
UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO.

ESCUELA NACIONAL DE ESTUDIOS
PROFESIONALES "A C A T L Á N"

# DATA WAREHOUSE: SOLUCIONES SOBRE TABLAS DESNORMALIZADAS.

TESINA

QUE PARA OBTENER EL TÍTULO DE:

LIC. EN MATEMÁTICAS APLICADAS Y COMPUTACIÓN.

PRESENTA:

Diana Olivia López Melchor

Asesor: Judith Jaramillo López



Abril 2003.

Tesina dedicada a:

Mis padres.

Todos mis profesores desde preprimaria hasta licenciatura.

Alberto Flores
Alberto Tovar
Alejandro García
Alfredo Lazcano
Alma Ontiveros
Adriana Bravo
Adriana Carpio
Adriana Peña
Araceli Gutiérrez
Araceli Moedano
Arturo Herrera
Asiyadeth Hernández
Bárbara Birt
Beatriz Angel
Beatriz Arroyo
Beatriz Hernández
Beatriz Vázquez
Bernardo García
Blanca Flores
Candy Valdés
Carlos Villa
César Sandoval
Citlalli Ramírez
Claudia Palma
Dalila Pérez
Damián Vázquez
Daniel Torres
Daniela Ortiz
Elizabeth Miranda
Enrique Bravo
Enrique Ríos
Erika Ortiz
Erika Rodríguez
Evelyn Cruz
Fernando Fuentes
Gerardo Victoria
Gissela Pérez
Guadalupe Meléndez
Haydeé Avendaño
Hernando Piña
Hiram Jiménez
Idalia Salazar
Iván Ibañez
Jaime Correa
Jonathán Ramírez
José Roldán
Josué Altamirano
Judith Flores
Judith Nieto
Laura Montiel
Laura Tirado
Lelia Cabrera
Leopoldo González
Leticia de la Cruz
Lourdes Alvarez
Lucía Hernández
Lucía Leal
Lucía Martínez
Luis Angeles
Luis Reyes
Manuel Pérez
Ma. Cristina Jiménez
Ma. Cristina Reyna
Ma. Isabel Barcenas
Marlenne Gualito
Marta Herrera
Maximiliano Zarco
Miguel Corona
Miguel Pérez
Miguel Ruiz
Miguel Urquiza
Miriam Reyes
Nancy Casasola
Nancy Soto
Nenetzin Campos
Olga Salas
Olga Silva
Omar Vázquez
Patricia Salinas
Pilar Arvea
Pilar Díaz
Rafael Arredondo
Raúl Gutiérrez
Ricardo del Razo
Ricardo Domínguez
Rocío Altamirano
Rocío Salas
Rogelio Sanchéz
Rosa Aguayo
Rosa Felix
Rosalinda Alcántara
Sandra Guarneros
Sandra Hernández
Tamara Sánchez
Ubaldo Acosta
Ulises Zamora
Verónica Reza
Víctor Murillo
Vidal Ruiz

A los demás:

miembros de la generación 1996-2000, amigos que hice en Acatlán, compañeros de la ENP 9.

y a **DIOS** por que sin él, no los hubiera conocido.

# Indice.

Página

Página

# Introducción.

# PAGINACION DISCONTINUA

Los motivos que propiciaron la creación de la tesina titulada "Data warehouse: soluciones sobre tablas desnormalizadas" son:

- Me gusta la materia de base de datos.
- Tengo interés por aprender nuevas tecnologías de software, relacionadas con dicha materia.

Mientras navegaba por Internet me encontré con una página web que describía a grandes rasgos las características del software **OLAP**. Me pareció un buen tema de titulación y me dedique a darle forma a mi proyecto.

Este libro está dirigido a estudiantes y profesionistas interesados en la materia de base de datos, quienes en algún momento deseen tener referencias sobre los sistemas de información gerencial y cómo dichos sistemas pueden ser desarrollados en modelos dimensionales utilizados en **ROLAP**. Mientras fue escrito se contempló que sus lectores cuenten con conocimientos básicos en el modelo entidad-relación y modelos lógicos como el modelo relacional. Es importante que el lector este familiarizado con la notación Case*Method, las formas normales y el lenguaje **SQL**.

"Data warehouse: soluciones sobre tablas desnormalizadas" tiene por objetivo describir el uso del modelo relacional desnormalizado, para la construcción de un sistema de información gerencial.

El modelo relacional desnormalizado corresponde al modelo dimensional. un conjunto de tablas desnormalizadas que representan un proceso vital para un área o una organización entera. La tabla al centro es denominada proceso y está rodeada de tablas dimensión.

La solución a los requerimientos de información a nivel gerencial parte de analizar los procesos descritos por los sistemas de transacción y las variables que los definen. Los datos arrojados al consultar un modelo dimensional son analizados para tomar decisiones administrativas, presentándose éste en tres modalidades básicas estrella, copo de nieve y constelación.

El primer capítulo expone los antecedentes de los sistemas de información gerencial y el papel del software **OLAP** como elemento clave de data warehousing. El enfoque de bases de datos antes de finales del siglo XX, desarrollaba sistemas de transacciones, registrando aquellas realizadas en un área o departamento. En la década de los 90's, surgieron los sistemas de información gerencial implementados en data warehouse para apoyar la toma de decisiones.

Aunado al data warehousing, surgió una categoría de software denominada **OLAP**, con dos versiones principales **ROLAP** basado en software relacional y **MOLAP** que utiliza bases multidimensionales. El primer caso se implementa sobre base de datos relacionales, con ayuda del modelo dimensional. El segundo sobre arreglos multidimensionales que contienen datos de interés.

La información en sistemas de información gerencial, no se encuentra en línea proviene de sistemas de transacción apoyados en software **OLTP**, siendo filtrada para su presentación en el servidor consumiéndose tiempo y recursos computacionales. Por eso un modelo dimensional es diseñado para la mayor cantidad de consultas posibles.

En el segundo capítulo se expone el modelo dimensional y estrategias de diseño para resolver requerimientos de consulta mediante el uso de tablas relacionadas desnormalizadas. Una relación de datos desnormalizada conserva redundancia de estos dentro de un modelo relacional. Una tabla desnormalizada conserva dependencias transitivas y parciales entre sus atributos. En el capítulo se explica el proceso de normalización y después la desnormalización presente en el modelo dimensional.

Cuando las tablas dimensión de un modelo dimensional son normalizadas hasta la tercera forma normal, se tiene un esquema copo de nieve. Si las tablas dimensión tienen una normalización incipiente o desnormalización se tiene un esquema estrella, que en la práctica éste es más popular que el anterior, pues reduce el número de juntas entre las tablas dimensión.

El diseño de modelos dimensionales trabaja sobre variables, cuya naturaleza obedece a su capacidad para arrojar información consistente, mientras se navega a través de una jerarquía de atributos. Si las variables son porcentajes o datos fijos a lo largo de un periodo de tiempo, las operaciones denominadas roll-up o drill-down, arrojarán datos incorrectos. Sobre un modelo dimensional se pueden diseñar agregaciones, donde las dimensiones son adaptadas para un nivel de detalle específico. En otras ocasiones trabajan coordinadamente dos modelos dimensionales lo que genera esquemas constelación.

Los modelos dimensionales son utilizados para implementar data mart o enterprise data warehouse utilizando **ROLAP**. En ocasiones se construyen esquemas que generalizan información, junto con aquellos que manejan un caso particular de los primeros generándose tablas núcleo y tablas de caracterización. También se recurre a realizar Snapshot, donde se coordinan sistemas basados en un modelo dimensional con sistemas de transacciones para aplicar un monitoreo sobre la información de un departamento y su rendimiento.

En el tercer capítulo se exponen casos sencillos enfocados a la calidad en el servicio y el contraste de normalización entre un sistema de información gerencial y un sistema de transaccciones. El primero modela el proceso de arrendamiento de autobuses, dando satisfacción al cliente al presentar soluciones acerca de puntualidad en el servicio y razones de los clientes para solicitar un reembolso. El segundo modela la tasa de ocupación de una cadena de hoteles, ya sea por hotel o por cliente. Mostrando un ejemplo para inventario de distribución. Ambos casos muestran la no aditividad de los porcentajes para un modelo dimensional. Finalmente se establecen ventajas y desventajas entre normalización y desnormalización tanto en sistemas de transacciones como en sistemas de información gerencial. Se comparan la normalización de un sistema de transacciones para una agencia de autos y la normalización/desnormalización presente en un data mart que maneja las ventas de automóviles para varias sucursales.

Resumen del capitulado por tema.[1]

Capítulo I. Data warehouse y OLAP.

1.1. Antecedentes históricos.
- 1.1.1. Información y toma de decisiones.
- 1.1.2. La necesidad de información.
- 1.1.3. Enfoques para procesamientos de datos.
- 1.1.4. Sistemas que registran transacciones.
- 1.1.5. Surgimiento de sistemas que apoyan la toma de decisiones.
- 1.1.6. Los sistemas que apoyan las decisiones.
- 1.1.7. Evolución de los sistemas que apoyan la toma de decisiones.

Se describe brevemente el uso de la información a lo largo de la historia, el surgimiento de los enfoques de procesamiento de información y los modelos de base de datos. Posteriormente se narra la incursión de los sistemas que apoyan la toma de decisiones **DSS**, exponiendo los elementos que los integran y los primeros desarrollos de estos.

1.2. Data warehouse.
- 1.2.1. La necesidad de data warehouse.
- 1.2.2. Estructura de un data warehouse.
- 1.2.3. Data warehouse y data mart.
- 1.2.4. Data warehouse e Internet.

Se establece el concepto de data warehouse, su aparición en el desarrollo de sistemas y su estructura, partiendo de que un data warehouse es un **DSS**. Se definen los conceptos de data mart y data warehouse, mostrando lo que es un data mart independiente y otro dependiente. Se expone el uso de páginas web para consultar un sistema de data warehouse por procesos o departamentos con la finalidad de navegarlas eficientemente.

1.3. Data warehousing.
- 1.3.1. Consideraciones para implementar un data warehouse.
- 1.3.2. Estrategias de desarrollo en data warehouse.
- 1.3.3. Elección de una estrategia de desarrollo adecuada.

Se define al data warehousing como el diseño y desarrollo de un sistema de data warehouse. Se exponen los pasos básicos para construir un sistema de data warehouse, sugiriéndose estrategias para su implementación.

1.4. OLAP.
- 1.4.1. Dimensionalidad.

Se define el término **OLAP**, mencionando las categorías de tecnología **OLAP** existentes. La dimensionalidad es una característica básica de dicha tecnología ya que

[1] El resumen de cada subtema queda incluido dentro del resumen correspondiente al tema que describen.

permite ver los datos de un proceso como una matriz o un cubo de datos, donde cada celda contiene un dato significativo.

1.5 Arquitectura OLAP.
- 1.51. Perspectiva funcional.
- 1.5.2. Perspectiva física.
- 1.5.3. Software intermedio.
- 1.5.4. Extracción, filtrado y presentación de datos.
- 1.5.5. Características de los servidores.
- 1.5.6. Bases de datos multidimensionales.
- 1.5.7. Arquitectura ROLAP.
- 1.5.8. Arquitectura MOLAP.
- 1.5.9. Aspectos a considerar entre MOLAP y ROLAP.
- 1.5.10. Arquitectura HOLAP.

Se explica los objetivos de una consulta sobre tecnología **OLAP** (perspectiva funcional) , la arquitectura cliente servidor que da soporte a dicha tecnología, la utilidad de software intermedio para integrar los sistemas fuente y los procesos de extracción, transformación y presentación de los datos para unificar los diferentes formatos de los sistemas fuente. Además, se describe el uso de procesamiento paralelo en el caso de **ROLAP** para minimizar el tiempo de consulta, finalmente se detallan las categorías de tecnología **OLAP** conocidas, comparando aspectos como: almacenamiento y acceso a datos, tamaño de la base de datos y actualización entre **ROLAP** Y **MOLAP**.

1.6 Funcionalidad de OLAP.

Se definen las características del software **OLAP** como son su dimensionalidad (multidimensionalidad) y el acceso compartido a datos históricos, para su análisis y posterior uso en la toma de decisiones.

1.7. Data warehouse y herramientas de consulta.
- 1.7.1. Evaluación de herramientas de data warehouse.
- 1.7.2. Cara a cara, sistemas de información gerencial y sistemas de transacciones.

Se describen las características de los tipos de herramientas basadas en tecnología **OLAP** existentes. Sugiriendo aspectos claves para evaluar su desempeño de acuerdo con las necesidades de la comunidad usuaria, finalmente se comparan las tecnologías **OLTP** y **OLAP** tomando en cuenta entre otros aspectos: enfoque, diseño de la base de datos, acceso, funciones y operaciones realizadas en los sistemas que las utilizan.

1.8. Data warehouse y OLAP.
- 1.8.1. Data warehouse en niveles.
- 1.8.2. Minería de datos.

Se establece el modelo de data warehouse en niveles en función del significado de los datos para el usuario, entre más diverso sea mayor numero de niveles de integración serán requeridos en el diseño de data warehouse. Se define el concepto de minería de

datos y como los programas mineros de datos trabajan localizando patrones de datos interesantes en sistemas que trabajan con software **OLAP**.

Capítulo II. El modelo dimensional.

El capítulo comienza con un bosquejo de los elementos que integran un modelo dimensional y su papel para implementar software **ROLAP**. Lo anterior a partir del enfoque de procesos que son descritos por medio de dimensiones.

2.1. Dimensiones.
- 2.1.1. Dimensión y Jerarquía.
- 2.1.2. Relaciones entre atributos de una dimensión.
- 2.1.3. Operaciones entre dimensiones.
- 2.1.4. Nivel de detalle.
- 2.1.5. Manipulación de datos en función del tiempo.
  - 2.1.5.1. ROLAP.
- 2.1.6. Variables y dimensiones.
- 2.1.7. Clasificación de variables.
- 2.1.8. Datos esparcidos.

Se definen los conceptos de dimensión, jerarquía, nivel de detalle y variables, ejemplificando jerarquías que corresponden a un ordenamiento total ó parcial. Se definen los tipos de relaciones entre atributos de una jerarquía y las operaciones entre dimensiones como roll-up y drill-down que permiten navegar sobre jerarquías de dimensiones. Se expone la jerarquización del tiempo y como trabajan sobre el ella el software **MOLAP** y **ROLAP**. También la clasificación de las variables de acuerdo al desempeño que muestran estas ante las operaciones de roll-up y drill-down. Se concluye con el esparcimiento de la base de datos, consistente en celdas vacías para **MOLAP** y el surgimiento de ceros innecesarios en los registros de las tablas de un modelo dimensional.

2.2 Modelo dimensional.
- 2.2.1. Tabla dimensión.
  - 2.2.1.1. Tabla de relaciones.
- 2.2.2. Tabla proceso.
- 2.2.3. Esquema estrella.
- 2.2.4. Esquema copo de nieve.
- 2.2.5. Esquema constelación.
- 2.2.6. Observaciones de la tabla proceso.
- 2.2.7. Capacidades de resumen de un esquema estrella.

Se detallan los elementos del modelo dimensional y se muestran los tipos de modelos dimensionales (esquema estrella, copo de nieve y constelación) . Se expone el uso de tablas proceso para describir el comportamiento diario de estos y la construcción de matrices de datos en **MOLAP** sobre niveles de detalle que pueden ser obtenidos de una latice de dimensiones.

2.3. Dimensiones especiales.
- 2.3.1. Dimensiones degeneradas.
- 2.3.2. Dimensiones de lento cambio.
- 2.3.3. Minidimensiones.

Para manejar números de transacciones o bien, cambios que se presentan en los datos de las dimensiones se han generado dimensiones degeneradas (atienden el primer problema), dimensiones de lento cambio y minidimensiones. Las dimensiones de lento cambio utilizan rangos de llaves primarias de las dimensiones para registrar cambios en los atributos de las dimensiones. Las minidimensiones son tablas que registran los datos susceptibles de cambios en una dimensión origen, están relacionadas con la tabla proceso y la dimensión de origen para detectar, combinaciones interesantes al consultar solamente la dimensión origen y aquellas registradas históricamente para el modelo dimensional.

2.4. Normalización.
- 2.4.1. 1FN, 2FN y 3FN.
- 2.4.2. Forma Normal de Boyce-Codd.
- 2.4.3. Cuarta forma normal.
- 2.4.5. Quinta forma normal.
- 2.4.6. Descomposición sin perdida.

Se definen las tres primeras formas normales, explicándolas a través de un ejemplo que normaliza una relación de datos. Las cuarta y quinta formas normales junto con la de Boyce-Codd son explicadas de forma teórica y su anexión puede ser considerada como breviario cultural. Finalmente se explica el objetivo de la normalización al explicar el concepto de descomposición sin pérdida, donde una relación normalizada es vuelta a construir. A lo largo del tema se definen los tipos de dependencias funcionales, multivariadas y de junta utilizadas para aplicar los criterios conocidos como formas normales.

2.5. Desnormalización.
- 2.5.1. Grados de normalización para un modelo dimensional.
  - 2.5.1.1. Modelo A.
  - 2.5.1.2. Modelo B.
    - 2.5.1.3. Modelo C.
- 2.5.2. Vistas materializadas.

A partir de un modelo base se ejemplifican gradualmente diferentes tipos de desnormalización al omitir la segunda y tercera forma normales. Permitiendo dependencias transitivas entre los atributos. Por ejemplo la tabla almacén contiene las llaves primarias y los atributos de las tablas plaza y región evitando la junta de almacén con alguna de ellas. El modelo A muestra la desnormalización de un esquema copo de nieve y el modelo C de un esquema estrella. Junto con los diagramas se presentan consultas que muestran el número de juntas de cada esquema a fin de compararlos. Las vistas materializadas funcionan para omitir procesos de extracción largos y tediosos permitiéndole a los usuarios trabajar sobre los datos más utilizados por ellos con la última actualización periódica.

2.6. Consolidación de modelos dimensionales.

Se expone la presentación de los datos con un nivel de detalle específico o bien, con todos los niveles de detalle en una vista para un análisis más detallado.

2.7. Tabla núcleo y tabla de caracterización.
2.7.1. Data mart de servicio residencial.

Cuando los procesos manejan tipos de productos y es necesario evaluarlos en algunos aspectos de forma distinta, se construye un esquema núcleo que contiene los atributos comunes para consultas generales y esquemas para consultas específicas de cada tipo denominados esquemas de caracterización. En el desarrollo del subtema se muestra el ejemplo de los esquemas estrella diseñados para analizar la información de los servicios residencial y comercial de una compañía de teléfonos.

2.8. Procesos combinados.
2.8.1. Ordenes y embarques.
2.8.2. Pagos, compromisos y presupuesto.
2.8.2.1. Presupuesto.
2.8.2.2. Compromisos.
2.8.2.3. Pagos.
2.8.3. Ceros innecesarios.
2.8.4. Semiaditividad.
2.8.5. Snapshot.

Se expone la conveniencia de combinar esquemas propios de modelos que interactúan, esto de forma conjunta como en el caso de órdenes y embarques ó separada para analizar presupuestos en base a pagos y compromisos. Además se hacen observaciones sobre los ceros innecesarios que pueden surgir por el uso de las variables presentes al combinar esquemas y como la semiaditividad afecta la combinación de procesos. Finalmente se muestra la combinación de sistemas de transacciones y sistemas de información gerencial para hacer reportes de un proceso determinado.

2.9. Agregación.
2.9.1. Pre-agregación.
2.9.2. Agregación y relaciones entre atributos.
2.9.3. Radio de compresión.
2.9.4. Métodos.
2.9.5. Ejemplo.
2.9.6. Uso posterior de datos agregados.

La agregación es un recurso utilizado en el diseño de modelos dimensionales a fin de optimizar la ejecución de las consultas, esta puede realizarse durante el proceso de extracción o a través de roll-up o drill-down antes de presentar los datos. Antes se definen las consultas que requieren agregación por la frecuencia con la que los datos han de ser consultados en un nivel de detalle (pre-agregación). Se toman en cuenta también, el radio de compresión de los datos para optimizar el espacio de almacenamiento y las relaciones entre los atributos de las dimensiones. Los datos agregados a veces son reciclados

después de un cierto tiempo o permanecen presentes en la base de datos para análisis de propósito específico.

Capítulo III. Aplicación de modelos dimensionales.

3.1. Calidad y servicio.
3.1.1. Tipos de servicios.
3.1.2. Círculo de calidad.
3.1.3. Competitividad.

3.2. Modelos y data warehousing.
3.2.1. Satisfacción del cliente.
3.2.1.1. Arrendamiento de autobuses.
3.2.1.2. Hacia la construcción del esquema.
3.2.1.3. Puntualidad.
3.2.1.4. Reembolso y cambio.

Se exponen varios modelos dimensionales construidos para satisfacer las necesidades del cliente en una compañía dedicada al arrendamiento de autobuses. Se parte de analizar las variables involucradas y la estructura del modelo base modificándolo según los requerimientos planteados por el problema hipotético planteado.

3.2.2. Inventario de distribución.
3.2.2.1. Ocupación hotelera.
3.2.2.2. Propuesta.
3.2.2.3. Enfoque detallado.

Se exponen varios modelos dimensionales para conocer la tasa de ocupación de una cadena de hoteles. Se parte de la naturaleza de las variables involucradas y se toman diversos aspectos a fin de elegir el modelo más viable para la consulta requerida.

3.3. Ventajas y desventajas de normalización entre los sistemas de transacciones y los sistemas de data warehouse.
3.3.1. Normalización vs. Desnormalización.

Para redondear el objetivo de la tesina se realizo la comparación entre la normalización presente en un sistema de transacciones y aquella que se presenta en un sistema de información gerencial. Concluyendo que mientras que la normalización es indispensable para el funcionamiento de software **OLTP**, la normalización puede presentarse en forma irrestricta en sistemas que trabajan con software **OLAP**. Debido a esto las tablas que forman un modelo dimensional pueden cumplir con la primera, segunda o tal vez a tercera forma normal a conveniencia de la consulta requerida. Desnormalizar también es una alternativa para dar solución a problemas de información usando sistemas que apoyan la toma de decisiones.

# Capítulo I.

# Data warehouse y OLAP.

**Antecedentes históricos.**

Información y toma de decisiones.

La información es uno de los bienes más preciados para cualquier organización, dado que es necesaria para tomar una decisión adecuada, oportuna y eficiente a fin de obtener ventajas sobre otras organizaciones.

Una decisión es la elección de la mejor alternativa o curso de acción ante un problema. Las grandes corporaciones como los negocios pequeños, se enfrentan a sus competidores diariamente y necesitan tener información para evaluar su desempeño y tomar decisiones que corrijan los rasgos anómalos existentes.

Sin embargo la toma de decisiones es un trabajo delegado a un grupo reducido de miembros de la organización: los gerentes, miembros de un consejo director o bien, el dueño si se trata de un negocio pequeño.

Tomar decisiones en una organización no es tarea fácil. Se necesitan reportes que otros miembros de la misma recopilan, ordenan y presentan. Puede considerarse que ordenarlos es una tarea más ardua y compleja que recopilarlos. Ante la carencia de sistemas de información, este trabajo era realizado por varias personas en forma manual con ayuda de calculadoras y máquinas de escribir, tardaba tiempo en efectuarse y los errores hacían tediosa una revisión.

A partir de la incursión de las computadoras, los especialistas en computación comenzaron a desarrollar sistemas de información, software con el propósito de generar información sobre datos relevantes para una empresa. Dicho software se apoya en un depósito de datos, cuyas características y modo de empleo, ha evolucionado a través de enfoques distintos para procesar información, incluyendo sistemas de transacciones y sistemas de información gerencial, explicados en el trascurso del capítulo presente.

La necesidad de información.

Si se hiciera una línea en el tiempo, la necesidad de transmitir información ha sido inherente a la especie humana, puesto que de ella ha dependido su sobrevivencia. Antes de descubrir la agricultura, que condujo el florecimiento de grandes civilizaciones. Los seres humanos subsistieron a través de la caza y la recolección de frutos silvestres, sin embargo para transmitir sentimientos, experiencias e ideas necesitaron de un lenguaje oral y paulatinamente de un lenguaje escrito.

El lenguaje es un código, con reglas para su interpretación, al leer decodificamos las palabras escritas para codificarlas internamente o bien, de forma externa mediante el habla. La diferencia entre el sonido y el símbolo, es que el primero se pierde con el tiempo y el segundo se preserva de generación en generación.

