

# Diseño Lógico de Datos

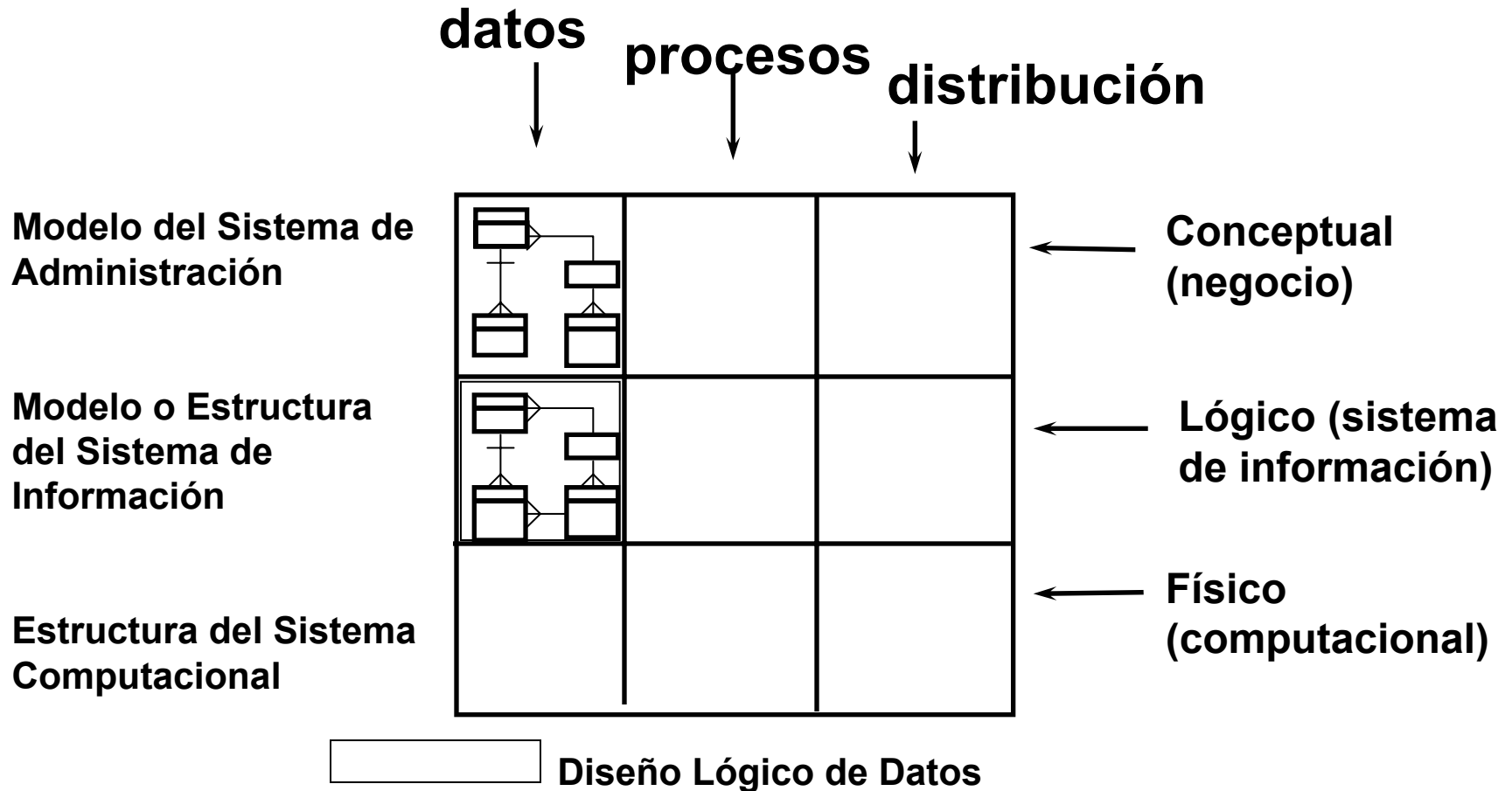
## Clase 10

---

Sistemas de Información Administrativos  
Departamento de Ingeniería Industrial  
Universidad de Chile  
derechos reservados ©

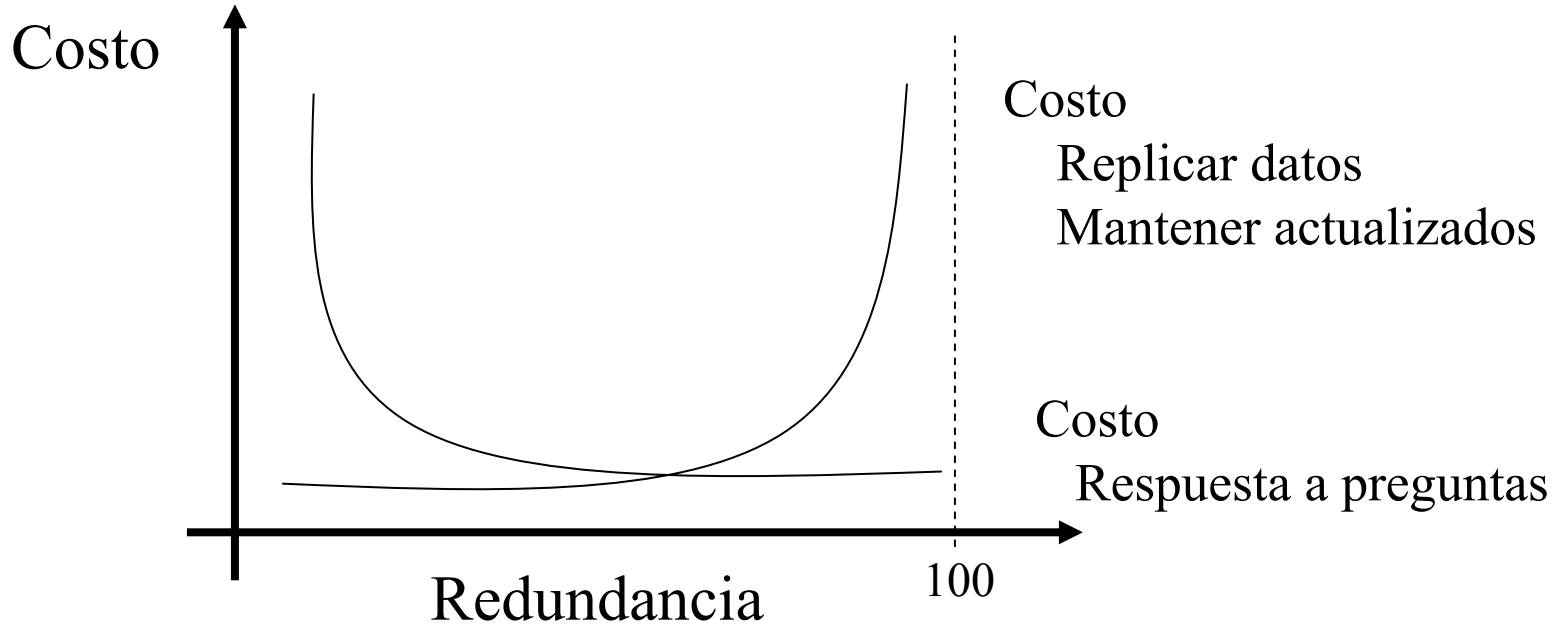
# Arquitectura de sistemas

---



# Diseño Lógico de Datos

---



- ❑ Generar entidades y relaciones adicionales basadas en las existentes
- ❑ Es necesario crearlas y mantenerlas en base a cambios en las originales
- ❑ Las respuestas son más eficientes

# Diseño Lógico

---

- Apunta a buscar **eficiencia** en el modelo lógico, o sea que además de satisfacer los requerimientos (efectividad) lo hagan en forma eficiente o al menos respetando las restricciones de tiempo de respuesta, etc.
- Para ello deben analizarse en detalle los atributos de los requerimientos:
  - Frecuencia o periodicidad.
  - Tiempo de respuesta.
  - Nivel de complejidad.
  - Identificadores de ordenamiento y de búsqueda.
  - Volumen (importante en la actividad de los datos).
  - Medio (disco, diskette, cinta, etc).

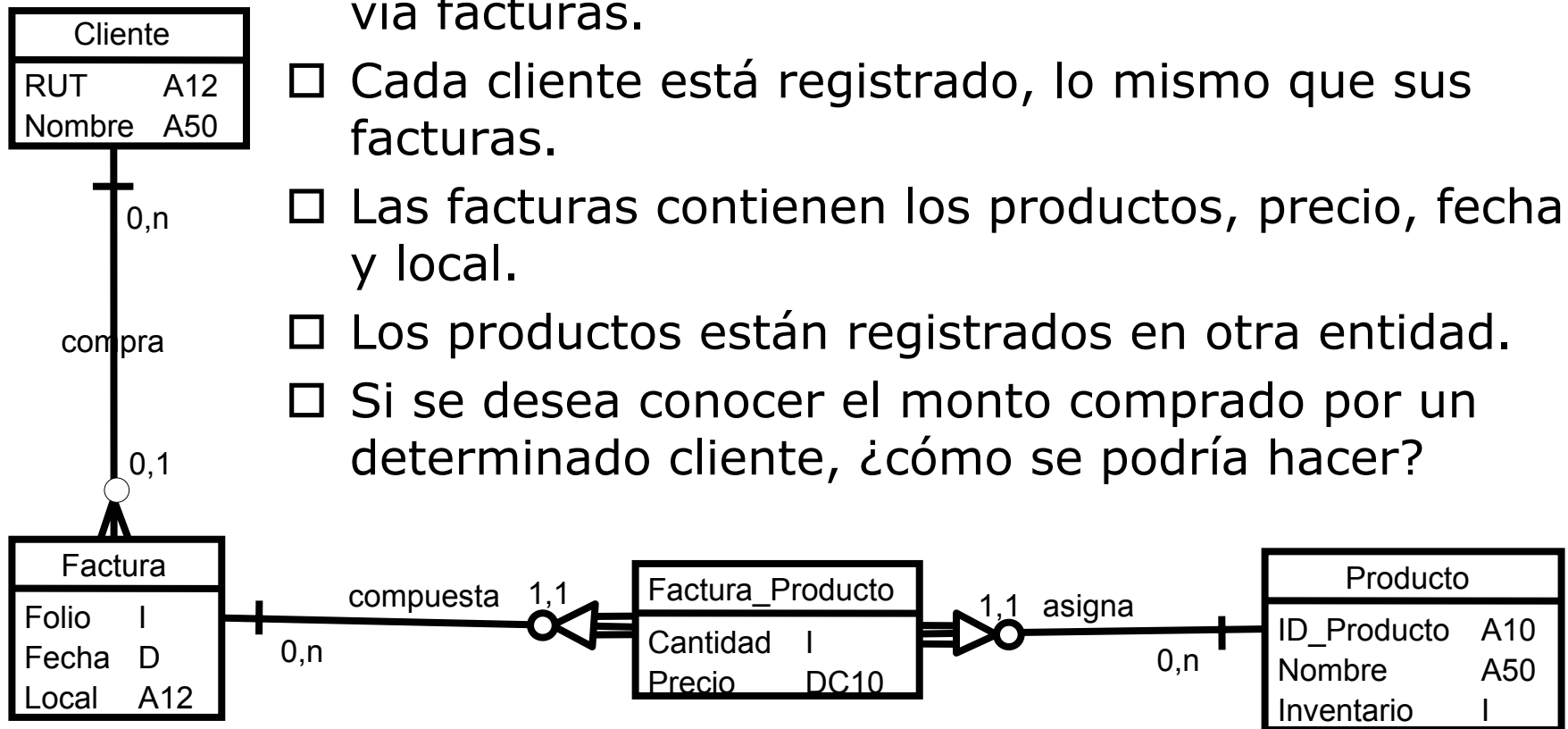
# Modelo Lógico de Datos

---

- Para diseñar el Modelo Lógico de Datos es necesario conocer en detalle
  - Requerimientos al Interior, es decir: las salidas, las entradas y las transformaciones con su contenido
  - Parámetros tales como: llaves de búsqueda, llaves de ordenamiento, tiempo de respuesta, etc.

# Ejemplo

- ❑ Una tienda de venta mayorista vende productos vía facturas.
- ❑ Cada cliente está registrado, lo mismo que sus facturas.
- ❑ Las facturas contienen los productos, precio, fecha y local.
- ❑ Los productos están registrados en otra entidad.
- ❑ Si se desea conocer el monto comprado por un determinado cliente, ¿cómo se podría hacer?



# Fases del Diseño Lógico de Datos

---

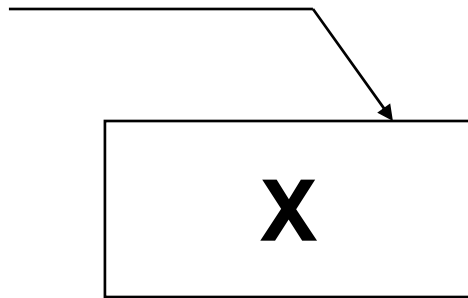
- ◆ FASE I: Enfoque cualitativo del Modelo Lógico de Datos (carga del modelo con los requerimientos).
- ◆ FASE II: Enfoque cuantitativo del Modelo Lógico de Datos (accesos lógicos)
- ◆ FASE III: Reestructurar el Modelo para buscar eficiencia lógica (nuevos accesos lógicos)

# FASE I: Enfoque cualitativo

---

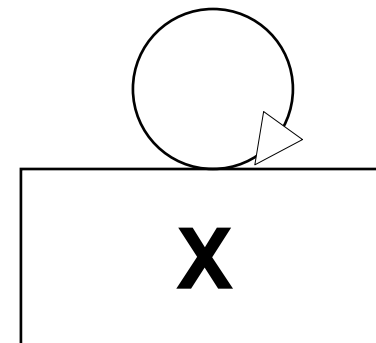
- ❑ Definir puntos de entrada al modelo (ACCESO LOGICO), ya sea buscando o recorriendo.
- ❑ Definir los tipos de acceso (Entrada, Lectura, Escritura, Búsqueda, etc.)
- ❑ Nota: En el modelo lógico de datos las instancias de entidades se denominan REGISTROS

## Llave(s) búsqueda



**Buscando dado el valor del  
identificador o llave de  
búsqueda**

## Llave(s) búsqueda



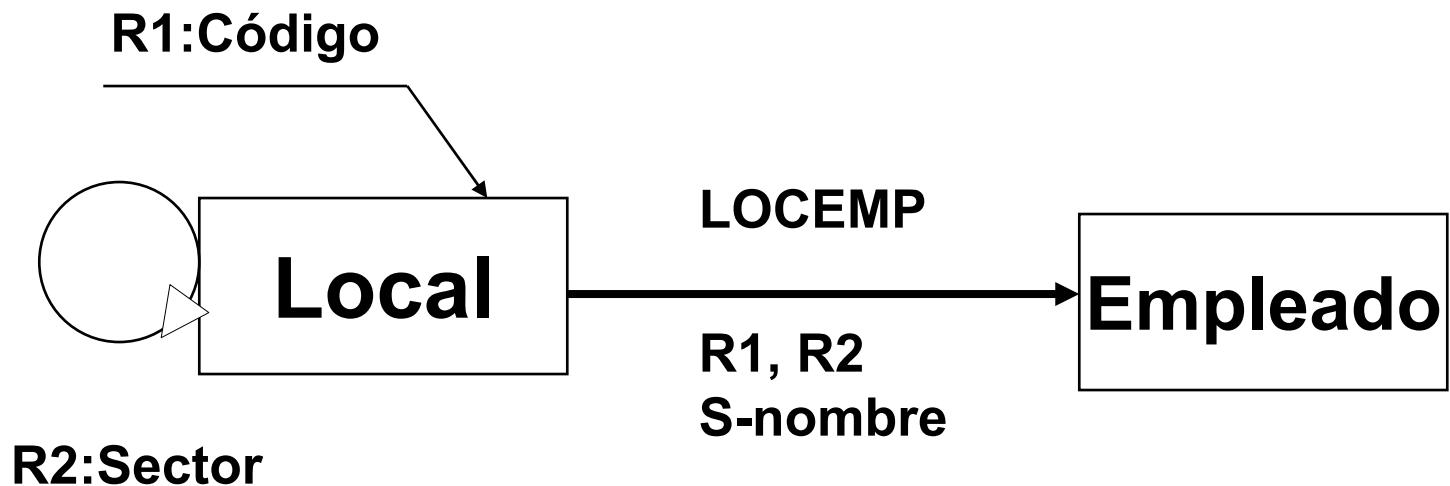
**Recorriendo según secuencia de  
valores del identificador o llave de  
ordenamiento**



