

# KIMBALL Y EL MODELADO DIMENSIONAL

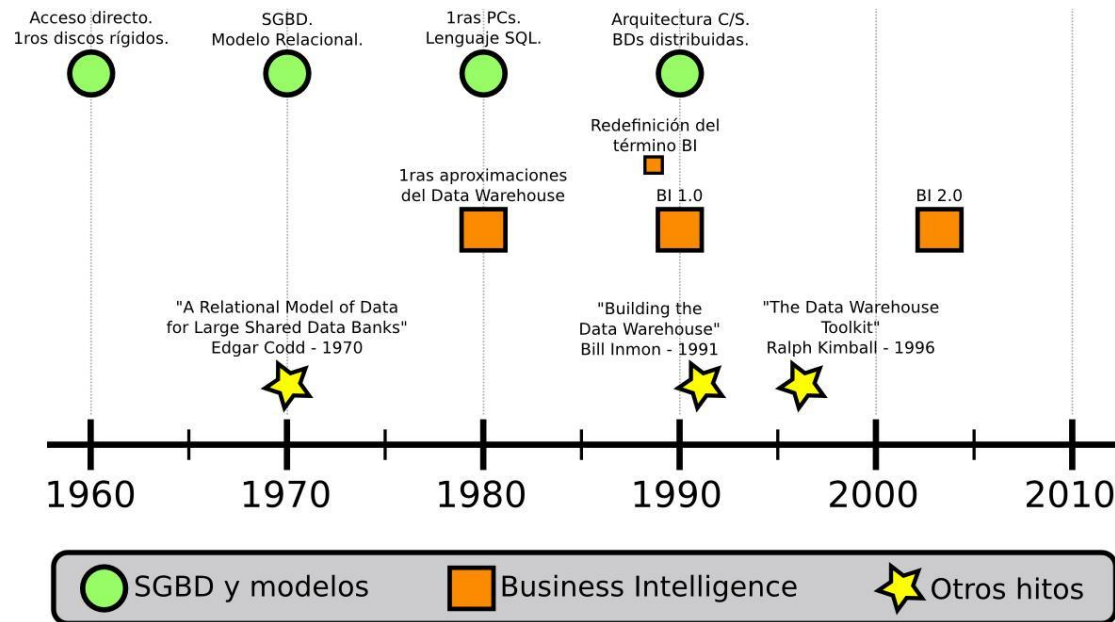
Fundamentos del modelado de datos moderno

Rubén Hermoso Díez

# Un poco de historia...

Desde los años 60 y 70, las empresas empezaron a informatizar sus procesos. Sistemas diseñados para registrar operaciones: compras, ventas, pagos, etc. Así nacen los sistemas OLTP (Online Transaction Processing).

*En los inicios de la informática empresarial, lo importante era que todo funcionara: vender, facturar, cobrar... La información se registraba, pero nadie pensaba en analizarla.*



# ¿Qué es OLTP?

OLTP significa Online Transaction Processing, o Procesamiento de Transacciones en Línea. Es un tipo de sistema que usa bases de datos para registrar operaciones del día a día de una empresa: ventas, compras, reservas, pagos, movimientos de almacén, etc.

- Se basa en **bases de datos relacionales**, normalmente bien **normalizadas** (para evitar duplicidades y asegurar la integridad).
- Su diseño está optimizado para **operaciones rápidas de lectura y escritura**: insertar, actualizar, eliminar o consultar un solo registro.
- Es el sistema que está “por detrás” de muchas aplicaciones operativas: un TPV en una tienda, una app de reservas, un sistema de facturación, etc.

## Ejemplos de sistemas OLTP

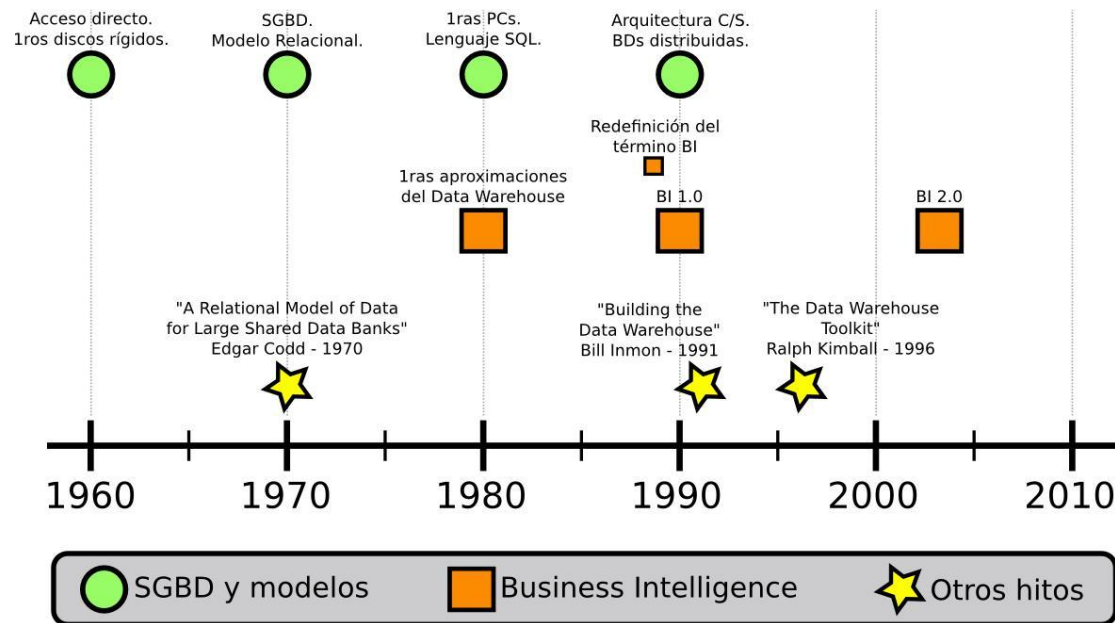
- *Cuando haces una compra en Amazon, se registra el pedido, se actualiza el stock y se genera una factura: todo eso lo hace un sistema OLTP.*
- *Cuando reservas una habitación en Booking, el sistema OLTP guarda tu reserva y bloquea la disponibilidad de esa habitación.*



## En los 80 y 90 se cambia de mentalidad...

A finales de los 80, las empresas se dan cuenta de que tienen datos... pero no respuestas.  
La competencia empieza a usar la información para tomar decisiones estratégicas.  
Nace la necesidad de un sistema que permita analizar, no solo registrar.

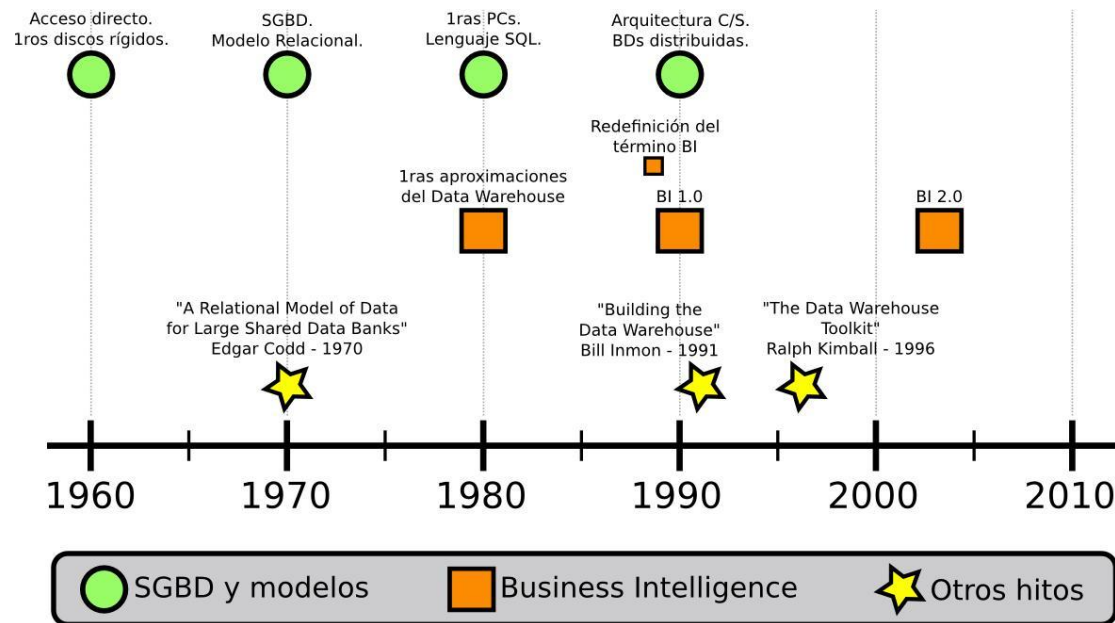
*Las empresas empezaron a preguntarse cosas como: ¿Quiénes son nuestros mejores clientes? ¿Qué productos generan más beneficios? Y se dieron cuenta de que sus sistemas no podían responder. Tenían los datos, pero no el conocimiento.*



# La aparición del Data Warehouse

En los 90 se consolidan los Data Warehouse como solución para el análisis  
Integración de datos desde distintas fuentes: OLTP, hojas Excel, sistemas externos.  
Enfoque analítico, información histórica, diseño orientado a consultas.

*Un Data Warehouse es como construir una biblioteca a partir de todos los documentos sueltos que hay por la empresa. Ya no solo tienes papeles: tienes capítulos, índices, contexto y sobre todo... claridad.*



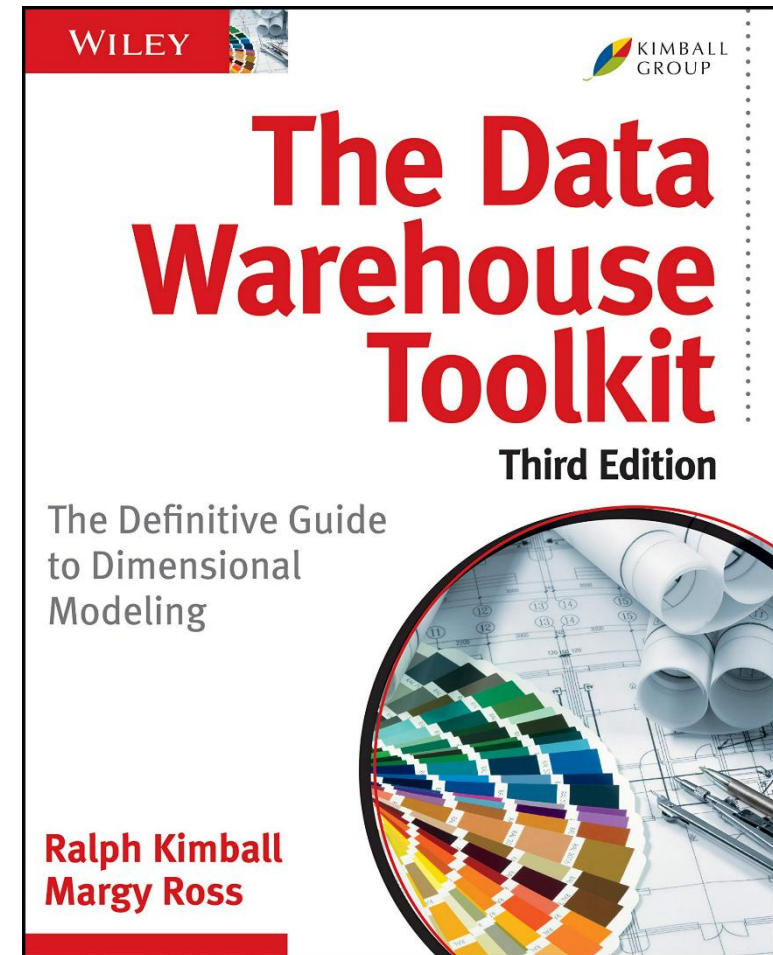
# La guía definitiva

Un libro publicado en los 90 que hace de guía para diseñar y organizar un modelo de datos analítico correctamente. El término con el que llaman a esta nueva forma de organizar la información, es **datawarehouse**. Pero no es mas que un método de organizar las tablas en la base de datos

## Propósito

- Proporcionar a los usuarios datos de forma comprensible
- Permitir que las consultas a ese modelo funcionen rápido

Como es una metodología, es independiente de cualquier tecnología o producto tecnológico. Es decir... puedo crear un datawarehouse sobre PostgreSQL, MySQL, Amazon Redshift...