Los documentos escritos más antiguos que se conocen, son tablillas de barro, hechas presumiblemente por los sumerios (4000 a 3800 A. C.) para preservar y transmitir

datos acerca de la contabilidad de sus actividades agropecuarias y de los tributos pagados por sus pueblos dominados.

Desde la antigüedad, se han utilizado diversos materiales para conservar fechas importantes o datos relevantes, como pieles de animales, tejidos y roca. La información aumenta conforme las actividades humanas, se vuelven más complejas. Así evolucionó, de las cacerías descritas en las pinturas rupestres a las obras científicas y literarias de las distintas civilizaciones.

En la edad media la información fue un privilegio de la iglesia y esta se remitió a conservarla, dejando de lado el análisis, divulgación y discusión de la misma. Fueron las cruzadas y el comercio con oriente, los generadores de un cambio en el pensamiento del viejo mundo

Gracias al papel y los sellos chinos, la imprenta fue perfeccionada por Guttenberg y el conocimiento se volvió más accesible. La concepción del universo cambio y vinieron siglos de descubrimientos geográficos (XV al XVIII), provocando una sociedad más compleja, con nuevos esquemas económicos y sociales. Hacia finales del siglo XIX y durante el siglo XX, la ciencia, el arte y la cultura sufrieron una rápida evolución y aparecieron nuevos inventos: el telégrafo, el teléfono, la radio, la televisión y la computadora. Actualmente ésta última, forma parte de la mayoría de las actividades humanas, ayudando en labores de: cálculo, redacción, publicidad e incluso comunicación entre personas de distintos países.

La conjunción de estos inventos y la ampliación del comercio mundial, incrementa la necesidad de tener información adecuada, para efectuar una buena decisión. Problemática que desde la segunda mitad del siglo XX, ha motivado diversas investigaciones.

Enfoques para procesamiento de datos.

La capacidad de una computadora para realizar muchos cálculos a la vez y ejecutar bloques de instrucciones, fomento la idea de resolver problemas humanos en base a algoritmos, implementados en software.

Nadie sospechaba el capital que la computación generaría y gran parte de los avances que hoy disfrutan los usuarios de computadoras, provienen de científicos que buscaron la solución de requerimientos presentes en su entorno o que ellos vislumbraron a mediano plazo.

Uno de éstos últimos, es la creación e implementación de sistemas de información. Cuando las computadoras se hicieron accesibles a las empresas hacia 1957, trabajaban con datos leídos de forma secuencial en cintas magnéticas, era necesario revisar toda una serie de datos para encontrar el de interés, además de que no existía orden, una estructura o formato que permitiera analizarlos eficazmente. Más allá de los dispositivos utilizados para almacenaje de datos, la estructura conceptual para manipularlos se volvió la piedra angular en la solución del problema.

Entonces, surgieron dos enfoques para procesamiento de datos: el enfoque de los Sistemas manejadores de archivos y el enfoque para bases de datos. El primero, contenía la información en archivos, organizada en campos y registros; donde las aplicaciones para acceso a la información se crearon, en lenguajes como COBOL y se implementaron en software enfocado al manejo de archivos como CLIPPER y DBASE.

El enfoque de bases de datos desarrolló a través de diferentes modelos: jerárquico, red y relacional. En 1961, se dio un primer intento de **DBMS** (*Data Base Management System*) en **IDS** (*Integrated Data Store*) de GE, diseñado por Bachmann, pionero de modelos en red. Más tarde, entre los años 1965-1970, surgió **IMS** (*Information Management System*) de IBM, utilizando como lenguaje anfitrión a DL/1, ejemplo del modelo jerárquico. En esos mismos años, se genera el sistema IMS **DB/DC** (*Data Base/Data Connection*) que manejaba vistas de red superpuestas a las jerárquicas.

IMS permitió la existencia de una base de datos grande y sus desarrolladores se volvieron pioneros en el tratamiento de concurrencia, recuperación, integridad y procesamiento de consultas. Más tarde se creo Fast Path una estrategia de IMS, que permitió guardar la sección más activa de la base de datos en memoria principal.

Para 1970, E. F. Codd, dedica sus investigaciones hacia el álgebra relacional, solucionando por medio de operaciones matemáticas entre tablas relacionadas los requerimientos de consulta sobre un conjunto de datos.

Hacia 1971 **CODASYL-DBTG** (*Conference Data Base System Language*) propone una norma para las bases de datos, a fin de estandarizar el software relacionado y perfilar las investigaciones al respecto en dirección conjunta. Los sistemas comerciales siguen la propuesta pero ninguno la implementa por completo.

Hacia 1975 se diseminan investigaciones sobre bases de datos en las conferencias de **SIGMOD** (*Special Interest Group on Management of Data*) y **VLDB** (*Very Large Data Base Fundation*). Hasta que en 1976 Peter Chen, propone el modelo entidad-relación. El modelo propuesto por Chen es un modelo conceptual que puede implementarse en software de: enfoque relacional, jerárquico, o en un enfoque de red.

Las relaciones entre las entidades se transforman en las líneas conectoras entre los nodos de un árbol para el modelo jerárquico, o enlaces entre los apuntadores a los registros de un modelo de red. Sin embargo, en relaciones muchos a muchos el modelo jerárquico crea redundancia y el modelo de red implica manejar colas circulares y consultas muy elaboradas para obtener un dato.

Debido a lo anterior, el modelo relacional resulto ser más sencillo. Desarrollándose varios proyectos de investigación a este respecto como: System R (IBM), INGRES (University of California, Berkeley), System 2000 (University of Texas, Austin), Proyecto Socrate (Universidad de Grenoble, Francia) y ADABAS (Universidad Técnica de Darmstadt, Alemania).

Comenzaron a surgir especialistas en tecnologías de información, científicos e ingenieros dedicados a analizar, el flujo de información como parte de la interacción entre los componentes de un sistema. [Libro # 15] y [Libro # 4].

Sistemas que registran transacciones.

Los gerentes desde mediados de los 70´s, recurrían a los especialistas en tecnología de información para que solucionarán sus necesidades, su número era limitado y todavía no se dependía de los Sistemas de Información. Los mainframe, se concentraron en tareas básicas para base de datos: inserción o modificación de datos y reportes esporádicos sobre la gestión de un área o departamento. La presentación de los datos era sencilla y las aplicaciones eran poco amigables para el usuario.

Era una época de experimentación y los fabricantes de software, tomaron la decisión de estandarizar los requerimientos de los usuarios, con un enfoque dirigido a registrar las transacciones de los departamentos de una empresa, lo que dio origen a los sistemas de transacciones u operacionales. La justificación de estos sistemas, descansaba en un incremento de velocidad en las transacciones y la precisión con que estas eran ejecutadas.

En los 80's, los mainframe son reemplazados por la arquitectura cliente servidor y el software relacional se hace popular. Dicho software, administra todas las operaciones sobre las tablas del sistema de información, con un subconjunto especial de tablas que sirve como diccionario de datos y un lenguaje de consulta estructurado **SQL** (*Structured Query Language*), donde las operaciones definidas por Codd, pueden realizarse en bloque de sentencias. El software relacional permitió la incursión de **RDBMS** (*Relational Data Base Management System*) o visión relacional de un DBMS, idea acaecida desde los 60's, para administrar de forma centralizada la información de un sistema de información.

A mediados de esa década, el modelo relacional, parecía satisfacer un gran número de requerimientos de usuarios finales. Los profesionales en tecnologías de información pudieron modelar dichas necesidades y estructurar una base de datos que las satisficiera.

Sin embargo, la tendencia de hacer sistemas de información para cada departamento, había generado inconsistencias en el momento que los datos eran intercambiados entre los mismos. Haciendo patente la necesidad de un enfoque que se apegará a las necesidades de los tomadores de decisiones. [Libro # 4]

Surgimiento de sistemas que apoyan la toma de decisiones.

Las escuelas de negocios, comenzaron a tener acceso a los sistemas de tiempo compartido o *mainframe*, surgiendo la necesidad de sistemas que apoyaran las decisiones administrativas; los estudiantes y profesores de las mismas, documentaron los requerimientos de dichos sistemas; tal es el caso de HEC (Francia,1967) y el proyecto MAC para Sloan School.

Antes en los años 50´s y 60´s, el trabajo teórico sobre los **DSS** (*Decision Support System*) comenzó a desarrollarse en Massachussets Institute of Technology. Para 1971,

los sistemas que apoyan la toma de decisiones tuvieron un predecesor, **MDS** (*Management Decision System*) un **DSS** experimental, generado por M.Scott Morton, alumno del MIT, utilizado para coordinar las áreas de mercadotecnia y producción de un equipo de lavandería.

A mediados de la década de los 70's, el **DSS** Brandaid, fue utilizado para apoyar las decisiones de publicidad, destacándose por su simplicidad y reportes detallados. En la misma década Kevin Iverson creo APL, acrónimo de "A program Language" implantado por IBM que debido a su complejidad tuvo poca aceptación.

A partir de la aparición de dichos prototipos, se desarrollaron proyectos para solucionar las necesidades de los usuarios que requerían reportes más estructurados que los generados por sistemas de transacciones, estos últimos muestran la información en un periodo reciente, por departamento sin definir operaciones que permitan comparaciones. Se trata de los sistemas de información gerencial.

Los sistemas que apoyan las decisiones.

Los **DSS** también llamados **EIS** (*Executive Information System*), tuvieron su origen en **MIS** (*Management Information System*). Por otro lado, surgieron los sistemas de administración de modelos, software encargado del desarrollo, almacenamiento, manipulación, control eficiente y utilización de modelos de datos en una organización. Dos técnicas han predominado en el desarrollo de **MMS** (*Model Management System*), inteligencia artificial y modelado de base de datos. Las tareas de estos sistemas se concentran en: la selección, síntesis, acceso e integración de los modelos. Se requiere que los modelos permitan un manejo equivalente al empleado en el modelo entidad-relación. Aunque se han realizado esfuerzos con respecto a la ejecución, todavía no han cubierto el área de lenguajes de modelado algebraico y de consulta.

MIS.

Sistemas de manejo de información. Se trata de un sistema formal o informal que proporciona información oral o escrita, relacionada con las operaciones internas de una organización y su medio ambiente.

Un sistema formal es aquel que descansa sobre procesos establecidos como obtención de información computarizada y reuniones ejecutivas posteriores, en cambio un sistema informal descansa sobre reuniones improvisadas. Un sistema **MIS** incluye a ambos.

Los elementos de un sistema **MIS** de acuerdo con un esquema propuesto en 1986 por McLeod, son: Sistema físico (trabajadores, recursos y equipo para desempeñar las funciones de una organización), comité de ejecutivos y administradores, base de datos, recursos para el proceso de información (equipos de cómputo y documentos) y medio ambiente. Mostrados en la figura 1.1.

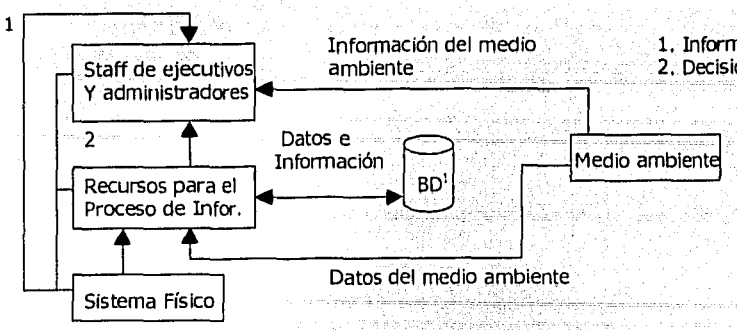



Figura 1.1. Elementos de un MIS.

DSS.

El concepto de **DSS** aparece del uso indebido del término **MIS**, este era usado para apoyo administrativo, pero con el paso del tiempo empezó a asociársele con actividades de cómputo más amplias. En un artículo publicado por "*Sloan management Review*" titulado "*A framework for Management Information System*" en 1971. G.Anthony Gorry y Michael S. Scott Morton aplicaron el término **DSS** para un proyecto suyo, utilizando tipos de niveles de decisión administrativa, que incluía decisiones estructuradas (realizadas en base a reglas o procesos específicos) y su contra parte las no estructuradas.

Peter G. Keen definió a un **DSS** en base a sus objetivos:

- Asistencia a los administradores en toma de decisiones para tareas semi-estructuradas.
- Apoyo en juicios administrativos.
- Eficiencia en la toma de decisiones.

Lo que implicaba ver a los **DSS** como **ESS** (*Executive Support System*), los tres elementos base de un **DSS** son:

- Una base de datos.
- Procesador de información, encargado de generar información.
- Biblioteca de software.

Los **DSS** necesitaron apoyo descriptivo para su mantenimiento por lo que incorporaron los sistemas **IRDS** (*Information Resource Dictionary System*).

IRDS.

Sistema de Diccionario Recursos de Información. Se trata de un antecedente de metadato, es el diccionario de los datos, describiendo su significado y su estructura lógica. Un IRDS es una herramienta computacional que es utilizada para administrar y controlar el acceso a los metadatos de la información.

[1] BD. Base de datos.

El término EIS fue inventado por Rockart y Treacy para un sistema orientado a datos, con habilidad de monitorear y enfatizar factores críticos de éxito, extraer y filtrar información de varias fuentes, combinar información gráfica y textual en formato sencillo para usuarios específicos como ejecutivos de negocios.

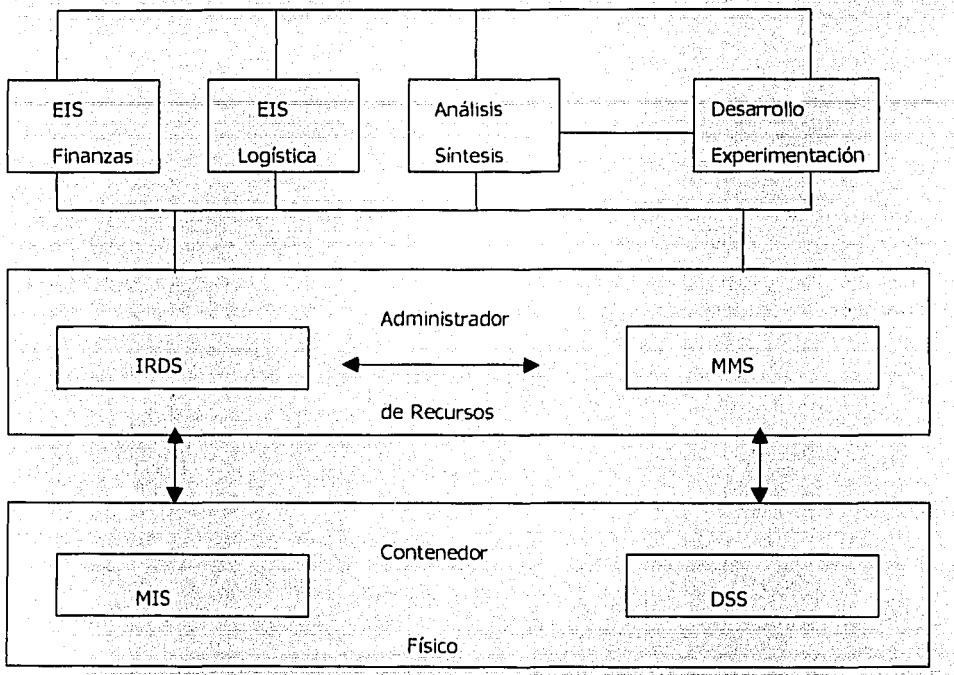



Figura 1.2. Interacción entre sistemas que apoyan la toma de decisiones.

Los **MSS** (*Management Support System*) están compuestos por: Sistemas de Información Administrativa (**OAS**), **DSS** y **EIS**, la integración de estos junto con sistemas de transacciones es clave para un buen desempeño. Los OAS son sistemas que facilitan la comunicación y transferencia de información entre oficinas. [Revista # 6]

Todos estos se concentran en un solo esquema, mostrado en la figura 1.2. Se compone de tres niveles, el primero contiene a los sistemas **MIS** y **DSS**, el segundo contiene a los sistemas administradores **MMS** e **IRDS** y el tercero sistemas **EIS** que permiten experimentación y análisis.

El contenedor físico, debe ser un mecanismo de almacenamiento, para consulta de información relevante para la organización. Sirve de soporte a los modelos financieros, estadísticos y de otro tipo que trabajan de forma conjunta con herramientas de modelado matemático.

El administrador de recursos busca un mejor uso de los recursos informáticos y los modelos existentes. Permite la interacción de **MIS** y **DSS**, proporcionando bases metodológicas para la conceptualización unificada de estos y del conocimiento que las organizaciones tienen y desean utilizar en el futuro.

Utiliza comandos para actualizaciones sobre datos históricos de la organización. También aquellos que facilitan el desarrollo, almacenamiento, manipulación, control y utilización de modelos. Además, aquellos que plantean la resolución de problemas en base a los modelos propuestos en la empresa.

Otros comandos permiten la transferencia de datos a diferentes niveles y la generación de modelos de un nivel más alto que combinados a los diseñados por la empresa satisfacen las necesidades de esta última.

Por último, las vistas ejecutivas, de análisis, síntesis y experimentación están enfocadas a realizar síntesis complejas de los modelos. Este modelo conceptual, corresponde a las ambiciones de un data warehouse, donde se busca que los usuarios en modo texto o con herramientas gráficas obtengan la información de interés para definir sus estrategias administrativas.

Evolución de los sistemas que apoyan la toma de decisiones.

En los 80's con el desarrollo de la arquitectura cliente/servidor y los RDBMS, los **DSS**, comenzaron a llamarse EIS ó ESS, siendo Pilot Command Center el primero de ellos, introduciendo las hojas de cálculo para consultas.

La primera generación de **DSS**, fue constituida por EIS, quienes estuvieron diseñados para análisis predefinido. Su mantenimiento era caro y debido a su construcción, requerían de un gran equipo de profesionales encargados de codificarlos sin incluir una porción de código para interacción con el cliente.

Entre 1984 y 1988, los primeros data warehouse fueron implementados, en diversas compañías, tal es el caso de ABN AMRO, banco holandés que entre 1986 y 1990, desarrollo un proyecto en DB2, a partir de un sistema originalmente codificado en COBOL.

Un data warehouse es una colección de datos, no volátiles, variables en el tiempo, integrados y orientados al apoyo del proceso de toma de decisiones, los datos están dirigidos a ejecutivos, guardados por ejemplo en: semestres, periodos fiscales o años. En formatos consistentes, de modo que no sufran actualizaciones, sino que se preserven como datos históricos de la organización.

Los data warehouses tomaron la idea de los **DSS** que se combinan con los IRDS, con la finalidad de obtener información que conduzca a la creación de estrategias administrativas para una empresa. La información es procesada a través de análisis matemáticos sobre los datos históricos.

**Data warehouse.**

La necesidad de data warehouse.

Ante la recesión económica que experimentaron diversos sectores productivos, los gobiernos controlaron los mercado mundiales, obligando a las empresas a ser más competitivas.

Tal fue el caso, de las aerolíneas, que observaron la presencia de viajeros frecuentes. Hombres de negocios que necesitaban un programa especial de boletos disponibles. Vinculando los sistemas de venta de boletos en un plan diseñado especialmente para ellos, se lograba preferencia del cliente y ventajas en el mercado.

La industria de los alimentos necesito una nueva forma de usar los datos, al darse cuenta de la importancia de los puntos de venta. Era más fácil que el staff dedicado a manufactura tuviera control de la venta y el traslado a almacenes que una alta gerencia. Los tomadores de decisiones obtenían información de los puntos de venta y los distribuidores. Los negocios necesitaban varias perspectivas de sus actividades por separado y en conjunto. Este tipo de ventajas convenció a varias compañías para adquirir lo necesario e instalar sistemas basados en data warehouse.

No obstante, hubo escepticismo por parte de las compañías debido al costo que implicaba poner en marcha un data warehouse. Con el tiempo y gracias a los beneficios proporcionados, utilizar data warehouse se volvió un hecho muy conveniente.

Estructura de un data warehouse.

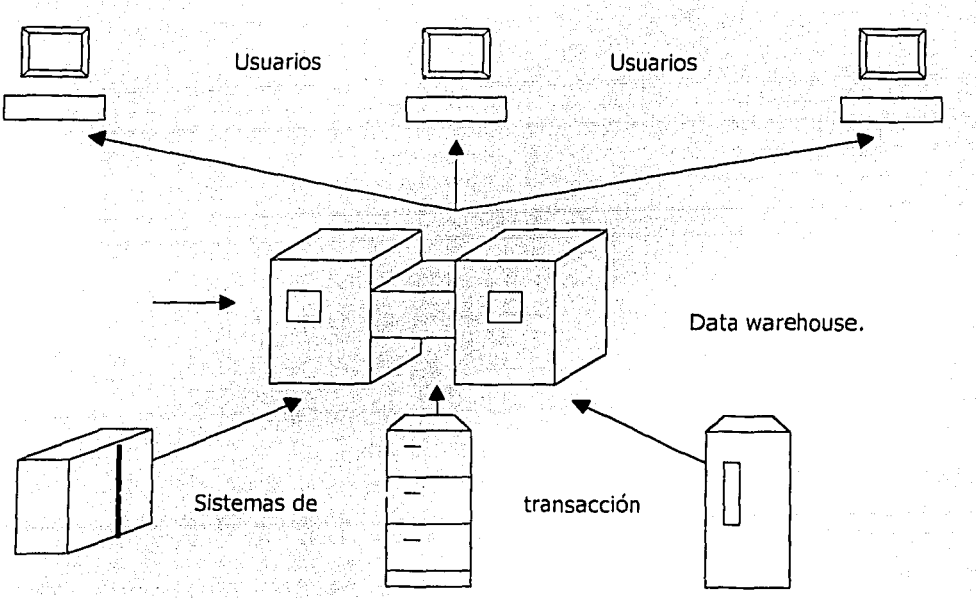



Figura 1.3. Estructura de un data warehouse.

La figura 1.3 muestra la estructura general de un data warehouse y la figura 1.4 la de un data warehouse que utiliza **ODS** (*Operational Data Store*). La diferencia entre uno y otro, consiste en que el primero además de procesar las vistas requeridas por el usuario, contiene el almacén de datos y el segundo solo procesa vistas.

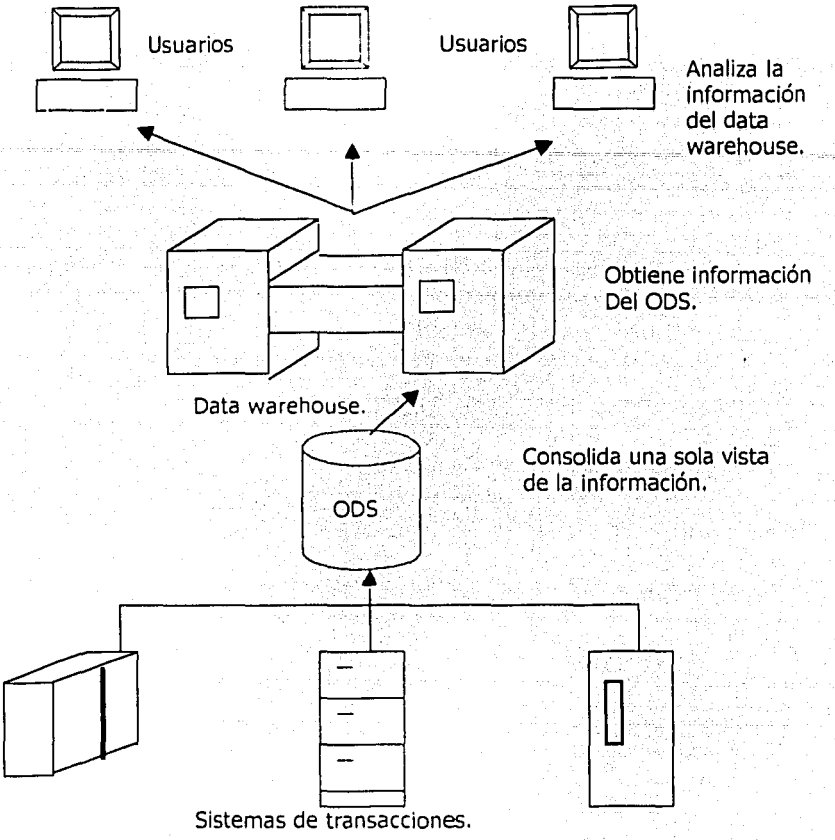



Figura 1.4. Data warehouse con ODS.