# FASE I: Enfoque cualitativo

---

- Al usar una TRAYECTORIA de búsqueda se le da un nombre a la relación y un ordenamiento a los registros del lado N de la relación

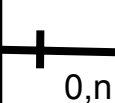
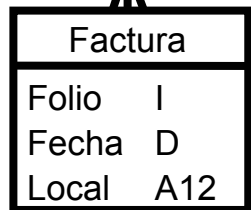
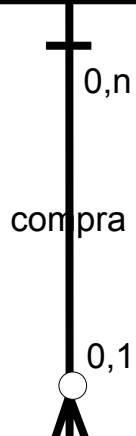
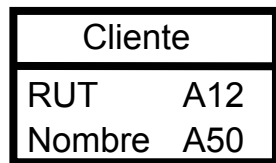


# FASE II: Enfoque cuantitativo

---

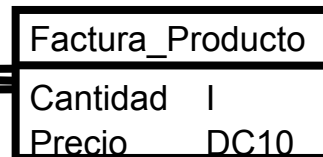
- ❑ El objetivo es determinar en forma cuantitativa la conducta del modelo lógico para resolver los requerimientos al interior (eficiencia).
- ❑ Tener claramente especificado en los requerimientos los parámetros de cada entrada, salida o transacción
  - Frecuencia
  - Llave: (Búsqueda, Ordenamiento)
- ❑ Conocer el número promedio de ocurrencia de los registros
  - Valor promedio
  - Tasa de crecimiento
- ❑ Definir una estrategia de acceso y recorrido por la estructura para cada requerimiento

# Ejemplo



compuesta

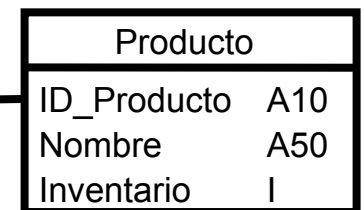
1,1



1,1

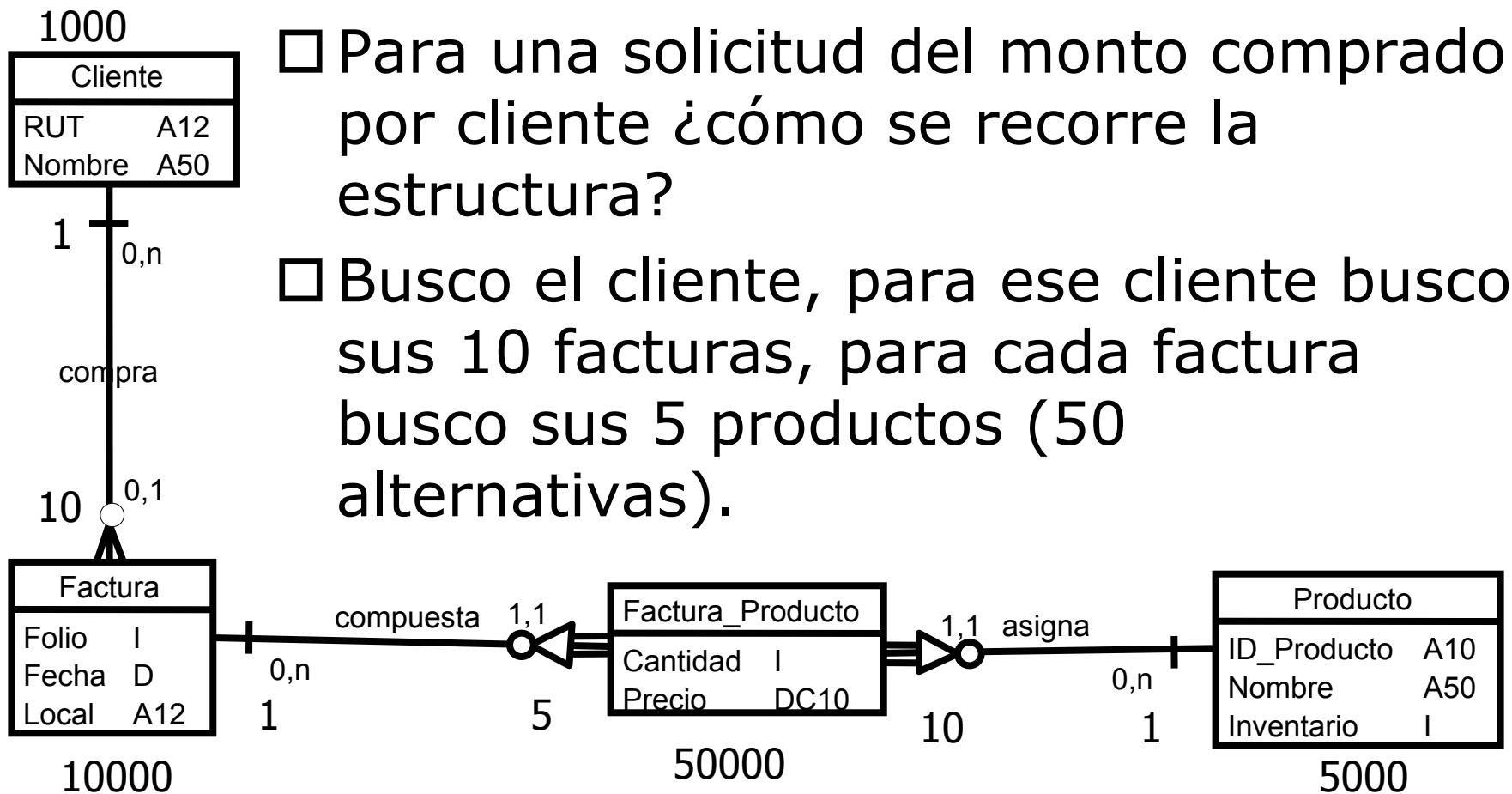
asigna

0,n



- Considere que hay: 1000 clientes, 5000 productos, 10 facturas por clientes y 5 productos por factura.
- En promedio ¿cuántos productos por factura?  $\Rightarrow 10$ .
- ¿Cuál es el esfuerzo de conocer el monto comprado por un cliente?

# Ejemplo ...



- ❑ Para una solicitud del monto comprado por cliente ¿cómo se recorre la estructura?
- ❑ Busco el cliente, para ese cliente busco sus 10 facturas, para cada factura busco sus 5 productos (50 alternativas).

# Ejemplo ...

---

100 consultas por período

				Número de Accesos Lógicos		Comentarios
Número	Trayectoria de Búsqueda Utilizada	Tipo de Acceso	Frecuencia Promedio	Por Transacción	Por Período	
1	Cliente	Entrada	1	1	100	
2	Cliente/Factura	Lectura	10	10	1000	
3	Factura/Factura-Producto	Lectura	5	50	5000	
Total					6100	

- ☐ Por cada consulta busco al cliente
- ☐ Busco sus 10 facturas
- ☐ Busco los 5 productos por factura

# Mapa de uso de Transacción

Modelo		Volumne	100		Diseñado por	
Transacción		Período	diaria		Fecha	

*Transacción: ¿cuál es el monto comprado del cliente k del producto i?*

				Número de Accesos Lógicos		
Número	Trayectoria de Búsqueda Utilizada	Tipo de Acceso	Frecuencia Promedio	Por Transacción	Por Período	Comentarios
1	Cliente	Entrada	1	1	100	
2	Cliente/Factura	Lectura	10	10	1000	
3	Factura/Factura-Producto	Lectura	5	50	5000	
Total					6100	

# Para Aplicar el Enfoque

---

- ❑ Tener claramente especificado en los requerimientos, los parámetros de cada salida, entrada y transacción.
  - ❑ Frecuencia.
  - ❑ Identificador: Búsqueda y Ordenamiento.
- ❑ Conocer el número promedio de ocurrencia de los registros, dueños, y de los miembros (relación de cada conjunto).
  - ❑ Valor promedio.
  - ❑ Desviación estándar.
  - ❑ Tasa de crecimiento.

# Para Aplicar el Enfoque ...

---

- Definir una estrategia de acceso y recorrido de la estructura, la cual puede especificarse de las siguientes formas, para cada requerimiento:
  - Gráfica. Para representar un requerimiento en forma gráfica; cuantitativamente, se marcará el acceso y recorrido, indicando la relación y el acceso de utilización.
  - Con castellano estructurado. El castellano estructurado es un conjunto de palabras del castellano; verbos permitidos para las acciones y bloques de programación estructurada.
    - Secuencia DO ... END
    - Decisión IF ... THEN ... ELSE
    - Iteracción 9DO ... UNTIL , DO ... WHILE



# FASE III: Reestructurar el Modelo

---

- Apunta a buscar eficiencia en el modelo lógico por: Frecuencia o Periodicidad, Tiempo de Respuesta, Nivel de Complejidad, Llaves de Ordenamiento y de Búsqueda, Volumen, Medio u otro.
- Se puede buscar minimizar tiempos de respuesta, tiempo de CPU, espacio en disco, tránsito sobre la estructura, etc.

# FASE III: Reestructurar el Modelo

---

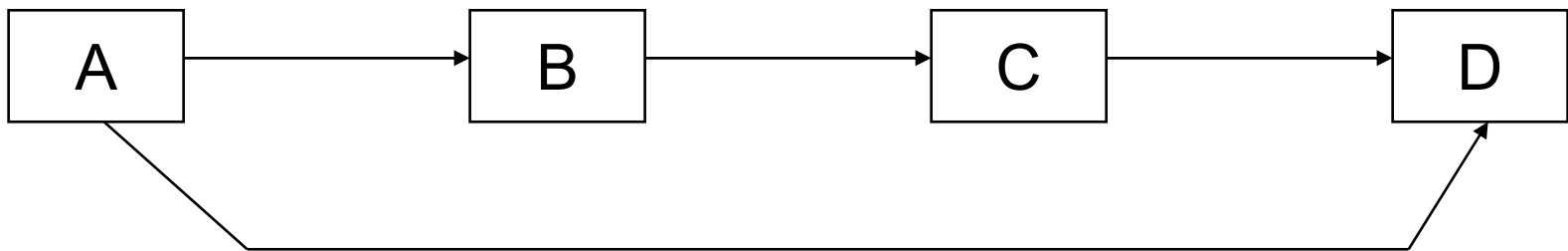
- Se mejorará la eficiencia mediante:
  - Agregar o eliminar relaciones
  - Sacar o incorporar datos a registros
  - Identificar relaciones redundantes y agregarlas o eliminarlas
  - Colocar datos resumen (que son datos redundantes)
  - Buscar formas alternativas de obtener los resultados pedidos (procesamiento fuera del enfoque)

# FASE III: Reestructurar el Modelo

---

## Ejemplos:

### a) Acortar trayectorias de búsqueda



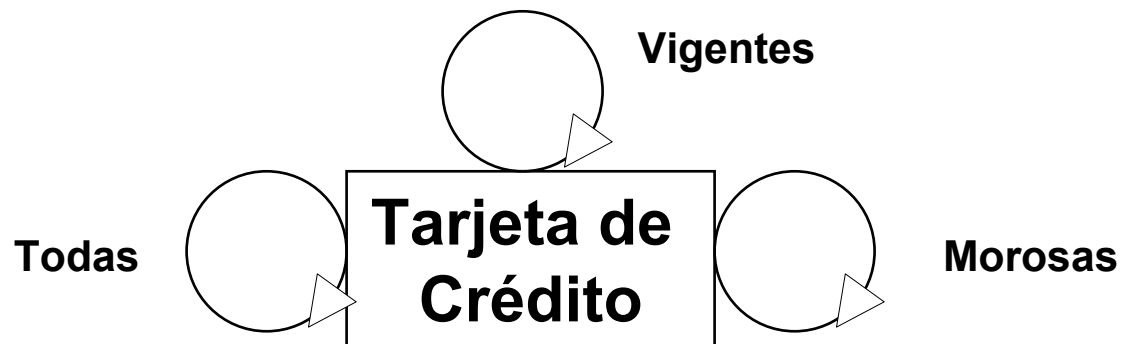
Se agrega relación AD

# FASE III: Reestructurar el Modelo

---

## Ejemplos:

- b) Incorporar relaciones que tienen una sola ocurrencia para acceder a una entidad por distintos conceptos



# FASE III: Reestructurar el Modelo

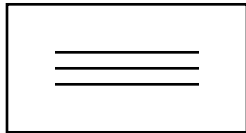
---

## Ejemplos:

### c) Agregar y mantener datos resumen:

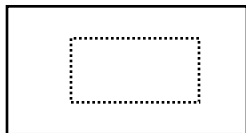
◆ Como dato único (registro resumen)

◆ Como dato repetitivo resumen:



Sumatoria de campos repetitivos en otros registros de la estructura

◆ Como dato repetido resumen:

