Sistemas de Información para el Análisis de GVDatos

Instituto de Computación - Facultad de Ingeniería

2024

Diseño Lógico

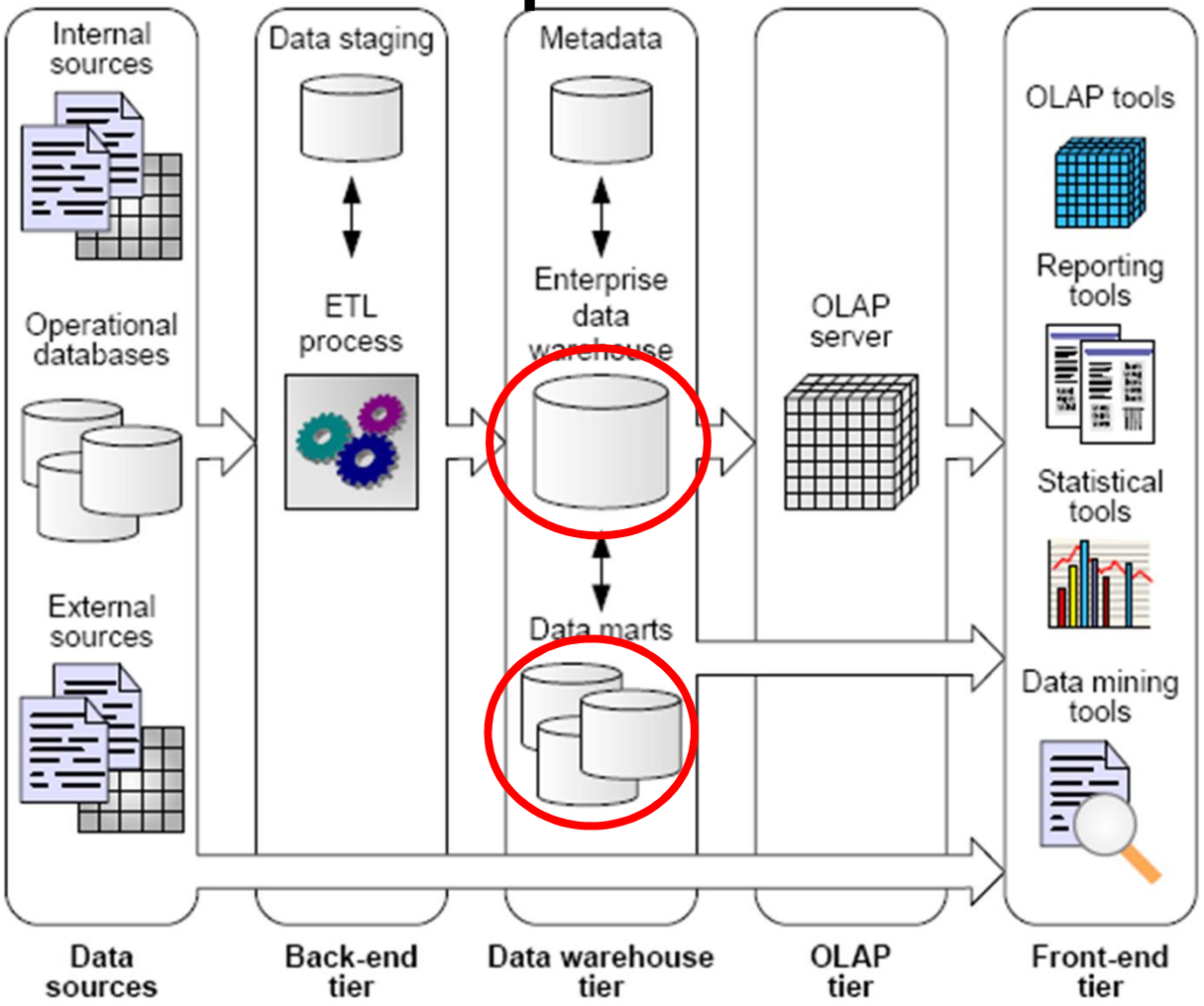
Temario: Diseño Lógico

- Introducción
- Modos de almacenamiento
- Modelo Dimensional
- Proceso de Diseño
- Traducción desde el esquema conceptual
- Refinamiento del esquema lógico relacional

Motivación

- Objetivos del Diseño Lógico del DW.
  - Construir el esquema lógico del DW o DM.
    - Sobre un DBMS Relacional o Multidimensional.
- Problemas a resolver.
  - Transformaciones de modelos y especificaciones:
    - Esquema Conceptual: abstracto.
    - Esquema Lógico: implementado.
  - Obtener estructuras adecuadas a la función.
  - Tener en cuenta la “Carga de Trabajo”.

Arquitectura típica de SDW



[Malinowski 2008]

Objetivo: el DW

- Propiedades del DW.
  - Volúmenes importantes de datos.
  - Operación crítica:
    - Consultas interactivas complejas, muy frecuentemente con lógica multidimensional.
  - Actualización:
    - Batch (en lotes).
    - Volúmenes de datos importantes.
    - Los datos pasan por varias transformaciones.
- Solución depende del Modelo Lógico.
  - Relacional o Multidimensional.

Modos de Almacenamiento

ROLAP

- DW en BD Relacional (ROLAP)
  - Modelo estandarizado.
    - Tanto en estructura como lenguaje de manipulación (SQL).
  - Flexibilidad y potencial para consultas ad-hoc.
    - Especialmente consultas “por clave” y “por condición”.
  - Transformación compleja de datos y operaciones MD hacia estructuras y operaciones Relacionales.
  - Mala performance en Joins y Sumarizaciones.
    - Se ha avanzado mucho en optimización de consultas, índices, joins, etc., para aplicaciones DW.
  - Muchos sistemas comerciales utilizan almacenamiento ROLAP o HOLAP para grandes volúmenes de datos.

MOLAP

- DW en BD Multidimensional (MOLAP)
  - Modelo MOLAP:
    - Datos y operaciones multidimensionales se mapean directamente
    - Almacenan datos en estructuras especializadas.
  - Muy alta performance en consultas dimensionales.
  - Poco usables para consultas “por clave” o “por condición”.
  - Muy poco flexibles para actualización:
    - Solo append a la BD (carga incremental).
  - Modelo menos estandarizado.
    - No hay estándares de estructura pero si de lenguaje de consulta (MDX).

ROLAP – MOLAP – HOLAP

- MOLAP:
  - Acceso a datos relacionales sólo durante la carga.
    - Adecuado cuando se quiere independencia del DBMS relacional.
  - Mejor performance para sistemas accedidos con mucha frecuencia.
- ROLAP:
  - Menor espacio en disco. Adecuado para grandes conjuntos de datos.
- HOLAP:
  - Balance entre performance y espacio en disco.
    - Adecuado cuando se accede frecuentemente a datos resumidos y los datos detallados son poco accedidos.

Otros modelos – Big Data

- Definición de estructuras multidimensionales sobre otros modelos de datos
  - BDs documentales
    - Formato JSON, por ej. MongoDB
  - BDs de grafos
    - Ej., Neo4J
  - BDs columnares
    - Ej., Cassandra
  - BDs clave-valor
    - Ej., Hadoop

Diseño Lógico

Modelo Dimensional

[Kim 96]

Almacenamiento ROLAP

Diseño de un DW Relacional

- Características del DW
  - Acceso y mantenimiento de datos
    - Consultas complejas
    - Se considera solo-lectura. El mantenimiento no se hace vía sistema OLTP, sino en forma "batch".
    - Usuario final accede directamente al DW con herramientas de consulta (OLAP)
  - Modelo Relacional poco adecuado para consultas dimensionales.
    - Implican Joins entre varias tablas y Sumarizaciones, que son costosas y poco optimizables.

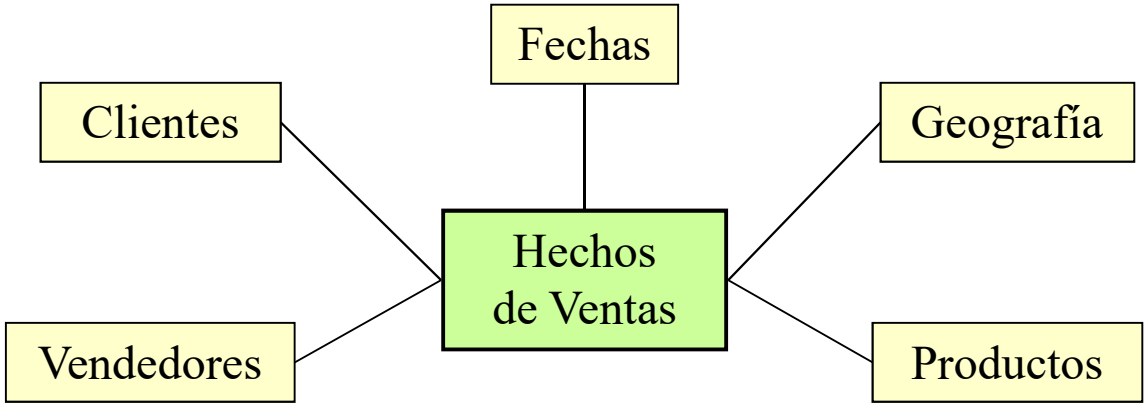
⇨ **Técnicas de diseño de DW *diferentes* a las tradicionales para BDs relacionales**

Diseño de un DW Relacional

- Modelo Dimensional [Kim96]
  - Orientados a consultas OLAP
  - Se representan los conceptos MD sobre el modelo Relacional .
  - Se trata de minimizar joins y totalizaciones:
    - Redundancia en cálculos.
    - Fuerte tendencia a desnormalizar.
- Estructuras básicas
  - Tablas de hechos (*fact tables*)
    - Donde se guardan las medidas numéricas del negocio
    - Granularidad: Intersección de las dimensiones
  - Tablas de dimensión (*dimension tables*)
    - Donde se guardan las descripciones de las dimensiones
    - Jerarquías: desnormalizadas o normalizadas

Esquema estrella - Estructura

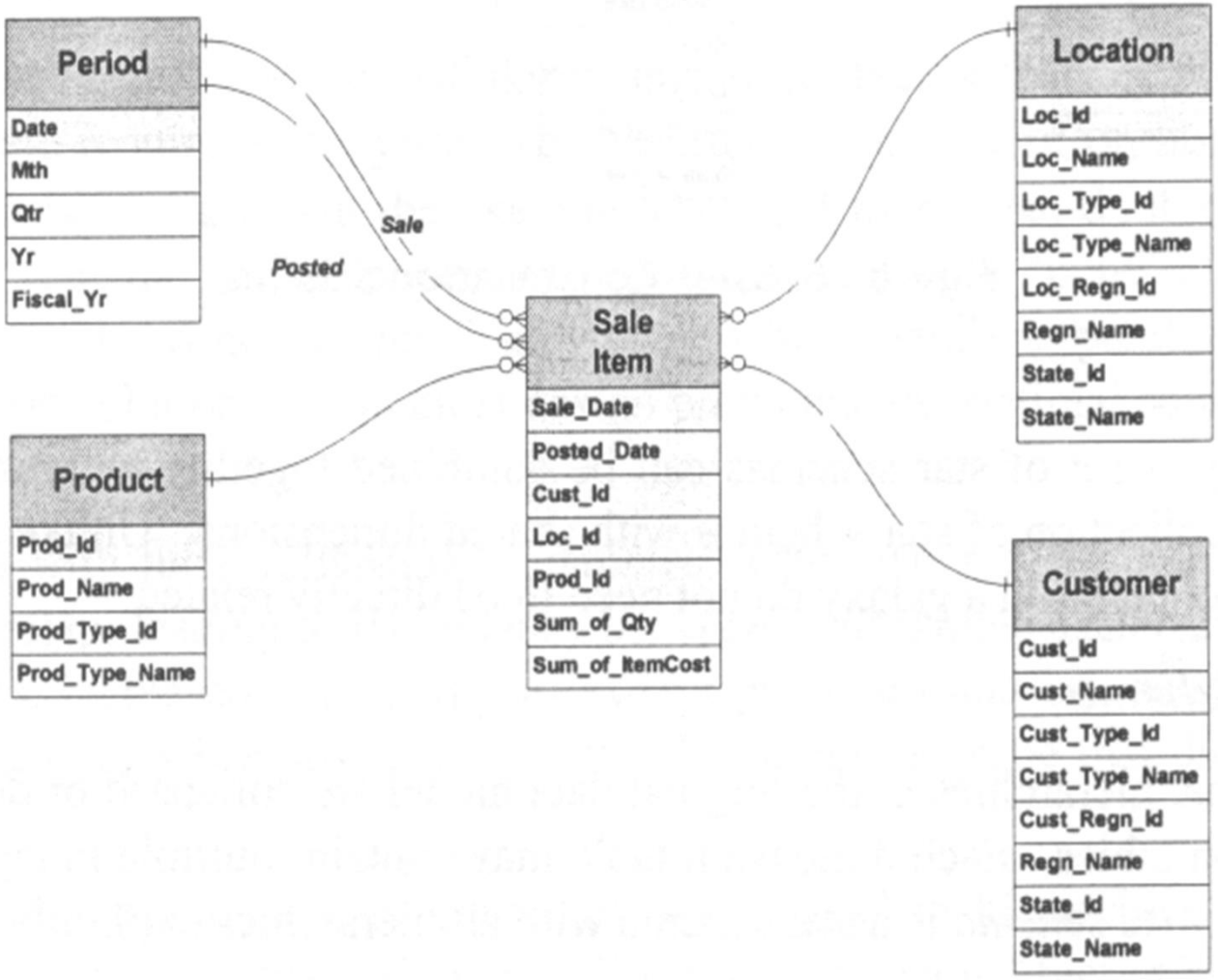
- Tabla central: hechos
- Conjunto de tablas usualmente con menos registros organizadas alrededor: dimensiones