Un sistema de data warehouse necesita preparar los datos a través de tres procesos básicos: extracción, transformación y presentación. Este procesamiento comprende el uso de software intermedio para unificar los datos de las diferentes fuentes de información y sistemas de transacciones en un mismo formato.

Data warehouse y data mart.

Un data mart es un pequeño data warehouse realizado específicamente para un departamento. Los data mart de acuerdo con la dependencia que tengan de la fuente de datos, se clasifican en dependientes, independientes e híbridos. Los primeros obtienen información del data warehouse de toda la empresa. Los segundos son sistemas *stand-*

*alone* (aislados de otros sistemas) que recopilan información de uno o más sistemas de transacciones propios o externos a la organización. Los híbridos permiten combinar entradas de diferentes fuentes. Son útiles cuando se necesita integración Ad hoc, en el caso de que un nuevo grupo o producto sea añadido a la organización.

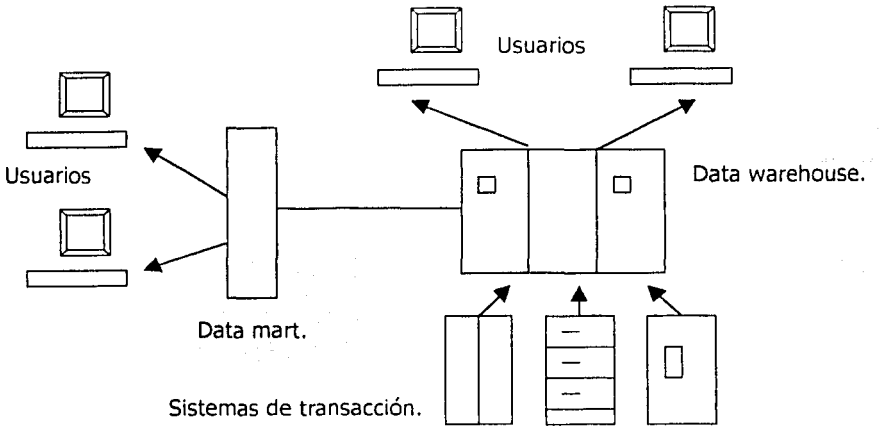



Figura 1.5. Data mart dependiente.

En los data marts dependientes, representados en la figura 1.5, los datos se encuentran: limpios, estructurados en un formato base, resumidos y dentro del data warehouse central. El trabajo de las herramientas de extracción y transformación de datos, es identificar el conjunto de datos requerido por el data mart y copiarlos.

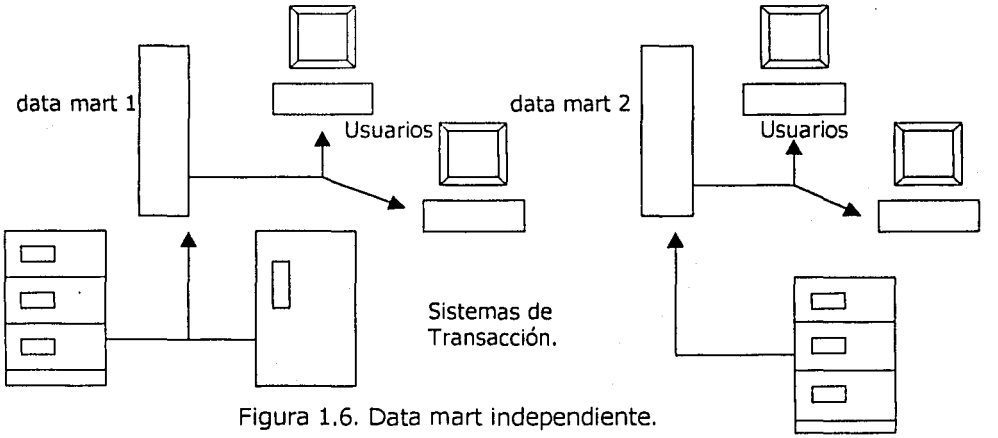



Figura 1.6. Data mart independiente.

En cambio en el independiente, figura 1.6, los desarrolladores se ven involucrados con los procesos de extracción, transformación y transportación a más detalle. El data mart híbrido involucra la posibilidad de tener datos estructurados del data warehouse y procesar parte de la información como sucede en un data mart independiente.

Los data marts dependientes son construidos para mejorar ejecución y disponibilidad, teniéndose un mejor control y menores costos de telecomunicación para acceso local de un departamento específico. Los data marts independientes son llevados a cabo por la necesidad de obtener resultados en menor tiempo. La tabla 1.1, establece la diferencia entre data warehouse y data mart.

| | **Data warehouse** | **Data mart** |
|---|---|---|
| **Visión** | Corporativa | Línea de negocios |
| **Materias** | Múltiple | Solo una materia |
| **Tamaño** | 100 GB a TB | Menor que 100 GB |
| **Tiempo de implementación** | Meses a años | Meses |
| **Fuentes de datos** | Muchas | Pocas |

Tabla 1.1. Diferencias entre Data warehouse y Data mart.

Existen tres aspectos en la tabla 1.1 que deben ser sometidos a consideración. En primer lugar la visión, para un data warehouse los datos tienen significado más amplio. Son utilizados por varios departamentos y cada uno los aprovecha diferentemente.

En segundo lugar el tamaño, es menos laborioso cargar datos en una base de datos hecha para un data mart que para un data warehouse. Por último, el tiempo de implementación hace que los sistemas de data mart sean más factibles y satisfagan los requerimientos de una organización. Debido a que pocos departamentos, con mucha frecuencia consultarían un sistema de DSS. [Libro # 7]

Data warehouse e Internet.

Cuando un data warehouse es puesto sobre una Intranet, los usuarios dudan entre realizar reportes de datos estructurados en forma relacional o navegar sobre ellos de forma no estructurada. Lo primero se logra con **SQL** y lo segundo utilizando lenguajes para construcción de páginas Web (HTML, Javascript, PERL, PHP, ASP o Dynamic HTML). Además se cuenta con una capa analítica y un servidor dedicado a generar código **SQL**, ejecutar cálculos y presentar reportes para los usuarios.

Este servidor permite el acceso de un cliente usando la página web, realizando consultas sobre una base de datos y asegurando conexiones de alta velocidad a todos los elementos de la red. La capacidad de procesamiento es suministrada por una configuración de varios procesadores paralelos, con memoria suficiente para minimizar operaciones virtuales de entrada/salida. Comparte algunas tareas con hojas de cálculo. Brindándole a los usuarios la capacidad de duplicar formulas para cálculos sobre los datos estructurados en columnas y renglones.

Un hecho fundamental es que un sistema de data warehouse requiere no solo de aplicaciones amigables, sino aplicaciones que permitan contactar todos los sistemas fuente de forma ordenada, como una unidad temática y con un tiempo óptimo de respuesta. Las páginas Web permiten visualizar al tema principal en subcapítulos, procesos o departamentos y están diseñadas para mostrar la información indispensable, permitiendo en algunas secciones la interacción o consulta definida por el usuario.

**Data warehousing.**

Data warehouse implica imaginar, toda una estructura lógica y física que soporta los requerimientos de información de un grupo de usuarios sobre un depósito de datos.

Data warehousing en cambio es la serie de metodologías para llevar a cabo las ideas de data warehouse.

Cuando apareció la construcción de data warehouse o data warehousing, la primera fase consistió en estructurar la arquitectura que haría a los datos disponibles al usuario, luego se generó un modelo de datos y por último una interfaz apropiada para presentarlos. Hasta el momento se describen los pasos y las tendencias para su construcción.

Consideraciones para implementar un data warehouse.

El enfoque más exitoso para convertir la información proveniente de sistemas de transacciones en conocimiento es el data warehouse. Como resultado de eliminar fragmentaciones, inconsistencias y heterogeneidades, así la información contenida son datos históricos y datos actuales. La metodología que lo permite consta de las siguientes etapas:

La figura 1.7, detalla las etapas descritas. [Revista #1].

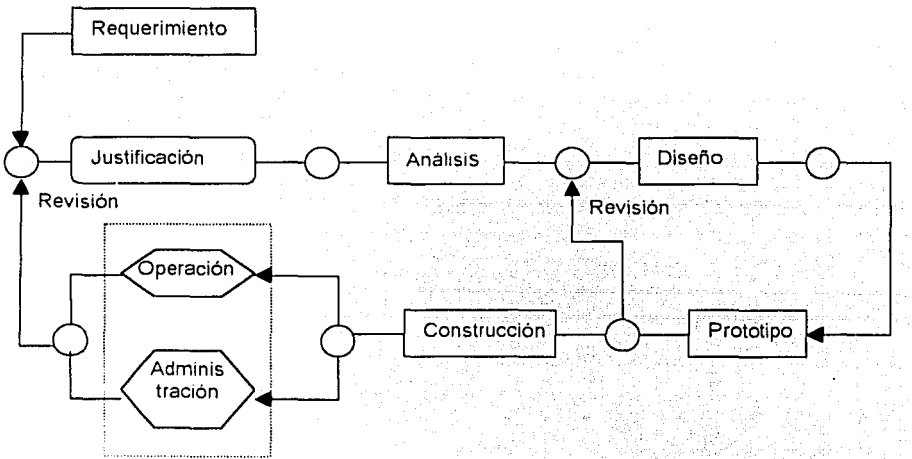



Figura 1.7. Etapas en la construcción de un data warehouse.

**Justificación.** Después de revisar las condiciones actuales de lo sistemas de información, se deben definir los resultados del proyecto: una versión única de la información, consistencia, seguridad y facilidad de actualización. Determinar si se justifica la inversión en base a un análisis costo-beneficio y definir un plan de implementación, ya sea por áreas o general.

**Análisis.** Se definen las áreas o aspectos del negocio involucradas y los requerimientos de los usuarios finales. Hacia el final se debe contar con un documento de objetivos y definiciones de los requerimientos. También se debe incluir una evaluación de las herramientas a utilizar y la arquitectura tecnológica propuesta para el desarrollo del proyecto.

**Diseño/Modelado.** Su diseño es menos preciso que el requerido para un sistema de transacciones. Siguiendo los siguientes pasos:

- Diseño de la base de datos.
- Requerimientos de la extracción de datos.
- Diseño del sistema de extracción de datos.
- Preparación de los datos.
- Diseño de los metadatos (IRDS de data warehouse).
- Diseño de la administración de los datos.

Se incluye la creación de un prototipo, tomando en consideración el tamaño del proyecto y verificando la viabilidad y funcionalidad del mismo.

**Construcción.** Se crean físicamente los componentes previstos en el diseño. Utilizándose: convenciones de nomenclatura, calendarización de procesos, carga del data warehouse, carga de los metadatos, seguridad y administración; además de pruebas y capacitación de los usuarios.

Algunos aspectos a considerar son:

- Los objetivos del negocio.
- Metas reales y de alcance limitado.
- Establecer criterios de evaluación.
- Metadatos definidos acerca de cómo y cuándo se generó la información.

No se debe olvidar la inclusión de los siguientes tipos de herramientas: acceso a datos, transformación de datos, análisis, entrega de información y conectividad, así como, las estructuras de almacenamiento requeridas.

Estrategias de desarrollo en data warehouse.

Independientemente de la tecnología asociada al almacenamiento (relacional o arreglos multidimensionales), se tienen cuatro estrategias en data warehouse:

**Enterprise warehouse**: Contempla la creación de un almacén de datos independiente a los sistemas de transacciones, dedicado exclusivamente a la toma de decisiones. Contiene datos resumidos desde unos pocos gigabytes a cientos de ellos o terabytes. Es implementado sobre mainframe, superservidores UNIX o en plataformas de arquitectura paralela. Su diseño es exhaustivo y su realización puede llevar años.

**Data mart**: Contiene un subconjunto de la información de la organización, enfocado a un grupo específico de usuarios. Es implementado sobre servidores de bajo costo, en

sistemas operativos UNIX y Windows NT. Su integración es compleja en el largo plazo, sino son diseñados con una visión total de la empresa. A diferencia del sistema global, donde se satisfacen las necesidades de bajo nivel en los distintos segmentos de la comunidad, de arriba hacia abajo; los data marts se desarrollan en sentido contrario, de abajo hacia arriba.

**Database Getaways**: Plantea la instalación de software de acceso (*getaway*) que conecta directamente y en línea la aplicación del cliente con la información de sistemas específicos de operación. Permite aplicaciones particulares y de bajos volúmenes de información, en ambientes cliente/servidor. Algunos autores como Alan Simon la han denominado *Warehouse Lite*, identificándose como principales ventajas, la incorporación de herramientas gráficas en PC e inversión relativamente moderada. Sin embargo presenta problemas en la integración de datos, puesto que respeta las estructuras definidas en los sistemas de transacciones.

**Virtual warehouse**: Es un conjunto de vistas de las bases de datos de sistemas de transacciones, algunas de ellas pueden ser grabadas físicamente. Aunque es fácil de construir requiere servidores de datos operacionales de gran capacidad. Se concibe como parte de la evolución del uso de getaways y está fundamentado en la instalación de un depósito central de metadatos, donde se manejan las reglas del negocio y los apuntadores a donde se encuentra localizada físicamente la información. Permite la incorporación de diversas aplicaciones sin invertir en infraestructura de hardware, sin embargo los sistemas de transacciones pueden ser afectados en el desempeño.

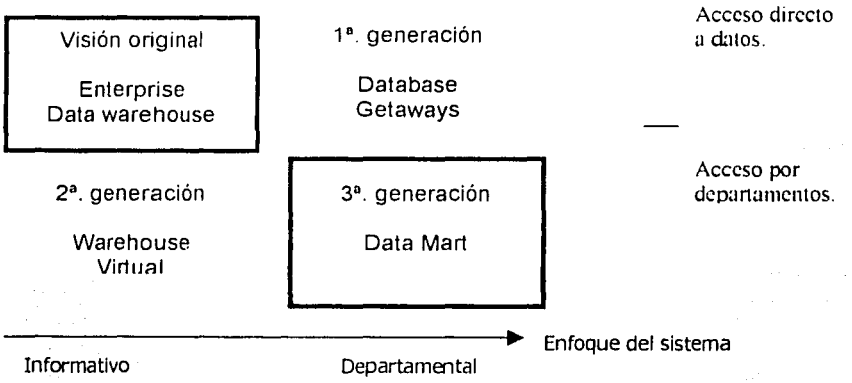



Figura 1.8. Estrategias de data warehousing.

Elección de una estrategia de desarrollo adecuada.

Utilizar algunas de las estrategias planteadas anteriormente depende de las necesidades del negocio y de un análisis posterior sobre la infraestructura de cómputo y comunicaciones en relación con el volumen de información y la cantidad de usuarios.

La tendencia general busca a través del uso de una metodología recortar tiempos de desarrollo y programar la inversión de recursos de manera eficiente. Cada proveedor

tiene su metodología, por ejemplo **IWM** (*Incremental Warehouse Methodology*) de *Information Builders* desarrollada por Earl Hadden, se fundamenta en la estrategia de data mining y plantea un proceso recursivo de dos fases: arquitectura e implementación. Cada fase tiene una duración de 90 días, siendo iteradas para cada aplicación de **DSS**. De éste modo, al final de cada ciclo se tendrán lista las necesidades de un determinado grupo de ejecutivos y analistas con inversiones moderadas.

La estrategia de desarrollar data marts, ha sido identificada como la aproximación al éxito en data warehousing. Se recomienda comenzar con varios data marts, hasta que toda la empresa posea tecnología de data warehouse y evolucione en conjunto. Pasando de esta etapa inicial hacia aquella donde los data marts corren paralelamente junto con un warehouse para la empresa entera. Finalmente se tienen data marts distribuidos que constituyen un data warehouse de arquitectura de múltiples niveles.

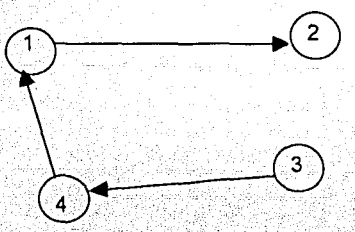
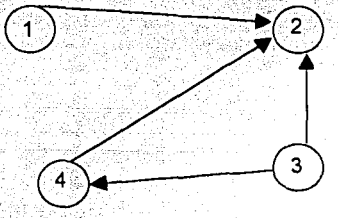



Figura 1.9. Enfoque inicial para las rutas de desarrollo de un data warehouse.

Las figuras 1.9 y 1.10, muestran las rutas para desarrollar data warehouse. En la figura 1.9 se tiene el enfoque tradicional y en la figura 1.10 el enfoque actual.




1. Enterprise warehouse
2. Data mart.
3. Database getaways.
4. Warehouse virtual.

Figura 1.10. Enfoque actual para las rutas de desarrollo de un data warehouse.

Con los data marts que utilizan RDBMS, se crea un mecanismo sencillo para el manejo de datos y con el auxilio de los administradores de red, se reduce significativamente el tiempo de transferencia y se eliminan los obstáculos para un despliegue a gran escala. De este modo, la organización adquiere las siguientes ventajas:

- Reduce cuellos de botella.
- Sustituye los ciclos para lotes de datos por ciclos interactivos.
- Utiliza hardware económico para un procesamiento interactivo de respuestas.
- Apoyo a sitios remotos. [Revista #2].

**OLAP.**

**OLAP** (*On Line Analytical Processing*) es una categoría de tecnología de software enfocada a analistas y ejecutivos, quienes toman las decisiones en una organización. Por medio de acceso consistente, interactivo, rápido y seguro a diversas perspectivas de la información. Se trata de un entorno de desarrollo de aplicaciones. Destinados a obtener información depurada y acorde a las necesidades del usuario. Alimentado principalmente de sistemas de transacciones.

De acuerdo al tipo de almacenamiento tenemos **ROLAP** (*Relational On line Analytical Processing*) para base de datos relacionales, **MOLAP** (*Multidimensional On line Analytical Processing*) para bases de datos multidimensionales y un híbrido de las anteriores, conocido como **HOLAP**. **OLAP** corresponde a los sistemas de información gerencial o **DSS**.

Las PC drill tools emergieron hacia el final de los 80's, como una alternativa a los EIS desarrollados en mainframe. Son el antecedente más directo de **OLAP**, al introducir un avance significativo, la dimensionalidad reflejada en hipercubos (cubos de 4 o más dimensiones).

Sin embargo tenían limitantes, Requerían que un experto modificará y conservará la base de datos. Con limitantes como un porcentaje mínimo de cambios que no tenían significado para usuarios finales. En ocasiones se necesitaba apoyo técnico para mover físicamente los datos, y analizarlos.

Un ejemplo es System W, fabricado por Comshare, que utilizaba datos multidimensionales aportando una estructura similar al hipercubo, destinándose hacia aplicaciones financieras. Le siguieron Commander Prism, Essbase y Metaphor, este último enfocándose a arquitectura cliente servidor, procesamiento multidimensional con datos relacionales y desarrollo orientado a objetos, el hardware contemporáneo no fue capaz de ofrecer rendimiento adecuado para la herramienta por lo cual no adquirió popularidad.

Dimensionalidad.

La dimensionalidad surgió de la necesidad de interpretar a los datos desde una lógica de negocios y no en base a la tecnología que utiliza el sistema.

Se trata de ejemplificar, la perspectiva desde la cual los usuarios finales ven los datos, sin importar, si el almacenamiento del data warehouse, es relacional o multidimensional.

En almacenamiento relacional un conjunto de tablas se encuentran relacionadas. Cada una representa una entidad con sus atributos (las columnas de la tabla). En la tabla 1.2, se tiene la entidad cliente, cada renglón esta asociado con un solo cliente por medio de Id_cliente.

| Id_ cliente | Nombre_cliente | Telefono | Direccion | Ciudad | Estado |
|---|---|---|---|---|---|
| 7 | Microchips | 53-91-13-88 | Asbaje # 8 | Delicias | Chihuahua |
| 9 | Cajita de sorpresas | 53-92-24-62 | Av. Principal # 120 | San Juan del Río | Querétaro |

Tabla 1.2. Tabla de clientes.

Los atributos en cada renglón por sí solos carecen de sentido, el renglón es una unidad indivisible que permite vincular los datos contenidos en la tabla 1.2 con aquellos que están localizados en tablas relacionadas a ella. Si se forma a partir de la tabla 1.2 una matriz de dos dimensiones, donde los renglones corresponden a una dimensión llamada cliente y la columna a otra dimensión denominada teléfono, como se muestra en la tabla 1.3. Se tiene un ejemplo de dimensionalidad aplicado sobre una tabla.

| | Telefono |
|---|---|
| Microchips | 53-91-13-88 |
| Cajita de sorpresas | 53-92-24-62 |

Tabla 1.3. Teléfonos de clientes.

Al igual que en un plano cartesiano, donde una abscisa y una ordenada indican la ubicación de un punto. En **OLAP** cada dato es representada en una celda que corresponde a la intersección de dos o más dimensiones.

El ejemplo anterior es sencillo y muestra como el uso de dimensiones facilita la búsqueda del número telefónico de un cliente. Los usuarios de **DSS**, requieren una forma sencilla de identificar datos específicos entre una enorme colección de estos. Localizar objetos en un plano o en el espacio resulta es tarea tan natural, que usar dimensiones para hallar datos es simple. El siguiente ejemplo muestra la eficacia de utilizar dimensiones para representar información, donde los datos interesantes están distribuidos en varios registros.

| Producto | Región | Ventas |
|---|---|---|
| Licuadora | Este | 50 |
| Licuadora | Oeste | 60 |
| Licuadora | Centro | 100 |
| Batidora | Este | 40 |
| Batidora | Oeste | 70 |
| Batidora | Centro | 80 |
| Lavadora | Este | 90 |
| Lavadora | Oeste | 120 |
| Lavadora | Centro | 140 |
| Horno | Oeste | 10 |
| Horno | Centro | 30 |
| Horno | Este | 20 |





Tabla. 1.4. Relación de ventas de electrodomésticos por región y producto.

En la tabla 1.4, se tienen las ventas de producto por región. El atributo producto presenta varias combinaciones para un nombre dado, complicando la búsqueda de las ventas para una región y producto determinado. Una forma rápida de ubicar el valor de

las ventas por producto y región es usando una matriz. El atributo ventas es la variable representada en las celdas de dicha matriz. Cada celda contiene la intersección del producto y las regiones de venta. Si se quiere saber cuantos hornos fueron vendidos en la región Este, solamente es necesario encontrar el renglón y la columna adecuados, como se muestra en la tabla 1.5.

| | Este | Oeste | Centro |
|---|---|---|---|
| Licuadora | 50 | 60 | 100 |
| Batidora | 40 | 70 | 80 |
| Lavadora | 90 | 120 | 140 |
| Horno | 20 | 10 | 30 |

Tabla 1.5. Matriz formada a partir de la tabla 1.4.

Las hojas de cálculo de Pilot Command Center, propusieron manejar la información dimensionalmente y no en tablas de una base de datos relacional. Este programa inspiró a los desarrolladores de **DSS**, para buscar procedimientos que permitieran manipular los datos en matrices o cubos (objetos de dos o más dimensiones) utilizando almacenamiento relacional o multidimensional.

Para representar 4 dimensiones, la cuarta dimensión es vista por el usuario, como una serie de cubos. La tercera dimensión es una de las aristas del cubo y las dos restantes forman una matriz. Cada matriz es una rebanada del cubo conceptual, en cuyas celdas los datos contenidos son una combinación única de las cuatro dimensiones.

En **OLAP** la información es multidimensional. El usuario trabaja simuladamente con un cubo o un hipercubo como el mostrado en la figura 1.11, hasta encontrar el dato de su interés. Teniendo la facultad de obtener una parte de ese cubo o hacer resúmenes a partir de los datos contenidos en él.

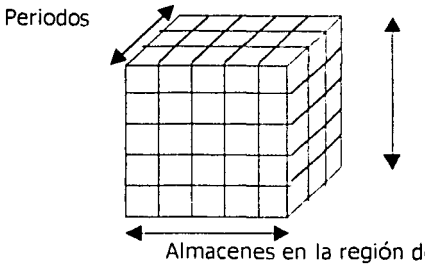





Figura 1.11.
Ejemplo de un cubo de dimensiones.

Los cubos son una representación conceptual de los datos proporcionada al usuario de software OLAP. Físicamente los datos están almacenados en matrices cuando el almacenamiento es multidimensional o en tablas si el almacenamiento es relacional.