# ¿Y ahora que? Del Data Warehouse al Business Intelligence

Con los Data Warehouses ya consolidados, el siguiente paso ha sido construir herramientas que exploten esa información de forma visual, ágil y en tiempo real.

Así nace el concepto de **Business Intelligence (BI)**: un conjunto de herramientas y procesos que permiten transformar los datos en conocimiento útil para tomar decisiones.

Gracias a esta evolución, hoy en día es posible que un directivo vea en su pantalla la evolución de ventas, el rendimiento de sus equipos o las desviaciones presupuestarias... sin necesidad de pedirle informes al departamento de IT.

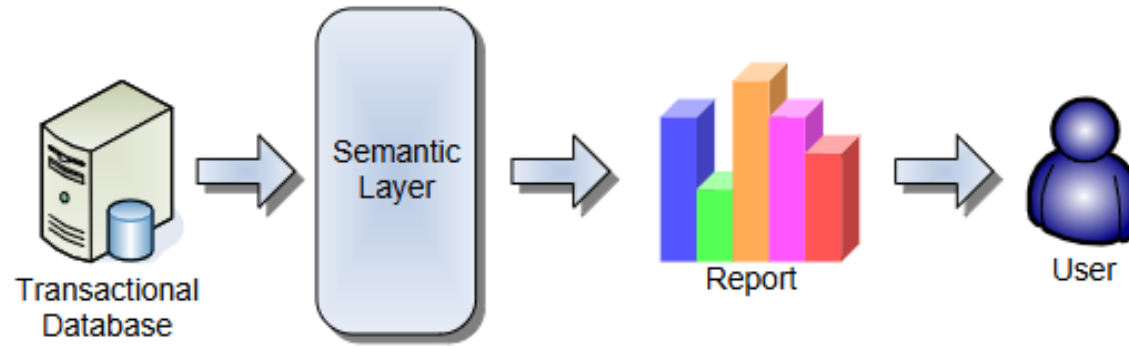


**¿POR QUÉ NECESITO UN ALMACEN DE DATOS?**



# ¿Por qué tengo que tomarme el tiempo en construir un almacén de datos?

Por qué no ataca la aplicación de análisis e informes directamente a las tablas que ya hay?



Si probásemos esto nos encontraríamos...

- Problemas de rendimiento
- Complejidad para mostrar la información y construir informes
- Dificultades en el mantenimiento

# Un almacén de datos te permite hacer cosas que normalmente no podrías hacer...

- Velocidad**

Acceso rápido a grandes volúmenes de datos para consultas y análisis.

- Facilidad de uso**

Interacción sencilla sin necesidad de ser experto en bases de datos.

- Capacidad para agregar valor**

Generación de informes detallados e insights valiosos para decisiones.

- Consolidar datos**

Integración de datos de diversas fuentes en un único lugar.

- Consistencia / Una única versión de la verdad**

Acceso a datos consistentes y actualizados por todos los usuarios.

- Gestionar la calidad de los datos**

Procesos para asegurar datos precisos y confiables.



# TIPOS DE ALMACENES DE DATOS

# Tipos de almacenes de datos



## ¿Qué es un Data Warehouse?

Un **Data Warehouse (DWH)** es una base de datos central pensada para **almacenar grandes volúmenes de datos históricos** procedentes de diferentes fuentes (ERP, CRM, archivos, APIs...).  
Piensa en él como "**el repositorio central de datos para toda la empresa**".



## ¿Qué es un Data Mart?

Un **Data Mart** es un subconjunto del Data Warehouse, más enfocado a una **unidad de negocio concreta** (por ejemplo, ventas, marketing o finanzas). Contiene solo los datos necesarios para ese departamento.  
Piensa en él como "**un trocito del DWH para un equipo o tema concreto**".

# Métodos para crear un Data Warehouse

## Kimball: Bottom - Up

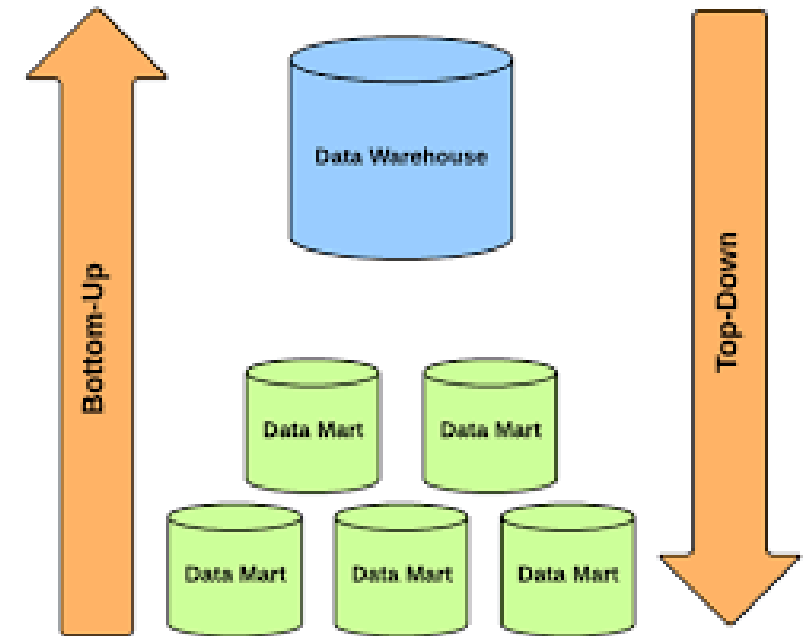
Propuesto por Ralph Kimball, muy popular en entornos de BI.  
Parte de Data Marts independientes para cada área del negocio (ventas, finanzas, etc.)

Concepto: *Primero creo soluciones pequeñas por área (Data Marts) y después las conecto para formar el DWH.*

## Immon: Top-Down

Propuesto por Bill Inmon  
Parte de un Data Warehouse centralizado y normalizado (modelo corporativo único).  
Desde ese DWH se generan Data Marts dependientes, más enfocados al negocio.

Concepto: *construyo el almacén entero y luego creo vistas Data Marts*



## Comparativa almacenes de datos

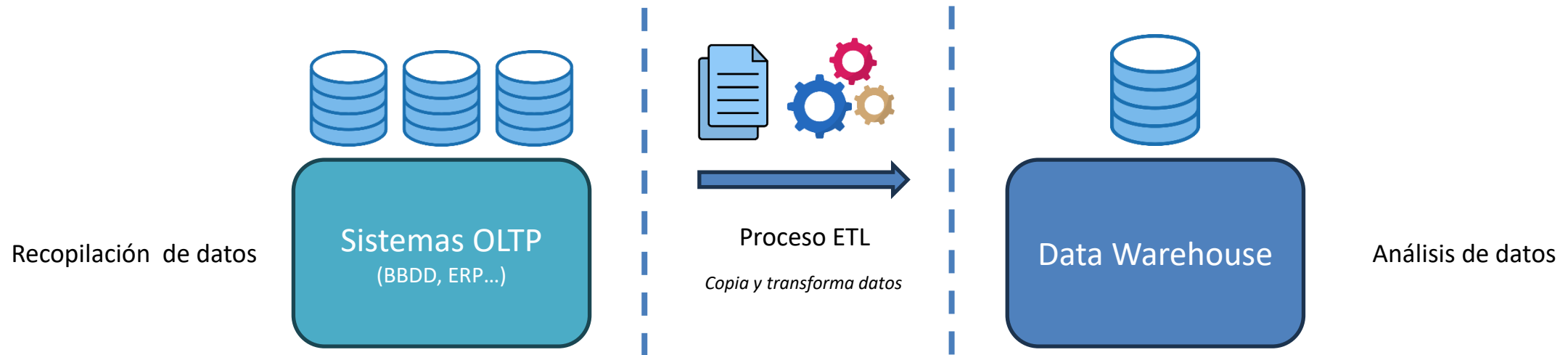
PROPIEDAD	DATAWAREHOUSE	DATAMART
<b>Alcance</b>	Corporativo	Departamental
<b>Temas</b>	Múltiples	Enfocados
<b>Fuentes</b>	Muchas	Pocas
<b>Tiempo Implementación</b>	Meses a años	Meses

# CARACTERÍSTICAS DE UN DATA WAREHOUSE



## Entonces... ¿Qué es un Data Warehouse?

Para conseguir que los datos de negocio estén bien definidos y consolidados, sean consistentes y se puedan acceder fácilmente, se estructuran y se almacenan en lo que se denomina un Data Warehouse, que son almacenes de datos corporativos que se cargan a través de un proceso de **extracción transformación y carga (ETL)**, para **posteriormente analizarlos** y obtener información relevante sobre los procesos mediante la realización de informes, estadísticas, modelos de machine learning...





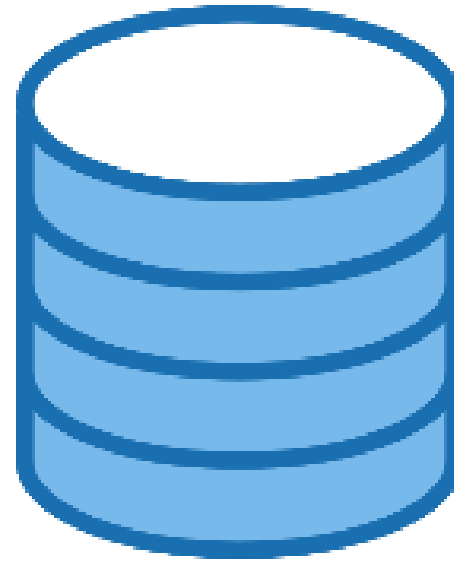
# Características de un Data Warehouse

## Orientado a temas

Un data warehouse organiza la información según los temas más importantes para una organización, como por ejemplo: clientes, productos, ventas o proveedores

## Integrado

Un data warehouse está diseñado para almacenar todos los datos importantes de una empresa, y para eso, se integran diferentes fuentes de información, como bases de datos, archivos de texto o registros de transacciones de sistemas operativos.



## Cambiante con el tiempo

Un data warehouse almacena datos no solo para mostrar lo que está pasando ahora, sino también para tener un historial de lo que ha ocurrido en el pasado

## Cambiante con el tiempo

Un data warehouse es no volátil, lo que significa que una vez que los datos son cargados, no se cambian ni se eliminan. Aunque los datos pueden ser actualizados o añadidos con nueva información, los datos antiguos siempre se mantienen tal como están

