



# Diseño Lógico de Almacenes de Datos

---

Tema 4



# Indice

---

- Modelado Lógico de Almacenes de Datos
- Diseño Relacional de Almacenes de Datos
- Implementación Relacional del Modelo Conceptual
- La Dimensión Tiempo
- Representación Lógica de Jerarquías
- Aspectos Avanzados de Modelado
- Dimensiones que cambian despacio



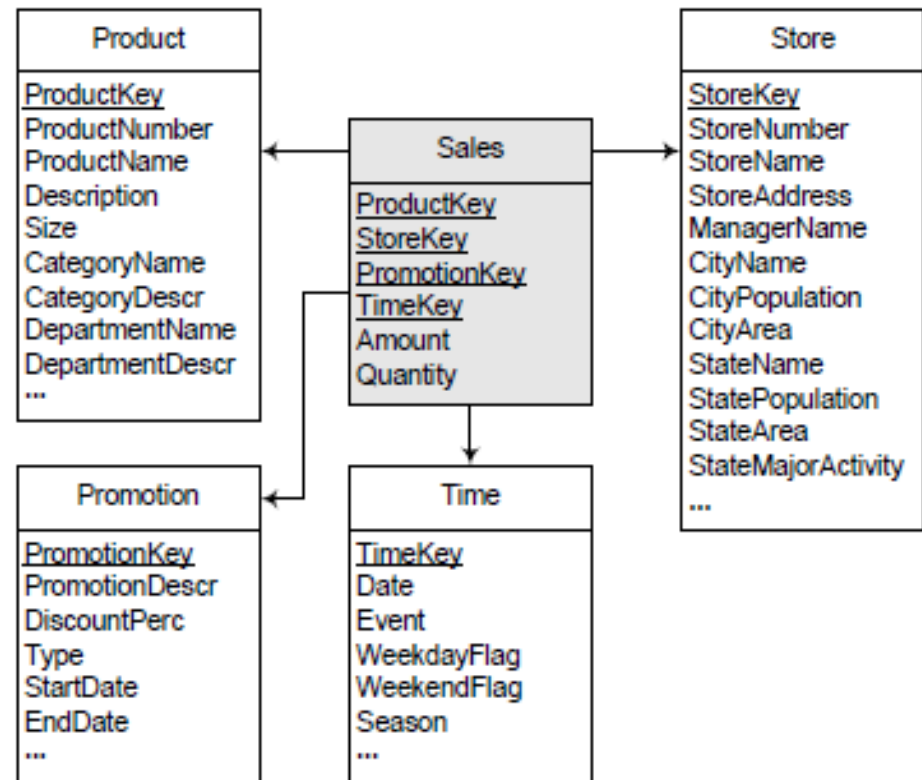
# Tecnologías OLAP

---

- Relational OLAP (ROLAP): Almacena los datos en Bases de Datos Relacionales, soporta extension de SQL y métodos de acceso especiales para implementar de manera eficiente el modelo y sus operaciones.
- Multidimensional OLAP (MOLAP): Almacena los datos en estructuras especiales (e.g., arrays) e implementa las operaciones OLAP en estas estructuras
  - Mejor rendimiento que ROLAP para consulta y agregación, pero menor capacidad de almacenaje
- Hybrid OLAP (HOLAP): Combina ambas
  - E.g., datos detallados en BDR y agregaciones en un almacén MOLAP

# Diseño de Almacenes de Datos Relacionales. Esquema estrella

- En los sistemas OLAP, las tablas se organizan en estructuras especializadas
- Esquema **Estrella** (*star scheme*): Una tabla de hechos y un conjunto de tablas de dimensiones
  - Restricciones de Integridad referencial entre la tabla de hechos y las de dimensiones
  - Las tablas de dimensiones pueden presentar redundancias por las jerarquías
  - Tablas de dimensiones denormalizadas. Tablas de hechos normalizadas

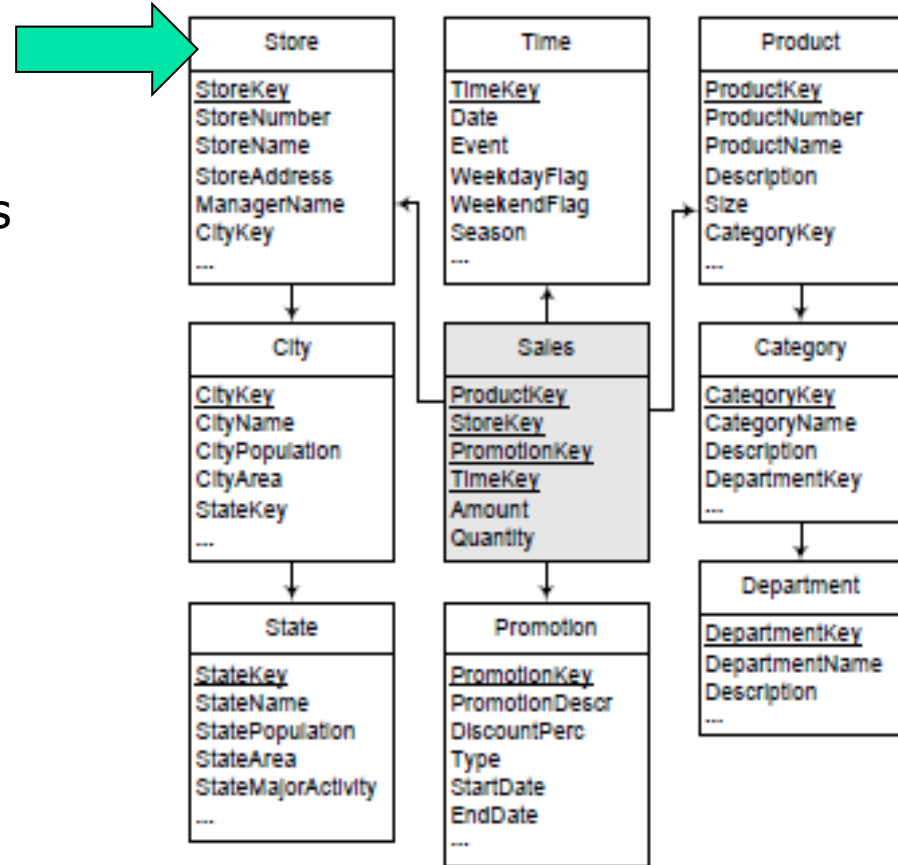


# Diseño de Almacenes de Datos Relacionales. Esquema Copo de nieve

- Esquema de **Copo de Nieve** (*snowflake scheme*): Evita la redundancia del esquema en estrella normalizando las tablas de dimensiones