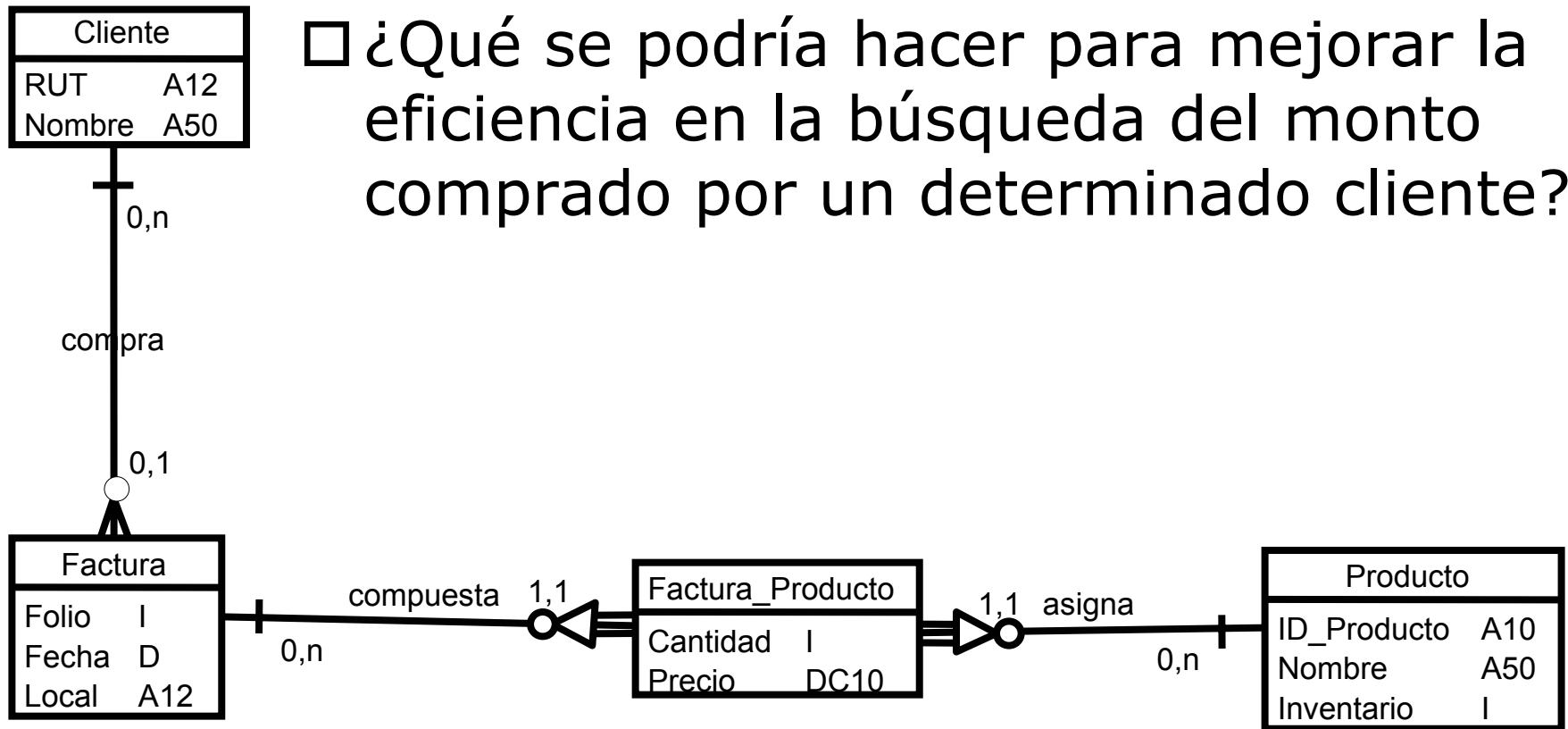


Sumatoria de campos repetidos en otros registros de la estructura

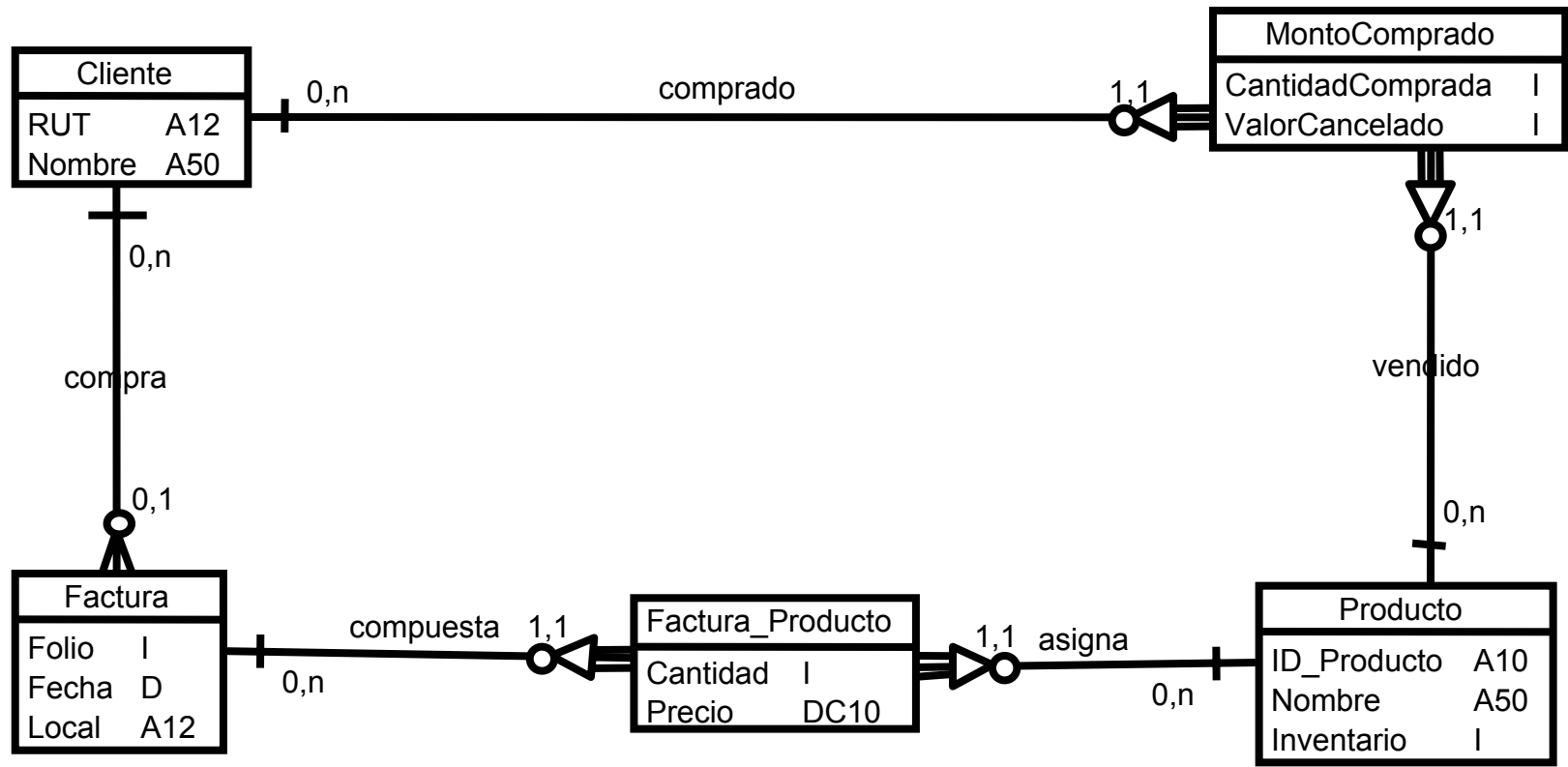
◆ Como relación resumen equivale a una relación redundante

# Ejemplo

- ¿Qué se podría hacer para mejorar la eficiencia en la búsqueda del monto comprado por un determinado cliente?



# Ejemplo ...



# Ejemplo ...

---

100 consultas por período

				Número de Accesos Lógicos		Comentarios
Número	Trayectoria de Búsqueda Utilizada	Tipo de Acceso	Frecuencia Promedio	Por Transacción	Por Período	
1	Cliente	Entrada	1	1	100	
2	Cliente/MontoComprado	Lectura	50	50	5000	
Total					5000	

- ☐ Por cada consulta busco al cliente
- ☐ Busco sus 50 productos



# Comentarios

---

- ❑ Este mecanismo de 3 fases permite determinar la carga por accesos lógicos y estimar el esfuerzo de trabajo asociado.
- ❑ Para mejorar la eficiencia hay que identificar aquellos de mayor frecuencia y esfuerzo.
- ❑ Se generan nuevas atributos, relaciones y/o entidades que es necesario mantener actualizadas.

# Modelamiento multidimensional (MMD)

---

- ❑ Las bases de datos relacionales están organizadas en torno de una lista de registros.
- ❑ Cada registro contiene información que es organizada en campos.
- ❑ Un ejemplo típico podría ser una lista de clientes con campos como la dirección y el teléfono.

Nombre	Id	Teléfono	Dirección
Banco de Chile	20454	565-8748	Alameda 1200
Endesa	24545	326-4569	Santa Rosa 55
CTC	17365	215-4563	Providencia 111

# MMD (2)

---

- ❑ A pesar de que la tabla tiene varias columnas de información, cada segmento se relaciona solamente a un nombre de cliente.
- ❑ Si se trata de crear una matriz bi-dimensional con el nombre del cliente, por un lado, y cualquier otro campo por el otro (como el teléfono), se podría ver que la correspondencia es uno a uno.
- ❑ Esta tabla no debe prestarse para ser utilizada en una base de datos multidimensional.

# MMD (3)

---

- **En una tabla relacional donde hay una correspondencia entre los campos mayor que uno a uno es posible apreciar situaciones como: datos de ventas de productos en cada región.**

Producto	Región	Ventas
Leche	Norte	100000
Leche	Centro	120000
Leche	Sur	40000
Crema	Norte	30000
Crema	Centro	25000
Crema	Sur	15000
Queso	Norte	75000
Queso	Centro	65000
Queso	Sur	70000

# MMD (4)

---

- Una manera mucho más clara de representar estos datos podría ser la siguiente matriz bi-dimensional:

	Norte	Centro	Sur
Leche	100.000	120.000	40.000
Crema	30.000	25.000	15.000
Queso	75.000	65.000	70.000

# MMD (5)

---

- ❑ Los datos de ventas corresponden a una matriz de dos dimensiones: Productos y Regiones.
- ❑ A pesar de que los datos pueden ser almacenados en una tabla relacional de tres campos, es mucho más natural hacerlo en una matriz de dos dimensiones.
- ❑ En lenguaje multidimensional podríamos decir que esta matriz representa las Ventas dimensionadas por Producto y Región.

# MMD (6)

---

- ❑ Si las consultas que se requiere responder son del tipo ¿Cuál fue la venta de Leche en el norte ? o ¿Cuál fue la venta de Queso en el sur ? y otras consultas que retornan sólo un valor, entonces no es necesario poner los datos en una base de datos multidimensional.
- ❑ Sin embargo, si se quiere responder a preguntas como ¿Cuales fueron las ventas totales de leche ? o ¿Cuales fueron las ventas totales en el norte ? entonces estamos hablando de consultas que involucran la recuperación de múltiples valores y su agregación.

# MMD (7)

---

- El tiempo de respuesta de una base de datos multidimensional, sin embargo, depende de la cantidad de valores que han sido pre-totalizados.
- Si queremos que el tiempo de respuesta sea consistentemente rápido independiente de la consulta realizada, entonces es necesario mantener consolidados todos los totales y subtotales lógicos.



# MMD (8)

---

- En el ejemplo se tiene que la matriz bi-bimensional queda de la siguiente forma:

	Norte	Centro	Sur	Total
Leche	100.000	120.000	40.000	260.000
Crema	30.000	25.000	15.000	70.000
Queso	75.000	65.000	70.000	210.000
Total	205.000	210.000	125.000	540.000

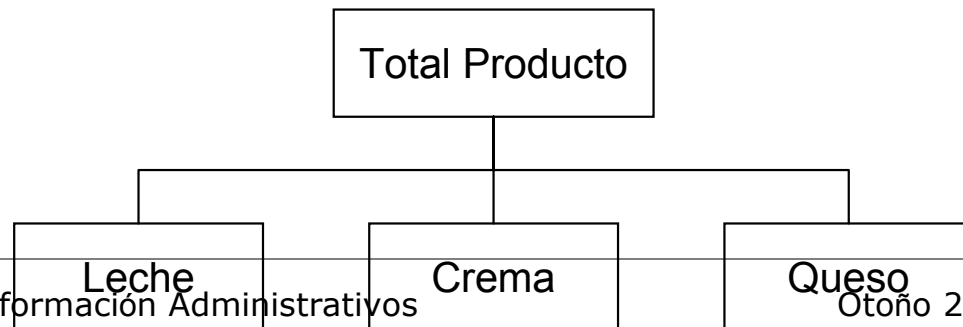
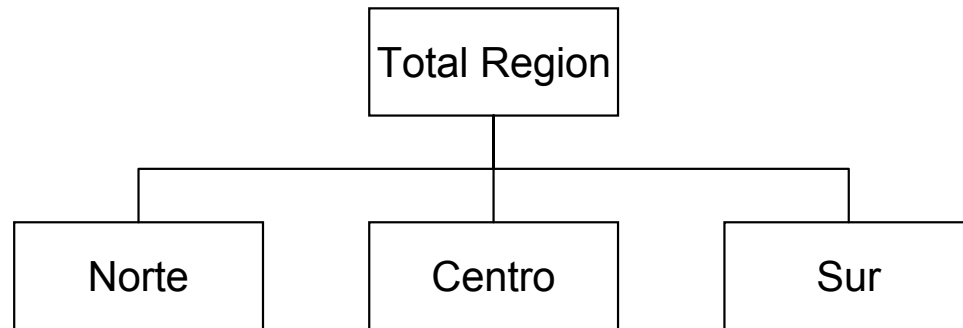
# MMD (9)

---

- Jerarquías. Los miembros de una dimensión pueden ser organizados en relaciones de padre-hijo, donde típicamente el miembro padre representa la consolidación de los miembros que tiene por hijos.
- Esta organización es llamada jerarquía, y las relaciones padre-hijo son llamadas relaciones jerárquicas.
- Una propiedad importante de las jerarquías es que pueden *colapsar y expandir*, es decir, navegar de un nivel de datos detallado a uno más sumariado o resumido, y viceversa.

# MMD (10)

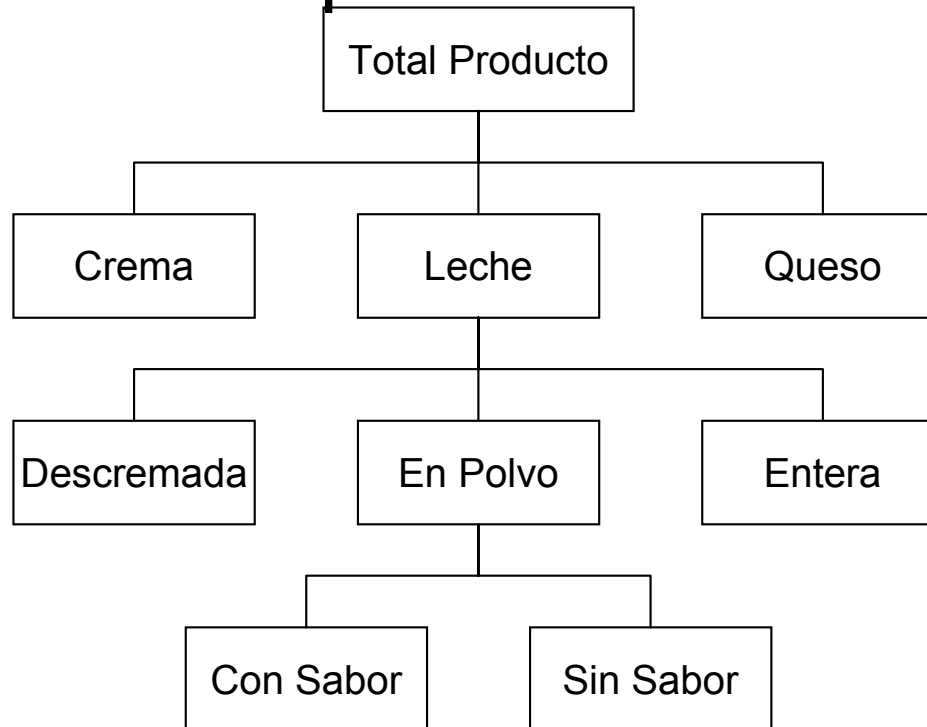
- 
- Estas jerarquías pueden ser representadas gráficamente como:



