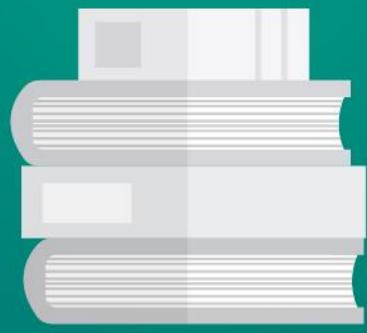


RDM y MDM



Base de Datos II

UNIVERSIDAD
SIGLO 21

MIEMBRO DE LA RED
ILUMNO



RDM y MDM

Análisis

Multidimensional

El análisis multidimensional nos facilita el análisis de un hecho desde distintas perspectivas o dimensiones. Esta es la forma natural que se aplica para analizar la información por parte de los tomadores de decisiones, ya que los modelos de negocio normalmente son multidimensionales.

(Cano, 2007, p. 126)

Introducción al modelado

¿Qué es un modelo?

Un **modelo** es una abstracción de la realidad que, como sabemos, suele ser muy compleja. En particular, podemos hablar de un **modelo de datos**, que no es otra cosa que la representación de la estructura de los datos almacenados en un Sistema de Gestión de Bases de Datos (DBMS).

El paradigma de modelado de datos por excelencia en los sistemas transaccionales/operacionales es el *Relacional*. Si bien se utiliza también en las bases de datos analíticas o decisionales, el modelado *Multidimensional* es, por lejos, el más usado.

Modelado Relacional

De acuerdo con Josep Lluís Cano (2007, p. 71), “una vez definido el modelo de negocio debemos determinar qué información tenemos para analizarlo”.

Para ello, la clave está en comprender el modelo Entidad - Relación de la base de datos origen o fuente de datos.

Se trata del modelo relacional desarrollado por E. F. Codd en el año 1970: Está formado por tablas y relaciones entre las mismas.

La mayoría de las aplicaciones de gestión utilizan bases de datos fundamentadas en el modelo relacional. El modelo relacional utiliza un lenguaje de interrogación conocido por Standard Query Language (SQL).

El modelo relacional se basa, pues en tablas con distintos atributos o campos y las relaciones entre las tablas. Cada tabla tiene una Clave Primaria (“Primary Key” o PK) formada por uno o más atributos y las tablas se relacionan entre ellas mediante las Claves Externas o Foráneas (“Foreign Key” o FK) que actúan como claves primarias en sus propias tablas.

(Cano, 2007, p. 71)

Es decir, en el modelo Relacional tenemos Tablas (Entidades) y relaciones entre ellas. Cada tabla tiene N atributos o campos, de los cuales los más importantes son la Clave Primaria (PK) y las Claves Foráneas (FK).

Y el lenguaje por excelencia para la consulta de datos es el SQL. En el Anexo del libro de referencia de Josep Lluís Cano (2007) se puede encontrar un resumen con las características principales del lenguaje SQL (páginas 353 a 379 inclusive).

Una base de datos relacional es una base de datos que es percibida por el usuario como una colección de tablas. Cada tabla está formada por filas (registros o tuplas) y columnas (atributos o campos). Las tablas están compuestas por registros o tuplas y cada uno de los registros tiene distintos atributos o campos.

Las tablas o relaciones deben cumplir varios requisitos:

- No existen registros repetidos.

- El orden de los registros no es significativo.
- El orden de los atributos no es significativo.
- Los valores de los atributos son atómicos.

Codd propuso las bases de datos relacionales en 1970, pero fue posteriormente, en 1974, cuando Chanberlin y Boyce publicaron un artículo proponiendo un lenguaje estructurado que llamaron SEQUEL, que es el origen del SQL.

El modelo relacional fue desarrollado posteriormente por C.J. Date y H. Darwen, entre otros.

Las bases de datos relacionales tienen al menos dos características:

- La información está integrada, generalmente en un solo lugar de la base de datos. Los usuarios de los distintos procesos acceden a la misma información en una sola ubicación, lo que minimiza las redundancias de información.
- La información es relacional, está interrelacionada con otras informaciones en otras partes de la base de datos.

(Cano, 2007, pp. 354-355)

Modelado Multidimensional

Como sostiene Josep Lluís Cano (2007), a partir del esquema entidad-relación, original de los sistemas transaccionales, se deberá construir el esquema multidimensional (usualmente tipo estrella, como veremos más adelante en esta misma lectura) que nos permitirá analizar la información en la base de datos analítica.

A pesar de algunas desventajas que ya vimos en la unidad 2 (según la opinión de William Inmon), de acuerdo con el enfoque de Ralph Kimball (2013) el modelado multidimensional es ampliamente aceptado como la técnica preferida para presentar datos analíticos ya que trata dos requerimientos simultáneos:

- Entregar datos que sean comprendidos por los usuarios del negocio.

- Responder consultas con rápida performance.

El **Modelado Multidimensional** es una técnica que permite simplificar el diseño de bases de datos. La simplicidad es crucial porque asegura que los usuarios entienden los datos, además de permitirle al software navegar y entregar resultados rápida y eficientemente.

Muchas personas encuentran intuitivo pensar en un negocio en términos de un cubo de datos con los ejes: producto, mercado y tiempo. Los puntos dentro del cubo contienen las métricas tales como volumen de ventas o ganancias. Es decir, el Modelado Multidimensional es una técnica que permite representar los requerimientos de datos de forma simple y entendible, ya que se puede ver la información (hechos/indicadores/métricas, generalmente numéricos) desde diferentes puntos de vista (dimensiones).

Según Ralph Kimball (2013), si bien los modelos multidimensionales frecuentemente se diseñan también en **RDBMS** (Sistemas de Gestión de Bases de Datos **Relacionales**), difieren de los modelos en tercera forma normal en tanto que se permiten eliminar las redundancias. Las estructuras normalizadas dividen los datos en muchas entidades discretas, cada una de las cuales se convierte en una tabla relacional.

