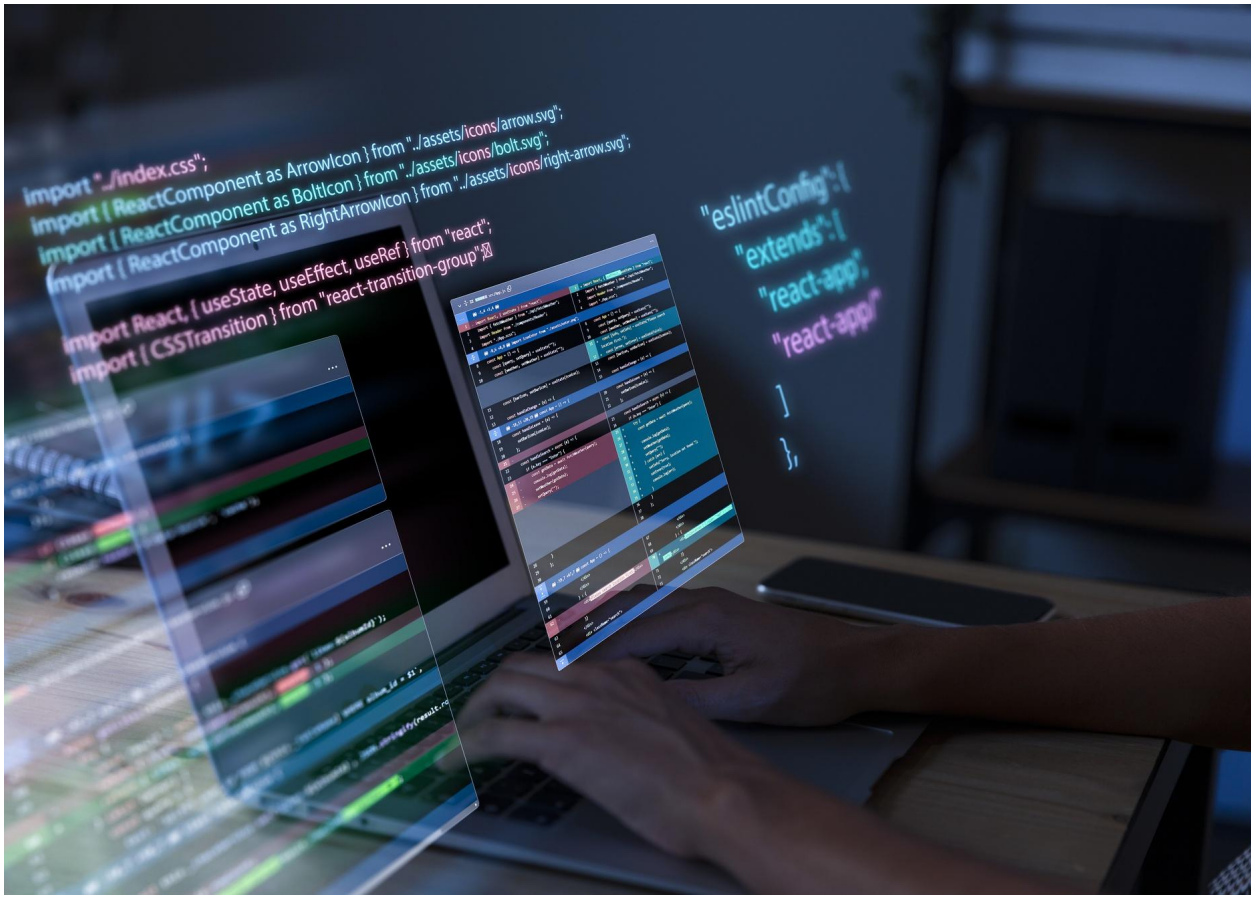
1º DAM/DAW - BASES DATOS

UNIDAD 4

Normalización



Autor: Luz María Álvarez Moreno

Fecha: 13/1/2026

SESIÓN 12 y 13: DEL MODELO E-R AL MODELO RELACIONAL
Introducción a la Normalización 1FN y 2FN
3FN y Boyce-Codd (BCNF)

SESIÓN 14: Otras formas normales (4FN, 5FN) y cuándo aplicarlas.

¿Qué es normalizar?

Normalizar es un proceso para **organizar** las tablas de una base de datos de forma que:

- **no se repitan datos** (menos redundancia),
- se eviten **errores** al insertar/actualizar/borrar,
- y los datos tengan **más integridad** y sean más fáciles de mantener.

Idea clave:
Cada dato debe guardarse *una sola vez* y en *su sitio*.

¿Qué problemas (anomalías) soluciona?