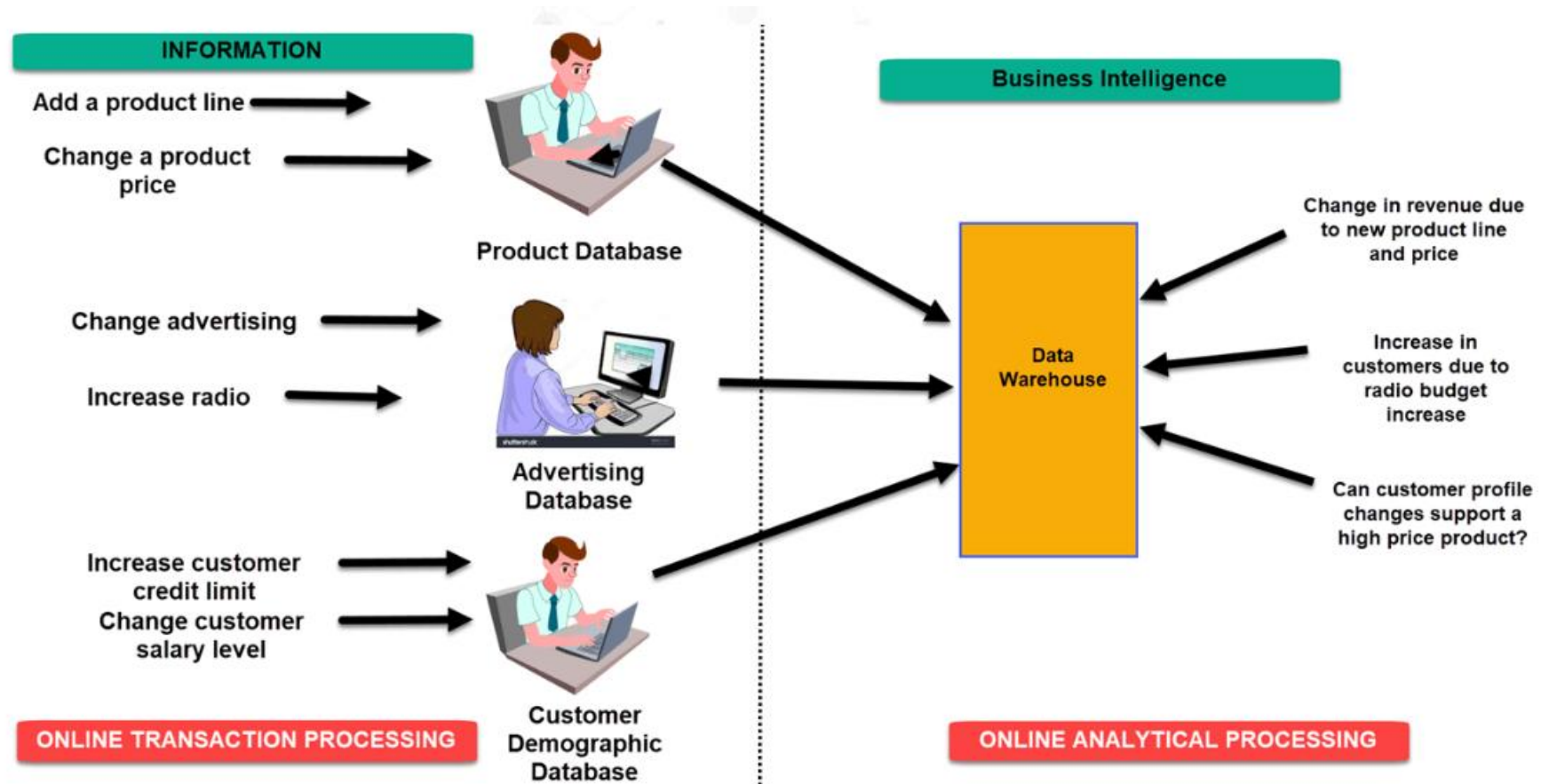


# Comparativa BBDD Convencional (OLTP) vs Data Warehouse (OLAP)

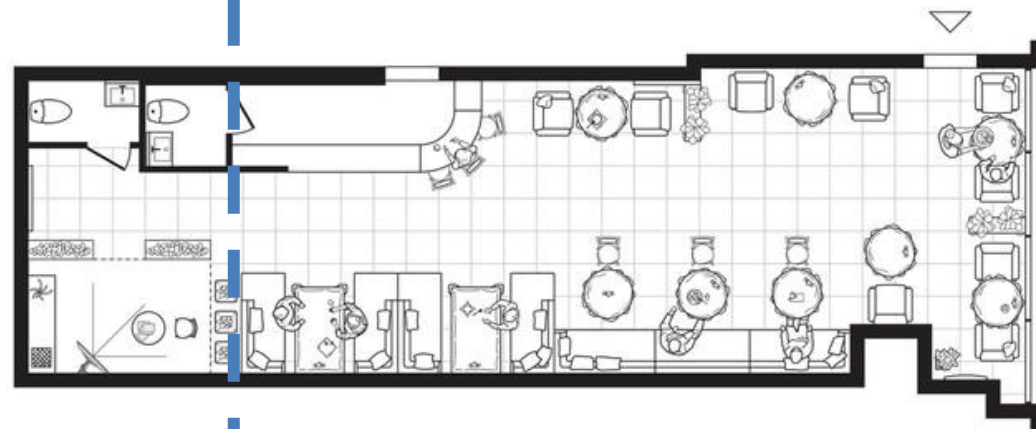
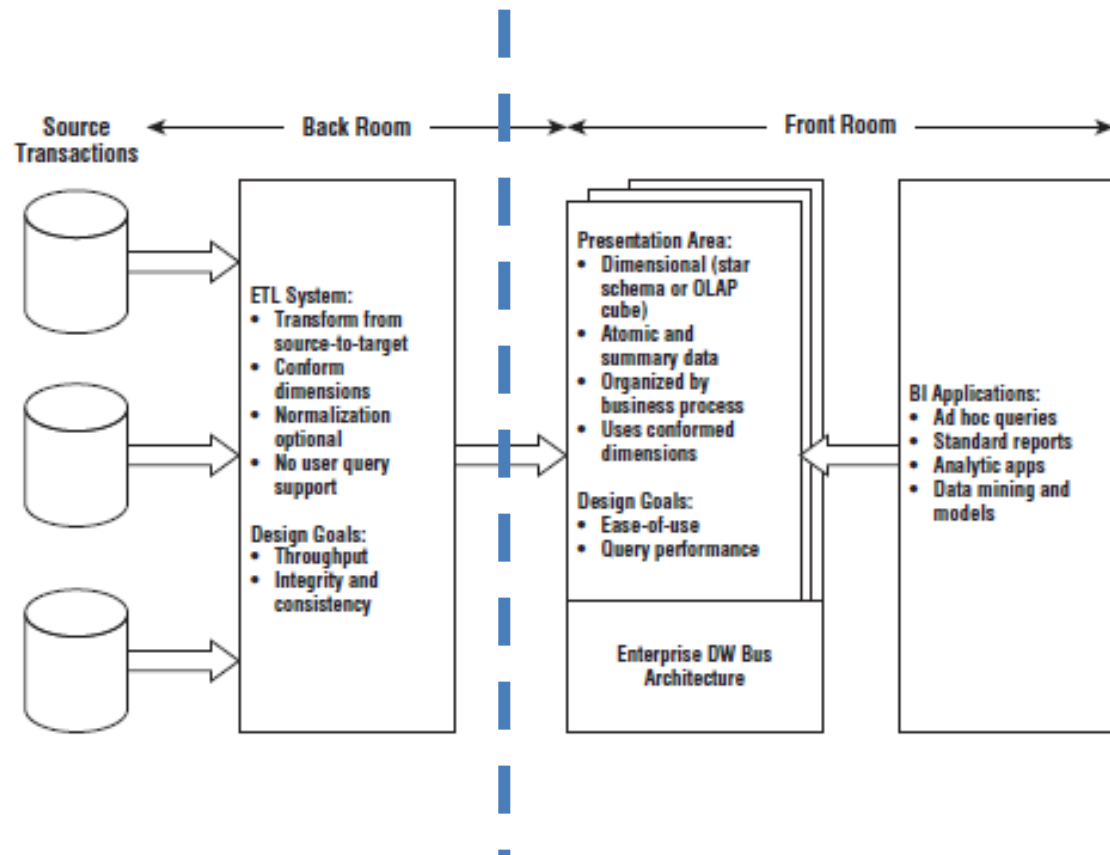
En las empresas, ambos sistemas conviven. Siempre se necesitará uno o varios sistemas para registrar toda la información de la empresa. Mientras que de cara a analizar la información, crearemos nuevos sistemas de datos que permitan el análisis de forma mas sencilla y eficiente

Categoría	OLTP	OLAP
<b>Objetivo</b>	Operaciones del día a día	Análisis y toma de decisiones
<b>Usuarios</b>	Personal operativo	Directivos, analistas
<b>Tipo de datos</b>	Actuales, muy detallados	Históricos, resumidos
<b>Diseño de datos</b>	Modelo relacional (normalizado)	Modelo estrella o copo de nieve
<b>Operaciones típicas</b>	Alta escritura (insert/update)	Lectura intensiva (consultas complejas)
<b>Volumen de datos</b>	MB a pocos GB	De decenas a cientos de TB
<b>Procesamiento de dato</b>	Rápidas, sobre pocos registros	Depende de la cantidad de datos, se recargan diariamente
<b>Tecnología asociada</b>	Bases de datos operacionales (SQL, etc.)	BI, ETL, herramientas analíticas
<b>Frecuencia de uso</b>	Constante, en tiempo real	Periódica, para análisis estratégico
<b>Backup</b>	Backup constante, ya que almacena procesos críticos para la organización	En vez de hacer backups, se suele recargar el sistema con datos si ocurre cualquier cosa

# Ejemplo real de una organización



# Metáfora de la cocina



# Características

## Características del Data Warehouse:

- Integra datos de múltiples fuentes.
- Se actualiza normalmente por la noche.
- Los analistas trabajan con una versión congelada durante el día.
- Puede estar desactualizado hasta 24h, pero esto es aceptable para análisis.

## ¿Por qué no usar el sistema transaccional (OLTP) directamente?

- Las consultas OLAP son complejas y pesadas.
- Afectan a grandes volúmenes de datos.
- Ejecutarlas en un sistema OLTP ralentizaría las operaciones normales (ej. ventas en tiempo real).

**Separar OLTP y OLAP evita conflictos de rendimiento y permite análisis complejos sin afectar a las operaciones diarias.**

# MODELADO DIMENSIONAL DE DATOS

# ¿Qué es el modelo dimensional?

El modelo dimensional, es una técnica aplicada en el diseño lógico de almacenes de datos, utilizada para modelar la información en base a indicadores de negocio que podrán ser analizados desde diferentes perspectivas o dimensiones de análisis. (Ponniah, 2001).

Los elementos que conforman al modelo dimensional son los siguientes: hecho, dimensión, jerarquías, cubo, celda, así como tablas de dimensión y hecho. (Kimball & Ross, 2002).



Habitualmente una entidad de negocio se corresponde con una tabla

**1 entidad negocio = 1 tabla**

- **Fábrica:** Operarios, Lineas de producción, Producción de piezas, Máquinas...
- **Hospital:** Visitas, Quejas, Hospitales, Doctores, Medicamentos, Recetas...
- **Productos:** Ventas, Devoluciones, Vendedores, Clientes, Marcas, Productos...

# Hay dos tipos de tablas

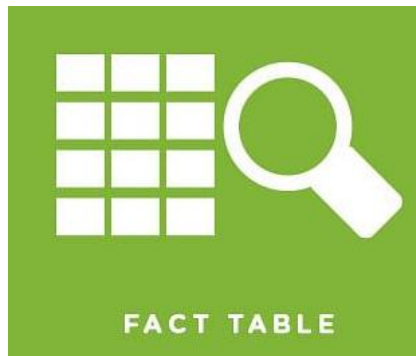
## Tablas de Hechos

“Lo que voy a medir”

- **Fábrica:** Producción de piezas
- **Hospital:** Visitas, Quejas,
- **Productos:** Ventas, Devoluciones

Hace referencia a **eventos** que pueden **medirse**. Suelen tener millones de registros y pocas columnas.

Suele contener columnas de **IDENTIFICADOR** que servirán para unirse con las dimensiones (atributos del dato)



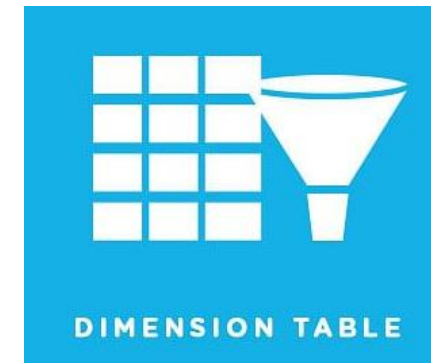
## Tablas de dimensiones

“Por lo que voy a filtrar”

- **Fábrica:** Operarios, Lineas de producción, Máquinas...
- **Hospital:** Hospitales, Doctores, Medicamentos, Recetas...
- **Productos:** Vendedores, Clientes, Marcas, Productos...

Se usan para **filtrar o segmentar** las tablas de hechos.

Suelen responder a las preguntas ¿Qué? ¿Quién? ¿Cómo? ¿Cuándo? O ¿Dónde? Sucedió el evento medido. Hace referencia a datos poco cambiantes, maestros





# Tablas de dimensión

## Descripción detallada:

- Las tablas de dimensión ofrecen información adicional sobre las medidas en la tabla de hechos (ejemplo: producto, cliente, tiempo).

## Atributos significativos:

- Incluyen descripciones detalladas para facilitar la comprensión de los datos (ejemplo: nombre del cliente, dirección).

## Gran número de atributos:

- Una tabla de dimensión puede tener entre **50 a 100 atributos** para cubrir distintas características de la dimensión.

