

INTELIGENCIA DE NEGOCIOS Y SISTEMAS DE INFORMACIÓN. INFORMES

FINALIDAD DE LOS SISTEMAS DE INFORMACIÓN Y ORIGEN DEL BUSINESS INTELLIGENCE

La información *reduce nuestra incertidumbre* (sobre algún aspecto de la realidad) y, por tanto, *nos permite tomar mejores decisiones*.

Inicialmente la finalidad de los sistemas de información era recopilar información sobre una parcela del mundo para ayudar en la toma de decisiones y se basaba en recuentos, censos civiles y militares, libros contables, etc. Actualmente, con la informatización de las organizaciones y la aparición de aplicaciones software operacionales sobre el sistema de información, la finalidad principal de los sistemas de información es dar soporte a los procesos básicos de la organización (ventas, producción, personal, etc.).

Una vez satisfecha la necesidad de tener un soporte informático para los procesos básicos de la organización (**sistemas de información para la gestión**), las organizaciones exigen nuevas prestaciones de los sistemas de información (**sistemas de información para la toma de decisiones**). Es aquí donde aparece el Business Intelligence.

HERRAMIENTAS PARA LA TOMA DE DECISIONES EN BUSINESS INTELLIGENCE. OLAP, INFORMES Y MINERÍA

Ante el problema de la toma de decisiones han aparecido diferentes herramientas de inteligencia de negocio o DSS que coexisten: EIS, OLAP, consultas & informes, minería de datos, etc.

Un EIS (*Executive Information System*) es un sistema de información y un conjunto de herramientas asociadas que tiene las siguientes características:

- Proporciona a los directivos acceso a la información de estado y sus actividades de gestión.
- Está especializado en analizar el estado diario de la organización (mediante indicadores clave) para informar rápidamente sobre cambios a los directivos.
- La información solicitada suele ser, en gran medida, numérica (*ventas semanales, nivel de stocks, balances parciales, etc.*) y representada de forma gráfica al estilo de las hojas de cálculo.

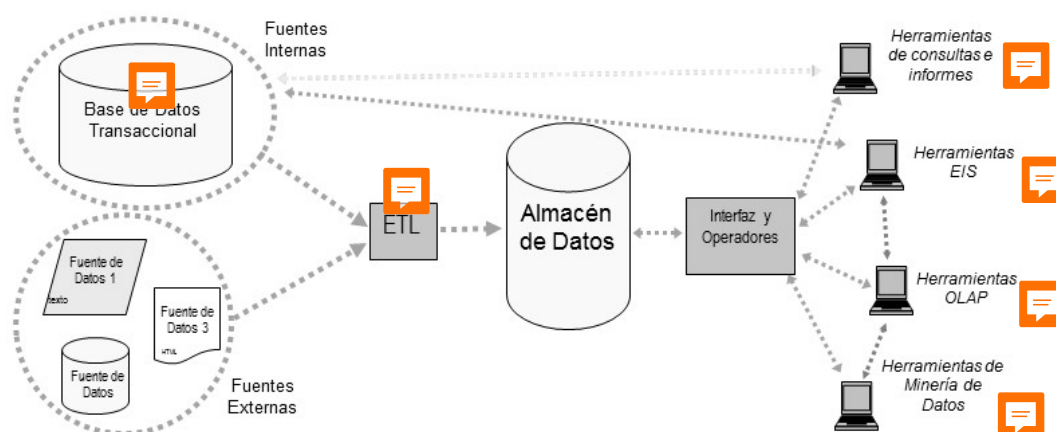
Las herramientas OLAP (*On-Line Analytical Processing*) son más genéricas:

- Funcionan sobre un sistema de información (transaccional o almacén de datos).
- Permiten realizar agregaciones y combinaciones de los datos de maneras más complejas y ambiciosas, con objetivos de análisis más estratégicos.
- Están basadas, generalmente, en sistemas o interfaces multidimensionales.
- Se utilizan operadores específicos (además de los clásicos): *drill, roll, pivot, slice & dice,...*
- El resultado se presenta de una manera matricial o híbrida.
- Proporcionan facilidades para “manejar” y “transformar” los datos.
- Producen otros “datos” (más agregados, combinados).
- Ayudan a analizar los datos porque producen diferentes vistas de los mismos.

Por otro lado, los *sistemas de informes* o *consultas avanzadas* están basados, generalmente, en sistemas relacionales u objeto-relacionales, utilizan los operadores clásicos como concatenación, proyección, selección, agrupamiento... (en SQL y extensiones) y el resultado se presenta de una manera tabular.

Instrumentos más avanzados para la toma de decisiones en inteligencia de negocios son las herramientas de Minería de Datos. Son muy variadas y permiten “extraer” patrones, modelos, descubrir relaciones, regularidades, tendencias, etc. También producen “reglas” o “patrones” (“conocimiento”).

La interrelación entre todas estas herramientas se presenta en el esquema siguiente:



Mediante las herramientas y técnicas ELT (extraer, cargar y transformar), o actualmente ETL (extraer, transformar y cargar) se extraen los datos de distintas fuentes externas e internas (bases de datos transaccionales), se depuran y preparan (homogeneización de los datos) para luego cargarlos en un almacén de datos. En el centro del esquema aparece el *almacén de datos*, que es el “**sistema de información central**” en todo este proceso. Un almacén de datos es una colección de datos orientada a un dominio, integrada, no volátil y variante en el tiempo para ayudar en la toma de decisiones. A partir del almacén de datos, mediante interfaces y operadores se utilizan las herramientas de informes, EIS, OLAP y Minería de Datos.

Los almacenes de datos y las técnicas OLAP son las maneras más efectivas y tecnológicamente más avanzadas para **integrar, transformar y combinar los datos para facilitar** al usuario o a otros sistemas **el análisis de la información**. La tecnología


OLAP generalmente se asocia a los almacenes de datos, aunque podemos tener almacenes de datos sin OLAP, y viceversa.

La minería de datos es solo una etapa del proceso de extracción de conocimiento a partir de datos. Consta de varias fases: Preparación de Datos (selección, limpieza y transformación), Análisis de Datos, Evaluación, Difusión y Uso de Modelos. Incorpora diferentes técnicas como árboles de decisión, regresión lineal, redes neuronales artificiales, técnicas predictivas, técnicas de segmentación, etc. Se aplica en campos diversos como el aprendizaje automático e inteligencia artificial, estadística, bases de datos, clasificación, categorización, estimación y regresión, agrupamiento, etc.