La implementación de una vista relacional es útil cuando se quiere saber el total de ventas para un producto y región en particular. En cambio cuando se quiere saber

¿cuántas licuadoras fueron vendidas en la región este? el manejo de la información en base a dimensiones puede ser más adecuado.

La dimensionalidad fue creada para sistemas que apoyan la toma de decisiones, donde el usuario requiere una presentación entendible de los datos, sin tener conocimientos previos sobre un tipo de software. [Libro # 6], [Libro # 16] y [Pagina Web # 2].

**Arquitectura OLAP.**

En data warehousing, se comentó la existencia de una arquitectura que de soporte a data warehouse. La arquitectura o la estructura lógica y física para trabajar con software **OLAP**, se adapta a las características de un data warehouse.

Perspectiva funcional.

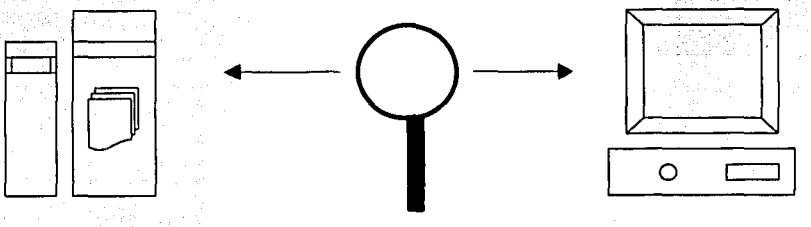



Figura 1.12. Perspectiva funcional.

Son los aspectos a considerar por el equipo de data warehousing para diseñar y desarrollar una aplicación que utilice software **OLAP**:

1.Vista OLAP. La presentación multidimensional de los datos en el *data warehouse* o en los data marts usados por el usuario.

2.Tecnología de almacenamiento de datos. Las opciones tecnológicas de cómo y en dónde se encuentran almacenados los datos. Existen dos alternativas almacenamiento multidimensional y almacenamiento relacional.

De los elementos de la figura 1.12, se concluye que una aplicación desarrollada en software **OLAP**, deberá proveer al usuario de vistas multidimensionales de los datos y funciones que permitan modelar situaciones futuras, o revisar la situación actual de estos. Requiriéndose un profesionista en tecnología de información, quien determinará el tipo de almacén de datos y como se podrá acceder a éste, asegurando manejo de datos y la ejecución efectiva de las vistas.

Perspectiva física.

La perspectiva física describe a los elementos que dan soporte a la perspectiva funcional. Básicamente consiste de tres servicios: servicios de almacenamiento de datos, servicios **OLAP** y servicios de presentación para usuarios, inmersos en una arquitectura cliente/servidor de tres niveles.

Primeramente en la arquitectura cliente/servidor de dos niveles, el manejo datos y las funciones asociadas a su presentación residen en una computadora denominada "cliente" o "estación de trabajo", mientras que otra computadora o equipo llamado "servidor" se encarga de procesarlos para el "cliente".

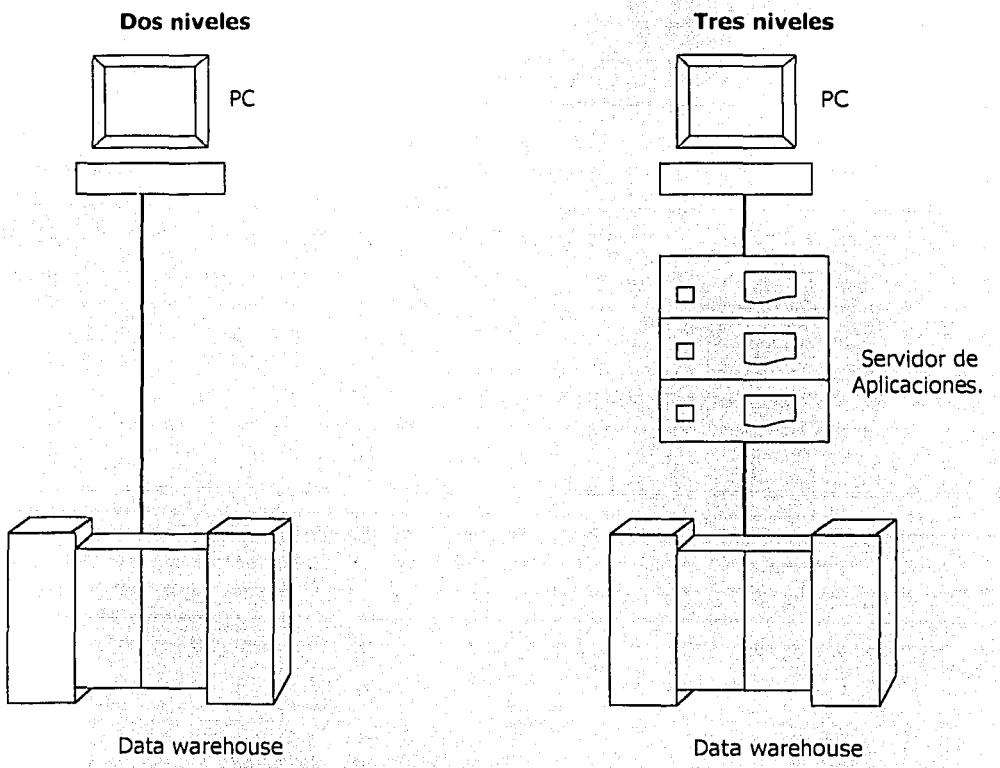



Figura 1.13
Arquitectura para dos niveles y tres niveles.

Como se muestra en la figura 1.13, desde el escritorio de una PC (cliente) se tiene acceso directamente a la información, proporcionada por un servidor de aplicaciones que la obtiene de un data warehouse.

En una arquitectura de tres o n niveles, las funciones del cliente son realizadas por un servidor intermediario (servidor de aplicaciones) que ejecuta la lógica de la aplicación y proporciona sus servicios a muchos clientes, utilizando middleware.

Cualquiera de los dos tipos de arquitectura descritos, puede ser utilizada para implementar un data warehouse. Si se instala data warehouse en dos niveles, se utiliza tecnología relacional y las consultas de las aplicaciones son diseñadas por el desarrollador, impidiéndoles a los usuarios facilidad para contestarse su propios requerimientos sobre los datos. En el caso de tres niveles, la lógica de la aplicación es centralizada en un servidor dedicado, el cliente utilizando alguna API, llama a la aplicación conectándose al servidor a través de middleware.[Libro # 6] y [Libro # 16].

Software intermedio.

La labor del middleware o software intermedio, radica en: convertir los datos de distintos formatos, por ejemplo de DB2 a Oracle, vigilar que la información viaje entre diferentes protocolos (TCP/IP, IPX de Novell, SNA de IBM), establece la relación entre código de presentación Visual Basic y código de acceso **SQL**, vigilando la seguridad de la información.

Necesita ir más allá, de conectar servidores y transferir datos, debe establecer una sesión interoperable. Donde los sistemas operativos y los DBMS o Sistemas Manejadores de Archivos no sean obstáculo. La posibilidad de ofrecer interoperabilidad para nuevos sistemas operativos, familias de protocolos y soporte a distintas herramientas de acceso. Lo anterior ha llevado a un grupo de especialistas a coincidir en la idea, de una arquitectura middleware, la cual ofrezca:

- Un rendimiento adecuado.
- Adaptarse a cambios y adiciones.
- Fácil de usar y administrarse.

Generalmente contemplará diversos elementos de distintos fabricantes de software. Para lograr que cumpla con sus objetivos un middleware debe realizar las siguientes funciones:

- Cobertura. Significa revisar y establecer prioridades entre, fuentes de datos, protocolos, aplicaciones y sistema operativo. Por ejemplo, considerando el soporte de DB2 en mainframe y Oracle en UNIX, se sugiere utilizar software con conexión DB/Datacom.

- Facilidad. El software debe incluir diversos extractores de datos, que hagan todas las conversiones de formatos necesarias entre tipos de datos (ASCII y EBCDIC) y lenguajes de acceso. Las facilidades del programa extractor varían en función del proveedor. Algunos requieren código adicional por parte del equipo de desarrollo. Otros utilizan interfaz para definir los diversos tipos de extracción que se necesita realizar. Es aquí donde las API's, son revisadas para estructurar el programa que "converse" con el middleware.

- Apertura. Se trata de poner todo junto, estándares como: ODBC, SAG-CLI, DRDA de IBM, CORBA u OLE, documentación del proyecto y documentación sobre API's. Incluyendo integración con middleware propietarios como: ORACLE SQL*Net o Sybase Db-Library.

- Seguridad. Consiste en permitir encriptamiento y autentificación de mensajes. Controlando el acceso de usuarios a través de estándares como: DES, RSA o Kerberos/DCE.

- Administración. Incluye el monitoreo del rendimiento de todas las piezas del ***middleware***. Obteniéndose información sobre utilización de recursos, detección de fallas y control de las distintas configuraciones. Por medio de estadísticas de las consultas realizadas e información de las bases de datos afectadas, con el fin de recomendar tácticas para manejar los programas de extracción.

Se recomienda después de definir el esquema de data warehouse requerido, definir las opciones de herramientas viables y en base a criterios previamente establecidos decidir cuál proveedor ofrece el middleware más adecuado. [Revista #3].

Extracción, transformación y presentación de datos.

El software intermedio es un enlace, entre los servicios **OLAP** y el servidor, y entre el cliente y los servicios **OLAP**. El cliente cuenta con herramientas de consulta y reporte. Los servicios **OLAP** están formados por herramientas de extracción, transformación y carga de datos dentro del data warehouse. El problema más fuerte se presenta en la transformación o mapeo de los datos. Las herramientas deben ser capaces de realizar las siguientes tareas:

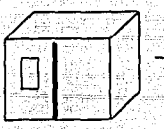
- Convertir los datos de un formato a otro.
- Grabar los datos modificados en el data warehouse.
- Cambiar el formato de los datos para presentarlos de formas significativas.
- Obtener los datos de los sistemas de transacciones originales.

Estas herramientas trabajan con **GUI** (*Graphic User Interface*) y reglas para realizar la transformación de los datos. Generan metadatos y definiciones, para los datos del sistema origen y los destinados al data warehouse.

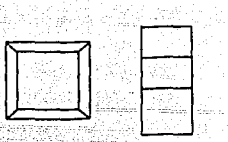
Los metadatos y definiciones generados en la transformación, son utilizados por herramientas de modelado de datos para presentarlos al usuario final. La figura 1.14, muestra el proceso de preparación de los datos para sistemas basados en data warehouse. [Libro # 7].

Sistema Operacional (mainframe o cliente/servidor)

Extracción: Los datos son hallados y transferidos del sistema operacional al data warehouse o a la plataforma de transformación.



Transformación: Un programa especial "limpia" los datos originales en base a reglas predefinidas.



Plataforma de transformación.

Presentación: Un programa transfiere los datos de la plataforma de transformación a las estructuras de datos del data warehouse.

Data warehouse.

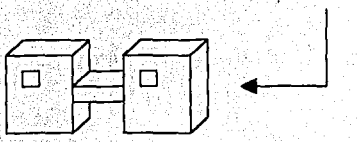


Figura 1.14. Preparación de los datos.

Características de los Servidores.

Los **DSS**, requieren la mezcla de servidores **LAN** (*Local Area Network*) servidores de multiprocesamiento y **MPM** (*Massively Parallel Machines*); de otra forma se utiliza **SMP** (*Symmetric Multiprocessing*) que ofrece sistemas **UNIX** de memoria compartida con un rango de 2 a 64 procesadores, aunque algunos sistemas prefieren usar de 4 a 8 procesadores.

Los servidores con tecnología **MPM** tienen cerca de 500 procesadores con su propia unidad de memoria por lo que pueden manejar terabytes de datos. A pesar de la diversas variedades para el almacenaje de datos, la mayoría concuerda en algunas reglas básicas respecto a la capacidad de servidor.

- Bases de datos pequeñas con consultas sencillas. Servidores **LAN** para data marts, donde la base tiene 5 Gb de datos.

- Bases de datos de tamaño medio y consultas complejas. El tiempo de respuesta es más rápido si se trabaja con **SMP**, además son más económicos que los **MPM**. Las cantidades grandes de memoria reducen el tiempo de consultas.

- Bases de datos enormes y consultas muy complejas. Cuando la base de datos crece más allá de lo que puede manejar un sistema SMP, para aumentar el rendimiento se

usarán clusters o máquinas MPP. La experiencia convencional sugiere que las máquinas **SMP**, necesitan ser retiradas para almacenes de 500 GB.

IBM por ejemplo usa tres estrategias de procesamiento paralelo. En la primera estrategia se tienen varios discos duros que vacían su contenido en un almacenamiento central al que tienen acceso los procesadores para entrada y salida de datos en paralelo IOP. La segunda estrategia es SMP para DB2, bajo AS/400. La tercera denominada *Loosely Coupled Systems*, adjunta sistemas múltiples (más de 32) a un cluster (conjunto de 4) con unidad de memoria propia, expandiendo la capacidad de disco a 16 terabytes. Los sistemas múltiples se conectan a través de protocolos en intranet. Cuando se combina SMP y *Loosely Coupled System* resulta un procesamiento **MPM**.

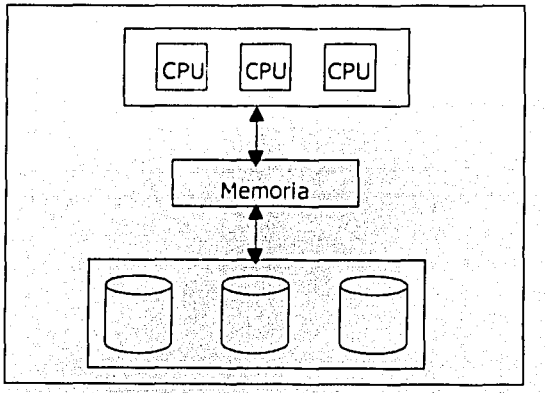



Figura 1.15. Arquitectura SMP.

En la arquitectura SMP figura 1.15, el rendimiento conforme aumenta el número de procesadores converge aun valor límite, debido a que se comparten recursos entre los procesadores.

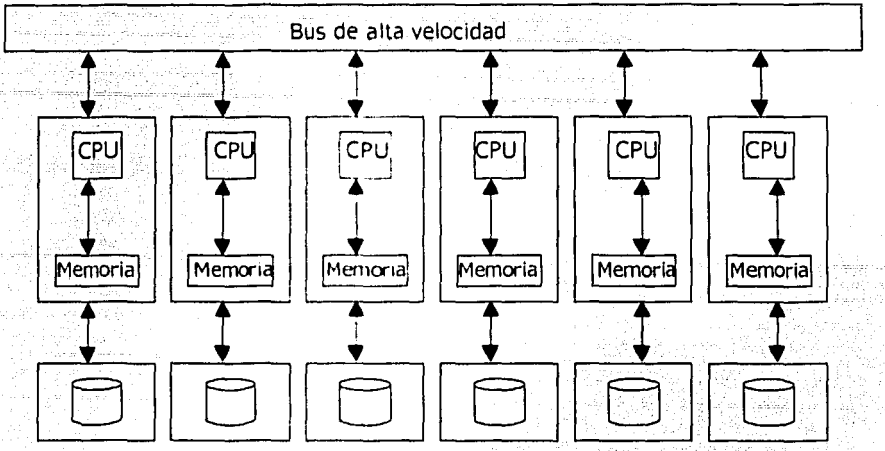



Figura 1.16. Arquitectura MPM.

En la arquitectura **MPM** figura 1.16, conforme se incrementa el número de procesadores se incrementa el rendimiento, puesto que cada procesador tiene su propia unidad de memoria. [Libro # 14] y [Página Web # 5].

Bases de datos multidimensionales.

Una base de datos multidimensional o **MDDB** (*Multidimensional Data Base*), es una "maquinaria" especializada que almacena datos en formato de arreglos, los cuales corresponden a las dimensiones que manipulan los usuarios. Dejando de lado el uso de **SQL** y utilizando una **API** (*Application Programming Interface*) logrando más funcionalidad analítica en el mismo servidor.

Debido a que sus métodos de acceso y técnicas de almacenamiento son diferentes al almacenamiento relacional, es difícil para los usuarios familiarizarse con ellas. La ejecución de las consultas depende de precálculos y consolidaciones efectivas. Si se introduce un archivo de 200 MB, éste se expandirá a un archivo de 5 GB dentro de la base de datos multidimensional, cantidad que es cargada y consolidada en varios días.

Una base de datos de 16 dimensiones con 5 miembros en cada una llegará a tener alrededor de 150 billones de celdas. Aunque dichas bases de datos están diseñadas para almacenar 32 dimensiones, la mayoría de las veces utilizan menos de 10 dimensiones. Como solución, se usa una base de datos relacional de procesamiento paralelo con datos resumidos y extraídos del data warehouse. Otra forma de coexistencia entre bases de datos relacionales y MDDB sucede cuando la base de datos relacional es utilizada para Enterprise data warehouse y la **MDDB** en un data mart.

Estas son algunas de las desventajas detectadas hacia el final de los años 90's en bases de datos multidimensionales:

- No soportan la junta lógica de arreglos multidimensionales.
- No soportan la creación de múltiples arreglos relacionados.
- El cubo de datos es vuelto a cargar por completo cuando algún dato es modificado. [Libro # 5].

Arquitectura ROLAP.

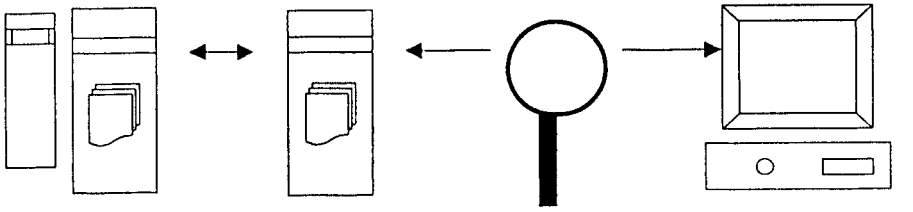



Figura 1.17.
Diagrama de Arquitectura ROLAP.

En **ROLAP** los datos son almacenados en tablas. A partir de la figura 1.17, se observa que un servidor **OLAP** convierte los requerimientos del usuario en sentencias **SQL**, después toma los resultados y los presenta en forma multidimensional. Se utilizan modelos dimensionales diseñados específicamente para su implementación, siendo necesario programar procesos de carga de información en los clientes y manejar metadatos. Cuando el Data warehouse o el Data mart son muy grandes, debido al uso de índices y muchas tablas, se necesitan consultas paralelas.

Las técnicas de paralelización de acuerdo con el lema "divide y vencerás" reparten la carga de trabajo entre varios procesos simultáneos. Cada fabricante tiene su propia técnica, sin embargo se aconseja dividir el trabajo de cada consulta, en dos tareas, extracción de datos y carga de los mismos en el cliente, a su vez éstas últimas son subdivididas.

Entre los índices que maneja se tienen el índice multitablas y el índice de bitmaps. El índice multitablas, implica la operación de juntar varias tablas por lo cual llega a ser muy lento. El índice de bitmaps, se aplica a datos binarios. Agilizan la lectura de datos de baja cardinalidad y son más pequeños que los índices tradicionales. Entre los datos de baja cardinalidad se encuentran tipos de compra (efectivo, crédito o cheque) o el sexo de una persona (F ó M).

Para diseño de sistemas **ROLAP** se recomienda:

1. Construir el modelo dimensional: esquemas estrella, copo de nieve y constelaciones.
2. Añadir agregación y resúmenes.
3. Particionar grandes volúmenes de datos en segmentos más manejables.
4. Añadir índices.
5. Crear y almacenar metadatos.

Para la realización de una consulta en estos sistemas, se llevan a cabo los siguientes pasos:

1. Construir la herramienta cliente usando una vista dimensional de los datos.
2. Consultar al servidor **OLAP** y analizar los metadatos asociados.
3. Crear sentencias SELECT multipaso y/o subconsultas correlacionadas que trabajan sobre la base de datos relacional.
4. Ejecutar funciones multidimensionales, cálculos y fórmulas. Traducir la información obtenida de la consulta a la base de datos relacional en descripciones del negocio.

Las funciones realizadas por sistemas con arquitectura **ROLAP** son:

1. Cálculo de funciones estadísticas y financieras.
2. Obtención de datos por ciudad o por región, por ejemplo.
3. Navegación de la información a través de metadatos.
4. Elección de herramientas front-end.
5. Niveles de seguridad determinando privilegios a determinados usuarios.
6. Dividir las dimensiones en categorías o tipos.
7. Copias de seguridad de la base de datos.
8. Consultas Ad hoc. Consultas donde el usuario utiliza sus propios conocimientos de **SQL**.

   Las consultas Ad hoc utilizan criterios de selección escogidos por el usuario.

La administración y manejo de estos sistemas requiere:

- Actualizar periódicamente el servidor **OLAP** ó no cargarlo "loading" inicialmente.
- Usar la copia de seguridad estándar existente y los procesos de seguridad.
- Sincronizar los nuevos datos con los ya existentes.
- Monitorear índices e información desnormalizada.

**ROLAP** se recomienda cuando se construye una aplicación grande y donde la necesidad de detallar la información requiere cambios en la definición de las dimensiones y las relaciones entre las mismas.

Para su implementación física se tienen, las siguientes tendencias:

**Thick Client**. Consiste de la aplicación cliente, un motor local de base de datos y un servidor. Cuando el servidor tiene nuevos datos, el cliente actualiza su base de datos local con la nueva versión, esto llega a ser útil si el cliente es una Lap top y sólo se accede a

una porción de los datos. Presenta limitaciones para el espacio en disco, por lo que se debe determinar que porciones de datos van a ser distribuidas.

**Thin Client**. Se utiliza un cache local para base de datos. La aplicación se comunica con el cache por medio de **SNAPI** (*Structured N-dimensional Application Program Interface*). Los datos permanecen en el cache durante la sesión, eliminando la necesidad de recuperación de datos más de una vez. Sin embargo, tiene un impacto negativo en la ejecución porque las cláusulas IF revisan constantemente, si el buffer esta lleno.

**Web browser**. Busca que el cliente sea todavía más "delgado", representa un costo bajo en mantenimiento e implementación donde las sesiones remotas son necesarias. Produce páginas Web dinámicas con tablas, gráficas y otros elementos codificados en HTML que permiten la exportación de datos y la publicación de reportes. Utiliza un software adicional entre el cliente y la maquinaria **OLAP**, convirtiéndose en una arquitectura de cuatro niveles, su trabajo es recibir los requerimientos del usuario en código HTML. [Libro # 6], [Libro # 13], [Libro # 14] y [Libro # 16].

Arquitectura MOLAP.

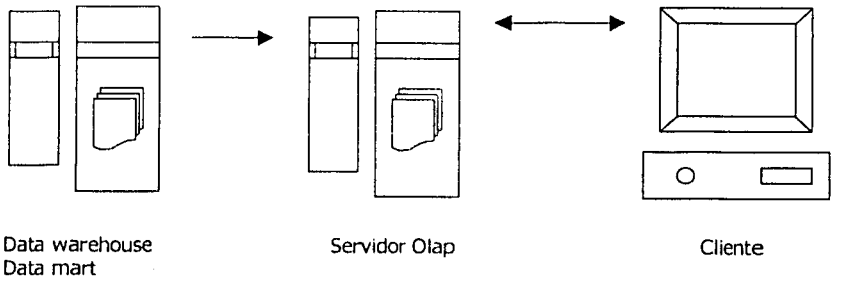



Figura 1.18
Diagrama de arquitectura MOLAP.

En **MOLAP**, cuya arquitectura puede verse en la figura 1.18, los datos son almacenados en arreglos o matrices, que físicamente están constituidos por almacenes de soporte magnético (disco) o en RAM. La RAM es 150 veces más rápida que el disco pero más cara, el disco nos ofrece un acceso aceptablemente rápido, aunque el volumen de datos puede ser grande. Como solución al problema de espacio limitado, se usa tecnología de compresión, denominada *sparse matrix compression*.

**MOLAP** permite acceso concurrente multiusuario para lectura y escritura, así como, también se permiten, accesos concurrentes escritura-varias lecturas. Los datos organizados, son almacenados en resúmenes de datos agrupados llamados consolidaciones. Los índices utilizados para los arreglos de datos son pequeños obteniéndose menor tiempo de respuesta para consultas complejas, además la actualización de los datos no afecta su desempeño.