## Pocas filas, muchas columnas:

- Suelen ser **más anchas que profundas**. Tienen muchas columnas, pero el número de filas es relativamente pequeño (menos de 1 millón).

## Clave primaria (PK):

- Cada tabla de dimensión tiene una **clave primaria** única que se usa para enlazarla con la tabla de hechos, asegurando la integridad referencial.

Product	
PK	<u>ProductKey</u>
	ProductBusinessKey ProductName Color Size Weight Group Category

Product Key	Product Business Key	Product Name	Color	Size	Weight	Group	Category
1	AC-3A4	Aero Helmet-S	Red	S	1.6 lbs	Helmets	Accessories
2	BK-IB1	Le Tour 970-M	Black	M	20.1 lbs	Road Bikes	Bikes
3	AC-3A6	Mtn. Bike Gloves-S	Black	S	0.25 lbs	Gloves	Accessories
4	BK-IB7	Le Tour 970-L	Black	L	22.5 lbs	Road Bikes	Bikes
5	AC-5B2	Racing Shorts-M	Blue	M	0.4 lbs	Shorts	Accessories

# Tablas de dimensión: ¿Qué nos podemos encontrar en una dimensión?

Product Key	Product Business Key	Product Name	Color	Size	Weight	Group	Category
1	AC-3A4	Aero Helmet-S	Red	S	1.6 lbs	Helmets	Accessories
2	BK-1B1	Le Tour 970-M	Black	M	20.1 lbs	Road Bikes	Bikes
3	AC-3A6	Mtn. Bike Gloves-S	Black	S	0.25 lbs	Gloves	Accessories
4	BK-1B7	Le Tour 970-L	Black	L	22.5 lbs	Road Bikes	Bikes
5	AC-5B2	Racing Shorts-M	Blue	M	0.4 lbs	Shorts	Accessories

## Claves Surrogadas

Una clave artificial sin significado en el negocio. Suele ser un número secuencial.

**Independiente del negocio:** no cambia si cambian las reglas o formatos del negocio

**Consistente entre tablas:** usar siempre claves numéricas facilita el diseño y el desarrollo.

**Rendimiento mejorado:** al ser más cortas, ocupan menos espacio y aceleran búsquedas e índices.

**Evita duplicidades:** se genera un nuevo valor para cada registro, útil al mover datos entre entornos (test, producción...).

**Menor mantenimiento de índices:** los valores crecen de forma secuencial, lo que reduce la fragmentación en los índices.

## Clave Natural

Es una clave que ya existe en los datos y tiene sentido en el negocio (como un DNI, email o número de cuenta).

**Tiene sentido para el usuario:** es fácil de entender y utilizar en búsquedas.

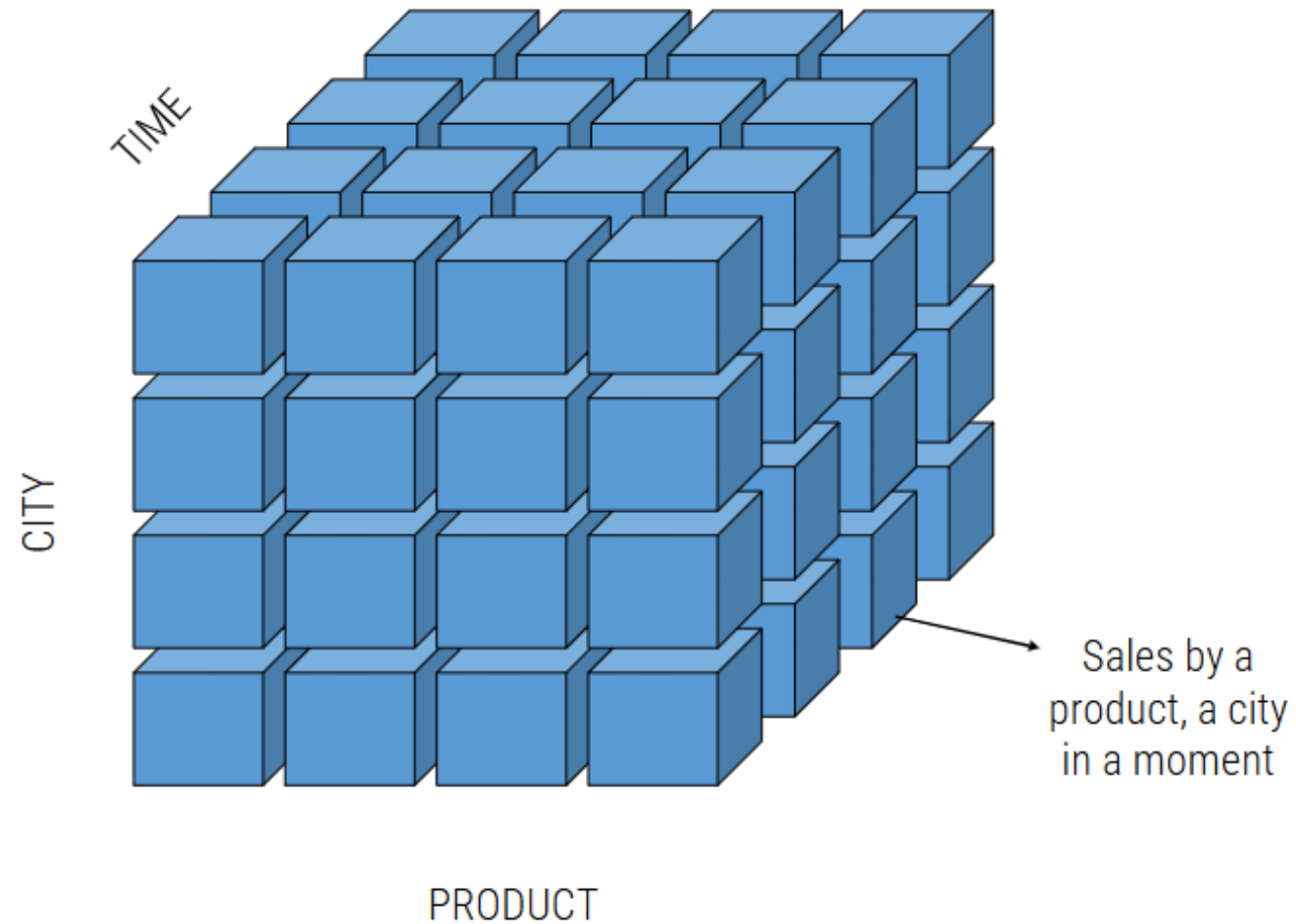
**Sin columnas extra:** ya existe en la tabla, así que no hace falta crear otra columna ni otro índice.

## Atributos

Son las columnas de una tabla de dimensión que describen los elementos del negocio.

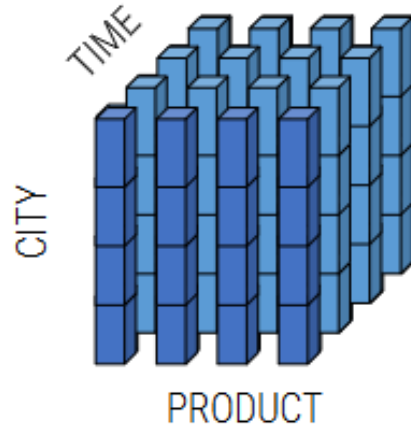
- Son los **campos descriptivos** que se usan para filtrar, agrupar o etiquetar los datos en un informe o consulta.
- En una frase como “quiero ver las ventas por semana y por marca”, los atributos son **semana** y **marca**.
- Se pueden organizar en **jerarquías** (por ejemplo: Día → Mes → Año).
- No habrá atributos con valor **null** en una dimensión, en cambio se establecerá un valor NA o No definido. Esto es así porque dependiendo de la bbdd los nulos se gestionan de formas diferentes
- Siempre suele haber una dimensión fecha

## Concepto de cubo dimensional (ejemplo con 3 dimensiones de análisis)

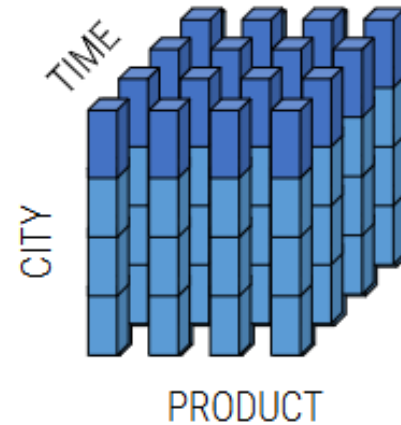


# Concepto de cubo dimensional con 3 dimensiones de análisis

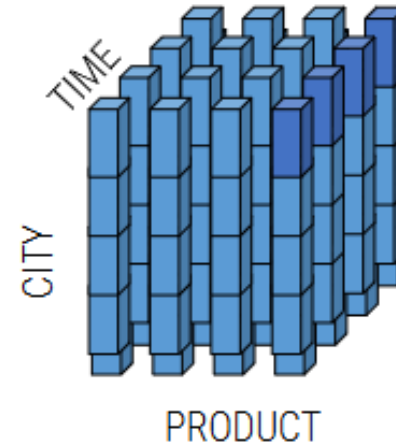
- Sales by a particular producto (example: cars)



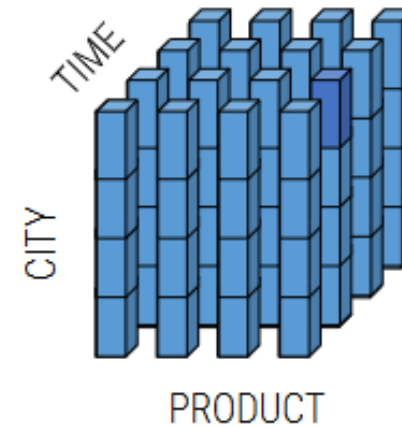
- Sales by a particular city (example: Zaragoza)



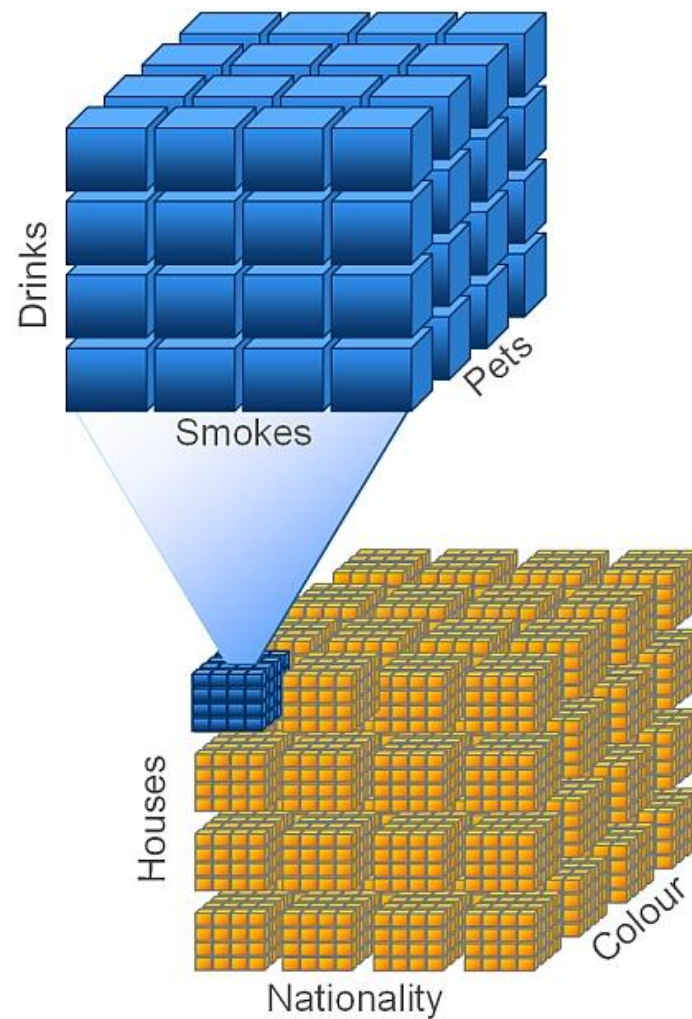
- Sales by a particular product (cars) in a given city (Zaragoza)



- Sales by time (February) for a given product (cars) in a given city (Zaragoza)



## Concepto de cubo dimensional con 6 dimensiones de análisis



# Tablas de dimensión: Jerarquía de atributos

## ¿Qué es?

- Una jerarquía es una estructura lógica que organiza los datos en niveles ordenados. Cada nivel agrupa los datos de una manera más general.

