

UD 3. MODELO RELACIONAL

Parte 2. Normalización

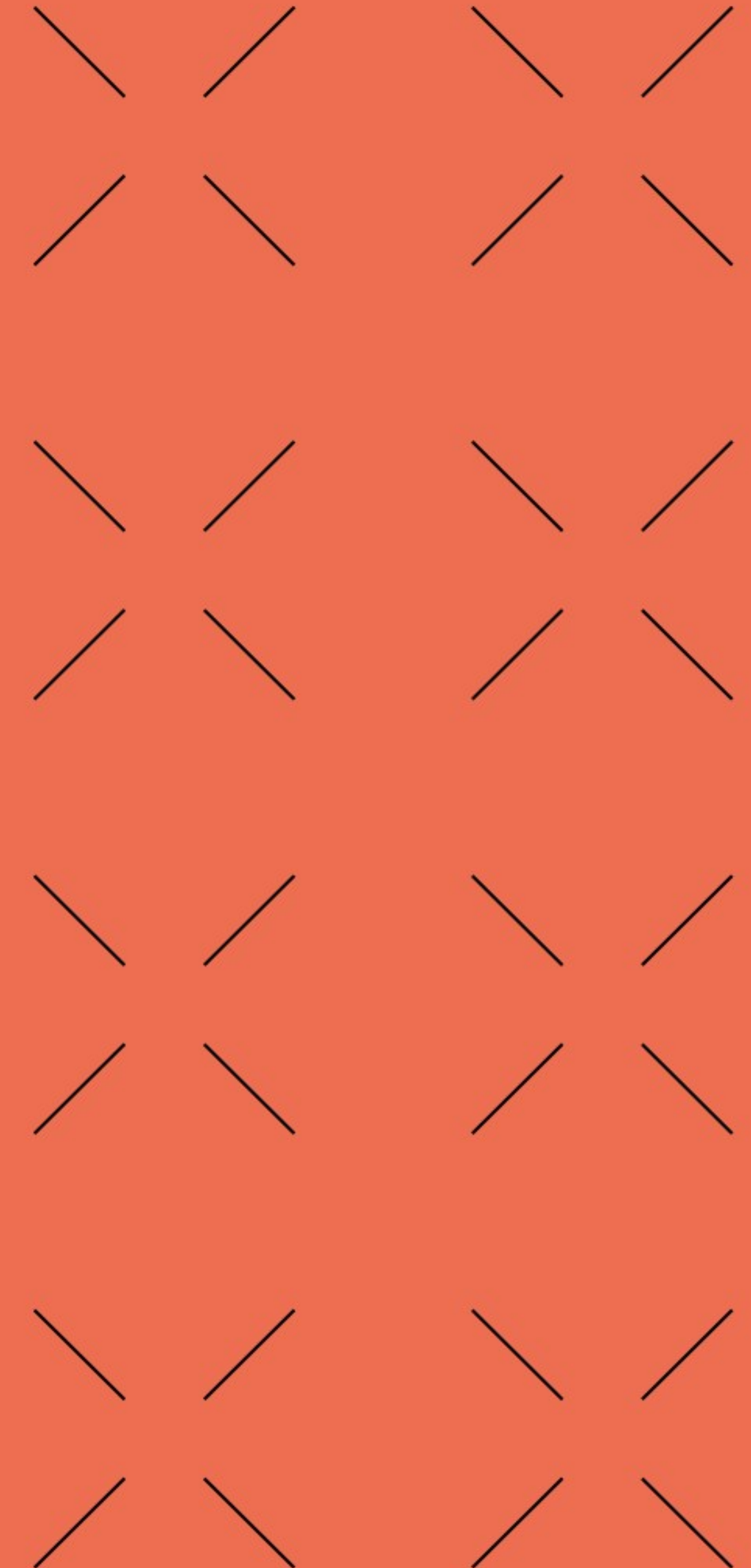
Bases de Datos (DAW/DAM)

CFGS Desarrollo de Aplicaciones Web (DAW)

CFGS Desarrollo de Aplicaciones Multiplataforma (DAM)

Abelardo Martínez y Pau Miñana

Curso 2023-2024



Créditos



- Apuntes realizados por Abelardo Martínez y Pau Miñana.
- Basados y modificados de Sergio Badal (www.sergiobadal.com) y Raquel Torres.
- Las imágenes e iconos empleados están protegidos por la licencia [LGPL](#) y se han obtenido de:
 - https://commons.wikimedia.org/wiki/Crystal_Clear
 - <https://www.openclipart.org>

Contenidos

¿Qué veremos en esta parte?



- Dependencias
- Formas normales
- Actividades propuestas (no evaluables)
 - Boletín C: diagramas de complejidad alta

Contenidos

1. ¿QUÉ ES LA NORMALIZACIÓN?
2. DEPENDENCIAS
3. PRIMERA FORMA NORMAL
4. SEGUNDA FORMA NORMAL
5. TERCERA FORMA NORMAL
6. MÁS FORMAS NORMALES
7. EJEMPLO COMPLETO
8. ACTIVIDADES PROPUESTAS
9. BIBLIOGRAFÍA



1. ¿QUÉ ES LA NORMALIZACIÓN?

¿Qué es la normalización?

La **normalización** es un proceso de refinamiento para comprobar la calidad de nuestro modelo verificando que las relaciones o tablas del modelo Relacional obtenido no tienen redundancias ni inconsistencias.

El proceso de normalización consiste en la aplicación de una serie de reglas que nos sirven para verificar, y en algunas ocasiones modificar, nuestro modelo relacional.

Para ello, vamos a aplicar las siguientes **fases de la normalización** (existen más fases pero se salen de los objetivos de este curso):

- Primera forma normal → **1FN**
- Segunda forma normal → **2FN**
- Tercera forma normal → **3FN**

Proceso de normalización

El proceso de normalización se realizará de la siguiente forma:

- 1) Primero comprobaremos que todas nuestras relaciones o tablas están en 1FN y, si no es así, realizaremos los cambios necesarios para que así sea.
- 2) Una vez lo tenemos todo en 1FN aplicaremos las reglas para ver si están en 2FN y así sucesivamente.
- 3) Por último, si decimos que nuestro modelo está en 3FN, ya sabemos que cumple 1FN, 2FN y 3FN.

Antes de comenzar a ver las reglas de la normalización debemos conocer el concepto de dependencia funcional y sus tipos.

