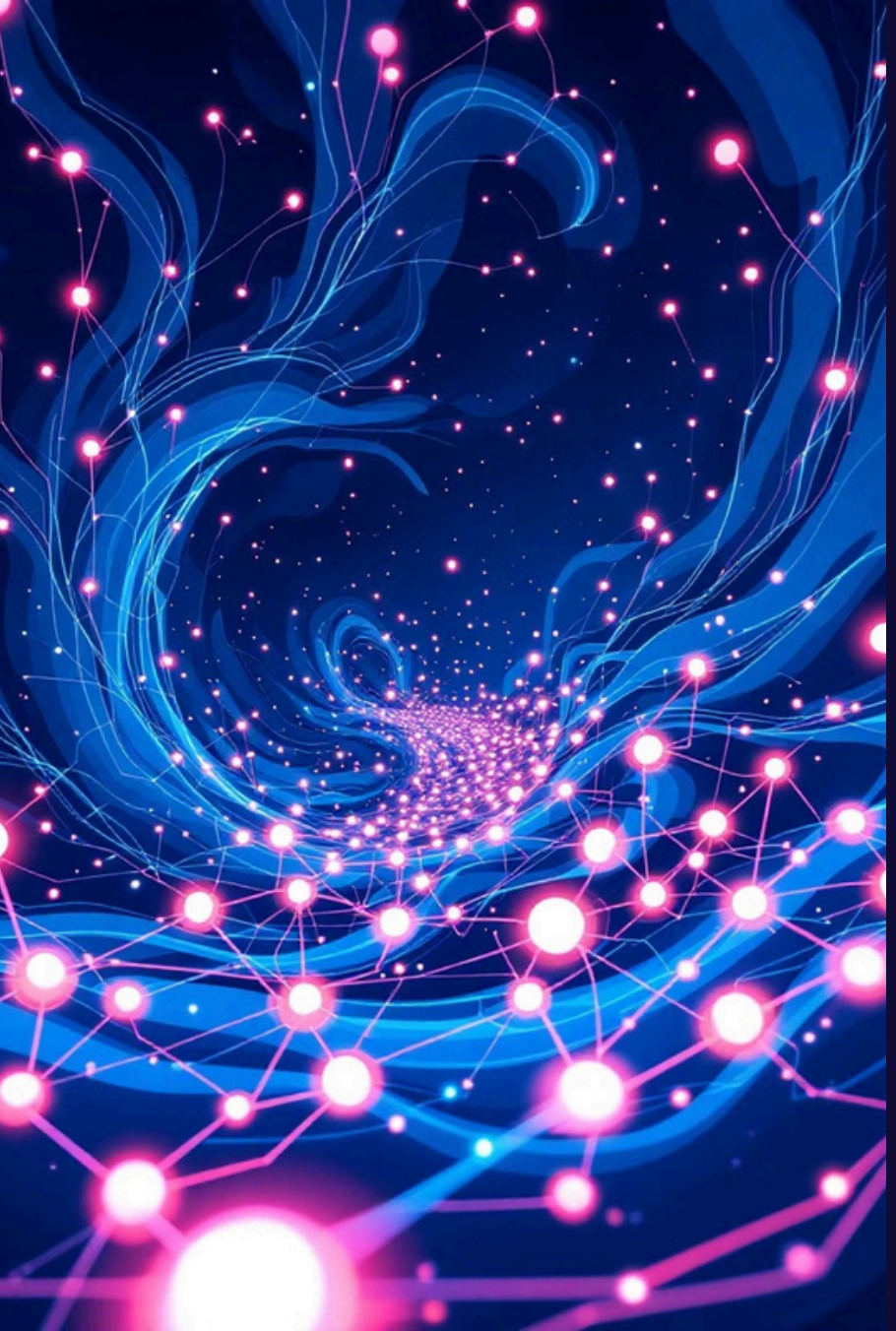


Normalización de Bases de Datos

Descubre cómo transformar datos caóticos en sistemas ordenados y eficientes. Trabajo realizado por Miguel Ángel Antúnez Ruiz, Iván Gutiérrez Mateos y Juan Manuel Olmo Lobo.

