Diseño de un *data warehouse*

Josep Curto Díaz

PID_00238513

UOC
Universitat Oberta de Catalunya

Índice

Introducción

Como se ha comentado, el concepto y el enfoque de la inteligencia de negocio ha evolucionado bastante en los últimos años y convive con otras estrategias. Uno de los conceptos que más ha evolucionado ha sido el repositorio o almacén de datos, también conocido como *data warehouse*.

La importancia del *data warehouse* dentro del BI es máxima por ser el repositorio de la información relevante para la organización. Y de su diseño óptimo depende parte del éxito de la comprensión del rendimiento de la organización y del despliegue de un sistema de inteligencia de negocio.

En la última década han aparecido nuevos enfoques para el *data warehouse* tanto a nivel tecnológico como estratégico:

a) Aparición de bases de datos especializadas en el despliegue de *data warehouse* en forma de software o *appliances*.

b) Desarrollo de múltiples metodologías que buscan cubrir el crecimiento exponencial de datos en una organización.

c) Uso de SaaS o *in-memory*, que busca reducir el tiempo de creación de estructuras de datos.

d) Inclusión de capacidades *multi-parallel processing* (MPP) para habilitar la manipulación y el análisis de grandes volúmenes de datos estructurados.

e) Bases de datos especializadas que combinan capacidades de *data warehouse* con el uso de datos georreferenciados y/o analíticos.

El enfoque que se toma en este capítulo de introducción es el uso de una de las metodologías más frecuentes en el desarrollo de estos sistemas.

El objetivo de este módulo es introducir el concepto de *data warehouse* o almacén de datos y ejemplificar su diseño empleando una solución *open source*. Por ello, el contenido cubre desde la definición del concepto de *data warehouse*, los principales conceptos de una arquitectura de un *data warehouse*, la presentación del caso práctico y su resolución mediante las técnicas de modelización introducidas, y su cristalización usando una herramienta *open source*.

Appliance

El término hace referencia a la combinación optimizada de software y hardware para un determinado objetivo.

SaaS

Es el acrónimo de *software as a service* y hace referencia al modelo de distribución y consumo de software basado en suscripción o pago por uso empleando una arquitectura *multi-tenant* compartida.

In-memory

En este contexto, el término hace referencia a una base de datos que utiliza la memoria principal para almacenar los datos en lugar del almacenamiento en disco.

MPP

Consiste en el procesamiento coordinado de un programa por múltiples procesadores que trabajan en diferentes secciones del programa usando su propio sistema operativo y memoria.

1. El núcleo de un sistema de inteligencia de negocio: el *data warehouse*

Como ya se ha comentado, un sistema de inteligencia de negocio está formado por diferentes elementos (ETL, OLAP, *reporting*, etc.), pero de todas las piezas la principal de ellas es el *data warehouse* o almacén de datos.

Data warehouse

Según W. H. Inmon (considerado por muchos el padre del concepto), un *data warehouse* es un conjunto de datos orientados por temas, integrados, variantes en el tiempo y no volátiles, que tienen por objetivo dar soporte a la toma de decisiones. Según Ralph Kimball (considerado el principal promotor del enfoque dimensional para el diseño de almacenes de datos), un *data warehouse* es una copia de los datos transaccionales específicamente estructurada para la consulta y el análisis.

Un *data warehouse* es un repositorio de datos que proporciona una visión global, común e integrada de los datos de la organización, independiente de cómo se vayan a utilizar posteriormente por los consumidores o usuarios, con las propiedades siguientes: **estable**, **coherente**, **fiable** y con **información histórica**. Al abarcar un ámbito global de la organización y con un amplio alcance histórico, el volumen de datos puede ser muy grande (centenas de terabytes). Las bases de datos relacionales son el soporte técnico más comúnmente usado para almacenar las estructuras de estos datos y sus grandes volúmenes.

Resumiendo, el *data warehouse* presenta las siguientes características:

- **Orientado a un tema**: organiza una colección de información en torno a un tema central.
- **Integrado**: incluye datos de múltiples orígenes y presenta consistencia de datos.
- **Variable en el tiempo**: proporciona información histórica de distintos hechos de interés.
- **No volátil**: la información es persistente y solo de lectura para los usuarios finales.

Frecuentemente el *data warehouse* está constituido por una base de datos relacional, pero no es la única opción factible; también es posible considerar las bases de datos orientadas a columnas, basadas en lógica asociativa o *appliances* especializadas.

Debemos tener en cuenta que existen otros elementos en el contexto de un *data warehouse* que se combinan para poder responder a las necesidades de negocio:

a) *Data warehousing*: es el proceso de extraer y filtrar datos de las operaciones comunes de la organización, procedentes de los distintos sistemas de información operacionales y/o sistemas externos, para transformarlos, integrarlos y almacenarlos en un almacén de datos con el fin de acceder a ellos para dar soporte en el proceso de toma de decisiones de la organización.

b) *Data mart*: es un subconjunto de los datos del *data warehouse* con el objetivo de responder a un determinado análisis, función o necesidad y con una población de usuarios específica. Está pensado para cubrir las necesidades de un grupo de trabajo o de un determinado departamento dentro de la organización. Por ejemplo, un posible uso sería para la minería de datos o para la información de marketing. Al igual que en un *data warehouse*, los datos están estructurados en modelos de estrella o copo de nieve. Un *data mart* puede ser dependiente o independiente de un *data warehouse*.

c) *Operational data store*: es un tipo de almacén de datos que proporciona solo los últimos valores de los datos y no su historial; además, resulta admisible generalmente un pequeño desfase o retraso sobre los datos operacionales.

d) *Staging area*: es el sistema que permanece entre las fuentes de datos y el *data warehouse* con el objetivo de:

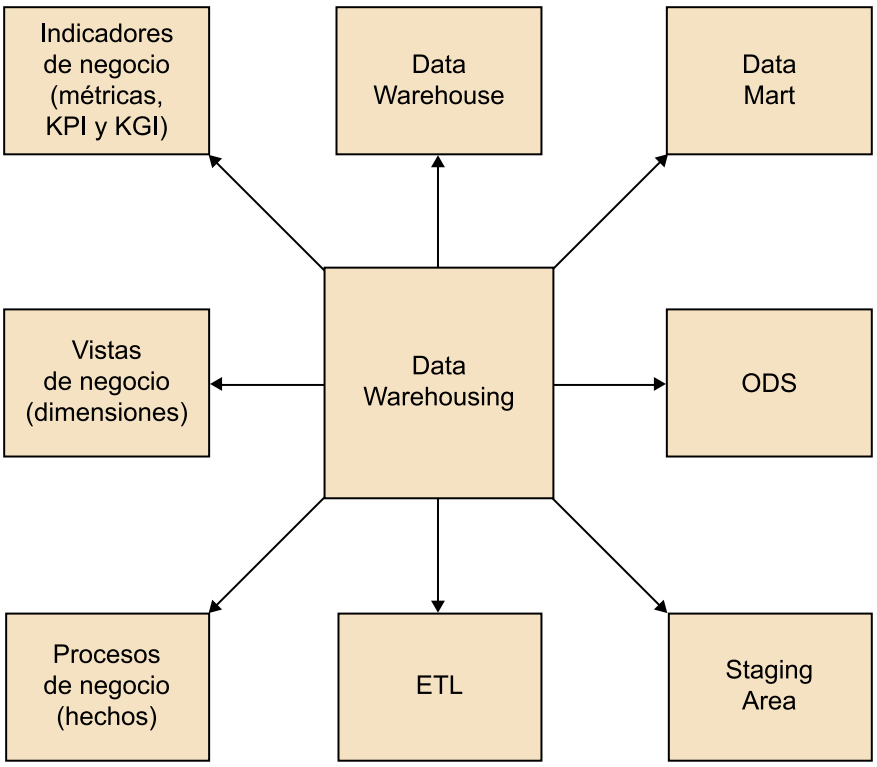
- Facilitar la extracción de datos desde fuentes de origen con una heterogeneidad y complejidad grande.
- Mejorar la calidad de datos.
- Ser usado como caché de datos operacionales con el que posteriormente se realiza el proceso de *data warehousing*.
- Acceder en detalle a información no contenida en el *data warehouse*.

e) Procesos ETL: tecnología de integración de datos basada en la consolidación de datos que se emplea tradicionalmente para alimentar almacenes de datos de cualquier tipo: *data warehouse*, *data mart*, *staging area* y ODS. Usualmente se combina con otras técnicas de consolidación de datos.

f) Metadatos: datos estructurados y codificados que describen características de instancias conteniendo informaciones para ayudar a identificar, descubrir, valorar y administrar las instancias descritas.