## ¿Para qué sirve?

- Agrupar datos: Permite hacer sumas o agregaciones de datos a diferentes niveles. Por ejemplo, desde el mes hasta el año en una dimensión de tiempo.
- Navegación: Permite hacer "drill-down" o profundizar en los datos para ver detalles más específicos.

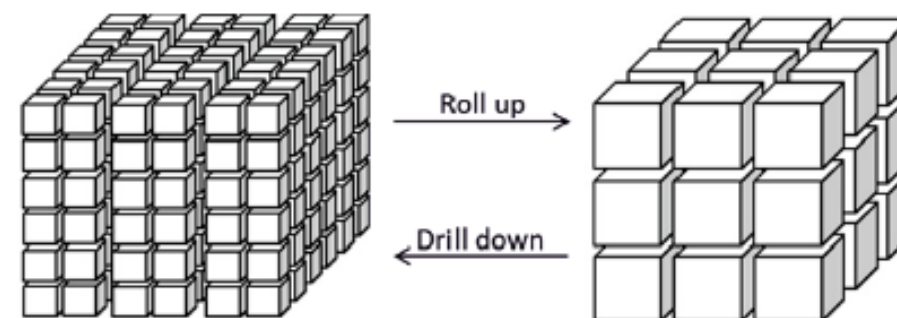
## Ejemplo de jerarquía en productos:

Categoría → Subcategoría → Producto → Proveedor

## Beneficio clave

Facilita el análisis en diferentes niveles de granularidad, desde lo más general hasta lo más específico.

Product Key	Product Description	Brand Name	Category Name
1	PowerAll 20 oz	PowerClean	All Purpose Cleaner
2	PowerAll 32 oz	PowerClean	All Purpose Cleaner
3	PowerAll 48 oz	PowerClean	All Purpose Cleaner
4	PowerAll 64 oz	PowerClean	All Purpose Cleaner
5	ZipAll 20 oz	Zippy	All Purpose Cleaner
6	ZipAll 32 oz	Zippy	All Purpose Cleaner
7	ZipAll 48 oz	Zippy	All Purpose Cleaner
8	Shiny 20 oz	Clean Fast	Glass Cleaner
9	Shiny 32 oz	Clean Fast	Glass Cleaner
10	ZipGlass 20 oz	Zippy	Glass Cleaner
11	ZipGlass 32 oz	Zippy	Glass Cleaner



# Ejercicios

**Ejercicio 1 – ¿Cuáles son atributos dimensionales?**

**Ejercicio 2 – Creación de entornos** 

**Ejercicio 3 - Tabla de dimensiones DIM PRODUCTO** 

**Ejercicio 4 – Tabla de dimensiones DIM CLIENTE** 



## Tablas de hechos

Es la tabla principal en un modelo dimensional. Contiene los datos numéricos que representan los hechos o eventos del negocio (por ejemplo, ventas, ingresos, unidades vendidas).

- Cada **fila** representa una **medición** (hecho) en un momento determinado.
- Todas las filas deben tener la misma **granularidad** (nivel de detalle).
- Las columnas suelen ser **medidas numéricas y aditivas**
- Incluye **claves foráneas** que enlazan con las tablas de dimensión.

Tipos de tablas de hechos:

- Transaccionales
- De Snapshot Periódico
- De Snapshot Acumulativo

OrderLineItems	
PK, FK1	<u>OrderDateKey</u>
PK, FK2	<u>Customer Key</u>
PK, FK3	<u>ProductKey</u>
PK, FK4	<u>RequestedDateKey</u>
PK	<u>OrderNumber</u>
PK	<u>LineNumberItem</u>
OrderQuantity OrderAmount Shipping/Amount	

Order Date Key	Customer Key	Product Key	Requested DateKey	Order Number	LineItem Number	Order Quantity	Order Amount	Shipping Amount
236	27	1	250	7867	1	2	49.98	3
236	27	3	250	7867	2	1	12.99	3
236	39	2	257	7868	1	1	1,321.99	132
236	19	5	243	7869	1	1	39.99	6
236	19	2	257	7870	1	1	1,321.99	132




## Tablas de hechos: Transaccional

Una tabla de hechos transaccional es la más básica y detallada de todas. Cada fila representa una transacción individual, sin resumir ni agregar nada. Una fila por evento.

**Ejemplo:** Ejemplo típico: Cada línea de una venta en un ticket de supermercado.

### ¿Qué suele tener este tipo de tabla?

- Una **clave primaria** que identifica la transacción
- Una clave primaria que identifica la transacción
- Muchas claves foráneas a dimensiones: producto, cliente, empleado...
- Métricas de detalle, como unidades vendidas, importe, descuentos

 **Uso:** Permite analizar comportamiento a nivel fino: por cliente, por producto, por día...

id_linea	fecha_venta	id_ticket	id_producto	cliente_id	cantidad	precio_unitario	total
1	2025-04-01	T1001	P001	C001	2	1,20 €	2,40 €
2	2025-04-01	T1001	P002	C001	1	1,00 €	1,00 €
3	2025-04-02	T1002	P003	C002	3	0,90 €	2,70 €

## Tablas de hechos: Periodic Snapshot

Una tabla de hechos periódica toma una especie de foto resumen del negocio en un momento concreto, por ejemplo: cada día, semana o mes.

**Ejemplo:** Imagina que queremos ver cómo ha ido cada vendedor cada mes. ! Esto no es un dato puntual, sino un resumen mensual que viene de sumar o contar registros de la tabla transaccional

### 12 34 ¿Qué suele tener este tipo de tabla?

- Una fecha representativa del periodo
- Claves de dimensión
- Métricas agregadas (ventas, clientes, beneficios...)

 **Uso:** seguimiento del rendimiento con periodicidad

### Tabla de hechos snapshot

FechaMes	ID_Vendedor	Nombre	VentasTotales	NºPedidos	NºClientesNuevos
2025-01-01	101	Laura	1.000 €	3	2
2025-01-01	102	Marcos	400 €	1	1
2025-02-01	101	Laura	1.000 €	2	2
2025-02-01	102	Marcos	1.100 €	2	2

### Origen Transaccional del que salen los datos

ID_Pedido	FechaVenta	ID_Vendedor	NombreVendedor	ClientelID	Importe
10001	2025-01-03	101	Laura	C001	300 €
10002	2025-01-04	102	Marcos	C002	400 €
10003	2025-01-10	101	Laura	C003	500 €
10004	2025-01-15	101	Laura	C001	200 €
10005	2025-02-01	102	Marcos	C004	600 €
10006	2025-02-08	101	Laura	C005	700 €
10007	2025-02-12	102	Marcos	C006	500 €
10008	2025-02-28	101	Laura	C001	300 €

Laura - Enero

## Tablas de hechos: Snapshot acumulativo

Una tabla de hechos acumulativa se usa para representar procesos que tienen un inicio claro y un final definido.

**Ejemplo típico:** El proceso de gestión de un pedido.

! En vez de crear una fila nueva cada vez que el pedido avanza, **se crea una única fila por pedido y se va actualizando** a medida que se completan esas etapas.

### ¿Qué suele tener este tipo de tabla?

- Varias columnas de fechas, una por cada hito
- Un campo que indique el estado actual
- Al principio, muchas fechas estarán vacías porque aún no se ha llegado a esa etapa.

 **Uso:** seguimiento de un proceso

PedidoID	FechaPedido	FechaPreparación	FechaEnvío	FechaEntrega	EstadoActual
1001	01/04/2025	03/04/2025	NULL	NULL	Preparando

Luego, cuando el pedido se envía:

PedidoID	FechaPedido	FechaPreparación	FechaEnvío	FechaEntrega	EstadoActual
1001	01/04/2025	03/04/2025	05/04/2025	NULL	Enviado

Y cuando se entrega:

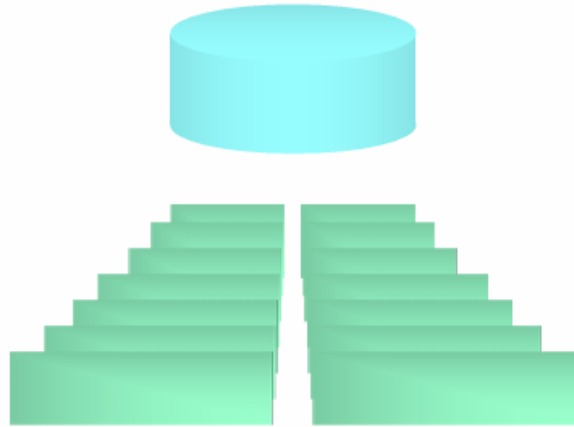
PedidoID	FechaPedido	FechaPreparación	FechaEnvío	FechaEntrega	EstadoActual
1001	01/04/2025	03/04/2025	05/04/2025	06/04/2025	Entregado

# Tablas de hechos: ¿Qué tipo de “hechos” hay en las tablas de hechos?

Tipo de Hecho	¿Qué significa?	Ejemplo	¿Se puede sumar?
Aditivo	Se puede sumar en todas las dimensiones	Ventas diarias de un producto	Sí, se suman todas las ventas
Semi-aditivo	Se puede sumar solo en algunas dimensiones (por ejemplo, por día)	Nº de personas por día suscritas a Netflix	Sí, pero solo en dimensiones específicas
No aditivo	No se puede sumar, son ratios o porcentajes	Porcentaje de descuento	No, no tiene sentido sumar porcentajes

# Granularidad

En función del caso de análisis, tendremos una granularidad alta o baja. **Depende del caso de negocio, es decir, depende de lo que los usuarios finales necesiten para sus análisis**