Cuando una BD no está normalizada suelen aparecer:

Redundancia
El mismo dato se repite (ej: el nombre del cliente aparece en muchas filas).

Anomalía de actualización
Cambias un dato en una fila, pero se te olvida cambiarlo en otra → **inconsistencia**.

Anomalía de inserción
No puedes añadir un dato si falta otro (ej: no puedes añadir un curso si aún no hay alumnos).

Anomalía de eliminación

Borras una fila y **pierdes** información que no querías perder.

Conceptos base

Clave primaria (PK)

Campo(s) que identifican una fila de forma única.

- **Simple**: PedidoID
- **Compuesta**: (PedidoID, Producto)
    - Si la clave es compuesta, **2FN** cobra importancia.

Dependencia funcional (la base de todo)

Se escribe:
A → **B** (A determina B)

Ejemplo:
DNI → Nombre
Si sé el DNI, sé el nombre.

Traducción fácil: B depende de A.

Formas normales

1FN: Primera Forma Normal

Una tabla está en **1FN** si:

- **cada campo tiene valores atómicos** (un solo valor),
- **no hay listas** ni "grupos repetidos".

Ejemplo NO 1FN (lista en una celda)

PedidoID	Cliente	Productos
1001	Juan	Ratón, Teclado

Ejemplo 1FN (separar productos en filas)

PedidoID	Cliente	Producto	Cantidad
1001	Juan	Ratón	2
1001	Juan	Teclado	1

Regla fácil: Si hay comas o listas dentro de una celda, no es 1FN.
2FN: Segunda Forma Normal

Teoría

Una tabla está en **2FN** si:

- ya está en 1FN,
- y **no hay dependencias parciales**:
  todo campo que no sea clave debe depender de TODA la clave, no de una parte.

Traducción fácil: Si la clave es compuesta, ningún dato debe 'depender solo de la mitad'.

Ejemplo típico NO 2FN

PK = (PedidoID, Producto)

PedidoID	Producto	Cliente
1001	Ratón	Juan
1001	Teclado	Juan

Cliente depende solo de PedidoID, no de (PedidoID,Producto) → parcial.

Arreglo a 2FN (separar)

Pedidos

PedidoID	Cliente
1001	Juan

DetallePedido

PedidoID	Producto
1001	Ratón
1001	Teclado

3FN: Tercera Forma Normal

Una tabla está en **3FN** si:

- está en 2FN,
- y **no hay dependencias transitivas**:
  un campo no clave **no puede depender de otro campo no clave**.

Traducción fácil: Los datos que describen otra cosa (por ejemplo un curso o un departamento) se van a su tabla.

Ejemplo NO 3FN

Alumno	Curso	Profesor
Ana	SQL	Marta
Juan	SQL	Marta

Aquí:

- Curso → Profesor
- Profesor no depende del alumno, depende del curso → **transitiva**.

Arreglo a 3FN

Cursos

Curso	Profesor
SQL	Marta

Matriculas

Alumno	Curso
Ana	SQL
Juan	SQL

BCNF – Boyce-Codd (más estricta que 3FN)
BCNF exige que:
Todo determinante (A en A→B) sea clave candidata.
Traducción fácil: Si algo determina otra cosa, debe ser una clave válida.

4FN: Cuarta Forma Normal

Elimina **dependencias multivaluadas** (cuando hay 2 listas independientes mezcladas).

Ejemplo
Un profesor puede impartir varios cursos y hablar varios idiomas, pero **curso e idioma no están relacionados**:

Profesor	Curso	Idioma
Marta	SQL	Inglés
Marta	Python	Inglés
Marta	SQL	Español
Marta	Python	Español

Aparecen combinaciones “artificiales” por mezclar 2 cosas independientes.

Arreglo 4FN
ProfesorCurso(Profesor, Curso)
ProfesorIdioma(Profesor, Idioma)

5FN: Quinta Forma Normal
Elimina redundancias que aparecen al “reconstruir” relaciones con **descomposiciones complejas** (dependencias de unión).

Resumen
1. **1FN**: No listas (una celda = un dato).
2. **2FN**: Si la clave es compuesta, nadie depende de media clave.
3. **3FN**: Nadie depende de otro no-clave; lo que describe otra cosa, a otra tabla.

4. **BCNF/4FN/5FN**: niveles avanzados para dependencias más raras.

Sesión 15 - Práctica de normalización completa sobre un caso real.