La industria se refiere a los modelos en tercera forma normal como modelos entidad-relación (o simplemente **Modelos Relacionales**).

Los Diagramas de Entidad-Relación (DER) permiten comunicar las relaciones entre tablas. Tanto los modelos Relacionales como los modelos Multidimensionales pueden representarse con DER porque consisten en tablas relacionadas; la diferencia clave es el grado de normalización.

Las estructuras normalizadas son útiles en el procesamiento operacional porque un "update" o un "insert" toca la base de datos en un solo lugar.

Los modelos normalizados, sin embargo, son demasiado complicados para las consultas de BI. Los usuarios no pueden comprender, navegar o recordar modelos normalizados. Por otra parte, muchos RDBMS no pueden consultar eficientemente un modelo normalizado; la complejidad de las impredecibles consultas de los usuarios supera los optimizadores de bases de datos causando mala performance.

En otras palabras, un modelo multidimensional contiene la misma información que un modelo relacional normalizado, pero empaqueta los datos en un formato tal que permite entregar al usuario un mayor grado de comprensibilidad y performance en las consultas.

RDM vs. MDM

Como hemos visto en el Módulo 2, punto 2.3: *Tipos de Implementación*, existen dos modelos básicos para el diseño de la base de datos de un Data Warehouse: el modelado Relacional (RDM) y el modelado Multidimensional (MDM).

Si bien cada uno tiene sus ventajas y desventajas, en la actualidad el modelado Multidimensional, tal como señalábamos en el apartado anterior, se ha convertido en el enfoque más utilizado para el diseño de un Data Warehouse (y/o un Data Mart).

Ralph Kimball (2013) establece que existen cinco mitos sobre el modelado Multidimensional:

- **Mito 1:** Los modelos multidimensionales solamente son para datos summarizados.
- **Mito 2:** Los modelos multidimensionales son departamentales, no empresariales.
- **Mito 3:** Los modelos multidimensionales no son escalables.
- **Mito 4:** Los modelos multidimensionales sólo son para uso predictivo.
- **Mito 5:** Los modelos dimensionales no pueden ser integrados.

A continuación, veremos en detalle cada uno de estos falsos mitos, según lo que establece este mismo autor (Kimball, 2013).

Mito 1: Los modelos multidimensionales solamente son para datos summarizados

Dado que no se pueden predecir todas las consultas que harán los usuarios, es necesario que se acceda a los datos detallados.

Los datos summarizados deberían complementar los datos detallados para una mejor performance de consultas comunes, pero no reemplazarlos.

Por otra parte, no hay límites en cuanto a la cantidad de registros históricos que pueda tener un modelo multidimensional.

Mito 2: Los modelos multidimensionales son departamentales, no empresariales

En lugar de tener en cuenta los límites en términos de departamentos organizacionales, los modelos multidimensionales deberían organizarse en términos de los procesos de negocio.

Múltiples departamentos frecuentemente quieren analizar las mismas métricas del mismo proceso de negocio. Múltiples extracciones ETL de la misma fuente de datos, que crean múltiples e inconsistentes bases de datos analíticas, deberían evitarse.

Mito 3: Los modelos multidimensionales no son escalables

Los modelos multidimensionales son extremadamente escalables; las tablas de *hechos* frecuentemente tienen millones de filas. Los proveedores de soluciones de BI continúan incorporando capacidades en sus productos para optimizar la escalabilidad y performance de los modelos multidimensionales.

Ambos modelos, normalizado y multidimensional, contienen la misma información y relaciones de datos, el contenido lógico es idéntico. Cada relación de datos expresada en un modelo puede ser precisamente expresada en el otro, y ambos modelos pueden responder exactamente las mismas preguntas, aunque varía la dificultad.

Mito 4: Los modelos multidimensionales sólo son para uso predictivo

Los modelos multidimensionales no deberían diseñarse enfocados en reportes o análisis predefinidos, sino que el diseño debería centrarse en procesos de medición.

Si bien es importante considerar los requerimientos de filtros de las aplicaciones de BI, no debería basarse solo en ello porque esto varía a menudo haciendo al modelo multidimensional muy cambiante. La clave es enfocarse en los eventos de medición de la organización, que

generalmente son estables, excepto por los análisis que están continuamente evolucionando.

Debido a su simetría, las estructuras multidimensionales son flexibles al cambio. El secreto está en diseñar las tablas de *hechos* con el máximo nivel de detalle, ya que esto da más flexibilidad y extensibilidad.

Mito 5: Los modelos multidimensionales no pueden ser integrados

Al contrario, los modelos multidimensionales sí pueden ser integrados. Las dimensiones compartidas se diseñan de manera centralizada, persistentes y son reutilizadas entre los distintos modelos multidimensionales para permitir la integración de datos y asegurar una consistencia semántica. La integración de datos depende de etiquetas, valores y definiciones estandarizadas, lo cual requiere de consenso organizacional.



Bibliografía de referencias

- **Cano, J. L.** (2007). *Business Intelligence: Competir con Información*. España: Banesto Fundación Cultural y ESADE.
- **Inmon, W.** (2005). *Building the Data Warehouse* (4º Edición). Estados Unidos: Wiley Publishing.
- **Kimball Ralph, R. M.** (2013). *The Data Warehouse Toolkit* (3º Edición). Estados Unidos: Wiley Publishing.
- **QlikTech** (2010). *QlikView architectural overview* (White Paper). Recuperado el 7 de Mayo de 2014: http://www.swbi.co.uk/pdf/QlikView_Architectural_Overview.pdf
- **QlikTech** (s.f.). *QlikView es pionero del BI en memoria*. Recuperado el 7 de Mayo de 2014: <http://www.qlik.com/es/explore/products/overview>

Componentes MDM



Base de Datos II

UNIVERSIDAD
SIGLO 21

MIEMBRO DE LA RED
ILUMNO



Componentes MDM

Principales Componentes de un Modelo Multidimensional

De acuerdo con Ralph Kimball (2013), los principales componentes de un modelo Multidimensional son:

- Hechos
- Dimensiones
- Elementos
- Atributos
- Miembros
- Tablas de Hechos
- Tablas de Dimensión
- Esquemas.