- Tablas Normalizadas: Optimizan espacio, peor rendimiento

- Esquema **Copo de Estrella** (*starflake scheme*): Combinación de las anteriores. Algunas dimensiones normalizadas, otras no



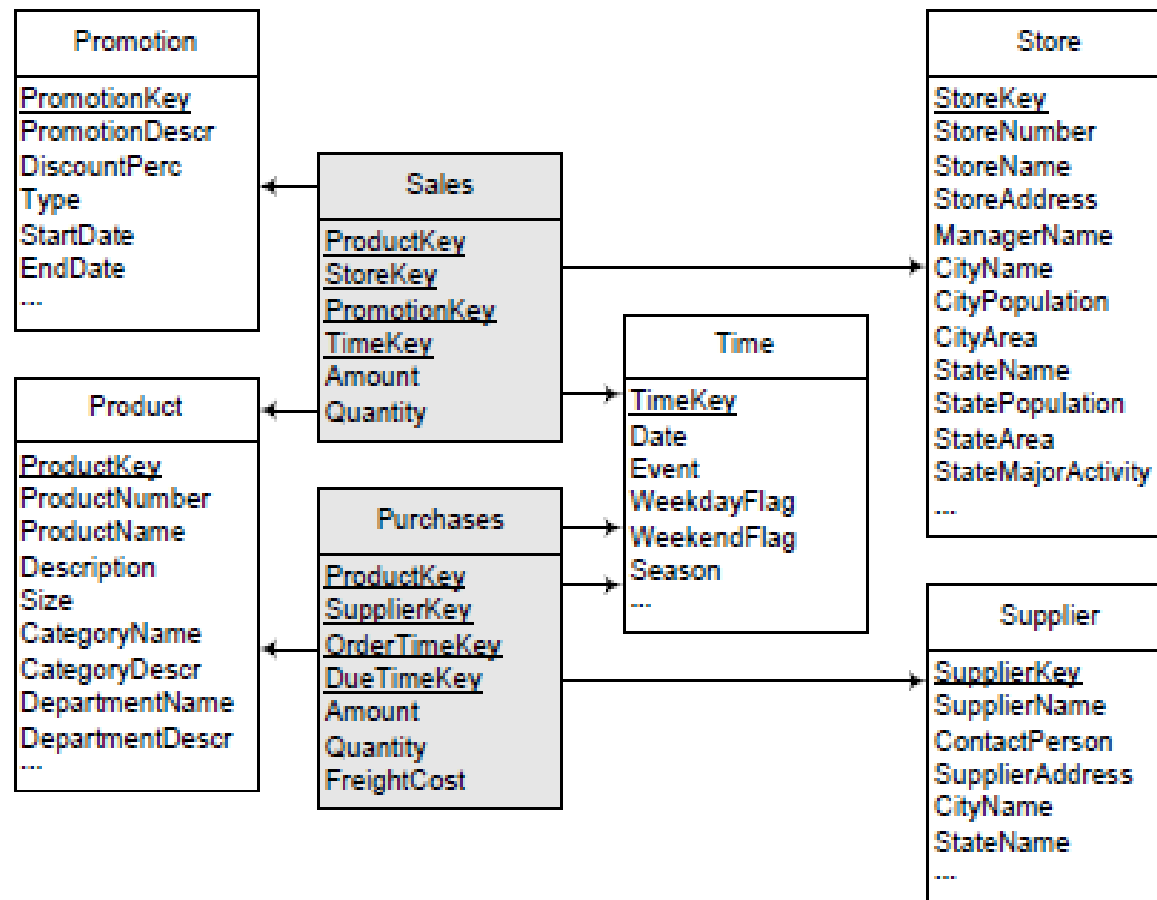
Ejemplo. Consulta SQL que obtenga el "Total de ventas por categoría"

Esquema estrella?

Esquema copo de nieve?

# Diseño de Almacenes de Datos Relacionales. Esquema constelación

- Esquema **Constelación** (*constellation scheme*): Múltiples tablas de hechos que comparten tablas de dimensiones





# Implementación Relacional del Modelo Conceptual

## ■ Regla 1. Niveles

- Por cada Nivel L (que no tenga una relación 1:1 con la tabla de hechos) se crea una Tabla TL
- TL contiene todos los atributos del Nivel
- Se puede añadir una clave sustituta (simple numérica)
- Si no, el identificador del nivel es la clave.
- Se pueden añadir otros atributos si existen relaciones (ver más adelante)

La aplicación de todas las reglas dará lugar a un diseño en  
**copo de nieve**



# Implementación Relacional del Modelo Conceptual

---

- Regla 2. Hechos
  - Un Hecho F se implementa en una Tabla TF
  - TF contiene como atributos las medidas del Hecho
  - Se puede añadir una clave sustituta (simple numérica)
  - Las claves de todas las relaciones que se añadan forman juntas una clave





# Implementación Relacional del Modelo Conceptual

---

- Regla 3a. Relaciones 1:1
  - Una relación 1:1 entre un hecho F y un nivel L se implementa añadiendo a TF los atributos de L
  - Una relación 1:1 entre un nivel hijo LC y uno padre LP se implementa añadiendo a la tabla TC los atributos de LP



# Implementación Relacional del Modelo Conceptual

---

- Regla 3b. Relaciones 1:M
  - Una relación 1:M entre un hecho F y un nivel L se implementa añadiendo en TF un atributo del tipo de la clave de TL y creando una Foreign Key hacia TL
  - Una relación 1:M entre un nivel LP (parent) y otro LC (child) se implementa añadiendo en TC un atributo del tipo de la clave de TP y creando una Foreign Key hacia TP