El siguiente gráfico resume las diferentes componentes que encontramos en el contexto del *data warehouse*.

Figura 1. Contexto del *data warehouse*



Fuente: Josep Curto.

1.1. Elementos de un *data warehouse*

La estructura relacional de una base de datos operacional sigue las formas normales en su diseño. En un *data warehouse* no debe seguirse ese patrón de diseño. La idea principal es que la información sea almacenada de manera desnormalizada para optimizar las consultas. Para ello, debemos identificar en el seno de nuestra organización los procesos de negocio, las vistas de análisis para el proceso de negocio y medidas cuantificables asociadas a estos.

Es decir, se estructura el dato en procesos de negocio, vistas de análisis y las medidas para comprender su evolución. De esta manera, hablaremos de:

1) Tabla de hecho: es la representación en el *data warehouse* de los procesos de negocio de la organización. Por ejemplo, una venta puede identificarse como un proceso de negocio, de manera que es factible, si corresponde en nuestra organización, considerar la tabla de hecho ventas.

2) Dimensión: es la representación en el *data warehouse* de una vista para un cierto proceso de negocio. Si regresamos al ejemplo de una venta, para esta tenemos el cliente que ha comprado, la fecha en la que se ha realizado,

etc. Estos conceptos pueden ser considerados como vistas para este proceso de negocio. Puede ser interesante recuperar todas las compras realizadas por un cliente. Ello nos hace entender por qué la identificamos como una dimensión.

3) Métrica: son los indicadores de un proceso de negocio; aquellos conceptos cuantificables que permiten medir nuestro proceso de negocio. Por ejemplo, en una venta tenemos su importe.

Existen principalmente dos tipos de esquemas para estructurar los datos en un almacén de datos:

a) Esquema en estrella: consiste en estructurar la información en procesos, vistas y métricas recordando a una estrella (de ahí el nombre). Desde el punto de vista del diseño, consiste en una tabla de hechos (lo que en los libros encontraremos como *fact table*) en el centro para el hecho objeto de análisis y una o varias tablas de dimensión por cada punto de vista de análisis que participa de la descripción de ese hecho. En la tabla de hecho encontramos los atributos destinados a medir (cuantificar): sus métricas. La tabla de hechos solo presenta uniones con dimensiones.

b) Esquema en copo de nieve: es un esquema de representación derivado del esquema en estrella, en el que las tablas de dimensión se normalizan en múltiples tablas. Por esta razón, la tabla de hechos deja de ser la única tabla del esquema que se relaciona con otras tablas, y aparecen nuevas uniones. Es posible distinguir dos tipos de esquemas en copo de nieve:

- **Completo**: en el que todas las tablas de dimensión en el esquema en estrella aparecen ahora normalizadas.

- **Parcial**: solo se lleva a cabo la normalización de algunas de ellas.

Es conveniente profundizar en los conceptos de tabla de hecho, dimensión y métrica.

1.2. Tipos de tabla de hecho

Una tabla de hecho es una representación de un proceso de negocio. En cuanto a diseño, es una tabla que permite guardar dos tipos de atributos diferenciados:

- Medidas del proceso/actividad/flujo de trabajo/evento que se pretende modelizar.

- Claves foráneas hacia registros en una tabla de dimensión (o en otras palabras, como ya sabemos, hacia una vista de negocio).

Existen diferentes tipos de tablas de hecho:

a) ***Transaction fact table*** **(tabla de hechos de transacciones)**: representan eventos que suceden en un determinado espacio-tiempo. Se caracterizan por permitir analizar los datos con el máximo detalle. Por ejemplo, podemos pensar en una venta que tiene como resultado métricas como el importe de esta.

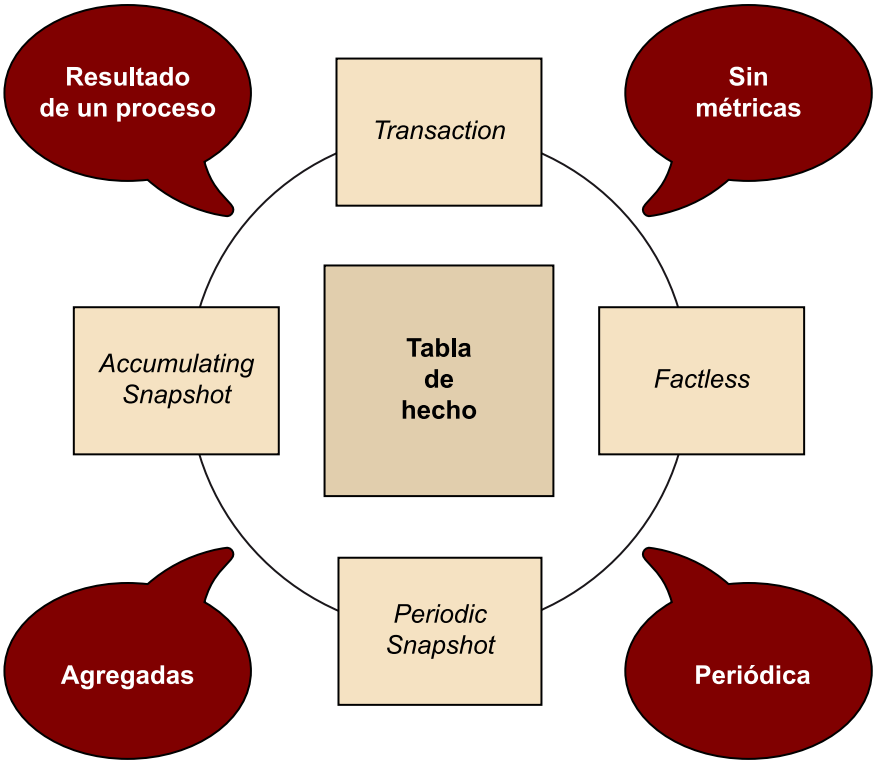
b) ***Factless fact tables/coverage table*** **(tablas de hechos sin medidas)**: son tablas que no tienen medidas y tiene sentido, dado que representan el hecho de que el evento suceda. Frecuentemente se añaden contadores a dichas tablas para facilitar las consultas SQL. Por ejemplo, podemos pensar en la asistencia a un acto benéfico en el que por cada persona que asiste tenemos un registro pero podríamos no tener ninguna métrica asociada más.

c) ***Periodic snapshot fact table*** **(tablas de hechos periódicas)**: son tablas de hecho usadas para recoger información de manera periódica a intervalos de tiempo regulares. Dependiendo de la situación medida o de la necesidad de negocio, este tipo de tablas de hecho son una agregación de las anteriores o están diseñadas específicamente. Por ejemplo, podemos pensar en el balance mensual. Los datos recogen acumulados de forma mensual.

d) ***Accumulating snapshot fact table*** **(tablas de hecho agregadas)**: representan el ciclo de vida completo de una actividad o proceso, que tiene un principio y final. Se caracterizan por presentar múltiples dimensiones relacionadas con los eventos presentes en un proceso. Por ejemplo, podemos pensar en un proceso de matriculación de un estudiante y que recopila datos durante su periodo de vida que suelen sustituir los anteriores (superación y recopilación de asignaturas, por ejemplo).