¿Qué pasa cuando no normalizas?

Datos duplicados

La misma información aparece repetida en múltiples lugares, creando inconsistencias y desperdiciando espacio de almacenamiento.

Actualizaciones complicadas

Cambiar un dato requiere modificarlo en decenas de sitios diferentes, aumentando el riesgo de errores.

Consultas poco confiables

Los resultados pueden variar según qué copia de los datos se consulte, destruyendo la confianza en el sistema.

¿Qué es la normalización?

La normalización es el proceso de organizar los datos en tablas lógicas y estructuradas. Su objetivo principal es reducir la redundancia y mejorar la integridad de los datos.

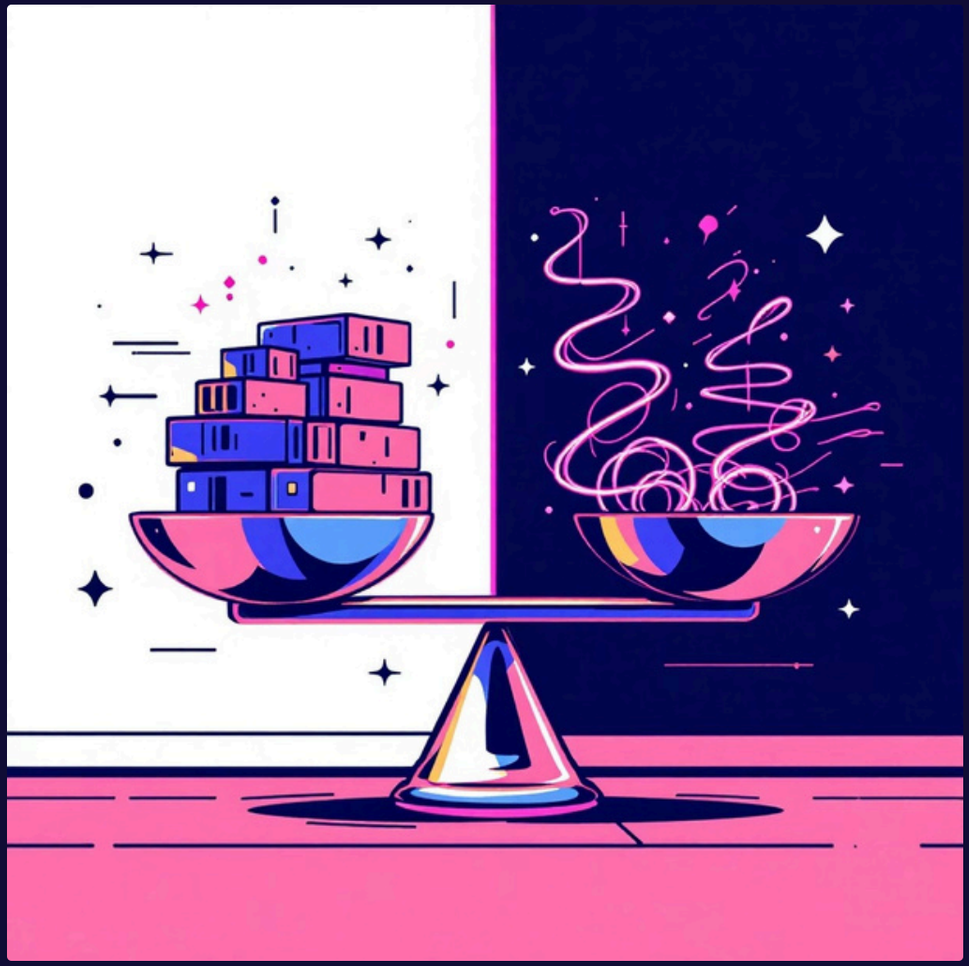
> Piensa en ordenar tu armario: cada prenda tiene su lugar específico, todo está organizado y es fácil de encontrar.