Para el diseño de sistemas **MOLAP** se recomienda:

1. Seleccionar funciones interesantes del negocio.
2. Identificar los valores numéricos asociados.
3. Determinar las dimensiones y el nivel de detalle de cada una.
4. Definir el modelo lógico, para obtenerlo del almacenamiento multidimensional o filtrarlo del contenido seleccionado, ya sea del data warehouse o del data mart.

Se desea que **MOLAP** realice las siguientes funciones:

1. Respuesta rápida a consultas intensivas.
2. Actualización interactiva de los datos.
3. Explota las relaciones entre las dimensiones y descubre nuevas relaciones.
4. Análisis comparativo de: porcentajes, promedios y cálculos; por periodos o clases.
5. Cálculos a nivel de renglón para aplicaciones orientadas a hojas de cálculo.
6. Funciones definidas por el usuario.
7. Funciones estadísticas y financieras.
8. Manejo inteligente del tiempo.
9. Operaciones entre dimensiones.

**MOLAP** se usa para aplicaciones fijas, o donde las relaciones entre las dimensiones no necesitarán ajustes conforme el usuario haga consultas, además es recomendable, si se quiere una ejecución rápida y bajo costo de memoria.

Para su implementación física se tiene:

**Fat Client.** Dado el volumen de información que maneja puede presentar cuellos de botella en el proceso de cargar de datos. Afectándose negativamente la ejecución del sistema y la seguridad de los datos. Buscando soluciones se ha generado una variación que consiste de un almacenamiento multidimensional al nivel de data mart, para distribuir subconjuntos seleccionados de datos en cada estación de trabajo, con acceso y almacenamiento local.

**OLAP Data mart.** Aquí los servicios **OLAP** y el almacenamiento multidimensional se combinan para extraer los datos del Data warehouse y transformarlos en estructuras multidimensionales almacenadas en el servidor de Data mart. Muchos de los data marts son refinados antes de mostrarse al usuario, con funciones especialmente reestructuradas de refinamiento adicional para filtrar subconjuntos de datos.

Una variación de **OLAP** Data mart, separa el almacenamiento multidimensional del Data mart del servidor **OLAP**. Es utilizada cuando el almacén es grande y también lo es el número de usuarios. A fin de compartir, datos en estaciones de trabajo con menos software **OLAP**. [Libro # 14], [Libro # 6] y [Libro # 16].

Aspectos a considerar entre MOLAP y ROLAP.

Todos los sistemas **OLAP** deben hacer un canje entre el grado de compilación y el tiempo de ejecución. Ambos aspectos dependen del número de datos que han de ser

obtenidos o calculados, del procesamiento por parte de servidores, del modelo lógico sobre el cual alguna de las tendencias **OLAP** trabaje para satisfacer los requerimientos del usuario y del software intermedio.

En ocasiones será más importante el tiempo de ejecución, en otras este puede verse disminuido en favor del beneficio que otorga una buena compilación, hacer posible la consulta de los datos en tiempo real.

Los siguientes aspectos contrastan la labor de **MOLAP** y **ROLAP**.

Manejo de dimensiones.

**ROLAP** tiene ventaja sobre **MOLAP**, puesto que al incluir un modelo dimensional, permite modelar el comportamiento de la dimensión en consulta y prever soluciones.

**MOLAP** tiene limitaciones, puesto que depende del número y tamaño de las dimensiones, ya que para cada dimensión se reserva un espacio en memoria, sin que haya una estructura clara de su funcionamiento en consultas.

Para un número entre 3 y 10 dimensiones **ROLAP** con un grado de compilación entre 50 y 60 % trabaja satisfactoriamente. Si el número excede las 10 dimensiones, **ROLAP** requerirá más trabajo por parte del servidor, obteniéndose en mediano plazo la respuesta esperada por el usuario. **MOLAP**, requiere arduo trabajo por parte del servidor, realizado en un lapso de tiempo mucho mayor que el efectuado por **ROLAP**.

Tamaño.

El tamaño de la base de datos en sistemas **MOLAP**, crece más rápidamente que en sistemas **ROLAP**, debido a la dispersión de los mismos. En ambas se utilizan técnicas que manejan la dispersión de los datos, en el caso de **ROLAP** implica el manejo de estructuras de datos y algoritmos, o bien estrategias de **DML** usando comandos como INSERT y DELETE.

Escalabilidad

Este término implica la compilación de datos en consultas. Los datos más simples o atómicos son utilizados para obtener consultas más generales, en vez de consultar las ventas semanalmente se consulta por mes o año. Lo anterior conlleva varias operaciones en el servidor, operaciones que se ven afectados por el crecimiento de la base de datos.

Por su carácter relacional **ROLAP** es más escalable que **MOLAP**, ya que utiliza procesamiento en paralelo para el tratamiento de la información, así como una particiones de la base de datos.

**MOLAP** cuenta con un esquema de memoria predefinido, en las celdas se encuentra contenida la información, de forma que no hay una estructura lógica que permita manipular los datos necesarios para la consulta.

Análisis

La consulta sobre la información es transparente para el usuario, quien solo maneja dimensiones sobre herramientas de consulta visuales. En el caso de **ROLAP** es el propio software quien crea las consultas **SQL** sobre las tablas de datos. **MOLAP** pocas veces maneja datos atómicos por lo que no utiliza un lenguaje como **SQL** al generar consultas.

El modelo dimensional de **ROLAP**, implica la junta de varias tablas y sentencias **SQL** complejas dependientes del grado de normalización de las tablas. Aunque **MOLAP** trabaja sobre los datos de una MDDB, con series de tiempo y secciones de código para obtener resultados estadísticos y financieros, **ROLAP** permite análisis anidados de estructuración más compleja, filtros de comparación entre los datos y consultas Ad hoc, que pueden o no, encontrarse en distintos segmentos, para unir sus resultados en reportes.

Si se desea ver un segmento del cubo de datos, muchas veces **MOLAP** tendrá que construir el cubo de datos con las dimensiones adecuados para obtener el subconjunto de información, lo que no sucede con **ROLAP**; que puede localizar la información necesaria dentro de tablas relacionadas.

Ejecución.

**MOLAP** para bases de datos pequeñas ofrece un acceso rápido, a los datos o a sus resúmenes, ya que muchas consultas están hechas previamente y almacenadas dentro de un cubo de datos.

En **ROLAP** el administrador debe agregar tablas de resumen, por lo que los resúmenes son calculados de acuerdo con los requerimientos del usuario, esto disminuye su velocidad, pero le ofrece al usuario la posibilidad de trabajar con datos atómicos.

La ejecución de lectura entre tablas y matrices indica que cientos de renglones son leídos de una tabla en un segundo, mientras de una matriz se obtienen datos en columnas y renglones a razón de 10 000 por segundo. Un vector de datos, es obtenido más rápidamente de un sistema que utiliza **MOLAP** que de aquel que utiliza **ROLAP**.

Mantenimiento.

**ROLAP** requiere más esfuerzo que **MOLAP**, se administran los índices y las tablas agregadas, basándose en el conocimiento del número de registros y su contenido.

| Característica | ROLAP | MOLAP |
|---|---|---|
| Almacenamiento, acceso y vista de datos | • Tablas de columnas y renglones.<br>• Sentencias SQL.<br>• Herramientas que invocan alguna o varias APIs. | • Arreglos<br>• Hipercubos y multicubos<br>• Tecnología para manejar densidad de datos.<br>• Hojas de cálculo |
| Uso y empaquetado | • Motor RDBMS.<br>• Vistas y análisis desde lo más general a lo más particular y viceversa.<br>• Amplio rango de consultas. | • Motor multidimensional<br>• Vistas y análisis, para diferentes niveles de generalización y presentación de resúmenes de datos.<br>• Consultas rápidas |
| Tamaño de la base de datos y actualización | • Gigabytes a terabytes.<br>• Incremento del tamaño al almacenar índices y efectuar desnormalización.<br>• Consultas paralelas.<br>• Actualización en uso. | • Gigabytes.<br>• Compresión de datos[2].<br>• No hay actualizaciones en tiempo de ejecución.<br>• La actualización consiste en rediseñar todas las celdas de la MDDB. |

Tabla 1.5. Cuadro comparativo ROLAP vs. MOLAP.

La efectividad de **MOLAP** depende de un alto grado de compilación de la información. En cambio **ROLAP** utiliza varios grados de compilación que le permiten manejar sistemas con grandes volúmenes de datos, volátiles y ubicados en muchas dimensiones.

**ROLAP** es más flexible y se adapta a una amplia variedad de necesidades para varios sistemas **DSS**. **MOLAP** es recomendable para departamentos o data marts con pequeños volúmenes de información y pocas dimensiones. [Página Web # 1], [Página Web # 4] y [Libro # 6].

Arquitectura HOLAP

Los adelantos recientes en las herramientas han permitido un enfoque que utiliza lo mejor de la funcionalidad de **MOLAP** y **ROLAP**. Estas herramientas, llamadas **HOLAP** son los híbridos de **MOLAP** y **ROLAP**, combinan el acceso dinámico de **ROLAP** y las capacidades analíticas más sofisticadas de **MOLAP**.

[2] Manejo de datos esparcidos.

Los beneficios de este enfoque son que el aumento de datos solicitados se regula dinámicamente, hay facilidad de mantenimiento y la flexibilidad para cambiar estructuras. Sin embargo, los diferentes conceptos sobre como diseñar e implementar sistemas que utilizan tecnología **OLAP**, ocasionan incompatibilidad sobre todo si se mezclan herramientas y sistemas de bases de datos de diferentes fabricantes de software.

Los productos híbridos varian en la naturaleza con respecto al nivel de solución que proveen. Por ejemplo, el servidor puede formular los conjuntos multidimensionales para el cliente y el usuario tiene la opción de almacenarlos en el servidor. Sin que se necesite conseguirlos de la base de datos relacional cada vez que alguien requiera trabajar con ellos.

La fuente de datos obtiene valores ya almacenados del conjunto multidimensional o los generados a partir de preguntas hechas sobre la marcha, de forma totalmente transparente para el usuario. Materializando vistas se tienen los datos asociados almacenados en vez de calculados.

Materializar vistas es una técnica de optimización de respuesta, poderosa y común pero es un recurso demasiado caro, si se tienen que materializar todas las vistas. Otra técnica, consiste en utilizar diversos algoritmos para encontrar los conjuntos que son mejores candidatos para materializar sus vistas. Dando buenos resultados en relación al espacio utilizado y el tiempo promedio para consultas. Si el análisis involucra, simulaciones, proyecciones y pronósticos (cálculos que no pueden expresarse en **SQL**) se usa un cache multidimensional.

También los administradores de base de datos pueden mezclar y equiparar la cantidad de datos que requiere el data warehouse. Como sucede en operaciones que involucran gran cantidad de datos (decenas o centenas de gigabytes) procesos costosos para una **MDDB** y lentos para una base de datos relacional. Teniéndose datos más significativos para el usuario, o sea un preanálisis. [Libro # 14] Y [Página Web # 1]

**Funcionalidad de OLAP.**

La funcionalidad del software **OLAP** está caracterizada por un análisis multidimensional dinámico, siendo implementado en una arquitectura cliente/servidor. Ayudando al usuario final a sintetizar información en vistas personalizadas, proyecciones sobre datos históricos, y operaciones que permiten obtener subconjuntos de los cubos de datos.

Aunque no es un estándar, en general cualquier software **OLAP**, posee algunas de las siguientes características:

- Rapidez. Se trata de un rango de tolerancia en el tiempo de respuesta de los sistemas que utilizan **OLAP**, comprendida en un intervalo de [1,20] segundos. Es decir, respuestas de menos de 5 segundos para preguntas sencillas (pocos recursos de programación y memoria para consultar y sintetizar información).

- Análisis. Se refiere a que el sistema puede trabajar con cualquier análisis estadístico o lógico del negocio, que haya sido requerido por el usuario. Sin necesidad de contar con el apoyo del departamento de Sistemas o de una pre-programación.

- Multidimensionalidad. El sistema debe proveer una vista de los datos de forma multidimensional, ya sea en cubos o hipercubos, sin importar la tecnología utilizada para administrar la base de datos. Las herramientas deben ser capaces de analizar los datos en cualquier dimensión y en distintos niveles de agregación, sin que el usuario tenga que comprender la estructura del sistema a detalle para poder realizar sus vistas.

- Acceso compartido. Implica la necesidad de proteger la información confidencial y asegurar la concurrencia de los usuarios a los datos en cada actualización, para operaciones de lectura/escritura.

Además, debe ser capaz de crear resumenes para todas las combinaciones posibles entre las dimensiones. Como ventajas la tecnología **OLAP** ofrece: mayor rapidez y efectividad en el análisis de datos, elaboración de gráficas, extracción de relaciones entre los datos que no son percibidos por expertos humanos.

Las herramientas **OLAP** son utilizadas en el área Financiera y de publicidad, para obtener: análisis de ventas, informes de gestión, informes financieros, análisis de rentabilidad y análisis de calidad.

**Data warehouse y herramientas de consulta.**

El data warehouse permite análisis de los datos del negocio. La selección y uso de las herramientas de consulta se reflejará en el éxito del mismo. Dichas herramientas básicamente deben cumplir con los siguientes criterios:

1.Vistas de datos y reportes de cierta regularidad.
2.Facilidad para que el usuario final desarrolle sus propias consultas y reportes.

Se les ha clasificado en tres áreas:

- Reportes estándar.
- Consultas Ad hoc.
- Análisis multidimensional.

Los reportes estándar son aquellos que pueden ser distribuidos como entidades *stand-alone*. Después de ser creados son distribuidos electrónicamente o a través de una red.

El acceso a datos Ad hoc representa un conjunto de requerimientos de usuarios finales que tienen cierta experiencia en **SQL**. Las herramientas de consulta Ad hoc, realizan operaciones directamente sobre las tablas (almacenamiento relacional).

El tercer grupo está formado por herramientas OLAP que permiten, manipular a los datos multidimensionalmente.

Era necesario un cambio de enfoque, para salvar costos y mejorar eficiencia. Se busco solucionar inconsistencias entre sistemas de transacciones de forma que los usuarios pudieran utilizar los datos en formas innovadoras. Analizando y sintetizando datos significativos surgiendo los sistemas de información gerencial.

Evaluación de herramientas de data warehouse.

La evaluación de cada herramienta debe descansar en las necesidades de la comunidad usuaria y en los siguientes lineamientos.

- Facilidad de uso. Es quizás la más importante de las características a evaluar. Está centralizada en dos áreas: construcción de reportes y flexibilidad de presentación. Los datos pueden cambiar su presentación y se pueden realizar investigaciones profundas sobre ellos para generar reportes.

- Ejecución. Involucra todo el ambiente del data warehouse, la interacción entre la base de datos y la metodología (**SQL** en software relacional) utilizada para tener acceso a los mismos.

- Múltiples fuentes de datos. Hace referencia a la posibilidad de analizar datos que propiamente no están dentro del data warehouse. Consultar información de un disco o de otra base de datos (sistema de transacciones u operacional).

- Seguridad de los datos. No sólo consiste de otorgar permisos a usuarios para consultar información, sino de asegurar la consistencia de la información ante los diferentes análisis a los que será sometida.

- Análisis integrado. Implica realizar operaciones de los datos de modo dimensional, incluyendo filtros sobre columnas o datas específicos.

Cara a cara, sistemas de información gerencial y sistemas de transacciones.

En las secciones anteriores se hablo de data warehouse y **OLAP**, viendo que este último tema está vinculado al primero. Sin embargo para comprender el avance que implica **OLAP** en tecnologías de información es necesario compararlo con **OLTP** (*On Line Transaction Processing*).

**OLTP** trabaja sobre los requerimientos de un sistema que es utilizado dentro de un ambiente operacional. Está enfocado a registrar las actualizaciones sobre archivos o tablas, tan pronto como las transacciones sucedan, o se reciba un mensaje de actualización. Entonces un proceso por lotes almacena a las transacciones involucradas y solo actualiza los registros necesarios en una fecha posterior.

Se debe destacar que la diferencia entre los sistemas que utilizan **OLAP** y aquellos que manejan la tendencia **OLTP,** radica en el objetivo. Mientras que en **OLTP** se busca el registro minucioso de las transacciones de un departamento de la organización, **OLAP** está enfocado a manejar los datos históricos contenidos en un Data warehouse o en un Data mart. En OLAP no se permiten actualizaciones por cada transacción.

En un sistema **OLTP**, se puede estimar matemáticamente el tamaño y tipo de consultas. Los sistemas OLAP, no pueden conocer estos datos con precisión, debido a que los datos pueden ser usados por los usuarios de distintas maneras.

Otra diferencia es la presentación. Muchos sistemas **OLTP** son implementados en RDBMS, utilizan **SQL** y es necesario conocer dicho lenguaje para obtener información del sistema.

Para tener una idea más amplia del tema, véase la tabla 1.6. Entre los aspectos mencionados resaltan: los usuarios, el enfoque, el acceso y las funciones. Estos resumen lo expuesto, en las secciones anteriores.

También son de suma importancia las operaciones realizadas y el diseño de la base de datos. De hecho una base de datos relacional subsiste tanto en un sistema de transacciones como en un sistema de información gerencial. En el capítulo siguiente se muestran las diferencias en el diseño de las tablas, usando normalización o desnormalización.

Desde el enfoque de data warehousing, la interfaces apropiadas están hechas en software **OLAP.** Manifestándose un binomio software **OLAP** y data warehouse. [Libro # 6].

| CARACTERÍSTICA | OLTP | OLAP |
|---|---|---|
| Orientación | Registro de transacciones. | Análisis de datos. |
| Usuario | Profesionales de Bases de datos | Ejecutivos, analistas y administradores. |
| Función | Operaciones día con día. | Apoyo en la toma de decisiones en base a información histórica. |
| Diseño de la base de datos. | Modelo entidad-relación. | Modelo dimensional |
| Vista | Detallada, plataforma relacional. | Multidimensional, con resúmenes. |
| Acceso | Lectura/escritura | Lectura |
| Enfoque | Hacia datos actuales y consistentes. | Hacia tomos de datos históricos. |
| Operaciones | Index/hash sobre llave primaria. | Lecturas en un modelo dimensional. |
| Prioridad | Disponibilidad y ejecución en alto nivel. | Flexibilidad y autonomía del usuario final. |
| Tamaño de la base | 100 Mb a 1 GB. | 100 Gb a 1TB. |

Tabla 1.6. Cuadro Comparativo entre OLAP y OLTP.

## Data warehouse y OLAP.

Data warehouse en niveles.

También llamado *tired data warehouse*, es una solución de arquitectura para el problema de control de costos en la administración de un data warehouse. Al igual que otras técnicas de ingeniería de software que practican información oculta para el usuario, disfraza las complejidades semánticas de los sistemas fuente. Aprovechándose las inversiones hechas en depuración e integración de datos y guarda los resultados intermedios en una reserva general de datos.

Su origen parte del trabajo realizado en IBM, en 1992. Donde se estableció una conexión entre el movimiento de información de sistemas fuente a los data warehouses. La estructura básica se compone de tres capas o niveles:

- Sistema de registro. El o los sistemas fuente, sistemas **OLTP** verticalmente integrados, o bien, datos externos de otros proveedores o fuentes.

- Base del data warehouse. La reserva general de datos integrados y racionalizados, luego de extraerlos de los sistemas fuente. Los datos se alinean de acuerdo a temas y no a semántica, presentándose en orden cronológico.

- Acceso al data warehouse. Es este nivel el sistema de data warehouse tiene contacto con los usuarios, sus herramientas y aplicaciones para la toma de decisiones. Las estructuras de datos pueden variar, desde almacenes de datos estratégicos para áreas temáticas hasta matrices de datos para propósitos específicos.

La pregunta esencial es ¿cuántos niveles deben emplearse y en qué momento dentro del ciclo de vida del proyecto?, el autor opina que existen tres tipos básicos que deben tomarse en cuenta:

- La volatilidad semántica del dominio de datos.
- La independencia de organización de la comunidad.
- El riesgo del ciclo de vida.

El principal factor que influye sobre la medida en que deben usarse los niveles, es el grado de inestabilidad semántica. A mayor grado de volatilidad semántica en los sistemas de transacciones, mayor número de niveles requerido.

Por ejemplo, en una compañía de artículos empacados, se utilizan datos provenientes de scanner. Los cuales son estables en su semántica por lo que no es necesario construir niveles adicionales. En cambio en la industria farmacéutica, los datos de las recetas son un desorden semántico. Un médico puede tener varias especialidades, dar consulta en diferentes localidades y recetar en cada una de ellas. Por lo tanto, en un nivel intermedio se filtran y clasifican los datos.

Si el dominio del primer caso los datos del scanner, se expandiera para convertirse en una cadena de valor del fabricante aumentaría la necesidad de niveles. La inestabilidad semántica, puede generarse desde dos fuentes: los sistemas de transacciones (abajo) y

los usuarios finales (arriba). Existen diferentes significados de la información entre los departamentos de la empresa, en ocasiones estos se resisten a una visón generalizada, solicitando una visión descentralizada.

Por lo tanto a mayor grado de independencia de organización en la comunidad de usuarios, mayor número de niveles. Sin embargo, la implementación de niveles adicionales es un riesgo para el ciclo de vida del data warehouse. Por ello, se recurre a construir niveles de aislamiento de datos entre las etapas de carga de los datos en el data warehouse. En la figura 1.19, se muestra la configuración de un data warehouse de 3 capas. [Revista #4]

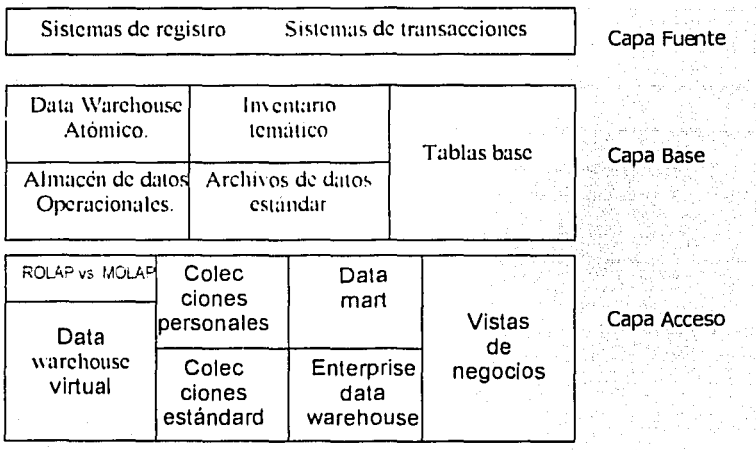



Figura 1.19. Data Warehouse de tres niveles.

Si nos enfocamos en el proceso de extracción de datos en secuencia con el valor del data warehouse, en cada proceso de extracción los datos del sistema fuente están: depurados, racionalizados, normalizados y orientados al tema. Lo cual no es común y permite la creación de almacenes intermedios, que pueden ser virtuales al definirse la semántica de cada nivel e implementarse a través de interfaces de software.

Minería de datos.

Hasta el momento el lector, puede identificar la necesidad, de algo más que un sistema operacional, tales sistemas, son implementados en data warehouse y utilizan software **OLAP**, para cumplir con sus funciones.

Los data warehouses, llevan consigo una evolución, en donde nuevos conceptos han surgido para cumplir con los requerimientos más complejos de los usuarios. De hecho, la gran cantidad de datos que involucra un análisis, ha llevado a la aparición de un grupo de trabajo denominado "knowledge workers", conformado por estadísticos y analistas de mercado. Los trabajadores del conocimiento necesitan aprender del sistema de su interés. Esta tarea implica una dedicación constante y experiencia que toma varios años de

formación. Un programa que ayude a la construcción de estos conocimientos ha impulsado la incursión de la minería de datos, que se basa en los mineros de datos.

Los mineros de datos, son programas encargados de buscar situaciones interesantes: anomalías, tendencias o desviaciones en la base de datos de una organización. Su labor es considerablemente intelectual, al nivel de un gerente o planificador. Su tarea depende de criterios establecidos y de su capacidad de aprendizaje a las preguntas formuladas por el usuario. Dicha capacidad nos lleva hacia el concepto de inteligencia artificial.

