Plan General

- **Introducción al Data Warehousing.**
- **Diseño Conceptual.**
  - Conceptos generales y proceso de diseño.
  - Modelos Multidimensionales.
  - Estrategia basada en requerimientos.
  - Estrategia basada en datos.
- **Diseño Lógico.**
  - **Conceptos generales y proceso de diseño.**
  - **Diseño Lógico Multidimensional.**
  - **Diseño Lógico Relacional.**
  - **Proceso de Carga y Actualización.**
- **Aspectos Tecnológicos y Metodológicos**
  - Arquitecturas de Sistemas de DW
  - Tecnologías en DBMS.
  - Incorporación de la tecnología.
- **Conclusiones y Perspectivas.**

Diseño Lógico

Conceptos generales y proceso de diseño

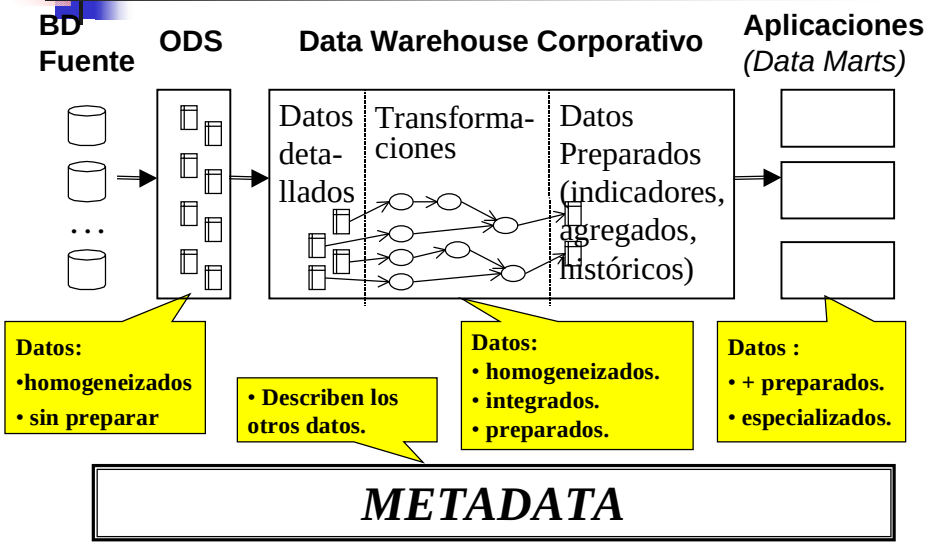
- **Temas:**
  - Introducción.
  - Encares del Diseño.
  - Estructuras Multidimensional-Relacional.

Introducción

- **Objetivos del Diseño Lógico del DW.**
  - Construir el esquema lógico del DW o DM.
    - Sobre un DBMS Relacional o Multidimensional.
  - Diseñar el Proceso de Carga y Actualización.

- **Problemas a resolver.**
  - Transformaciones de modelos y especificaciones:
    - Esquema Conceptual: abstracto.
    - Esquema Lógico: implementado.
  - Obtener estructuras adecuadas a la función.
  - Tener en cuenta la *"Carga de Trabajo"*.