El siguiente gráfico resume las diferentes tablas de hecho que existen.

Figura 2. Tipos de tablas de hecho



Fuente: Josep Curto.

1.3. Tipos de dimensiones

SCD

Significa *slowly changing dimension* y se refiere a la política de actualización de datos en una dimensión.

Las dimensiones recogen los puntos de análisis de un hecho. Por ejemplo, una venta se puede analizar respecto al día de venta, producto, cliente, vendedor o canal de venta, entre otros. Respecto al punto de vista de la gestión histórica de los datos, las tablas de dimensiones –que se denotan por SCD– se pueden clasificar de diferentes maneras:

a) SCD Tipo 0: no se tiene en cuenta la gestión de los cambios históricos y no se realiza esfuerzo alguno. Nunca se cambia la información, ni se reescribe.

b) SCD Tipo 1: no se guardan históricos. La nueva información sobrescribe la antigua siempre. Principalmente la sobrescritura se realiza por errores de calidad de datos. Este tipo de dimensiones es fácil de mantener y son usadas cuando la información histórica no es importante.

c) SCD Tipo 2: toda la información histórica se guarda en el *data warehouse*. Cuando hay un cambio se crea una nueva entrada con su fecha y *surrogate key* apropiadas. A partir de ese momento será el valor usado para las futuras entradas. Las antiguas usarán el valor anterior.

d) SCD Tipo 3: toda la información histórica se guarda en el *data warehouse*. En este caso se crean nuevas columnas con los valores antiguos y los actuales son remplazados con los nuevos.

e) SCD Tipo 4: es lo que se conoce habitualmente como tablas históricas. Existe una tabla con los datos actuales y otra con los antiguos o los cambios.

f) SCD Tipo 6 / híbrida: combina las aproximaciones de los tipos 1, 2 y 3 (y claro, entonces 1 + 2 + 3 = 6). Consiste en considerar una dimensión de tipo 1 y añadir un par de columnas adicionales que indican el rango temporal de validez de una de las columnas de la tabla. Si bien el diseño es complejo, entre sus beneficios podemos destacar que reducen el tamaño de las consultas temporales. Existe otra variante para este tipo de dimensión, que consiste en tener versiones del registro de la dimensión (numerados de 0 a $n + 1$, donde 0 siempre es la versión actual).

Existen otros tipos de dimensiones, cuya clasificación es funcional:

1) Degeneradas: se encuentran como atributos en la tabla de hecho, si bien tiene el significado de un punto de vista de análisis. Contiene información de baja cardinalidad formada por relaciones dicotómicas. Frecuentemente, contienen solo un atributo y por ello, no se crea una tabla aparte. Por ejemplo, el sexo de un paciente.

Cardinalidad

El término hace referencia a la cantidad de registros de una tabla.

2) *Monster*: es conveniente comentar que algunas dimensiones pueden crecer desmesuradamente. Una buena práctica es romper la dimensión en dos tablas: una que contenga los valores estáticos u otra que contenga los valores volátiles. Un ejemplo claro de ello puede ser la información de cliente. Debemos ser conscientes de cuál es la información primordial de este respecto a la que solo se usa puntualmente en los informes u otros análisis.

3) *Junk*: que contiene información volátil que se utiliza puntualmente y que no se guarda de manera permanente en el *data warehouse*.

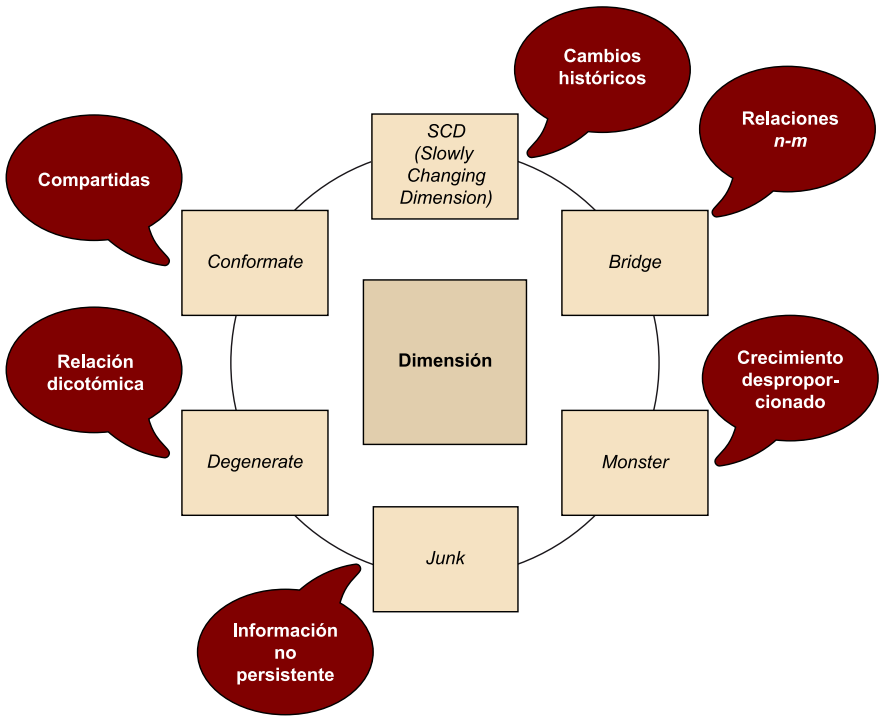
4) Conformadas: que permite compartir información entre dimensiones. Se trata de dimensiones definidas correctamente para que sean usadas por dos tablas y poder así realizar consultas comunes. El ejemplo más fácil es la dimensión temporal.

5) *Bridge* (puente): que permiten definir relaciones n a m entre tablas de hecho. Necesarias para definir, por ejemplo, la relación entre un piloto y sus múltiples patrocinadores.

6) *Role-playing* (roles): que tienen asignado un significado. Por ejemplo, podemos tener la dimensión fecha, pero también fecha de entrega.

7) Alta cardinalidad: que contienen una gran cantidad de datos difícilmente consultables en su totalidad. Por ejemplo, cada uno de los habitantes de un país.

Figura 3. Tipos de dimensiones



Fuente: Josep Curto.

1.4. Tipos de métricas

Podemos distinguir diferentes tipos de medidas, basadas en el tipo de información que recopilan, así como su funcionalidad asociada:

a) Métricas: valores que recogen el proceso de una actividad o sus resultados. Estas medidas proceden del resultado de la actividad de negocio.

- **Métricas de realización de actividad (*leading*)**: miden la realización de una actividad. Por ejemplo, la participación de una persona en un evento.

- **Métricas de resultado de una actividad (*lagging*)**: recogen los resultados de una actividad. Por ejemplo, la cantidad de puntos de un jugador en un partido.