# MMD (11)

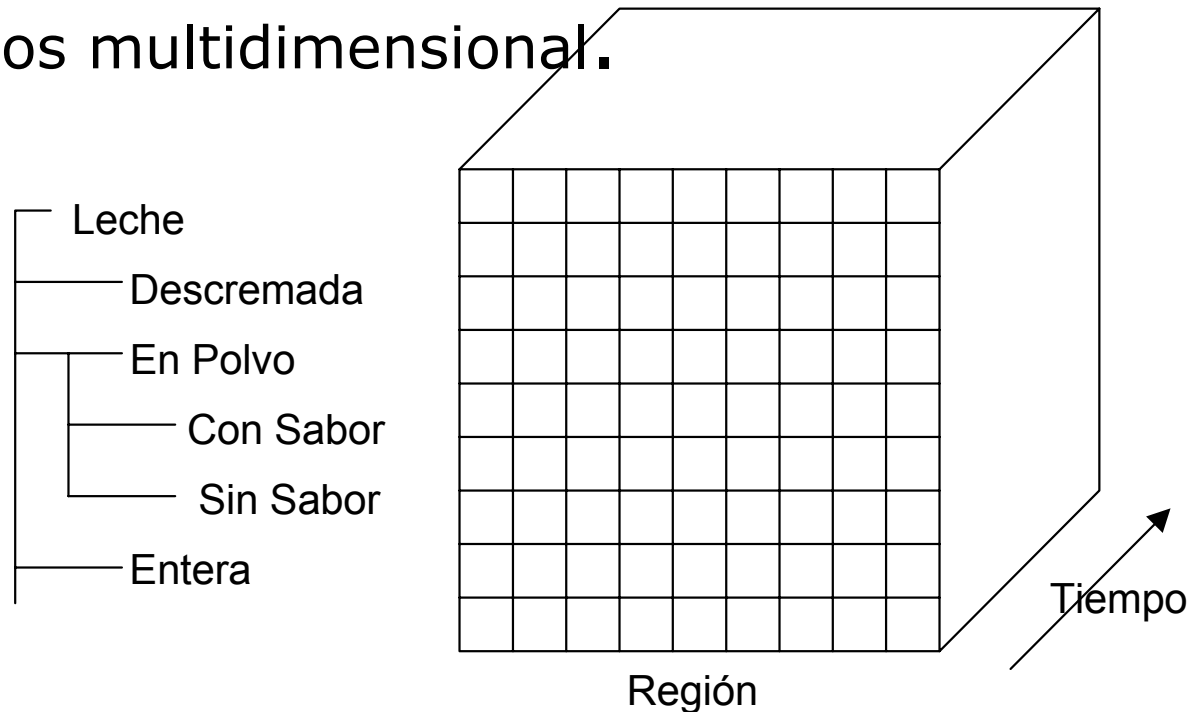
---

- Las jerarquías también pueden contener múltiples niveles

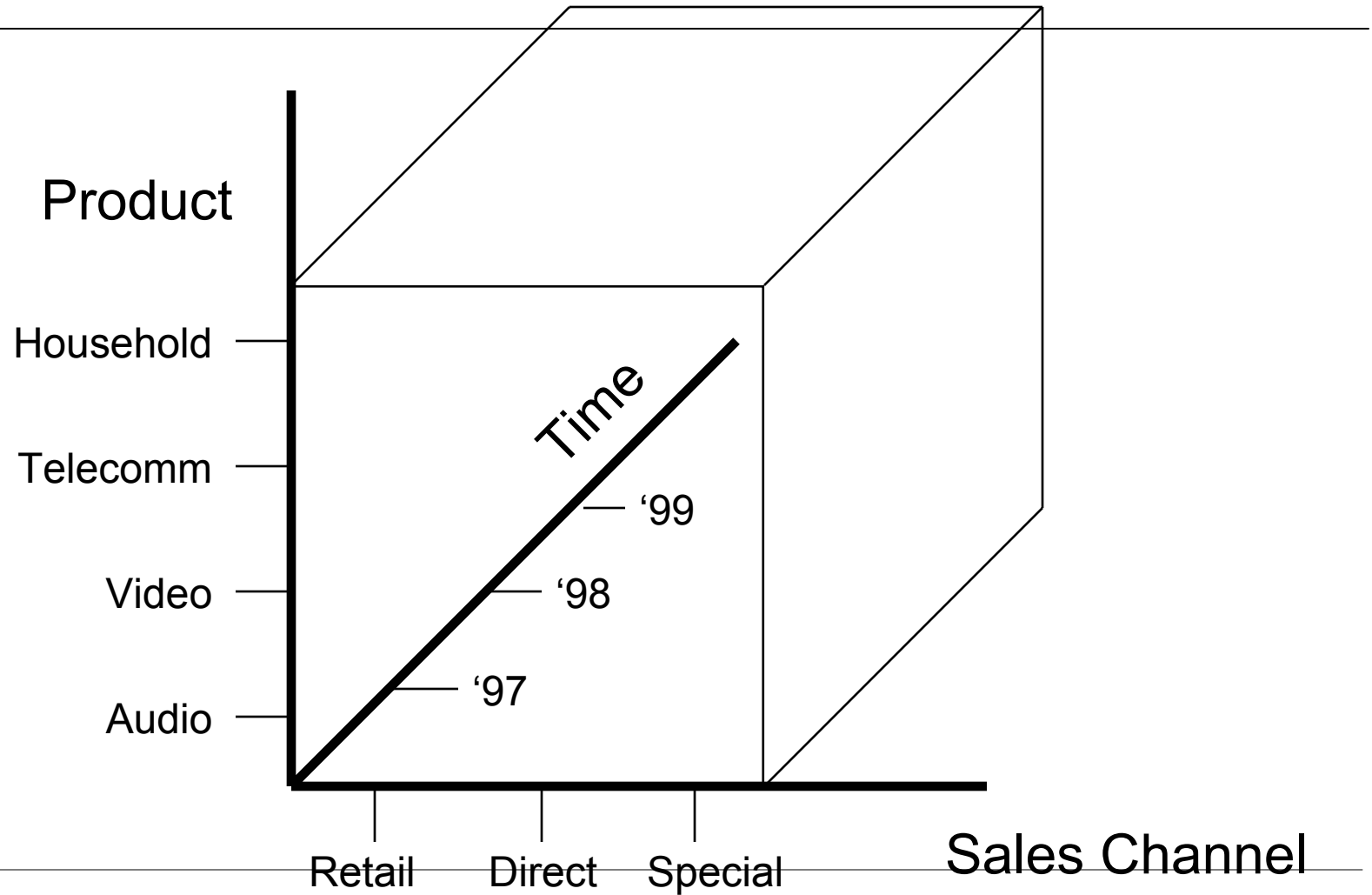


# MMD (12)

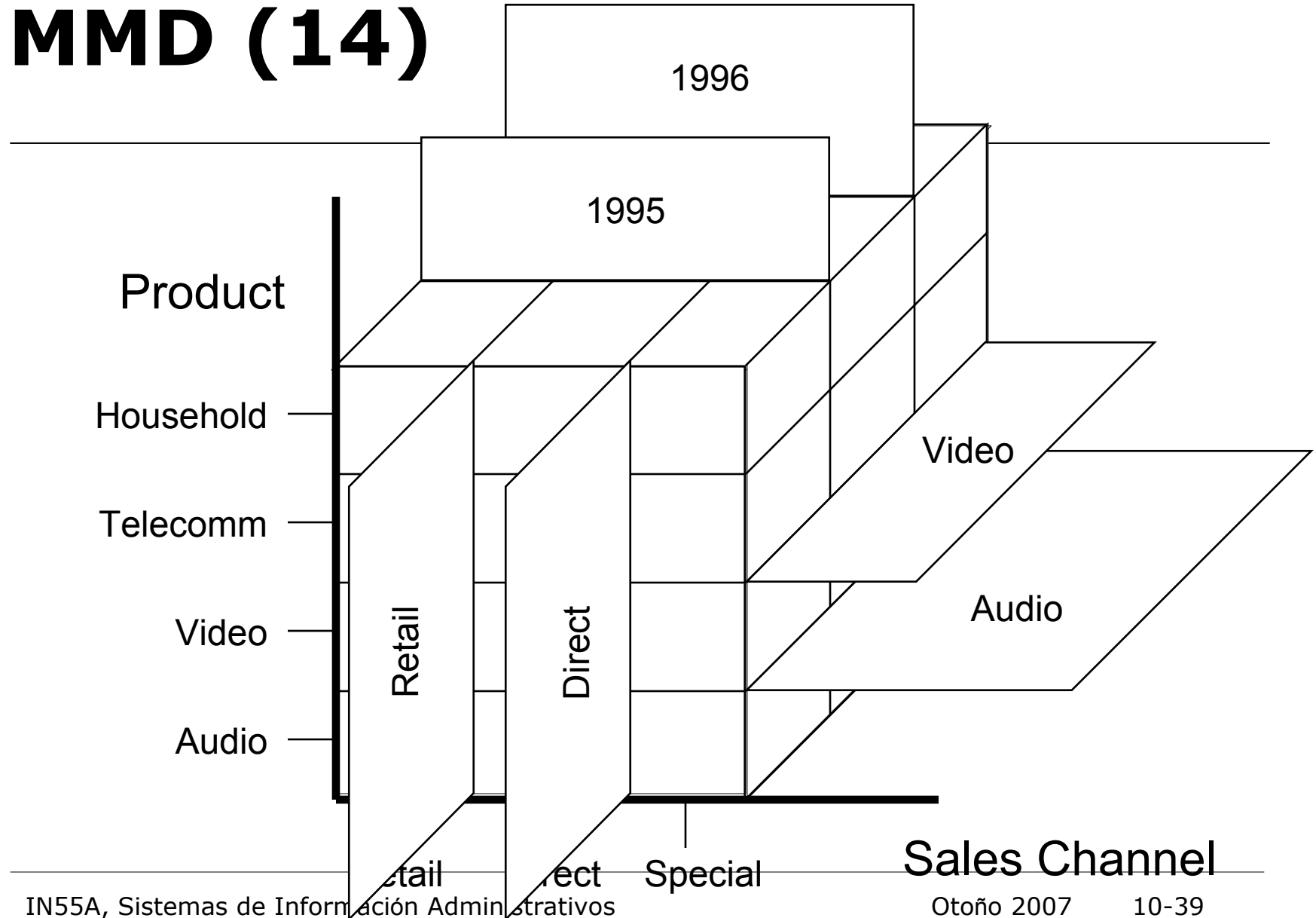
- Ahora si se introduce una tercera dimensión, como puede ser el tiempo, tenemos la siguiente estructura para nuestra base de datos multidimensional.



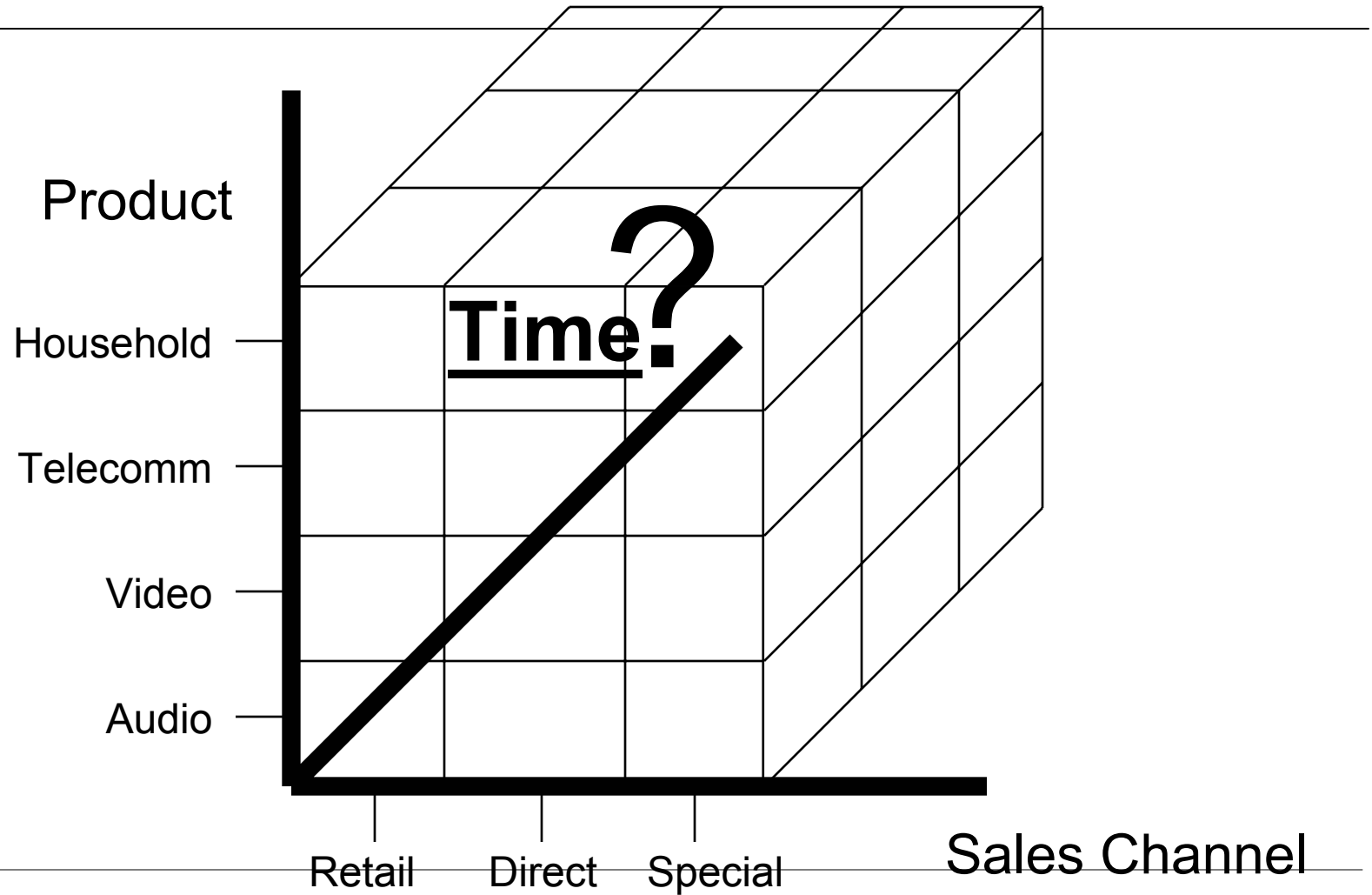
# MMD (13)