**Granularidad fina:** detalle de todas las transacciones de todos los clientes del banco por mes



**Granularidad gruesa:** resumen del saldo bancario a final de mes de los clientes del banco.



# Ejercicios

**Ejercicio 5 – Distinguir tipos de hechos: aditivo, semiaditivo, no aditivo**

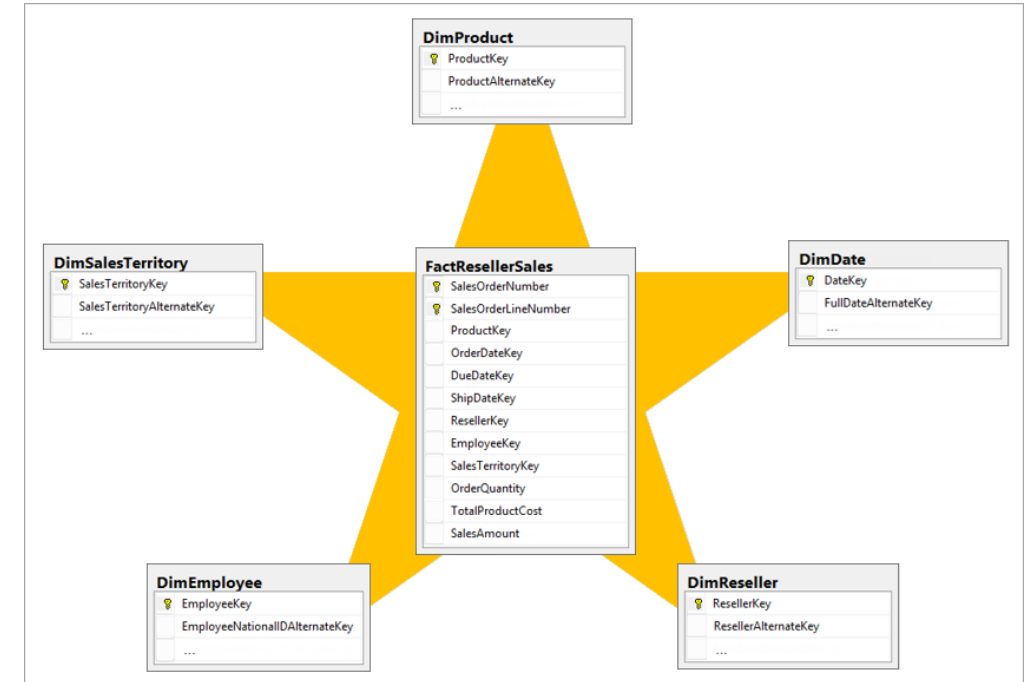
**Ejercicio 6 – Análisis de hechos**

**Ejercicio 7 – Tabla de hechos de ventas**



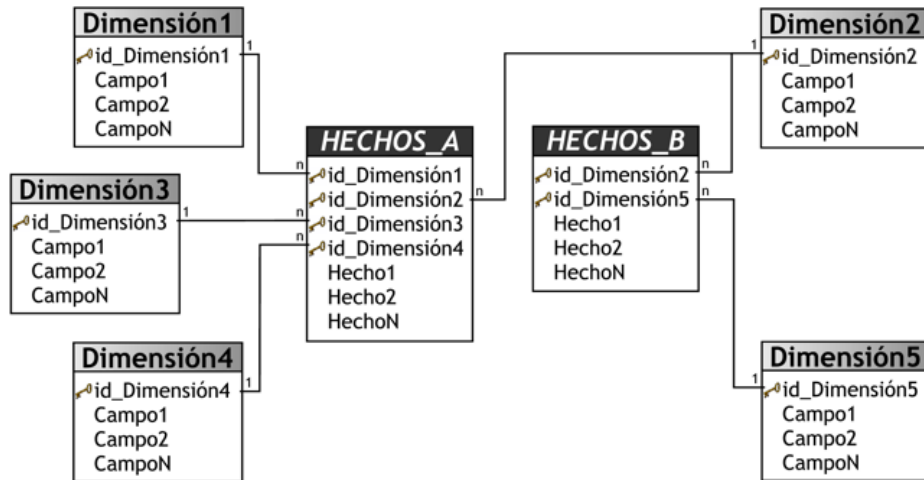
# Esquema Estrella

- 1.Tabla de hechos central** con datos numéricos (ventas, unidades...) y claves hacia las dimensiones.
- 2.Tablas de dimensiones** alrededor, con info descriptiva (producto, cliente, fecha...).
- 3.Relaciones directas** entre hechos y dimensiones, como si fueran los puntos de una estrella 🌟.
- 4.Diseño simple y orientado al análisis**, fácil de entender y muy usado en BI
- 5.Puede haber redundancia** en las dimensiones, pero se prioriza la facilidad para el análisis.

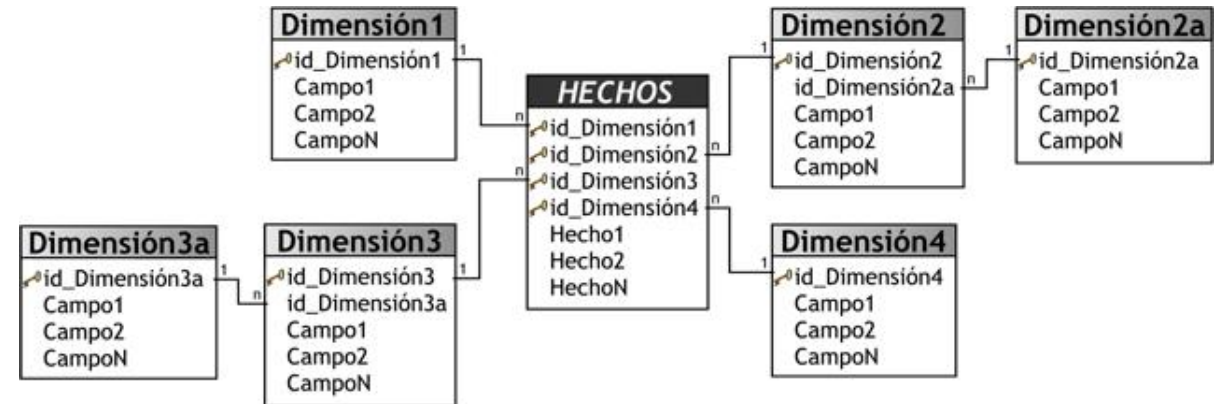


# Alternativas al modelo estrella

Esquema Constelación ✨



Esquema Copo de Nieve ❄️





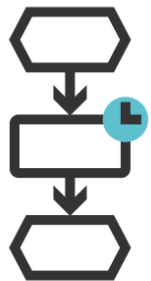
# DISEÑO DE UN DATA WAREHOUSE

## Proceso de diseño de un Data Warehouse

**Paso 1.** Elegir los procesos o actividades de negocio a modelar o analizar



Proceso: actividad de la empresa soportada en una base de datos relacional (OLTP) de la cual se puede extraer información para crear un almacén de datos



Implica analizar las tablas y sus relaciones

**Paso 2.** Definir la granularidad (nivel de detalle para representar el proceso)



Granularidad: nivel de detalle de la información a almacenar sobre el proceso que vamos a modelar

Identificar lo que se desea medir:

- Ayuda a definir el nivel atómico de los datos en el almacén de datos
- Ayuda a determinar las dimensiones básicas del modelo de datos
- Ayuda a saber cómo se relacionarán los datos en la tabla de hechos

## Proceso de diseño de un Data Warehouse

**Paso 3.** Pensar en las dimensiones que caracterizan el proceso



Dimensiones: las dimensiones que caracterizan la actividad al nivel de detalle (granularidad) que se ha elegido

**DEFINIR LAS TABLAS DE DIMENSIONES**

**Paso 4.** Definir información a almacenar sobre el proceso



Hechos: información (sobre la actividad) que se desea almacenar en cada tupla de la tabla de hechos y que será el objeto del análisis

**DEFINIR LA TABLA DE HECHOS**

**! TIPS**

- Claves primarias sin significado
- Evitar normalizar (no copo nieve)
- Dimensión tiempo

# Ejercicio: Diseño de un Data Warehouse para una Universidad 🎓

🎓 Una universidad quiere diseñar empezar a analizar sus datos y crear un modelo de datos que pueda evolucionar correctamente con el tiempo

## 🎯 Objetivo

Diseñar **un modelo en estrella** que permita analizar el **rendimiento académico de los estudiantes**:

- Qué notas sacan en los exámenes
- En qué asignatura
- Con qué profesor
- En qué curso...



# RESUELTO: Diseño de un Data Warehouse para una Universidad 🎓

¿Qué es lo que se quiere analizar?

*Analizar el rendimiento académico de los estudiantes*

¿Qué datos necesitaremos?

*Información de las notas de los exámenes. El hecho principal que vamos a analizar son las notas finales alumnos.*

Para cada nota final, necesitamos saber:

- *¿Quién es el **estudiante**?*
- *¿Qué **asignatura** cursó?*
- *¿Quién fue el **profesor**?*
- *¿En qué **semestre** o **año** se cursó?*
- *¿Qué **convocatoria** se usó (primera, segunda, etc.)?*

Identificación de dimensiones y hechos:

- *Hechos: Las notas finales de los estudiantes en las asignaturas*
- *Dimensiones: Estudiantes, Asignaturas, Profesores, Tiempo, Convocatoria...*



# RESUELTO: Diseño de un Data Warehouse para una Universidad 🎓

Estructura de las tablas y relaciones entre campos



Preguntas que se pueden responder

- ¿Cuál es el **promedio de las notas finales** de los estudiantes en cada asignatura?
- ¿Cómo varían las **notas finales** de los estudiantes según el **profesor**?
- ¿Qué **porcentaje de estudiantes aprueban o suspenden** en cada **convocatoria**?
- ¿Cómo es el rendimiento de los estudiantes por **titulación** o por **año**?
- ¿Cuántos **estudiantes por profesor** y asignatura han pasado en la **primera convocatoria**?



# Ejercicio: Diseño de un Data Warehouse para una Universidad 🎓

🎓 Una universidad quiere diseñar empezar a analizar sus datos y crear un modelo de datos que pueda evolucionar correctamente con el tiempo

## 🎯 Objetivo

Diseñar **un modelo en estrella** que permita analizar el **la asistencia a clase en las diferentes asignaturas**:

- Qué estudiantes asisten
- Asignaturas a las que asisten los alumnos
- Con qué profesores
- En qué aula
- En qué fechas han asistido mas...



# RESUELTO: Diseño de un Data Warehouse para una Universidad 🎓

¿Qué es lo que se quiere analizar?

¿Qué datos necesitaremos?

Para cada nota final, necesitamos saber:

Identificación de dimensiones y hechos:

- *Hechos:*
- *Dimensiones:*





# TÉCNICAS DE MODELADO

# Matriz para el diseño de un DWH a gran escala: Enterprise Bus Matrix

La **Enterprise Bus Matrix** es un concepto para construir un almacén de datos (DW) empresarial, basado en la metodología **Kimball**.

En términos sencillos, es una matriz que se refiere a un conjunto de reglas y estándares que permiten que diferentes áreas del negocio (y sus respectivos modelos dimensionales) se conecten de manera coherente dentro de un entorno de **Business Intelligence (BI)**.

Dimensión / Proceso	Ventas	Inventarios	Pedidos
Cliente	X		X
Producto	X	X	X
Tiempo	X	X	X
Tienda	X	X	
Ubicación		X	

# Matriz para el diseño de un DWH: Enterprise Bus Matrix

### 1. Hecho de Ventas

Este hecho se refiere a la transacción de venta de un producto a un cliente. Los hechos de ventas suelen tener datos como el **número de unidades vendidas**, el **precio total** o el **descuento aplicado**.

ID Venta	ID Cliente	ID Producto	Fecha	Unidades Vendidas	Total Venta (€)
1	101	1001	2025-04-01	2	1200
2	102	1002	2025-04-02	1	300

### 2. Hecho de Inventarios

Este hecho refleja la cantidad de stock disponible de un producto en un momento determinado. Los hechos de inventario suelen contener información sobre la cantidad en stock y el **valor total del inventario**.

ID Producto	Fecha	Stock Disponible	Valor Total Inventario (€)
1001	2025-04-01	50	5000
1002	2025-04-01	200	3000

### 3. Hecho de Pedidos

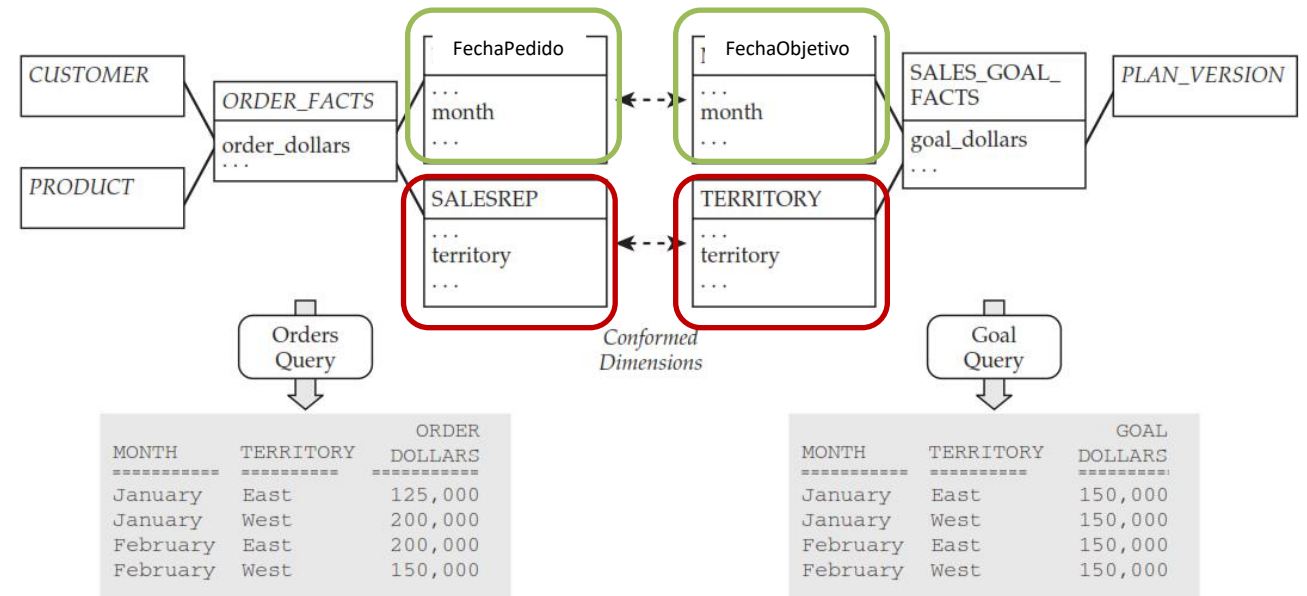
Este hecho está relacionado con el proceso de creación de un pedido de compra de productos, con detalles sobre la **cantidad pedida** y el **costo del pedido**.