Estructura del Data Warehouse



Objetivo: el DW

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

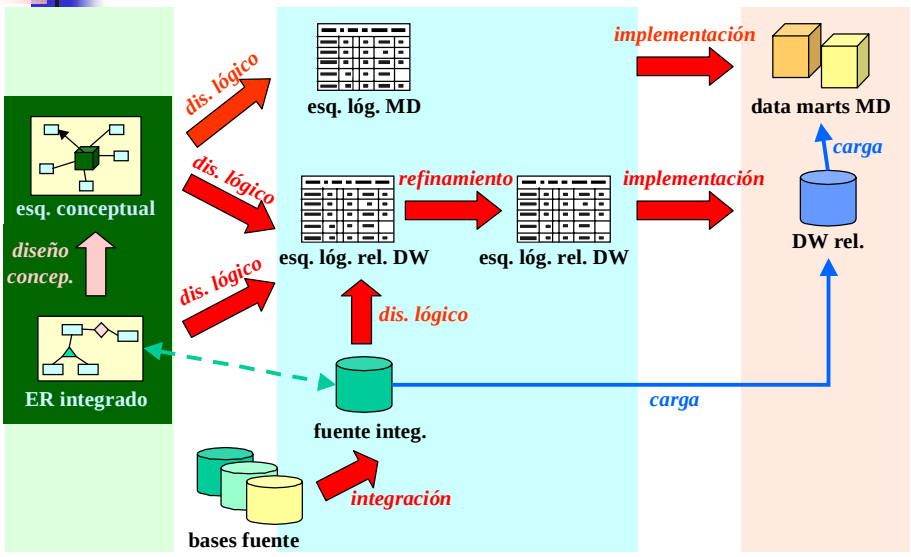
Objetivo: el proceso de carga

- **El Proceso de Carga.**
  - Grafo (o workflow) de transformaciones de datos.
    - Extracción, Filtrado, Integración, Agregación, Historización.
- **Propiedades:**
  - Consistencia del resultado:
    - Atomicidad de la operación de carga.
    - **Problema:** demasiado grande para ser transacción de BD.
    - Evitar duplicación de programas de cálculo que deben realizar la mismo.
  - Performance.
    - Debe poder ser ejecutable en una “ventana” de tiempo.
    - Aprovechar resultados de operaciones intermedias.
  - Mantenibilidad y Portabilidad.
    - Especificación clara de workflow de carga.
    - Uso de estándares.

Introducción: el Diseño Lógico

- **El proceso de Diseño Lógico:**
  - Entradas:
    - Información a representar en el DW.
      - Esquema Conceptual o especificación equivalente.
    - Características de las BD Fuente.
      - Estructura.
      - Características de los datos.
    - Información sobre *DB Load* (carga al DW).
      - O especificación equivalente asociada a reqs no funcionales.
  - Salida:
    - Esquema Lógico del DW.
    - Diseño de los Procesos de Carga.

Proceso de Diseño



Encare del Diseño (1)

- **Desde Esquema Conceptual:**
  - Conectar (acercar) la especificación Conceptual al Modelo Lógico.
    - Incorporar requerimientos no funcionales:
      - Operaciones críticas.
      - Volúmenes de datos.
  - Agregar al esquema conceptual:
    - Indicaciones orientadas a reqs no funcionales:
      - Especificación de carga sobre el DW.
      - Lineamientos de Diseño *(Design Guidelines)*.
    - Mappings a BD Fuentes.
  - Construir estructuras del DW con estos 3 elementos:
    - Esquema Conceptual + Guidelines + Mappings

Encare del Diseño (2)

- **Desde BDs Fuente:**
  - Los requerimientos se asocian sobre las BD fuente:
    - Qué datos interesa acceder y con qué performance.
  - Pasos:
    - Seleccionar información en BDs fuentes.
    - Transformar para obtener un esquema de DW con acceso eficiente.
      - Teniendo en cuenta reqs no funcionales.
  - Trabajos existentes:
    - [Kim96], [Mar00].

Encare del Diseño (3)

- **Hacia DW en BD Relacional**
  - Flexibilidad y potencial para consultas ad-hoc.
    - Especialmente consultas “por clave” y “por condición”.
  - Mala performance en Joins y Sumarizaciones.
  - Modelo ROLAP:
    - Transformación compleja de datos y operaciones MDim hacia estructuras y operaciones Relacionales.
  - Muy flexible para actualización.
    - Updates a tablas y registros.
  - Modelo estandarizado.
    - Tanto en estructura como lenguaje de manipulación (SQL).

Encare del Diseño (4)

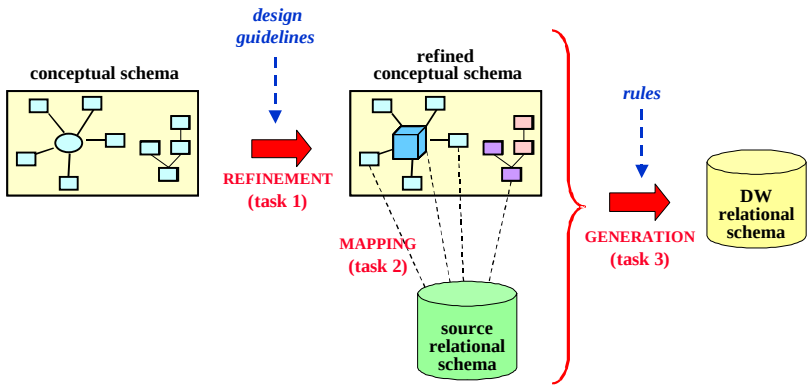
- **Hacia DW Multidimensional (OLAP).**
  - Modelo MOLAP:
    - datos y operaciones multidimensionales se mapean directamente
  - Almacenan datos en estructuras especializadas.
  - Muy alta performance en consultas dimensionales.
  - Poco usables para consultas “por clave” o “por condición”.
  - Muy poco flexibles para actualización:
    - Solo “append” a la BD (carga incremental).
  - No hay estandard.

Desde Esquema Conceptual

- **Lineamientos de Diseño**
  - **Objetivo:**
    - Complementar especificación CMDM con indicaciones para resolver reqs no funcionales.
    - Tender a procesamiento automático (CASE Tools).
  - **Tipos de Lineamientos:**
    - Materialización de Relaciones Dimensionales.
    - Fragmentación (vertical) de Dimensiones.
    - Fragmentación (horizontal) de Cubos.
  - **Referencia base:**
    - V. Peralta. *Diseño lógico de Data Warehouses a partir de esquemas conceptuales multidimensionales.* Tesis Maestría (Pedeciba - InCo).

