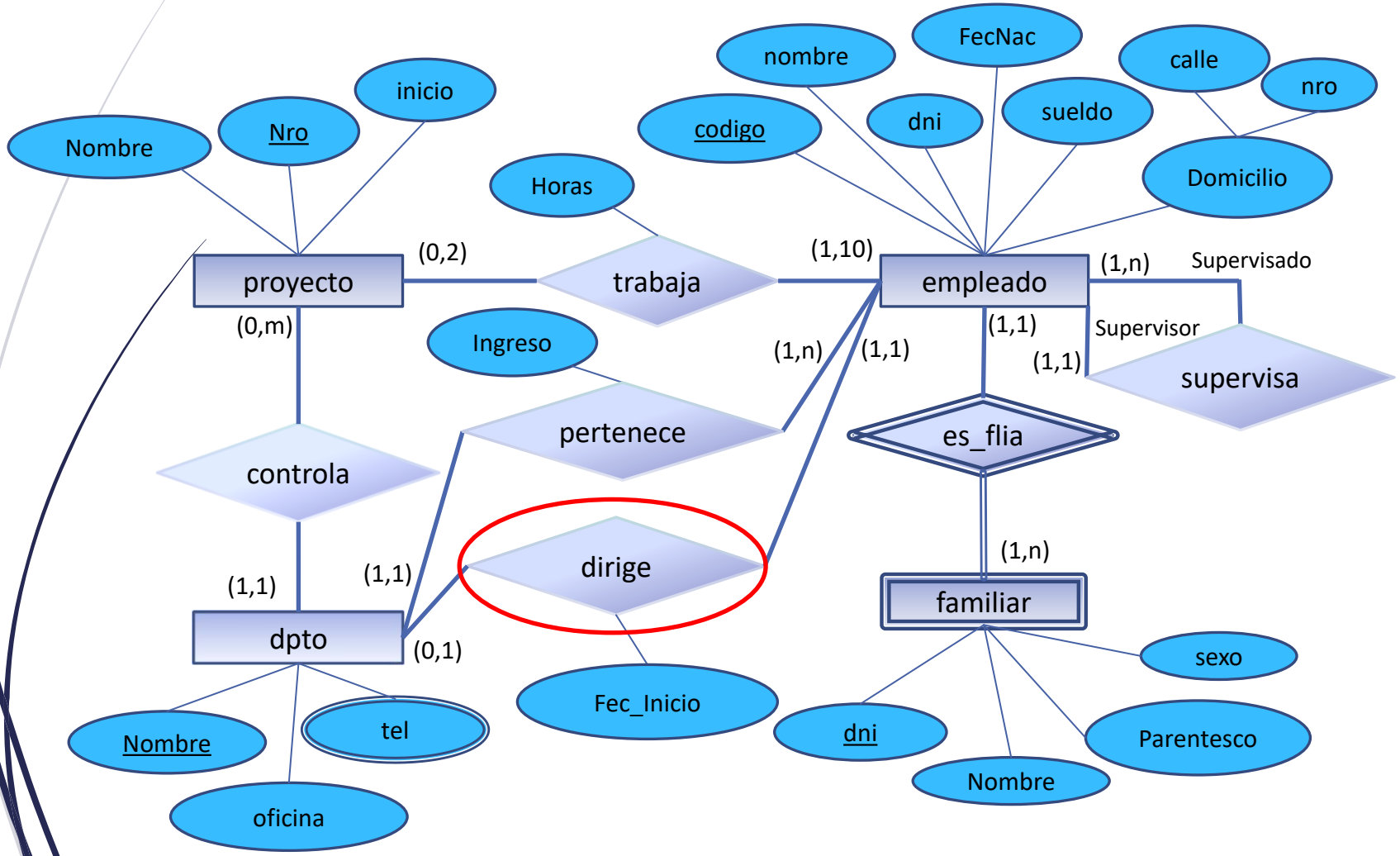
Paso 2:

La clave primaria de R es la combinación de las claves primarias de las propietarias y la clave parcial de D, si existe.
En el ejemplo se crea entonces la relación Dependiente que corresponde al tipo de entidades débil Dependiente. La relación o tipo de vínculo Dependiente_De no se transformará en los pasos correspondientes a conversión de tipos de vínculos puesto que es una ***relación mandatoria*****.**

empleado (codigo, dni, nombre, fecNac, sueldo, calle, nro)
proyecto (Nro, nombre, inicio)
dpto (nombre, oficina)
familiar (codigoE, dni, nombre, sexo parentesco)

Pasos para la Transformación

Paso 3: Representación de conjunto de relaciones o vínculos **1:1**



Pasos para la Transformación

Paso 3:

- Por cada relación binaria 1:1 **R** del esquema E-R, se identifican las relaciones **S** y **T** que corresponden a las entidades que participan en **R**.
- Se escoge una de las relaciones (supongamos **S**) y se incluye como clave externa en **S** la clave primaria de **T**.
- Es mejor elegir una entidad con participación Total en R en el papel de S. Se incluyen todos los atributos simples (o componentes simples de atributos compuestos) de la relación 1:1 R como atributos de S.