# Implementación Relacional del Modelo Conceptual

---

- Regla 3c. Relaciones M:M
  - Una relación M:M se implementa creando una tabla puente TB
  - TB contiene como atributos las claves de las tablas que representan los niveles (o hechos si es una relación entre hechos)
  - La clave primaria de TB es la combinación de todas las claves
  - Si la relación tiene un atributos adicionales, se añaden a la tabla



# La Dimensión Tiempo

- Almacén de Datos: Base de Datos histórica
- La dimensión tiempo está presente en casi todos los almacenes de datos
- En un esquema en estrella o copo de nieve el tiempo se incluye como una clave foránea en la tabla de hechos y como una dimensión Tiempo conteniendo los niveles agregados.
- Bases de datos OLTP: la información temporal se deriva generalmente de atributos de tipo DATE
  - Ejemplo: Un fin de semana se computa sobre la marcha utilizando las funciones adecuadas
- En un Almacén de Datos la información del tiempo se almacena de forma explícita en la dimension tiempo
  - Fácil de computar: Total de ventas durante el fin de semana  
`SELECT SUM(SalesAmount) FROM Time T, Sales S  
WHERE T.TimeKey = S.TimeKey AND T.WeekendFlag`
- La granularidad de la dimension tiempo depende de su uso
  - Granularidad mensual, 5 años = 60 filas
- Puede tener más de una jerarquía

¿Granularidad por segundo?



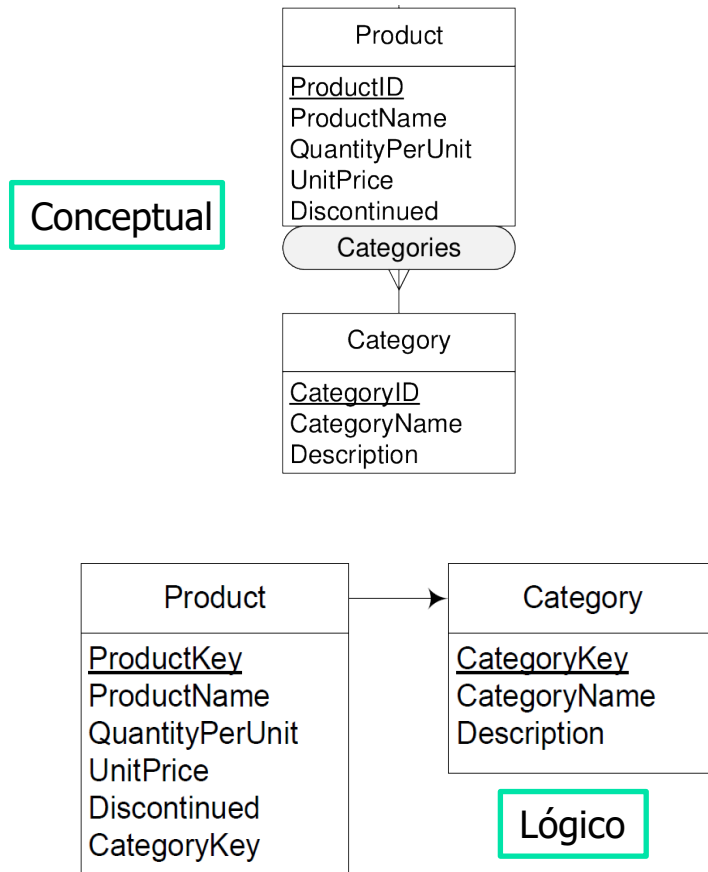
# Jerarquías

---

- Jerarquías equilibradas:
  - Se modelan mediante relaciones 1:N
  - Nos conducen a esquemas copo de nieve
  - Si es necesario un esquema en estrella, se pueden modelar en una sola tabla con la clave y los atributos de todos los niveles. Por ejemplo: dimensión tiempo.

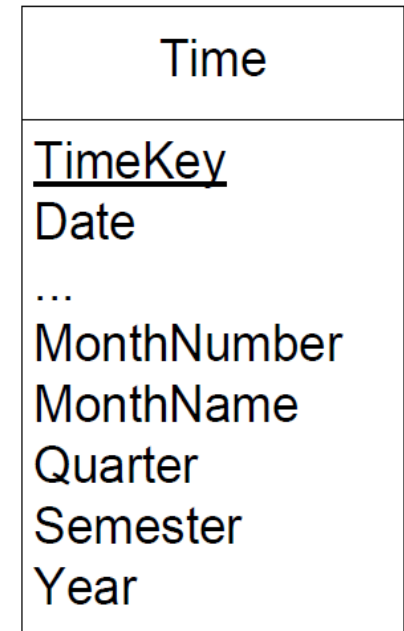
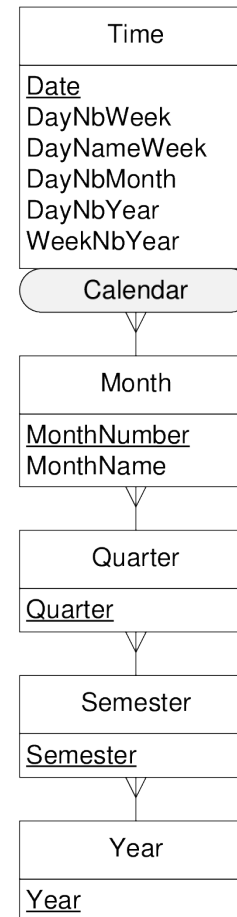
# Relaciones en jerarquía equilibrada