Los almacenes de datos no son *imprescindibles* para hacer extracción de conocimiento a partir de datos. Se puede hacer minería de datos sobre un simple fichero de datos. Las ventajas de organizar un almacén de datos para realizar minería de datos se amortizan sobradamente a medio y largo plazo cuando tenemos grandes volúmenes de datos, o estos aumentan con el tiempo, o provienen de fuentes heterogéneas o se van a combinar de maneras arbitrarias y no predefinidas.

ALMACENES DE DATOS. DATA WAREHOUSE

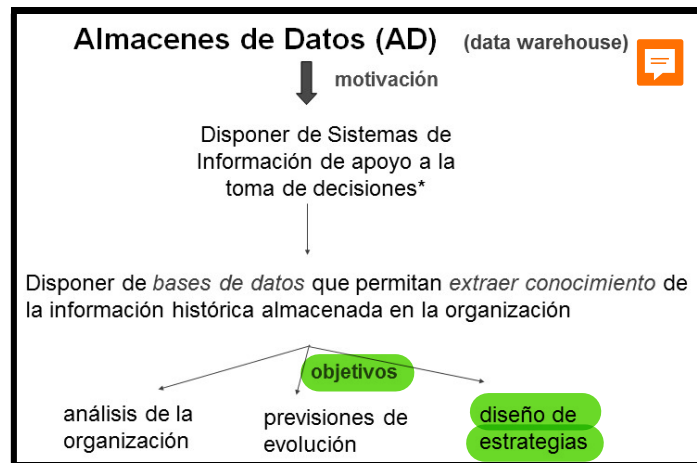
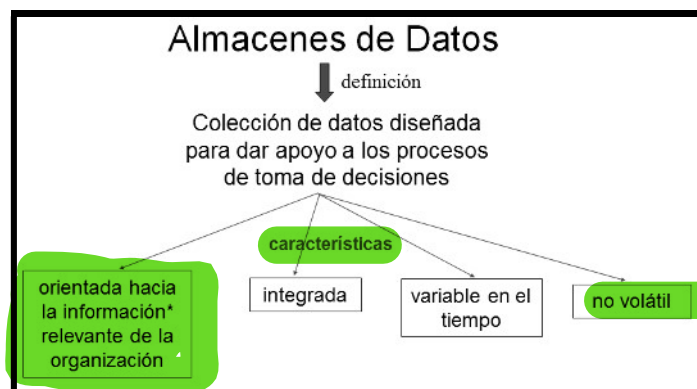
Generalmente, la información que se quiere investigar sobre un cierto dominio de la organización se encuentra en bases de datos y otras fuentes muy diversas, tanto internas como externas. Muchas de estas fuentes son las que se utilizan para el trabajo diario (*bases de datos operacionales*). Sobre estas mismas bases de datos de trabajo ya se puede extraer conocimiento (visión tradicional).

 Una *base de datos transaccional* es una fuente de datos mediante la cual se mantiene el trabajo transaccional diario de los sistemas de información originales (conocido como OLTP, *On-Line Transactional Processing*). También se hacen análisis de los datos en tiempo real sobre la misma base de datos (conocido como OLAP, *On-Line Analytical Processing*).

Como problemas más comunes, la base de datos transaccional perturba el trabajo transaccional diario de los sistemas de información originales (“killer queries”). Se debe hacer por la noche o en fines de semana. Además, la base de datos está diseñada para el trabajo transaccional, no para el análisis de los datos. Generalmente no puede ser en tiempo real.

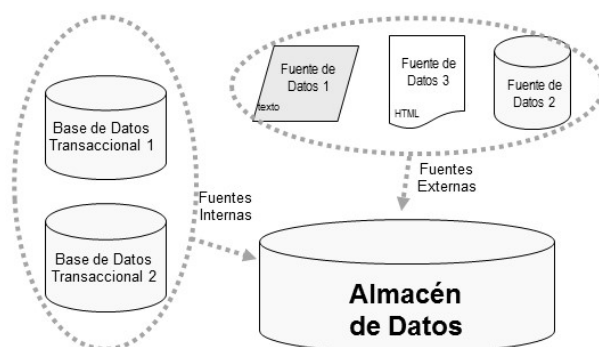
Para operar eficientemente con los datos, los costes de almacenamiento masivo y conectividad se han reducido drásticamente en los últimos años. Parece razonable recopilar los datos (información histórica) en un sistema separado y específico. Aparece así el *Data warehouse* (Almacén o Bodega de Datos).

Los esquemas siguientes muestran la definición, la motivación y los objetivos de los almacenes de datos.

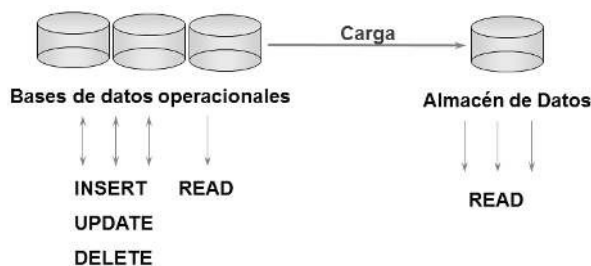


Un almacén de datos siempre está orientado hacia la información relevante de la organización. Se diseña para consultar eficientemente información relativa a las actividades (ventas, compras, producción...) básicas de la organización y no para soportar los procesos que se realizan en ella (gestión de pedidos, facturación, etc.).

Un almacén de datos integra datos recopilados de diferentes sistemas operacionales de la organización, incluyendo bases de datos transaccionales y/o fuentes externas.



Los datos en un almacén de datos son relativos a un periodo de tiempo y deben ser incrementados periódicamente. Los datos son almacenados como fotos (*snapshots*) correspondientes a periodos de tiempo. Además, los datos almacenados no son actualizados, solo son incrementados. Las operaciones de inserción, actualización y borrado de los datos se realizan en la base de datos operacional antes de que sean cargados en el almacén de datos.