Pasos para la Transformación

Paso 3:

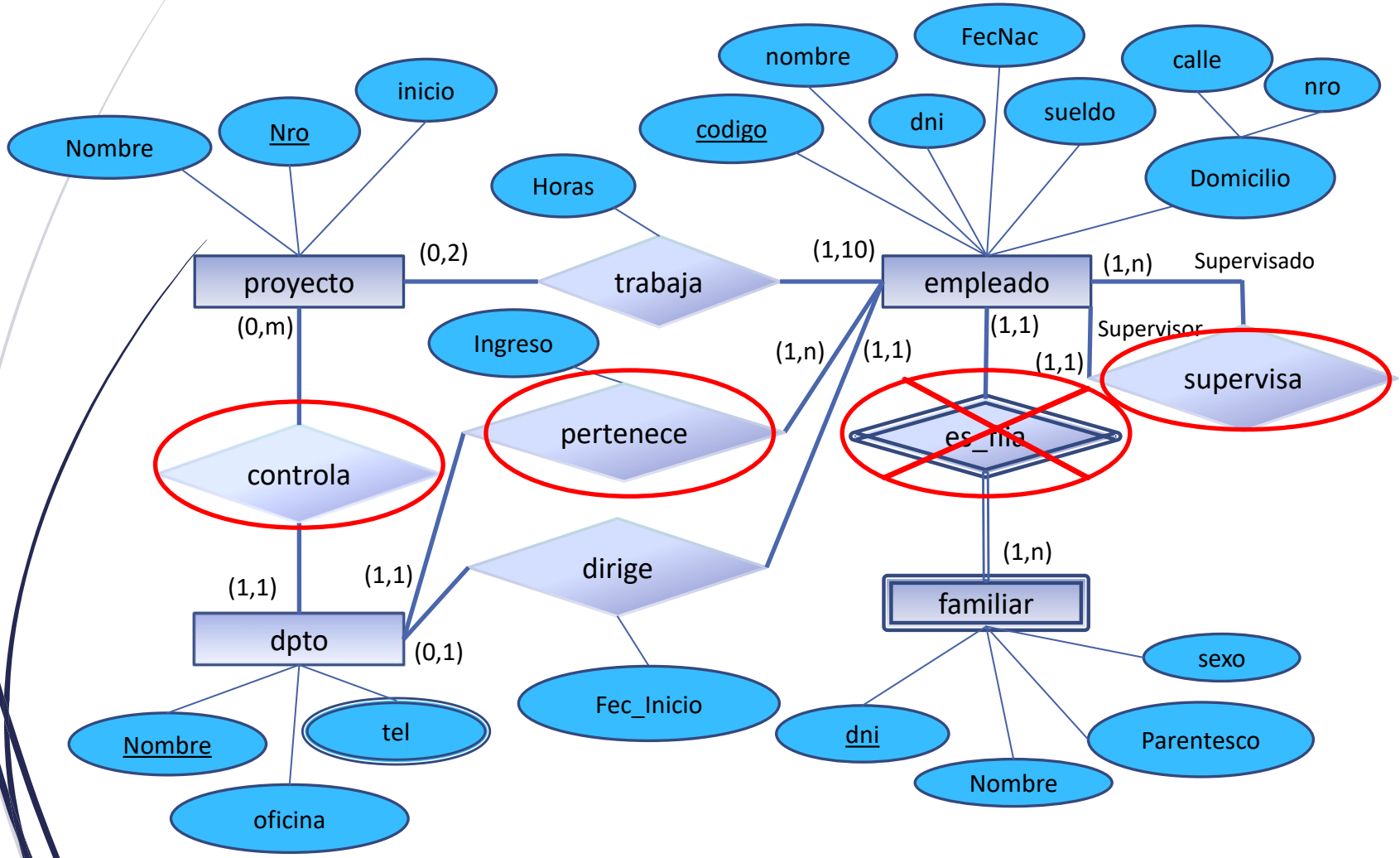
En el ejemplo la relación 1:1 **dirige** de la figura será el que se transformará, eligiendo **dpto** para desempeñar el papel de S, debido a que su participación en Dirige es TOTAL (todo dpto tiene un "gerente"). Se incluirá la clave primaria de la relación **empleado** como clave externa de la relación **dpto** y se cambia el nombre a codigoE. También se incluye el atributo simple fec_Inicio de **dirige** en la relación **dpto** y cambiamos su nombre a fec_InicioE.