2. DEPENDENCIAS

Dependencia funcional

Un atributo B depende funcionalmente de otro atributo A si a **cada valor de A** le corresponde un **único valor de B**. Se dice que A implica B o lo que es lo mismo $A \rightarrow B$. En este caso A recibe el nombre de **implicante**.

Luego cuando tenemos un atributo A que implica B ($A \rightarrow B$) es lo mismo que decir que B depende funcionalmente de A.

Por **ejemplo**, en la siguiente tabla Personas podemos tener:

Personas (DNI, Nombre, Fecha_Nacimiento);

- Podemos decir que $DNI \rightarrow Nombre$, ya que un DNI se corresponde con un único Nombre, es decir, a través del DNI podemos localizar el nombre de la persona a la que pertenece.

Dependencia funcional completa

Dado un conjunto de atributos A formado por a_1 , a_2 , a_3 , etc. (estos atributos formarán una clave primaria compuesta) existe una dependencia funcional completa cuando B depende de A pero no de un subconjunto de A.

Por **ejemplo**, cuando tenemos una tabla de Compras de la siguiente forma:

Compras (Referencia_Producto, Código_Proveedor, Dirección_Proveedor, Cantidad, Precio)

- En este caso, nuestra clave primaria está formada por dos campos: la Referencia_Producto y el Código_Proveedor. Debemos comprobar si todos los demás atributos tienen una dependencia funcional de toda la clave primaria.
- Podemos observar que la *Dirección_Proveedor*, no dependerá de ambos, sino solamente del Código_Proveedor, luego para ese atributo no existiría dependencia funcional completa.
- Sin embargo, el resto de los campos -tanto la Cantidad, como el Precio acordado- dependen del conjunto; es decir, dependen completamente de la clave.

Dependencia transitiva

Este tipo de dependencia implica a tres atributos. Cuando existe una dependencia $A \rightarrow B$ y a su vez tenemos que $B \rightarrow C$, podemos decir que existe una dependencia funcional transitiva entre A y C.

Por **ejemplo**, si tenemos una tabla de Productos como la siguiente:

Productos (Referencia_Producto, Nombre, Precio, Stock, Fabricante, País);

- Aquí podemos encontrar que el *Fabricante* depende de la *Referencia_Producto*, es decir:
 $\text{Referencia_Producto} \rightarrow \text{Fabricante}$
- Y, por otro lado, el *País* también puede depender del *Fabricante*, luego: $\text{Fabricante} \rightarrow \text{País}$.
- Con estas dos dependencias podemos afirmar que el *País* **depende transitivamente** de la *Referencia_Producto*.

Diagrama de dependencias

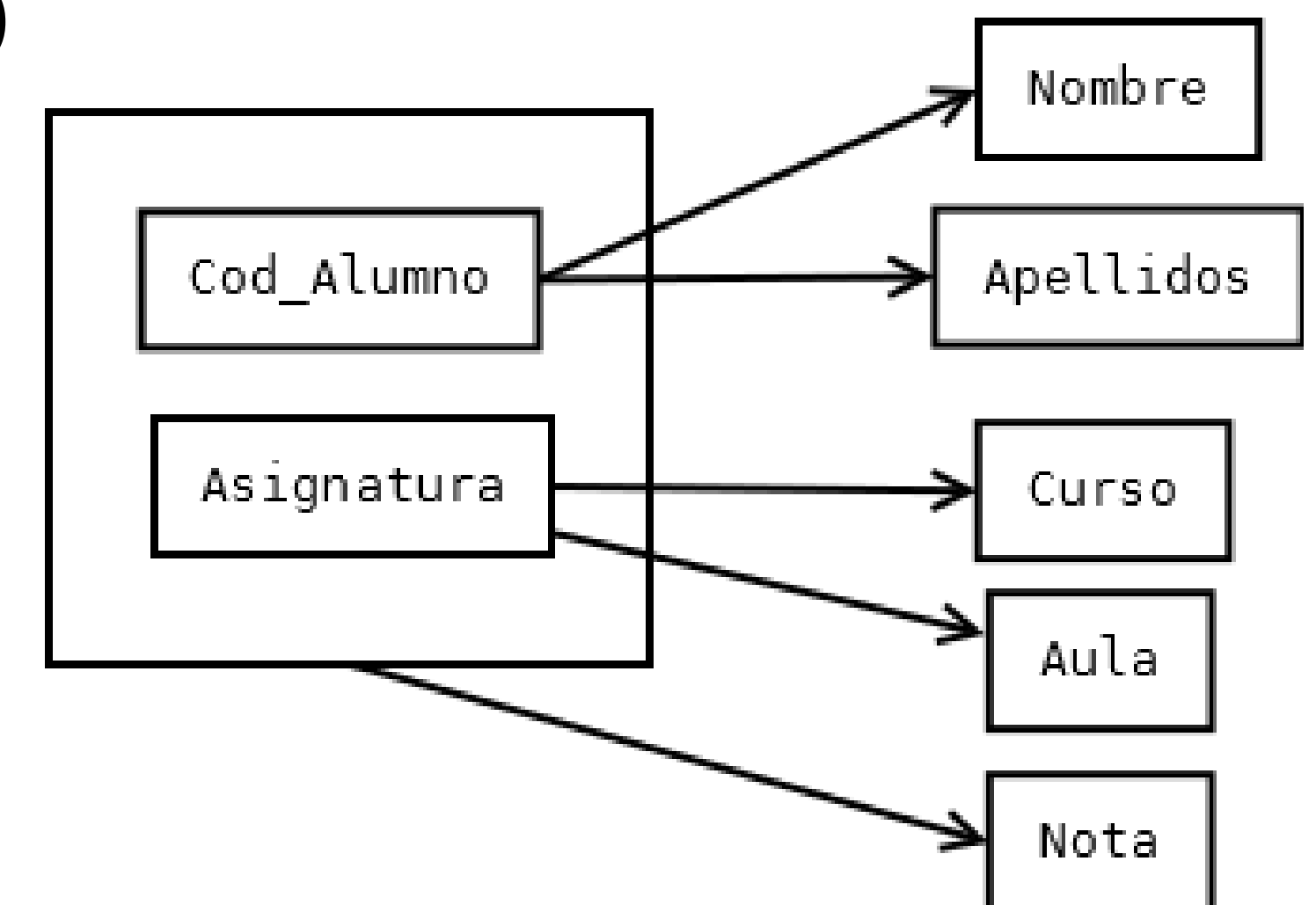
Los diagramas de dependencias muestran todos los atributos de una relación y sus dependencias. Habitualmente **se encierran en una caja el conjunto de atributos de la clave primaria y se unen mediante flechas** los atributos que muestren alguna dependencia entre ellos y también con el conjunto de claves.