Hechos

Un **Hecho** es un atributo que representa una métrica del negocio respecto a una o más dimensiones. Se almacenan en las Tablas de Hechos.

Ejemplo: Ventas en \$.

Dimensiones

Una **Dimensión** hace referencia a los distintos puntos de vista (o perspectivas) a través de los cuales se puede analizar un Hecho. Se almacenan en las Tablas de Dimensión.

Ejemplos: Tiempo (año, mes, día, entre otros), zona geográfica (país, provincia/estado, localidad, entre otros), etc., lo cual permite analizar, por ejemplo, las ventas totales por provincia y por mes.

Elementos de Dimensión

Un **Elemento de Dimensión** es un componente de una Dimensión, que permite agrupar lógicamente determinados atributos del negocio. Generalmente, tienen una organización jerárquica para que se pueda navegar la información por distintos niveles de detalle.

Ejemplo: la Provincia dentro de la dimensión Zona Geográfica.

Atributos de Dimensión

Un **Atributo de Dimensión** contiene la descripción y otros datos relacionados con cada Elemento de Dimensión. Es decir, un mismo Elemento de Dimensión puede tener varios Atributos.

Ejemplo: el elemento provincia dentro de la dimensión Zona Geográfica tiene varios atributos tales como: código, descripción y región (sur, norte, este, oeste).

Miembro de Dimensión

Un **Miembro de Dimensión** representa cada una de las instancias reales y concretas de una Dimensión en particular.

Ejemplo: para la dimensión Zona Geográfica, un miembro es la ciudad de Mendoza en la provincia de Mendoza, en el país Argentina.

Tablas de hechos para mediciones

Según Ralph Kimball (2013), la **tabla de Hechos** en un modelo multidimensional almacena las mediciones resultantes de los eventos de los procesos de negocio de una organización.

Cada medición debería almacenarse en un único modelo multidimensional. Permitir a usuarios de múltiples departamentos acceder a un repositorio centralizado asegura el uso de datos consistentes.

En una tabla de hechos, cada fila se corresponde con un evento de medición y los datos en cada fila están en un nivel específico de detalle/granularidad (por ejemplo: un producto vendido en una transacción de venta). Además, todas las filas deben tener la misma granularidad. Con la disciplina de crear tablas de hechos con igual nivel de detalle se asegura que las mediciones sean correctas.

Los hechos más útiles son numéricos y aditivos, como por ejemplo, las ventas en pesos. La **aditividad** es crucial, porque las aplicaciones de BI rara vez consultan una sola fila de la tabla de hechos; por lo general, consultan miles de filas para traer un valor sumarizado. Sin embargo, hay hechos no aditivos, como el precio unitario, que nunca pueden ser sumados; en cambio, se podrá hacer cuentas o promedios.

No se debe almacenar información textual redundante en una tabla de hechos, excepto que el texto sea único por cada fila, caso en el cual debería ir a una tabla de dimensión.

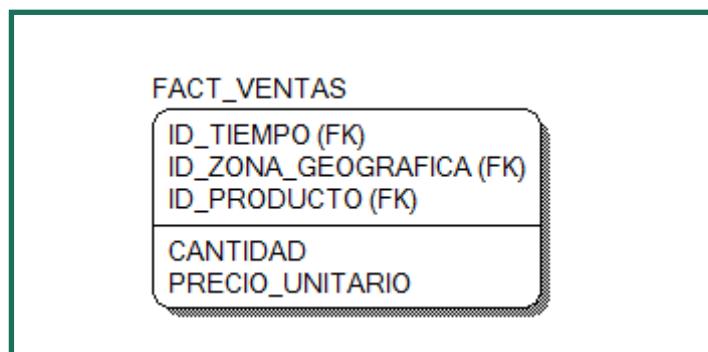
Por otro lado, si no hubo ventas para un producto, no debe incluirse una fila con ceros en la tabla de hechos. No obstante, las tablas de hechos consumen el 90% de espacio de una base de datos multidimensional. Dichas tablas tienden a ser profundas en cantidad de filas, pero generalmente de pocas columnas.

Todas las tablas de hechos tienen 2 o más claves foráneas (FK) que las conectan con las tablas de dimensión. La tabla de hechos generalmente tiene su propia clave primaria, compuesta por un subconjunto de las FK de las dimensiones. Toda tabla que tenga clave compuesta es una tabla de hechos, dado que expresan relaciones muchos a muchos; las restantes son tablas de dimensión.

Por ejemplo, podemos tener una tabla de Hechos con los siguientes campos:



Imagen 1: Tabla de Hechos



Fuente: Elaboración propia

En este ejemplo, por cada combinación de tres Dimensiones distintas: Tiempo (año, mes, día, etc.), Zona Geográfica (país, provincia, localidad, etc.) y Producto (categoría y descripción), contamos con dos Hechos: la Cantidad de Unidades Vendidas del Producto (en cada día y localidad) y el Precio Unitario al cual se vendió.

En definitiva, la Tabla de Hechos es el núcleo de un modelo multidimensional. Cada registro de la tabla representa una combinación de los valores claves de las dimensiones, representando, por lo tanto, un evento o una transacción del negocio.

Tablas de Dimensión para Contexto Descriptivo

De acuerdo con Ralph Kimball (2013), las **tablas de Dimensión** acompañan a una tabla de Hechos. Aquellas contienen el contexto textual asociado con un evento de medición del proceso de negocio; describen el "quién, qué, dónde, cuándo, cómo y por qué" asociados al evento.

Frecuentemente, dichas tablas tienen muchas columnas o atributos. A veces hasta 50 o 100 atributos, aunque algunas tablas de dimensión naturalmente tienen solo unos cuantos. Además, tienden a tener menos filas que una tabla de hechos. Cada tabla se define por una clave primaria (PK) simple, lo que permite integridad referencial con cualquier tabla de hechos.