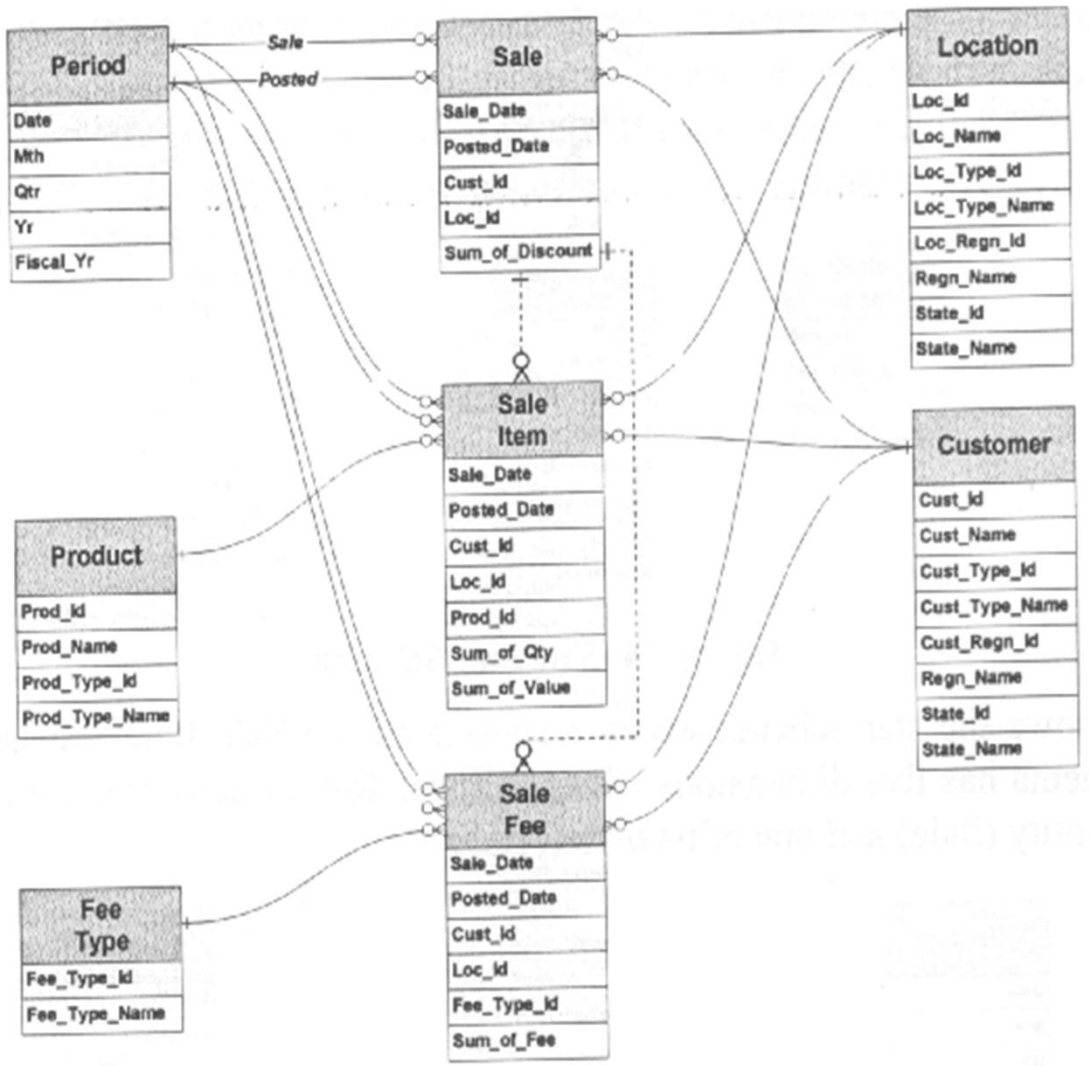
Esquema estrella - Características

- Tablas de dimensión.
  - Desnormalizadas y con jerarquías embebidas.
- Tablas de hechos.
  - Unidas a todas las de dimensión por relaciones 1:N
  - Clave primaria = concatenación de las claves primarias de todas las dimensiones.

Esquema Estrella (ejemplo)

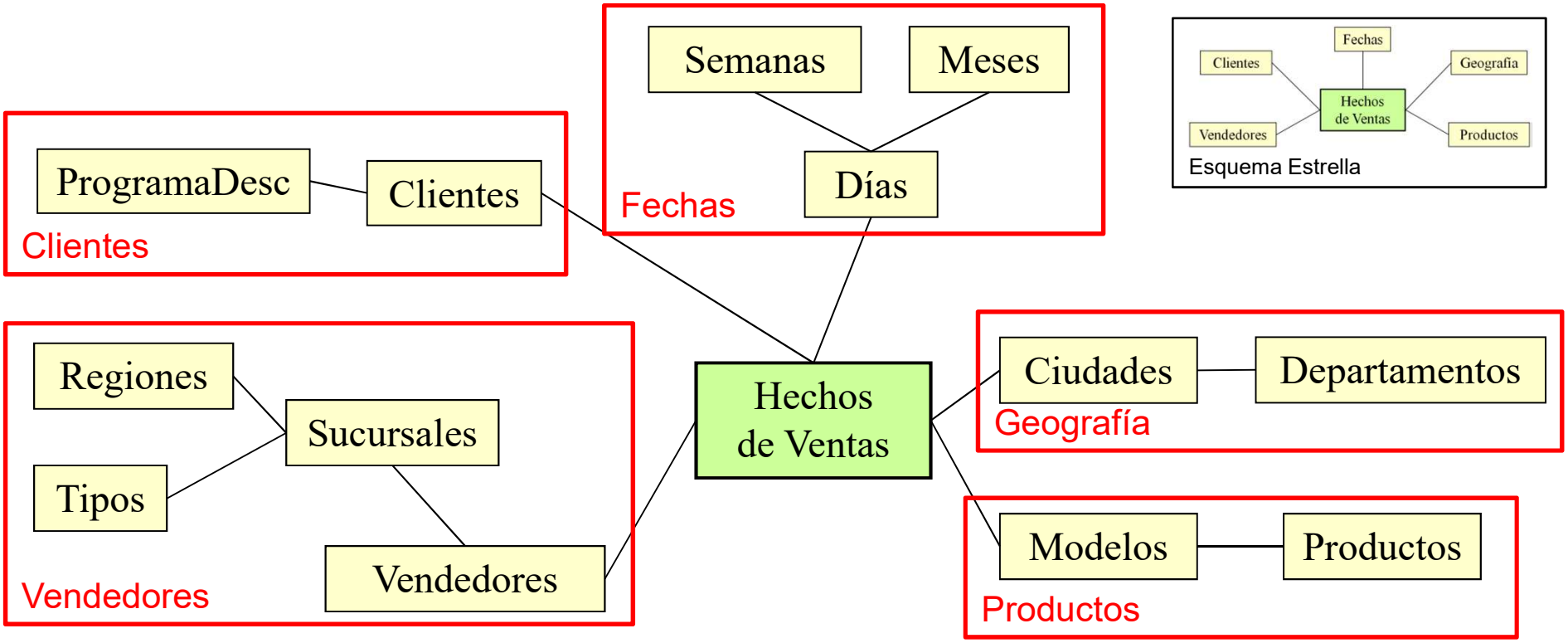


Esquema Estrella: Constelación y Galaxia



Esquema Snowflake - Estructura

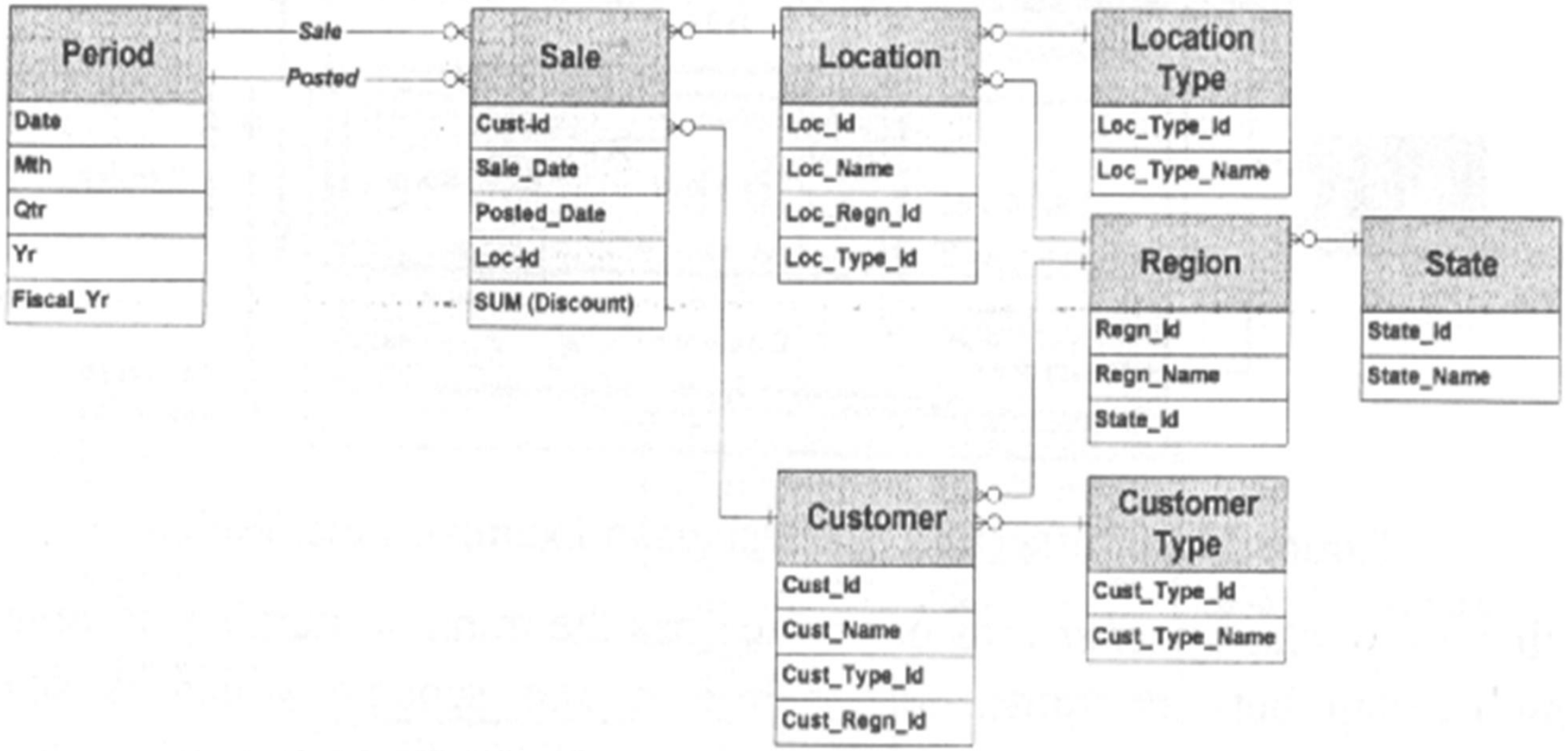
- Es el resultado de descomponer una o más tablas de dimensiones en varias tablas, que forman jerarquías.