Ejemplo. Supongamos que tenemos una relación Alumno en la que representamos los datos de los alumnos y las notas de cada una de las asignaturas en que está matriculado.

Alumno (cod_alumno, nombre, apellidos, asignatura, nota, curso, aula)

Es obvio que no todos los atributos dependen completamente de la CP. Veamos las dependencias funcionales de cada uno de los atributos con respecto a los atributos de la clave:

- $\text{cod_alumno} \rightarrow \text{Nombre}$
- $\text{cod_alumno} \rightarrow \text{Apellidos}$
- $\text{asignatura} \rightarrow \text{Curso}$
- $\text{asignatura} \rightarrow \text{Aula}$
- $\{\text{cod_alumno}, \text{asignatura}\} \rightarrow \text{Nota}$



3. PRIMERA FORMA NORMAL

Primera forma normal

1FN

Una tabla está en 1FN si y solo si los valores que componen cada atributo de una tupla son **atómicos** (ni compuestos, ni multivaluados) **y no derivados**.

Es decir, cada atributo de la relación toma un único valor del dominio correspondiente y no hay valores definidos en función de otros atributos.

¿Cómo realizar la transformación?

Los atributos **derivados** se sustituyen por los atributos de los que dependen si es necesario.

- Por ejemplo, dado que el atributo edad suele ser derivado de la fecha de nacimiento, eliminamos dicho atributo siempre que la fecha de nacimiento esté almacenada, ya que podemos recuperar su valor.

Los atributos **compuestos** se dividen en tantos campos como tenga la composición.

- Por ejemplo, el típico atributo nombre completo se sustituye por nombre y apellidos.

Los atributos **multivaluados** se sustituyen por una relación en ER y se convierten a tablas como proceda.

- Por ejemplo, el típico atributo teléfonos se sustituye por la relación que veremos a continuación.

Ejemplo 1FN

Supongamos que tenemos la siguiente tabla **Centros**.

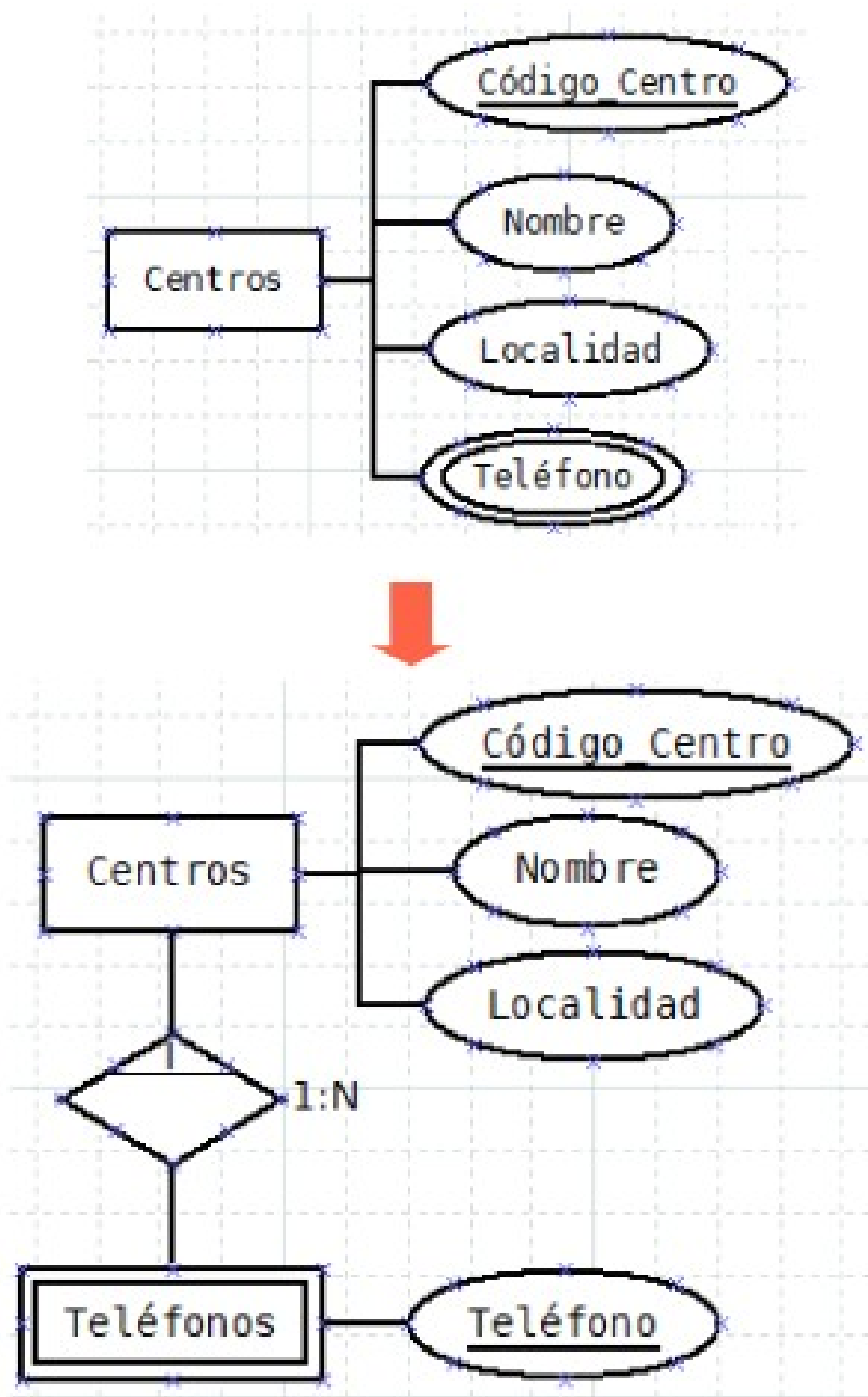
Código_Centro	Nombre	Localidad	Teléfono
45005467	IES Ribera del Tajo	Talavera de la Reina	925722233 - 925722804
45005239	IES Azarquiel	Toledo	925267843 – 925637843
36003748	IES El Plantío	Huelva	973847284

Nueva tabla **Centros**.