Los **atributos de dimensión** sirven como la fuente primaria de restricciones de las consultas, agrupamientos y etiquetas de reportes. En una consulta, los atributos se identifican con la palabra "por". Ejemplo: cuando un usuario quiere ver las ventas por marca, las marcas deben estar disponibles como un atributo de dimensión.

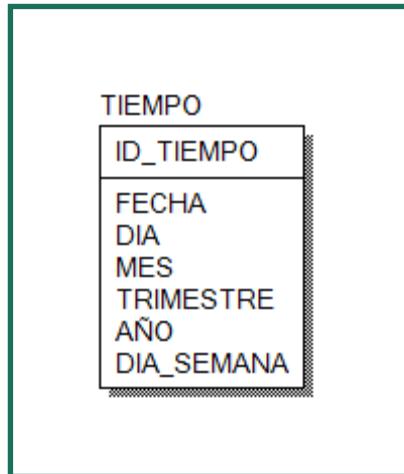
Los atributos de dimensión juegan un rol clave en un sistema de BI. Deben estar escritos con palabras reales y no abreviaturas. Debe minimizarse el uso de códigos, reemplazándolos por atributos textuales. Cuando los códigos tienen significancia real para los usuarios, deben aparecer como atributos explícitos. El poder analítico de un Data Warehouse es directamente proporcional a la calidad y profundidad de los atributos de dimensión.

Las tablas de dimensión suelen representar relaciones jerárquicas. Por ejemplo: los productos podrían agruparse en marcas, y las marcas en categorías. De este modo, por cada fila de la dimensión *Producto* deberían almacenarse las descripciones de la marca y categorías asociadas. Esta información redundante permite una mayor facilidad de uso y mejor performance de las consultas. Es decir, la característica principal de las tablas de dimensión es la **de-normalización**. Debería evitarse crear una tabla *Marcas* y otra *Categorías*. Cuando se aplica la normalización, se llama **copo de nieve**. En lugar de usar la tercera forma normal, las tablas de dimensión generalmente están altamente de-normalizadas con relaciones muchos a uno dentro de una misma tabla de dimensión. Dado que tienen un tamaño menor que las tablas de hechos, mejorar la eficiencia de almacenamiento a través de la normalización o copo de nieve casi no tiene impacto sobre el tamaño total de la base de datos.

Siguiendo con el ejemplo anterior, tenemos las tablas de dimensión correspondientes a las tres dimensiones mencionadas:



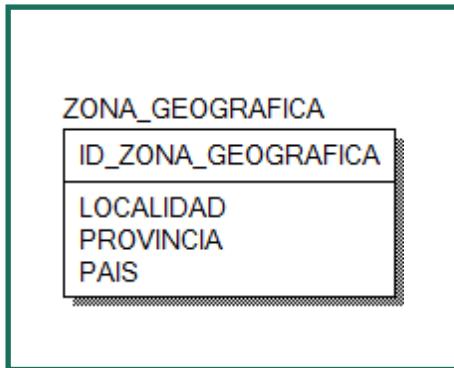
Imagen 2: Tabla de Dimensión Tiempo



Fuente: Elaboración propia



Imagen 3: Tabla de Dimensión Zona Geográfica



Fuente: Elaboración propia



Imagen 4: Tabla de Dimensión Producto



Fuente: Elaboración propia

Hechos y Dimensiones relacionados en un Esquema Estrella

Cada proceso de negocios se representa por un modelo multidimensional que consiste en una tabla de hechos que contiene las mediciones numéricas del evento, rodeada de varias tablas de dimensiones que contienen el contexto textual que fue correcto o verdadero en el momento en que ocurrió el evento. Esta estructura particular se denomina Esquema Estrella.

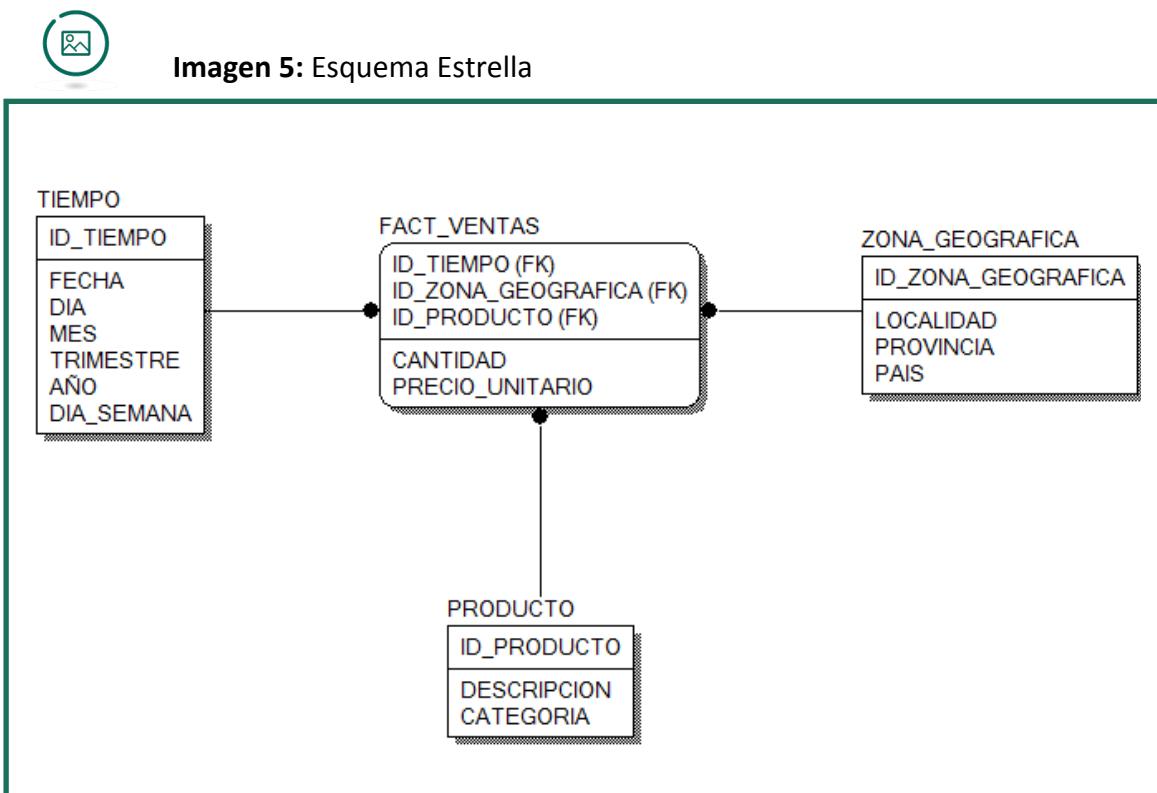
Una de las características de este modelo es su simplicidad: los datos son más fáciles de comprender y navegar gracias al número reducido de tablas y descripciones amigables; además, es beneficiosa en cuanto a la performance, al necesitar las consultas menos joins. Por otra parte, se adapta mejor a los cambios. Es un modelo simétrico.