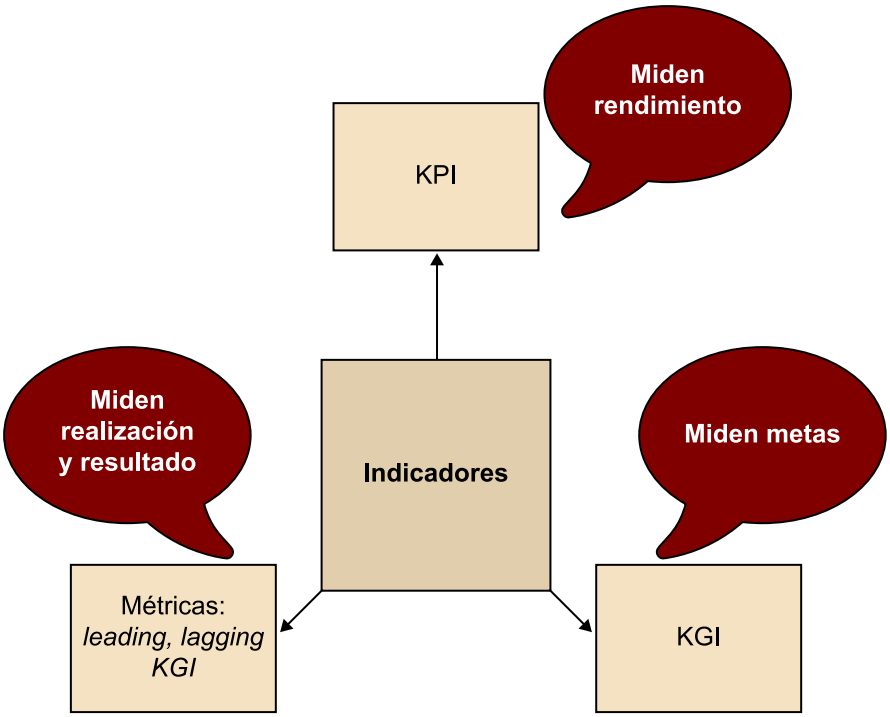
b) Indicadores clave: entendemos por indicadores clave los valores correspondientes que hay que alcanzar, y que suponen el grado de asunción de los objetivos. Estas medidas proporcionan información sobre el rendimiento de una actividad o sobre la consecución de una meta.

- ***Key performance indicator* (KPI)**: indicadores clave de rendimiento. Más allá de la eficacia, se definen unos valores que nos explican en qué rango óptimo de rendimiento nos deberíamos situar al alcanzar los objetivos. Son métricas del proceso.

- ***Key goal indicator*** **(KGI)**: indicadores de metas. Definen mediciones para informar a la dirección general si un proceso TIC ha alcanzado sus requisitos de negocio, y se expresan por lo general en términos de criterios de información.

Debemos distinguir que existen también **indicadores de desempeño**. Los indicadores clave de desempeño (en definitiva, son KPI) definen mediciones que determinan cómo de bien se está desempeñando el proceso de TI para alcanzar la meta. Son los indicadores principales que indican si será factible lograr una meta o no, y son buenos indicadores de las capacidades, prácticas y habilidades. Los indicadores de metas de bajo nivel se convierten en indicadores de desempeño para los niveles altos.

Figura 4. Tipos de métricas



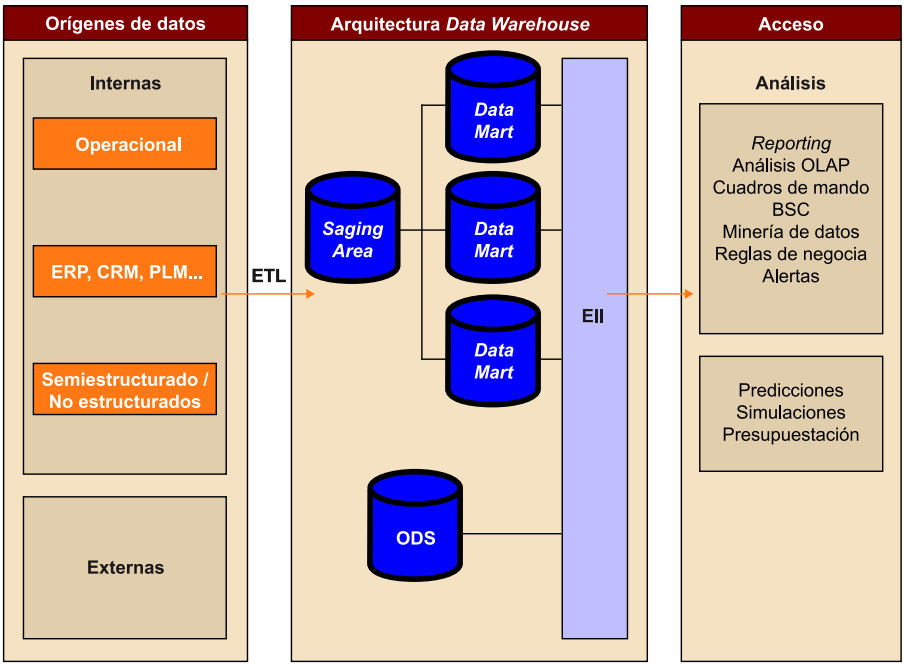
Fuente: Josep Curto.

1.5. Arquitectura de un *data warehouse*

Existen principalmente tres enfoques en la arquitectura corporativa de un *data warehouse*:

1) ***Enterprise bus architecture*** **(o** ***data warehouse*** **virtual/federado)**: también conocido como MD (*multidimensional architecture*), consiste en una arquitectura basada en *data marts* independientes federados que pueden hacer uso de una *stagingarea* en el caso de ser necesario. Federados significa que se hace uso de una herramienta EII (*enterprise information integration*) para realizar las consultas como si se tratara de un único *data warehouse*. Puede existir en el caso de ser necesario un ODS.

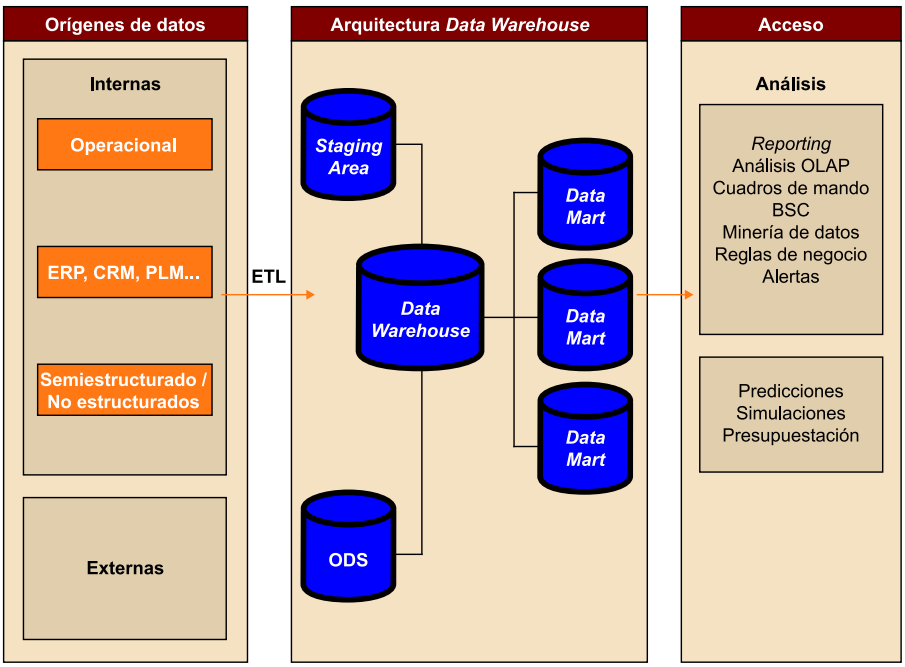
Figura 5. *Enterprise bus architecture*



Fuente: Josep Curto.

2) ***Corporate information factory*** **(o *enterprisedata warehouse*)**: consiste en una arquitectura en la que existe un *data warehouse* corporativo y unos *data marts* (o incluso cubos OLAP) dependientes de este. El acceso a datos se realiza a los *data marts* o a la ODS en caso de existir, pero nunca al propio *data warehouse*. Puede existir en el caso de ser necesaria una *staging area*.