Esquema Snowflake - Características

- Es un Esquema Estrella con las dimensiones normalizadas según las jerarquías que contienen.
- Tabla de hechos
  - Igual que en esquema estrella
- Tablas de dimensión
  - Una tabla para cada nivel de una dimensión
  - Las tablas de los niveles de una misma dimensión se relacionan a través de foreign- keys

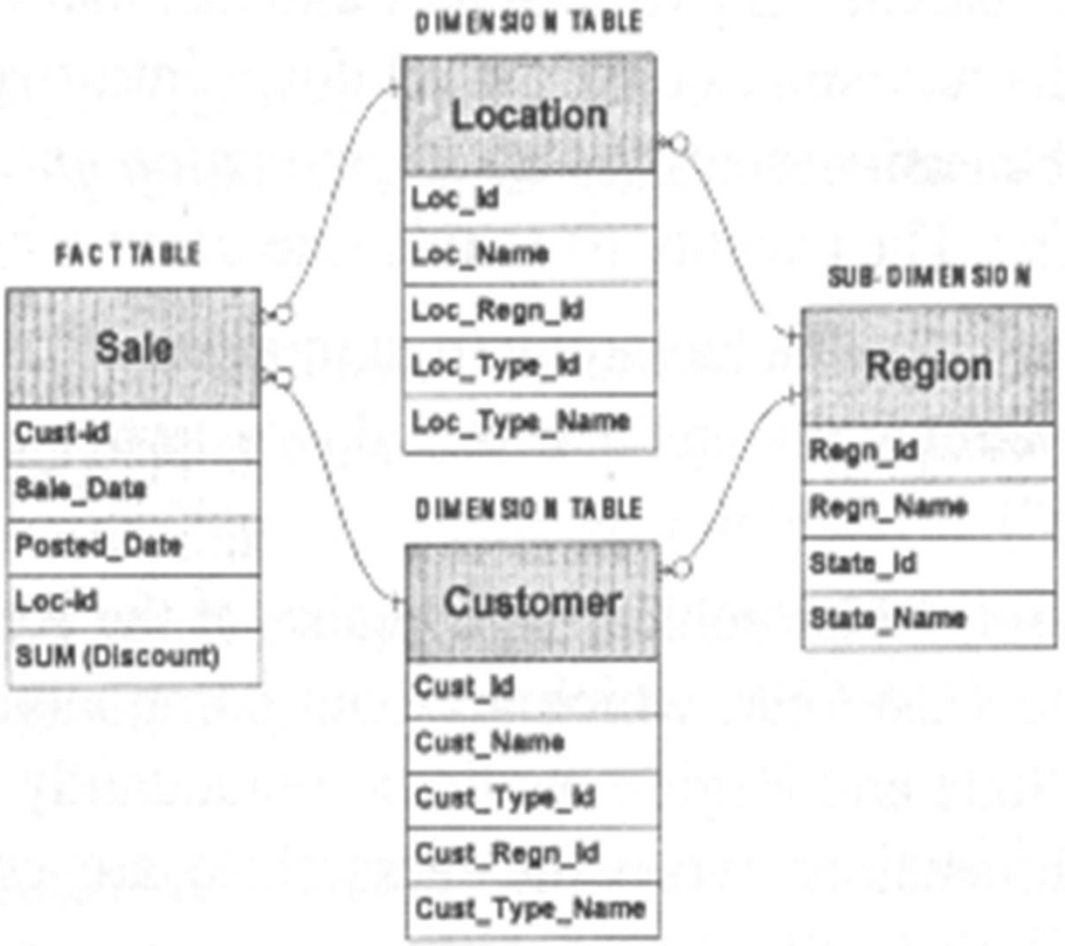
Esquema Snowflake - Ejemplo



Esquema Star Cluster

- Estructura:
  - Es una combinación de Star y Snowflake.
  - Selectivamente separa los fragmentos de jerarquías compartidos entre diferentes dimensiones.
  - El resto de las jerarquías se desnormalizan.
- Características:
  - Mínimo número de tablas, y a la vez evita solapamiento entre dimensiones.
- Referencia: [Moo00]

Esquema Star Cluster - Ejemplo



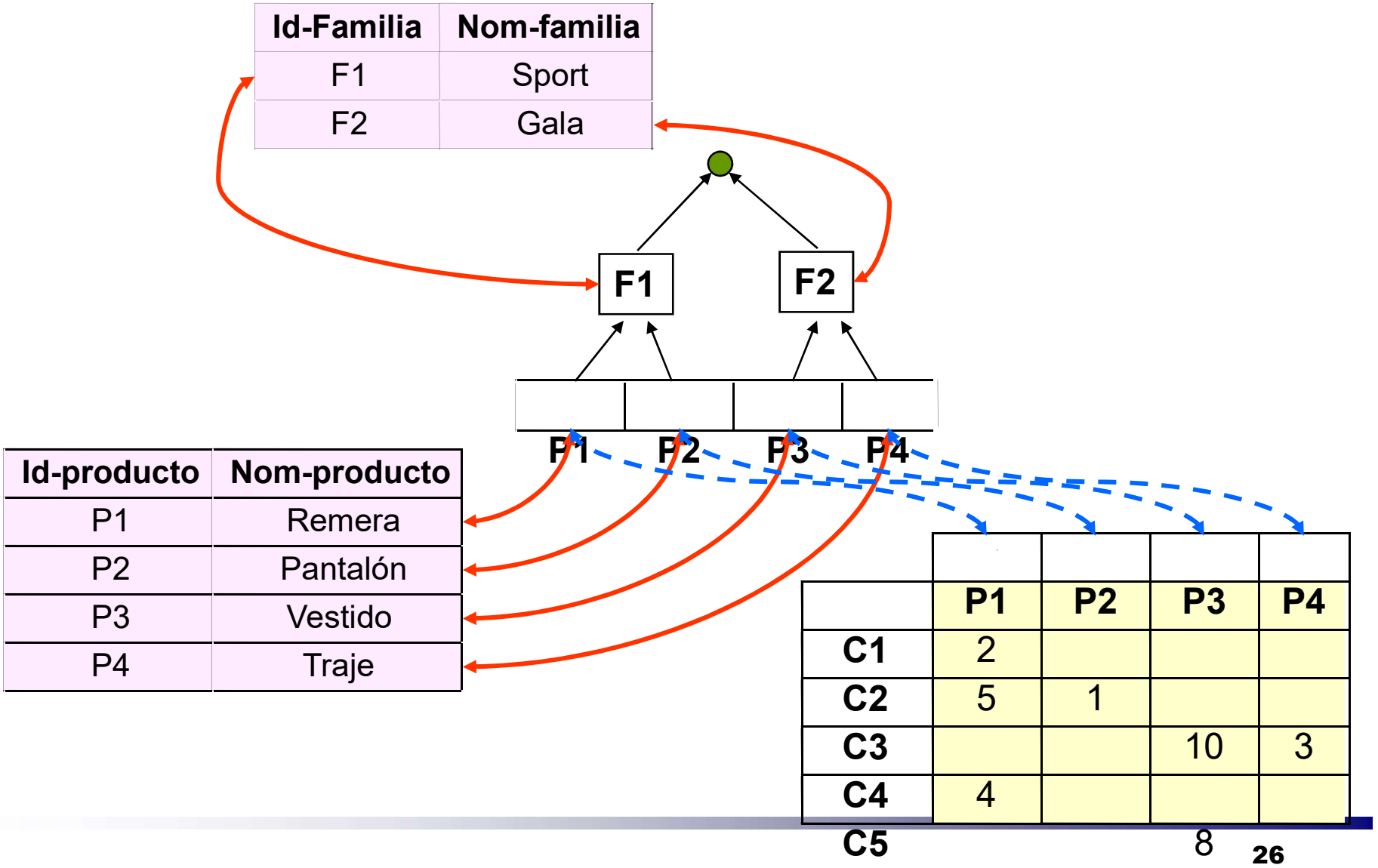
Diseño Lógico

Almacenamiento MOLAP

Diseño de un DW Multidimensional

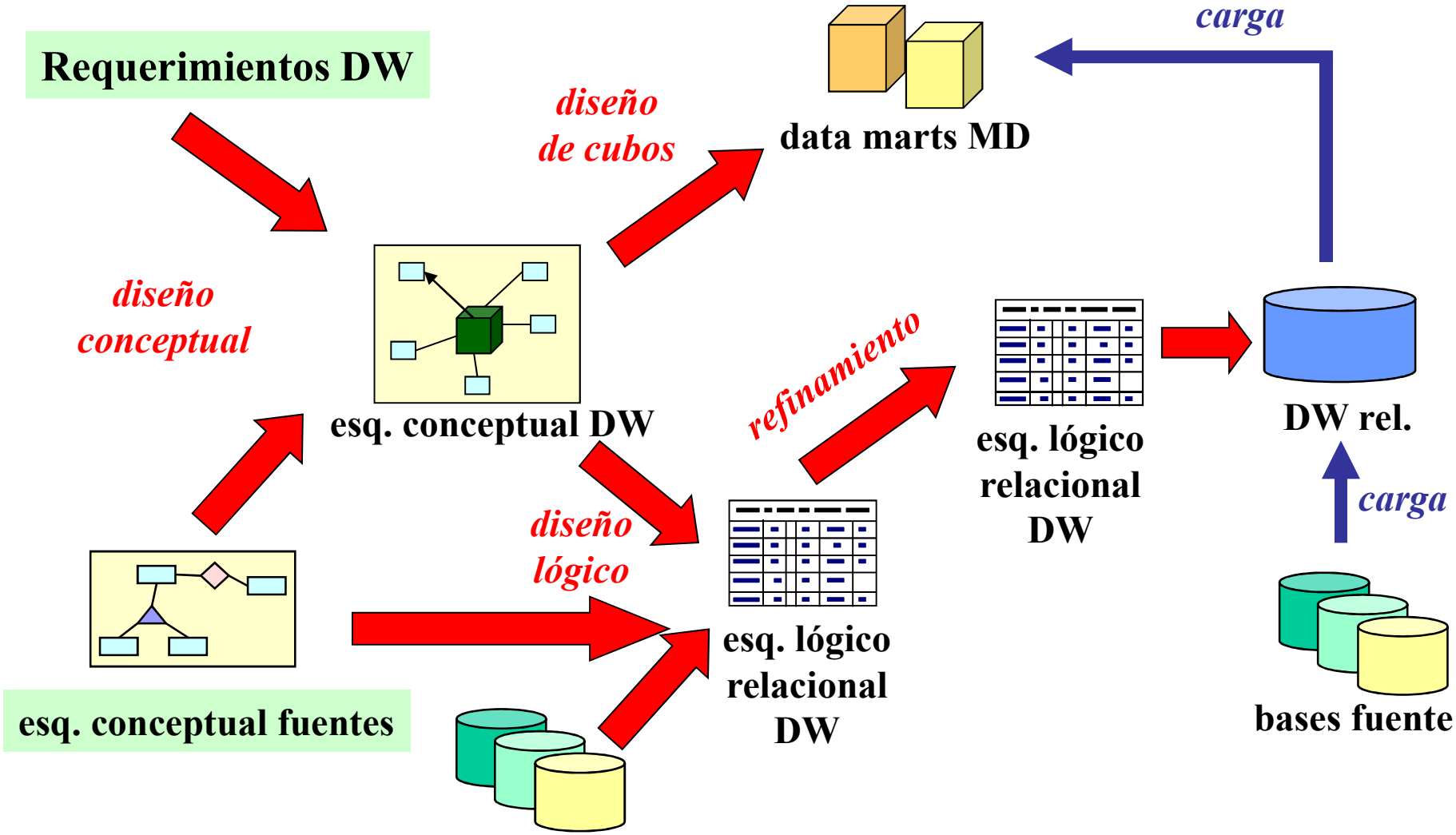
- Modelos Multi-dimensionales
  - Orientados a consultas OLAP
  - Se implementan los conceptos MD sobre estructuras MD.
  - Estructuras propietarias.
- Intuición de estructuras
  - Arrays multidimensionales.
    - Granularidad: Intersección de las dimensiones
    - Dimensiones en los ejes, medidas en las celdas.
    - Unidas a jerarquías de dimensiones por referencias.
  - Jerarquías de dimensiones
    - Árboles de jerarquías (índices)
    - Diccionarios con datos descriptivos.

Diseño de un DW Multidimensional



Proceso de Diseño

Proceso de Diseño



Diseño Lógico

- Consiste en:
  - Diseñar las BDs a partir de esquemas conceptuales
- Problemas a resolver:
  - Ofrecer acceso eficiente y simple
  - Llevar los esquemas conceptuales al modelo implementado
  - Tener en cuenta requerimientos no funcionales

Metodología

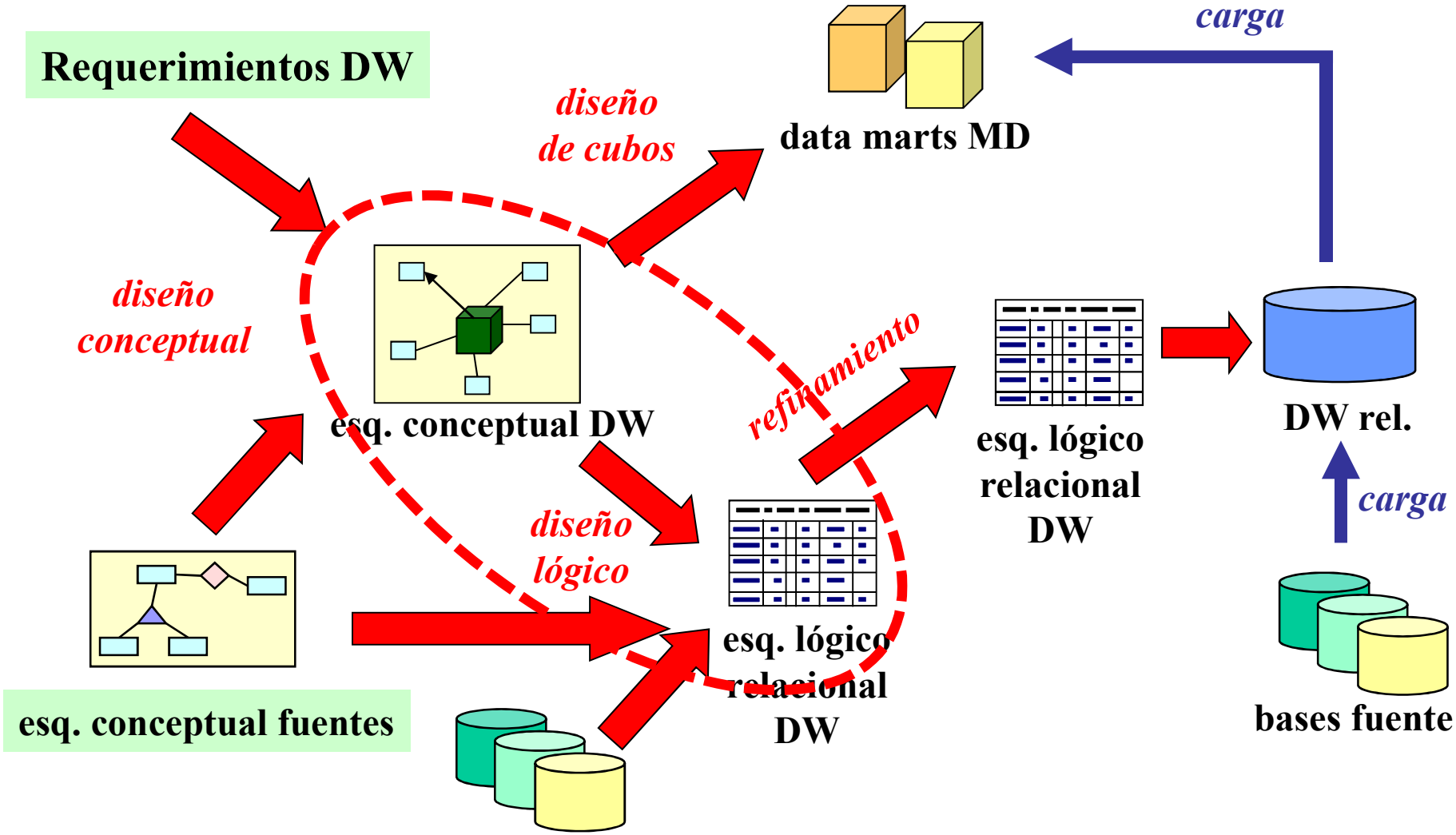
- Determinar requerimientos no funcionales:
  - Uso del sistema
  - Performance requerida
  - Limitaciones de espacio en disco
- Elegir modo de almacenamiento:
  - MD, relacional, híbrido, etc.
- Definir estrategias de almacenamiento
  - Para satisfacer reqs. no funcionales
- Adaptar las estructuras a restricciones de las herramientas