Cuento más granulares o atómicos sean los datos, mayor dimensionalidad habrá. Los datos detallados deberían ser el fundamento para cada diseño de tablas de hechos.

Con los modelos multidimensionales se pueden agregar nuevas dimensiones y hechos, suponiendo que el nivel de detalle es consistente con la tabla existente.

Mientras los hechos proveen los valores numéricos de un reporte, los atributos de dimensión proveen los filtros de un reporte.

A continuación, podemos ver cómo nos quedó el ejemplo con el esquema Estrella:



Fuente: Elaboración propia

Como puede deducirse, la consulta SQL escrita o utilizada por una herramienta BI es muy simple.

En el SELECT se ubican los atributos de dimensión que se quieren leer, seguida por la métrica correspondiente summarizada, por ejemplo un SUM (CANTIDAD).

La cláusula FROM identifica todas las tablas de dimensión involucradas. En el WHERE se identifica el filtro solicitado en el reporte por el usuario, además de los joins correspondientes, y en el GROUP BY se establecen las distintas summarizaciones o agrupaciones.

Para poder obtener las Ventas Totales es necesario multiplicar los campos CANTIDAD y PRECIO_UNITARIO.



Bibliografía de referencias

- **Cano, J. L.** (2007). *Business Intelligence: Competir con Información*. España: Banesto Fundación Cultural y ESADE.
- **Inmon, W.** (2005). *Building the Data Warehouse* (4º Edición). Estados Unidos: Wiley Publishing.
- **Kimball Ralph, R. M.** (2013). *The Data Warehouse Toolkit* (3º Edición). Estados Unidos: Wiley Publishing.
- **QlikTech** (2010). *QlikView architectural overview* (White Paper). Recuperado el 7 de Mayo de 2014: http://www.swbi.co.uk/pdf/QlikView_Architectural_Overview.pdf
- **QlikTech** (s.f.). *QlikView es pionero del BI en memoria*. Recuperado el 7 de Mayo de 2014: <http://www.qlik.com/es/explore/products/overview>

Esquema de implementación



Base de Datos II

UNIVERSIDAD

SIGLO 21

MIEMBRO DE LA RED
ILUMNO



Esquema de Implementación

Como adelantamos en los apartados anteriores, pueden establecerse distintos esquemas de implementación de un modelo Multidimensional:

- **Estrella** (cuando un modelo está de-normalizado, con una sola tabla de hechos y varias tablas de dimensión).
- **Copo de Nieve** (semejante al anterior, pero existen varias tablas normalizadas por cada dimensión, por ejemplo: para la dimensión Zona Geográfica tendríamos tres tablas: País, Provincia y Localidad).
- **Constelación** (cuando se unen varios modelos Estrella a través de dimensiones compartidas).

Esquemas Estrella

Según Josep Lluís Cano (2007, p. 75), “para la construcción del esquema estrella debemos distinguir entre las **tablas de hechos** (aquellos que queremos medir o analizar) y las **tablas de dimensiones** (cómo lo queremos medir)”.

Las características del esquema estrella son:

- Una tabla de hechos que contiene los datos sin redundancias.
- Una sola tabla por dimensión.
- La tabla de hechos (Fact table) tiene un atributo columna que forma la clave de cada dimensión.
- Cada tabla de dimensión (Dimension table) es una tabla simple desnormalizada.

(Cano, 2007, p. 79)

El esquema Estrella (*Star*, en inglés) es el más común y, en general, el más recomendado.

Esquemas Copos de Nieve

Generalmente, se tiende a no utilizar esquemas Copos de Nieve (también conocidos como *Snowflake* en inglés).