Los almacenes de datos presentan múltiples ventajas para las organizaciones entre las que destacan la rentabilidad de las inversiones realizadas para su creación, el aumento de la competitividad en el mercado y el aumento de la productividad de los técnicos de dirección. Pero también presentan problemas como la infravaloración del esfuerzo necesario para su diseño y creación, la infravaloración de los recursos necesarios para la captura, la carga y el almacenamiento de los datos, el incremento continuo de los requisitos de los usuarios y la privacidad de los datos. El esquema siguiente presenta las diferencias esenciales entre una base de datos operacional y un almacén de datos.

| Sistema Operacional (OLTP) | Almacén de datos (DW) |
|---|--|
| <ul style="list-style-type: none"> - almacena datos actuales - almacena datos de detalle - bases de datos medianas (100Mb-1Gb) - los datos son dinámicos (actualizables) - los procesos (transacciones) son repetitivos - el número de transacciones es elevado - tiempo de respuesta pequeño (segundos) - dedicado al procesamiento de transacciones - orientado a los procesos de la organización - soporta decisiones diarias - sirve a muchos usuarios (administrativos) | <ul style="list-style-type: none"> - almacena datos históricos - almacena datos de detalle y datos agregados a distintos niveles - bases de datos grandes (100Gb-1Tb) - los datos son estáticos - los procesos no son previsibles - el número de transacciones es bajo o medio - tiempo de respuesta variable (segundos-horas) - dedicado al análisis de datos - orientado a la información relevante - soporta decisiones estratégicas - sirve a técnicos de dirección |

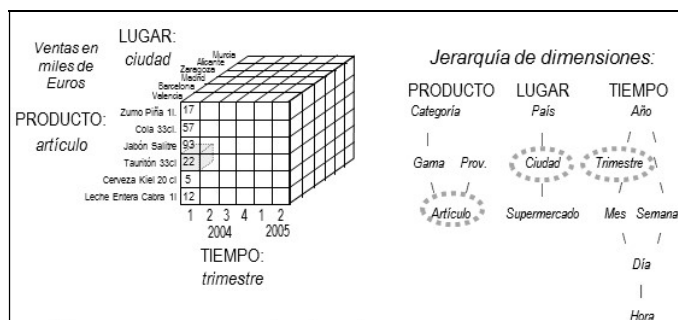
La *arquitectura* de un AD viene determinada por su situación central como fuente de información para las herramientas de análisis, tal y como se muestra en la primera ilustración de este capítulo (página 3).

Las *componentes* típicas de un almacén de datos pueden enumerarse como se indica a continuación:

- **Sistema ETL** (*Extraction, Transformation, Load*): realiza las funciones de *extracción* de las fuentes de datos (transaccionales o externas), *transformación* (limpieza, consolidación...) y la *carga* del AD, realizando:
 - extracción de los datos.
 - filtrado de los datos: limpieza, consolidación, etc.
 - carga inicial del almacén: ordenación, agregaciones, etc.
 - refresco del almacén: operación periódica que propaga los cambios de las fuentes externas al almacén de datos.
- Repositorio Propio de Datos: información relevante, metadatos.
- Interfaces y Gestores de Consulta: permiten acceder a los datos y sobre ellos se conectan herramientas más sofisticadas (OLAP, EIS, minería de datos).
- Sistemas de Integridad y Seguridad: se encargan de un mantenimiento global, copias de seguridad...

Las *herramientas de explotación* de los almacenes de datos han adoptado un **modelo multidimensional de datos**. En un esquema multidimensional se representa una actividad que es objeto de análisis (*hecho*) y las dimensiones que caracterizan la actividad (*dimensiones*). La información relevante sobre el hecho (*actividad*) se representa por un conjunto de indicadores (*medidas o atributos de hecho*). La información descriptiva de cada dimensión se representa por un conjunto de atributos (*atributos de dimensión*). Entre los atributos de una dimensión se definen *jerarquías*.

Se pueden obtener hechos a diferentes niveles de agregación. Es posible la obtención de medidas sobre los hechos parametrizadas por atributos de las dimensiones y restringidas por condiciones impuestas sobre las dimensiones. Un nivel de agregación para un conjunto de dimensiones se denomina **cubo**. El cubo que se muestra a continuación ilustra el hecho de las ventas en miles de euros de un artículo (PRODUCTO) en determinadas ciudades (LUGAR) en distintos momentos del tiempo (TIEMPO). De esta forma observamos la jerarquía de dimensiones PRODUCTO → LUGAR → TIEMPO.



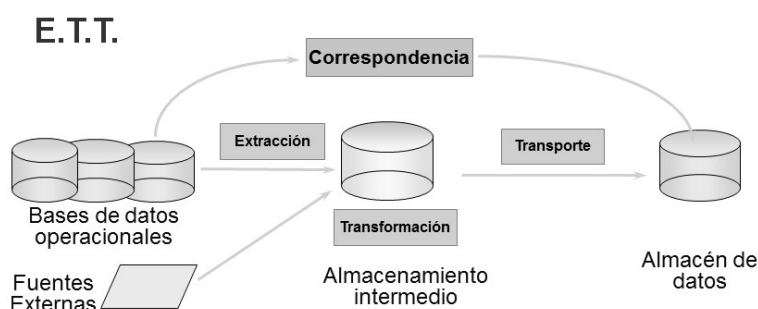
La información de un almacén de datos se recopila en varios esquemas, cada uno de los cuales se denomina **datamart**. Los datamarts se definen para satisfacer las necesidades de un departamento o sección de la organización y contienen menos información de detalle y más información agregada. El almacén de datos puede estar formado por varios datamarts y, opcionalmente, por tablas adicionales.

El sistema encargado de la carga y mantenimiento del almacén de datos es el **Sistema E.T.T.** (Extracción - Transformación - Transporte). La construcción del Sistema E.T.T. es responsabilidad del equipo de desarrollo del almacén de datos. El Sistema E.T.T. es construido específicamente para cada almacén de datos. Aproximadamente 50% del esfuerzo. En la construcción del E.T.T. se pueden utilizar herramientas del mercado o programas diseñados específicamente.

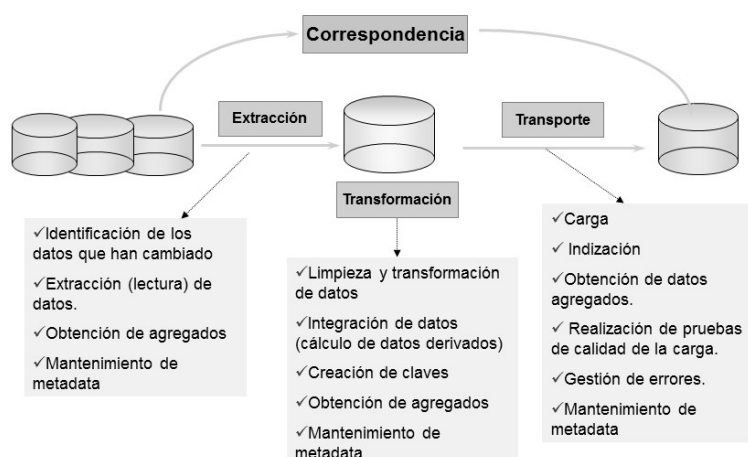
Las funciones del Sistema E.T.T. son la carga inicial (*initial load*) y el mantenimiento o *refresco* periódico: inmediato, diario, semanal, mensual... (*refreshment*).