Diseño Lógico Relacional

- En 2 pasos
  - Traducción desde esquema conceptual a esquema lógico relacional
  - Refinamiento del esquema lógico relacional

Traducción Esquema Conceptual – Esquema Lógico Relacional

Proceso de Diseño

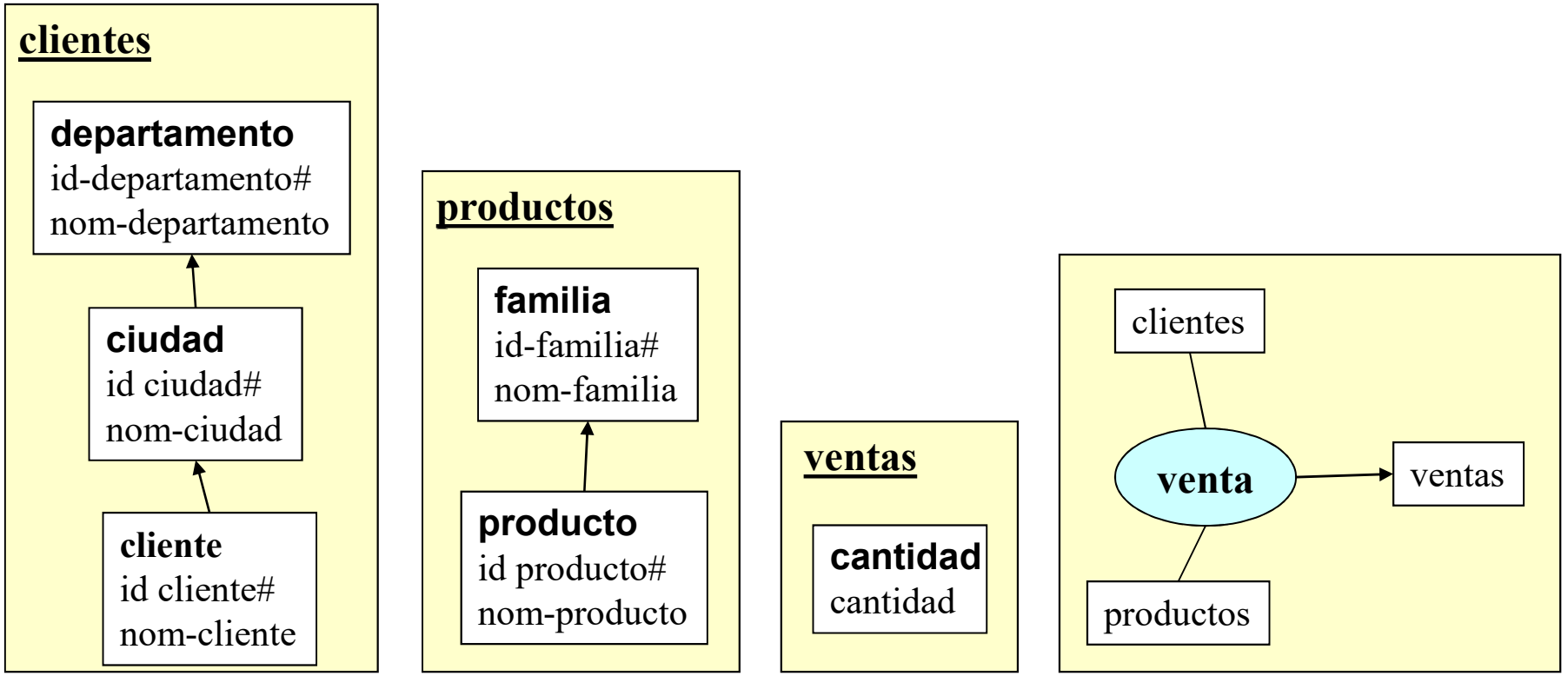


Almacenamiento

- Se resuelve almacenamiento de:
  - Dimensiones
  - Relaciones dimensionales
  - Agregaciones
    - Resúmenes pre-calculados (*aggregates*)
    - Un registro representa un resumen de registros de nivel básico de una tabla de hechos o cubo

Traducción a un esq. lógico

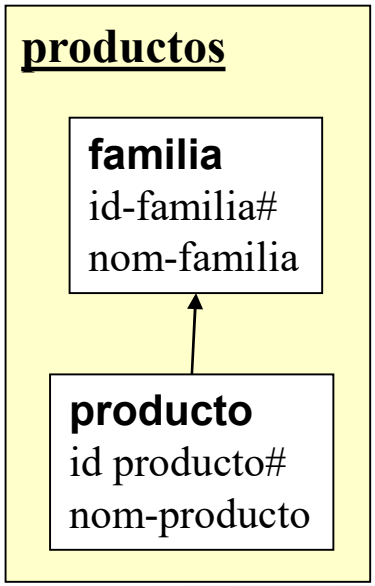
- Ejemplo:
  - Cantidad vendida por cliente por producto.

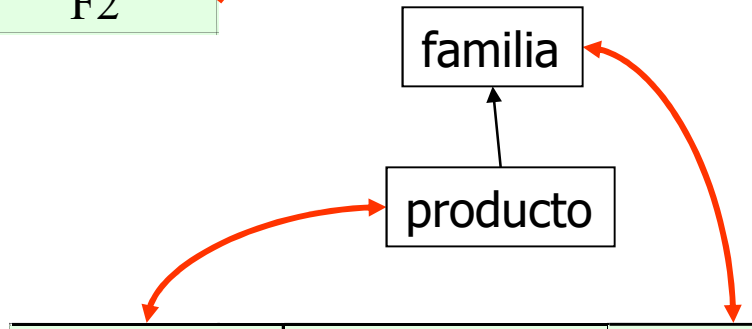


Traducción a relacional

Id-producto	Nom-producto	Id-Familia
P1	Remera	F1
P2	Pantalón	F1
P3	Vestido	F2
P4	Traje	F2

Id-Familia	Nom-familia
F1	Sport
F2	Gala

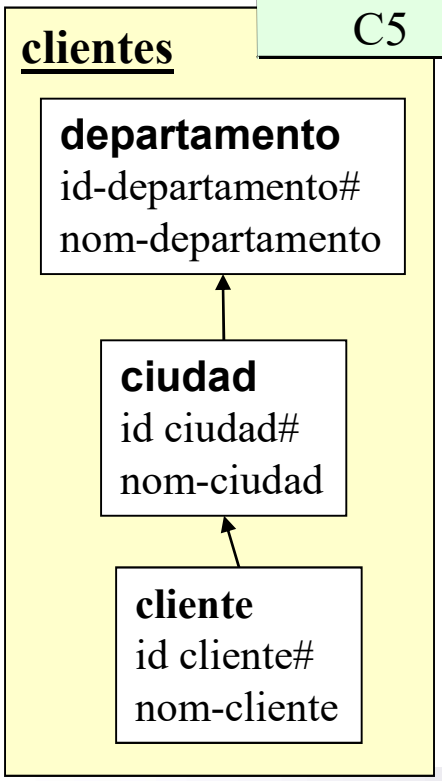




Id-producto	Nom-producto	Id-Familia	Nom-familia
P1	Remera	F1	Sport
P2	Pantalón	F1	Sport
P3	Vestido	F2	Gala
P4	Traje	F2	Gala

Traducción a relacional

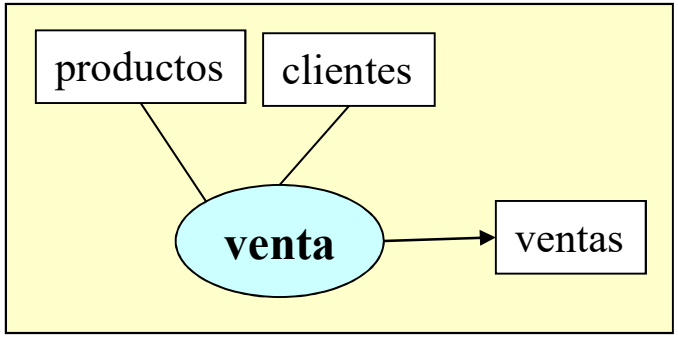
Id-cliente	Nom-cliente	Id-ciudad	Nom-ciudad	Id-depto	Nom-depto
C1	Juan	1	Mercedes	K	Soriano
C2	Ana	1	Mercedes	K	Soriano
C3	Pablo	2	Colonia	L	Colonia
C4	José	2	Colonia	L	Colonia
C5	María	3	Carmelo	L	Colonia



Traducción a relacional

Id-producto	Nom-producto	Id-Familia	Nom-familia
P1	Remera	F1	Sport
P2	Pantalón	F1	Sport
P3	Vestido	F2	Gala
P4	Traje	F2	Gala

Id-cliente	Id-producto	Cantidad
C1	P1	2
C2	P1	5
C2	P2	1
C3	P3	10
C3	P4	3
C4	P1	4
C5	P3	8



Id-cliente	Nom-cliente	Id-ciudad	Nom-ciudad	Id-depto	Nom-depto
C1	Juan	1	Mercedes	K	Soriano
C2	Ana	1	Mercedes	K	Soriano
C3	Pablo	2	Colonia	L	Colonia
C4	José	2	Colonia	L	Colonia
C5	María	3	Carmelo	L	Colonia

Traducción a relacional

Agregaciones

Id-producto	Nom-producto	Id-Familia	Nom
P1	Remera	F1	S
P2	Pantalón	F1	Sport
P3	Vestido	F2	Gala
P4	Traje	F2	Gala

Id-Familia	Nom-familia
F1	Sport
F2	Gala

Id-cliente	Id-producto	Cantidad
C1	P1	2
C2	P1	5
C2	P2	1
C3	P3	10
C3	P4	3
C4	P1	4
C5	P3	8

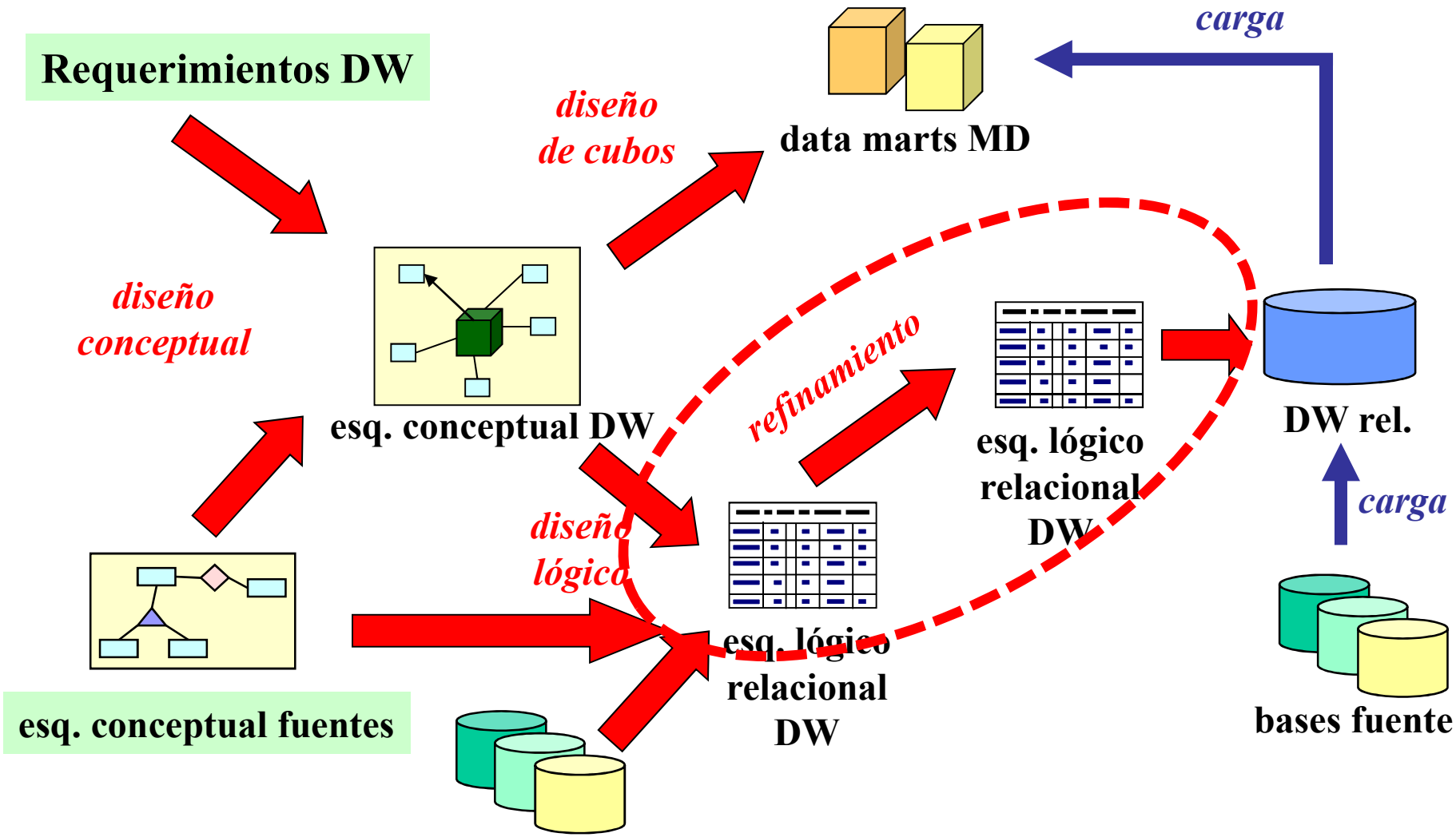
Id-ciudad	Id-familia	Cantidad
1	F1	8
2	F2	13
2	F1	4
3	F2	8