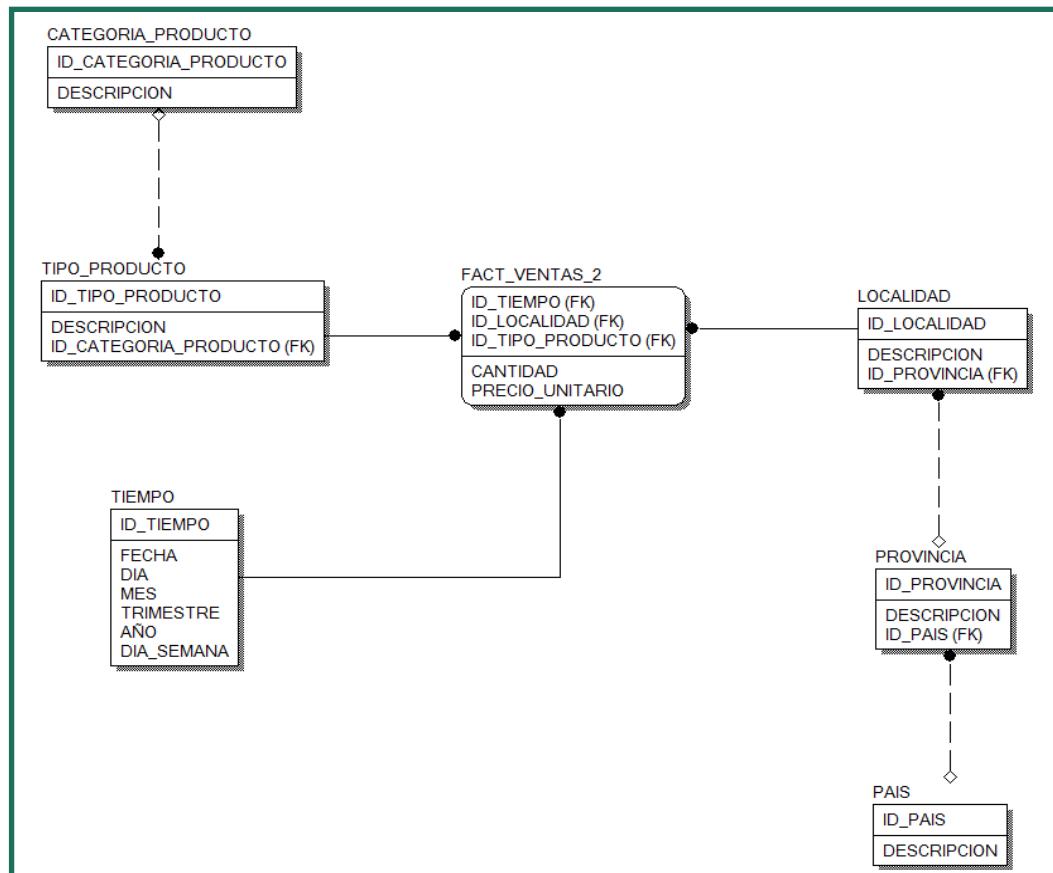
En el esquema “copo de nieve” aparecen relaciones entre las tablas de dimensiones, mientras que en el esquema “estrella” sólo hay relaciones entre la tabla de hechos y las de dimensiones.

En este caso, las tablas de dimensiones están totalmente normalizadas, lo que reduce el espacio que ocupan, aunque en algunos casos esta diferencia no es significativa. (Cano, 2007, p. 80)

Para el ejemplo anteriormente mencionado, tendríamos el siguiente modelo de esquema Copo de Nieve, en el cual la dimensión Zona Geográfica es dividida en tres tablas distintas para almacenar las Localidades, Provincias y Países; mientras que la dimensión Producto es dividida en dos tablas distintas, una para almacenar los Tipos de Productos y otra para las Categorías de Productos:



Imagen 6: Esquema Copo de Nieve



Fuente: Elaboración propia

Esquemas Constelación

Es frecuente ver un esquema Constelación (*Constellation* en inglés) en grandes sistemas de BI en donde hay varias tablas de hechos (es decir, varios modelos multidimensionales unidos a través de dimensiones compartidas). Como indica Josep Lluís Cano, «Cuando unimos distintos esquemas “estrella” que tienen distintas tablas de hechos, pero comparten las de las dimensiones, hablamos de constelaciones de hechos; algunos autores hablan incluso de esquema “galaxia”» (2007, p. 79).

Si bien puede haber una Constelación de esquemas Estrella, también podría suceder que se trate de una Constelación de esquemas Copos de Nieve.

Multidimensionalidad

Según Josep Lluís Cano (2007, p. 84), “al poder utilizar las distintas dimensiones a la vez estamos utilizando la funcionalidad de la multidimensionalidad. La multidimensionalidad nos permite analizar la información por distintas dimensiones a la vez”.

Así, para el ejemplo anterior, podemos analizar las Ventas Totales por varias dimensiones en simultáneo, es decir, ver la evolución de las ventas por cada una de las Provincias de Argentina, para el 3º Trimestre de 2012 y la categoría de Productos de Alta Gama.

La multidimensionalidad es un aspecto clave y, desde luego, de muy frecuente consulta por parte de los usuarios finales.

Conceptos relacionados

Esquemas Estrella vs. Cubos OLAP

Según Ralph Kimball (2013), los modelos multidimensionales implementados en RDBMS son denominados esquemas **Estrella** por su estructura similar. En cambio, los modelos multidimensionales implementados en bases de datos multidimensionales (MDBMS) son llamados **cubos OLAP**. Tanto una arquitectura como la otra incluyen modelos multidimensionales, es decir, un diseño lógico común; la diferencia está en la implementación física.

Cuando los datos se cargan en un cubo OLAP, se almacenan e indexan usando formatos y técnicas diseñadas para datos multidimensionales. Además, los datos pre-calculados o summarizaciones son calculadas por el motor OLAP. Por ello, los cubos OLAP tienden a tener mejor performance de consulta.

Los usuarios de negocio pueden hacer *drill-down* o *drill-up*, agregando o eliminando atributos de sus análisis con excelente performance sin hacer nuevas consultas.

Los cubos OLAP también proveen funciones analíticas más robustas que las disponibles en el lenguaje SQL. La desventaja es que se paga un precio en performance de carga, especialmente en grandes volúmenes de datos. Aunque las capacidades de la tecnología OLAP están continuamente mejorando, generalmente se recomienda que la información atómica y detallada se cargue en un esquema estrella sobre un RDBMS. Los cubos OLAP, de desarrollo opcional, luego se cargan a partir del esquema estrella inicial.

Hay que tener en cuenta que un cubo OLAP contiene atributos dimensionales y hechos, pero se accede a través de lenguajes con más capacidades analíticas que el SQL; entre ellos, podemos mencionar XML for Analysis (XMLA) y MDX. Estos lenguajes tienen su propia sintaxis, diferente del SQL tradicional.