PRÁCTICA DE NORMALIZACIÓN : Academia de formación

EL PROBLEMA REAL (ANTES DE NORMALIZAR)

Una academia quiere guardar información sobre:

- Alumnos
- Cursos
- Profesores
- Aulas
- Horarios
-

Y decide hacerlo **todo en una sola tabla**, así:

Tabla inicial (mal diseñada)

AlumnoID	NombreAlumno	Curso	Profesor	Aula	Horarios
1	Ana	SQL, Python	Marta	A1	Lunes 9-11, Miércoles 9-11
2	Juan	SQL	Marta	A1	Lunes 9-11
3	Laura	Python	Luis	B2	Martes 10-12

¿POR QUÉ ESTA TABLA ES UN PROBLEMA?

- **Curso** tiene varios valores: *no es un solo dato*
- **Horarios** tiene varios valores: *otra lista*
- El nombre del profesor y el aula se repiten
- Si cambia el aula de SQL: hay que cambiarla en varias filas
- Si borramos a Ana: perdemos información del curso Python

Conclusión:
Esta tabla mezcla muchas cosas distintas que **no dependen unas de otras**.

PASO CLAVE ANTES DE NORMALIZAR: ¿QUÉ REPRESENTA CADA FILA?

Esto es lo que **más cuesta**, pero es lo más importante.

Cada fila **NO representa un alumno**,

- **NO representa un curso**,
    - representa una **MATRÍCULA**.

Una matrícula = *un alumno se apunta a un curso en un horario*

IDENTIFICAR LA CLAVE PRIMARIA

¿Qué campos necesito para identificar UNA matrícula?

Respuesta:

- AlumnoID
- Curso

Clave primaria compuesta:
(AlumnoID, Curso)

Guárdate esto, porque ahora **la 2FN va a tener sentido**.

PRIMERA FORMA NORMAL (1FN)

Una celda = un solo dato

Una tabla está en **1FN** cuando:

- No hay listas
- No hay varios valores en un mismo campo

¿Qué falla aquí?

- Curso → “SQL, Python”
- Horarios → “Lunes…, Miércoles…”

¿Qué hacemos?
Separar los valores en filas

Tabla en 1FN

AlumnoID	NombreAlumno	Curso	Profesor	Aula	Horario
1	Ana	SQL	Marta	A1	Lunes 9-11
1	Ana	SQL	Marta	A1	Miércoles 9-11
1	Ana	Python	Luis	B2	Martes 10-12
2	Juan	SQL	Marta	A1	Lunes 9-11
3	Laura	Python	Luis	B2	Martes 10-12

Qué hemos logrado:

- Cada celda tiene un solo valor
- La tabla ya se puede trabajar bien

Cumple 1FN

SEGUNDA FORMA NORMAL (2FN)

Nadie depende de media clave.

¿Cuándo importa la 2FN?
SOLO cuando la clave es compuesta.
Y aquí **lo es**: (AlumnoID, Curso)

Pregunta clave de 2FN

¿Todos los campos dependen de **AlumnoID Y Curso a la vez**?

Vamos campo por campo:

- **NombreAlumno** → depende solo de AlumnoID
- **Profesor** → depende solo del Curso
- **Aula** → depende solo del Curso
- **Horario** → depende del Curso

Conclusión:

- Hay **dependencias parciales**
- **NO cumple 2FN**

SOLUCIÓN
Vamos a separar las cosas **por lo que dependen**.

Tabla ALUMNOS
(Datos que dependen solo del alumno)

AlumnoID	NombreAlumno
1	Ana
2	Juan
3	Laura

Tabla CURSOS
(Datos que dependen solo del curso)

Curso	Profesor	Aula
SQL	Marta	A1
Python	Luis	B2

Tabla MATRÍCULAS
(Relación alumno–curso)

AlumnoID	Curso
1	SQL
1	Python
2	SQL
3	Python

Tabla HORARIOS
(Un curso puede tener varios horarios)

Curso	Horario
SQL	Lunes 9-11
SQL	Miércoles 9-11
Python	Martes 10-12

Qué hemos conseguido:

- Cada dato está donde le toca
- Si cambia el profesor de SQL: se cambia **una vez**
- Ya no hay dependencias parciales

Cumple 2FN

TERCERA FORMA NORMAL (3FN)

Un dato no clave no depende de otro no clave

Miramos tabla por tabla

Tabla CURSOS

Curso	Profesor	Aula

- Curso es clave
- Profesor y Aula dependen del curso

Correcto
Cumple 3FN
No hay que tocar nada más.

RESULTADO FINAL (MODELO NORMALIZADO)

Tenemos 4 tablas:

- **ALUMNOS**
- **CURSOS**
- **MATRÍCULAS**
- **HORARIOS**

Cada una:

- Tiene sentido por sí sola
- No repite información
- Se puede ampliar sin romper nada

RESUMEN

- **1FN**: no listas.
- **2FN**: nadie depende de media clave.
- **3FN**: nadie depende de otro no clave.
- **Normalizar no es dividir tablas, es pensar**