ID Pedido	ID Cliente	ID Producto	Fecha Pedido	Unidades Pedidas	Costo Pedido (€)
1	101	1001	2025-04-01	5	250
2	103	1002	2025-04-02	10	1500

Dimensión / Proceso	Ventas	Inventarios	Pedidos
Cliente	X		X
Producto	X	X	X
Tiempo	X	X	X
Tienda	X	X	
Ubicación		X	

## Otras tablas: Dimensiones conformadas

- **Definición sencilla:** Es una tabla que se usa de manera consistente en diferentes áreas del negocio para facilitar el análisis de datos.
- **Propósito:** Permite comparar datos de diferentes áreas usando la misma estructura y valores.
- Habitual en los **modelos de constelación**



## Otras tablas: Role playing dimensions

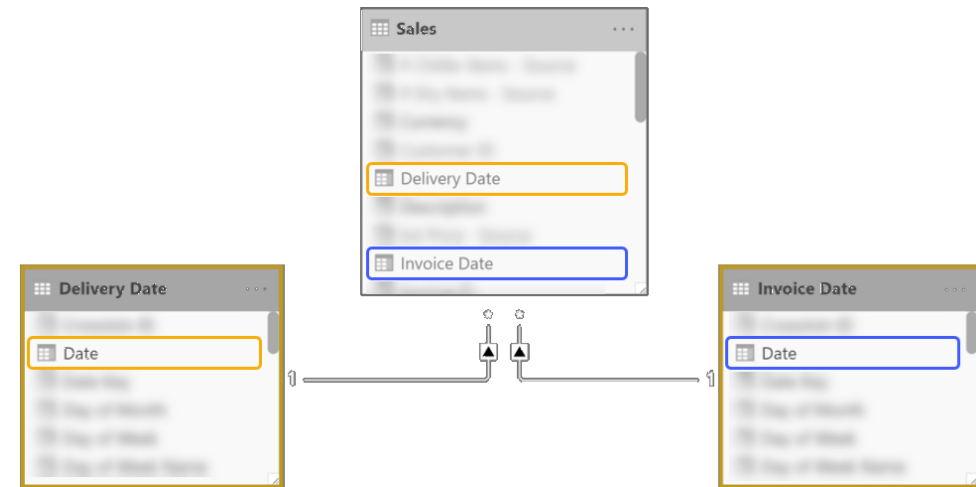
A menudo, en un almacén de datos existen **temas consistentes** que requieren dimensiones separadas pero basadas en los mismos datos.

### Ejemplos Comunes:

- **Tiempo:** Una fecha puede representar diferentes eventos, como fecha de envío, fecha de pedido o fecha de recepción.
- **Ubicaciones:** Se pueden tener dimensiones separadas para **origen** y **destino**, aunque ambos se basan en la misma información.

### Concepto

- Las dimensiones representan roles distintos dentro de la información, pero se basan en los mismos datos.
- Se crea una tabla maestra única para gestionar todos los datos.
- Usar vistas y nombrar distintamente las tablas y los atributos para proporcionar los roles.



## Otras tablas: Dimensión degenerada

Son **atributos clave** que:

- Son **importantes para analizar** o identificar datos.
- Pero **no tienen más información adicional** para justificar una tabla de dimensión propia, ya son una columna.



### Ejemplo: Facturas

Tenemos una **factura** con varias líneas (productos vendidos).

Queremos **guardar el número de factura** para hacer filtros, agrupaciones, etc. pero NO creamos una dimensión a propósito para esto

Factura	Producto	Cliente	Fecha	Cantidad	Precio
F001	P001	C001	2025-04-01	2	50€
F001	P002	C001	2025-04-01	1	30€
F002	P003	C002	2025-04-02	5	10€
F003	P002	C003	2025-04-03	3	30€
F003	P004	C003	2025-04-03	1	80€

## Otras tablas: Junk Dimensions

En muchas fuentes transaccionales hay campos sueltos como:

- Flags (Sí/No)
- Indicadores (Activo/Inactivo)
- Tipologías simples (Tipo cliente, Canal, etc.)

Estos campos no justifican una dimensión propia.  
Tampoco deben ir sueltos en la tabla de hechos.  
Pero pueden ser útiles para análisis.

💡 **¿Qué hacemos?** Creamos una **Junk Dimension**:

- Una **dimensión que agrupa todos estos campos** pequeños.
- Combinamos sus valores posibles y les damos un **ID único**.
- Así limpiamos el modelo y **mantenemos la información**.

### 🎯 Ventajas

- Reduce el número de dimensiones.
- Mantiene el modelo limpio y fácil de entender.

📊 Tabla de Hechos – Ventas

ID Fact	ID Producto	ID Cliente	Fecha Venta	Cantidad	Importe (€)	ID Junk
1	P001	C001	2025-04-01	2	100	1
2	P002	C002	2025-04-01	1	50	2
3	P003	C003	2025-04-02	3	90	3
4	P001	C004	2025-04-02	1	50	4

📋 Dimensión Junk asociada

ID Junk	Promo Aplicada	Venta Online	Tipo Cliente
1	Sí	Sí	Nuevo
2	Sí	No	Recurrente
3	No	Sí	Nuevo
4	No	No	Recurrente

# NULLs en las dimensiones

Aparecen:

- Cuando **una fila de dimensión no tiene todos sus datos** (ej. cliente sin fecha de nacimiento).
- Cuando **un atributo no aplica** a todos los registros (ej. "nombre del tutor" en adultos).

## ⚠ ¿Qué problemas causan?

Los valores nulos (NULL) se comportan de forma inconsistente según la base de datos:

- Agrupar por nulos puede excluir filas.
- Los filtros a veces no devuelven los resultados esperados.

## ✅ ¿Qué se recomienda hacer?

En lugar de dejar un NULL, usar un valor de sustitución descriptivo, por ejemplo:

- "Desconocido"
- "No aplicable"
- "Sin información"



## Podemos añadir nuevas columnas a nuestras tablas

A veces queremos filtrar o agrupar dimensiones (como clientes) por su comportamiento:

- Clientes que gastaron más de 1.000 € el año pasado.
- Clientes con más de 10 compras en total.

### ✅ ¿Cómo lo resolvemos?

Calculamos métricas **agregadas** (como gasto total, pedidos...) y las añadimos como **atributos de la dimensión**.

Esto permite:

- Filtrar por rendimiento.
- Agrupar clientes en segmentos.
- Hacer análisis más directo desde la dimensión.

ID Cliente	Nombre	Total Compras (€)	Segmento
C001	Juan	2500	Alto valor
C002	Marta	800	Medio valor
C003	Pedro	150	Bajo valor

**SLOWLY CHANGING DIMENSIONS**

## Gestión de cambios en una tabla de dimensión

¿Cómo gestionar los cambios en una dimensión?

- **Tipo 0:** Ignorar el cambio  
Se conserva el valor original. Útil para atributos que no deben cambiar nunca (por ejemplo, el DNI de una persona).
- **Tipo 1:** Sobrescribir el valor  
Se actualiza el valor directamente, sin conservar el histórico. Recomendado cuando el historial no es importante (ej: corrección de errores).
- **Tipo 2:** Añadir un nuevo registro  
Se guarda una nueva fila con una versión distinta del valor, permitiendo mantener el histórico completo. Se suele usar un campo de fechas o flags de vigencia (StartDate, EndDate, IsCurrent).
- **Tipo 3:** Añadir una columna adicional.  
Se conserva el valor anterior en una columna extra (ej: EstadoAnterior). Permite comparar con la versión previa, pero no guarda histórico completo.
- **Tipo 6:** Combinación de tipo 1 y 2 (híbrido)  
Se crea una nueva fila como en el tipo 2, pero además se incluyen columnas con valores sobrescritos como en tipo 1 y/o tipo 3. Útil cuando se necesita tanto histórico como fácil acceso al valor actual.

## SCD Tipo 0: Ignorar el cambio

Los datos nunca cambian. Una vez hecha la carga inicial los valores de la dimensión no varían ni se añaden nuevos registros.

- **Ventajas:** Sencillez absoluta. No requiere almacenamiento adicional ni lógica compleja.
- **Desventajas:** No permite actualizar la información en caso de errores iniciales. Imagina que se ha cometido un fallo

Original data:	Customer SK	Customer NK	Customer Name
	1	ABC	Brian Jones
	2	DEF	Sally Baker

## SCD Tipo 1: Sobrecribir el valor

Empezamos con los datos que cambian. En este caso se actualiza el valor del atributo sobre el mismo registro, sin guardar un historial de cambios.

- **Ventajas:** Fácil de implementar. No requiere almacenamiento adicional dado que sobrescribimos en el mismo registro.
- **Desventajas:** No mantiene un historial de cambios. Puede causar inconsistencias en los análisis históricos.

Podemos usar la SCD de tipo 1 cuando un cliente actualiza su dirección y no necesitamos guardar la antigua. En la tabla de clientes, la dirección antigua se sobrescribe con la nueva. Es habitual para corregir errores y situaciones donde el historial de cambios no es relevante.

Original data:	Customer SK	Customer NK	Customer Name	AuditRow UpdateDate
	1	ABC	Brian Jones	6-4-2014
	2	DEF	Sally Baker	10-1-2015

Change to Customer Name occurs.				
Updated data:	Customer SK	Customer NK	Customer Name	AuditRow UpdateDate
	1	ABC	Brian Jones	6-4-2014
	2	DEF	<b>Sally Walsh</b>	<b>12-2-2016</b>

## SCD Tipo 2: Añadir una fila

A partir de la SCD de Tipo 2 empieza a ser relevante conservar el histórico. En este caso, añadimos una nueva fila para cada cambio, guardando el historial con fechas de inicio y fin de vigencia del atributo y una clave subrogada.

*Una clave subrogada es un identificador único artificial, normalmente una secuencia autogenerada, que no viene de la aplicación origen de los datos, por lo tanto, no tiene sentido para negocio.*

- **Ventajas:** Mantenemos el historial completo de cambios, lo que nos va a permitir el análisis del historial de datos.
- **Desventajas:** Aumenta la volumetría de la tabla. Añade algo de complejidad ya que requiere gestión de claves subrogadas y las fechas que hemos comentado antes.

Cuando cambian a la directora de un departamento por otra podría ser un ejemplo de SCD Tipo 2. En la fila de la antigua directora tendríamos las fechas en las que empezó en su puesto, en la que finalizó

Tabla: DimCliente


CustomerKey	CustomerID	Nombre	País	StartDate	EndDate
101	C001	Laura	España	2021-01-01	2023-03-15
102	C001	Laura	Alemania	2023-03-16	NULL
103	C002	Pedro	Francia	2022-06-10	NULL

## SCD Tipo 3: Añadir una columna

Para el caso de las SCD Tipo 3 se añade una columna adicional para guardar el valor anterior del atributo.

- **Ventajas:** Permite comparar el valor actual con el anterior.
- **Desventajas:** Limitado a solo dos estados; el actual y el anterior.

La misma directora de antes, la han cambiado a un nuevo departamento y queremos guardar el departamento inicial y el anterior.

 **Tabla:** DimEmpleado (Tipo 3)

EmployeeKey	EmployeeID	Nombre	DepartamentoActual	DepartamentoAnterior	FechaCambio
201	E001	Marta	Finanzas	IT	2024-05-01
202	E002	Carlos	Marketing	NULL	NULL

# SCD Tipo 4: Tabla de histórico separada

La estrategia a seguir para las SCD Tipo 4 es construir una tabla adicional para almacenar todos los cambios históricos. También tendríamos una tabla principal donde solo encontraríamos la versión actual de ese atributo.