Este proceso transforma estructuras caóticas en sistemas coherentes y mantenibles.

¿Por qué normalizar?

- Evita datos duplicados
- Asegura coherencia y calidad
- Facilita mantenimiento y escalabilidad

Cuándo NO normalizar tanto



Demasiadas tablas pueden afectar el rendimiento. A veces se desnormaliza estratégicamente para optimizar consultas críticas.

El equilibrio entre orden y velocidad es esencial

Etapas de la normalización

Primera Forma Normal (1NF)

Eliminación de valores múltiples en campos individuales. Cada celda debe contener un solo valor atómico.

Segunda Forma Normal (2NF)

Dependencia funcional completa de la clave primaria. Cada atributo debe depender totalmente de la clave.

Tercera Forma Normal (3NF)

Eliminación de dependencias transitivas. Los atributos no clave solo dependen de la clave primaria.

Forma Normal de Boyce-Codd (BCNF)

Versión más estricta que maneja dependencias funcionales complejas con múltiples claves candidatas.

Primera Forma Normal (1NF)

La regla fundamental: cada campo debe contener un solo valor. No se permiten listas, arrays o valores múltiples dentro de una celda.

Incorrecto

Cliente: "Juan Pérez"
Teléfonos: "123456789, 789012345, 345678912"

Correcto

Crear filas separadas:
Juan Pérez | 123456789
Juan Pérez | 789012345
Juan Pérez | 345678912

Segunda Forma Normal (2NF)

Cada dato debe depender completamente de la clave primaria. Si un valor depende solo de una parte de la clave compuesta, debe moverse a otra tabla.

1

Detectar dependencias parciales

Identificar atributos que dependen solo de parte de la clave primaria compuesta.