Figura 6. *Corporate information factory*

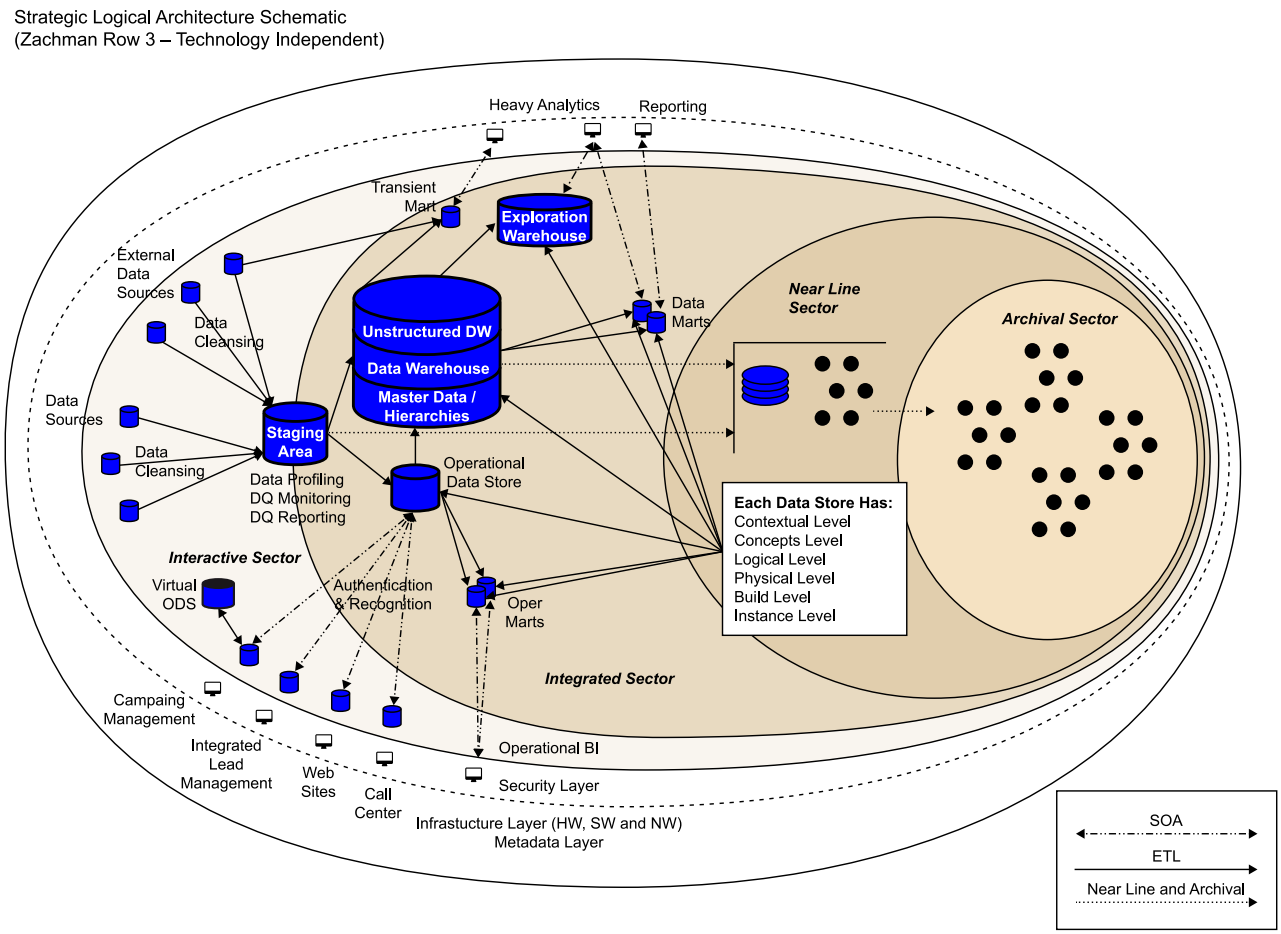


Fuente: Josep Curto.

3) ***Enterprisedata warehouse* 2.0**: consiste en la revisión de la metodología de Bill Inmon para incluir toda la experiencia de los últimos veinte años. El punto diferencial es que se separa la información por la edad de esta y la cla-

sifica por su uso. Se caracteriza por completar tanto la inclusión de información estructurada como no estructurada y por focalizarse en tener el objetivo de responder a todas las necesidades actuales de negocio. Es una propuesta para evitar que la factoría de información crezca de manera desordenada. El siguiente gráfico representa una arquitectura completa:

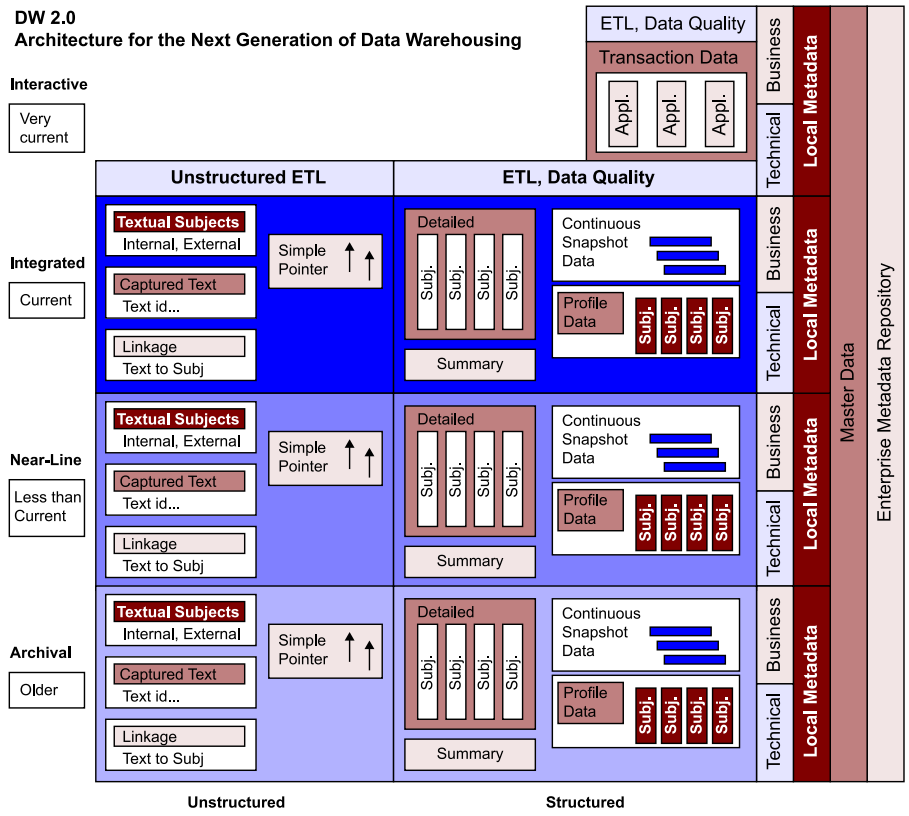
Figura 7. *Enterprisedata warehouse* 2.0



Fuente: Bill Inmon e Inmon Data Systems (2006).

Es decir, distingue diferentes entornos en función de lo actual que es la información y su uso.

Figura 8. Distinción del dato en función de su antigüedad



Fuente: Bill Inmon e Inmon Data Systems (2006).