El sistema E.T.T. es conocido también por E.T.L. (Extracción – Transformación – Load o carga).

En el proceso de carga y mantenimiento, a partir de fuentes externas de las que se realiza la extracción (que generalmente son bases de datos operacionales), se lleva a cabo el almacenamiento intermedio previo al transporte al almacén de datos. El almacenamiento intermedio permite realizar transformaciones sin paralizar las bases de datos operacionales y el almacén de datos, almacenar metadatos y facilitar la integración de fuentes externas. El esquema siguiente ilustra estas fases:



El esquema que se presenta a continuación especifica las tareas a realizar en cada una de las fases del proceso de carga y mantenimiento de un almacén de datos.



HERRAMIENTAS OLAP, ROLAP Y MOLAP

Las herramientas de OLAP presentan al usuario una visión multidimensional de los datos (esquema multidimensional) para cada actividad que es objeto de análisis.

El usuario formula consultas a la herramienta OLAP seleccionando atributos de este esquema multidimensional sin conocer la estructura interna (esquema físico) del almacén de datos. Una consulta a un almacén de datos consiste generalmente en la obtención de medidas sobre los hechos parametrizadas por atributos de las dimensiones y restringidas por condiciones impuestas sobre las dimensiones. La herramienta OLAP genera la correspondiente consulta y la envía al gestor de consultas del sistema (p.ej. mediante una sentencia SELECT).

Por ejemplo, podemos plantearnos una consulta del tipo “Importe total de las ventas durante este año de los productos del departamento **bebidas**, por **trimestre** y por **categoría**”. En este caso, las restricciones son: productos del departamento bebidas y ventas durante este año, mientras que los parámetros de la consulta **son**: por categoría de producto y por trimestre.

Se pueden presentar en forma tabular (relacional) los datos seleccionados asumiendo dos categorías en el departamento de *bebidas*: refrescos y zumos.

| Categoría | Trimestre | Ventas |
|-----------|-----------|---------|
| Refrescos | T1 | 2000000 |
| Refrescos | T2 | 1000000 |
| Refrescos | T3 | 3000000 |
| Refrescos | T4 | 2000000 |
| Zumos | T1 | 1000000 |
| Zumos | T2 | 1500000 |
| Zumos | T3 | 8000000 |
| Zumos | T4 | 2400000 |

Pero también se puede realizar una presentación matricial (multidimensional) de los datos seleccionados.

| trimestre categoria | T1 | T2 | T3 | T4 |
|------------------------|---------|---------|---------|---------|
| Refrescos | 2000000 | 1000000 | 3000000 | 2000000 |
| Zumos | 1000000 | 1500000 | 8000000 | 2400000 |

Los parámetros de la consulta (“por trimestre” y “por categoría”) determinan los criterios de agrupación de los datos seleccionados (ventas de productos del departamento *Bebidas* durante este año). La agrupación se realiza sobre dos dimensiones (Producto, Tiempo).

Pero lo interesante no es poder realizar consultas que, en cierto modo, se pueden hacer con selecciones, proyecciones, concatenaciones y agrupamientos tradicionales. Lo realmente interesante de las herramientas OLAP son sus operadores de refinamiento o manipulación de consultas como DRILL, ROLL, SLICE & DICE y PIVOT.

El carácter agregado de las consultas en el Análisis de Datos, aconseja la definición de nuevos operadores que faciliten la agregación (consolidación) y la disgregación (división) de los datos. Para la agregación tenemos el operador ROLL, que permite eliminar un criterio de agrupación en el análisis, agregando los grupos actuales. Para la desagregación tenemos el operador DRILL, que permite introducir un nuevo criterio de agrupación en el análisis, disgregando los grupos actuales.

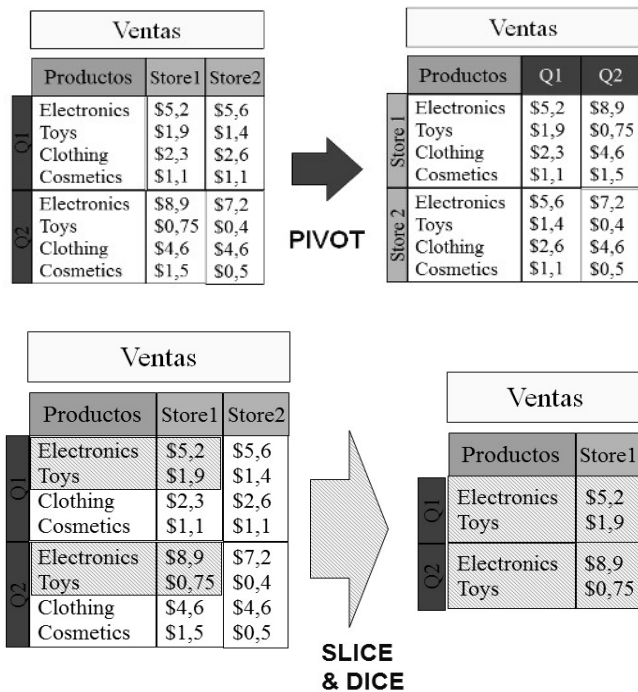
Por ejemplo, en el esquema siguiente, mediante una operación DRILL, cada grupo (categoría-trimestre) de la consulta original se disgrega en dos nuevos grupos (categoría-trimestre-ciudad) para las ciudades de León y Valencia.

| Categoría | Trimestre | Ventas | | Categoría | Trimestre | Ciudad | Ventas |
|-----------|-----------|---------|-------|-----------|-----------|----------|---------|
| Refrescos | T1 | 2000000 | drill | Refrescos | T1 | Valencia | 1000000 |
| Refrescos | T2 | 1000000 | | Refrescos | T1 | León | 1000000 |
| Refrescos | T3 | 3000000 | | Refrescos | T2 | Valencia | 400000 |
| Refrescos | T4 | 2000000 | | Refrescos | T2 | León | 700000 |
| Zumos | T1 | 1000000 | | | | | |
| Zumos | T2 | 1500000 | | | | | |
| Zumos | T3 | 8000000 | | | | | |
| Zumos | T4 | 2400000 | | | | | |

A continuación se presenta un esquema de una operación de ROLL para la agregación de categorías de productos por trimestres y ventas.

| Categoría | Trimestre | Ventas | | Categoría | Ventas |
|-----------|-----------|---------|------|-----------|----------|
| Refrescos | T1 | 2000000 | roll | Refrescos | 8000000 |
| Refrescos | T2 | 1000000 | | | |
| Refrescos | T3 | 3000000 | | | |
| Refrescos | T4 | 2000000 | | | |
| Zumos | T1 | 1000000 | | Zumos | 12900000 |
| Zumos | T2 | 1500000 | | | |
| Zumos | T3 | 8000000 | | | |
| Zumos | T4 | 2400000 | | | |

Existen otras operaciones de OLAP típicas como SLICE & DICE (para seleccionar y proyectar datos en el informe) y PIVOT (para reorientar las dimensiones en el informe).