2

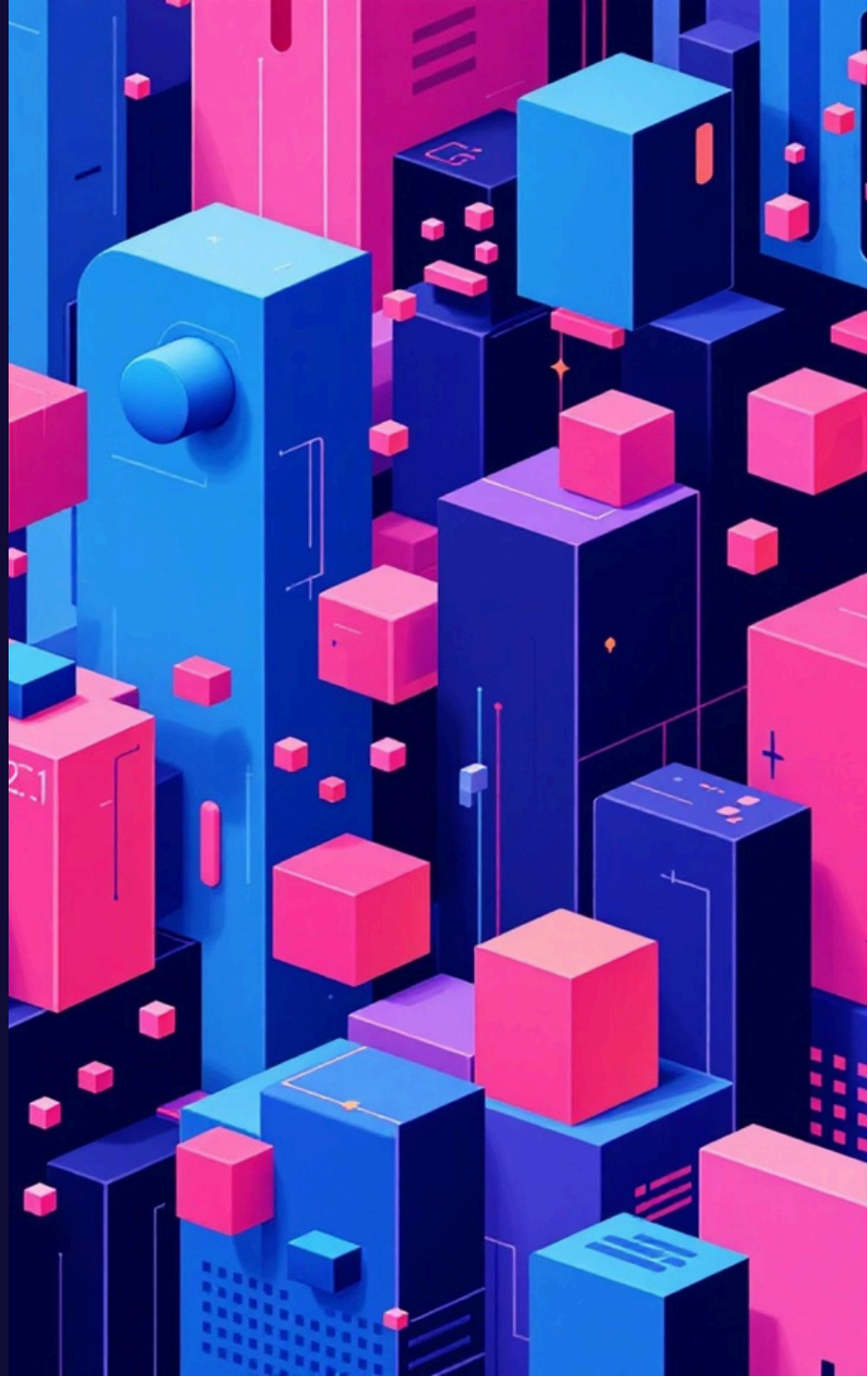
Crear nuevas tablas

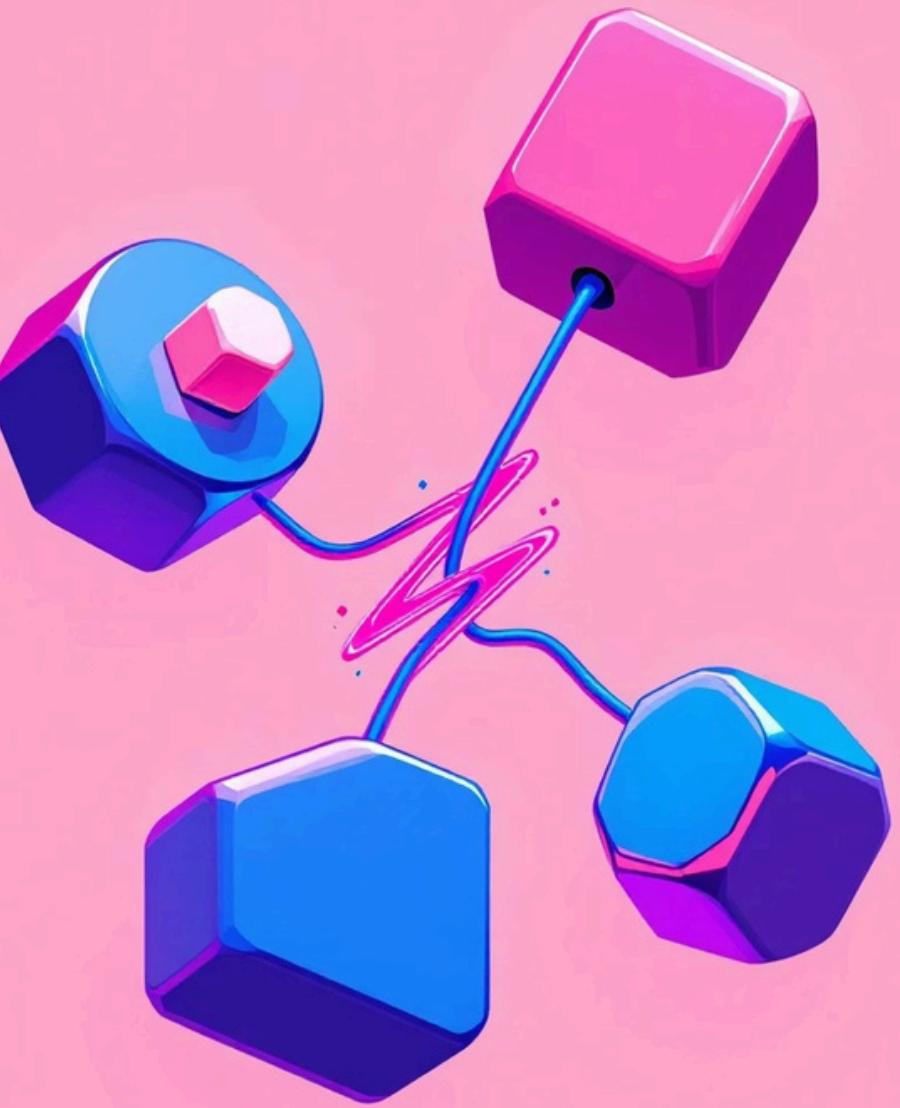
Separar estos atributos en tablas independientes con sus propias claves.