empleado (codigo, dni, nombre, fecNac, sueldo, calle, nro)
proyecto (Nro, nombre, inicio)
dpto (nombre, oficina, codigoE, fec_InicioE)
familiar (codigoE, dni, nombre, sexo parentesco)

Campos agregados

Pasos para la Transformación

Paso 4: Representación de conjunto de relaciones o vínculos **1:N**



Pasos para la Transformación

Paso 4:

- Por cada relación (no débil) de 1:N R, se identifica la relación S que representa la entidad participante de lado :N de la relación.
- Se incluye como clave externa en S la clave primaria de la relación T que representa a la otra entidad que participa en R; *(la razón es que cada ejemplar de entidad del lado N está relacionado con un máximo de un ejemplar de entidad del lado 1)*:
- **Se incluyen todos los atributos simples (o componentes simples de los atributos compuestos) de la relación 1:N como atributos de S.**

Pasos para la Transformación

Paso 4: En este paso se deben transformar las relaciones **pertenece**, **controla** Y **supervisa**.

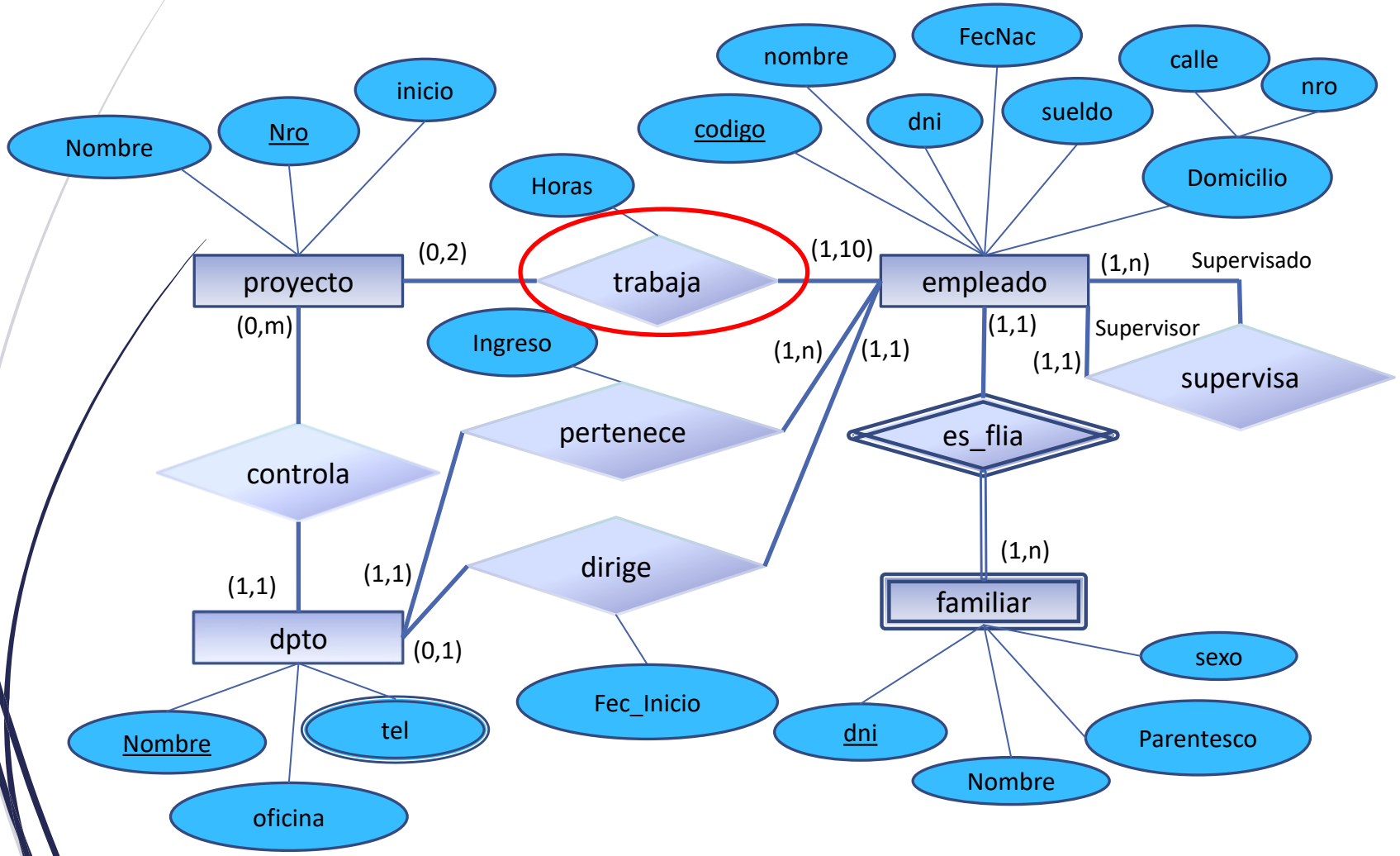
En el caso de **pertenece** se incluye la clave primaria de **dpto (nombre)** como clave ajena en **empleado**. En el caso de la relación **controla** se agrega el atributo de clave externa de **dpto** a proyecto.
En el caso de **supervisa**, se incluye la clave primaria de **empleado** como clave externa en la relación **empleado**