Consideraciones sobre el Despliegue en Cubos OLAP

Según Ralph Kimball (2013):

- Un esquema Estrella hosteado en una RDBMS es un buen fundamento físico para diseñar un cubo OLAP, y generalmente es una base más estable para *backup* y recuperación de datos.
- Los cubos OLAP tradicionalmente han sido destacados por presentar grandes ventajas en performance sobre los RDBMS tradicionales; sin embargo, esta distinción se ha vuelto cada vez menos importante con los últimos avances en hardware, BD ‘in-memory’, etc.
- Las estructuras de datos de cubos OLAP (MDBMS) son más variables entre los diferentes proveedores de bases de datos que los tradicionales RDBMS, por lo tanto, el despliegue final depende de qué motor OLAP se seleccione. Usualmente es más difícil portar aplicaciones BI entre diferentes herramientas OLAP que entre diferentes RDBMS.
- Los cubos OLAP, por lo general, ofrecen opciones de seguridad más sofisticadas que los RDBMS, limitando el acceso a datos detallados pero dando más acceso abierto a datos sumarizados.
- Al no tener las restricciones del SQL, los cubos OLAP ofrecen capacidades de análisis significativamente más ricas que los RDBMS. Esta es la principal justificación para utilizar un producto OLAP.
- Los cubos OLAP necesitan frecuentemente ser re-procesados parcial o totalmente.
- Los cubos OLAP soportan complejas jerarquías de profundidad indeterminada, tales como un organigrama, usando sintaxis de consulta nativa, superior a los enfoques para RDBMS.
- Los cubos OLAP pueden imponer restricciones detalladas en la estructura de las claves de dimensión que implementan jerarquías *drill-down* en comparación con los RDBMS.
- Algunos productos OLAP no permiten roles o alias dimensionales, sino que requieren dimensiones físicas separadas para poder ser definidas.

En definitiva, según Ralph Kimball (2013) –y como puede apreciarse en los puntos anteriores-, sólo en determinadas situaciones es recomendable avanzar hacia la inclusión de un MDBMS y un producto OLAP en particular.



Bibliografía de referencias

- **Cano, J. L.** (2007). *Business Intelligence: Competir con Información*. España: Banesto Fundación Cultural y ESADE.
- **Inmon, W.** (2005). *Building the Data Warehouse* (4º Edición). Estados Unidos: Wiley Publishing.
- **Kimball Ralph, R. M.** (2013). *The Data Warehouse Toolkit* (3º Edición). Estados Unidos: Wiley Publishing.
- **QlikTech** (2010). *QlikView architectural overview* (White Paper). Recuperado el 7 de Mayo de 2014: http://www.swbi.co.uk/pdf/QlikView_Architectural_Overview.pdf
- **QlikTech** (s.f.). *QlikView es pionero del BI en memoria*. Recuperado el 7 de Mayo de 2014: <http://www.qlik.com/es/explore/products/overview>

Mejoras de performance



Base de Datos II

UNIVERSIDAD
SIGLO 21

MIEMBRO DE LA RED
ILUMNO



Mejoras de Performance

Indexación, Tablas Sumarizadas y Particionamiento en el Data Warehouse

Como destaca William Inmon (2005), existen diferentes formas para mejorar la performance de las aplicaciones de BI. Puntualmente, podemos mencionar las siguientes técnicas que se pueden aplicar en el Data Warehouse:

- Indexación
- Tablas Sumarizadas
- Particionamiento

El caso del **Particionamiento** ya lo hemos analizado en detalle en la Unidad 2 (módulo 2). Así, para ejemplo de esta unidad, podríamos particionar la tabla de Hechos que contienen las Ventas con una tabla separada por cada año; esto impactará, lógicamente, en el tiempo de respuesta de las consultas que accederán a una cantidad menor de registros.

El ítem **Indexación** es esencial. Al menos las claves primarias de las distintas tablas (especialmente la tabla de hechos) deberían contar con su índice respectivo; pero también es conveniente crear índices para aquellas columnas que generalmente se usan como filtros en los reportes o sobre las cuales hay búsquedas frecuentes por parte de los usuarios.

En cuanto a las **Tablas Sumarizadas**, pueden ser determinantes para la performance de algunas consultas cuando la tabla de hechos contiene una gran cantidad de registros. En estos casos puede haber un procedimiento/goritmo que se encargue de llenar, por ejemplo, una tabla sumarizada todas las noches con las ventas totales por Categoría de Producto y otra por cada una de las provincias.

En definitiva, la búsqueda de una mejor performance es una actividad de *tuning* permanente por parte de un especialista en Business Intelligence puesto que aquí se juega, en muchas ocasiones, el prestigio de una solución de BI en términos de performance apropiada.

Business Intelligence In-Memory y Visualización por Lógica Asociativa

Sin lugar a dudas, el surgimiento en los últimos años de herramientas de BI de lógica asociativa con bases de datos ***in-memory*** ha mejorado sustancialmente la performance de las aplicaciones, aunque ello implique dejar de lado algunos de los principios tradicionales en cuanto a diseño arquitectónico.

El procesamiento *in-memory* es una tecnología emergente que habilita a los usuarios a contar con acceso inmediato a los datos con tiempos de respuesta óptimos. Esto se debe a que la tecnología tradicional de BI almacena la información en disco -ya sea tablas o cubos OLAP-.

Sin embargo, la tecnología *in-memory*, como lo indica su nombre, permite que los datos se carguen directamente en memoria RAM. Este tipo de herramientas no sólo obtiene ganancias en performance, sino también en tiempos de desarrollo de aplicaciones de BI; por ejemplo, al evitar la construcción de cubos OLAP.

No obstante, si bien algunos proveedores recomiendan el acceso directo a los datos transaccionales por parte de estas innovadoras herramientas, se aduce que no es lo suficiente aconsejable, puesto que si no existe un adecuado proceso ETL que alimente el Data Warehouse podemos perder significativamente niveles de calidad en los datos al no realizarse una apropiada integración y transformación de los datos primitivos operacionales.