Esta metodología aún está en proceso de despliegue en el contexto empresarial. Se combina, frecuentemente, con *Data vault*, una técnica de modelización basada en tres tipos de entidades:

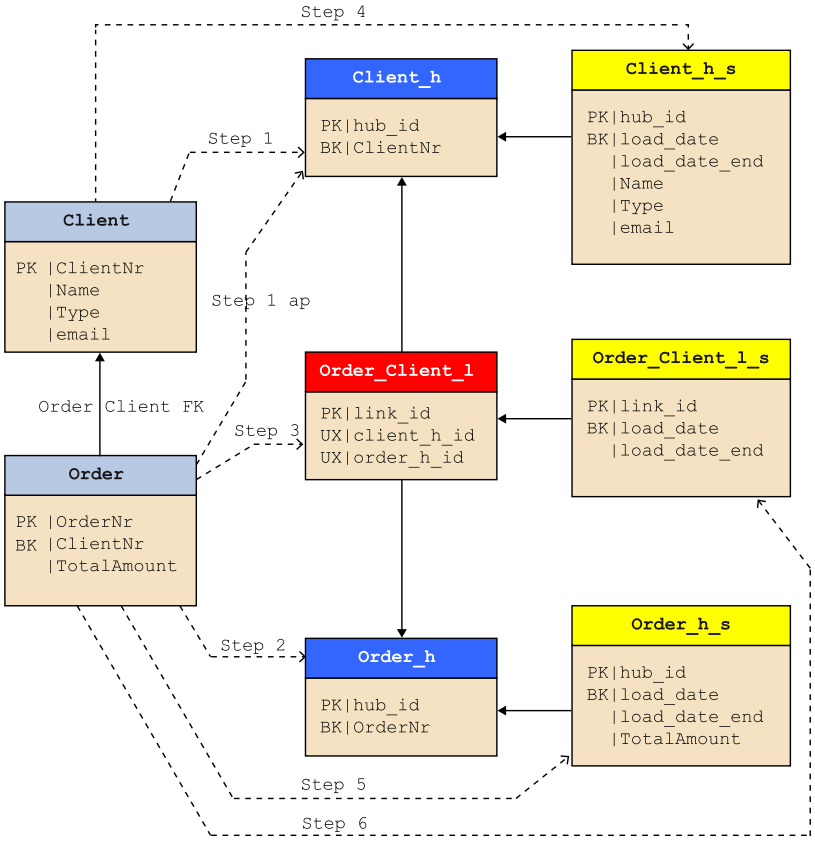
Data vault

Consiste en una técnica de modelización que busca crear un *data warehouse* capaz de adaptarse a un modelo de negocio que evoluciona con el tiempo.

- *Hubs*: contiene los indicadores claves de negocio.
- *Links*: contiene las relaciones.
- *Satellites*: contiene las descripciones.

Este tipo de diseño busca la máxima flexibilidad del *data warehouse* con el objetivo que este sea adaptable a cualquier evolución del modelo de negocio de la organización. Para ello, la información de la dimensión se encuentra en los *satellites* y podemos tener tantos como sea necesario, pudiendo extender nuestro modelo a medida que evoluciona el negocio.

Figura 9. Ejemplo de la modelización basada en *Data vault*



Fuente: Dan Lynsted.

2. Presentación de un caso práctico: análisis de estadísticas web

Actualmente la mayoría de las organizaciones tienen aplicaciones web mediante las cuales despliegan tanto soluciones de negocio como de soporte a este. Dichas aplicaciones generan ficheros de texto en los que se guardan los datos de acceso y consulta, comúnmente llamados archivos de *log*. El formato en el que están guardados estos datos y la cantidad generada de información no permite que puedan ser analizados directamente. Sin embargo, la información contenida en dichos archivos permite conocer las tendencias de los usuarios, los servicios más usados, etc.

Además, es conveniente recordar que la legislación vigente obliga a almacenar la información durante al menos un año.

El caso práctico del presente material consistirá en desarrollar un sistema de análisis de estadísticas web mediante herramientas *open source* de inteligencia de negocio. En particular, las herramientas que se usarán son:

Análisis de estadísticas web

Actualmente muchas empresas fundamentan este tipo de análisis en terceros, por ejemplo, usando Google Analytics. En este caso, ceden información a terceros para facilitar la analítica web.

- MySQL para el *data warehouse*.
- Pentaho y sus correspondientes herramientas de diseño para la solución de inteligencia de negocio.

El objetivo de este sistema es recopilar la información de los *logs* en un único repositorio de datos (el *data warehouse*) y construir una solución de negocio para el análisis consistente en informes, análisis OLAP y cuadros de mando.

A lo largo de los diferentes módulos se irá explicando progresivamente cómo construir la solución de inteligencia de negocio. El contenido de los módulos es:

- Módulo 2: Diseño de un *data warehouse*.
- Módulo 3: Diseño de procesos ETL.
- Módulo 4: Diseño de análisis OLAP.
- Módulo 5: Diseño de informes.
- Módulo 6: Diseño de cuadros de mando.

2.1. Formato de un *log*

Cualquier servidor web guarda sus *logs* conforme a una información básica común (extraída de los datos de cabecera del protocolo http). En resumen, los datos más relevantes son los siguientes:

Web de interés

Para conocer más sobre el formato de los *logs*, por ejemplo el de Apache, usado en este libro, consultad la siguiente página: Apache HTTP Server Version 2.5.

- **IP**: dirección desde la que se hace la visita.
- **Fecha/hora**: cuándo se realiza la visita.
- **URL**: dirección del recurso visitado o accedido. La URL contiene la información sobre el protocolo utilizado, el puerto, el dominio accedido, parámetros, etc.
- **Resultado**: código que indica si se tuvo éxito en el acceso o por el contrario hubo un error.
- **Tamaño**: en bytes del recurso accedido.
- **Agente de usuario**: indica por ejemplo el navegador utilizado para acceder. Este dato también puede tener codificado el sistema operativo sobre el que está ejecutándose dicho navegador.
- **URL previa**: indica la URL a partir de la cual se hizo la visita.

2.2. Necesidades de negocio

Algunas de las cuestiones de negocio a las que es posible responder a través de una herramienta de análisis de estadísticas web son las siguientes:

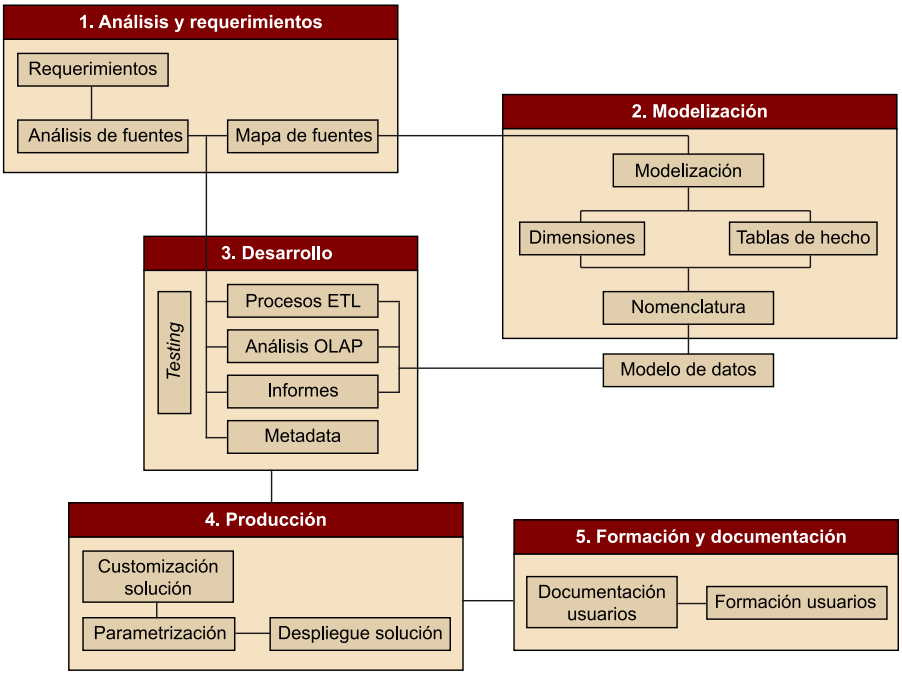
- Días más visitados.
- Horas con mayor afluencia de visitas.
- Páginas más visitadas.
- Número de visitas.
- Número de usuarios.
- Número de sesiones.
- Evolución de visitas por fecha, por respuesta del servidor, por referente, etc.

Estas necesidades se deben identificar previamente y el sistema que hemos de construir debe ser capaz de darles solución.

2.3. Fases de un sistema de BI

A lo largo de los diferentes módulos que componen este material recorreremos de manera resumida las diferentes fases de un sistema de inteligencia de negocio, si bien algunas de ellas no serán necesarias. A modo de resumen, se incluye un gráfico para tenerlas presentes.