## ■ Copo de nieve



## Tabla plana

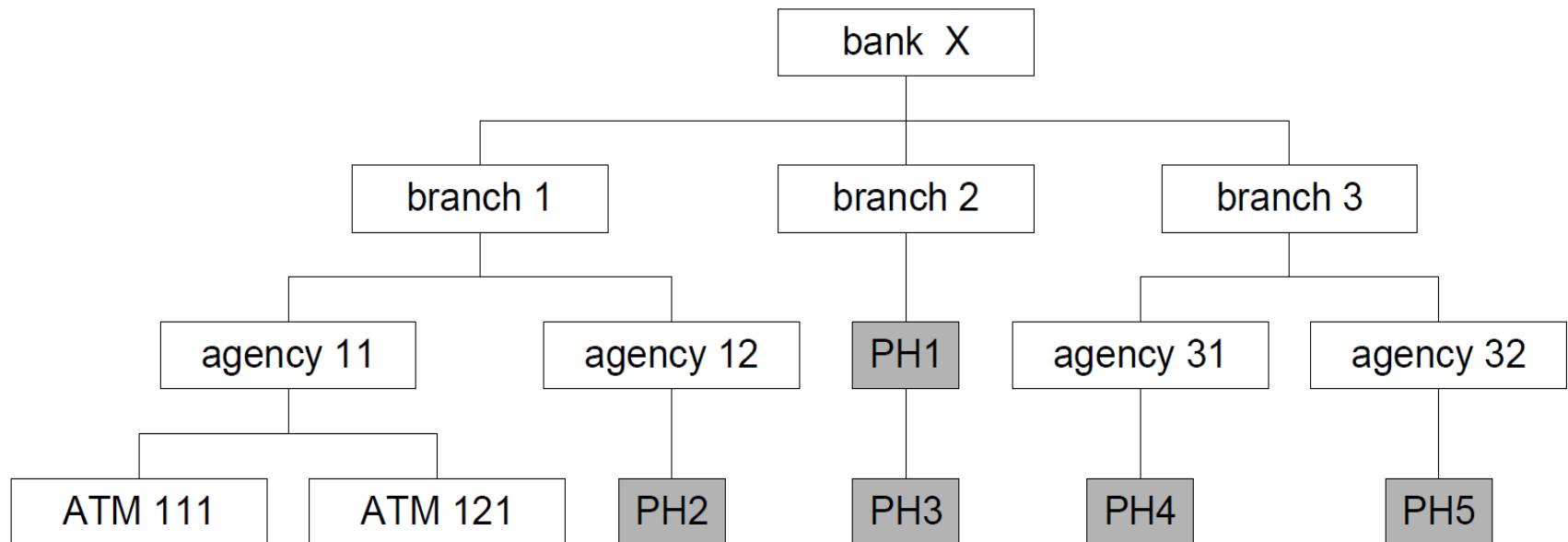
**Conceptual**



**Lógico**

# Jerarquías

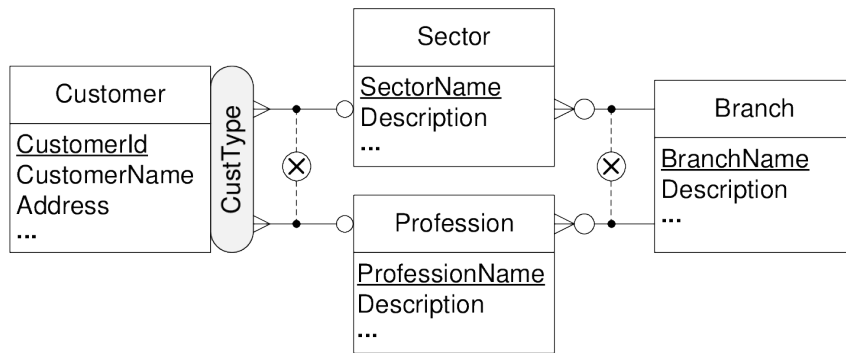
- Jerarquías no equilibradas:
  - Se pueden insertar valores ficticios en lugar de los valores nulos para transformarla en una equilibrada. Estos valores tienen que ser tratados correctamente (p.e. en agregaciones)



# Jerarquías

## ■ Jerarquías Generalizadas:

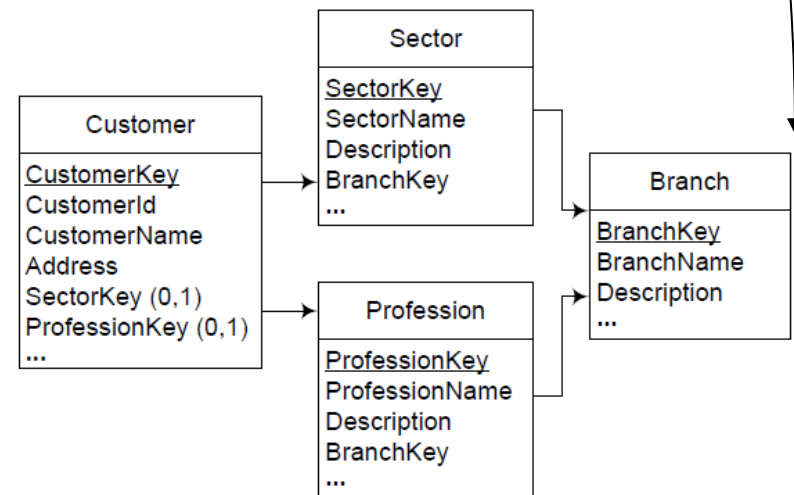
- Los miembros de una dimensión son de tipos distintos
- En el ejemplo los clientes pueden ser Compañías (sector) o personas (profesión)



Conceptual

## ■ Posibilidades:

- Todo en una tabla (con valores nulos donde aplique)
- **Una tabla por nivel**
- Una tabla para los niveles comunes y otra para los específicos