empleado (codigo, dni, nombre, fecNac, sueldo, calle, nro, nomD, inicio codE)
proyecto (Nro, nombre, inicio, nomD)
dpto (nombre, oficina, codigoE, fec_InicioE)
familiar (codigoE, dni, nombre, sexo parentesco)

Campos agregados

Pasos para la Transformación

Paso 5: Representación de conjunto de relaciones o vínculos **M:N**



Pasos para la Transformación

Paso 5:

- Por cada relación binaria M:N , **se crea una nueva TABLA** S para representar R.
- Se incluyen como atributos de clave externa en S las claves primarias de las relaciones que representan las entidades participantes; su combinación constituirá la clave primaria de S.
- También se incluyen todos los atributos simples (o componentes simples de atributos compuestos) del tipo de vínculos M:N como atributos de S.
- No se puede representar una relación M:N con un solo atributo de clave externa en una de las relaciones participantes (como se hizo en el caso de las relaciones 1:1 y 1:N) debido a la razón de cardinalidad M:N.

Pasos para la Transformación

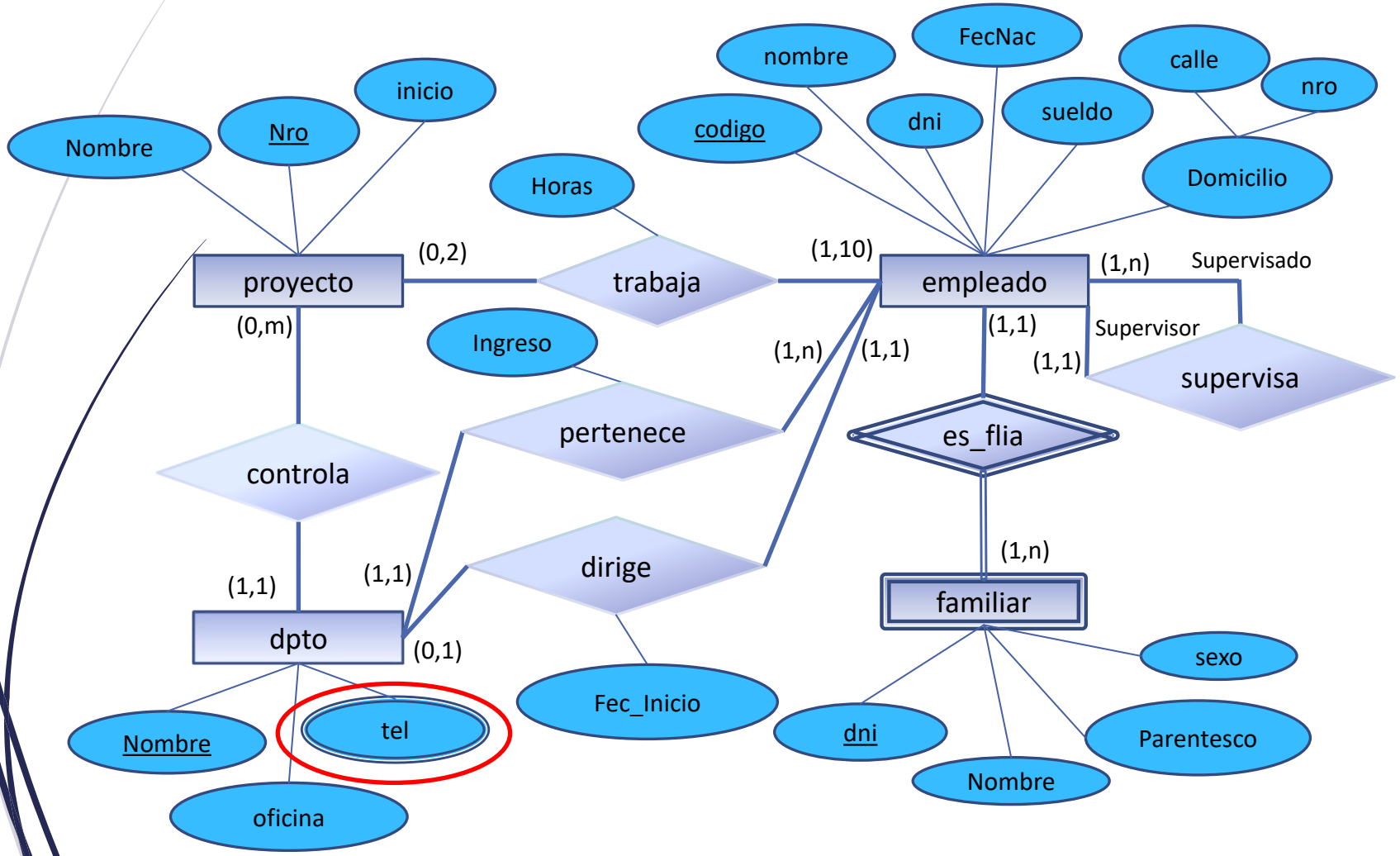
Paso 5:

En el ejemplo, se establece la transformación de la relación **trabaja**, agregando las claves primarias de las relaciones **proyecto** y **empleado** como claves ajenas en **trabaja**. También se incluyen todos los atributos que tiene la relación trabaja, en este caso, horas. La clave primaria de la relación trabaja es la combinación de los atributos de clave externa NSSE, NúmP.

empleado (codigo, dni, nombre, fecNac, sueldo, calle, nro, nomD, inicio codE)
proyecto (Nro, nombre, inicio, nomD)
dpto (nombre, oficina, codigoE, fec_InicioE)
familiar (codigoE, dni, nombre, sexo parentesco)
trabaja (códigoE, nombreD, horas) ← **Nueva Tabla**

Pasos para la Transformación

Paso 6: Conversión de atributos **multivaluados**.



Pasos para la Transformación

Paso 6:

- Por cada atributo multivaluado A se crea una relación nueva R que contiene una atributo correspondiente a A más el atributo de clave primaria K (como clave externa de R) de la relación que representa la entidad o relación que contiene a A como atributo. La clave primaria de R es la combinación de A y K.

- Si el atributo multivaluado es compuesto, se incluyen sus componentes simples.

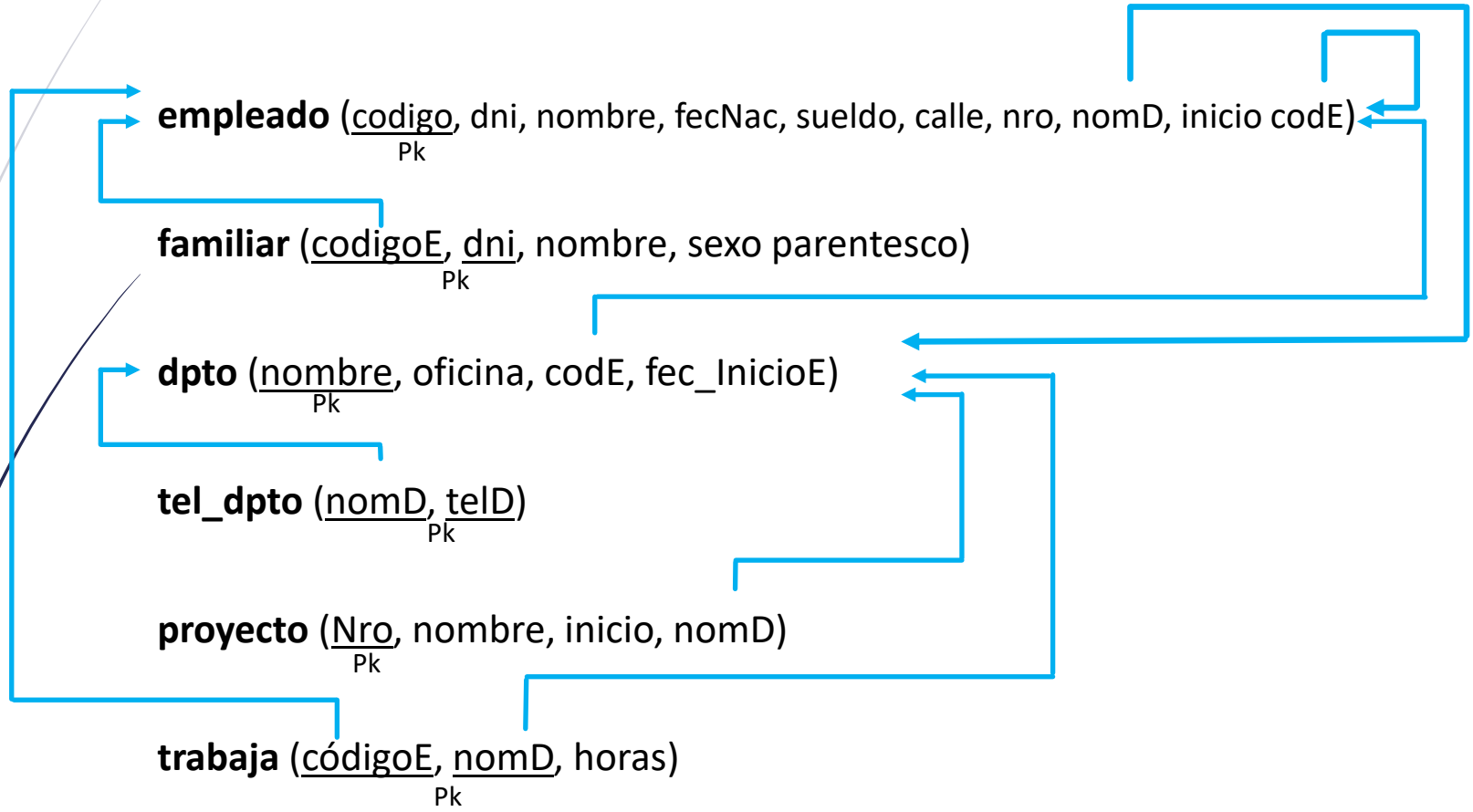
Pasos para la Transformación

Paso 6:

En el ejemplo, se crea la relación tel_dpto. El atributo telD representa el atributo multivaluado tel de dpto, en tanto que nombreD (como clave ajena) representa la clave primaria de la relación dpto.
La clave primaria de tel_dpto es la combinación (nombreD, telD). Habrá una tupla en tel_dpto por cada numero de teléfono de dpto.

empleado (codigo, dni, nombre, fecNac, sueldo, calle, nro, nomD, inicio codE)
proyecto (Nro, nombre, inicio, nomD)
dpto (nombre, oficina, codigoE, fec_InicioE)
familiar (codigoE, dni, nombre, sexo parentesco)
trabaja (códigoE, nombreD, horas)
tel_dpto (nombreD, telD) ← **Nueva Tabla**

Pasos para la Transformación



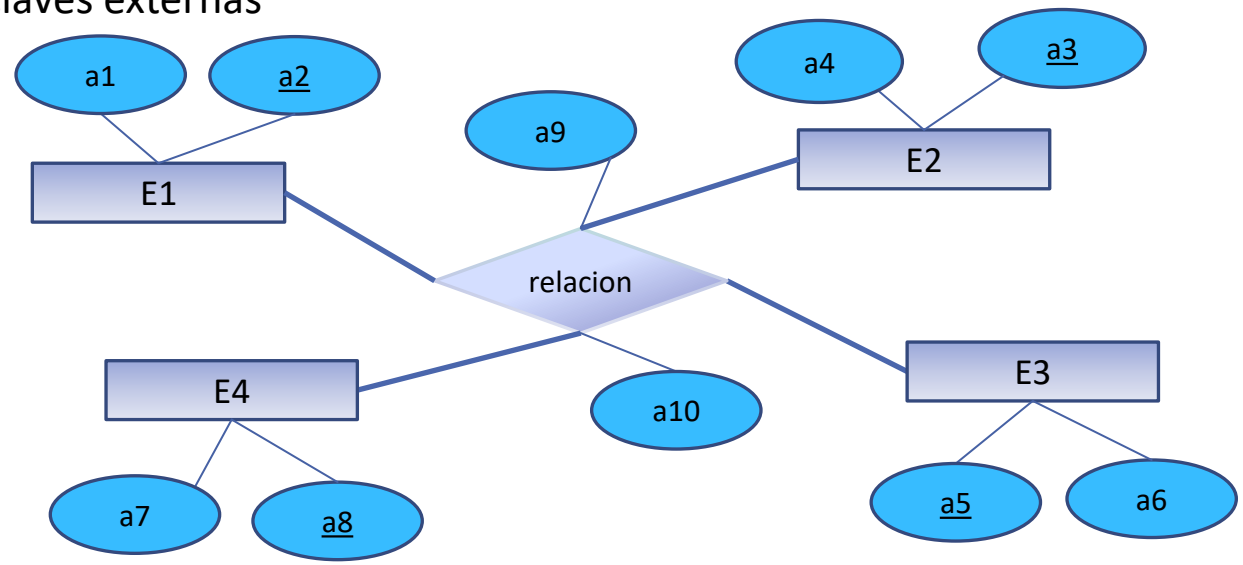
Pasos para la Transformación

Paso 7:

- Por cada relación n-aria R, n>2, se crea una nueva relación S que represente a R. Se incluyen como atributos de clave externa en S las claves primarias de las relaciones que representan las entidades participantes. También se incluyen los atributos simples de la relación n-aria como atributos de S.
- **La clave primaria de S casi siempre es una combinación de todas las claves externas que hacen referencia a las relaciones que representan las entidades participantes.**
- No obstante, si la restricción de participación (min, máx) de uno de los tipos de entidades E que participan en R tiene máx=1, la clave primaria de S podrá ser el único atributo de clave externa que haga referencia a la relación E' que corresponde a E; la razón es que cada una de las entidades e de E participará, en cuando más, un ejemplar de vínculo R, y por tanto, podrá identificar de manera única ese ejemplar.

Pasos para la Transformación

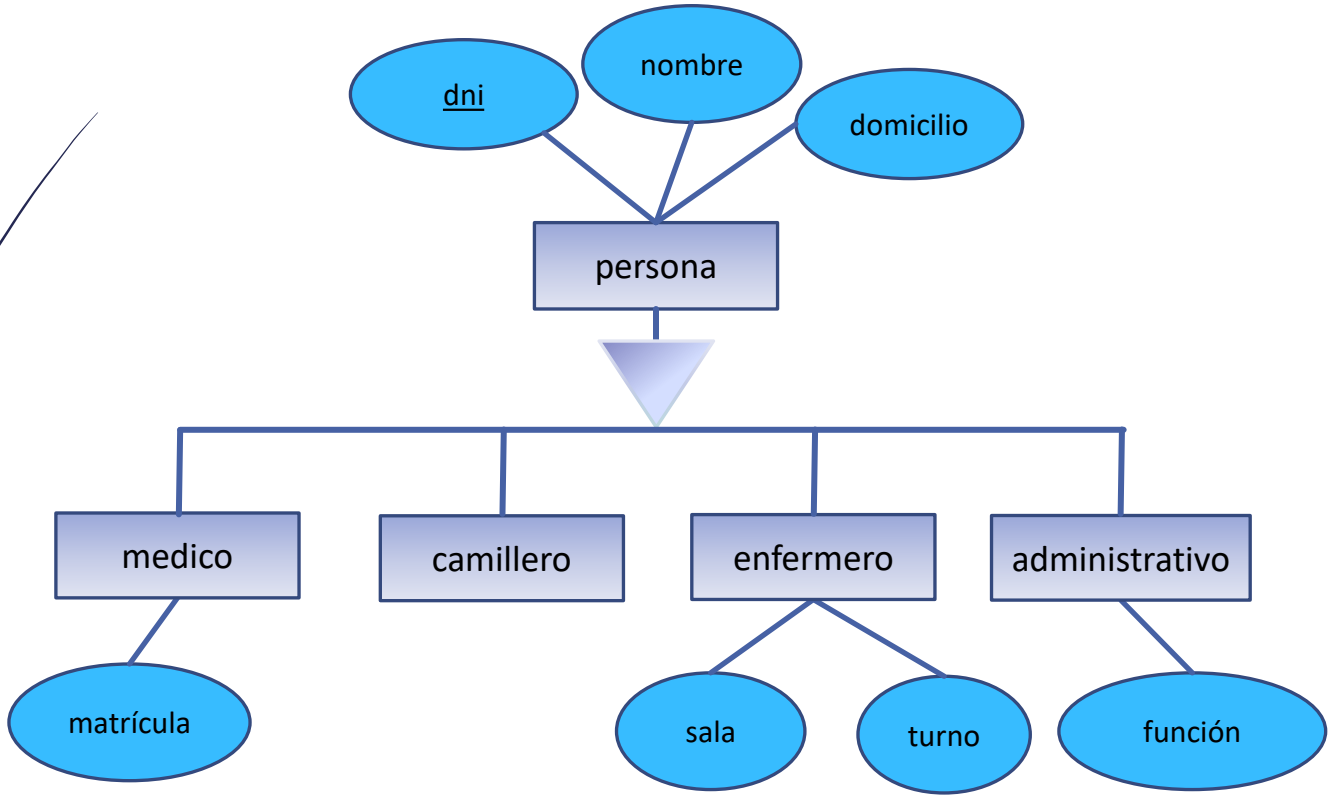
Paso 7: Las relaciones ternarias, cuaternarias y n-arias se transforman en una tabla que contiene los atributos de la relación mas las claves primarias de las entidades . La clave primaria esta formada por las claves externas



relacion (a2, a3, a5,a8 ,a9, a10)