# MMD (14)

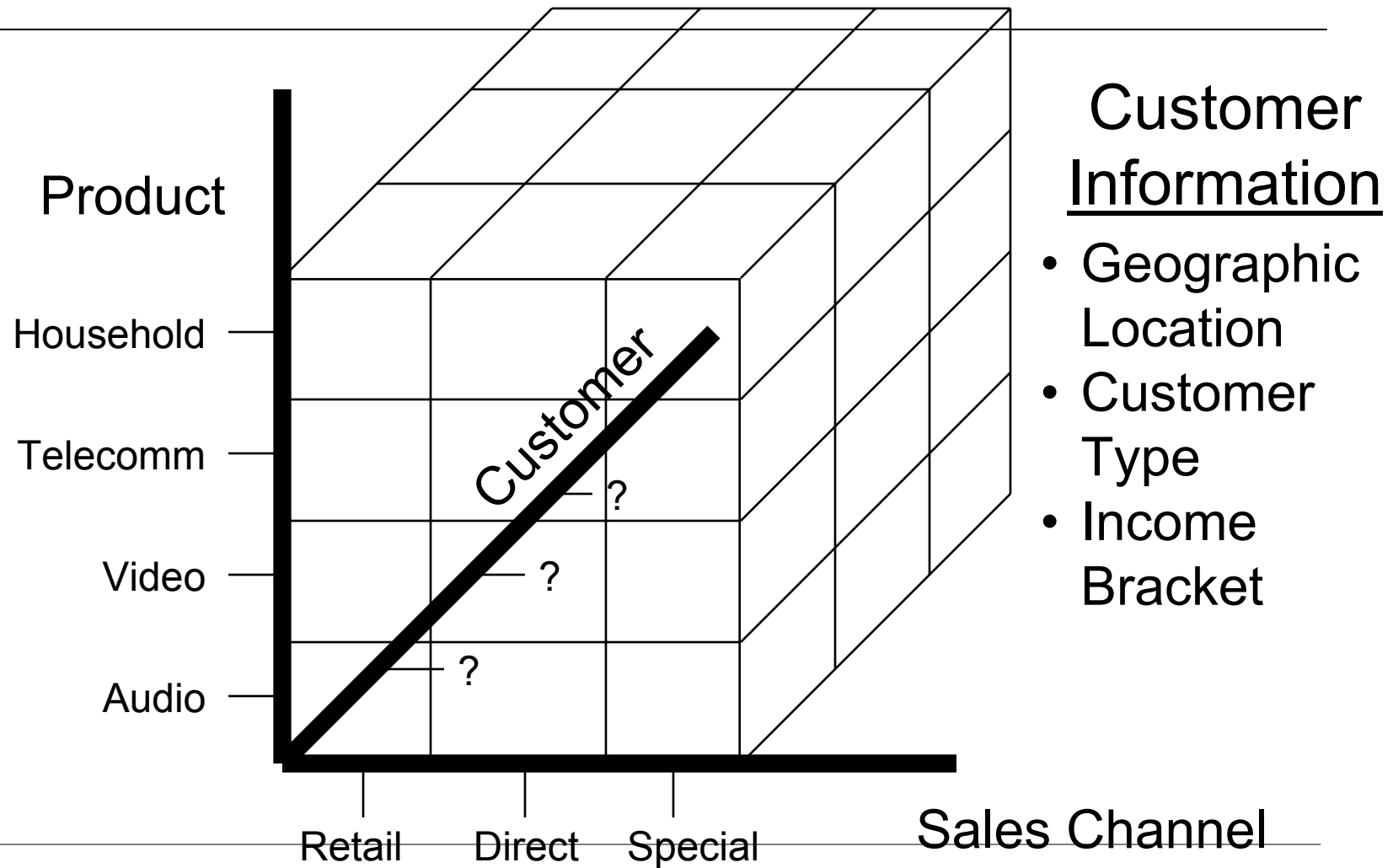


# MMD (15)

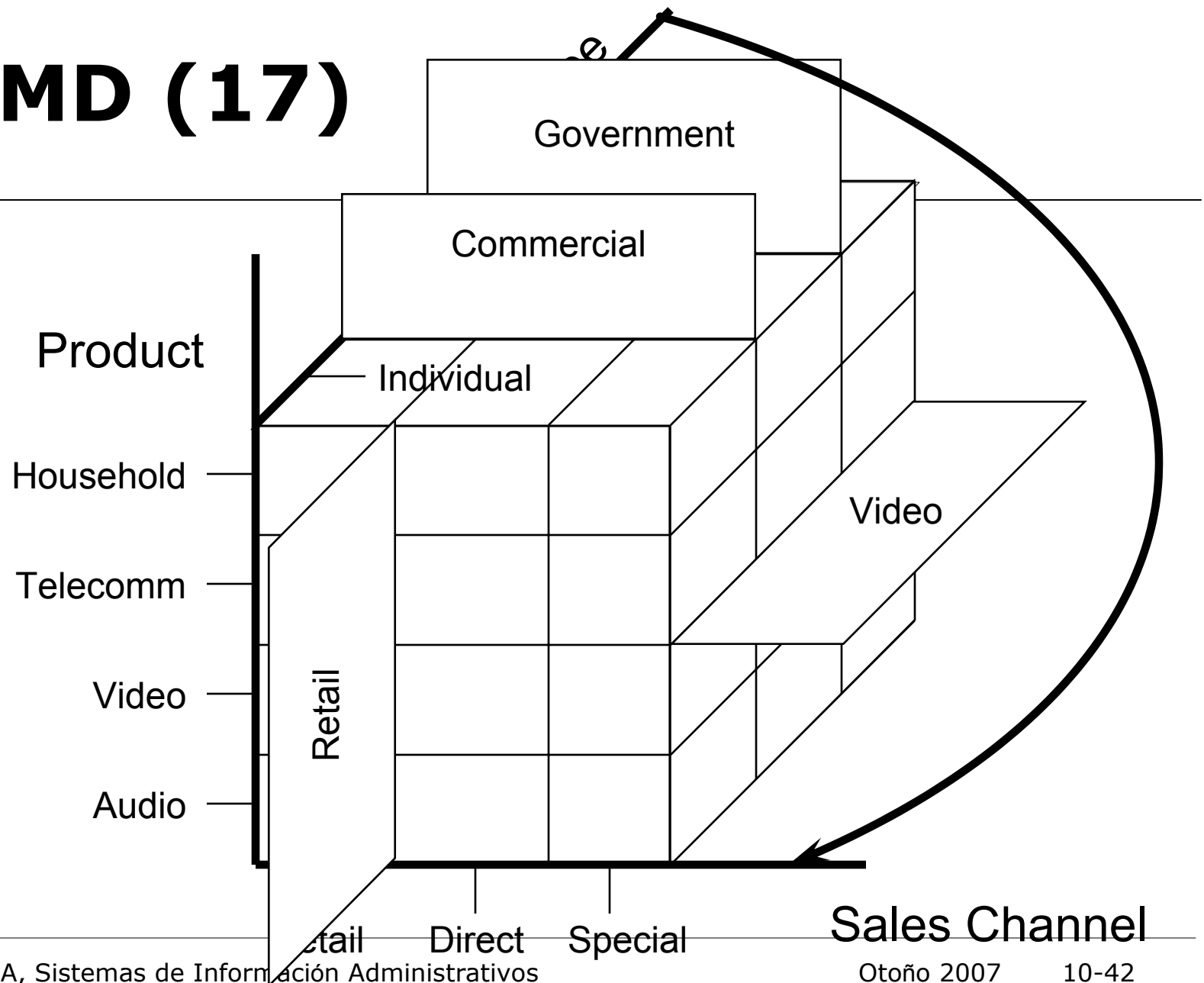




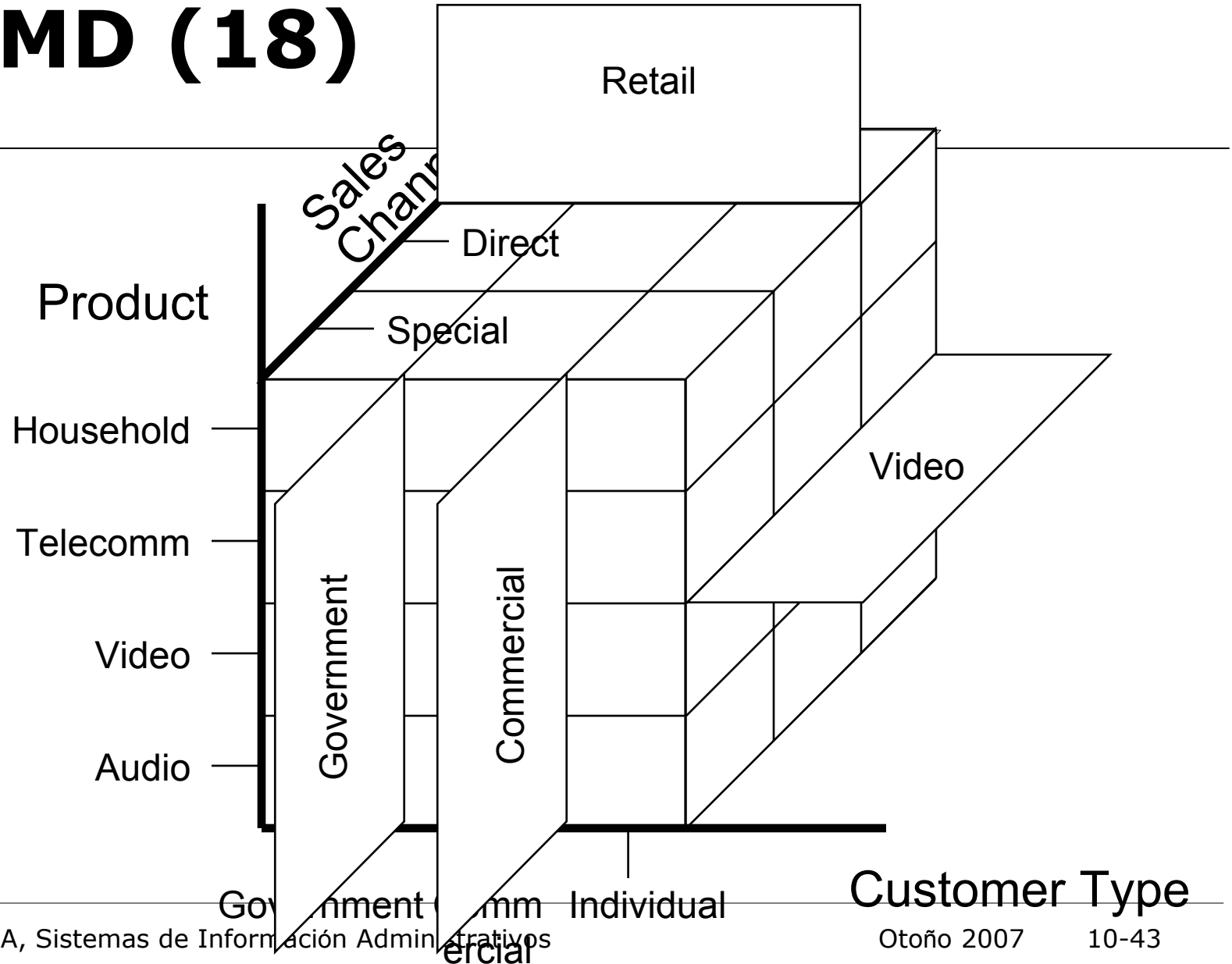
# MMD (16)



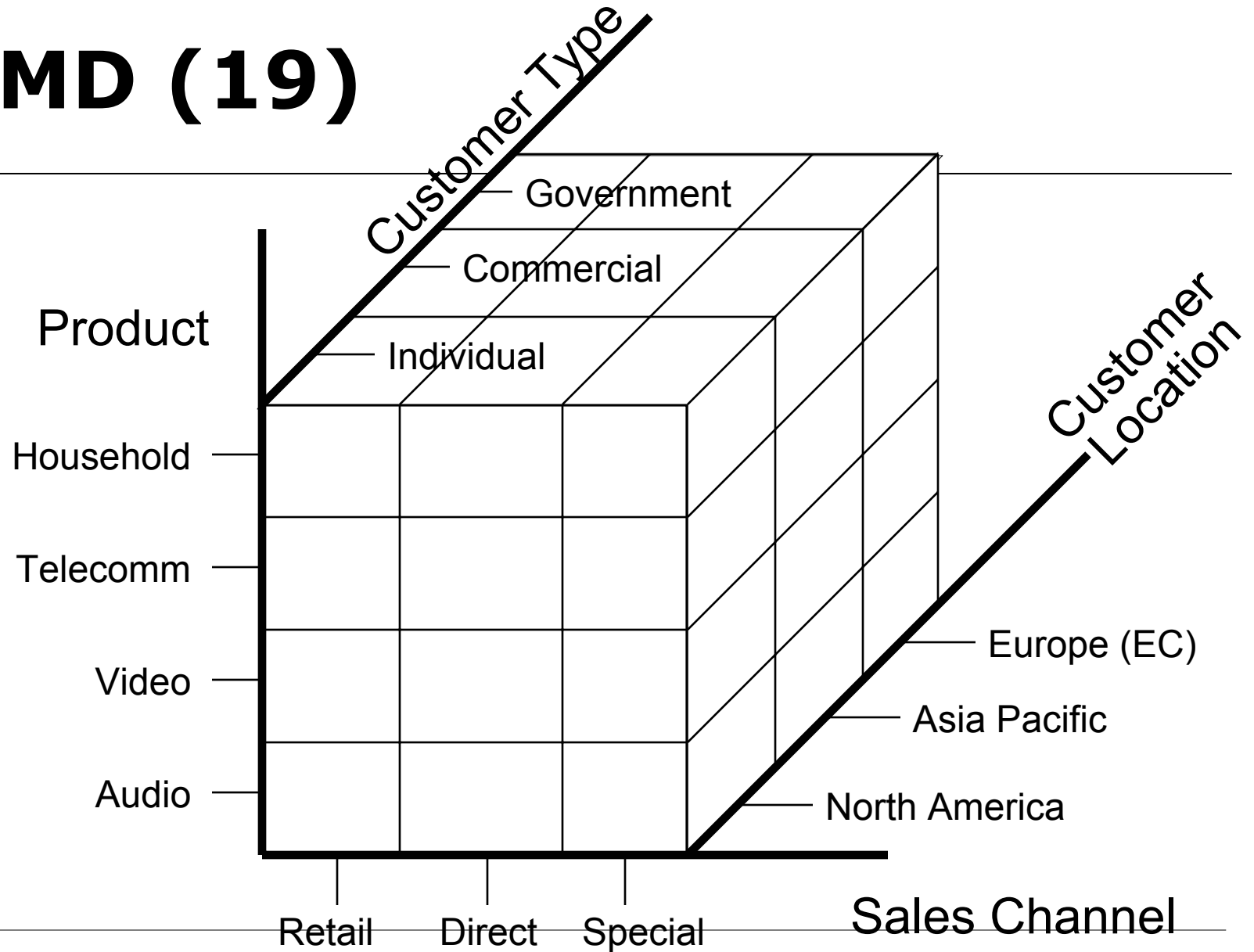
# MMD (17)



# MMD (18)



# MMD (19)



# MMD (20)

	Retail	Direct	Special
<b>Household</b>	<b>\$300</b>	<b>\$200</b>	<b>\$100</b>
Government	\$100	\$50	\$10
Commercial	\$100	\$75	\$70
Individual	\$100	\$75	\$30
<b>Telecomm</b>	<b>\$6000</b>	<b>\$3000</b>	<b>\$400</b>
Government	\$2000	\$500	\$100
Commercial	\$1000	\$1500	\$100
Individual	\$3000	\$1000	\$200
<b>Video</b>	<b>\$4444</b>	<b>\$2222</b>	<b>\$777</b>
Government	\$1030	\$150	\$222
Commercial	\$1311	\$1175	\$111
Individual	\$2103	\$897	\$444
<b>Audio</b>	<b>\$50</b>	<b>\$75</b>	<b>\$25</b>
Government	\$10	\$25	\$20
Commercial	\$10	\$25	\$0
Individual	\$30	\$25	\$5