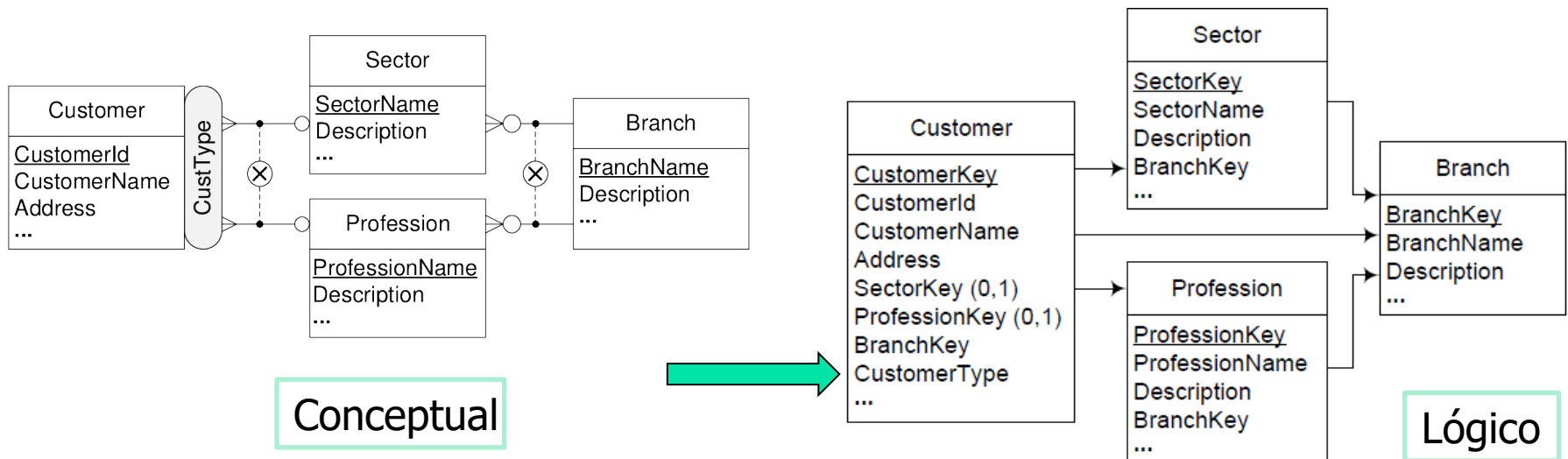
Lógico



# Jerarquías

## ■ Jerarquías Generalizadas:

- Se debe añadir un atributo que es una clave foránea del siguiente nivel común (si existe)
- Se puede añadir un atributo discriminante para indicar el camino de agregación





# Jerarquías

---

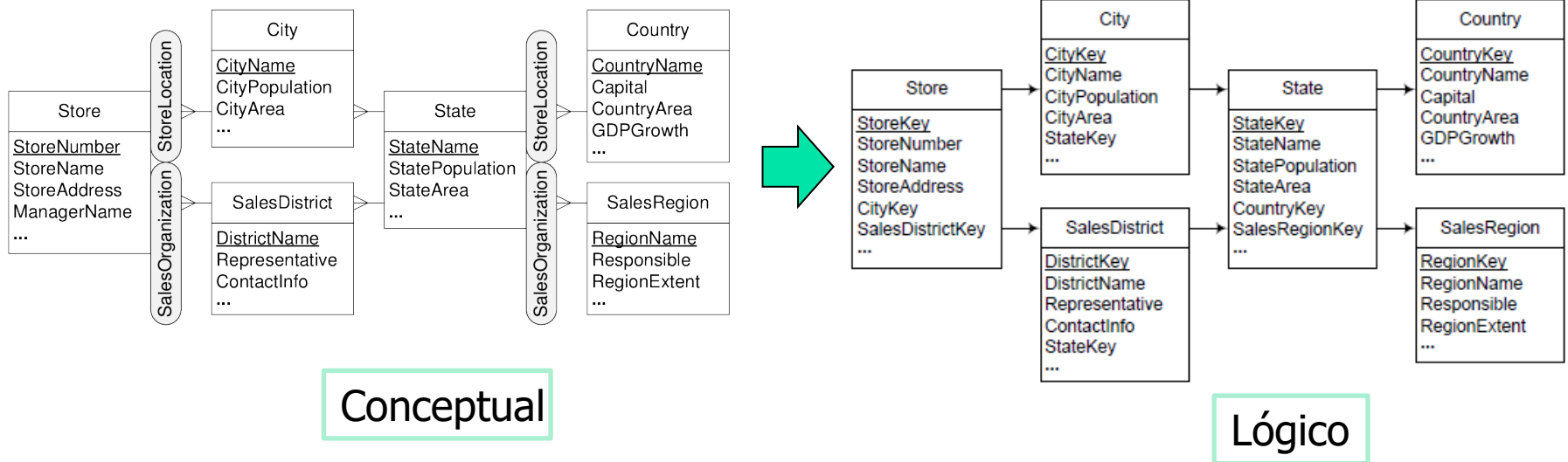
## ■ Jerarquías Generalizadas:

- Se pueden añadir restricciones para asegurarnos de que solo una de las FK para los niveles especializados tengan valor
- `ALTER TABLE Customer ADD CONSTRAINT CustomerTypeCK  
CHECK ( CustomerType IN ('Person', 'Company') )`
- `ALTER TABLE Customer ADD CONSTRAINT CustomerPersonFK  
CHECK ( (CustomerType != 'Person') OR ( ProfessionKey IS NOT NULL AND SectorKey  
IS NULL ) )`
- `ALTER TABLE Customer ADD CONSTRAINT CustomerCompanyFK  
CHECK ( (CustomerType != 'Company') OR ( ProfessionKey IS NULL AND SectorKey IS  
NOT NULL ) )`
- Recordad:  $p \rightarrow q$  es equivalente a:  $\text{NOT } p \text{ OR } q$