Código_Centro	Nombre	Localidad
45005467	IES Ribera del Tajo	Talavera de la Reina
45005239	IES Azarquiel	Toledo
36003748	IES El Plantío	Huelva

Y una nueva tabla que llamaremos **Teléfonos**.

Código_Centro	Teléfono
45005467	925722233
45005467	925722804
45005239	925267843
45005239	925637843
36003748	973847284



4. SEGUNDA FORMA NORMAL

Segunda forma normal

2FN

Una tabla está 2FN si y solo si está en 1FN y **cada atributo no clave primaria, tiene una dependencia funcional completa** con la clave primaria.

La 2FN se aplica a las relaciones que tienen claves primarias compuestas por dos o más atributos.

Si una relación está en 1FN y su clave primaria es simple, entonces también está en 2FN.

¿Cómo realizar la transformación?

Para pasar una relación que está en 1FN a 2FN hay que **eliminar las dependencias parciales de la clave primaria**.

Para ello, se eliminan los atributos que son parcialmente dependientes y se ponen en una nueva relación y como clave primaria la parte de la que dependen de la clave primaria.

De esta manera tendremos dos tablas: una con los atributos que dependen de manera completa de la clave y otra tabla con los atributos que solo dependen de una parte de la clave.

Ejemplo 2FN

Podemos ver que como clave principal se ha elegido el conjunto Empleado + Especialidad.

Sin embargo, podemos observar que el atributo Empresa no depende de toda la clave sino solo de parte de ella, en este caso de Empleado.

Supongamos que tenemos la siguiente tabla **Empleados**.

<u>Empleado</u>	<u>Especialidad</u>	<u>Empresa</u>
Juan Velasco	Bases de Datos	IBM
Juan Velasco	Sistemas Linux	IBM
Ana Comarce	Bases de Datos	Oracle
Javier Gil	Sistemas Windows	Microsoft
Javier Gil	Desarrollo .Net	Microsoft

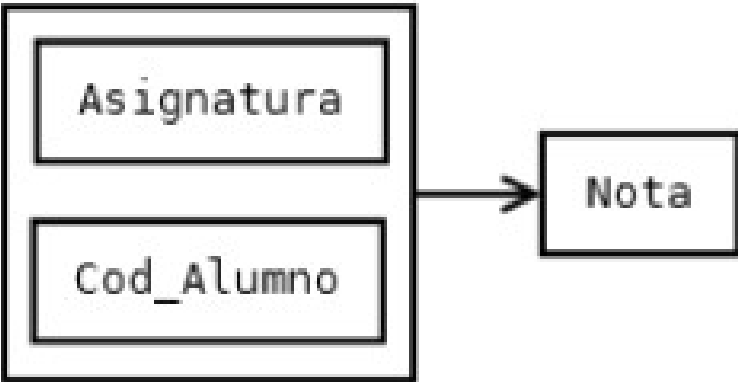
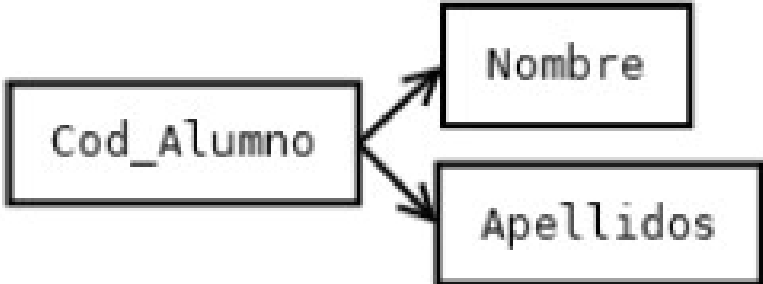
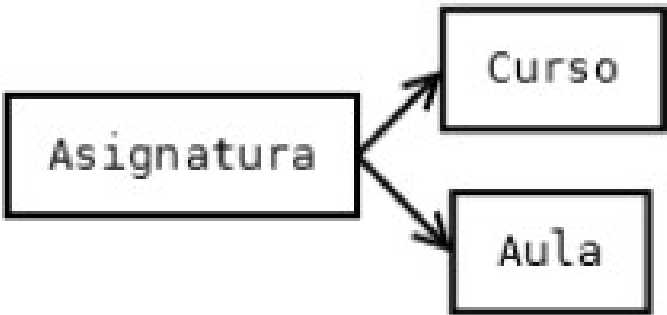
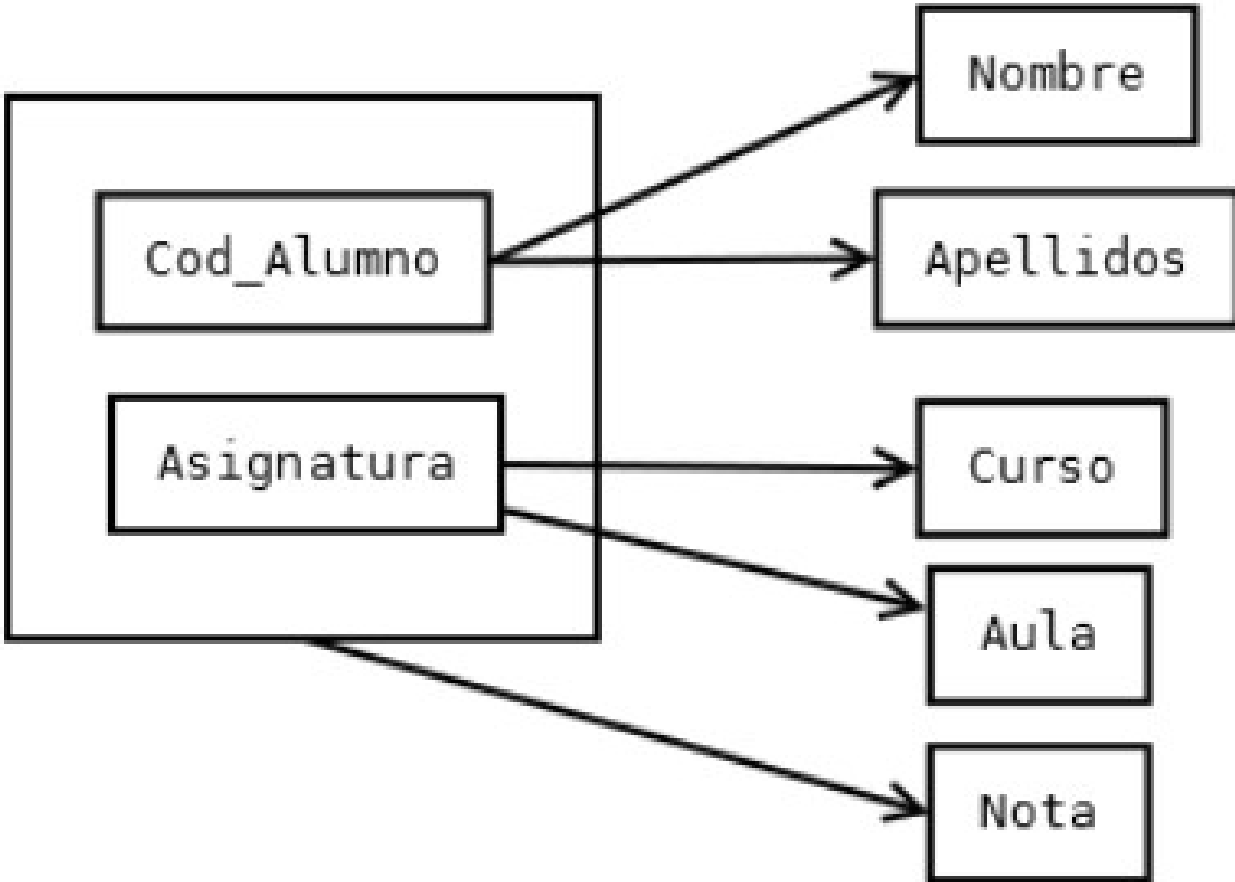
Nueva tabla **Empleados**.