Pasos para la Transformación

Generalización



Transformar Generalizaciones

1. **Pasar sólo el conjunto de entidades superior.**

Todos los atributos de la entidad superior, más cada uno de los atributos de las entidades inferiores, más un atributo distintivo (que indique el tipo de personal).

persona (dni, nombre, domicilio, matricula, función, sala, turno)

Es la menos recomendada, pues promueve atributos nuevos y dificulta la búsqueda

Cuando la instanciación sea de un camillero, los últimos 4 atributos serán **nulos**!!!

Transformar Generalizaciones

2. **Pasar las entidades de todos los niveles a tablas y en las entidades de nivel inferior, se pone la clave del nivel superior.**

   persona (dni, nombre, domicilio)
   médico (dni, matricula)
   enfermero (dni, sala, turno)
   camillero (dni)
   administrativo (dni, función)

3. **Representar sólo los niveles inferiores con los atributos propios, más lo de la entidad superior.**

   médico (dni, nombre, domicilio, matricula)
   enfermero (dni, nombre, domicilio, sala, turno)
   camillero (dni, nombre, domicilio)
   administrativo (dni, nombre, domicilio, función)

Transformar Especializaciones

También existen dos formas de conversión, que ***coinciden con las dos primeras de la generalización***.

- **La tercera no es viable porque pueden quedar filas en la entidad superior que no pertenezcan a ninguna entidad inferior y se perdería información.**

- **1)** *Docente* (dni, nyap, dir, categoría,tipo)

- 2) **Docente** (dni, nyap, dir) -
  Profesor (dni, categoría)

Evolución del Modelo

Cuando el MR triunfó comercialmente, muchos fabricantes que tenían productos "antiguos" **no relacionales** optaron por retocarlos o "***camuflarlos***" añadiéndoles la etiqueta relacional.

- Esto supuso una confusión que **Codd intentó arreglar publicando sus 12+1 reglas**, que indican las características que debe tener un DBMS para ser auténticamente relacional.

Regla 0:

Un DBMS relacional debe emplearse para gestionar en la BD exclusivamente sus facilidades relacionales. → De esta regla genérica se derivan 12 reglas detalladas**.**

Evolución del Modelo

1. Regla de información: Toda la información en la Base de datos es representada en una y solo una forma: valores en columnas de filas de tablas.

2. Regla de acceso garantizado: Cada valor escalar individual puede ser direccionado indicando los nombres de la tabla, columna y valor de la clave primaria de la fila correspondiente.

3. Tratamiento sistemático de valores nulos: El DBMS debe soportar la representación y manipulación de información desconocida y/o no aplicable.

4. Catálogo en línea (diccionario de datos) basado en el modelo relacional.

Evolución del Modelo

5. Sublenguaje de datos completo: El DBMS debe soportar al menos un lenguaje relacional:
   a) con sintaxis lineal.
   b) que pueda ser usado interactivamente o en programas (embebido).
   c) con soporte para operaciones de:
      - definición de datos (p.e. declaración de vistas).
      - manipulación de datos (p.e. recuperación y modificación de tuplas).
      - restricciones de seguridad e integridad.
      - gestión de transacciones.

6. Actualización de vistas: todas las vistas teóricamente actualizables deben poder serlo en la práctica.

Evolución del Modelo

7. Inserción, modificación y borrado de tuplas de alto nivel: todas las operaciones de manipulación de datos deben operar sobre conjuntos de filas (lenguaje de especificación en vez de navegacional).

8. **Independencia física de los datos**: cambios en los métodos de acceso físico o la forma de almacenamiento no deben afectar al acceso lógico a los datos.

9. **Independencia lógica de los datos**: los programas de aplicación no deben ser afectados por cambios en las tablas que preservan la integridad.

Evolución del Modelo

10. **Independencia de la integridad**: Las restricciones de integridad deben estar separadas de los programas, almacenadas en el catálogo de la BD para ser editadas mediante un sub lenguaje de datos.

11. **Independencia de la distribución**: Las aplicaciones no deben verse afectadas al distribuir (dividir entre varias máquinas), o al cambiar la distrib. ya existente de la BD.

12. **Regla de no subversión**: Si el sistema posee un interface de bajo nivel (p.e. mediante llamadas en C), éste no puede subvertir el sistema pudiendo evitar restricciones de seguridad o integridad.

Muchas gracias