# Jerarquías

## ■ Jerarquías Paralelas

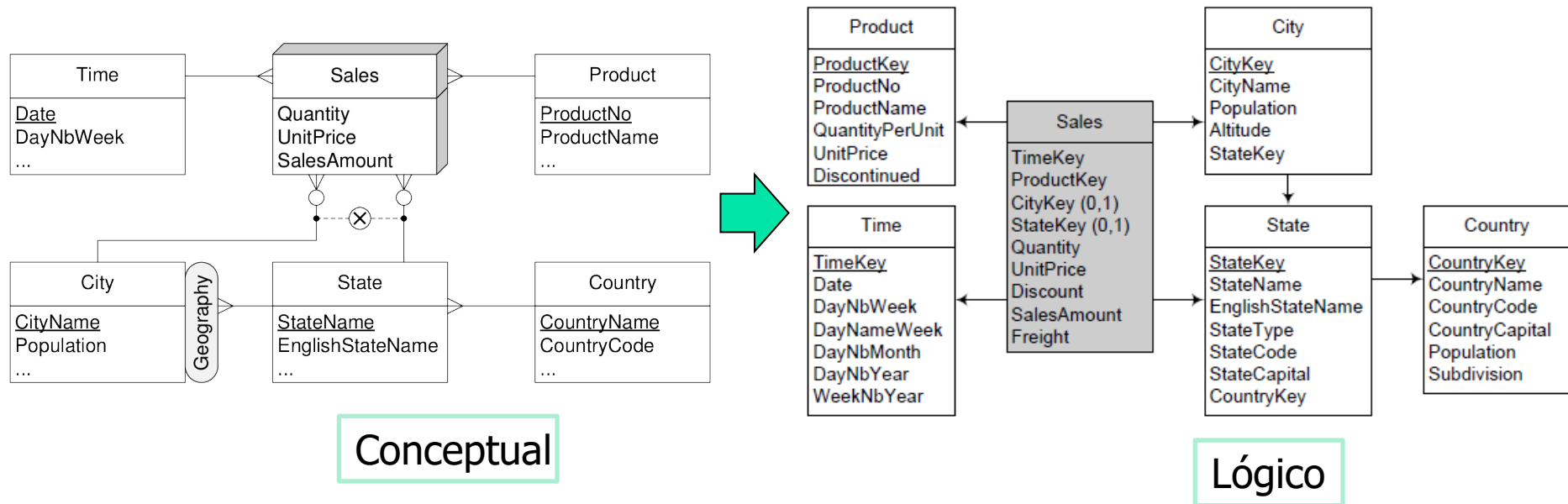
- Compuesta por varias jerarquías. El mapeo lógico combina el mapeo de cada tipo



- Niveles compartidos en una sola tabla (ej. State)

# Aspectos Avanzados de Modelado

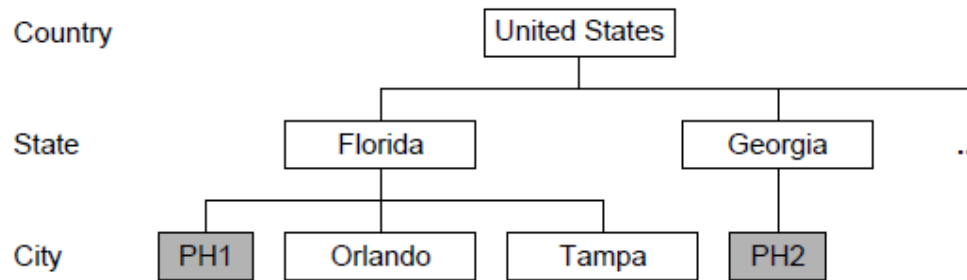
- Hechos con Granularidades Múltiples
- Primera solución: Usar FKs múltiples, una por cada granularidad alternativa



- CityKey y StateKey son opcionales, pero solo una debe tener un valor

# Aspectos Avanzados de Modelado

- Hechos con Granularidades Múltiples
- Segunda solución: Quitar la variación de granularidad usando marcadores de posición (valores ficticios)

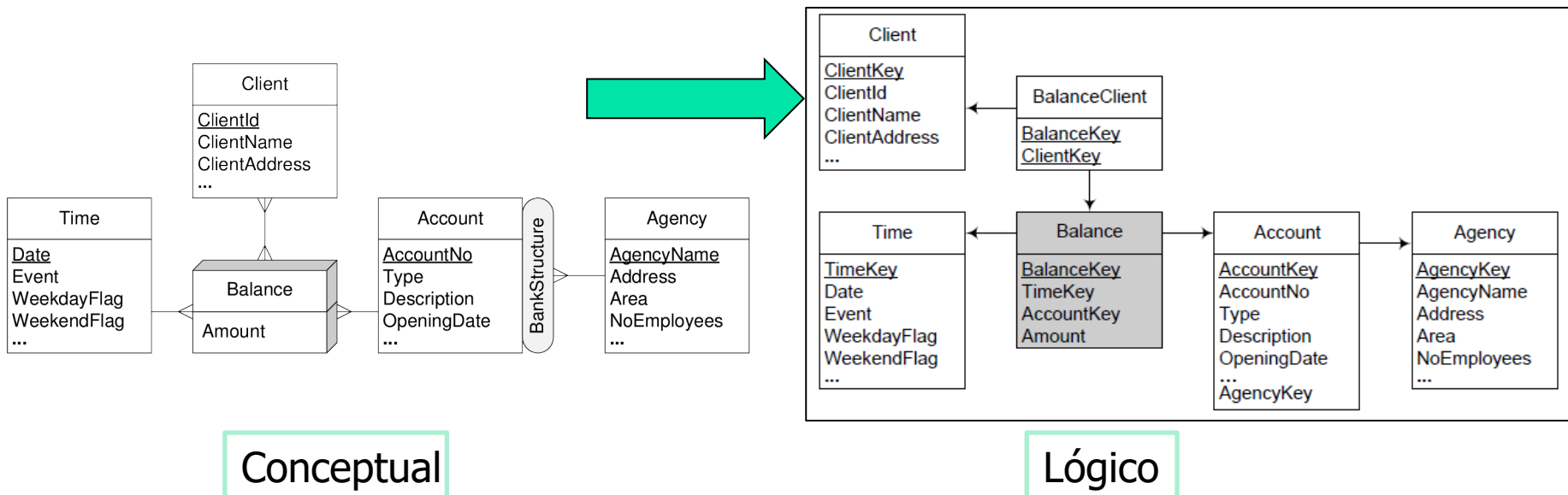


- Los marcadores se usan para hechos que se refieren a niveles no hoja
  - Un miembro de hecho apunta a un miembro no hoja que tiene hijos (PH1 representa a todas las ciudades distintas a las existentes)
  - Un miembro de hecho apunta a un miembro no hoja sin hijos (PH2 representa todas las ciudades del estado)

# Aspectos Avanzados de Modelado

## ■ Dimensiones M:M

- Las normas de correlación crean relaciones que representan el hecho, los niveles de dimensión, y una tabla puente que representa la relación de muchos a muchos entre la tabla de hechos y dimensiones
- La tabla puente BalanceClient relaciona la tabla de hechos Balance con la dimensión Cliente
- Una clave sustituta añadida a la tabla de hechos Balance relaciona los hechos con los clientes.