<u>Empleado</u>	<u>Empresa</u>
Juan Velasco	IBM
Ana Comarce	Oracle
Javier Gil	Microsoft

Y una nueva tabla que llamaremos **Lugares_Trabajo**.

<u>Empleado</u>	<u>Especialidad</u>
Juan Velasco	Bases de Datos
Juan Velasco	Sistemas Linux
Ana Comarce	Bases de Datos
Javier Gil	Sistemas Windows
Javier Gil	Desarrollo .Net

Ejemplo 2FN



<u>asignatura</u>	curso	aula
Lengua I	1	15
Inglés II	2	16
Ciencias II	2	14
Plástica I	1	18
Matemáticas I	1	12
Lengua II	2	11

<u>cod_alumno</u>	nombre	apellidos
1111	Pepe	García
2222	María	Suárez
3333	Juan	Gil
4444	Francisco	Montoya

Alumnos (cod_alumno, nombre, apellidos, asignatura, nota, curso, aula)

<u>cod_alumno</u>	nombre	apellidos	<u>asignatura</u>	nota	curso	aula
1111	Pepe	García	Lengua I	5	1	15
1111	Pepe	García	Inglés II	5	2	16
2222	María	Suárez	Inglés II	7	2	16
2222	María	Suárez	Ciencias II	7	2	14
3333	Juan	Gil	Plástica I	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 I	8	1	14

Alumnos (cod_alumno, nombre, apellidos)

Asignaturas (asignatura, curso, aula)

Notas (cod_alumno, asignatura, nota)

CAj: cod_alumno → Alumnos (cod_alumno)

CAj: asignatura → Asignaturas (asignatura)

<u>cod_alumno</u>	<u>asignatura</u>	nota
1111	Lengua I	5
1111	Inglés II	5
2222	Ciencias II	7
2222	Inglés II	7
3333	Plástica I	6
3333	Matemáticas I	6
4444	Lengua II	4
4444	Matemáticas I	6
4444	Ciencias I	8

5. TERCERA FORMA NORMAL

Tercera forma normal

3FN

Una tabla está en 3FN si está en 2FN y los atributos no-clave no dependen de otros atributos no-clave, es decir **no existen dependencias transitivas de la clave primaria**.

«Todo atributo no clave debe proporcionar información sobre la clave, sobre toda la clave y nada más que la clave... con la ayuda de Codd».

¿Cómo realizar la transformación?

Para eliminar las dependencias transitivas se suele crear una nueva tabla con el campo del que dependen como clave principal.

Ejemplo 3FN

Supongamos que tenemos la siguiente tabla **Torneos**.

<u>Torneo</u>	<u>Año</u>	Ganador	Fecha_Nacimiento_Ganador
Alcores	2008	Javier Gil	20 de Abril de 1990
Estoril	2008	Alberto Sanz	15 de Julio de 1991
Getafe	2008	Jacinto Martín	10 de Enero de 1989
Santa María	2008	Roberto Loaisa	25 de Febrero de 1989
Alcores	2009	Jacinto Martín	10 de Enero de 1989
Estoril	2009	Ignacio Gómez	20 de Septiembre de 1990
Lisboa	2009	Roberto Loaisa	25 de Febrero de 1989
Getafe	2009	Roberto Loaisa	25 de Febrero de 1989
Santa María	2009	Javier Gil	20 de Abril de 1990



Nueva tabla **Torneos**.

<u>Torneo</u>	<u>Año</u>	Ganador
Alcores	2008	Javier Gil
Estoril	2008	Alberto Sanz
Getafe	2008	Jacinto Martín
Santa María	2008	Roberto Loaisa
Alcores	2009	Jacinto Martín
Estoril	2009	Ignacio Gómez
Lisboa	2009	Roberto Loaisa
Getafe	2009	Roberto Loaisa
Santa María	2009	Javier Gil

- El atributo Ganador depende del nombre del Torneo y del Año del mismo (ambas forman la clave primaria).
- La fecha de nacimiento del ganador no depende de la clave principal, sino que depende del ganador. Aquí podemos ver una dependencia transitiva de la clave principal:
 - Torneo + Año → Ganador
 - Ganador → Fecha_Nacimiento_Ganador