Los mineros constan de tres elementos:

**Extractor**. Programa que obtiene datos de interés, de la base de datos.
**Revisor**. También llamado módulo verificador que mediante análisis matemático o estadístico, determina si hubo algo interesante en el subconjunto de datos extraídos.
**Arbol de conceptos**. Es la guía del programa extractor, que junto con los criterios de interés indica los datos que son convenientes para estudio.

El árbol de conceptos, es una jerarquía o arreglo de atributos y los mineros de datos requieren una tabla que describa los niveles que la integran. Su búsqueda es asíncrona, presentándose en momentos de poco procesamiento de información o en horas no laborales de los usuarios.

Existen varios tipos de mineros: el buscador dirigido diferido, el buscador de índices de productividad, aquel que utiliza funciones booleanas y aquellos que utilizan correlación entre señales de tiempo.

El primero utiliza un criterio o filtro, programado en C o en lenguajes similares a **SQL**, tomando en cuenta al árbol de conceptos. El segundo tiene una función a evaluar cuyos resultados son tomados como índices de desempeño o productividad. El minero reporta las n mejores regiones o las m peores, por ejemplo.

El tercer tipo determina si hay anomalía (1) o éxito (0), utilizando clases de equivalencia que son evaluadas por una función booleana. El último tipo de minero de datos, es más difícil de programar. Busca la correlación entre variables definidas en intervalos de tiempo. Siendo estas endógenas o de la organización y otras veces son parte del entorno o exógenas. Todos ellos almacenan la información en tablas o en archivos donde el usuario puede verlos y manipularlos si así lo desea. El proceso de aprendizaje del minero depende de su interfaz programada a este respecto y la comunicación con el usuario en términos de negocio. La interfaz debe apoyarse en un modelo de decisión que permita dar significado a las peticiones del usuario.

Sin embargo, los mineros enfrentan dos problemáticas: la carencia de datos cuando hay inconsistencias o no existe un registro organizado de los mismos que el minero solicita y saltos o discontinuidades en las variables, generados por un cambio de definición por parte del usuario. [Revista #5].

# Capítulo II.

# El modelo dimensional.

Un sistema de data warehouse, depende en gran medida de la arquitectura y el software sobre el que trabaja, para conseguir satisfacer las necesidades del usuario.

El data warehouse para su implementación, requiere dos tipos de diseño: lógico y físico. El diseño físico comprende adoptar la arquitectura cliente servidor o de múltiples niveles para lograr su objetivo.

El diseño lógico define las relaciones entre los objetos de la base de datos, el contenido de estos y da soporte a las consultas que se realizarán sobre ellos. Está enfocado hacia las necesidades informativas de los usuarios que son básicamente análisis multidimensional y resúmenes de datos. Un diseño bien estructurado permite el crecimiento y cambio en las necesidades de los usuarios.

En data warehousing un diseño bien estructurado se adapta a todas las posibles consultas, tanto para **ROLAP** como para **MOLAP**. En **MOLAP** las consultas están definidas anticipadamente y el diseño de un sistema de DSS conlleva conocimientos técnicos del software. En **ROLAP** las especificaciones del software no son demasiado importantes, el diseño utiliza conocimientos teóricos de bases de datos relacionales buscando satisfacer las preguntas improvisadas de los ejecutivos.

El análisis multidimensional implica efectuar operaciones sobre las dimensiones, para ello necesita una estructura lógica sobre la cual manipular los datos del data warehouse. En **MOLAP** no hay una estructura lógica definida como estándar. En **ROLAP** se utiliza el modelo dimensional.

Este modelo consiste de un diagrama que contiene:

- Dimensiones
- Atributos
- Relaciones
- Medidas o procesos.

Mientras que el diseño lógico para una base de datos relacional, descansa sobre el modelo entidad-relación, que está asociado a altos niveles de normalización. El modelo dimensional requiere niveles básicos de ésta. Las formas normales, son todavía útiles en un modelo dimensional con un enfoque distinto.

En apariencia es un esquema de tablas relacionadas, solo que no existen entidades, sino que se habla de dimensiones. Prácticamente muchas de las entidades que existen en un diagrama entidad-relación se transforman en dimensiones dentro de un modelo dimensional.

Los procesos y sus mediciones registradas en los sistemas de transacciones son el centro del modelo. La información de los procesos es organizada en tablas dimensión alrededor de una tabla proceso. Las dimensiones son las características cuantitativas y/o cualitativas del proceso. De acuerdo con su forma el modelo dimensional recibe el nombre de: esquema estrella, copo de nieve y constelación (galaxia). Su objetivo es descubrir unidades atómicas de información y todas las relaciones entre éstas últimas. Sin embargo puede enfocarse a un nivel particular de los datos formando agregaciones.

**Dimensiones.**

Dimensión y Jerarquía.

Primeramente, se debe saber ¿Qué es una dimensión?. Una dimensión es una categoría de datos, que describen a un proceso perteneciente a un sistema de data warehouse. En ocasiones las dimensiones tienen jerarquías, o un conjunto de niveles que la describen en distinto detalle. Así para una dimensión de artículos electrodomésticos se tendrían, los siguientes atributos: nombre, marca, modelo, voltaje, potencia, frecuencia, medidas y color.

Los atributos de un modelo entidad-relación algunas veces se transforman en entidades dentro del mismo. En un modelo dimensional pueden coexistir atributos y entidades en una dimensión y formar parte de una jerarquía. Tal es el caso de ítem que para un modelo entidad-relación es una entidad, mientras que en el modelo dimensional la entidad ítem y sus atributos (color y tamaño) forman parte de la dimensión producto.

Una jerarquía estaría integrada por las clasificaciones que agrupan a un electrodoméstico: electromecánicos, eléctricos, electrónicos y digitales (que utilizan algún procesador). Dentro de una jerarquía simple a cada nodo "hijo" le corresponde un nodo "padre". Puede ser descrita conceptualmente mediante un árbol.

Una jerarquía es un conjunto de atributos de una tabla dimensión parcial o totalmente ordenado. El conjunto de atributos para una dimensión ubicación está totalmente ordenando si se describe como: calle<ciudad<estado<region.

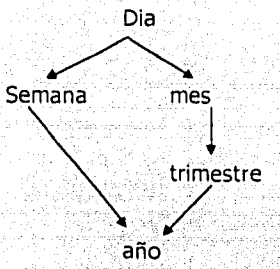



Figura 2.1 a. Orden parcial para la jerarquía de una dimensión tiempo.

El conjunto de atributos para la dimensión tiempo, estará parcialmente ordenado si se describe como en la figura 2.1a, o bien, como día<{ mes<trimestre; semana } < año. Además de día, se tienen las combinaciones: día-mes, día-trimestre y día-semana.

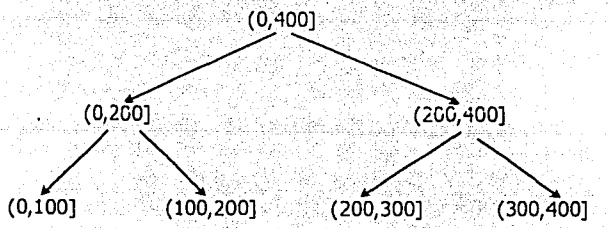



Figura 2.1 b. Jerarquía para el precio de un producto.

Las jerarquías también se aplican sobre conjuntos de valores discretos, el precio de un producto por ejemplo, puede estar ordenado en base a intervalos mixtos (x,y] como lo muestra la figura 2.1b. [Libro # 2] y [Página Web # 2].

Relaciones entre atributos de una dimensión.

Mientras que en un modelo entidad-relación las relaciones se presentan entre entidades, en un modelo dimensional se presentan entre los atributos de las dimensiones.

Son representadas mediante arboles. Pueden existir dos padres asociados a un mismo hijo, o un padre que tiene uno o más hijos. Son clasificadas en dinámicas y estáticas.

Relaciones dinámicas: Aquellas que cambian, como lo muestra la figura 2.2, donde el atributo ítem cambia de departamento en un lapso de tiempo dado. Los cambios en las relaciones de una jerarquía obedecen a reestructuraciones en la organización, la dimensión tiempo es raramente afectada por lo que las relaciones entre sus atributos son estáticas.

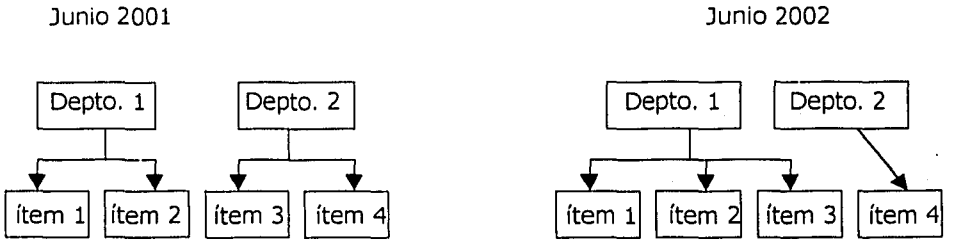



Figura 2.2. Cambios en una jerarquía producto.

Relaciones estáticas: Son aquellas que no cambian como la relación entre el atributo mes y el atributo año, en una dimensión tiempo.

Las relaciones son establecidas por medio de tablas de relaciones o por medio de relaciones transitivas. Si el atributo A está relacionado con el atributo B, y B está relacionado con C, entonces A esta relacionado con C.

Sin embargo las relaciones transitivas no se observan a simple vista dentro del diagrama. Son reconocidas al recorrer el diagrama de una jerarquía definida. Cuando dos o más atributos no pertenecen a la misma dimensión, la relación entre ellos es identificada al consultar la tabla proceso.

Las dimensiones pueden tener múltiples jerarquías como lo muestra la figura 2.1 a. La dimensión tiempo implica una jerarquía y existen varias formas de interpretar la precedencia de sus atributos. [Libro # 9]

Operaciones con dimensiones.

En el capítulo anterior, al describir las funciones ejecutadas por el software **ROLAP** y **MOLAP** se comento la posibilidad de mostrar datos por ciudad o por región. En esta sección se explica cuales son las operaciones entre dimensiones y como están trabajan sobre la jerarquía de una dimensión.

En la figura 2.3, tenemos un cubo con las dimensiones: temp (tiempo en trimestres), ubic (ubicación en ciudades) y artc (artículos electrónicos). Con sus celdas se pueden realizar las siguientes operaciones:

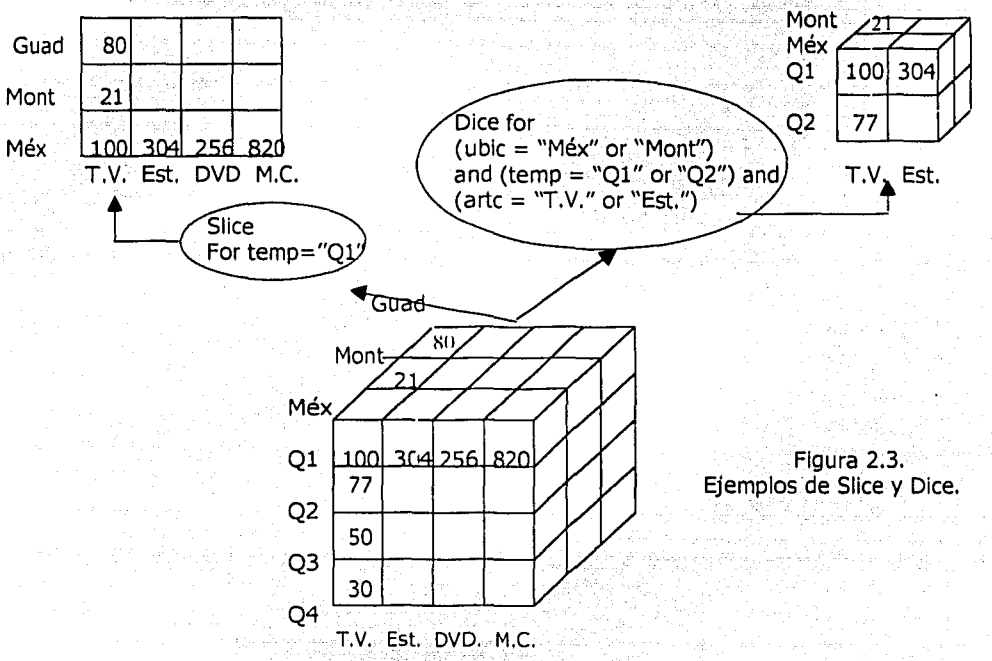



Figura 2.3.
Ejemplos de Slice y Dice.

Slice.

Selecciona un fragmento o toda una dimensión sobre un cubo de n dimensiones. Obteniéndose un subcubo de información. En la figura 2.3, se especifica un valor de la dimensión tiempo. El subcubo resultante es una matriz con las dimensiones artc y ubic.

Dice.

Consiste en ejecutar una selección sobre dos o más dimensiones. Obteniéndose un subcubo de información más específico que en un *slice*.

Pivote.

Consiste en girar las dimensiones de un cubo de datos, con el fin de obtener otra perspectiva de estos. Como se puede ver en la figura 2.4, se realiza un pivote con las

dimensiones artc y ubic. Se trata de girar una dimensión dada a 90°, para cubos de tres dimensiones se tienen 6 vistas y en general para n dimensiones se tienen n(n-1) vistas.

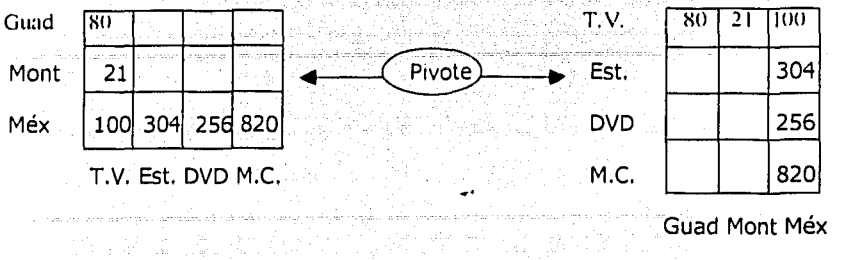



Figura 2.4.
Pivote entre las dimensiones ubic y artc.

Roll-Up.

También llamada drill-up, es una agrupación de datos contenidos en un cubo. Se aplica, por reducción de dimensiones o recorriendo de un nodo hijo hacia un nodo padre la jerarquía de una dimensión. En la figura 2.5, puede observarse una perspectiva de nivel más alto a ciudad donde los datos son agrupados por región, en la jerarquía de la dimensión ubic. La cual totalmente ordenada es: calle<ciud<estado<region.

Cuando se ejecuta Roll-Up por reducción de dimensiones, una o más de ellas son removidas del cubo de datos. Con solamente dos dimensiones localidad y tiempo, al realizarse una agrupación por tiempo, se obtendría un vector dimensión (o columna) localidad.

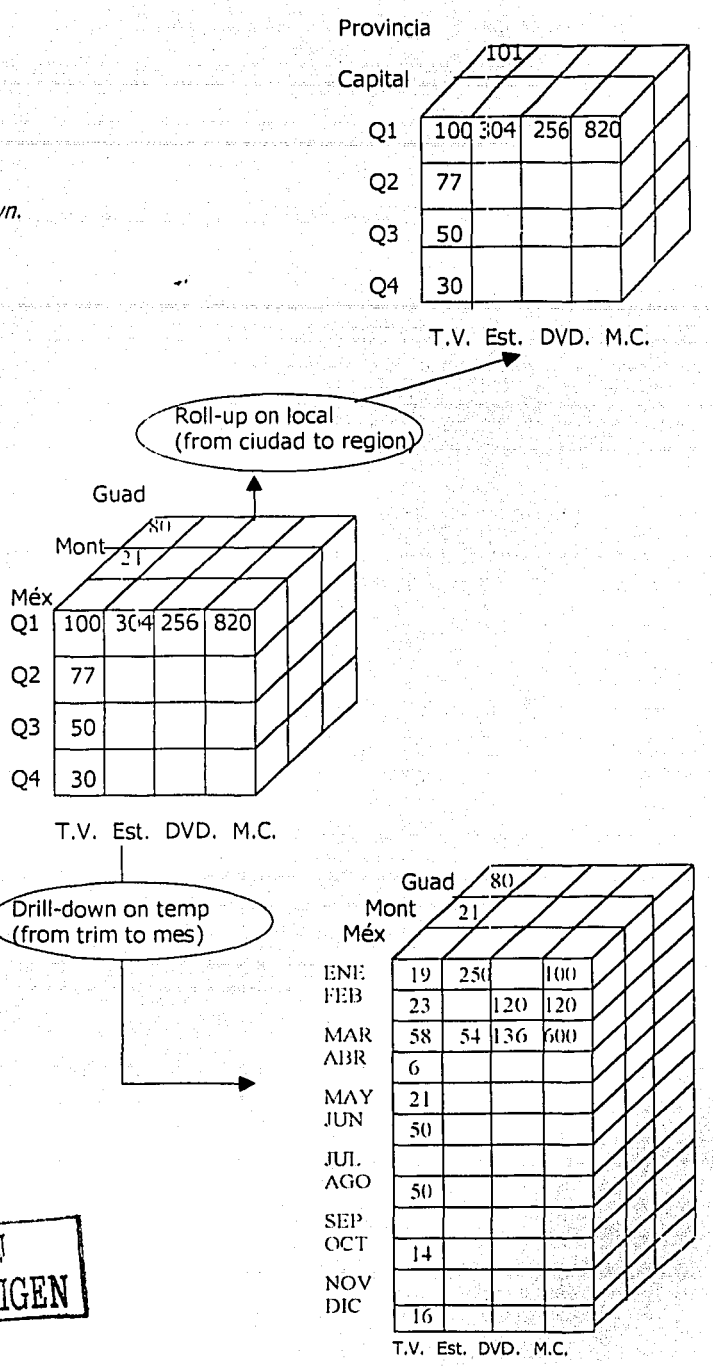
Al establecer una jerarquía es necesario simular operaciones de *roll-up* y *drill-down* para hallar relaciones convenientes entre los atributos de una dimensión. Si en la dimensión ubic se mezclan estados y ciudades en un mismo nodo, se tendrán errores al realizar *roll-up*. Pueden acomodarse cada uno en dimensiones separadas pero los datos serán esparcidos. La solución es colocarlos como dos nodos distintos en la misma jerarquía.

Dril-Down.

Operación inversa a *Roll-up*. Se realiza recorriendo la jerarquía de una dimensión de un nodo padre a un nodo hijo o introduciendo dimensiones adicionales. Para la figura 2.5, la jerarquía afectada temp ordenada totalmente corresponde a día<mes<trim<año. Los datos no se muestran por trimestre sino que están distribuidos en meses. [Libro # 2]

Figura 2.5.

Ejemplos de *Roll-up* y *Drill-down*.



Nivel de detalle.

Gr*ain* en inglés, se refiere a los nodos presentes en un diagrama de árbol para la jerarquía de una dimensión. Entre más fino sea el nivel de detalle, es decir a mayor número de niveles el usuario tendrá la posibilidad de efectuar operaciones de *roll-up* y *drill-down* más sofisticadas. Un menor nivel de detalle dentro de una jerarquía limita la capacidad de *drill-down* y *roll-up*.

El nivel de detalle hace factible una vista. Al diseñar un modelo dimensional, es conveniente identificar las combinaciones de niveles de detalle entre las dimensiones de un proceso, evitando posibles errores al combinar niveles distintos en los reportes que proporcione el sistema. El nivel de detalle interviene en la creación de dimensiones conformadas y de un grupo especial de modelos dimensionales llamados agregaciones.

Manipulación de datos en función del tiempo.

El tiempo es la dimensión más común entre los modelos dimensionales. Los usuarios finales, generalmente quieren ver sus datos de interés en varios periodos de tiempo. Por esa razón, se ha investigado sobre un tipo de datos especialmente diseñado para series de tiempo. Este tipo de dato se define como una serie de números que representan a una variable particular referida al tiempo, como los días de la semana inglesa o los meses del año en un balance.

Dentro de una celda para hojas de cálculo se almacena un valor numérico. En **MOLAP** se busca representar los datos históricos de 10 años en una celda, eliminando la necesidad de utilizar una dimensión para manipularlos como sucede en **ROLAP**. Se modelan tipos de datos para series de tiempo con características como: intervalo de tiempo en el que se establecen los datos y reglas para modificar al intervalo con otra medida de tiempo.

No ha sido una idea viable debido a las complejidades de programación que genera afectando el desempeño de los modelos dimensionales, ya que en muchos ocasiones los datos no se encuentran esparcidos uniformemente. Por otro lado se requiere un servidor hábil para manejar periodos fiscales y años en curso.

ROLAP.

El manejo del tiempo en **ROLAP** no incluye este tipo de datos. Generándose una dimensión tiempo con una jerarquía de varios niveles de detalle, haciendo referencia explícita a cada uno de ellos. Presentándose recursos de programación y mantenimiento adicional para las aplicaciones asociadas con la dimensión.

La mayoría de los modelos maneja una jerarquía pequeña o significativa. Sin embargo, cuando el modelo es modificado para manejar la información de semanas a meses, se pierde la posibilidad de analizar los datos semanalmente. Aumentando el uso de recursos de memoria por la nueva distribución de los datos y tiempo extra para cargar la base de datos.

Si en una semana se cruzan dos meses, este detalle no se maneja en el nivel semana, sino en el nivel año. Puesto que un año civil contiene 52 semanas se construye un procedimiento que controla la incidencia de dos meses en una semana sin afectar la jerarquía. [Página Web # 2]

Variables y dimensiones.

Son mediciones numéricas provenientes de sistemas de transacciones. En ocasiones son consideradas como dimensiones especiales. Sobre todo cuando están asociadas a otras dimensiones. El número de unidades vendidas es un ejemplo, puede analizarse por región, por producto o tipo de cliente. Otras como el precio de un producto, son constantes para una región y tipo de cliente, siendo necesario solamente dimensionarlo por producto.

Variables como el precio de un producto no pueden ser tratadas como dimensión. Si fuese así, al aplicársele a todas las dimensiones de un modelo los datos se esparcirían demasiado. Lo más recomendable es relacionarla con las dimensiones relevantes. Convirtiéndose en una variable dimensionada independientemente, una herramienta muy útil en bases de datos multidimensionales.

Las variables se definen como una relación matemática de otras variables llamándose variables complejas. Las operaciones utilizadas para definirlas son: sistemas de ecuaciones, parámetros estadísticos y series de tiempo.

Las variables intervienen en la consolidación de modelos dimensionales. Variables aditivas como el número de unidades vendidas se registran diariamente, al sumarse aritméticamente ofrece un total mensual en un *roll-up*. La naturaleza de las variables permiten al modelo dimensional adaptarse a varios periodos de tiempo.

Las variables forman parte de los metadatos. Estos contienen información sobre conversiones de moneda, unidades de medida y descripciones generales. Las variables derivadas son un tipo de variables complejas acumuladas en la base de datos o presentes durante la ejecución de algún reporte. Su utilidad radica en que optimizan la ejecución del servidor de datos, al no tener que calcularlas improvisadamente. [Página Web # 2]

Clasificación de variables.

De acuerdo a su capacidad para ofrecer un resultado real y conveniente, ante operaciones realizadas sobre dimensiones una variable puede ser:

Aditiva. Variable que puede relacionarse con otras, para varios reportes con distintos niveles de detalle.

No aditiva. Variable que no puede relacionarse con otras. Tal es el caso de los porcentajes que no son incluidos fácilmente en una dimensión, pues al aplicar *roll-up* sobre la dimensión producen datos erróneos. Sin embargo, dichos porcentajes son radios de medición de variables aditivas y pueden generarse reportes a partir de estas.

Semi aditiva. Variable cuyo número de relaciones con otras variables es limitado.

La aditividad es la habilidad para realizar *roll-up* sobre una variable. Si en vez de realizar un reporte mensual de ventas se requiere uno anual, las ventas por mes son sumadas aritméticamente para obtener el resultado.

Al aplicar *roll-up* o *drill-down* sobre un porcentaje la suma de estos en un lapso de tiempo es carente de significado. Se descompone en variables aditivas dentro de un modelo dimensional y se calcula el porcentaje total buscado.

Un ejemplo de variables semi aditivas son: cantidades inventariadas y balances de cuentas. No son buenas candidatas para realizar *roll-up* sobre la dimensión tiempo. Para efectuar reportes de distinta periodicidad, se requiere el uso de estrategias.

En el caso de los balances de cuenta, no se puede trabajar sobre la suma de balances diarios. Se pueden obtener los intereses a pagar en un mes en base al promedio de los balances.