- **Ventajas:** Mantiene un historial detallado sin sobrecargar la tabla principal.
- **Desventajas:** Añade complejidad a la gestión de la base de datos y a las consultas que podríamos necesitar.

En una empresa, podríamos querer una dimensión con los empleados actuales, pero quizá querríamos también mantener un histórico con todos los empleados

Tabla Principal: DimEmpleado (Versión actual)

EmployeeKey	EmployeeID	Nombre	DepartamentoActual	FechaCambio
201	E001	Marta	Finanzas	2024-05-01
202	E002	Carlos	Marketing	NULL

Tabla de Histórico: HistEmpleado (Historial completo)

EmployeeKey	EmployeeID	Nombre	Departamento	FechaInicio	FechaFin
201	E001	Marta	IT	2021-01-01	2024-04-30
201	E001	Marta	Finanzas	2024-05-01	NULL
202	E002	Carlos	Marketing	2022-02-01	NULL



## SCD Tipo 5: Combinación de tipos 1 y 4

Utiliza una mini-dimensión extra para registrar la versión actual relacionada con la tabla histórica para los cambios y no tener que “pasar” por la tabla de hechos para construir tu consulta.

- **Ventajas:** Combina lo mejor de los tipos 1 y 4.
- **Desventajas:** Mayor complejidad en la implementación y gestión.



## SCD Tipo 6: Combinación de los tipos 1, 2 y 3

Combina los tipos 1, 2 y 3, sobrescribiendo atributos del tipo 1, añadiendo filas para cambios significativos del tipo 2 y escribiendo el valor anterior del tipo 3. Este valor sería el mismo en el caso de tratarse del valor actual. También tendríamos un flag para indicar si el registro está en activo o no.

- **Ventajas:** Ofrece flexibilidad al combinar múltiples estrategias.
- **Desventajas:** Mayor complejidad en la implementación y gestión.

EmployeeKey	EmployeeID	Nombre	Departamento Actual	Departamento Anterior	FechaCambio	FlagActivo
201	E001	Marta	Finanzas	IT	2024-05-01	1
202	E002	Carlos	Marketing	NULL	NULL	1
203	E003	Laura	RRHH	Marketing	2023-11-01	0
204	E003	Laura	Legal	RRHH	2024-05-10	1

# SCD Tipo 7: Híbrida

Similar al tipo 6, pero con una tabla adicional que contiene solo la versión actual. Se puede acceder a la tabla de hechos mediante una dimensión de tipo 1 o una de tipo 2, la misma tabla de dimensión permite ambas perspectivas. Estas dos vistas se incluirían de forma separada en nuestro modelo semántico.

- **Ventajas:** Mejora el rendimiento en consultas.
- **Desventajas:** Mayor complejidad en la implementación y gestión.

Tabla principal

EmployeeKey	EmployeeID	Nombre	Departamento	DepartamentoAnt	FechaCambio	FlagActivo	CambiosDepto
201	E001	Marta	Finanzas	IT	2024-05-01	1	1
202	E002	Carlos	Marketing	NULL	NULL	1	1

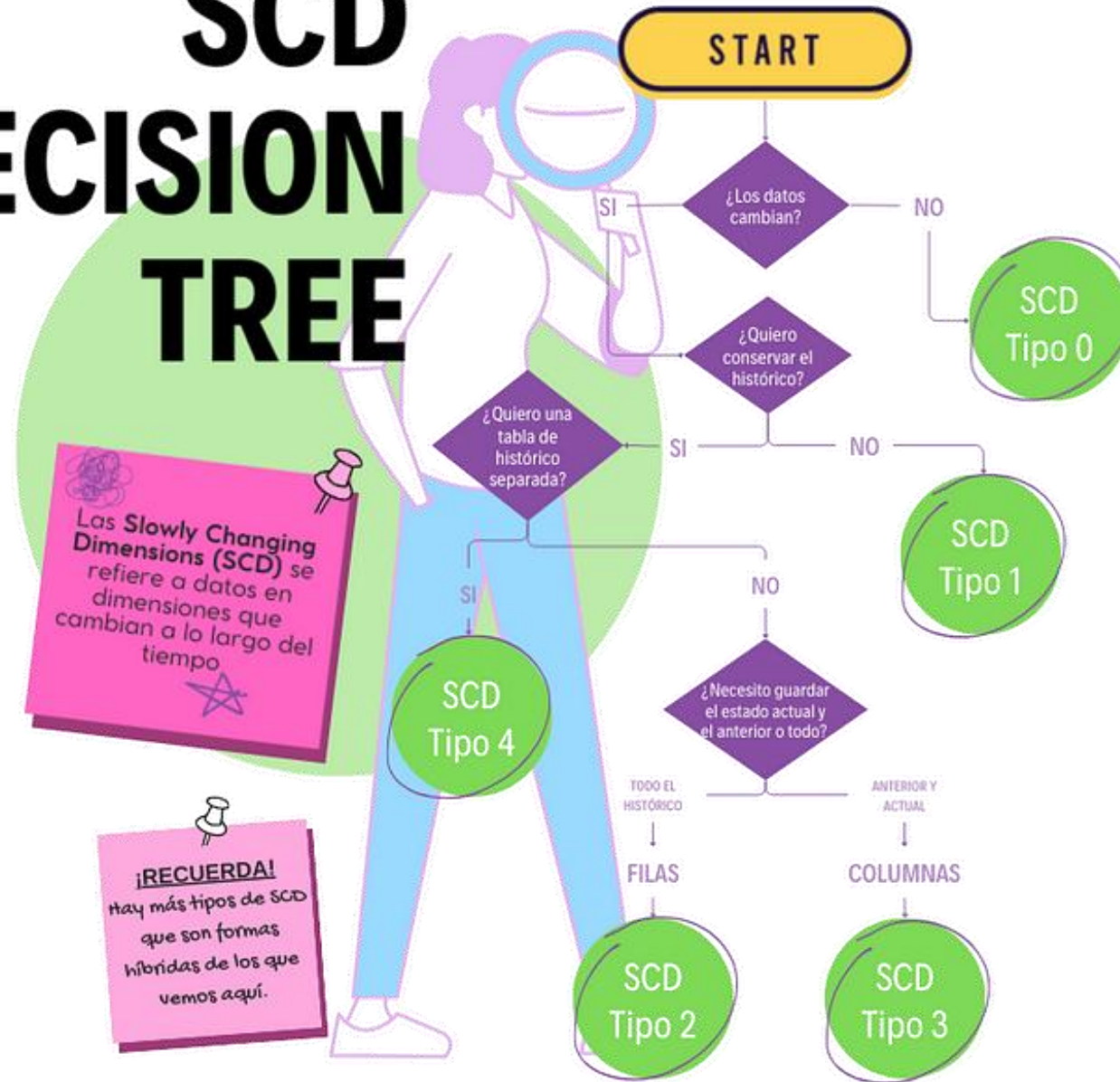
Histórico

EmployeeKey	EmployeeID	Nombre	DepartamentoHistorial	FechaInicio	FechaFin
201	E001	Marta	IT	2021-01-01	2024-04-30
201	E001	Marta	Finanzas	2024-05-01	NULL
202	E002	Carlos	Marketing	2022-02-01	NULL



# SCD DECISION TREE

**! IMPORTANTE**



# TIPOS DE SCD



# MODELADO DE TABLAS DE HECHOS

# Tablas de hechos

Es la tabla principal en un modelo dimensional. Contiene los datos numéricos que representan los hechos o eventos del negocio (por ejemplo, ventas, ingresos, unidades vendidas).

Tipos de tablas de hechos:

- Transaccionales
- De Snapshot Periódico
- De Snapshot Acumulativo

	Transaccional	Instantánea Periódica	Instantánea Acumulativa
<b>Periodicidad</b>	Punto de transacción discreto en el tiempo	Instantáneas recurrentes en intervalos regulares	Intervalo de tiempo indeterminado para un proceso/pipeline evolutivo
<b>Grano</b>	1 fila por transacción o línea de transacción	1 fila por período de instantánea más otras dimensiones	1 fila por ocurrencia del pipeline
<b>Dimensión de fecha(s)</b>	Fecha de transacción	Fecha de la instantánea	Múltiples fechas para los hitos clave del pipeline
<b>Hechos</b>	Rendimiento de la transacción	Rendimiento acumulado para el intervalo de tiempo	Rendimiento por ocurrencia del pipeline
<b>Sparsidad de la tabla</b>	Dispersa o densa, según la actividad	Predeciblemente densa	Dispersa o densa, según la ocurrencia del pipeline
<b>Actualizaciones tabla</b>	Sin actualizaciones, salvo correcciones	Sin actualizaciones, salvo correcciones	Se actualiza cuando ocurre una actividad en el pipeline

## Tablas de hechos sin hechos numéricos (I)

Es una tabla de hechos que no contiene medidas numéricas, pero sí registra la ocurrencia (o no) de un evento mediante claves foráneas a las dimensiones.

Aunque normalmente en una tabla de hechos ponemos métricas como "ventas", "importe", "tiempo", etc., a veces solo necesitamos saber que algo ocurrió, sin medir nada.

### Ejemplo típico:

Imagina una situación como esta:

| Un alumno asiste a una clase un día determinado.

No hay una métrica que necesitemos registrar (no hay nota, ni tiempo, ni coste), pero **sí queremos dejar constancia del evento.**

Fecha	AlumnoID	ClaseID	ProfesorID	AulaID
2025-04-21	123	456	789	A1



# Tablas de hechos sin hechos numéricos (II)

🔍 ¿Y cómo analizamos lo que *no* ocurrió?

Aquí entra otro uso típico: **análisis de cobertura**.

## Ejemplo:

Queremos saber qué alumnos NO asistieron a clase.

1. Creamos una **tabla de cobertura**: contiene todas las combinaciones posibles (por ejemplo, todos los alumnos y todas las clases del semestre).

2. Luego tenemos la **tabla de actividad real**: lo que sí ocurrió (asistencias reales).

3. Hacemos una resta entre cobertura y actividad → y obtenemos quiénes *no fueron* a clase.

### 1 Tabla de cobertura (todo lo que *podría* haber pasado)

Esta tabla contiene todas las combinaciones posibles entre alumnos, clases y fechas.

Fecha	AlumnoID	ClaseID
2025-04-21	A1	C1
2025-04-21	A1	C2
2025-04-21	A2	C1
2025-04-21	A2	C2

### 2 Tabla de actividad real (lo que *sí* pasó)

Esta tabla contiene las asistencias reales (es nuestra factless fact table de eventos ocurridos):

Fecha	AlumnoID	ClaseID
2025-04-21	A1	C1
2025-04-21	A2	C2

### 3 Resta lógica: Cobertura - Actividad

Queremos saber qué filas están en cobertura pero no en actividad.

Fecha	AlumnoID	ClaseID	¿Asistió?
2025-04-21	A1	C2	✗ No
2025-04-21	A2	C1	✗ No

## Conozco lo básico del modelado de datos Kimball si...

- ☐ **Entiendo qué es el enfoque Kimball:** Un modelo de datos centrado en el negocio, fácil de consultar y optimizado para rendimiento.
- ☐ **Conozco los componentes principales de un modelo en estrella:**
  - ☐ Tablas de hechos: Tablas que contienen los datos medibles (hechos).
  - ☐ Tablas de dimensión: Tablas que contienen las descripciones de las dimensiones (atributos contextuales).
- ☐ **Sé cómo se organiza un modelo en estrella:** Las tablas de hechos están en el centro, rodeadas por las Dimension Tables, relacionadas directamente.
- ☐ **Entiendo para qué sirven las SCD (Slowly Changing Dimensions):** Las SCD gestionan cómo los atributos de las dimensiones cambian con el tiempo, utilizando técnicas como SCD Tipo 1, Tipo 2 y Tipo 3.
- ☐ **Sé por qué la desnormalización es importante en el modelo Kimball:** Mejora el rendimiento de las consultas al reducir la necesidad de uniones (joins) complejas.
- ☐ **Entiendo qué es la granularidad en un modelo en estrella:** La granularidad define el nivel de detalle de los datos en las Fact Tables (por ejemplo, ventas por día, por cliente, por producto).
- ☐ **Conozco el proceso de modelado Kimball y sabría aplicarlo**



**Fundación  
iberCaja**