3

Eliminar redundancia

Conectar las tablas mediante relaciones, eliminando la duplicación de datos.





Tercera Forma Normal (3NF)

Elimina las dependencias transitivas entre columnas. Cada campo debe depender directamente de la clave primaria, no de otros campos no clave.

Problema común

Empleado Departamento Ubicación del Departamento

Solución 3NF

Separar departamentos en tabla independiente con su propia ubicación

Forma Normal de Boyce-Codd (BCNF)

Una versión más estricta de la 3NF. Cada determinante debe ser una clave candidata, manejando casos complejos con múltiples dependencias funcionales.

Cuándo aplicar BCNF

Cuando existen dependencias funcionales complejas que la 3NF no resuelve completamente.

Casos especiales

Múltiples claves candidatas que se superponen, creando dependencias no triviales.

Conclusión: cuidar los datos es cuidar el sistema

Integridad garantizada

La normalización mejora la integridad de los datos y previene errores costosos en el futuro.

Mantenimiento simplificado

Facilita el mantenimiento y permite que el proyecto evolucione sin complicaciones técnicas.

Base sólida para crecer

Una base de datos bien normalizada es el fundamento perfecto para el crecimiento empresarial.

Recuerda: los datos organizados de hoy son el éxito de mañana

CONCEPTOS IMPORTANTES

- Campo/ Atributo Es una columna en una tabla. Ejemplo: "Nombre", "Edad".
- Valor atómico: Es un dato único y simple. Ejemplo: “Carlos” (no una lista).
- Clave: Dato que identifica cada fila sin repetir. Ejemplo: un número de ID.
- Dependencia transitiva: Un dato depende de otro, que a su vez depende de otro. Ejemplo: A → B y B → C, entonces A → C.
- Claves candidatas: Campos que pueden servir como clave, porque no se repiten.
- Array: Una lista de valores en un solo campo. Ejemplo: [“rojo”, “azul”] (esto no es atómico).
- Clave compuesta: Clave formada por dos o más campos juntos. Ejemplo: (Curso, Estudiante).
- Determinante: Campo que define el valor de otro. Ejemplo: "DNI" determina "Nombre".
- Dependencia funcional: Un campo depende directamente de otro. Ejemplo: Con el “ID”, sabes el “Nombre”.
- Dependencia no trivial: Cuando un campo depende de otro que no forma parte del resultado. Ejemplo: A → B, y B no está en A.