Las variables no aditivas, están presentes en varios de los modelos dimensionales. En el capitulo 3 se exponen ejemplos que las utilizan. [Libro # 3]

Datos esparcidos.

Cuando se añaden nuevas dimensiones a una base de datos multidimensional, el número de celdas crece rápidamente. Generalmente entre un 80 y 95 por ciento de las celdas de una **MDDB** están vacías. Dicho fenómeno recibe la denominación de "base de datos poblada esparcidamente" o simplemente esparcida.

Otro tipo de esparcimiento de datos sucede cuando varias celdas comparten el mismo dato. Como se comento el precio de un producto es constante y puede llegar a almacenarse muchas veces. Lo aconsejable es capturarlo una sola vez junto con el número de días en que se mantendrá vigente. Para el almacenamiento de los datos esparcidos se utilizan técnicas como RAID y técnicas que involucran recursos de programación al cargar los datos en el data warehouse.

En el tema de procesos combinados, se expone un modelo para el status de compromisos, pagos y presupuesto. El modelo busca eliminar ceros innecesarios, otra forma de datos esparcidos. Con este ejemplo se muestran las alternativas realizables en un modelo dimensional ante bases de datos con problemas de almacenamiento. [Página Web # 2] y [Libro # 13]

**Modelo dimensional.**

Una base de datos relacional utiliza un modelo entidad-relación. Un data warehouse requiere un esquema conciso que facilite el análisis de datos en línea. La estructura más adecuada es el modelo dimensional, en cualquiera de sus modalidades: estrella, copo de nieve y constelación.

Cada uno describe un proceso inherente a un departamento, son utilizadas las tablas proceso y las tablas dimensión.

Se tiene acceso a la información con **SQL** y restricciones sobre las dimensiones solicitadas. Los metadatos permiten vincular procesos, por medio de una o más dimensiones en común. Si no hay compatibilidad en el nivel de detalle de las dimensiones se dice que tenemos dimensiones no conformadas.

Las dimensiones conformadas son aquellas que fuerzan a los datos de dos sistemas, el sistema de transacciones y el sistema de información gerencial a compartir atributos idénticos. Su objetivo es evitar conflictos en el nivel de detalle de una tabla dimensión.

Por ejemplo, si una fuente de datos está basada en semanas y el sistema de información gerencial utiliza meses como nivel de detalle para un balance será necesario incluir en ambos el menor nivel de detalle, que en este caso es día.

En la figura 2.6, se muestra un modelo dimensional en nivel cero, un modelo donde no se ven las tablas y sus relaciones, sino las tablas dimensión y la tabla proceso.

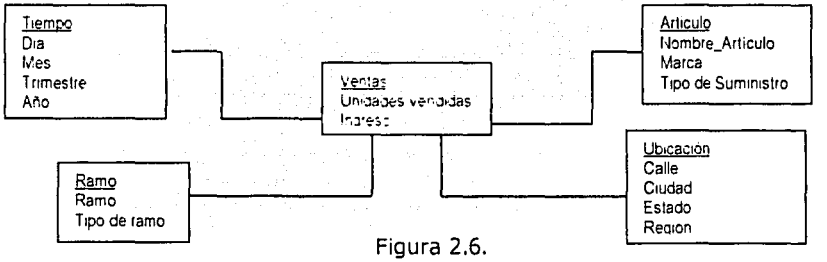



Figura 2.6.
Modelo dimensional para Ventas.

Tabla dimensión.

Una tabla dimensión (*dimension table)* contiene los atributos que describen un proceso. Es inconveniente normalizarla pues disminuye su eficacia en la ejecución, al repartir los atributos dimensionados en varias tablas, lo que conlleva a consultas más complejas. Conservar las dimensiones desnormalizadas se traduce en acceso rápido a las jerarquías de las dimensiones.

En la mayoría de los casos una tabla dimensión es más pequeña que su tabla proceso asociada, pero crecen cuando hay cambios en las dimensiones. En tales casos se

puede reducir su tamaño separando algunos atributos en otra tabla denominada dimensión separada.

Lo más difícil es representar una relación recursiva. Dentro de una dimensión vendedor, el jefe del vendedor está incluido en la misma tabla. Si se desea agrupar esta dimensión es necesario identificar el nivel más alto de la jerarquía y el número de niveles hacia abajo del nivel que se quiera reportar.

A fin de navegar en este árbol hasta llegar al nivel requerido. Aquellos renglones que correspondan al vendedor de mayor rango en el árbol, contienen valores no aplicables en las columnas de menor jerarquía. [Libro # 3]

Tabla de relaciones.

Son aquellas que contienen las llaves primarias de dos o más atributos de una dimensión, definiendo la asociación existente entre ellos. Su funcionalidad radica en definir la jerarquía de una dimensión a fin de navegar en ella con *drill-down* o *roll-up*.

Cuando se presentan relaciones muchos a muchos entre dos atributos, la relación se establece con una tabla que físicamente almacena el contenido de las llaves primarias de ambos. Si la relación es una a muchos la relación se establece en la tabla del atributo hijo. [Libro # 9]

Tabla proceso.

Un proceso es una acción cuantificable del negocio como: ventas, embarques y rentas. Muchas de sus mediciones son aditivas, aunque otras no lo son.

Una tabla proceso (*fact table)* en la practica es extremadamente grande, contiene atributos y llave foráneas para cada tabla dimensión asociada. Una característica distintiva es su dispersión, ya que no contienen un renglón para cada combinación de renglones de alguna tabla dimensión. Gracias a esta característica su tamaño es controlable.

No requiere una llave primaria asignada por el sistema, su llave primaria está formada de varias llave foráneas. Cada renglón es identificado de acuerdo con las dimensiones involucradas en su nivel de detalle. [Libro # 3].

Esquema estrella.

Consiste de una tabla central denominada tabla proceso y las tablas asociadas a la principal, reciben el nombre de tabla dimensión. Los atributos de estas últimas pueden constituirse dentro de una jerarquía (orden parcial o total).

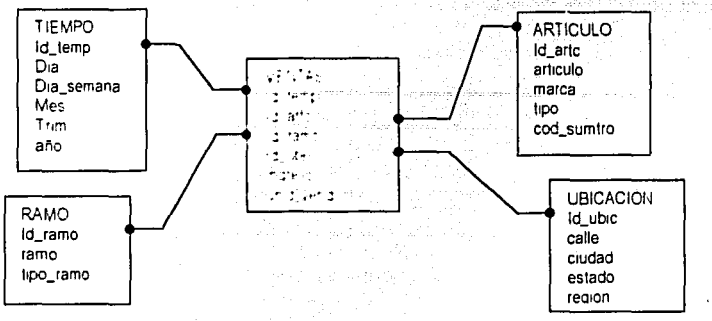



Figura 2.7.
Esquema estrella para Ventas.

Cada dimensión tiene su propia tabla de relaciones. Al tener pocas tablas se tienen pocas juntas entre esta. Soporta relaciones muchos a muchos entre atributos. Disminuye la lectura de tablas dimensión y tablas proceso, para atributos relevantes. La tabla proceso es quien maneja varios niveles de detalle.

En un esquema estrella solo se requiere una junta, para establecer la relación entre la tabla proceso y las tablas dimensión. Realiza consultas simples y rápidas, ya que toda la información acerca de cada nivel es almacenada en un renglón.

Es importante planear esquemas estrella para cada proceso del data warehouse, sin dejar de tomar en cuenta sus interrelaciones. Una vez hecho esto, a cada área descrita por un esquema estrella se le denominará data mart. [Libro # 3] y [Libro # 9]

Esquema copo de nieve.

Es una variante del esquema estrella, figura 2.8, donde algunas dimensiones son normalizadas, los atributos de las tablas dimensión tienen sus propias tablas normalizadas. El patrón resultante del diagrama parece un copo de nieve.

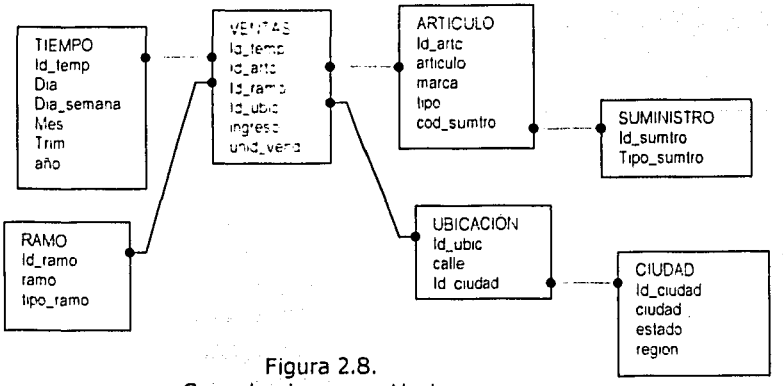



Figura 2.8.
Copo de nieve para Ventas.

Cada dimensión tiene su propia tabla relación, esta puede estar contenida dentro de la tabla dimensión o ser independiente. Sin embargo, no soporta relaciones muchos a muchos entre atributos y se especifica el nivel de detalle para todas las tablas. Muestra relaciones entre atributos que no muestra el esquema estrella, pero en casos detallados requiere la junta de varias tablas para obtener resultados satisfactorios.

La cantidad de juntas necesarias para hacer una consulta, así como su tamaño representan un inconveniente, para diseñar un data warehouse. Puesto que la tabla proceso maneja varios niveles de detalle. [Libro # 3] y [Libro # 9].

Esquema Constelación.

Las aplicaciones sofisticadas pueden requerir múltiples tablas proceso para compartir tablas dimensión, a estos diagramas se les ve como un conjunto de estrellas, llamándolos constelaciones o galaxias, figura 2.9.

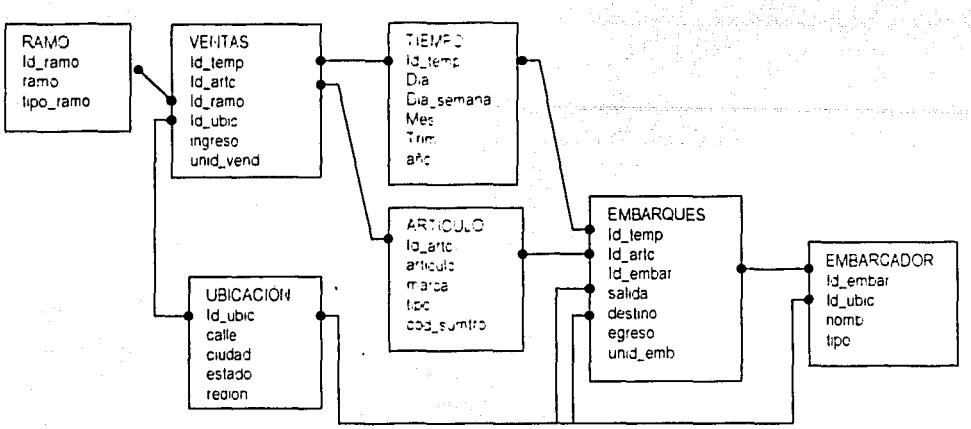



Figura 2.9.
Constelación formada por Ventas y Embarques.

Estos esquemas implican mayor número de juntas pero son candidatos de manejar en sus tablas proceso un nivel de detalle enfocado a satisfacer consultas predefinidas.

Observaciones de la tabla proceso.

En ocasiones no es necesario grabar todas las medidas en una tabla proceso, sólo con grabar la relación entre dimensiones es suficiente para medir un proceso. Entonces se utilizan tablas denominadas tablas proceso simples, las cuales graban reportes de datos actuales. Algunas veces se les añade un atributo con valor uno, para efectuar una suma sobre este y obtener el total de renglones que la forman en vez de usar la función count distinct( ) de **SQL**.

Una tabla proceso con frecuencia falla al grabar renglones que carece de disparadores ***(triggers)*** para vigilar su integridad. Para evitar grabar información que no convenga a las reglas del negocio, se crean las denominadas tablas de soporte con estructura similar a una tabla proceso simple, cuyo objetivo es forzar el registro de una relación indispensable entre dimensiones. [Libro # 3]

Capacidades de resumen de un esquema estrella.

Para llegar a conocer las capacidades de resumen de un esquema estrella se recurre ocasionalmente a establecer un ordenamiento parcial de sus dimensiones. Se construye una latice de hipercubos o cuboides, donde cada renglón de nodos representa un nivel de agrupación. El hipercubo con menor nivel de agregación es llamado cuboide base y el de mayor nivel es llamado cuboide ápice o cima. [Libro # 2]

La figura 2.12 muestra la latice asociada al modelo dimensional para ventas figura 2.6.

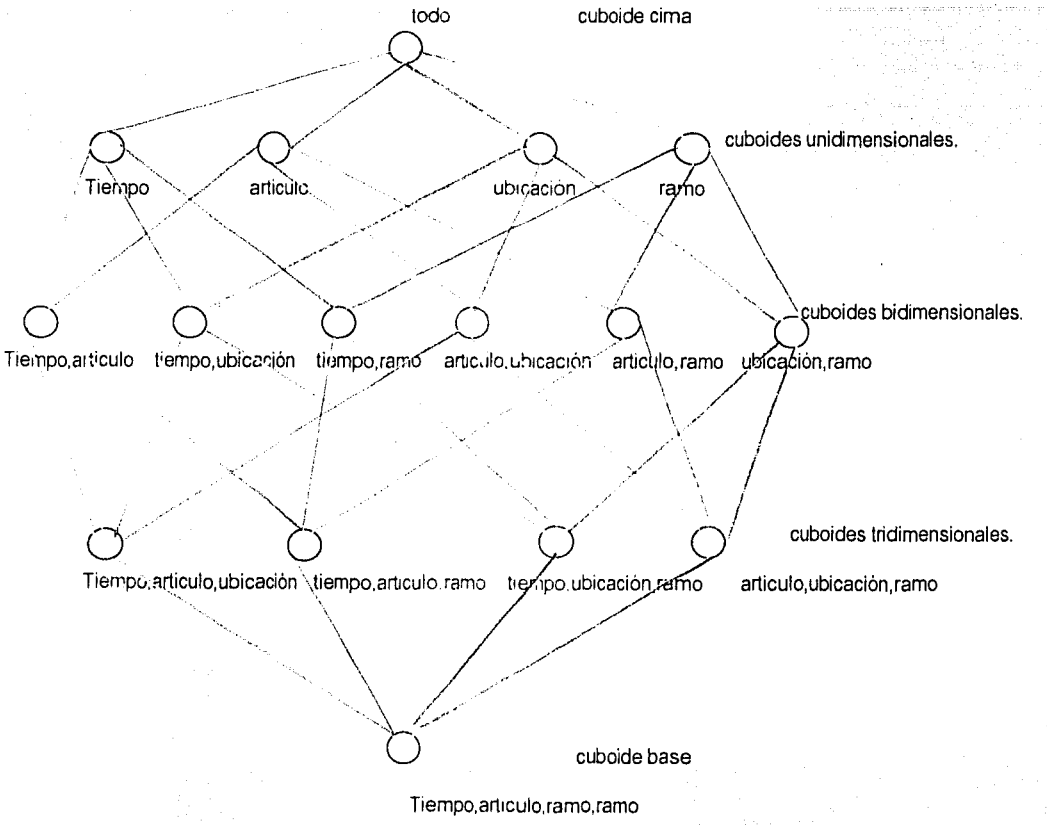



Figura 2.12.
Latice de las dimensiones del esquema Ventas.

Esta latice puede utilizarse en **MOLAP** para conservar los datos de consultas que involucren alguna de las combinaciones de las dimensiones. En el caso de una jerarquía de atributos, también se puede construir una latice de estos considerando las combinaciones más practicas.

**Dimensiones especiales.**

Dimensiones degenaradas.

*Degenerate dimensions* son frecuentemente incluidas en una tabla proceso. Aparecen como atributos de dicha tabla y no son incluidas como tablas en el esquema estrella, pues comparten atributos con alguna tabla dimensión presente y cumplen con un propósito específico en la descripción del proceso. Generalmente son números de identificación de documentos o transacciones.

En el esquema de la figura 2.10, se tienen dos dimensiones degeneradas, num_orden y num_linea orden, que son incluidas para diferenciar aquellas ordenes donde coincidan, el producto, la fecha y efectuadas por el mismo vendedor. [Libro # 3]

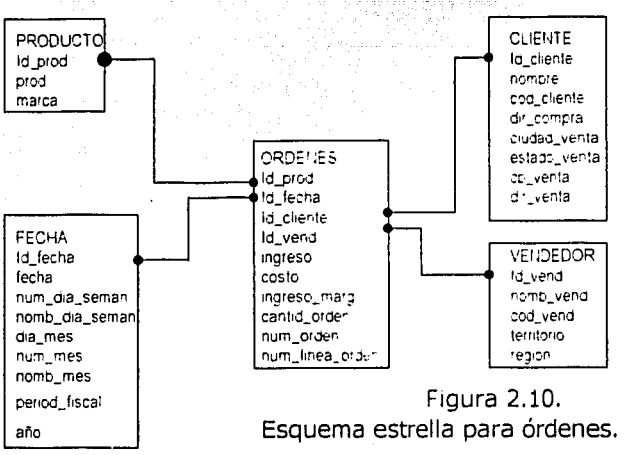



Figura 2.10.
Esquema estrella para órdenes.

Dimensiones de lento cambio.

En un **OLTP** cualquier cambio se sobrescribe, lo que no conviene a los usuarios de un data warehouse. Para las dimensiones cuya información cambia lentamente se tienen tres tipos de modificaciones.

El primer tipo se presenta cuando hay un error en la captura de la información, entonces se sobrescribe el dato correcto.

El segundo tipo, sucede cuando se requiere tener datos actuales y sus precedentes. Entonces se escriben los nuevos datos en otro renglón y el sistema deberá ser capaz de vincularlos para realizar los reportes requeridos por el usuario. Este error puede generarse por la inclusión dentro del esquema estrella de una llave primaria perteneciente a un sistema fuente, que no sufrirá modificaciones como los atributos de su tabla.

El último, implica la modificación de alguna tabla cuando los requerimientos futuros del sistema lo hagan necesario.

El esquema estrella propuesto en la figura 2.10, tiene a cliente una dimensión de lento cambio del segundo tipo. Al cambiar la dirección de un cliente, es necesario añadir un nuevo renglón que la contenga y vincularlo a la tabla proceso original.

Para lograrlo es necesario contemplar alguna de las dos siguientes alternativas:

1.Llaves derivadas o *derived keys*, son una concatenación de la llave producida con algunos dígitos que indican su versión. Por ejemplo, si Id_cliente está formada por 6 caracteres, debe añadirse otro(s) dígito(s), según convenga, para indicar la nueva versión de ID_cliente[1], dicha llave será añadida a la tabla proceso, indicándole cual de los renglones de la tabla dimensión contiene los datos actuales.

Para aplicarla se necesita cumplir con las siguientes condiciones: (1) el sistema fuente modifica los atributos de la tabla dimensión, (2) el sistema fuente no altera la llave primaria de la tabla dimensión y (3) la modificación de alguno de los atributos de la tabla dimensión es considerablemente importante para ser archivada en el data warehouse. Como desventajas, se tienen por un lado los recursos de programación para vigilar la versión de la llave primaria y el exceso de caracteres que ésta contiene que aunado al número de registros consume recursos de almacenamiento.

2.Llave entera asignada de forma secuencial. Id_cliente queda comprendida en un rango de números enteros que identifican a un cliente. Tales llaves son anónimas, más pequeñas y no transportan información por ellas mismas. Se considera que un entero de 4 bytes es suficiente para un aplicación con tablas de tamaño medio. Sin embargo como desventaja, presenta la necesidad de consultar una tabla de referencias cruzadas para asignar la llave correcta. Aunque la administración de dicha tabla es administrativamente menos compleja que la administración del número de versión de una llave derivada.

El uso de este tipo requiere que la dimensión sea generalizada. La creación de dichas llaves es usualmente responsabilidad del equipo de data warehousing y siempre requiere metadatos para conservar las llaves generalizadas que han sido utilizadas.

El segundo tipo automáticamente crea particiones históricas y una aplicación no debe contener restricciones sobre fechas efectivas.

Para ejemplificar el tercer tipo, considere que para el esquema de la figura 2.10, se quieren comparar las comisiones de un vendedor ante una nueva redistribución regional. Se necesita añadir dos dimensiones nuevas una llamada territorio_ant y otra llamada territorio_nuevo, en la primera se vaciaran los datos actuales y en la segunda los datos futuros. [Libro # 17].



Minidimensiones.

Supóngase que se tiene un modelo dimensional, donde es necesario registrar características del cliente como: edad, estado civil, sexo, ingresos y conducta de compra. El problema es el número de combinaciones entre las características del cliente. Una solución, es separarlas en uno o más conjuntos llamados minidimensiones.

[1] Esta llave primaria proviene de un sistema de legado, por eso se escribe distinto a su predecesora.

Las minidimensiones, son tablas que contienen atributos de una tabla dimensión, evitando la necesidad de utilizar dimensiones de lento cambio para registrar la diversidad de sus combinaciones.

Al construirlas se toman en cuenta todas las combinaciones posibles. En el caso de edad e ingreso que cambian paulatinamente, son nuevamente agrupados en bandas, o intervalos específicos de la llave principal.

A su vez una minidimensión puede ser separada en otras minidimensiones. Los conjuntos de atributos de cada una no necesitan ser ajenos. Las minidimensiones también son susceptible de cambios pero asumiendo que se definen todas las combinaciones posibles, cuando cambie el perfil de la dimensión que describen simplemente se activarán aquellas llaves necesarias, conforme se extraen registros de la tabla proceso.

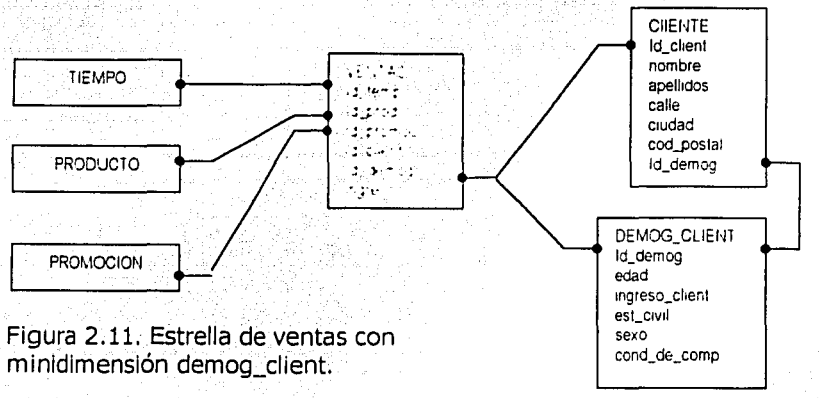



Figura 2.11. Estrella de ventas con minidimensión demog_client.

Nótese en la figura 2.11 que la llave primaria Id_demog para la minidimensión Demog_Client, es incluida como llave foránea en ventas y en la tabla cliente. Así se tienen los datos actuales del cliente y los que proporcionó al realizar la venta.

Se navega sobre el modelo dimensional y se obtienen los datos de la minidimensión directamente por medio de la llave foránea de la minidimensión. [Libro # 17].

**Normalización.**

La normalización es el proceso que transforma una **relación**[2] en relaciones más pequeñas y entendibles, que son equivalentes a la relación original. Tiene como objetivos:

- Eliminar redundancia en la información.
- Producir un diseño sobre las relaciones que representan la información, intuitivamente fácil de entender.
- Proteger la integridad de la información.

La normalización se utiliza en el diseño de base de datos relacionales, para que su mantenimiento sea más eficiente. Aunque la consulta de la información requiere la búsqueda de datos entre varias tablas relacionadas.

Normalizar implica el uso de un conjunto de preceptos denominados formas normales. Las tres primeras formas normales fueron definidas por Codd en 1972, para su trabajo titulado "*Further Normalization of the Data Base Relational Model*". Cada una de ellas implica una mejora sobre la anterior.

La tercera forma normal tuvo inconsistencias en su aplicación, entonces Codd junto con Boyce en el artículo "*Recent Investigations into Relational Data Base Systems*" de 1974, definieron una nueva forma normal, a partir de ella denominada Forma Normal de Boyce-Codd.

Más tarde Ronald Fagin definió la cuarta forma normal en su trabajo titulado "*Multi-Valued Dependencies and a New Formal Form for Relational Databases*" en 1977 y la quinta forma normal (también conocida como forma normal de proyección de junta) en el documento "*Normal Forms and Relational Database Operators*" de 1979.