Id-cliente	Nom-cliente	Id-ciudad	Nom-ciudad	Id-depto	Nom-depto
C1	Juan				
C2	Ana				
C3	Pablo				
C4	José				
C5	María	3	Carmelo	L	Colonia

Id-ciudad	Nom-ciudad	Id-depto	Nom-depto
1	Mercedes	K	Soriano
2	Colonia	L	Colonia
3	Carmelo	L	Colonia

Refinamiento del Esquema Lógico Relacional

Proceso de Diseño



Refinamiento del esquema relacional

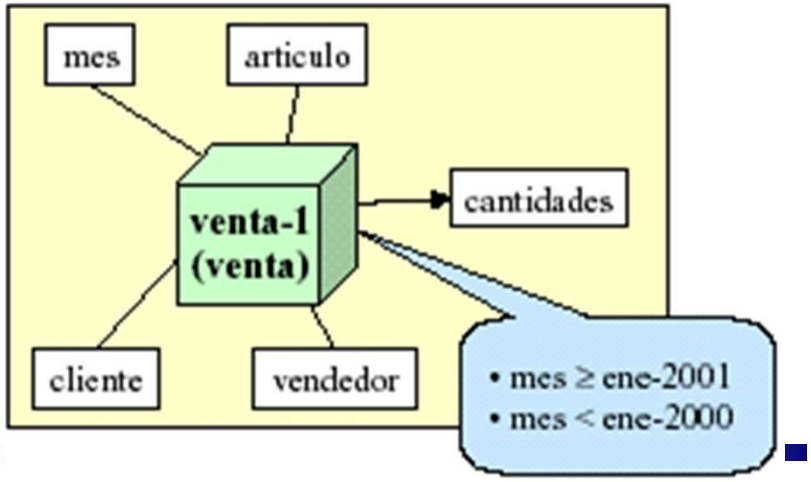
- Objetivo:
  - Utilizar experiencia/pautas para resolver problemas comunes
- Pautas:
  - Fragmentación de cubos
  - Agregaciones
  - Aditividad
  - Jerarquías
  - Versiones
  - Mini-dimensiones
  - Dimensiones “Many-to-Many”

- Referencias:
  - [Gol09][Mal08][Kim96][Kim02][Son01][Gol00]

Fragmentación Horizontal de Cubos

- Introducción.
  - Los Cubos tendrán asociados grandes conjuntos de datos, que comparten el mismo esquema pero pueden ser usados en forma diferente.
- Definición.
  - Consiste en la especificación de *bandas o franjas* correspondientes a **particiones horizontales de la tabla de hechos.**
- Ejemplo:

• Se definen *franjas* según el mes de Venta:
  • Antes de enero 2001.
  • Después de enero 2001.



Criterios de uso

- Fragmentación Horizontal de Cubos
  - Cuantas más franjas se definan, serán de menor tamaño y se obtendrá mejor tiempo de respuesta al consultarlas
  - Por otro lado, si algunas consultas involucran varias franjas, estas operaciones tendrán menos performance
    - Requieren drill-across
  - La redundancia (franjas no disjuntas) debe balancearse con restricciones de almacenamiento
  - Factores a analizar:
    - Tamaño del cubo
    - Subconjunto de registros que son usados juntos frecuentemente
    - Restricciones de almacenamiento

Agregaciones

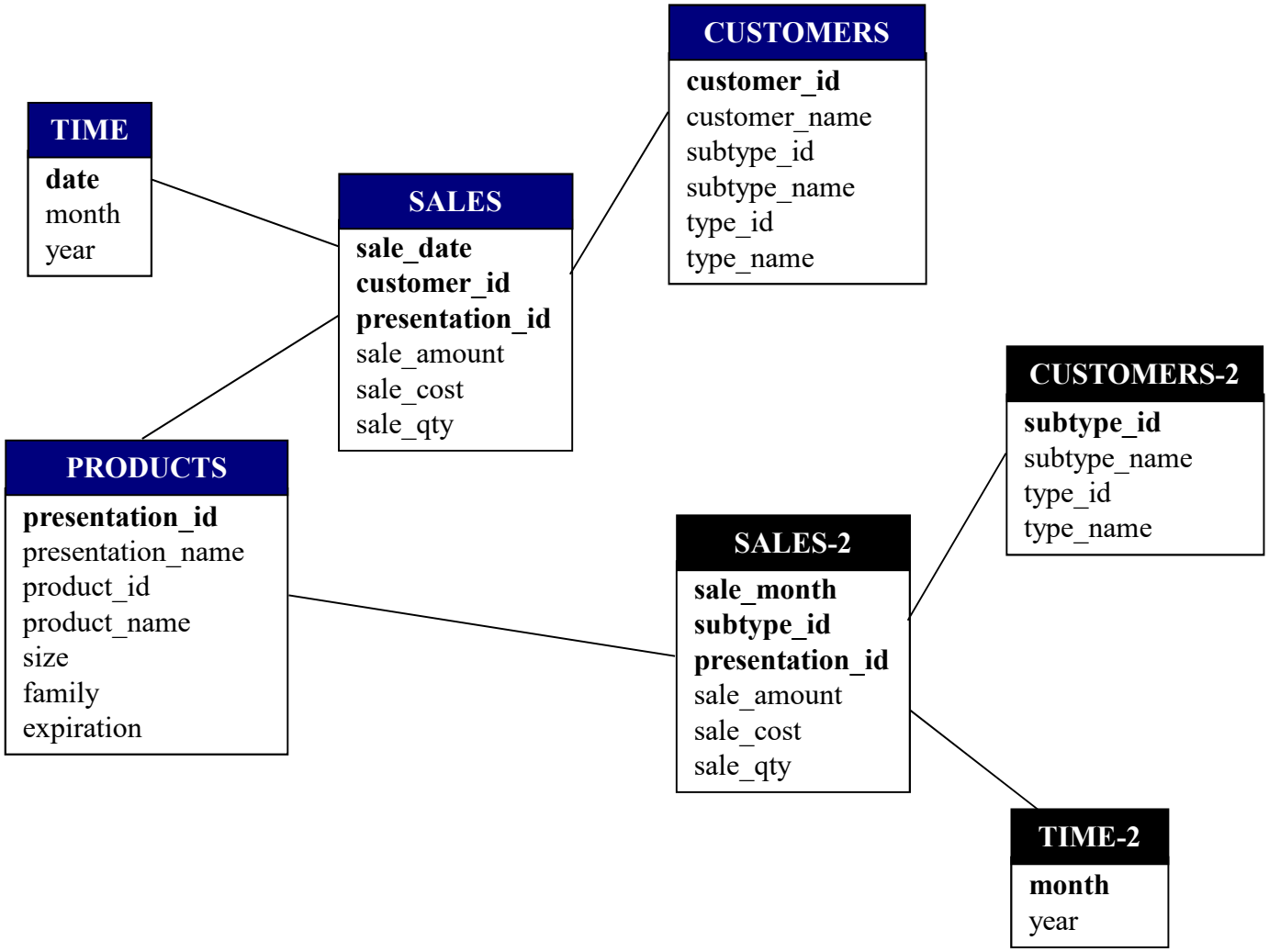
- Objetivo:
  - Tener un resumen precalculado de una tabla de hechos.
    - Tablas de hechos ***derivadas*** o ***agregaciones***.

- Se eligen:
  - Conjunto de atributos “que se agregan”.
  - Conjunto de atributos “por los que se agrega” (agrupar).

Agregaciones

- Características:
  - Perdemos información:
    - No se puede re-construir los detalles a partir de la agregación.
  - Las claves foráneas de la agregación serán diferentes de las de la tabla de hechos primaria.
  - Se debe prestar atención al uso adecuado de los operadores de agregación.

Sol1: Tabla de Hechos separada



Sol2: En una sola Tabla de Hechos

Tabla Hechos Enorme

VENTAS

ClaveTienda	ClaveFecha	ClaveProd	Cant	Importe
1	1	1	170	85
2	1	1	300	150
3	1	1	1700	850
……	……	……	……	……

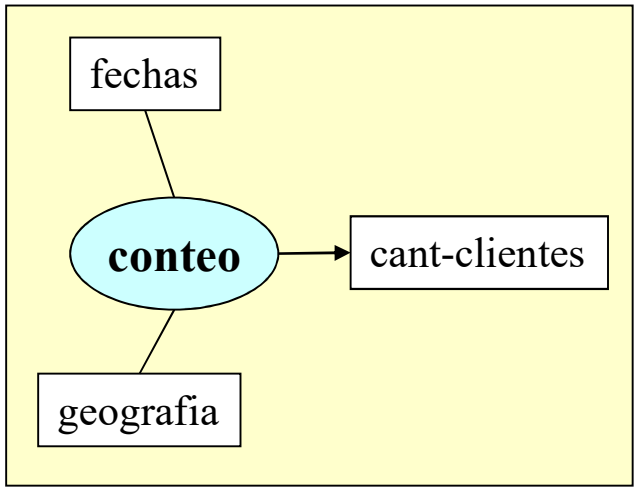
Cantidad de Prod. Vendidos en TODAS las tiendas de Bs.As.

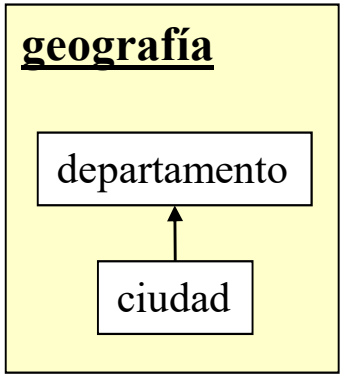
Cantidad de Productos Vendidos en la tienda T2 de Bs. As.

TIENDAS

ClaveTienda	Tienda	Ciudad	Pais	……
1	T1	Montevideo	Uru.	……
2	T2	Bs. As.	Arg.	……
3	-	Bs. As.	Arg.	……
4	-	-	Arg.	……
……	……	……	……	……

Aditividad

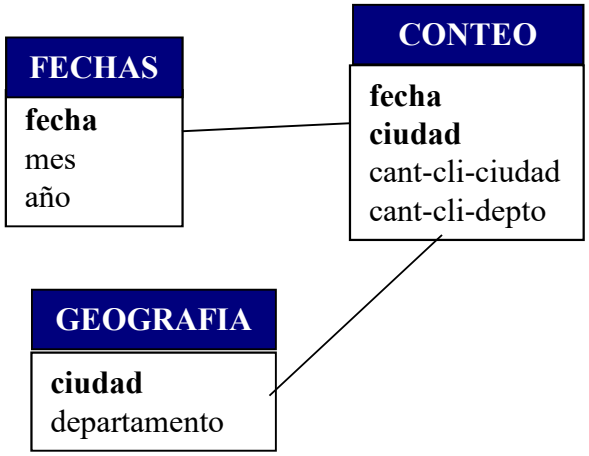




Cant-clientes no es aditiva por la dimensión Geografía.

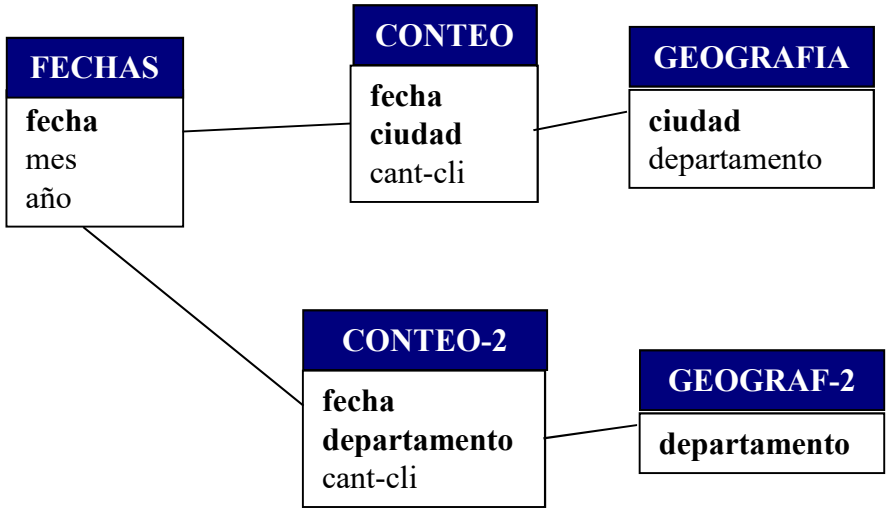
Cómo resolver el RollUp con el diseño lógico ?