Lineamientos de Diseño

- **En el proceso de Diseño Lógico.**

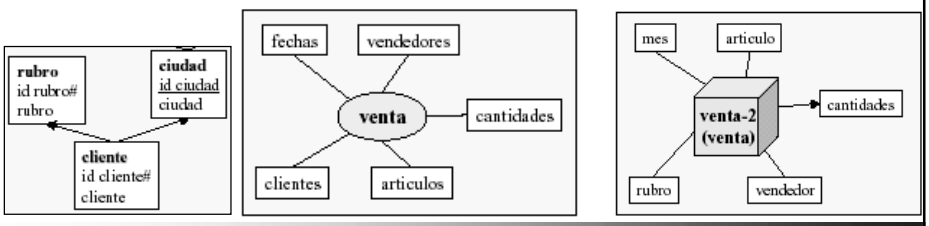


Lineamientos de Diseño

- **Materialización relaciones dimensionales.**
  - Permite indicar cubos a materializar, para evitar cálculos.
- **Fragmentación (vertical) de dimensiones.**
  - Especifica grupos de datos dimensionales a ser almacenados conjuntamente.
  - Impacto en grado de partición/unión en estructuras de datos para dimensiones.
- **Fragmentación (horizontal) de cubos.**
  - Permite almacenar por separado datos de igual esquema pero diferente uso.

Materializ de Relaciones Dimensionales

- **Introducción.**
  - En CMDM, las *relaciones dimensionales* especifican un conjunto de cruzamientos potenciales, dados por los niveles de las dimensiones.
  - A cada cruzamiento concreto se le denomina *Cubo*.
- **Definición:**
  - Un lineamiento de tipo *Materialización de Rel Dim*, consiste en la especificación de un *Cubo* indicando que se quiere almacenar materializado.
- **Ejemplo:**



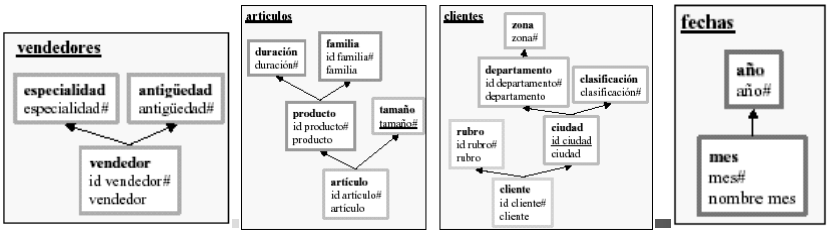
Materializ de Relaciones Dimensionales

- **Objetivos.**
  - Especificar cruzamientos con valores que se almacenan pre-calculados.
- **Criterios de uso.**
  - Aumentar performance en consultas, evitando cálculos en el momento de la operación.
- ***Trade-offs*:**

Favor	Contra
- Performance acceso Cubo.	**- Almacenamiento adicional.**

Fragmentación Vertical de Dimensiones

- **Introducción.**
  - Las *Dimensiones* consisten habitualmente en jerarquías formadas por múltiples niveles. Incluso pueden consistir en varias jerarquías alternativas.
- **Definición:**
  - Un lineamiento de tipo *Fragmentación Vertical de Dimensión* consiste en la especificación de grupos de niveles de una Dimensión a ser almacenados conjuntamente.
- **Ejemplo:**



Fragmentación Vertical de Dimensiones

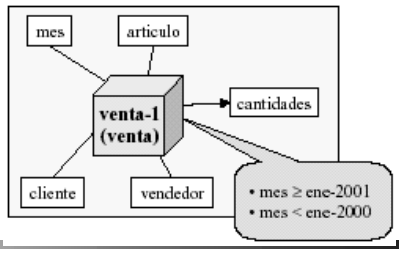
- **Objetivos.**
  - Especificar grupos de datos dimensionales a ser almacenados conjuntamente (Fragmentos).
- **Criterios de uso.**
  - Mejorar performance en acceso conjunto a *Fragmentos*.
- ***Trade-offs*:**

Favor	Contra
- Performance acceso Fragmentos.	- Eventual desnormalización.