Las herramientas de OLAP se caracterizan por ofrecer una visión multidimensional de los datos (matricial), no imponer restricciones sobre el número de dimensiones, ofrecer simetría para las dimensiones, permitir definir de forma flexible (sin limitaciones) sobre las dimensiones (restricciones, agregaciones y jerarquías entre ellas), ofrecer operadores intuitivos de manipulación (como drill-down, roll-up, slice-and-dice y pivot) y también por ser transparentes al tipo de tecnología que soporta el almacén de datos (ROLAP o MOLAP).

El almacén de datos y las herramientas OLAP se pueden basar físicamente en varias organizaciones:

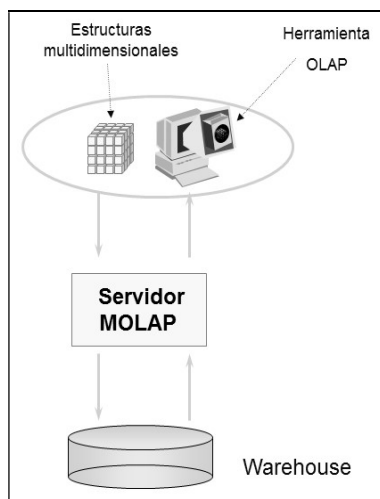
- *Sistemas ROLAP:* se implementan sobre tecnología relacional, pero disponen de algunas facilidades para mejorar el rendimiento (índices de mapas de bits, índices de JOIN).
- *Sistemas MOLAP:* disponen de estructuras de almacenamiento específicas (arrays) y técnicas de compactación de datos que favorecen el rendimiento del almacén.
- *Sistemas HOLAP:* sistemas híbridos entre los dos anteriores.

En el caso de los Sistemas ROLAP, el almacén de datos se construye sobre un SGBD Relacional. Los fabricantes de SGBD relacionales ofrecen extensiones y herramientas para poder utilizar el SGBDR como un Sistema Gestor de Almacenes de Datos. Entre estas extensiones se encuentran habitualmente los índices de mapa de bits, los índices de JOIN, técnicas de particionamiento de los datos, optimizadores de consultas y extensiones del SQL (operador CUBE, ROLL-UP, etc.).

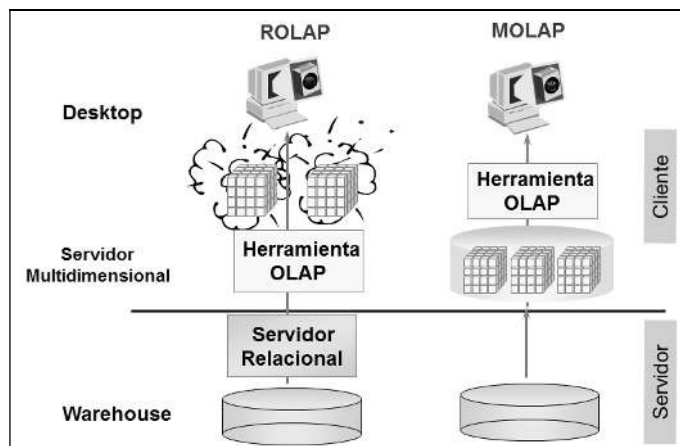
En el caso de los sistemas MOLAP, el propósito específico son las estructuras de datos (arrays) y las técnicas de compactación. El objetivo de los sistemas MOLAP es almacenar físicamente los datos en estructuras multidimensionales de forma que la representación externa y la representación interna coincidan.

El servidor MOLAP construye y almacena datos en estructuras multidimensionales. La herramienta de OLAP presenta estas estructuras multidimensionales. Los datos suelen ser arrays extraídos del almacén de datos. El almacenamiento y los procesos son muy eficientes y la complejidad de la base de datos se oculta a los usuarios. Además, el análisis se hace sobre datos agregados y métricas o indicadores precalculados.

El esquema siguiente pretende relacionar toda esta tipología de conceptos.



El diagrama siguiente muestra una comparativa entre los sistemas ROLAP y MOLAP.



Los sistemas ROLAP pueden aprovechar la tecnología relacional, pueden utilizarse sistemas relacionales genéricos (más baratos o incluso gratuitos) y el diseño lógico corresponde al físico si se utiliza el diseño de Kimball.

Los sistemas MOLAP generalmente son más eficientes que los ROLAP y presentan un coste de los cambios en la visión de los datos y de la construcción de las estructuras multidimensionales.

INFORMES OLAP CON CUBE Y ROLLUP EN SQL

Cuando tenemos que mostrar información de nuestra base de datos a los usuarios habitualmente creamos informes con Visual Basic, con Crystal Reports, con Analysis Services de Microsoft SQL Server o con cualquier lenguaje relacional de bases de datos. Con estas herramientas podemos realizar cualquier tipo de cálculo sobre los datos almacenados, y especialmente calcular sumas, totales, promedios, etc.

Pero el lenguaje SQL también nos proporciona herramientas para hacer la mayor parte del trabajo en el servidor (ahorrándonos posteriores problemas). Existen cláusulas como GROUP BY para agrupar y las funciones de agregado para contar, sumar, promediar. ¿Y si queremos calcular subtotales y totales generales en una misma consulta? Pues también existen los operadores CUBE y ROLLUP que son los que vamos a tratar aquí. Ambos operadores suelen ser parte de la cláusula GROUP BY de la sentencia SELECT, tanto en ORACLE, como en TRANSACT SQL, etc.