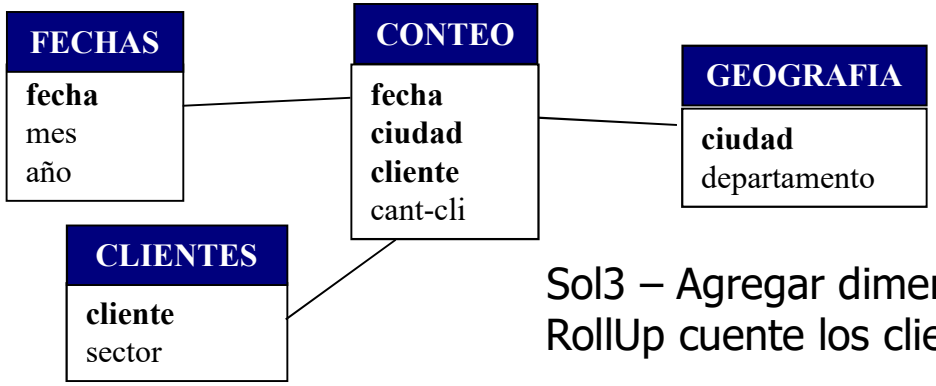
Aditividad



Sol1 – Tener 2 medidas: una para ciudad, otra para departamento

Sol2 – Tener una tabla agregación además de la primaria



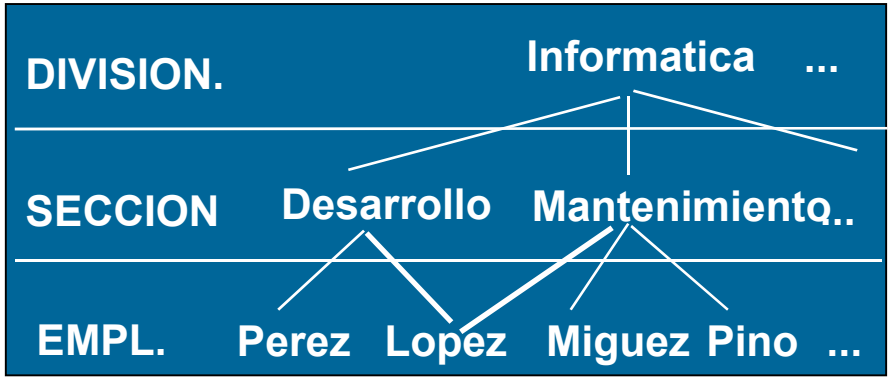


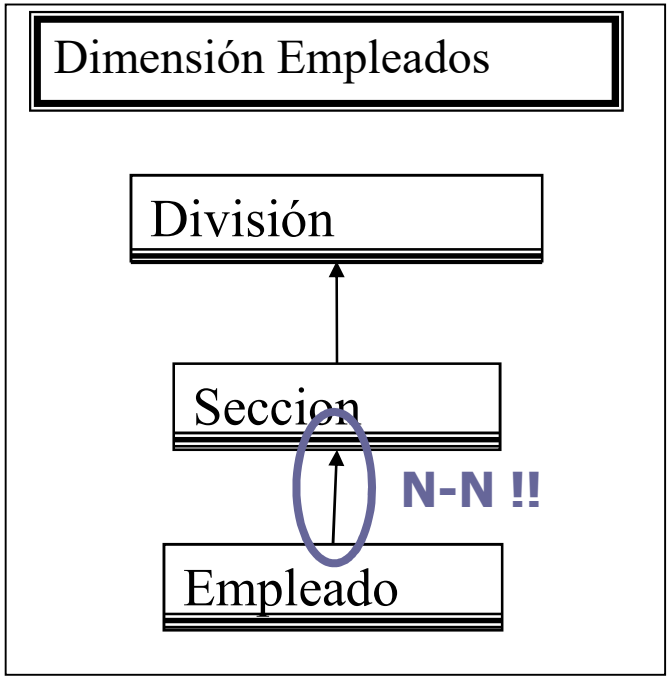
Sol3 – Agregar dimensión Clientes y que el RollUp cuente los clientes *distintos*

Jerarquías

- Jerarquías No-Estrictas

Dimensión: Empleados

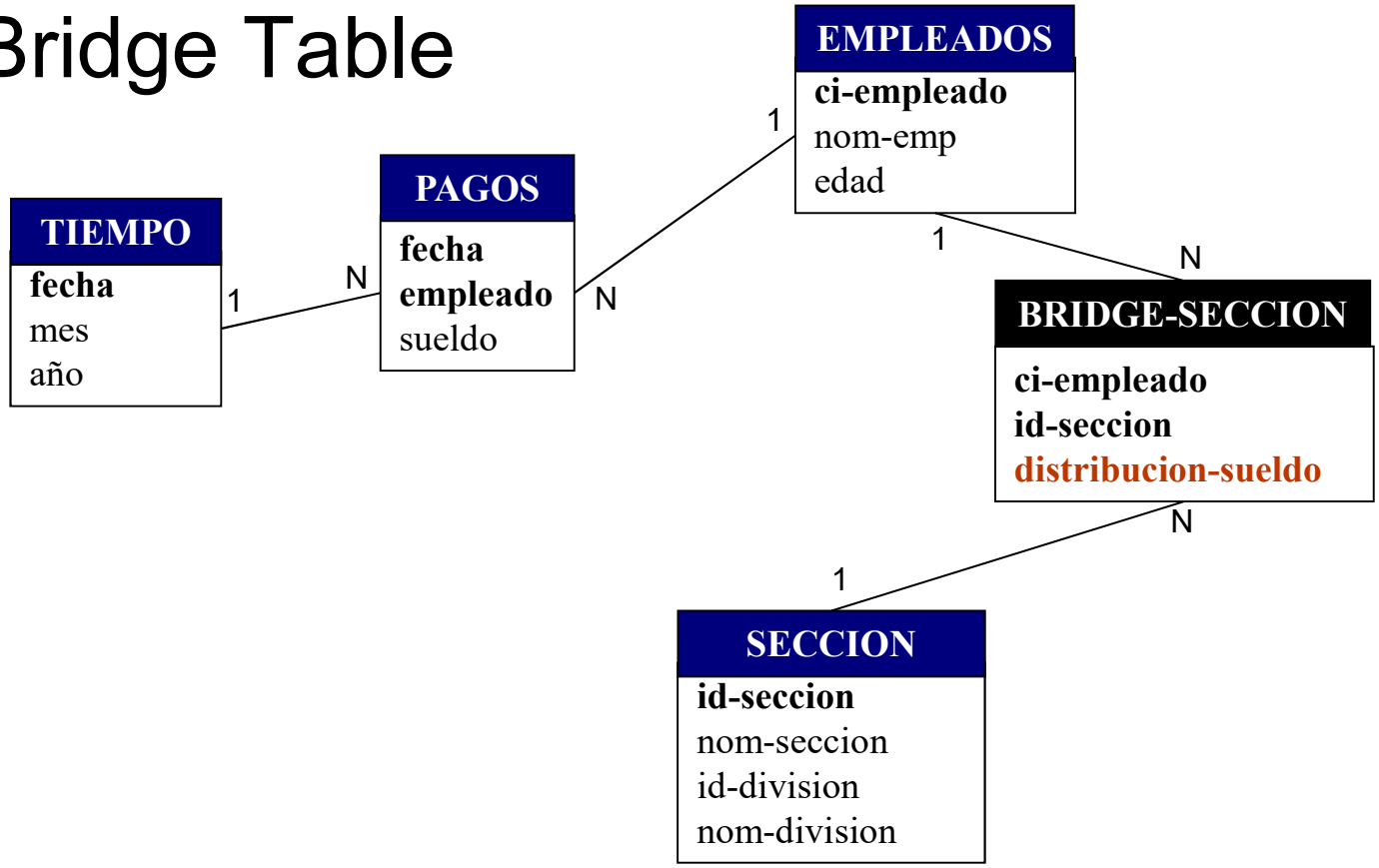




- Solución: *bridge table*

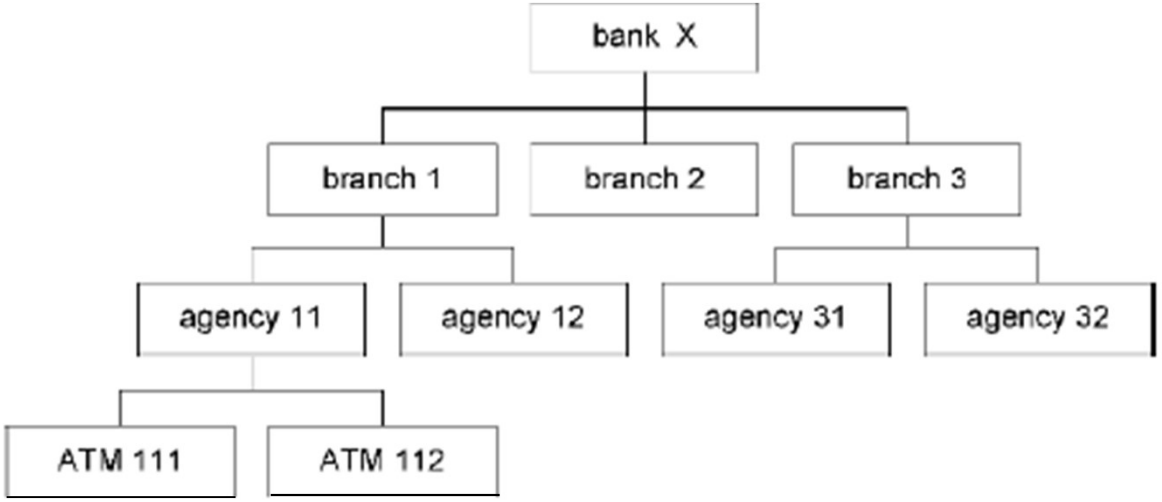
Jerarquía No-estricta

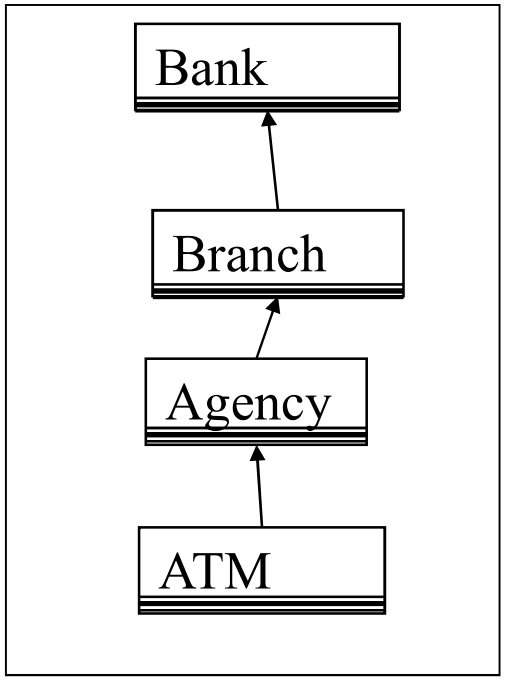
- Bridge Table



Jerarquías

- Desbalanceadas





- Problema: mantener el RollUp consistente

Jerarquía Desbalanceada

- “Rellenar” con un valor especial los que no tienen valor
  - Distintos criterios posibles para el valor:
    - uno genérico (por ej, “otro”)
    - el valor del atributo que lo precede en la jerarquía
    - el valor del atributo que lo sigue en la jerarquía

Versiones

- Objetivo:
  - Guardar la historia en dimensiones que cambian lentamente.
    - Dimensiones “casi” constantes a través del tiempo.
- Alternativas:
  - Nuevos registros:
    - Crear un registro adicional cuando hay un cambio, con los nuevos valores.
  - Nuevos campos:
    - Crear nuevos campos actuales, para guardar los nuevos valores conservando los viejos.

Versiones

- Alternativa 1: Nuevos registros
  - Tener diferentes registros para las diferentes versiones de la dimensión.
  - Manejo de la clave:
    - Generalizar la clave existente
      - Agregando dígitos de versión.
      - Agregando otros atributos.
      - Agregando atributo de tiempo.

Versiones

- Ejemplo
  - Cliente ‘Maria Perez’. Hasta 15/1/94 no era casada entonces el campo de *ec* tenia valor ‘S’. En esa fecha se caso.
  - Existen 2 descripciones correctas de Maria Perez.
  - Los registros de hechos hasta el 15/1/94 tendrán a M.P. como ‘S’ y los reg. de hechos luego de esa fecha tendrán a M.P. como ‘C’.
  - Creamos un 2o. registro de cliente para Maria Perez.
  - Generalizamos la clave de la dimensión: clave original + 2 dígitos de versión.
  - No es necesario guardar la fecha de cambio en la dimensión. Los datos de los hechos se relacionan con la versión correspondiente.
  - No se puede usar el nuevo valor con historia anterior.