Fragmentación Horizontal de Cubos

- **Introducción.**
  - Los Cubos tendrán asociados grandes conjuntos de datos, que comparten el mismo esquema pero pueden ser usados en forma diferente.
- **Definición.**
  - Un lineamiento de tipo *Fragmentación Horizontal de Cubo* consiste en la especificación de *bandas o franjas* correspondientes a particiones horizontales del conjunto de datos.
- **Ejemplo:**

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



Fragmentación Horizontal de Cubos

- **Objetivos.**
  - Permite almacenar por separado datos de igual esquema pero diferente uso (Fragmentos Horizontales).
- **Criterios de uso.**
  - Mejorar performance en acceso a *Fragmentos*.
- ***Trade-offs*:**

Favor	Contra
- Performance acceso Fragmentos.	**- Eventual desnormalización.**

Lineamientos: criterios de uso (1)

- **Usando los lineamientos el diseñador define:**
  - Cubos para cada RelDim.
  - Franjas para cada Cubo.
  - Fragmentos Verticales para cada Dimensión.
- **Para definirlos debe tener en consideración:**
  - Reqs de performance y almacenamiento.
  - Volúmenes de datos.
  - Funcionamiento del DBMS utilizado.
- **Materializ de Relaciones Dimensionales.**
  - Es obligatorio materializar los Cubos de menor granularidad, para no perder información.
  - Prever la materialización de datos no obtenibles de los datos detallados por problemas de sumarizabilidad.

Lineamientos: criterios de uso (2)

- **Fragmentación Horizontal de Cubos.**
  - El diseñador debe analizar dos factores:
    - Tamaño de la tabla.
    - Subconjunto de registros que son usados juntos frecuentemente.
  - Cuantas más *franjas* se definan, serán de menor tamaño y se obtendrá mejor tiempo de respuesta al consultarlas.
  - Por otro lado, si algunas consultas involucran varias franjas, estas operaciones tendrán menos performance.

Lineamientos: criterios de uso (3)

- **Fragmentación Vertical de Dimensiones.**
  - Depende de los requerimientos de consulta y browsing de las dimensiones.
  - Para obtener mejor performance en estas operaciones se tiende a mantener un único fragmento, pero esto genera valores nulos y/o redundancia.
  - Si los datos de la Dimensión cambian muy poco, entonces la redundancia no es un gran problema.
  - Si la Dimensión cambia frecuentemente y se desean registrar las diferentes versiones de los objetos, entonces la redundancia genera problemas de mantenimiento.
  - Lo recomendable es almacenar juntos los datos con igual comportamiento, tanto de los datos en sí mismos como del acceso a los mismos.

Diseño de un DW Relacional

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

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

Diseño de un DW Relacional

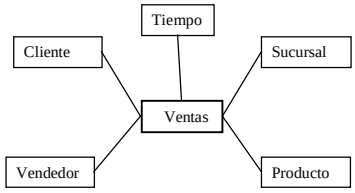
- **Vistas Materializadas**
  - Orientado a consultas SQL para mejorar su performance
  - DW:
    - conjunto de vistas materializadas
  - El problema de diseño del DW:
    - Dado:
      - consultas sobre relaciones en bases de datos fuentes
    - Salida:
      - un conjunto seleccionado de vistas a materializar de manera de mejorar la evaluación de las consultas

Diseño de un DW Relacional

- **Modelos Dimensionales**
  - Orientados a consultas OLAP
  - Se representan los conceptos multidimensionales sobre el M. Relacional .
- **Modelo Dimensional de [Kim96]**
  - Tablas de hechos (fact tables)
    - donde se guardan las ***medidas*** numéricas del negocio
    - Intersección de todas las dimensiones
    - granularidad
    - clave compuesta (la combinación de las fk)
  - Tablas de dimensión (dimension tables)
    - donde se guardan las descripciones textuales de las dimensiones del negocio
    - ***Jerarquías***: desnormalizadas o normalizadas

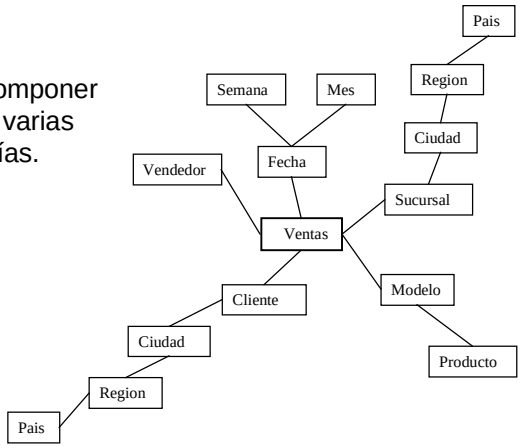
Tipos de esquemas en el MD-Rel (1)

- ***Star:* Es la estructura básica del MD**
  - 1 tabla grande central y un conjunto de tablas mas chicas organizadas alrededor de la tabla de hechos.



Tipos de esquemas en el MD-Rel (2)

- ***Snowflake:***
  - Resultado de descomponer cada dimensión en varias que forman jerarquías.



Otras opciones MD-Rel …

- ***Star-Cluster schema [MK00]***