Figura 10. Fases de un sistema de BI



Fuente: Josep Curto.

3. Resolución caso práctico con MYSQL

3.1. Modelo conceptual de datos

El modelo conceptual se basa en identificar qué tipo de procesos y vistas de negocio proporcionan la respuesta a las preguntas que tienen los usuarios finales. Normalmente en esta fase, se debe ser previsor y pensar más allá de las necesidades actuales y poder cubrir las futuras. Un buen consultor *business intelligence* conoce múltiples modelos de negocio y puede aportar opiniones clave en el diseño.

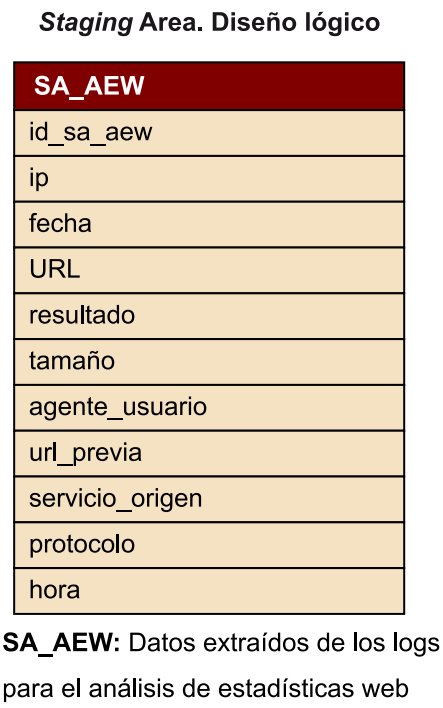
En el análisis de estadísticas web, los datos se extraen de archivos de texto que genera un servidor. Dado que las aplicaciones se despliegan en diferentes plataformas, no todos los *logs* están unificados, lo que dificulta la consolidación de la información o trabajar con el mismo nivel de detalle. Cuando sucede esto, lo habitual es crear una *staging area*. El objetivo de la creación de la *staging area* es facilitar el proceso de transformación de los datos. Si bien para el caso práctico la *staging area* no es necesaria, sus beneficios son claros:

- Realizar otro tipo de análisis *a posteriori* distinguiendo los datos mediante otros criterios, por ejemplo si el acceso ha sido realizado por robots o una amenaza (lo que se conoce como accesos no autorizados).

- Mejorar el rendimiento de la carga de datos al realizar la carga previa a la *staging area* y, *a posteriori*, realizar las transformaciones de los datos entre base de datos.

En el caso de crear una *staging area*, su modelo sería el siguiente:

Figura 11. *Staging area*



Fuente: Josep Curto.

En el momento de hacer el diseño conceptual, es necesario identificar las tablas de hecho y las dimensiones que se pueden deducir del caso práctico. Teniendo en cuenta la información que se extrae identificamos para nuestro ejemplo una tabla de hecho o proceso de negocio: la visita. Cada acceso se traduce en una visita.

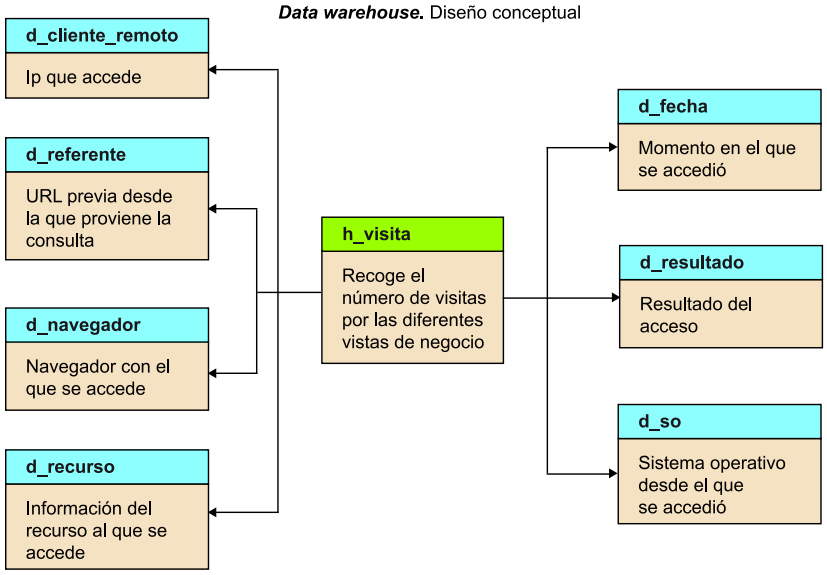
Existen otras tablas de hecho y queda como ejercicio para el lector pensar cómo afecta al modelo tener otra tabla de hecho.

Cada visita puede analizarse desde diferentes puntos de vista (lo que nos proporciona las dimensiones del proceso de negocio):

- Dimensión cliente remoto: la IP desde la que se realiza la visita.
- Dimensión referente: la URL previa desde la que se accede.
- Dimensión navegador: el navegador desde el que se realiza la visita.
- Dimensión recurso: información del recurso al que se accede.
- Dimensión fecha: momento en el que se realiza la visita.
- Dimensión resultado: resultado de la visita.
- Dimensión sistema operativo: sistema operativo desde el que se realiza la visita.

De manera que se obtiene el siguiente diseño conceptual:

Figura 12. Diseño conceptual



Fuente: Josep Curto.

3.2. Modelo lógico de datos

Después del modelo conceptual, a través del cual hemos identificado las tablas de hecho y las dimensiones, es necesario realizar el diseño lógico con el que se identifican las métricas de las tablas de hecho y los atributos de las dimensiones.

La tabla de hecho contiene la clave subrogada que identifica de manera única cada registro, las claves foráneas a las dimensiones relacionadas con la tabla de hecho y las métricas. Existen otras métricas como el tiempo de respuesta, el número de bytes servidos, el número de sesiones, etc.

Consideraremos la medida más natural en este caso práctico: el número de visitas.

Así, para la tabla de hecho h_visita tenemos:

Tabla 2. Tabla de hecho de h_visita

Tabla de hecho	Claves foráneas	Métricas
h_visita	id_os, id_resultado, id_temporal, id_recurso, id_browser, id_referer, id_cliente_remoto	Número de visitas

Y los atributos de cada una de las dimensiones son:

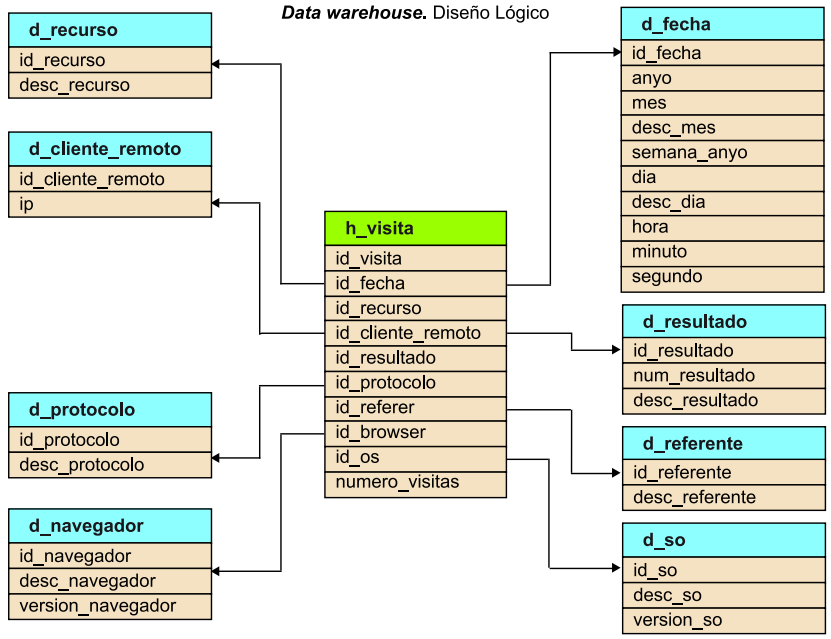
Tabla 3. Atributos de las dimensiones

Dimensión	Clave primaria	Atributos
d_recurso	id_recurso	desc_recurso
d_cliente_remoto	id_cliente_remoto	Ip

Dimensión	Clave primaria	Atributos
d_protocolo	id_protocolo	desc_protocolo
d_navegador	id_navegador	desc_navegador, versio_navegador
d_fecha	id_fecha	Anyo, mes, desc_mes, semana_anyo, dia, desc_dia, hora, minute, segundo
d_resultado	id_resultado	desc_resultado, num_resultado
d_referente	id_referente	desc_navegador, versio_navegador

Por lo que el diseño lógico resultante es el siguiente esquema en estrella:

Figura 13. Diseño lógico



Fuente: Josep Curto.