Vamos a trabajar con la siguiente tabla por no complicar los ejemplos ni el código SQL. Además, una tabla como esta aunque no sea real sirve perfectamente para mostrar cómo funciona WITH CUBE.

| Id | TipoTransaccion | Divisa | Cantidad |
|----|-----------------|--------|----------|
| 1 | Entrada | Euro | 200 |
| 2 | Entrada | Euro | 1300 |
| 3 | Salida | Dolar | 2000 |
| 4 | Entrada | Libra | 500 |
| 5 | Salida | Dolar | 1000 |
| 6 | Entrada | Euro | 300 |
| 7 | Entrada | Dolar | 5000 |
| 8 | Salida | Libra | 500 |
| 9 | Entrada | Euro | 700 |
| 10 | Entrada | Libra | 400 |
| 11 | Entrada | Yen | 20000 |
| 12 | Salida | Libra | 300 |
| 13 | Entrada | Euro | 4000 |
| 14 | Salida | Yen | 30000 |
| 15 | Entrada | Libra | 3000 |
| 16 | Entrada | Euro | 400 |
| 17 | Entrada | Euro | 900 |
| 18 | Salida | Dolar | 4000 |
| 19 | Entrada | Dolar | 1200 |
| 20 | Salida | Libra | 900 |
| 21 | Entrada | Euro | 2100 |
| 22 | Entrada | Libra | 200 |
| 23 | Entrada | Yen | 25000 |
| 24 | Entrada | Libra | 400 |
| 25 | Entrada | Euro | 700 |
| 26 | Entrada | NULL | 2000 |

Se trata de una tabla en la que se guarda información sobre transacciones económicas en las que tenemos tres tipos de datos. Si la transacción es de entrada o de salida, la moneda en la que se hace y la cantidad. Ahora vamos a empezar a hacer preguntas de tipo OLAP (*on-line analytical processing*) para analizar los datos que tenemos almacenados.

Podríamos preguntarnos: ¿cuántas transacciones tenemos de entrada y de salida? Para resolver esta cuestión podemos escribir una consulta sencilla con un GROUP BY y un COUNT:

```
SELECT TipoTransaccion, COUNT(IdTransaccion) Cantidad
FROM Movimientos GROUP BY TipoTransaccion
```

| TipoTransaccion | Cantidad |
|-----------------|----------|
| Entrada | 18 |
| Salida | 7 |

También podríamos preguntarnos: ¿qué divisa es la más usada? La respuesta es ahora un poco más elaborada, pero de nuevo nos basta con usar un GROUP BY. También utilizamos TOP 1 para quedarnos solo con el valor más alto después de ordenar la suma de manera descendente.

```
SELECT TOP 1 Divisa, SUM(Cantidad) Suma
FROM Movimientos GROUP BY Divisa
ORDER BY SUM(Cantidad) DESC
```

| Divisa | Suma |
|--------|-------|
| Yen | 75000 |

¿Y si queremos la cantidad de cada tipo de transacción en cada tipo de divisa? Ahora tenemos que agrupar por TipoTransaccion y Divisa para obtener la suma de las cantidades por esos conceptos. Podemos ver cuántos euros han salido o cuántos dólares han entrado.

```
SELECT TipoTransaccion, Divisa, SUM(Cantidad) Cantidad
FROM Movimientos GROUP BY TipoTransaccion, Divisa
ORDER BY TipoTransaccion
```

| TipoTransaccion | Divisa | Cantidad |
|-----------------|--------|----------|
| Entrada | Dolar | 6200 |
| Entrada | Euro | 10600 |
| Entrada | Libra | 4500 |
| Entrada | Yen | 45000 |
| Salida | Dolar | 7000 |
| Salida | Libra | 1700 |
| Salida | Yen | 30000 |

Pero podemos querer la información agrupada de más formas. Por ejemplo, para saber el total entrante, o el balance de yenes habrá que hacer cálculos adicionales o bien con nuevas consultas, o bien en nuestra aplicación cliente. Pero ahora tenemos WITH CUBE que nos permite crear nuevas dimensiones en nuestras consultas. Cuando usamos esta cláusula es como si estuviésemos haciendo a la vez todos los GROUP BY posibles y además mostrándolos en un único resultset. Añadamos WITH CUBE a la sentencia anterior.

```
SELECT TipoTransaccion, Divisa, SUM(Cantidad) Cantidad
FROM Movimientos GROUP BY TipoTransaccion, Divisa
WITH CUBE
```

| TipoTransaccion | Divisa | Cantidad |
|-----------------|--------|----------|
| Entrada | Dolar | 6200 |
| Entrada | Euro | 10600 |
| Entrada | Libra | 4500 |
| Entrada | Yen | 45000 |

BUSINESS INTELLIGENCE. TÉCNICAS, HERRAMIENTAS Y APLICACIONES

| | | |
|---------|-------|--------|
| Entrada | NULL | 66300 |
| Salida | Dolar | 7000 |
| Salida | Libra | 1700 |
| Salida | Yen | 30000 |
| Salida | NULL | 38700 |
| NULL | NULL | 105000 |
| NULL | Dolar | 13200 |
| NULL | Euro | 10600 |
| NULL | Libra | 6200 |
| NULL | Yen | 75000 |

Aquí se observan unos cuantos cambios. Primero vemos que sin necesidad de decirlo los datos se han ordenado por TipoTransaccion y dentro de TipoTransaccion por Divisa. Además, aparecen varios NULL por el medio de la tabla. Pero no os preocupéis que todo va bien y vamos a explicar este resultado con calma. Ahora cada fila es una de las posibles combinaciones de TipoTransaccion con Divisa, y las filas que contienen un NULL se tienen que leer pensando que donde está el NULL debería poner "todas". Es decir, la fila:

| | | |
|------|-------|-------|
| NULL | Dolar | 13200 |
|------|-------|-------|

indica que la cantidad total de dólares tanto en entradas como en salidas (todos los TiposTransaccipn) es 13200. La fila:

| | | |
|---------|------|-------|
| Entrada | NULL | 66300 |
|---------|------|-------|

indica que hay un valor total de 66300 para todas las entradas (es decir, para todas las divisas). Y por último la fila:

| | | |
|------|------|--------|
| NULL | NULL | 105000 |
|------|------|--------|

nos indica que el total de movimientos (todos los TipoTransaccion) de entrada y salida en cualquier divisa (todas las divisas) es de 105000.

Como vemos el NULL representa un superagregado en la columna en la que está colocado. Este tipo de NULL no lo debemos confundir con un NULL normal. Ya sabemos que un NULL normal indica que desconocemos el valor mientras que este NULL indica una agrupación.

Vamos a insertar una nueva fila en nuestra tabla:

```
INSERT INTO Movimientos (TipoTransaccion, Divisa, Cantidad)
VALUES ('Entrada', NULL, 2000)
```

¿Qué ocurre ahora si repetimos la consulta?