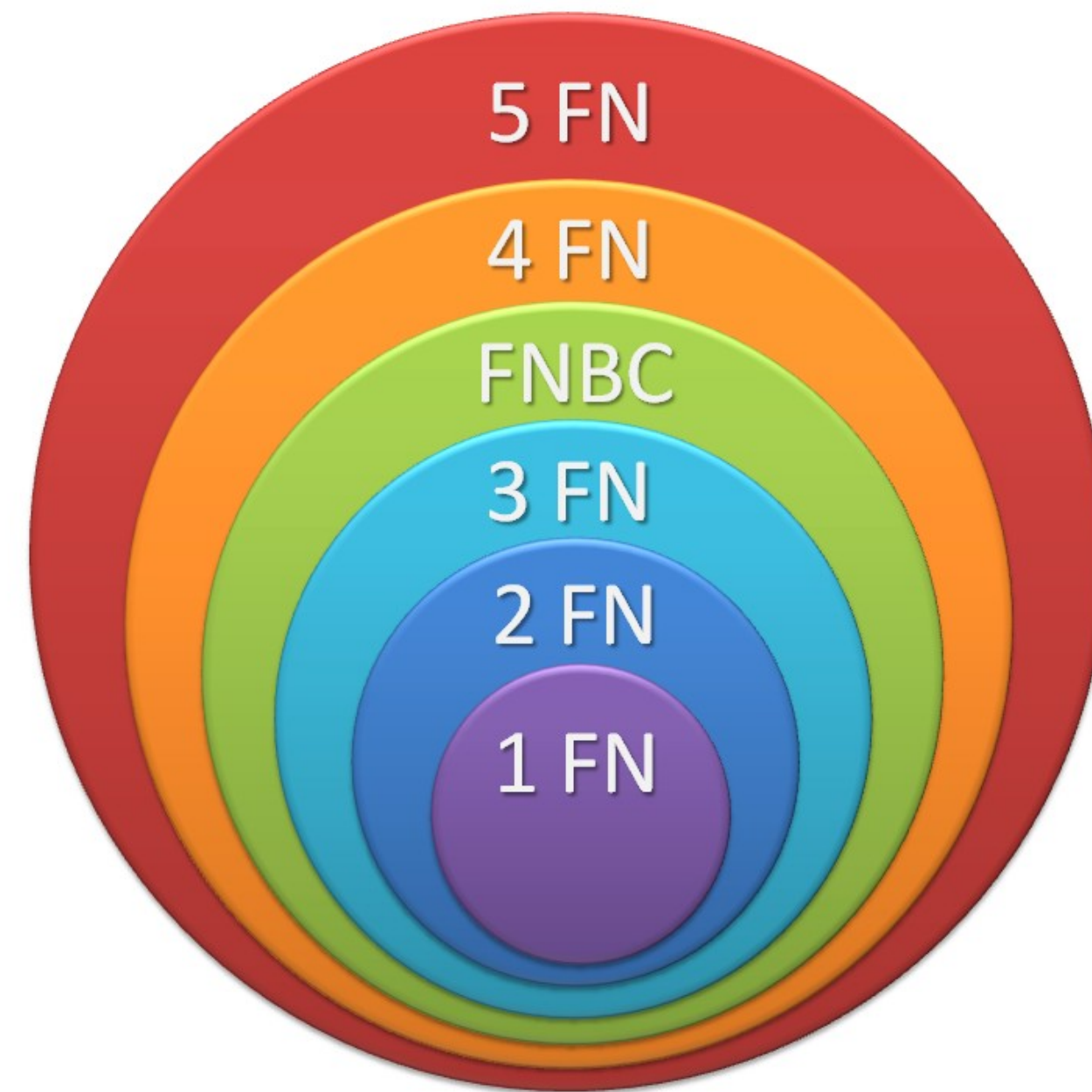
Luego, Fecha_Nacimiento_Ganador depende transitivamente de la clave primaria.

Y por otro lado se genera la tabla **Ganadores**.

<u>Ganador</u>	Fecha_Nacimiento_Ganador
Javier Gil	20 de Abril de 1990
Alberto Sanz	15 de Julio de 1991
Jacinto Martín	10 de Enero de 1989
Roberto Loaisa	25 de Febrero de 1989
Ignacio Gómez	20 de Septiembre de 1990

6. MÁS FORMAS NORMALES

Otras formas normales



Además de estas formas normales, existen otras más cuyo estudio es muchas veces más teórico que práctico.

Se pueden demostrar matemáticamente, pero prácticamente no se suelen aplicar al crear un diseño real, por ello no las vamos a tratar en este curso.

<https://picodotdev.github.io/blog-bitix/2018/02/las-6-plus-2-formas-normales-de-las-bases-de-datos-relacionales/>

7. EJEMPLO COMPLETO

Paso a 1FN

Cliente (numcli, nif, nombre, dir, nom_ciudad, nom_prov, telf, {cuenta}ⁿ)

Cuenta (num_cuenta, tipo_c, saldo, {tarjeta}ⁿ)

Tarjeta (num_tarjeta, tipo_t, comision, limite)

Lo primero es eliminar los atributos multivaluados y los atributos compuestos. La relación **Cliente** solo tiene un atributo multivaluado llamado “cuenta”. Para ello quitamos el atributo de la tabla principal, buscamos la clave primaria y añadimos las restricciones:

Cliente (numcli, nif, nombre, dir, nom_ciudad, nom_prov, telf)

CP: numcli

Cuenta (num_cuenta, tipo_c, saldo, numcli)

CP: num_cuenta

CAj: numcli → Cliente (numcli)

Tarjeta (num_tarjeta, num_cuenta, tipo_t, comisión, límite)

CP: num_tarjeta

CAj: num_cuenta → Cuenta (num_cuenta)

Paso a 2FN

La 2FN obliga a que todos los atributos que no forman parte de la clave, dependan completamente de ésta. Como ninguna de las relaciones obtenidas en el paso anterior tiene una clave compuesta, todas ellas se encuentran ya en 2FN.