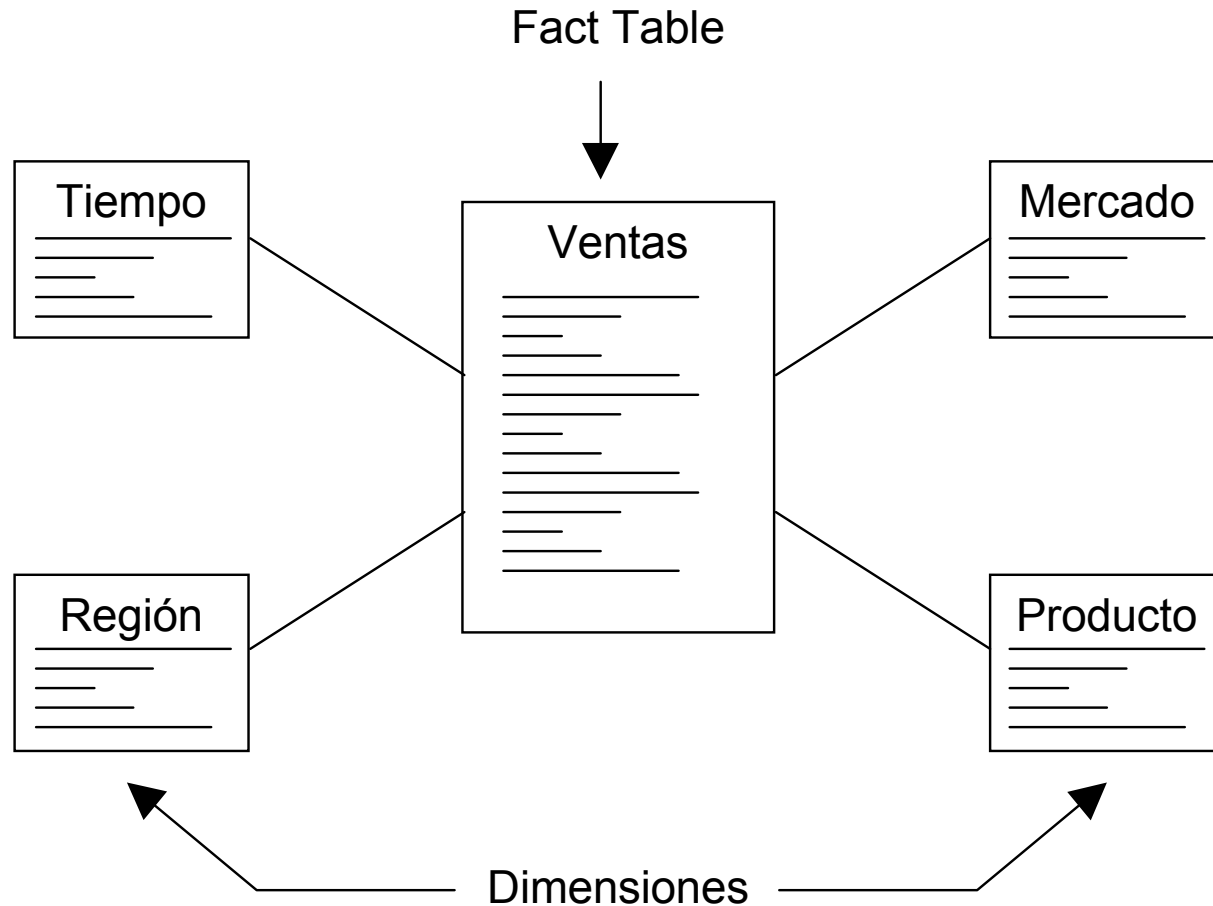
# Modelamiento estrella

---

- ❑ El cubo requiere de una base de datos multidimensional para ser implementado.
- ❑ Un esquema alternativo permite usar los conceptos de multidimensionalidad y aplicarlos en una base de datos relacional.
- ❑ Este esquema es el que se denominará *Modelo Estrella*, debido a que su representación gráfica se asemeja a la forma de una estrella.
- ❑ El esquema general de cómo se estructura un modelo estrella, considera una tabla central llamada *fact table* se ubican las tablas de *dimensiones*.

# Modelamiento estrella: Ejemplo

---



# Modelamiento estrella (3)

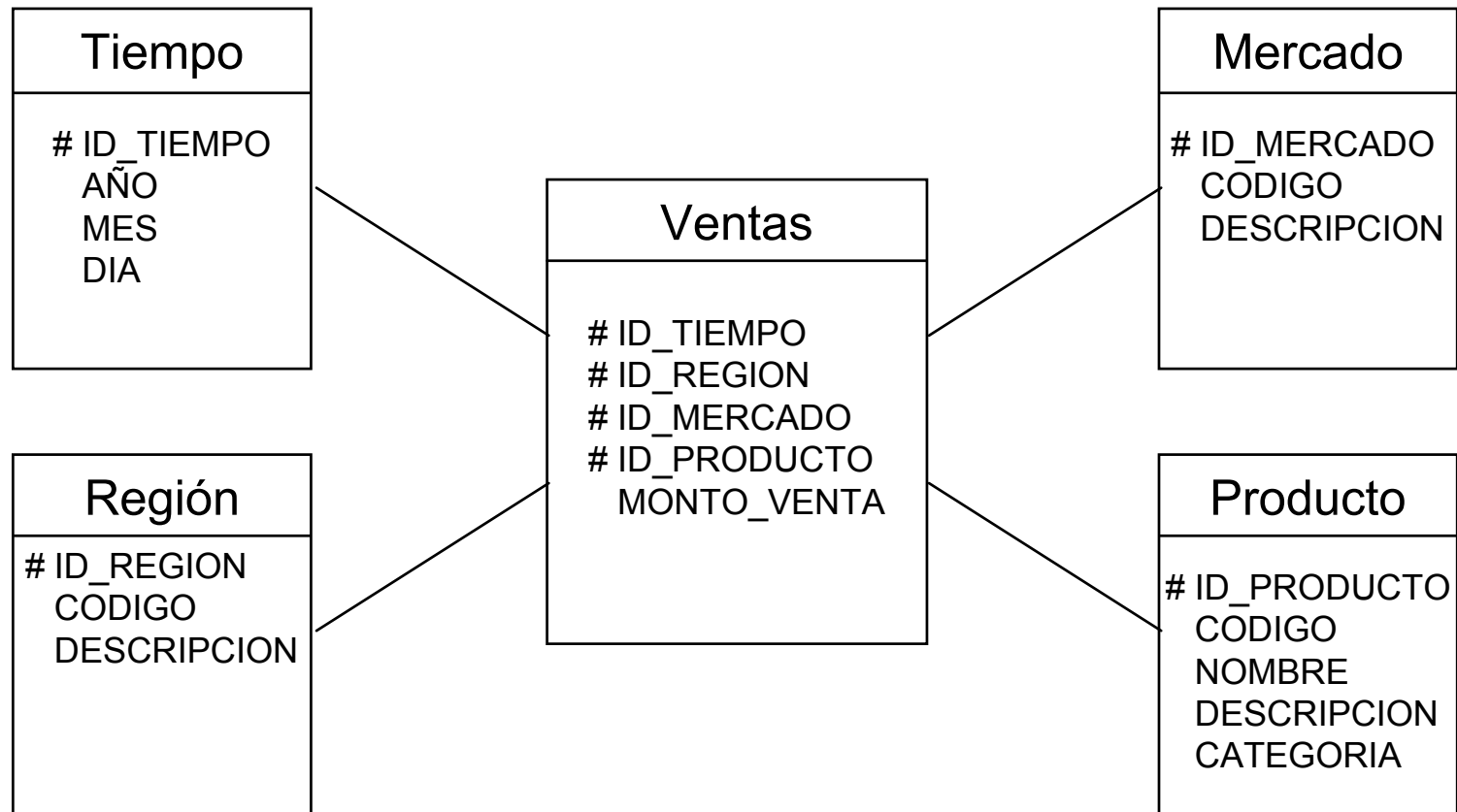
---

- ❑ Star Query (Consulta Estrella).
- ❑ Considera un “super join” entre todas las tablas del modelo estrella.
- ❑ La mayoría de los motores relacionales consideran optimizaciones para apoyar el rendimiento de la Star Query.
- ❑ Su implementación es a base del Select tradicional sobre un modelo estrella.



# Modelamiento estrella (4)

---



# Modelamiento estrella (5)

---

- ☐ Crea una base con tiempo de respuesta relativamente rápidos.
- ☐ Proporciona un diseño fácil de modificar.
- ☐ Simula como “ven” el problema los usuarios finales.
- ☐ Simplifica la navegación del Metadatos (lo veremos más adelante)
- ☐ Facilita el uso de herramientas relacionales.

# Modelamiento estrella (6)

---

```
SELECT SUM (monto_venta)
FROM    ventas, tiempo, region, mercado, producto
WHERE   tiempo.año          = 1999
AND     tiempo.mes          = 'SEPTIEMBRE'
AND     mercado.codigo      = 'ADULTOS'
AND     region.codigo       = 'NORTE'
AND     producto.categoria  = 'LIBROS'
AND

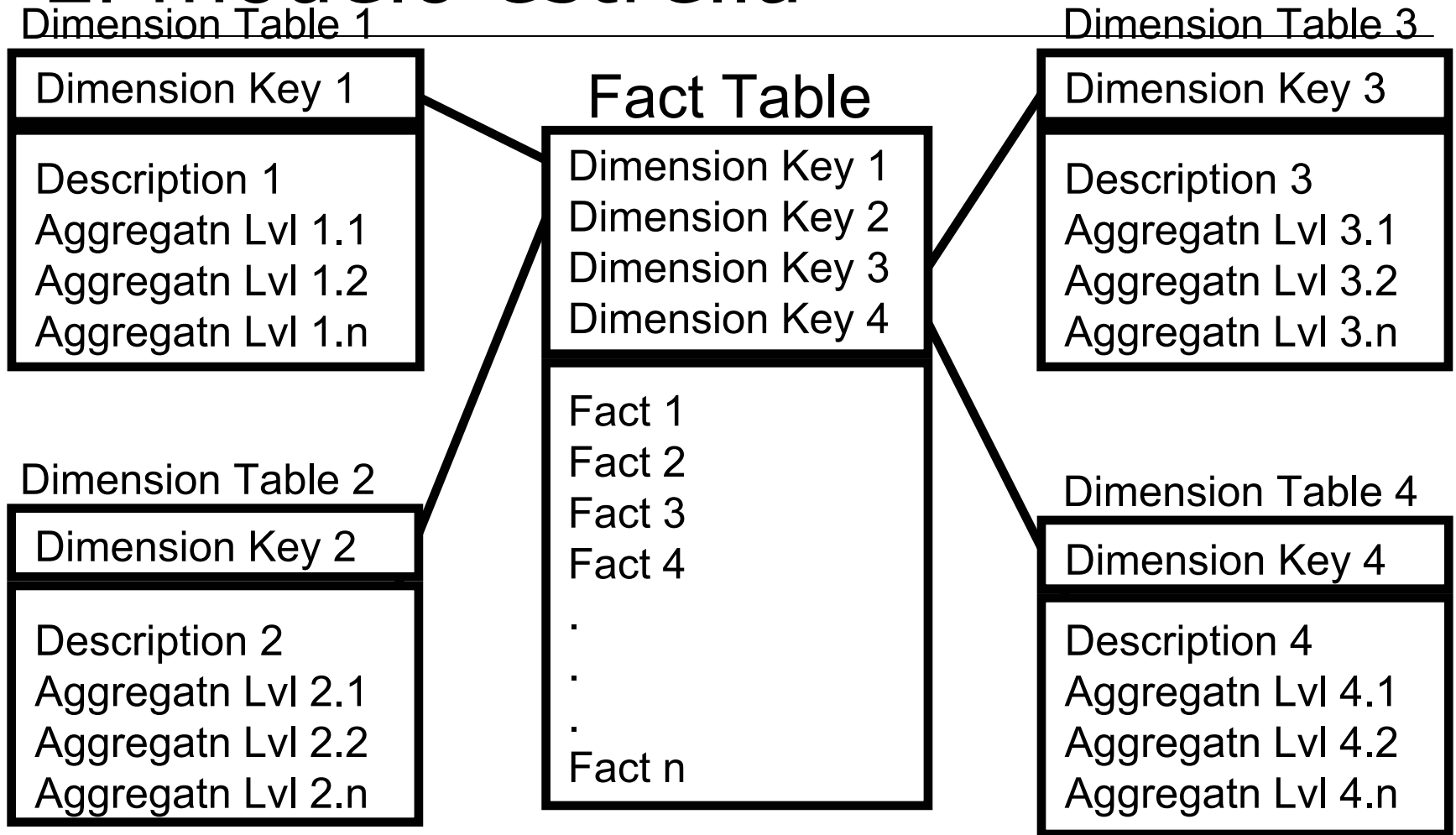
        /* Joins */
        tiempo.id_tiempo    = ventas.id_tiempo
AND     region.id_region    = ventas.id_region
AND     mercado.id_mercado  = ventas.id_mercado
AND     producto.id_producto = ventas.id_producto
```

# Requerimientos del negocio

---

- Preguntas del negocio:
  - “¿Qué tipo de clientes compra alguno de nuestros videos y dónde?
  - ¿Hay patrones geográficos de compra de un producto en particular?
  - ¿Existe alguna relación entre los datos demográficos y el comportamiento de compra?
- Medida:  
Ventas de video por Cliente, por Tipo de Producto, por Ubicación del almacén (en algún período de tiempo)

# El modelo estrella



# Tabla dimensional

Dimension Table 1

---

Dimension Key 1
Description 1
Aggregatn Lvl 1.1
Aggregatn Lvl 1.2
Aggregatn Lvl 1.n

- ☐ Describe los datos que han sido organizados en la tabla Fact.
- ☐ El número de niveles de agregación es configurable.
- ☐ Describen las entidades de negocio en una empresa, la cual usualmente representa información por jerarquías, tales como departamentos, lugares y productos.
- ☐ Usualmente cambian poco y no son muy largas, pero afectan el rendimiento de las consultas debido a los joins con la tabla fact.

# Fact Table

---

## Fact Table

Dimension Key 1  
Dimension Key 2  
Dimension Key 3  
Dimension Key 4

Fact 1  
Fact 2  
Fact 3  
Fact 4  
.  
.  
.  
Fact n

- ☐ Cuantifica los datos que han sido descritos por las tablas dimensionales.
- ☐ La llave unifica los valores de las llaves de las tablas dimensionales.
  - SIEMPRE contiene la fecha o una tabla dimensional sobre el tiempo
- ☐ Su tamaño, en la práctica representa el 70% del DW.
- ☐ En algunos motores, se puede particionar para mejorar el rendimiento.

# Fact table (2)

---

- ❑ La mayoría de los datos de un DW están almacenados en unas pocas fact table, que contienen muchos registros.
- ❑ Su clave es la concatenación de claves foráneas.
- ❑ Incluyen medidas tales como ventas, unidades e inventario:
  - Medida simple FACT.SALES.
  - Medida calculada a partir de una expresión como FACT.REVENUES - FACT.EXPENSES.



# Fact table: Granularidad

---

- ☐ Representa específicamente qué es lo que se va a almacenar en la fact table.
- ☐ Cada fila de la tabla fact debe almacenarse al mismo nivel de detalle.
- ☐ Información relevante a diferente nivel de granularidad, debe almacenarse en tablas fact distintas.
- ☐ Las claves foráneas que relacionan la fact con las dimensionales, no pueden ser nulas!!

# Fact table: Medidas

---

- ❑ Los valores resumen de la tabla fact, son generalmente numéricos y representan una medida.
- ❑ Las medidas, típicamente son aditivas y permiten sumas que involucren todas las tablas dimensionales.
- ❑ Por ejemplo, la cantidad en una tabla fact de ventas.
- ❑ Una suma de la cantidad por cliente, producto, fecha o cualquier combinación de las dimensiones, entrega un valor.

# Fact table: Medidas (2)

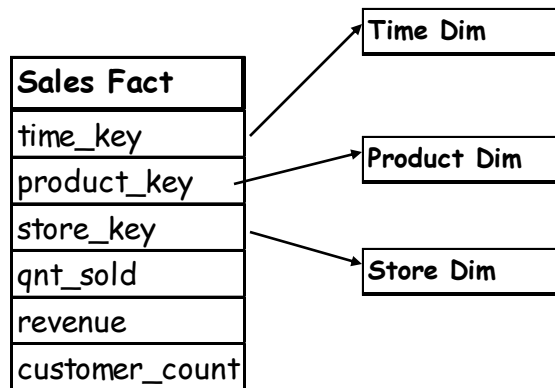
---

- ❑ Semi aditivas. Sólo tienen sentido si se involucran algunas y no todas las dimensiones.
- ❑ Los balances son un tipo clásico de estas medidas. Por ejemplo, cantidad en stock en un sistema de ventas, en un rango de tiempo.
- ❑ En este caso, única dimensión que importa es el tiempo.
- ❑ No aditivas. No se pueden aplicar funciones de agregación. Por ejemplo los márgenes en porcentaje.