Hay una función llamada GROUPING que nos dice cuándo nuestro NULL es de verdad y cuándo no. Esta función nos devuelve un 1 si el nombre de la columna pasada como parámetro se usa como resumen y un 0 si no es así. Veamos un ejemplo:

```
SELECT TipoTransaccion,
       Divisa, 'Todas las Divisas'=GROUPING(Divisa),
       SUM(Cantidad) Cantidad
FROM Movimientos GROUP BY TipoTransaccion, Divisa
WITH CUBE
```

| TipoTransaccion | Divisa | Todas las Divisas | Cantidad |
|-----------------|--------|-------------------|----------|
| Entrada | NULL | 0 | 2000 |
| Entrada | Dolar | 0 | 6200 |
| Entrada | Euro | 0 | 10600 |
| Entrada | Libra | 0 | 4500 |
| Entrada | Yen | 0 | 45000 |
| Entrada | NULL | 1 | 68300 |
| Salida | Dolar | 0 | 7000 |
| Salida | Libra | 0 | 1700 |
| Salida | Yen | 0 | 30000 |
| Salida | NULL | 1 | 38700 |
| NULL | NULL | 1 | 107000 |
| NULL | NULL | 0 | 2000 |
| NULL | Dolar | 0 | 13200 |
| NULL | Euro | 0 | 10600 |
| NULL | Libra | 0 | 6200 |
| NULL | Yen | 0 | 75000 |

Se observa que hay dos tipos de NULL en la columna de divisas. Las que corresponden al último registro que insertamos que tiene un NULL en divisa, y al que la función GROUPING le asocia un 0, y el NULL que podemos traducir por "Todas las divisas" al que la función GROUPING le asocia un 1.

Ahora mezclamos estas funciones nuevas con dos funciones conocidas, CASE e ISNULL, para darle un aspecto más elegante al resultado obtenido.

```
SELECT TipoTransaccion,
       'Divisa'= CASE
                   WHEN GROUPING(Divisa)=1 THEN 'Todas'
                   ELSE ISNULL(Divisa, 'N/D')
               END,
       SUM(Cantidad) Cantidad
FROM Movimientos GROUP BY TipoTransaccion, Divisa
WITH CUBE
```

| TipoTransaccion | Divisa | Cantidad |
|-----------------|--------|----------|
| Entrada | N/D | 2000 |
| Entrada | Dolar | 6200 |
| Entrada | Euro | 10600 |

BUSINESS INTELLIGENCE. TÉCNICAS, HERRAMIENTAS Y APLICACIONES

| | | |
|---------|-------|--------|
| Entrada | Libra | 4500 |
| Entrada | Yen | 45000 |
| Entrada | Todas | 68300 |
| Salida | Dolar | 7000 |
| Salida | Libra | 1700 |
| Salida | Yen | 30000 |
| Salida | Todas | 38700 |
| NULL | Todas | 107000 |
| NULL | N/D | 2000 |
| NULL | Dolar | 13200 |
| NULL | Euro | 10600 |
| NULL | Libra | 6200 |
| NULL | Yen | 75000 |

(donde pone "N/D" pues quiere decir no disponible)

Ahora vamos a escribir la consulta que nos devuelve toda la información que podemos pedir a los datos iniciales utilizando adecuadamente WITH CUBE.

```
SELECT      'TipoTransaccion'= CASE
                                WHEN GROUPING(TipoTransaccion)=1 THEN 'Todas'
                                ELSE ISNULL(TipoTransaccion, 'N/D')
                                END,
            'Divisa'= CASE
                        WHEN GROUPING(Divisa)=1 THEN 'Todas'
                        ELSE ISNULL(Divisa, 'N/D')
                        END,
            SUM(Cantidad) Cantidad
FROM Movimientos GROUP BY TipoTransaccion, Divisa
WITH CUBE
```

| TipoTransaccion | Divisa | Cantidad |
|-----------------|--------|----------|
| Entrada | N/D | 2000 |
| Entrada | Dolar | 6200 |
| Entrada | Euro | 10600 |
| Entrada | Libra | 4500 |
| Entrada | Yen | 45000 |
| Entrada | Todas | 68300 |
| Salida | Dolar | 7000 |
| Salida | Libra | 1700 |
| Salida | Yen | 30000 |
| Salida | Todas | 38700 |
| Todas | Todas | 107000 |
| Todas | N/D | 2000 |
| Todas | Dolar | 13200 |
| Todas | Euro | 10600 |
| Todas | Libra | 6200 |
| Todas | Yen | 75000 |

Mientras que WITH CUBE genera un conjunto de resultados que muestra agregados para todas las combinaciones de valores de las columnas seleccionadas, WHIT ROLLUP genera un conjunto de resultados que muestra agregados para una jerarquía de valores de las columnas seleccionadas.

Es decir, con CUBE aparecen los resultados totalizados por TipoTransaccion, por Divisa, y por totales absolutos, mientras que con ROLLUP solo aparecerían los totales agrupados por lo que nosotros indiquemos. Veámoslo agrupando por TipoTransaccion:

```
SELECT      'TipoTransaccion'= CASE
                                WHEN GROUPING(TipoTransaccion)=1 THEN 'Todas'
                                ELSE ISNULL(TipoTransaccion, 'N/D')
                                END,
            'Divisa'= CASE
                        WHEN GROUPING(Divisa)=1 THEN 'Todas'
                        ELSE ISNULL(Divisa, 'N/D')
                        END,
            SUM(Cantidad) Cantidad
FROM Movimientos
GROUP BY TipoTransaccion, Divisa
WITH ROLLUP
```

| TipoTransaccion | Divisa | Cantidad |
|-----------------|--------|----------|
| Entrada | N/D | 2000 |
| Entrada | Dolar | 6200 |
| Entrada | Euro | 10600 |
| Entrada | Libra | 4500 |
| Entrada | Yen | 45000 |
| Entrada | Todas | 68300 |
| Salida | Dolar | 7000 |
| Salida | Libra | 1700 |
| Salida | Yen | 30000 |
| Salida | Todas | 38700 |
| Todas | Todas | 107000 |

Obtenemos menos información que con WITH CUBE pero de manera más clara. Además, muchas veces con esto será suficiente.

Si la cantidad de datos a tratar es muy grande este tipo de consultas pueden consumir muchos recursos y tiempo, pero hay muchas soluciones para que esto no sea un problema. Por ejemplo cuando tenemos una consulta que va a resumir una serie de datos y la vamos a necesitar habitualmente podemos convertirla en una vista o guardar el resultado en una tabla, y así podemos recurrir al resultado sin perder tiempo volviendo a ejecutar la sentencia SQL.