3.3. Modelo físico de datos

El siguiente paso es el diseño físico. En nuestro caso, trabajaremos con MySQL como la base de datos que será usada como *data warehouse* y con una herramienta de modelización de base de datos. En este caso vamos a utilizar MySQL Workbench.

MySQL Workbench

Se trata de una herramienta de modelización de base de datos optimizada para los diferentes motores de MySQL. Permite reingeniería inversa, lo que significa que se puede generar el esquema de base de datos a partir del modelo físico. Es la herramienta usada en este capítulo, si bien no es la única existente en el mercado.

Un *data warehouse* está formado por una colección de tablas. El objetivo es definir, para cada tabla, el formato de cada clave y atributo. Debemos apuntar los siguientes criterios:

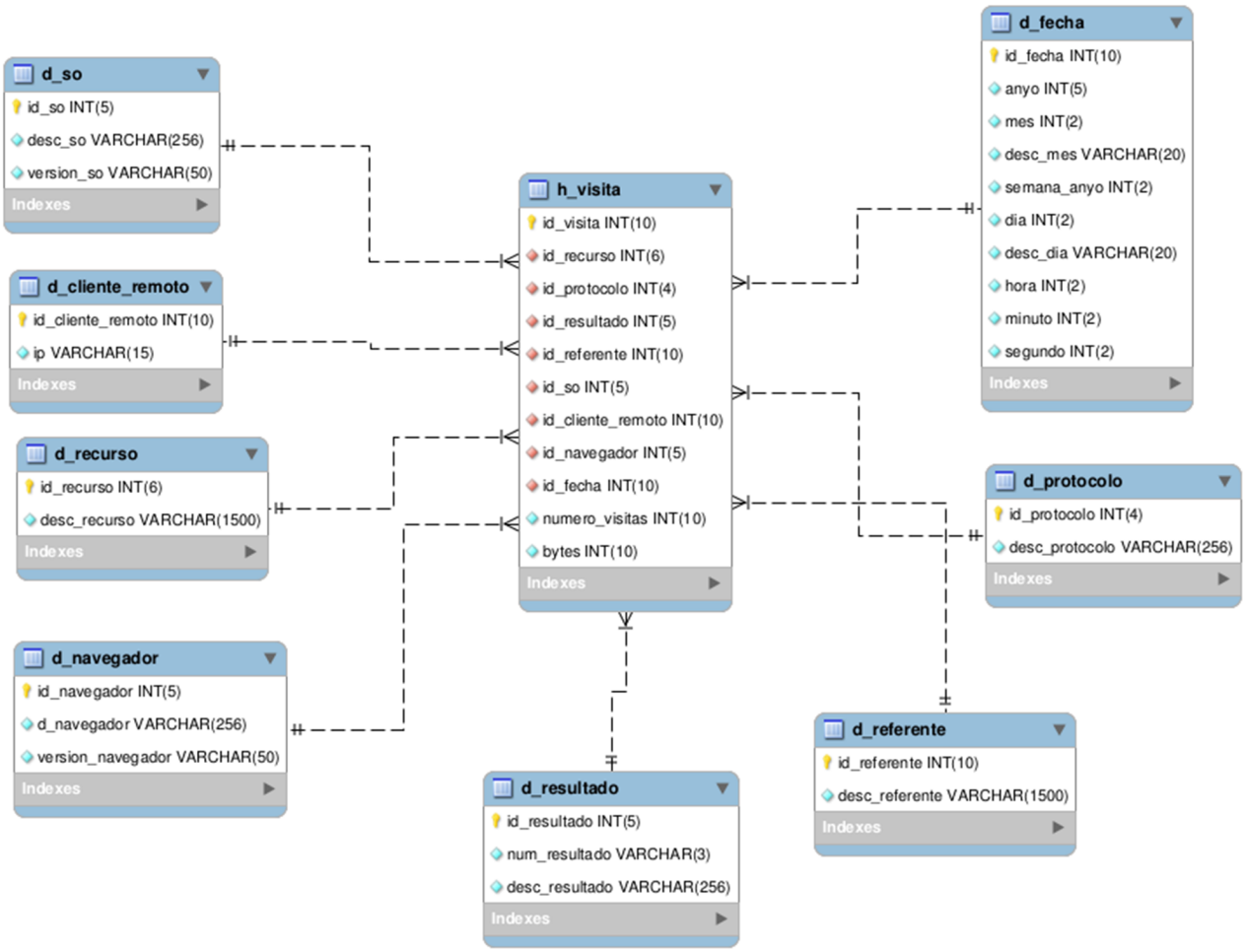
a) Se recomienda que las claves sean enteros y que sean independientes de las fuentes de origen.

b) Las métricas pueden ser aditivas (números), semiaditivas (con particularidades en el momento de acumular las cantidades) o no aditivas (entonces estamos hablando de atributos cualitativos). En las tablas de hecho deberíamos decantarnos por que todas las métricas fueran de las aditivas.

c) En un caso real, sería necesario incluir campos que permitieran la trazabilidad del dato, por ejemplo, fecha de carga, fecha de modificación, autor, fuente de origen. Para simplificar el modelo, no se incluyen.

Si consideramos cada una de las tablas y el detalle de cada uno de los atributos que la componen, entonces el diseño físico resultante es el siguiente:

Figura 14. Ejemplo de diseño físico usando MySQL Workbench



Fuente: Josep Curto.

Este diseño es un ejemplo que puede extenderse de múltiples maneras para responder a muchas más preguntas y, por lo tanto, incluir mucha más información consolidada. También es posible proponer diseños alternativos al modelo propuesto.

Abreviaturas

BI *Business intelligence*.

DSS *Decision support systems*.

EII *Enterprise information integration*.

ETL *Extract, transform and load*.

IBM *International business machines*.

KGI *Key goal indicator*.

KPI *Key performance indicator*.

ODS *Operational data store*.

OLAP *Online analytical processing*.

SaaS *Software as a service*.

SCD *Slowly changing dimension*.

SI Sistemas de información.

SQL *Structured query language*.

TI Tecnologías de la información.

URL *Uniform resource locator*.

Bibliografía

Bouman, R.; Van Dongen, J. (2009). *Pentaho® Solutions: Business Intelligence and Data Warehousing with Pentaho® and MySQL*. Indianápolis: Wiley Publishing.

Inmon, W. H. (2005). *Building the Data Warehouse* (4.ª ed.). Hoboken: John Wiley & Sons.

Inmon, W. H.; Linstedt, D. (2014). *Data Architecture: a primer for the Data Scientist*. Burlington: Morgan Kaufman Series.

Inmon, W. H.; Strauss, D.; Neushloss, G. (2008). *DW 2.0: The Architecture for the next generation of Data Warehousing*. Burlington: Morgan Kaufman Series.

Kimball, R. (2009). *Data Warehouse Toolkit Classics: The Data Warehouse Toolkit* (2.ª ed.); *The Data Warehouse Lifecycle Toolkit* (2.ª ed.); *The Data Warehouse ETL Toolkit*. Hoboken: John Wiley & Sons.