Muchas organizaciones cometen el error de aplicar la tecnología de BI *in-memory* sin construir un Data Warehouse. Vale decir, el *approach* más aconsejable (en caso de optar por este tipo de herramientas) consiste en seguir todos los pasos del Corporate Information Factory hasta la construcción del Data Warehouse/Data Mart, y utilizar las herramientas de BI *in-memory*/lógica asociativa solamente para la visualización/explotación de los datos.

Según uno de los productos líderes en este mercado, las claves para la mejor performance de las aplicaciones de BI *in-memory* con lógica asociativa están dados por los siguientes aspectos:

- Tiene un motor de inferencia que mantiene las asociaciones entre los datos automáticamente.
- Calcula las agregaciones sobre la marcha, según se van necesitando, para una experiencia de usuario ultra rápida.

- Comprime los datos hasta un 10% de su tamaño original para optimizar la potencia de los procesadores.

(QlikTech, <http://goo.gl/BMi8Mr>, s.f.)

En este tipo de productos, toda la información necesaria se carga inicialmente en la memoria del servidor. Lo cual en principio elimina la necesidad de aplicar algún tipo de optimización de bases de datos (tal como la indexación o el particionamiento, por ejemplo).

Este repositorio *in-memory* permanece en memoria hasta que algún usuario realice actividad alguna.

Así, se puede cargar en memoria un Data Mart completo o un pequeño Data Warehouse incluso.

Por otra parte, muchas de estas herramientas utilizan algoritmos de compresión de datos que reducen el tamaño que los datos ocupan en memoria. El acceso por parte de los usuarios finales a estos datos es más rápido, puesto que no es necesario buscar la información en disco.

La mayor demanda de estas herramientas se dio a partir de la confluencia de varios factores, entre ellos:

- Costos reducidos en hardware.
- Arquitectura de procesador multi-núcleo.
- Memoria flash.
- Técnicas y algoritmos de compresión de datos que permiten minimizar el espacio de almacenamiento de los datos una vez que están en memoria.
- Sistemas de gestión de bases de datos orientados a las columnas (Column-Oriented DBMS, en inglés).
- Sistemas operativos de 64 bits.
- Grandes volúmenes de datos que dificultan la ejecución de los procesos ETL (ya sea por aspectos de complejidad de los algoritmos o, más frecuentemente, por la cantidad de tiempo requerido para movilizar datos desde un esquema transaccional hacia el Data Warehouse).

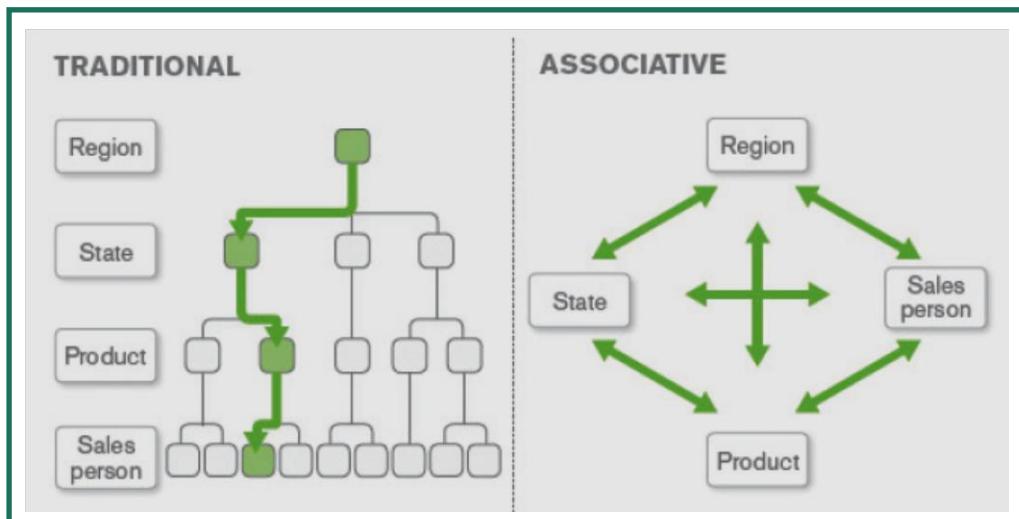
- Necesidad de analizar la información en tiempo real, dado que de manera frecuente las aplicaciones de BI se usan para tomar decisiones inmediatas a base de los perfiles históricos.
- Lógica de Consulta Asociativa (AQL, *Associative Query Logic* en inglés), que permite relacionar o asociar un determinado campo con cualquier otra columna de la base de datos.
- Otros aspectos.

La Lógica de Consulta Asociativa (AQL), mencionada en el párrafo anterior, realmente configura una innovación no solo en términos de performance, sino también en permitir análisis más desestructurados.

Por ejemplo, con la tecnología OLAP se diseñan dimensiones pre establecidas; es decir, la lógica de explotación de los datos por parte de un usuario final es muy estructurada en el sentido de que debe seguir la jerarquía de la dimensión, mientras que con AQL es posible detectar todo tipo de cruce de datos sin seguir una jerarquía de navegación pre establecida. Esto se debe a que AQL mantiene todas las asociaciones existentes entre los datos.



Imagen 7: AQL



Fuente: QlikView (2010). *QlikView Architectural Overview*. Pág. 3



Bibliografía de referencias

- **Cano, J. L.** (2007). *Business Intelligence: Competir con Información*. España: Banesto Fundación Cultural y ESADE.
- **Inmon, W.** (2005). *Building the Data Warehouse* (4º Edición). Estados Unidos: Wiley Publishing.
- **Kimball Ralph, R. M.** (2013). *The Data Warehouse Toolkit* (3º Edición). Estados Unidos: Wiley Publishing.
- **QlikTech** (2010). *QlikView architectural overview* (White Paper). Recuperado el 7 de Mayo de 2014: http://www.swbi.co.uk/pdf/QlikView_Architectural_Overview.pdf
- **QlikTech** (s.f.). *QlikView es pionero del BI en memoria*. Recuperado el 7 de Mayo de 2014: <http://www.qlik.com/es/explore/products/overview>