Siguiendo con nuestro ejemplo podemos almacenar el resultado que obtuvimos con el CUBE en una tabla nueva con SELECT . . . INTO.

```
SELECT      'TipoTransaccion'= CASE
                                WHEN GROUPING(TipoTransaccion)=1 THEN 'Todas'
                                ELSE ISNULL(TipoTransaccion, 'N/D')
                                END,
            'Divisa'= CASE
```

```

        WHEN GROUPING(Divisa)=1 THEN 'Todas'
        ELSE ISNULL(Divisa, 'N/D')
    END,
    SUM(Cantidad) Cantidad
INTO Resumen
FROM Movimientos GROUP BY TipoTransaccion, Divisa
WITH CUBE
select * from resultado

```

FUNCIONES DE CUBO EN EXCEL

Las **funciones de cubo en Excel** nos ayudan a obtener información de un cubo OLAP y colocar la información directamente en una hoja de Excel. De esta manera podemos combinar el potencial de las funciones de Excel y su motor de cálculo junto con los beneficios de un repositorio de datos multidimensional. Con las funciones de cubo podemos buscar datos de un cubo de OLAP como miembros, conjuntos, propiedades o valores y mezclarlos con otros cálculos y fórmulas de Excel. La tabla siguiente muestra las funciones de cubo de Excel.

| FUNCIÓN | INGLÉS | DESCRIPCIÓN |
|--------------|---------|---|
| CONJUNTOCUBO | CUBESET | <p>Define un conjunto de miembros o tuplas calculado enviando una expresión establecida al cubo del servidor, que crea el conjunto y lo devuelve a Microsoft Excel. Obtiene el valor máximo de una columna entre los registros que cumplen con los criterios establecidos</p> <p>Sintaxis:</p> <p><i>CONJUNTOCUBO(conexión, expresión_conjunto, [título], [criterio_ordenación], [ordenar_por])</i></p> <ul style="list-style-type: none"> conexión (obligatorio): La cadena de conexión al cubo. expresión_conjunto (obligatorio): La expresión MDX que nos ayudará a obtener el conjunto de miembros. título (opcional): Texto que será mostrado como el título del cubo. criterio_ordenación (opcional): Tipo de ordenación que se dará a los datos. 0 = Ninguno, 1 = Ascendente, 2 = Descendente. ordenar_por (opcional): Valor por el cual deseamos ordenar los datos. <p>Ejemplo:</p> <p>CONJUNTOCUBO("Finanzas","Order([Product].[Product].[Product Category].Members,[Measures].[Unit Sales],ASC)")</p> |

CAPÍTULO 1: INTELIGENCIA DE NEGOCIOS Y SISTEMAS DE INFORMACIÓN. INFORMES

| | | |
|------------------|------------------|---|
| MIEMBROCUBO | CUBEMEMBER | <p>Devuelve un miembro de un cubo OLAP.</p> <p>Sintaxis:</p> <p><i>MIEMBROCUBO(conexión, expresión_conjunto, [título])</i></p> <ul style="list-style-type: none"> • conexión (obligatorio): La cadena de conexión al cubo. • expresión_conjunto (obligatorio): La expresión MDX que nos ayudará a obtener el miembro del cubo. • título (opcional): Texto que será mostrado como el título del cubo. <p>Ejemplo:</p> <p>MIEMBROCUBO("Ventas","[Time].[Fiscal].[2010]")</p> |
| MIEMBROKPICUBO | CUBEKPIMEMBER | <p>Devuelve una propiedad de indicador clave de rendimiento (KPI) y muestra el nombre de KPI en la celda.</p> <p>Sintaxis:</p> <p><i>MIEMBROKPICUBO(conexión, nombre_kpi, propiedad_kpi, [título])</i></p> <ul style="list-style-type: none"> • conexión (obligatorio): La cadena de conexión al cubo. • nombre_kpi (obligatorio): Nombre del KPI en el cubo. • propiedad_kpi (obligatorio): Componente KPI devuelto. 1 = Valor real, 2 = Valor de destino. • título (opcional): Texto que será mostrado en lugar de nombre_kpi y propiedad kpi. <p>Ejemplo:</p> <p>MIEMBROKPICUBO("Ventas","MiKPIVentas",1)</p> |
| MIEMBRORANGOCUBO | CUBERANKEDMEMBER | <p>Devuelve el miembro Nth u ordenado de un conjunto.</p> <p>Sintaxis:</p> <p><i>MIEMBRORANGOCUBO(conexión, expresión_conjunto, clasificación, [título])</i></p> <ul style="list-style-type: none"> • conexión (obligatorio): La cadena de conexión al cubo. • expresión_conjunto (obligatorio): Cadena de texto con la expresión de conjunto. • clasificación (obligatorio): Valor superior que se debe devolver. 1 = Valor superior, 2 = Segundo más alto, etc. • título (opcional): Texto que será mostrado como el título del cubo. |

BUSINESS INTELLIGENCE. TÉCNICAS, HERRAMIENTAS Y APLICACIONES

| | | |
|-----------------------|--------------------|---|
| | | <p>Ejemplo:</p> <p>MIEMBRO RANGOCUBO("Ventas", \$A\$1,1)</p> |
| PROPIEDADMIEMBRO CUBO | CUBEMEMBERPROPERTY | <p>Devuelve el valor de una propiedad de miembro en el cubo.</p> <p>Sintaxis:</p> <p><i>PROPIEDADMIEMBRO CUBO(conexión, expresión_miembro, propiedad)</i></p> <ul style="list-style-type: none"> conexión (obligatorio): La cadena de conexión al cubo. expresión_miembro (obligatorio): Expresión multidimensional (MDX) de un miembro dentro del cubo. propiedad (obligatorio): Nombre de la propiedad a ser devuelta. <p>Ejemplo:</p> <p>PROPIEDADMIEMBRO CUBO("Ventas", "[Time].[Fiscal].[2010]", \$A\$1)</p> |
| RECuentoCONJUNTOCUBO | CUBESETCOUNT | <p>Devuelve el número de elementos de un conjunto.</p> <p>Sintaxis:</p> <p><i>RECuentoCONJUNTOCUBO(conjunto)</i></p> <ul style="list-style-type: none"> conjunto (obligatorio): Cadena de texto que se evalúa como un conjunto. Se puede utilizar la función CONJUNTOCUBO. <p>Ejemplo:</p> <p>RECuentoCONJUNTOCUBO(CONJUNTOCUBO("Ventas", "[Product].[All Products].Children", "Products", 1, "[Measures].[Sales Amount]"))</p> |
| VALORCUBO | CUBEVALUE | <p>Devuelve un valor agregado del cubo.</p> <p>Sintaxis:</p> <p><i>VALORCUBO(conexión, [expresión_miembro1], [expresión_miembro2], ...)</i></p> <ul style="list-style-type: none"> conexión (obligatorio): La cadena de conexión al cubo. expresión_miembro (opcional): Expresión multidimensional (MDX) que se evalúa como un miembro o tupla dentro del cubo. <p>Ejemplo:</p> <p>VALORCUBO("Ventas", "[Measures].[Profit]", "[Time].[2010]")</p> |