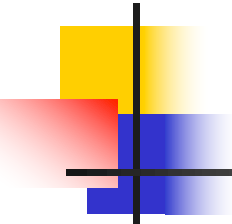




# Ejercicio 1

---

- Dado el modelo conceptual del almacén del proveedor de teléfonos realizado en el tema anterior, se pide realizar un modelado lógico proporcionando:
  - Un esquema estrella.
  - Un esquema copo de nieve.



# Dimensiones que cambian espacio

---

- Las instancias de dimensiones se pueden considerar fijas a lo largo del tiempo (ej. dimensión Tiempo)
- En muchas situaciones del mundo real, las dimensiones pueden cambiar a nivel estructural y de instancia
  - Ejemplo: a nivel estructural, cuando un atributo se elimina de las fuentes de datos y ya no está disponible también debe eliminarse de la tabla de dimensiones
  - A nivel de instancia, dos tipos de cambios
    - Una corrección se debe hacer en las tablas de dimensiones debido a un error, los nuevos datos deben sustituir al viejo
    - Cuando las condiciones del contexto de un escenario de análisis cambian, el contenido de las tablas de dimensión deben cambiar en consecuencia



# Dimensiones que cambian espacio

- Ejemplo: La tabla de hechos Sales relacionada con las dimensiones Time, Employee, Customer y Product, y una medida SalesAmount

TimeKey	EmployeeKey	CustomerKey	ProductKey	SalesAmount
t1	e1	c1	p1	100
t2	e2	c2	p1	100
t3	e1	c3	p3	100
t4	e2	c4	p4	100

ProductKey	ProductName	Discontinued	CategoryName	Description
p1	prod1	No	cat1	desc1
p2	prod2	No	cat1	desc1
p3	prod3	No	cat2	desc2
p4	prod4	No	cat2	desc2

- Entran nuevas filas en la table de hechos Sales según se producen ventas
- Otras actualizaciones que suelen ocurrir:
  - Un product empieza a ser comercializado → Nueva fila en Product
  - Un dato sobre el product es erróneo y se debe modificar
  - La categoría de un product necesita ser cambiado
- Estas dimensiones se llaman *slowly changing dimensions*

Consulta: Ventas totales por empleado y categoría de producto

# Dimensiones que cambian espacio

- Consulta: Ventas totales por empleado y categoría de producto

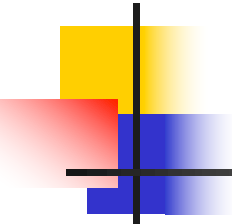
```
SELECT E.EmployeeKey, P.CategoryName, SUM(SalesAmount)
FROM Sales S, Product P
WHERE S.ProductKey = P.ProductKey
GROUP BY E.EmployeeKey, P.CategoryName
```

EmployeeKey	CategoryName	SalesAmount
e1	cat1	100
e2	cat1	100
e1	cat2	100
e2	cat2	100

- En un instante t después de t4 la categoría del producto p1 cambia a cat2
- Si solo sobreescribimos la categoría la misma consulta devuelve:

EmployeeKey	CategoryKey	SalesAmount
e1	cat2	200
e2	cat2	200

- Resultado Incorrecto: Los productos afectados por el cambio de categoría ya estaban asociados a los datos de venta
- Si la nueva categoría es el resultado de una corrección de error (la categoría real de p1 es cat2), este resultado podría ser correcto
- Siete tipos de dimensiones que cambian espacio



# Dimensiones que cambian espacio

---

## ■ Tipo 1

- El más simple, consiste en sobrescribir el antiguo valor del atributo con el nuevo
- Asume que la modificación se debe a un error en los datos de dimensión
- Simplemente escribir esto en SQL:

```
UPDATE Product
```

```
SET CategoryName = cat2
```

```
WHERE ProductName = p1
```

# Dimensiones que cambian espacio

## ■ Tipo 2

- Las tuplas de la tabla de dimensiones son versionadas: se inserta una nueva tupla cada vez que se produce un cambio
- Las tuplas de la tabla de hechos coinciden con la tupla de la tabla de dimensión correspondiente a la versión correcta
- Ejemplo: Product se amplía con dos atributos From y To (el intervalo de validez de la tupla)
  - Una fila de p1 se inserta en Product, con su nueva categoría cat2
  - Sales antes de t contribuirá a la agregación de cat1, los ocurridos después de t contribuirá a cat2

Product Key	Product Name	Discontinued	Category Name	Description	From	To
p1	prod1	No	cat1	desc1	2010-01-01	2011-12-31
<b>p11</b>	prod1	No	<b>cat2</b>	desc2	2012-01-01	Now
p2	prod2	No	cat1	desc1	2012-01-01	Now
p3	prod3	No	cat2	desc2	2012-01-01	Now
p4	prod4	No	cat2	desc2	2012-01-01	Now

- Now indica que la tupla sigue siendo válida
- Un producto participa en la tabla de hechos con tantos sustitutos como cambios haya

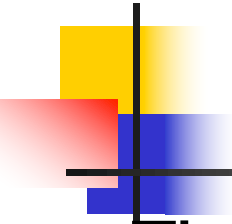
# Dimensiones que cambian espacio

## ■ Tipo 3

- Añadimos una columna por cada atributo susceptible de cambio, que tendrá el Nuevo valor del atributo
- Ejemplo, CategoryName y Description cambiaron → cuando p1 cambia su categoría de cat1 a cat2; la descripción asociada también cambia de desc1 a desc2

Product Key	Product Name	Discontinued	Category Name	NewCateg	Description	NewDesc
p1	prod1	No	cat1	cat2	desc1	desc2
p2	prod2	No	cat1	Null	desc1	Null
p3	prod3	No	cat2	Null	desc2	Null
p4	prod4	No	cat2	Null	desc2	Null

- Solo las 2 versiones más recientes del atributo pueden ser representadas en esta solución, y el intervalo de validez de las filas no se almacena



# Dimensiones que cambian despacio

## ■ Tipo 4

- Tiene como objetivo manejar grandes tablas de dimensiones y atributos que cambian con frecuencia
- Una mini dimensión, se crea para almacenar los atributos que cambian más frecuentemente
  - Ejemplo: En la dimension Product los atributos SalesRanking y PriceRange cambian frecuentemente
  - Creamos una nueva dimension llamada ProductFeatures, con una clave ProductFeaturesKey y atributos SalesRanking y PriceRange

Product FeaturesKey	Sales Ranking	Price Range
pf1	1	1-100
pf2	2	1-100
...	...	...
pf200	7	500-600

- Una fila en la minidimensión por cada combinación de SalesRanking y PriceRange encontrados en los datos
- La clave ProductFeaturesKey se añade a la table de hechos como clave foránea

# Dimensiones que cambian espacio

## ■ Tipo 5

- Es una extensión del tipo 4, donde la tabla de la dimensión primaria se extiende con una clave foránea a la tabla de la minidimensión
- Dimensión Product

Product Key	Product Name	Discontinued	CurrentProduct FeaturesKey
p1	prod1	No	pf1
...	...	...	...