Versiones

- Alternativa 2: Nuevos campos
  - Tener dos atributos, uno para el valor actual, y otro para el valor original.
  - Se puede agregar también un campo de fecha de efectividad del valor actual.
    - La única forma de partir la historia es usando la fecha de efectividad.
  - No se guardan los valores intermedios.
    - No se puede usar para describir con exactitud los cambios en un objeto.

Versiones

- El mismo ejemplo …
  - Agregamos campos *ec_actual* y *fecha_ec*.
  - Cada vez que el estado civil de ‘Maria Perez’ cambia, sobreescribimos el campo *ec_actual* y cambiamos la fecha de efectividad (*fecha_ec)*. Dejamos siempre igual el original, campo *ec*.

Minidimensiones

- Cambios en grandes dimensiones
  - Separar una o mas ***minidimensiones*** de la tabla de dimensión, c/u consistente de pequeños grupos de atributos que han sido administrados para tener una cantidad limitada de valores.
- Ganancia significativa en performance

Minidimensiones

- Ejemplo: Dimensión Cliente
  - Minidimensión demográfica
    - guardamos solo las distintas combinaciones de atributos demográficos que realmente ocurren
    - edad o ingresos se agrupan en franjas
  - Se puede hacer join directamente a la fact table o a la dimensión Cliente (puede existir la foreign key en las 2)

Ejemplo

CLIENTES

NOMBRE	EDAD	NIVEL-INGRESOS	DIRECCION	SEXO	CIUDAD	EC
R. Mendez	20	10000	Bvar. Artigas 3	F	Mont.	S
S. Nunez	30	15000	J. Herrera y Ob	M	Mont.	C
M. Garcia	20	10000	Garzon 2125	F	Salto	S
L. Lopez	50	5000	18 de Julio 643	M	Colonia	C

Ejemplo

DEMOGRAFICOS

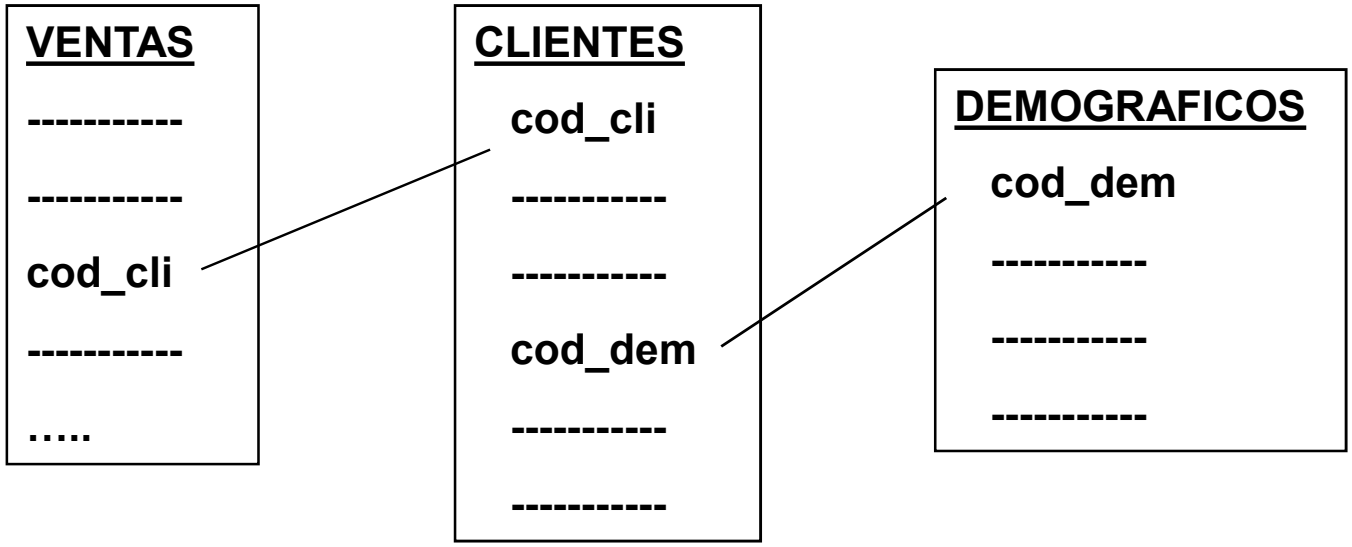
COD_DEM	EDAD	NIVEL-INGRESOS	SEXO	EC
100	20	10000	F	S
200	20	10000	M	S
300	50	5000	M	C

CLIENTES

NOMBRE	DIRECCION	CIUDAD	COD_DEM
R. Mendez	Bvar. Artiga	Mont.	100
S. Núñez	J. Herrera y	Mont.	200
M. Garcia	Garzon 2125	Salto	100
L. Lopez	18 de Julio	Colonia	300

Ejemplo: historia

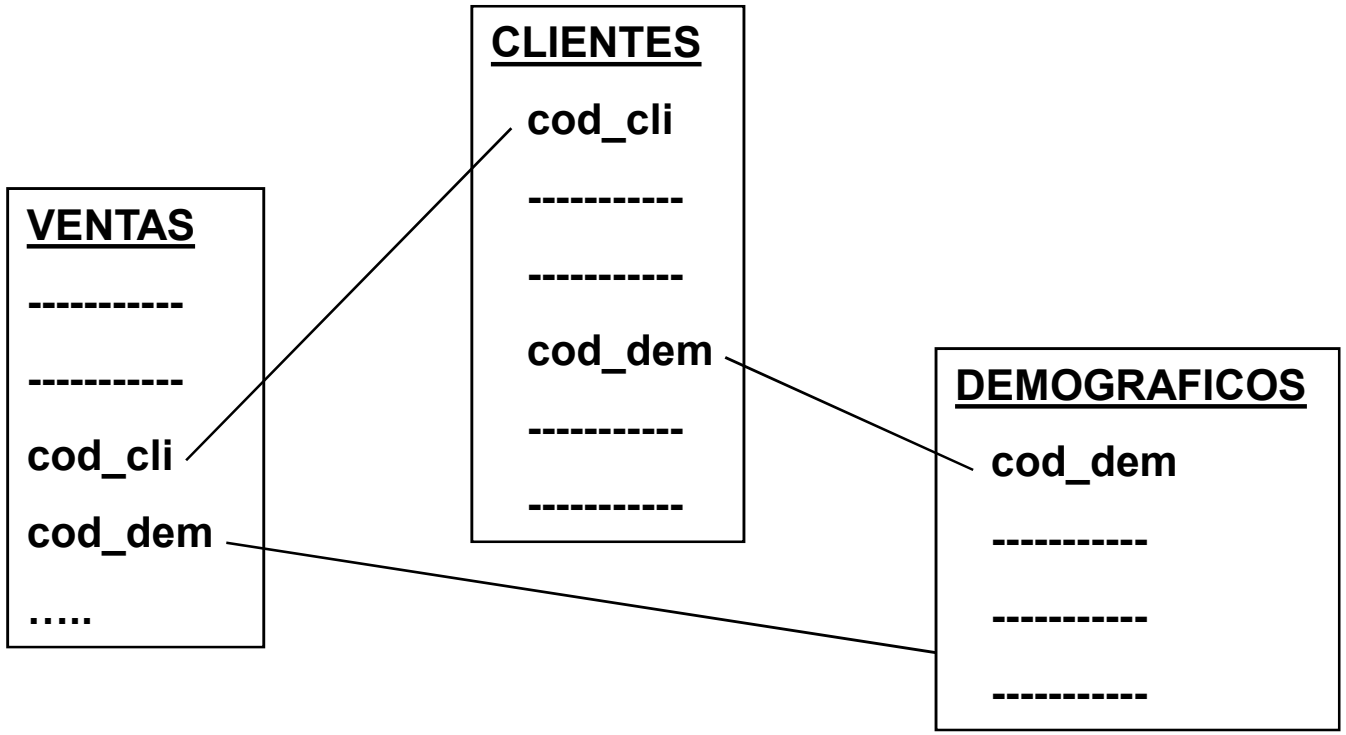
Opción A:



Si hay un cambio:
Genero otro registro en Clientes con la clave foránea correspondiente

Ejemplo: historia

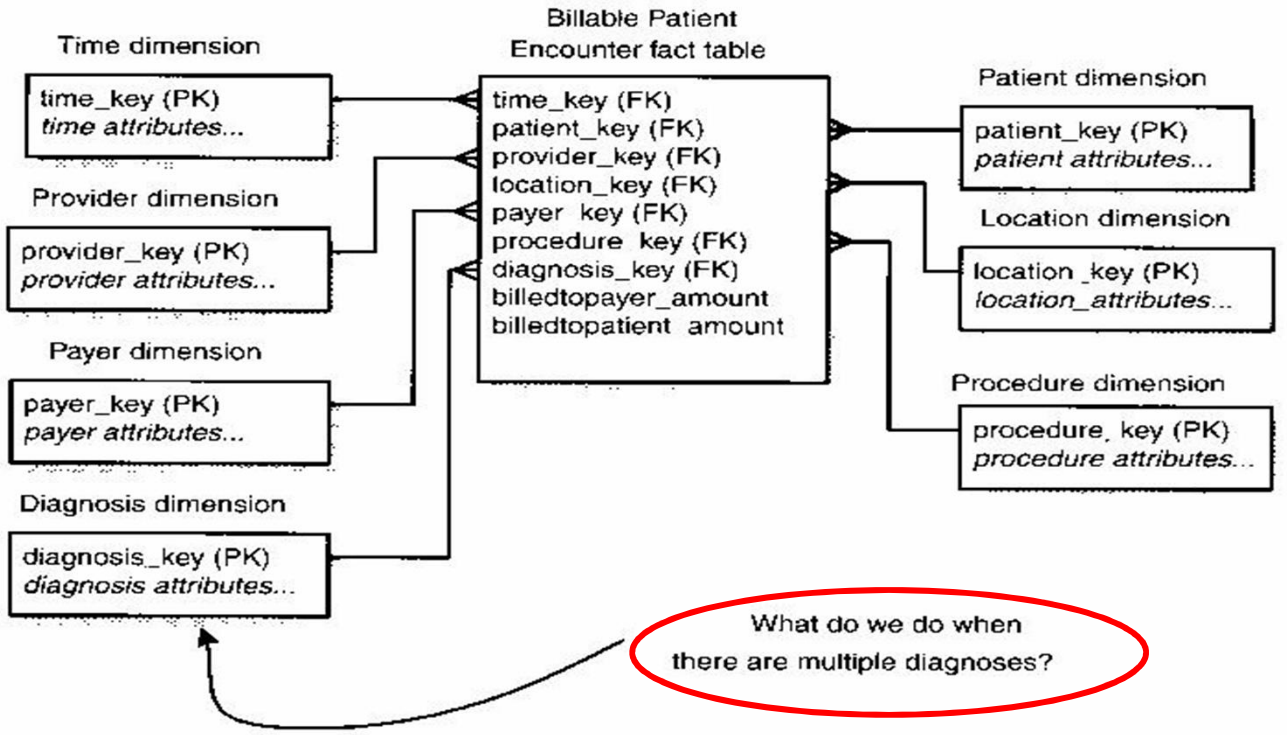
Opción B:



Si hay un cambio:
Cambio el valor de la clave foránea en Clientes.

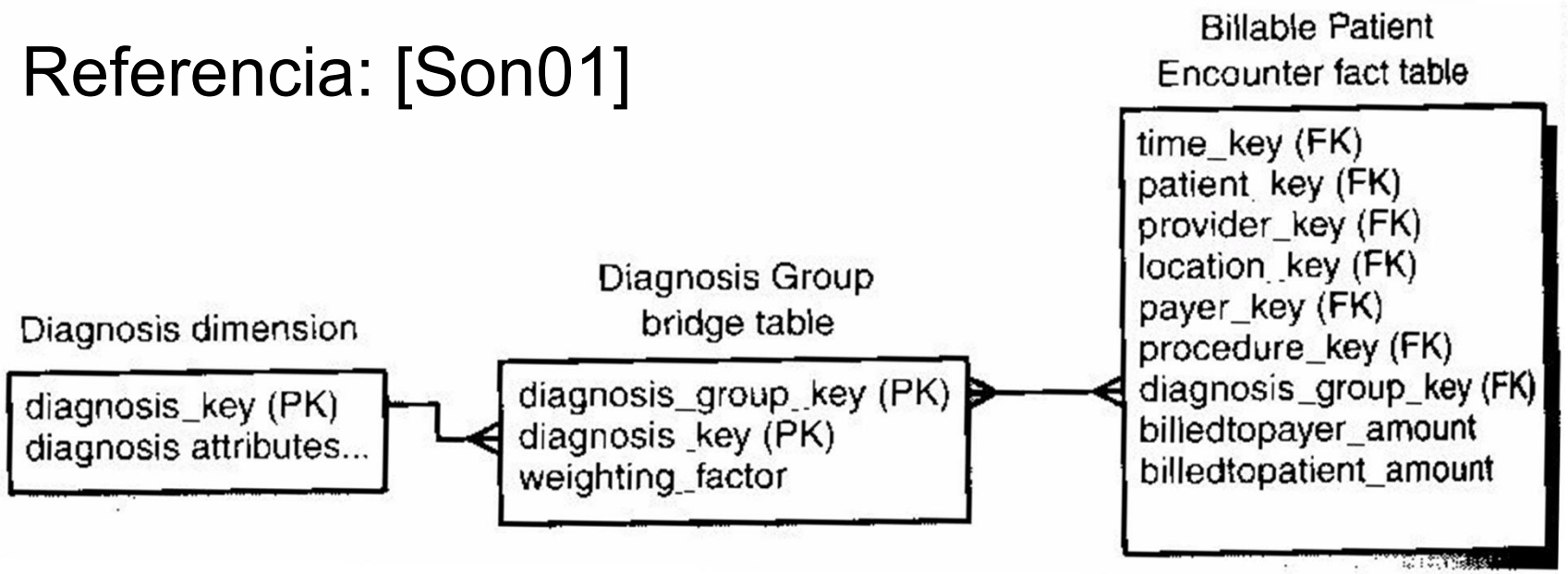
Dimensiones “Many-to-Many”