# Fact Table: Medidas y ejemplos

---

□ Supongamos la siguiente situación.



Cantidad vendida (qnt\_sold) y revenue (renta) son aditivas

Customer\_count (cantidad de clientes) es semi aditiva, si dos productos, tienen 40 y 60 clientes, el total de clientes puede ser cualquiera entre 40 y 60 (los clientes se pueden repetir)  
Así esta medida no es aditiva en la dimensión del producto.

# Dimensiones

---

- ❑ Su clave es única, preferentemente generada por el sistema (uso de sequence)
- ❑ Es pequeña comparada con la fact.
- ❑ No hace falta que los atributos estén relacionados.
- ❑ No se aplican reglas de normalización.
- ❑ Los atributos cualifican consultas y sirven para los cálculos.

# Vistas de consulta

---

- ❑ Se trata de un objeto que se almacena en la base de datos que soporta al DW.
- ❑ Su función es contener la lógica que permite interactuar con las tablas dimensionales y la fact para obtener un calculo.
- ❑ Se construyen en el DW para realizar operaciones de resumen o de cálculo por agregación.
- ❑ Con la herramienta adecuada, el End User puede cambiar la parametrización y extraer información en forma simple (Oracle Discover)

# Vistas de consulta (2)

---

- ❑ El problema es que cada vez que se invoca a la vista, se ejecuta la consulta contenida.
- ❑ En un DW los cambios no son en tiempo real.
- ❑ Se crea el concepto de “vista materializada” (MATERIALIZED VIEW)
- ❑ La idea es mantener los datos que trajo la vista cuando fue consultada.

# Vistas de consulta (2)

---

```
CREATE MATERIALIZED VIEW store_sales_mv
  PCTFREE 0 TABLESPACE mviews
  STORAGE (initial 16k next 16k pctincrease 0)
  BUILD DEFERRED
  REFRESH COMPLETE ON DEMAND
  ENABLE QUERY REWRITE
AS
SELECT
  s.store_name,
  SUM(dollar_sales) AS sum_dollar_sales
  FROM store s, fact f
  WHERE f.store_key = s.store_key
  GROUP BY s.store_name;
```



# Mejorando el rendimiento: Uso de estructuras de acceso rápido

---

- Bitmap index. Son muy usados en VLDB y proveen:
  - Tiempos de respuesta aceptables para un amplio rango de consultas.
  - Ocupan mucho menos espacio que el B\*Tree tradicional
  - Poseen un buen rendimiento en equipos poco dimensionados.
  - Son muy eficientes en mantención, ante el uso de comandos DML.

# Mejorando el rendimiento: Bitmap index (2)

---

- ❑ En un índice tradicional, se almacena el dato indexado y el rowid (dirección de memoria) de la fila que lo contiene.
- ❑ En un bitmap index, por cada valor indexado se usa un “mapa de bits” en vez de una lista de rowid.
- ❑ Son muy eficientes cuando hay consultas donde se usa mucho la cláusula WHERE.
- ❑ No se les recomienda en sistemas que usan mucho instrucciones OLTP.
- ❑ Su mejor rendimiento se obtiene cuando la columna es de baja cardinalidad.

# Mejorando el rendimiento: Bitmap index (3), ejemplo

customer	marital_status	region	gender	income_level
101	single	east	male	bracket_1
102	married	central	female	bracket_4
103	married	west	female	bracket_2
104	divorced	west	male	bracket_4
105	single	central	female	bracket_2
106	married	central	female	bracket_3

REGION='east'	REGION='central'	REGION='WEST'
1	0	0
0	1	0
0	0	1
0	0	1
0	1	0
0	1	0

```
SELECT COUNT(*) FROM customer WHERE MARITAL_STATUS = 'married'
AND REGION IN ('central','west');
```

status = 'married'	region = 'central'	region = 'west'
0	0	0
1	1	0
1	0	1
0	0	1
0	1	0
1	1	0

# Mejorando el rendimiento: Partition table

---

- ❑ La idea es que una tabla pueda ser almacenada en estructuras de datos físicamente separadas.

```
CREATE TABLE sales
(s_productid NUMBER,
 s_saledate DATE,
 s_custid NUMBER,
 s_totalprice NUMBER)
PARTITION BY RANGE(s_saledate)
(PARTITION sal94q1 VALUES LESS THAN TO_DATE (01-APR-1994, DD-MON-YYYY),
 PARTITION sal94q2 VALUES LESS THAN TO_DATE (01-JUL-1994, DD-MON-YYYY),
 PARTITION sal94q3 VALUES LESS THAN TO_DATE (01-OCT-1994, DD-MON-YYYY),
 PARTITION sal94q4 VALUES LESS THAN TO_DATE (01-JAN-1995, DD-MON-YYYY),
 PARTITION sal95q1 VALUES LESS THAN TO_DATE (01-APR-1995, DD-MON-YYYY),
 PARTITION sal95q2 VALUES LESS THAN TO_DATE (01-JUL-1995, DD-MON-YYYY),
 PARTITION sal95q3 VALUES LESS THAN TO_DATE (01-OCT-1995, DD-MON-YYYY),
 PARTITION sal95q4 VALUES LESS THAN TO_DATE (01-JAN-1996, DD-MON-YYYY));
```

# Diseño Multidimensional: resumen

---

- ❑ Los recientes avances tecnológicos han permitido administrar datos como nunca antes se había podido en la historia de la computación.
- ❑ El análisis multidimensional no es nada nuevo, pero la tecnología que ha permitido implementarlo sí lo es.
- ❑ Los RDBMS siguen siendo la mejor tecnología que existe para administrar datos a un nivel atómico.
- ❑ La tecnología que implementa los MDDBS está mejorando.

# Diseño Multidimensional: resumen (2)

---

- ❑ Los grandes fabricantes de software han agregado a sus productos funcionalidades ROLAP
- ❑ EL modelamiento estrella es una buena alternativa para crear ambientes multidimensionales en bases de datos relacionales.
- ❑ El modelo estrella no se recomienda para almacenar datos con un gran nivel de desagregación (nivel atómico).

# Bodegas de datos

---

---

El Data Warehouse es una  
*arquitectura,*  
no una tecnología.



# La regla del 80/20 del Data Warehousing

---

- ❑ El 20% del esfuerzo total dará el 80% de la meta perseguida.
- ❑ El uso de software de desarrollo tradicional (usualmente) produce bajos resultados.
- ❑ Sin embargo, si se orienta el desarrollo al procesamiento de información, dejando el 80% de la funcionalidad en manos de los usuarios y un 20% de esfuerzo en mantención, se obtienen resultados enormes.

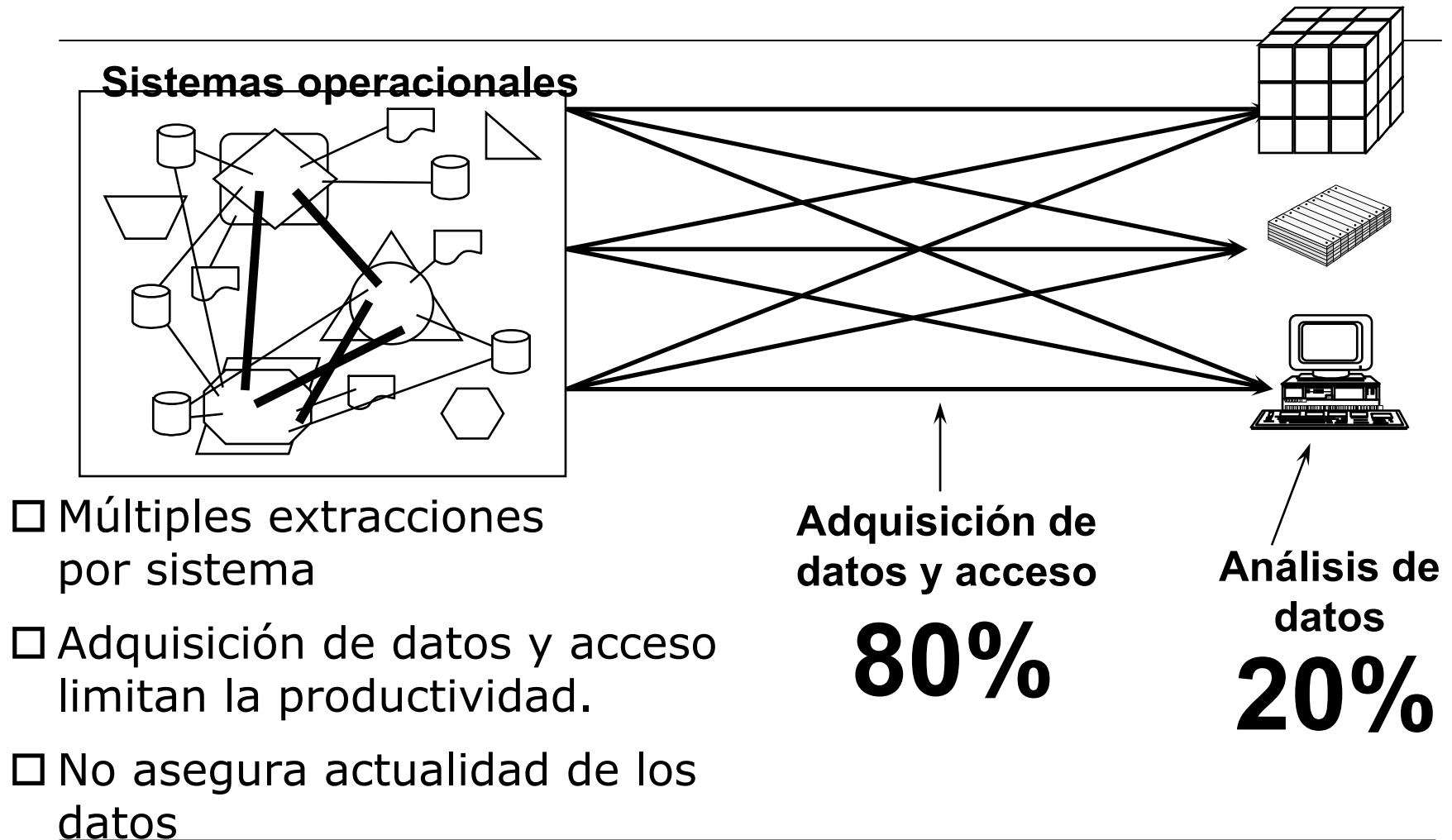
# Metas del proyecto Data Warehouse

---

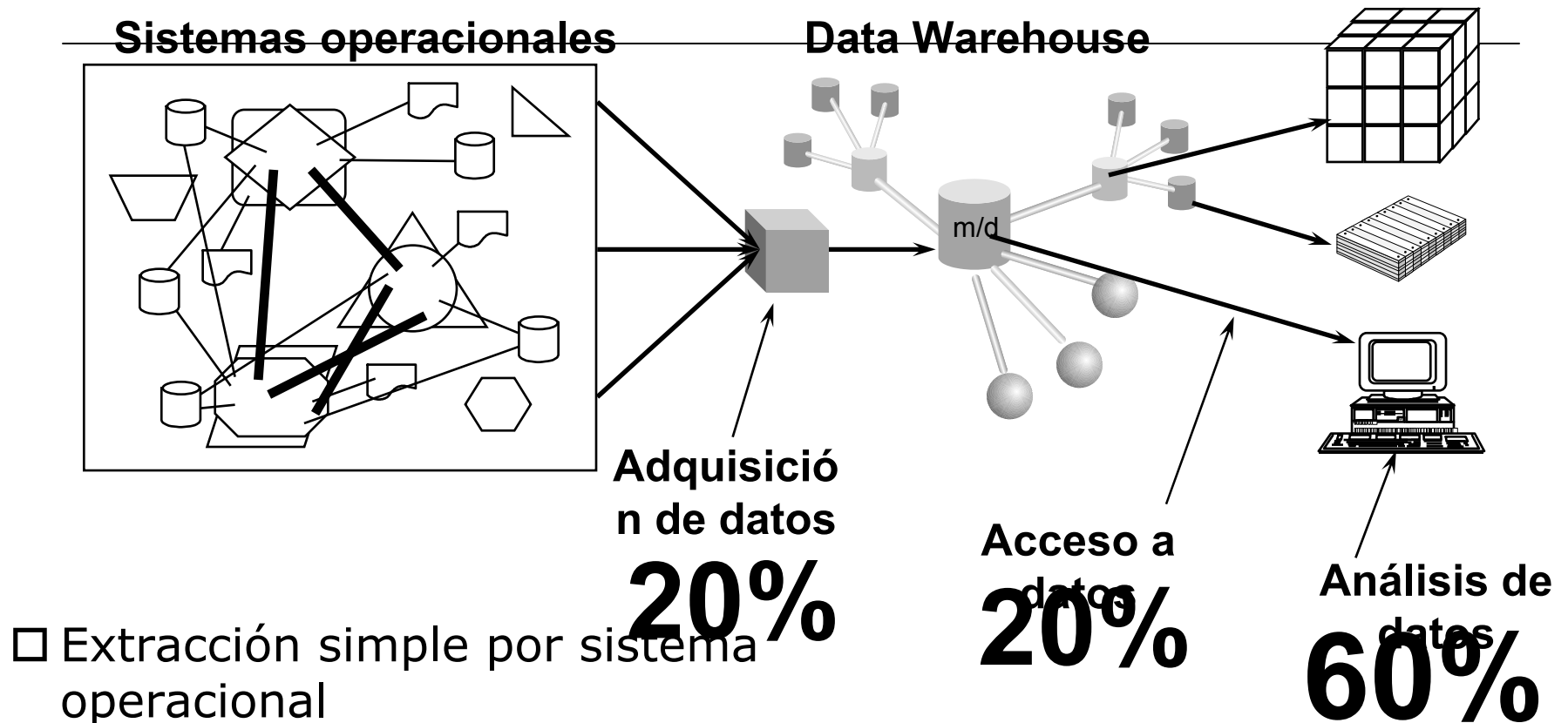
“Las organizaciones en un 75% de los casos prefieren los data warehouse para reducir el acceso de los usuarios a datos operacionales, por cuanto a mayor cantidad de accesos simultáneos, se reduce la disponibilidad de los datos y se dificulta la obtención de información para la gestión”