Nos permite analizar los valores actuales de una dimensión **sin acceder** a la tabla de hechos

- La FK es un atributo de Tipo 1: cuando cualquier característica del producto cambia, el valor actual de ProductFeaturesKey se almacena en la tabla Product
- La tabla de hechos incluye las claves externas ProductKey y ProductFeaturesKey
- CurrentProductFeaturesKey en la dimensión Product permite el roll up de hechos históricos basados en el perfil actual del producto

# Dimensiones que cambian espacio

## ■ Tipo 6

- Es una extensión del tipo 2, con una columna adicional que contiene el valor actual de un atributo
- Ejemplo: Product se extiende con From y To
- CurrentCategoryKey contiene el valor actual del atributo Category

Product Key	Product Name	Discontinued	Category Key	From	To	Current CategoryKey
p1	prod1	No	c1	2010-01-01	2011-12-31	c11
p11	prod1	No	c11	2012-01-01	9999-12-31	c11
p2	prod2	No	c1	2010-01-01	9999-12-31	c1
p3	prod3	No	c2	2010-01-01	9999-12-31	c2
p4	prod4	No	c2	2011-01-01	9999-12-31	c2

- CategoryKey se usa para agrupar hechos basados en la categoría del producto cuando el hecho ocurrió
- CurrentCategoryKey agrupa los hechos basados en la categoría actual



# Dimensiones que cambian espacio

## ■ Tipo 7

- Similar al tipo 6 cuando hay muchos atributos en la tabla de dimensión
- Añade una FK de la tabla de dimensión (natural no sustituta). ProductName en el ejemplo si es durable.
- Ejemplo: Product igual que en el tipo 2, pero la tabla de hechos sería así:

TimeKey	EmployeeKey	CustomerKey	ProductKey	Product Name	SalesAmount
t1	e1	c1	p1	prod1	100
t2	e2	c2	p11	prod1	100
t3	e1	c3	p3	prod3	100
t4	e2	c4	p4	prod4	100

- ProductKey: Se puede usar para análisis histórico basado en los valores efectivos del producto cuando el hecho ocurrió
- Para soportar el análisis actual, se necesita una vista adicional, llamada CurrentProduct: Mantiene solo los valores actuales de la dimensión Product

Product Name	Discontinued	Category Key
prod1	No	c2
prod2	No	c1
prod3	No	c2
prod4	No	c2

# Dimensiones que cambian espacio en Copo de Nieve

- Manejadas de forma similar

- Ejemplo

Product Key	Product Name	Discontinued	Category Key
p1	prod1	No	c1
p2	prod2	No	c1
p3	prod3	No	c2
p4	prod4	No	c2

Category Key	Category Name	Description
c1	cat1	desc1
c2	cat2	desc2
c3	cat3	desc3
c4	cat4	desc4

- Si p1 cambia su categoría a c2. En una solución de Tipo 2, añadimos 2 atributos temporales a la tabla Product:

Product Key	Product Name	Discontinued	Category Key	From	To
p1	prod1	No	c1	2010-01-01	2011-12-31
<b>p11</b>	prod11	No	<b>c2</b>	2012-01-01	Now
p2	prod2	No	c1	2010-01-01	Now
p3	prod3	No	c2	2010-01-01	Now
p4	prod4	No	c2	2011-01-01	Now

- La tabla Category permanece intacta

# Dimensiones que cambian espacio en Copo de Nieve

- Si el cambio ocurre en un nivel más alto de la jerarquía, por ejemplo, en una descripción, debe propagarse hacia abajo
- Ejemplo: La descripción de la categoría cat1 cambia

Category Key	Category Name	Description	From	To
c1	cat1	desc1	2010-01-01	2011-12-31
<b>c11</b>	cat1	<b>desc11</b>	2012-01-01	Now
c2	cat2	desc2	2012-01-01	Now
c3	cat3	desc3	2010-01-01	Now
c4	cat4	desc4	2010-01-01	Now

- El cambio debe propagarse a la tabla Product

Product Key	Product Name	Discontinued	Category Key	From	To
p1	prod1	No	c1	2010-01-01	2011-12-31
<b>p11</b>	prod1	No	<b>c11</b>	2012-01-01	Now
p2	prod2	No	c1	2010-01-01	Now
p3	prod3	No	c2	2010-01-01	Now
p4	prod4	No	c2	2011-01-01	Now



## Ejercicio 2. Wholesale furniture company

- Diseñar el *data warehouse* para una empresa mayorista de muebles. El *data warehouse* debe permitir analizar la situación de la empresa al menos en lo que respecta a Mobiliario, Clientes y Tiempo. Además, la empresa necesita analizar:
  - los muebles con respecto a su tipo (silla, mesa, armario, mueble...), categoría (cocina, sala, dormitorio, baño, oficina...) y material (madera, mármol...)
  - los clientes con respecto a su ubicación espacial, considerando al menos ciudades, regiones y estados
- La empresa está interesada en conocer al menos la cantidad, los ingresos y el descuento de sus ventas:
  - Determina los hechos, medidas y dimensiones



## Ejercicio 2. Wholesale furniture company

---

**SALES** (IDSale, Date, IDFurniture, IDCustomer, Quantity,  
Cost, Discount)

**FURNITURE** (IDFurniture, FurnitureType, FurnitureName,  
Category)

**CUSTOMER** (IDCustomer, Name, Surname, Birthdate, Sex, City)