- Problema
  - En algunos casos la fact table tiene una relacion N-N con una de sus dimensiones
- Ejemplo



Dimensiones “Many-to-Many”

- Solucion
  - Agregar una “bridge table” entre la fact table y la dimension
- Referencia: [Son01]



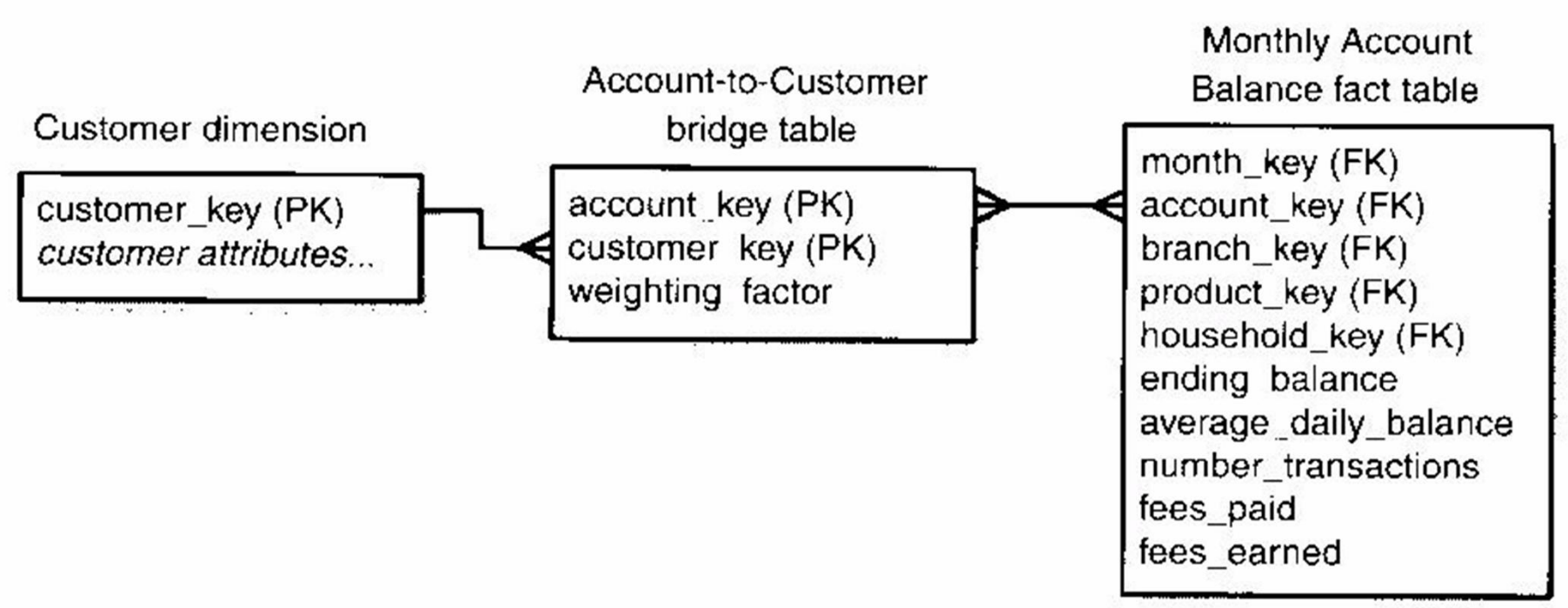
Dimensiones “Many-to-Many”

- Implica
  - Generalizar la clave en la fact table:

    diagnosis_key → diagnosis_group_key
  - Bridge table contiene:
    - Clave del grupo
    - Clave de la dimension (la original)
    - Peso (Weighting factor)
  - Como se agrupa por esa dimension?
    - Se hace join con la bridge table, y se multiplica la medida por el peso antes de sumar.

Dimensiones “Many-to-Many”

- Otro ejemplo



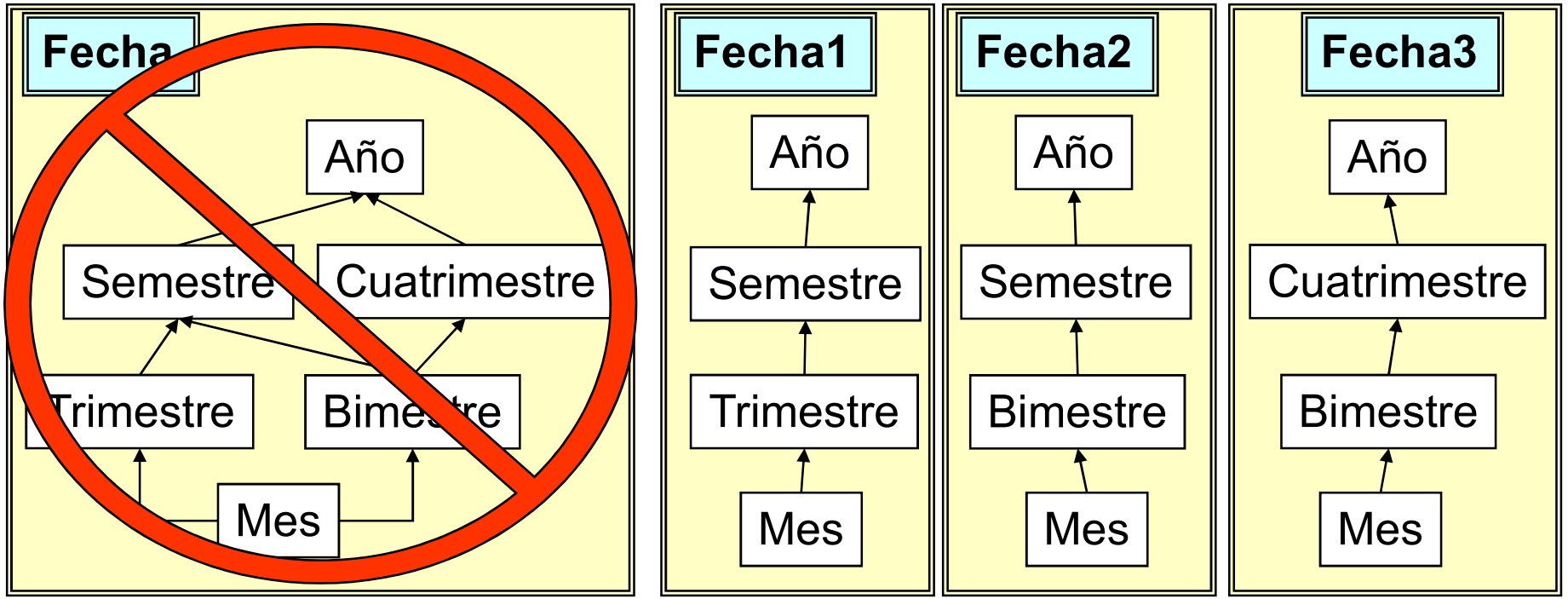
Restricciones de herramientas OLAP

Restricciones de herram. OLAP

- Principio:
  - Refinar las estructuras de datos no soportadas por el OLAP elegido.
- Ejemplos:
  - No se soportan jerarquías alternativas:
    - Definir varias dimensiones.
  - No se soportan diferentes roll-ups para medidas:
    - Definir varias medidas, una con cada roll-up.
  - No existe drill-across:
    - Tener medidas redundantes, en varios cubos.

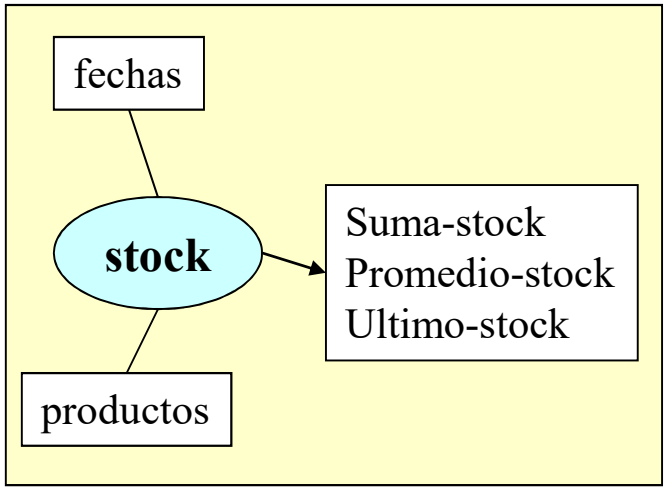
No soporta jerarquías alternativas

- Dimensiones formadas por una única jerarquía de niveles.



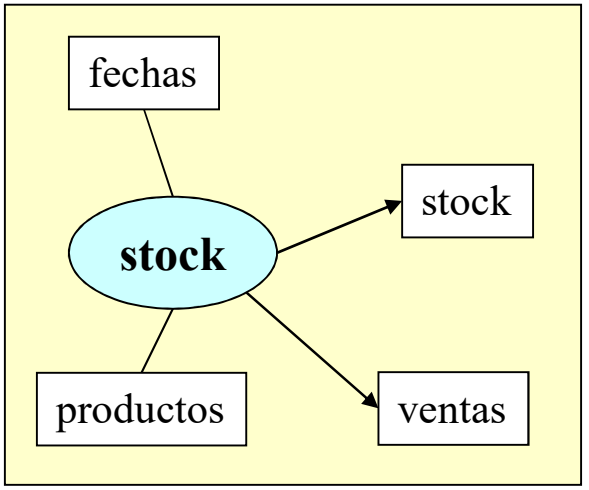
No soporta diferentes roll-up para misma medida

- Tener varias medidas una con cada roll-up.
- O tener calculadas agregaciones para diferentes roll-ups.
- Ejemplo:
  - Cantidad en stock:
    - Se quiere sumar por producto.
    - Se quiere promedio o último valor en período.



No existe Drill-accross

- A veces se fusionan varias relaciones dimensionales para evitar drill-across.
- Ejemplo:
  - Se desea comparar:
    - Cantidad en stock.
    - Cantidad vendida.



Resumen

- Vimos algunos problemas frecuentes y propuestas de diseño para resolverlos:
  - En base a ejemplos
  - Focalizando problemas específicos
- Limitaciones de los enfoques vistos:
  - Técnicas ad-hoc inducen a errores de diseño
  - No se obtienen correspondencias entre esquemas lógicos de DW y fuente
  - Pocas posibilidades de refinamiento (u optimización) del esquema lógico del DW
- Todavía no existe una teoría formal que sustente técnicas de diseño lógico relacional de DWs

Bibliografía

- **[Bal98]** Ballard, C. Herreman, D. Schau, D. Bell, R. Kim, E. Valncic, A.: “Data Modeling Techniques for Data Warehousing”. SG24-2238-00. IBM Red Book. 1998.
- **[Cab98]** Cabibbo, L. Torlone, R.:"A Logical Approach to Multidimensional Databases", EDBT, 1998.
- **[Gol98b]** Golfarelli, M. Rizzi, S.:”Methodological Framework for Data Warehouse Design", DOLAP’98, 1998.
- **[Gol00]** Golfarelli, M. Maio, D. Rizzi, S.:” Applying Vertical Fragmentation Techniques in Logical Design of Multidimensional Databases ", DAWAK’2000.
- **[Gol09]** M. Golfarelli, S. Rizzi: “Data Warehouse Design. Modern Principles and Methodologies”. McGraw Hill, 2009.
- **[Kim96]** Kimball, R.:" The Datawarehouse Toolkit ". John Wiley & Son, Inc., 1996.
- **[Kim02]** Kimball, R.:*" The Data Warehouse Toolkit "*. John Wiley & Son, Inc., 2002.

Bibliografía

- **[Mal08]** Malinowski E., Zimanyi E.: “Advanced Data Warehouse Design”. Springer. 2008.
- **[Mar00]** Marotta, M.: “Data Warehouse Design and Maintenance through Schema Transformations”. Tesis Maestría, Pedeciba, Universidad de la República, Uruguay, 2000.
- **[Moo00]** Moody, D., Kortink, M.: "From Enterprise Models to Dimensional Models: A Methodology for Data Warehouse and Data Mart Design". DMDW’00, 2000.
- **[Per01]** Peralta, V.: “Diseño lógico de Data Warehouses a partir de esquemas conceptuales multidimensionales. Tesis Maestría, Pedeciba, Universidad de la República, Uruguay, 2001.
- **[The99]** Theodoratos, D., Sellis, T.: “Designing Data Warehouses”. DKE‘99, 1999.
- **[Ham95]** Hammer, J., Garcia-Molina, H, Widom, J., Labio, W., Zhuge Y.: “The Stanford Data Warehousing Project”. IEEE Data Eng. Bull. 18(2): 41-48 (1995)