Cliente (numcli, nif, nombre, dir, nom_ciudad, nom_prov, telf)

CP: (numcli)

Cuenta (num_cuenta, tipo_c, saldo, numcli)

CP: num_cuenta

CAj: numcli → Cliente (numcli)

Tarjeta (num_tarjeta, num_cuenta, tipo_t, comisión, límite)

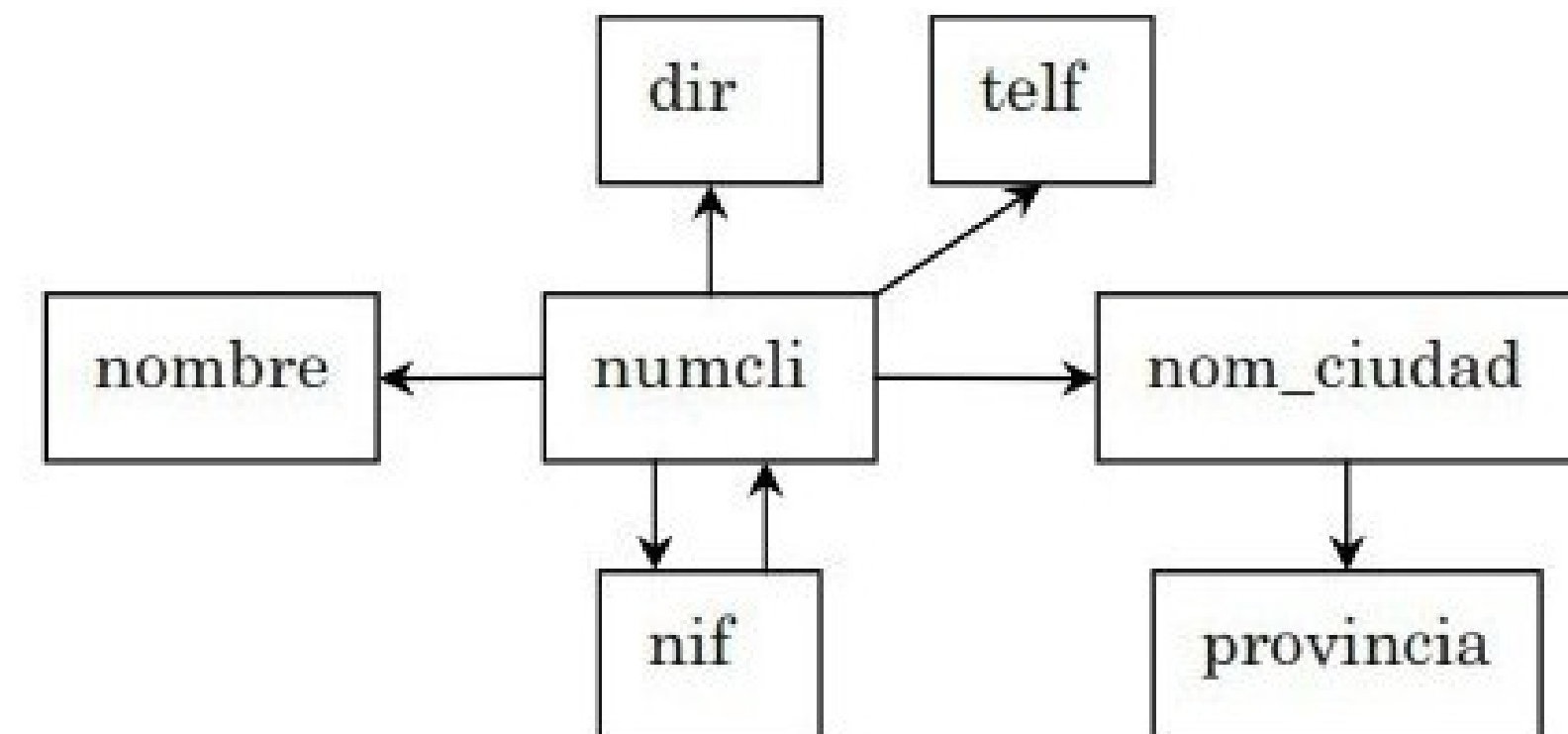
CP: num_tarjeta

CAj: num_cuenta → Cuenta (num_cuenta)

Paso a 3FN

Ahora queda eliminar las dependencias transitivas y entre atributos. Veamos primero las dependencias de cada una de las tablas:

Cliente(*numcli*: d_numcli, *nif*: d_nif, *nombre*: d_nombre, *dir*: dom_dir, *ciudad*: d_nom_ciudad, *provincia*: d_nom_prov, *telf*: d_telf)
CP: {numcli}
Único: {nif}



Ciudad (nom_ciudad, nom_prov)

CP: nom_ciudad

Cliente (numcli, nif, nombre, dir, nom_ciudad, telf)

CP: numcli

CAIt: nif

ÚNICO: nif

CAj: nom_ciudad → Ciudad (nom_ciudad)

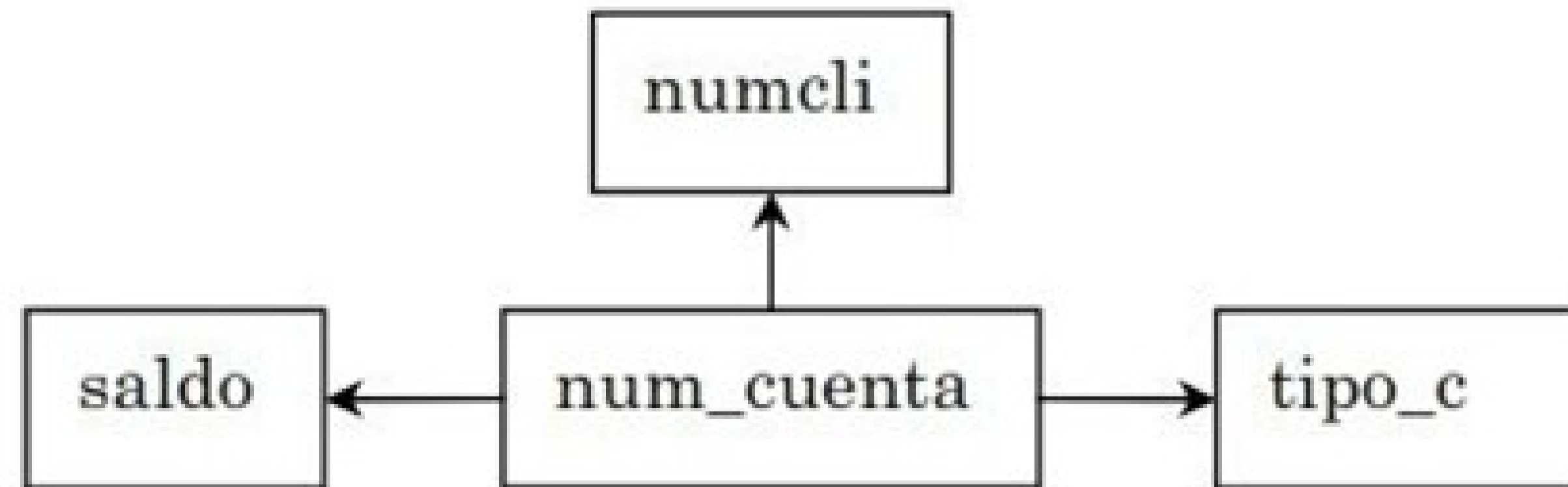
Paso a 3FN

Esta relación no tiene relaciones transitivas entre atributos, ya se encuentra en 3FN.