Fuente: Gartner Group, December 21, 1999

# Impacto del DW sobre la Adquisición y el acceso

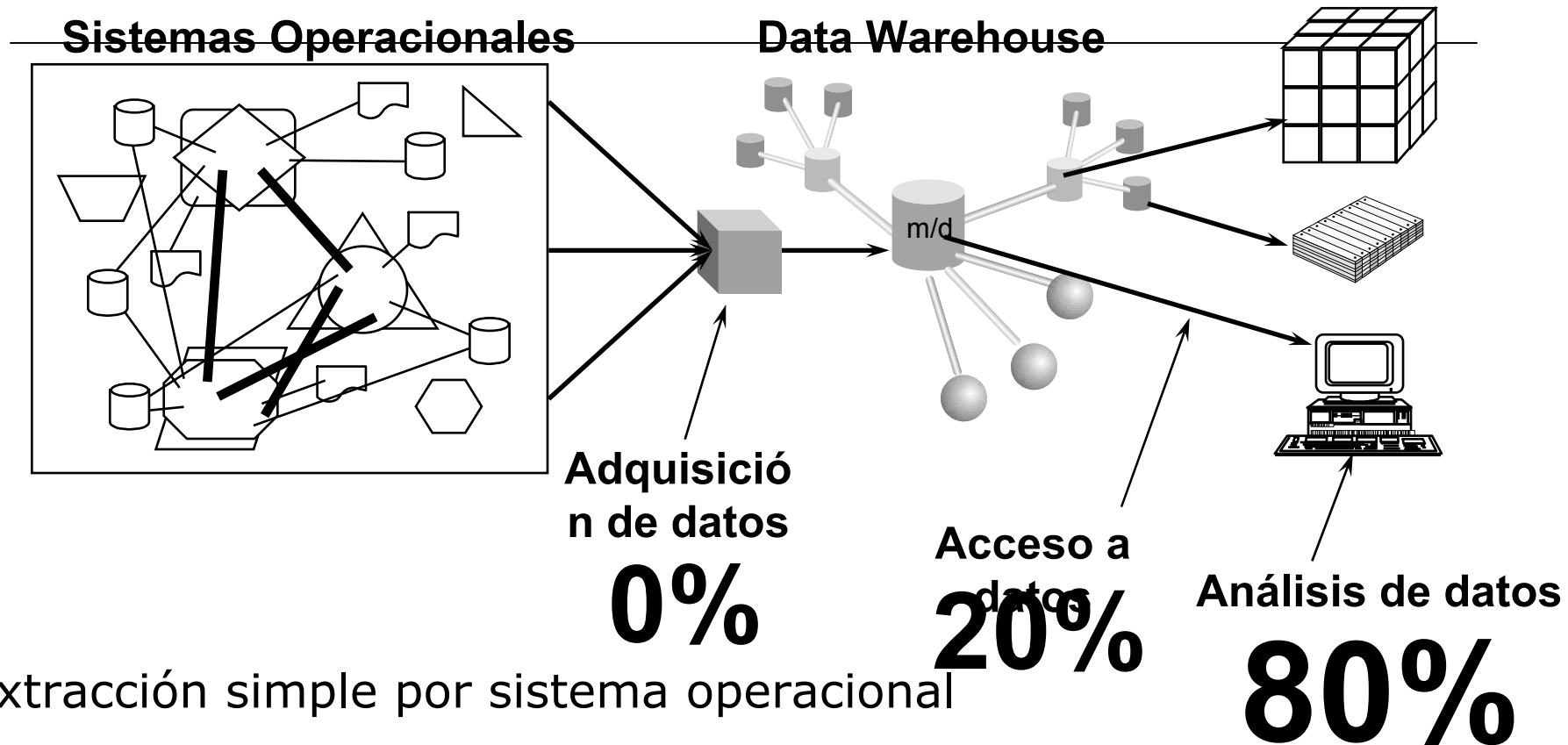


# Impacto del DW sobre la Adquisición y el acceso (2)



- ☐ Extracción simple por sistema operacional
- ☐ Metadatos asegura actualidad
- ☐ Incremento en la productividad

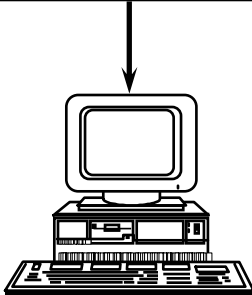
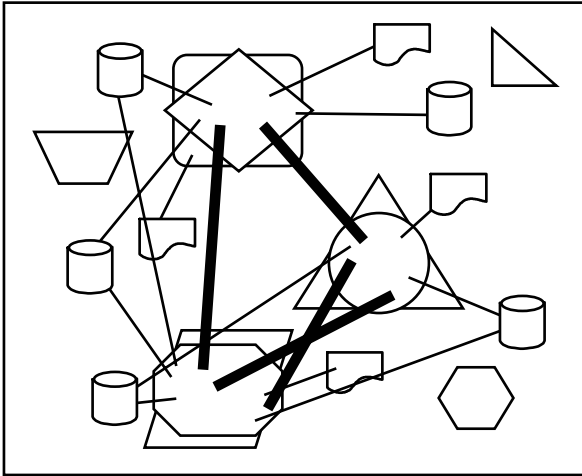
# Impacto del DW sobre la Adquisición y el acceso (3)



- Extracción simple por sistema operacional
- Metadatos asegura actualidad
- Incremento aun más de la productividad

# Impacto del DW sobre la Adquisición y el acceso (4)

Sistemas operacionales



Comparar estos  
costos con el de un

DW

Costo total para:

- ♦ Se requiere tiempo de CPU para los procesos de extracción
- ♦ Se requiere capacidad de Discos para almacenar las múltiples extracciones y procesamiento
- ♦ Programas destinados a la extracción
- ♦ Licencias del DBMS
- ♦ Tiempos invertidos en satisfacer requerimientos de información

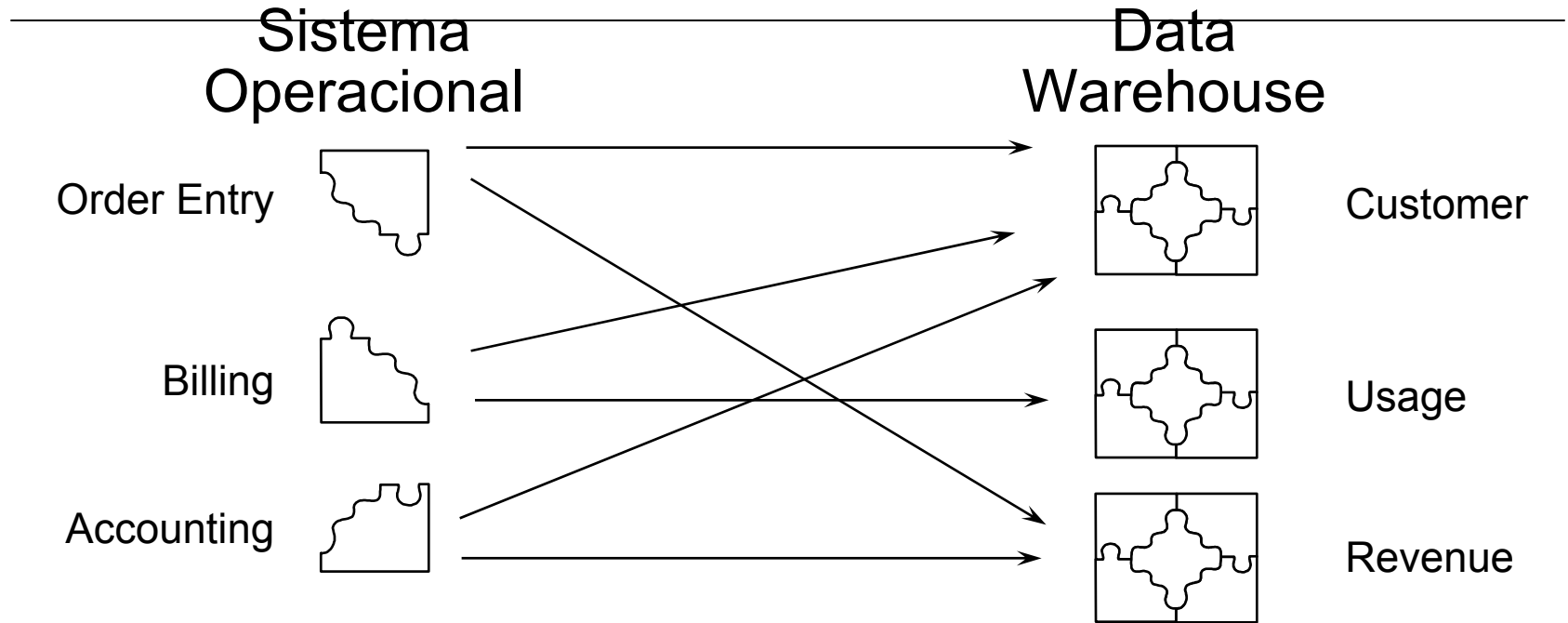
# Definición de un Data Warehouse

---

"... es una colección de datos orientados a temas, integrados, variantes en el tiempo, no volátiles, que ayudan en el proceso de toma de decisiones ".

W.H. Inmon

# Características: Por área temática



- ♦ Los datos operacionales son organizados por procesos específicos y se mantienen en sistemas separados

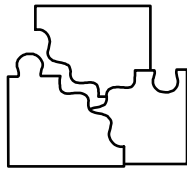
- ♦ Los datos del DW son organizados por área y poblados desde muchos sistemas operacionales



# Características: Integración

---

Sistema  
operacional

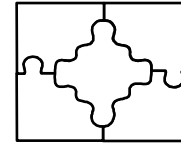


Aplicaciones específicas

- ♦ Las aplicaciones y sus bases de datos son diseñadas y construidas por separado
- ♦ Cubre largos períodos de tiempo

---

Data  
Warehouse

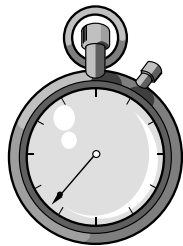
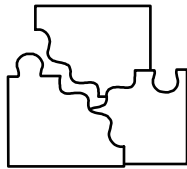


Integradas

- ♦ Integradas desde el comienzo
- ♦ Diseño realizado una sola vez, implementación iterativa sobre períodos cortos de tiempo

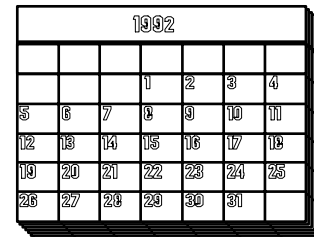
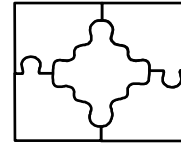
# Características: Variables en el tiempo

Sistema  
Operacional



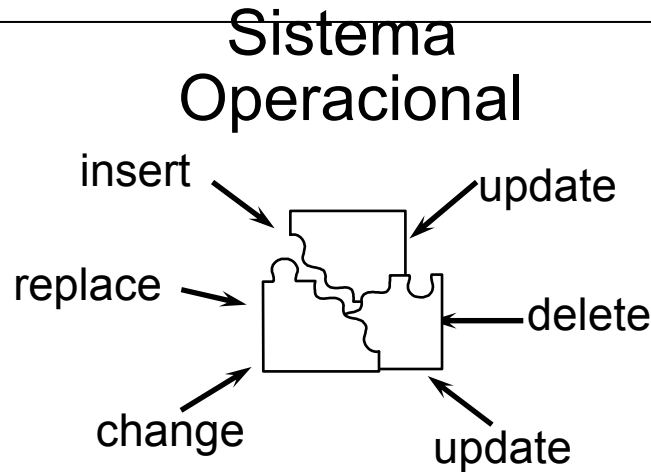
- ♦ Principalmente, datos de hoy!

Data  
Warehouse



- ♦ Generalmente, mantiene datos históricos

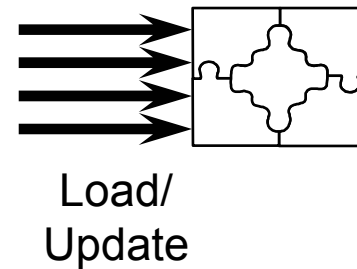
# Características: Volatilidad



Cambios constantes

- ♦ Modificación constante
- ♦ Cambio en los datos de acuerdo a la necesidad

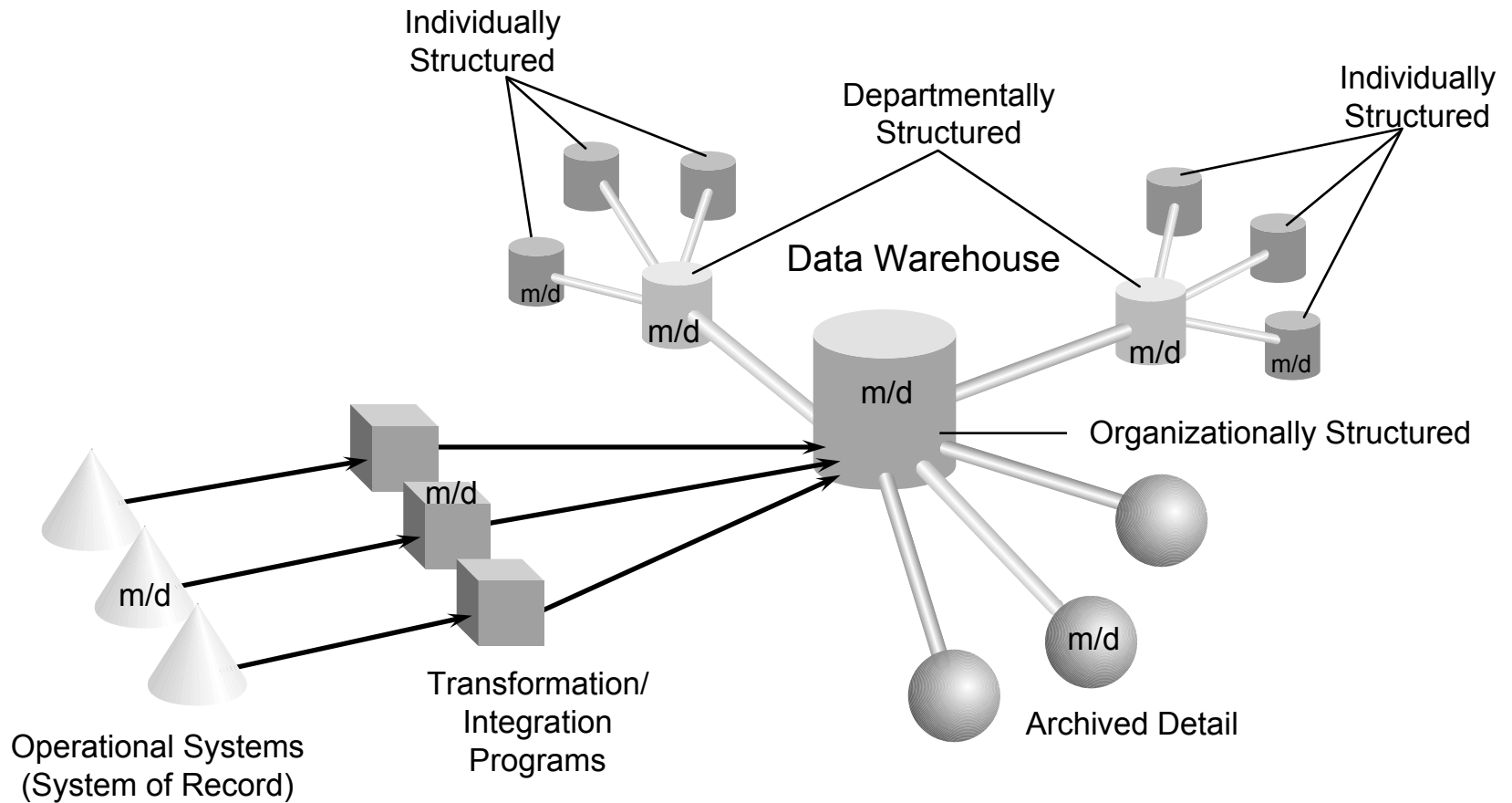
Data Warehouse



Consistentes en un punto del tiempo

- ♦ Se agregan datos regularmente, pero rara vez se realizan cambios directamente
- ♦ No significa que los DW nunca sufran cambios!!

# El ambiente DW



m/d: Metadata

# Un ambiente DW maduro