Cuenta(*numcli*: d_num_cli, *num_cuenta*: d_num_cuenta, *tipo_c*: d_tipo_c, *saldo*: d_saldo)

CP: {num_cuenta}

CA: {numcli} → Cliente



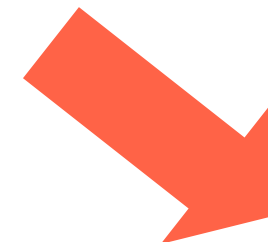
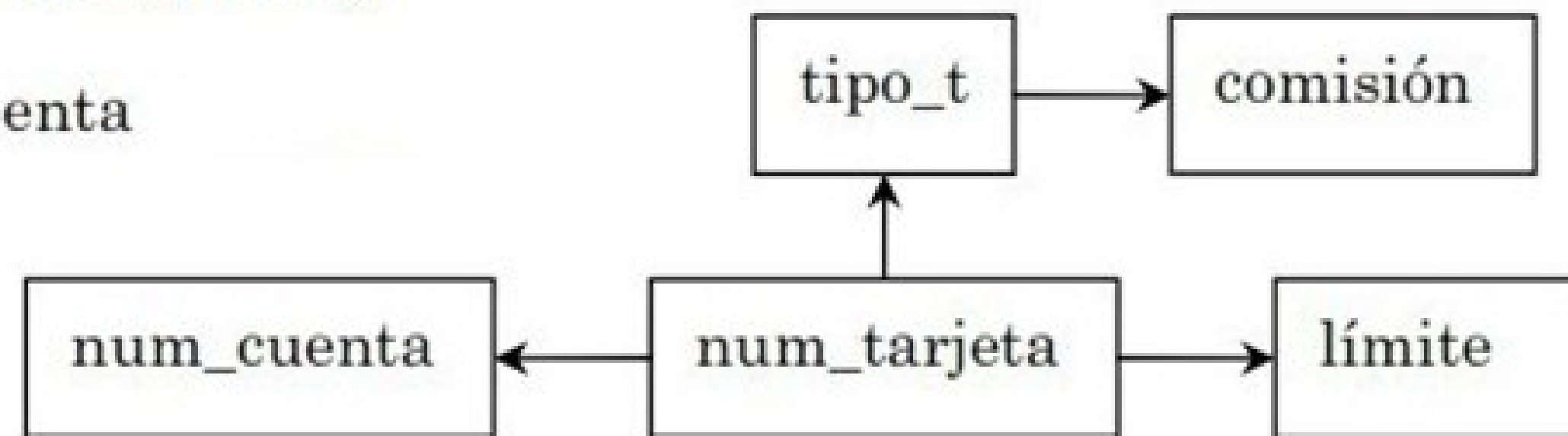
Paso a 3FN

Hemos encontrado una dependencia transitiva que hay que eliminar.

Tarjeta(*num_cuenta*: d_num_cuenta, *num_tarjeta*: d_num_tarjeta, *tipo_t*: d_tipo_t, *comisión*: d_comisión, *límite*: d_límite)

CP: {num_tarjeta}

CA: {num_cuenta} → Cuenta



Comisión (tipo_t, comisión)

CP: tipo_t

Tarjeta (num_tarjeta, num_cuenta, tipo_t, límite)

CP: num_tarjeta

CAj: num_cuenta → Cuenta (num_cuenta)

CAj: tipo_t → Comisión (tipo_t)

Paso a 3FN

El resultado final es:

Ciudad (nom_ciudad, nom_prov)

CP: nom_ciudad

Cliente (numcli, nif, nombre, dir, nom_ciudad, telf)

CP: numcli

CAIt: nif

ÚNICO: nif

CAj: nom_ciudad → Ciudad (nom_ciudad)

Cuenta (num_cuenta, tipo_c, saldo, numcli)

CP: num_cuenta

CAj: numcli → Cliente (numcli)

Comisión (tipo_t, comisión)

CP: tipo_t

Tarjeta (num_tarjeta, num_cuenta, tipo_t, límite)

CP: num_tarjeta

CAj: num_cuenta → Cuenta (num_cuenta)

CAj: tipo_t → Comisión (tipo_t)

8. ACTIVIDADES PROPUESTAS

Actividades propuestas



Estas actividades son **opcionales y no evaluables** pero es recomendable hacerlas para un mejor aprendizaje de la asignatura.

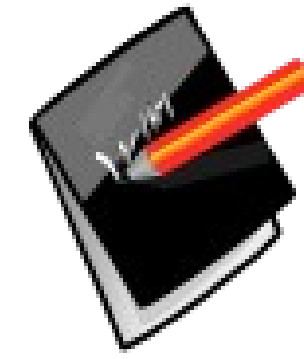
Mira los ejercicios propuestos que encontrarás en el “Aula Virtual”.

- **03. Boletín actividades C:** ejercicios de complejidad alta.

Próximamente encontrarás las soluciones a los ejercicios propuestos.

9. BIBLIOGRAFÍA

Recursos



- Bases de datos. Tuinstitutoonline. <https://www.tuinstitutoonline.com/>
- Iván López, M.^a Jesús Castellano. John Ospino. Bases de Datos. Ed. Garceta, 2a edición, 2017. ISBN: 978-8415452959
- Matilde Celma, Juan Carlos Casamayor y Laura Mota. Bases de datos relacionales. Ed. Prentice-Hall, 2003
- Cabrera Sánchez, Gregorio. Análisis y diseño detallado de aplicaciones informáticas de gestión. Ed. McGraw-Hill, 1st edition, 1999. ISBN: 8448122313
- Carlos Manuel Martí Hernández. Bases de dades. Desenvolupament d'aplicacions multiplataforma i Desenvolupament d'aplicacions web. Creative Commons. Departament d'Ensenyament, Institut Obert de Catalunya. Dipòsit legal: B. 12715-2016. <https://ioc.xtec.cat/educacio/recursos>