Solo se manejan cinco formas normales y cada una se encuentra sostenida sobre la anterior, como en una pirámide de basamentos. A partir de la segunda forma normal, se toman las relaciones resultantes de aplicar la forma normal anterior, para reducirlas en otras más sencillas.

1FN, 2FN y 3FN.

A continuación se describen las tres primeras formas normales. Se basan en el análisis de dependencias funcionales. Con ellas se expresa la dependencia entre un atributo A con un atributo B. como se muestra a continuación.

num_cliente ⟶ nombre del cliente

La parte izquierda se denomina determinante y la derecha dependiente. El determinante no necesariamente es un atributo, puede ser una combinación de ellos. Durante este tema la palabra **dependencia** hace referencia a **dependencia funcional**.

[2] Para Codd. una relación es un subconjunto del producto cruzado de dominios.

**1ª Forma Normal (1FN).** Una relación se encuentra en 1FN si los datos de sus atributos son atómicos, es decir los valores almacenados no son listas o valores agrupados.

**2ª Forma Normal (2FN).** Una relación se encuentra en 2FN, si está en 1FN y no existe un atributo no primario[3], que dependa parcialmente de la llave primaria.

Esta forma normal, se aplica en casos donde la llave primaria esta compuesta por dos o más atributos y existen atributos no primarios que dependen solamente de alguno(s) de los miembros de la llave primaria. Se procede a separar en una tabla aquellos atributos que dependen de la llave primaria completa y en otra a los atributos que dependen parcialmente de ella.

**3ª Forma Normal (3FN).** Decimos que una relación está en 3FN, si está en 2FN y no existe algún atributo no primario que depende transitivamente de la llave primaria.

En este caso dentro de una relación se tiene un atributo no primario que depende de otro atributo no primario, éste a su vez depende de la llave primaria (dependencia transitiva). La solución consiste en separar en una tabla a todos los atributos que dependen únicamente de la llave primaria y en otra se coloca al atributo que depende del otro atributo no primario. [Libro # 10]

Para comprender estas formas normales, se normalizará la relación empleado:

| num_ emp | nom_ emp | Num_ dept | nom_dept | num_ jefe | nom_jefe | num_ proy | nom_proy | fecha_inicio | horas_ paga |
|---|---|---|---|---|---|---|---|---|---|
| 23 | López | 30 | Venta | 10 | Dávalos | 15 | Línea A | 17-Ago-02 | 100 |
| | | | | | | 35 | Línea B | 13-Ago-02 | 100 |
| | | | | | | 45 | Antares | 17-Feb-02 | 200 |
| 42 | Ayala | 20 | RH | 13 | Melchor | 25 | Osa | 13-Feb-02 | 250 |
| | | | | | | 45 | Antares | 17-Feb-02 | 200 |
| 25 | Luna | 50 | Sistemas | 1 | Cantero | 16 | Calipso | 18-Oct-02 | 150 |
| | | | | | | 15 | Línea A | 17-Ago-02 | 100 |
| | | | | | | 35 | Línea B | 13-Ago-02 | 100 |

Tabla 2.6. Relación Empleado sin normalización.

La relación mostrada en la tabla 2.6, señala los datos asociados a la asignación de un proyecto dado a un empleado, el departamento a el que pertenece y su jefe. Como se puede observar existen grupos de valores para los atributos: num_proy, nom_proy, fecha_inicio y horas paga. No se cumple con la primera forma normal.

Para normalizar la tabla 2.6, se generan dos nuevas relaciones, la primera llamada Empleado1 y la segunda denominada Asignación. En el caso de Asignación se usa la

[3] Cualquier atributo distinto de la llave primaria.

combinación de dos atributos como superclave[4], num_emp y num_proy. Veáse la figura 2.13.

Se han eliminado los grupos de valores, se tiene la primera forma normal en ambas relaciones. Entonces se analizan las dependencias entre los atributos de las relaciones, a fin de probar la siguiente forma normal.

| num_emp | nom_emp | num_dept | nom_dept | num_jefe | nom_jefe |
|---|---|---|---|---|---|
| 23 | López | 30 | Venta | 10 | Dávalos |
| 42 | Ayala | 20 | RH | 13 | Melchor |
| 25 | Luna | 50 | Sistemas | 1 | Cantero |

Tabla 2.7. Empleado1.

| num_emp | num_proy | nom_proy | fecha_inicio | hora_paga |
|---|---|---|---|---|
| 23 | 15 | Línea A | 17-Ago-02 | 100 |
| 23 | 35 | Línea B | 13-Ago-02 | 100 |
| 23 | 45 | Antares | 17-Feb-02 | 200 |
| 42 | 25 | Osa | 13-Feb-02 | 250 |
| 42 | 45 | Antares | 17-Feb-02 | 200 |
| 25 | 16 | Calipso | 18-Oct-02 | 150 |
| 25 | 15 | Línea A | 17-Ago-02 | 100 |
| 25 | 35 | Línea B | 13-Ago-02 | 100 |

Tabla 2.8. Asignación.

Figura 2.13. Aplicación de 1FN sobre la relación Empleado.

En la relación Empleado1 todos dependen de la llave primaria num_emp, lo que no sucede en Asignación donde nom_proy depende parcialmente de llave primaria compuesta por num_emp y num_proy. Nom_proy solamente depende de num_proy, para que la relación Asignación se encuentre en 2FN, se construyen dos nuevas relaciones Asignación1 y Proyecto.

Proyecto es formada por la dependencia de nom_proy con num_proy, mientras que Asignación1 conserva los atributos de asignación que dependen de la llave compuesta por num_emp y num_proy, como se muestra en la figura 2.14.



[4] Una superclave es un atributo primario o llave primaria.

| num_emp | num_proy | fecha_inicio | hora_paga |
|---|---|---|---|
| 23 | 15 | 17-Ago-02 | 100 |
| 23 | 35 | 13-Ago-02 | 100 |
| 23 | 45 | 17-Feb-02 | 200 |
| 42 | 25 | 13-Feb-02 | 250 |
| 42 | 45 | 17-Feb-02 | 200 |
| 25 | 16 | 18-Oct-02 | 150 |
| 25 | 15 | 17-Ago-02 | 100 |
| 25 | 35 | 13-Ago-02 | 100 |

Tabla 2.9. Asignación1.

| num_proy | nom_proy |
|---|---|
| 15 | Línea A |
| 16 | Calipso |
| 25 | Osa |
| 35 | Línea B |
| 45 | Antares |

Tabla 2.10. Proyecto.

Figura 2.14. Aplicación de 2FN sobre la relación Asignación.

La tercera forma normal separa de la relación a aquellos atributos que dependen transitivamente de atributos no primarios. Las relaciones Asignación1 y Proyecto no presentan dependencias transitivas. Empleado1 presenta la dependencia de nom_dept con num_dept, un atributo no primario que depende de la llave primaria num_emp. Se procede a descomponer a la relación Empleado1 en Empleado2 y Departamento, como lo muestra la figura 2.15.

| num_emp | nom_emp | num_jefe | nom_jefe |
|---|---|---|---|
| 23 | López | 10 | Dávalos |
| 42 | Ayala | 13 | Melchor |
| 25 | Luna | 1 | Cantero |

Tabla 2.11. Empleado2.

| num_dept | nom_dept |
|---|---|
| 30 | Venta |
| 20 | RH |
| 50 | Sistemas |

Tabla 2.12. Departamento.

Figura 2.15. Aplicación de 3FN sobre la relación Empleado.

La relación Empleado queda normalizada hasta 3FN con la descomposición sin perdida de las relaciones Empleado2, Asignación1, Proyecto y Departamento. [Libro # 11]

Forma Normal de Boyce-Codd.

La Forma Normal de Boyce-Codd (FNBC), se generó a partir de los problemas que presenta la tercera forma normal ante relaciones que posean las siguientes características.

1. Posee dos o más claves candidatas[5].
2. Dichas claves son compuestas.
3. Las claves candidatas tienen un atributo en común, se traslapan.

En relaciones que no cumplen con las tres condiciones anteriores, 3FN y FNBC son equivalentes. FNBC es conceptualmente más sencilla que 3FN, no hace referencia a 1FN y 2FN, ni a dependencias transitivas. Es recomendable utilizar 3FN y analizar posteriormente la conveniencia de utilizar FNBC.

**Forma Normal de Boyce Codd.** Una relación se encuentra en FNBC si y sólo si los únicos determinantes son claves candidatas.

Otra definición más formal indica que una relación se encontrará en FNBC, si toda dependencia funcional no trivial e irreducible a la izquierda, tiene una clave candidata como su determinante.

num_proveedor ⟶ proveedor

La dependencia funcional entre proveedor y número de proveedor, es no trivial e irreducible a la izquierda. Su irreductibilidad a la izquierda consiste en que su determinante "no es demasiado grande".

Es no trivial puesto que el atributo proveedor no forma parte del determinante. La siguiente dependencia funcional es trivial.

num_proveedor, num_producto ⟶ num_proveedor

La relación Proveedor de la tabla 2.13, se descompone en dos relaciones, Prov-prod-cantidad y Proveedor1, mostradas en las tablas 2.14 y 2.15.

| num_proveedor | Proveedor | num_producto | cantidad |
|---|---|---|---|
| 12 | TYASA | 8 | 500 |
| 13 | ALMSA | 6 | 400 |
| 14 | APSA | 2 | 300 |

Tabla 2.13. Relación Proveedor.

[5] Una clave candidata es un identificador único para cada registro de una tabla.

| num_proveedor | proveedor |
|---|---|
| 12 | TYASA |
| 13 | ALMSA |
| 14 | APSA |

Tabla 2.14. Relación Proveedor1

| num_proveedor | num_producto | cantidad |
|---|---|---|
| 12 | 8 | 500 |
| 13 | 6 | 400 |
| 14 | 2 | 300 |

Tabla 2.15. Relación Prov-prod-cantidad

Estas dos últimas relaciones obedecen a las siguientes dependencias funcionales:

num_proveedor ⟶ proveedor
num_proveedor, num_producto ⟶ cantidad

Ambas Prov-prod-cantidad y Proveedor1 se encuentran en FNBC. Los determinantes de ambas dependencias funcionales, son claves candidatas e irreducibles.

Cuarta forma normal.

Para las tres primeras formas normales se utilizan dependencias funcionales, para la cuarta forma normal se utilizan dependencias multivaluadas.

En una **dependencia multivaluada** el atributo "B multidepende de A", es decir, "A multidetermina a B". Un ejemplo es la dependencia de profesor con curso. A cada curso le corresponde un conjunto de profesores.

curso ⟶⟶ profesor

Una **dependencia multivaluada** es una **dependencia funcional** donde el dependiente es un conjunto de valores que coinciden con un valor específico del determinante.

**4ª Forma Normal (4FN).** Una relación está en 4FN si sus dependencias multivaluadas no triviales se traducen a una dependencia funcional para una superclave. De forma equivalente una relación se encuentra en 4FN si y solamente si está en FNBC y todas sus dependencias multivaluadas son dependencias funcionales sin clave.

La siguiente relación Texto en clase mostrada en la tabla 2.16, no se encuentra en 4FN. Contiene una dependencia multivaluada de profesor con texto, que no es una depedencia funcional, ya que texto está determinado por curso y no por profesor.

Sin embargo dos dependencias multivaluadas (dependencias funcionales sin clave) como Curso-Profesor y Curso-Texto, dan origen a las relaciones correspondientes que si están en 4FN.

| curso | Profesor | texto |
|---|---|---|
| Física | Castro | Mecánica Clásica |
| Física | Murillo | Optica |
| Física | Castro | Estática |
| Matemáticas | González | Análisis vectorial |
| Matemáticas | Gutiérrez | Variable Compleja |
| Matemáticas | Salgado | Cálculo |

Tabla 2.16. Texto en clase.

| curso | profesor |
|---|---|
| Física | Castro |
| Física | Murillo |
| Matemáticas | González |
| Matemáticas | Gutiérrez |
| Matemáticas | Salgado |

Tabla 2.17. Curso-Profesor.

| curso | texto |
|---|---|
| Física | Mecánica Clásica |
| Física | Optica |
| Física | Estática |
| Matemáticas | Análisis vectorial |
| Matemáticas | Variable Compleja |
| Matemáticas | Cálculo |

Tabla 2.18. Curso-Texto.

Quinta forma normal.

Una dependencia de junta es aquella donde todo valor posible de una relación, se obtiene de la junta de un conjunto de proyecciones sobre dicha relación. La tabla 2.19 muestra la relación VPY, que se descompone en tres proyecciones VP, PY y YV, que corresponden a las tablas 2.20, 2.21 y 2.22.

| V | P | Y |
|---|---|---|
| V1 | P1 | Y2 |
| V1 | P2 | Y1 |
| V2 | P1 | Y1 |
| V1 | P1 | Y1 |

Tabla 2.19. Relación VPY

| V | P |
|---|---|
| V1 | P1 |
| V1 | P2 |
| V2 | P1 |

Tabla 2.20. VP

| P | Y |
|---|---|
| P1 | Y2 |
| P2 | Y1 |
| P1 | Y1 |

Tabla 2.21. PY

| V | Y |
|---|---|
| V1 | Y2 |
| V1 | Y1 |
| V2 | Y1 |

Tabla 2.22. VY

**5a. Forma Normal (5FN).** Una relación se encuentra en 5FN si y sólo si cada dependencia de junta no trivial utiliza en su definición una clave candidata.

La dependencia de junta es no trivial cuando alguna de las proyecciones de la relación no es idéntica a esta última.

Las proyecciones VP, PY y VY, se encuentran en 5FN porque no involucran alguna dependencia de junta no trivial. La clave candidata de la relación VPY es la combinación de V, P y Y. Por ejemplo, VP utiliza a V como clave candidata un miembro de la clave candidata de la relación.

La 5FN es la última forma normal con respecto a las operaciones de proyección y junta. Sin embargo la aplicación de la 5FN en la práctica es poco observada.

Descomposición sin perdida.

Una relación después de ser normalizada puede regresar a su estado original, la normalización es un proceso reversible. La junta de un conjunto de relaciones normalizadas a partir de la segunda forma normal, ofrece la relación original con integridad en la información.

Esto se debe a la descomposición sin perdida, que cada forma normal realiza sobre la relación o conjunto de relaciones que se quieren normalizar. Se busca crear dependencia entre las relaciones para evitar problemas de actualización en la información contenida en ellas. La actualización implica comandos de **DML** (*Data Manipulation Language*) como INSERT, UPDATE y DELETE utilizados en sistemas de transacción.

Si se tiene una relación con los atributos: num_proveedor, proveedor, status, ciudad como se aprecia en la tabla 2.1, se pueden efectuar dos descomposiciones. La figura 2.16, representa una descomposición sin perdida, puesto que al realizar la junta de las tablas 2.2 y 2.3 se obtiene la tabla 2.1. En este caso se aplico 2FN.

| num_proveedor | proveedor | status | Ciudad |
|---|---|---|---|
| 12 | TYASA | 15 | Monterrey |
| 13 | ALMSA | 15 | León |
| 14 | APSA | 16 | Saltillo |

Tabla 2.1. Relación Proveedor Status.

| num_proveedor | proveedor | status |
|---|---|---|
| 12 | TYASA | 15 |
| 13 | ALMSA | 15 |
| 14 | APSA | 16 |

Tabla 2.2. Relación Proveedor#1.

| num_proveedor | ciudad |
|---|---|
| 12 | Monterrey |
| 13 | León |
| 14 | Saltillo |

Tabla 2.3. Relación Proveedor#2.

Figura 2.16. Descomposición sin perdida.

La figura 2.17, representa una descomposición con perdida, se puede saber el status de cada proveedor en la tabla 2.4, pero no se puede conocer su ciudad. No se aplico una forma normal específica. [Libro # 10] [Página Web # 3]

| num_proveedor | Proveedor | status |
|---|---|---|
| 12 | TYASA | 15 |
| 13 | ALMSA | 15 |
| 14 | APSA | 16 |

Tabla 2.4. Relación Proveedor#3.

| status | ciudad |
|---|---|
| 15 | Monterrey |
| 15 | León |
| 16 | Saltillo |

Tabla 2.5. Relación Proveedor#4.

Figura 2.17. Descomposición con perdida.

**Desnormalización.**

La normalización proporciona un conjunto de relaciones entendibles e integridad en la información. A partir de ella se diseña una base de datos relacional. La desnormalización no implica el desuso de tablas relacionadas, sino consultas que involucren menos juntas entre estas. Su aplicación sobre los modelos dimensionales, radica en la rica variedad de combinaciones que origina la redundancia de datos.

Una tabla normalizada no permite conocer los diferentes domicilios que un cliente puede tener en un lustro. La tabla desnormalizada guarda dos registros aparentemente iguales, excepto en uno o más de sus atributos. Las tablas en un modelo dimensional se encuentran tan desnormalizadas como se requiera, existirán casos donde las dimensiones tendrán cierta normalización, son esquemas copo de nieve y constelaciones.

Grados de normalización para un modelo dimensional.

A continuación se parte de un modelo dimensional para ventas que posee las siguientes jerarquías.

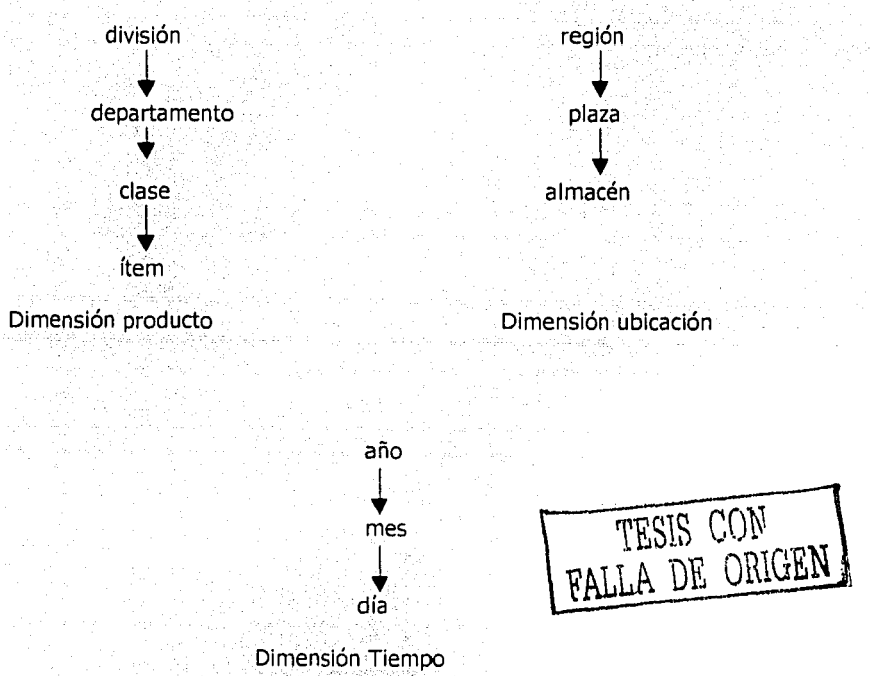





Figura 2.18. Jerarquías de un modelo de ventas.

Al modelo denominado A, se le aplican recursos de desnormalización que lo transforman en el modelo B y posteriormente en el modelo C.

Modelo A,

Corresponde a la figura 2.19, las características de sus tablas dimensión comprenden:

- Llaves primarias para los atributos, aparte de la llave primaria de la dimensión.
- Columnas descriptivas para la dimensión.
- Se indica el nivel de detalle de los datos contenidos.

La tabla proceso ventas contiene:

- Las llaves primarias de las dimensiones.
- No indica el nivel de las tablas dimensión.

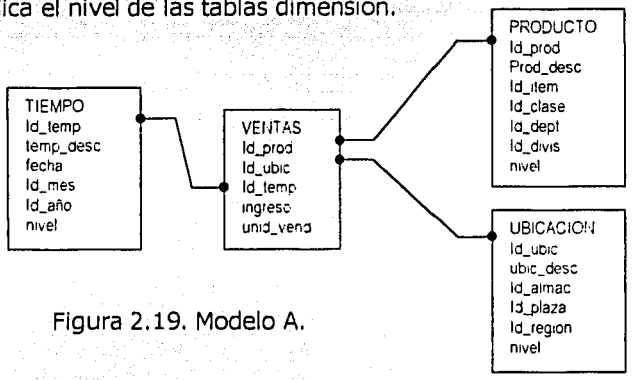



Figura 2.19. Modelo A.

Entre sus características, todos los atributos de las dimensiones se encuentran concentrados en una sola tabla física. Campos como Id_plaza en la dimensión ubicación, pueden tener valores nulos. El atributo nivel indica el nivel del atributo, como se muestra en la tabla 2.23 que contiene los posibles valores de la estructura correspondiente a la dimensión ubicación.

| Id_ubic | Ubic_desc | Id_almac | Id_plaza | Id_region | Nivel |
|---|---|---|---|---|---|
| 1012 | Centro | | 40 | 2 | 2 |
| 1013 | Toluca | 109 | 50 | 2 | 1 |
| 1014 | México | 110 | 20 | 2 | 1 |
| 1015 | Centro_occ | | | 2 | 3 |
| 1016 | Golfo | | | 1 | 3 |

Tabla 2.23.



Al aplicar *roll-up* sobre la tabla 2.23, se busca en el atributo nivel el valor máximo que es 3 asignado a la región y de este modo se obtienen los datos de ciudades como México, Toluca, Puebla, Morelia y Guadalajara.

Las relaciones muchos a muchos entre atributos no está permitida, lo que genera dificultades para desplegar la descripción de varios atributos pertenecientes a una misma dimensión. Una herramienta **OLAP** no soportaría este requerimiento o bien, procesaría la

información solicitada fuera de la base de datos. Para conocer almac_desc o region_desc se requerirán juntas adicionales sobre la tabla ubicación.

La razón de dichas juntas recae en la normalización que se hace sobre las dimensiones del modelo A. La secuencia de tablas de atributos presentes en la figura 2.20, muestra dicha normalización. Cada tabla de atributo contiene su llave primaria correspondiente, una columna descriptiva de esta y llaves foráneas para relacionarla con la tabla de un atributo de mayor nivel.

El modelo define llaves numéricas para cada atributo lo que permite al usuario, visualizar atributos de bajo nivel por medio de atributos de un nivel más alto. Se pueden ver todos los almacenes pertenecientes a una región dada. Presenta capacidad limitada para desplegar múltiples descripciones de la misma dimensión en un reporte. Recurriendo a diversas operaciones de junta sobre una dimensión.

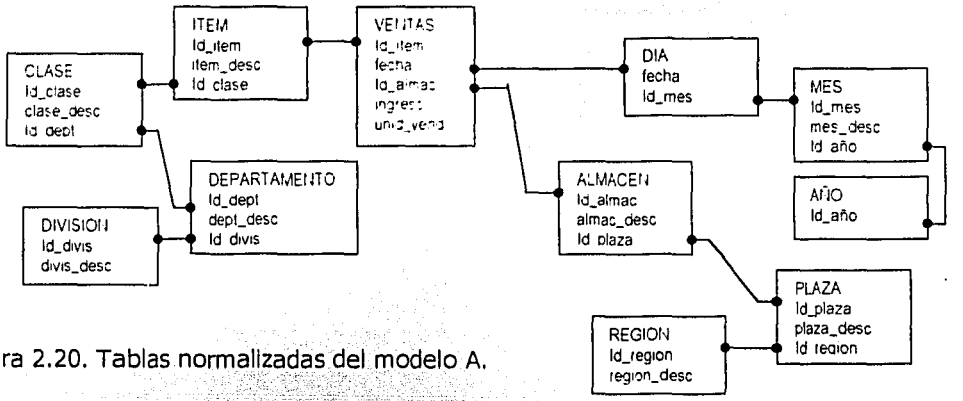



Figura 2.20. Tablas normalizadas del modelo A.

La tabla proceso guarda los datos diariamente. Su llave primaria es la combinación de las llaves foráneas que indican el nivel de los datos almacenados. Se trata de un esquema en la tercera forma normal, conocido como copo de nieve.

Tercera Forma Normal.

Para los modelos que utilizan esta forma normal, las tablas de los atributos contienen los identificadores de estos y las llaves de los atributos padres. Lo cual sucede en la tabla almacén que contiene su llave primaria Id_almac y Id_plaza que corresponde a la tabla de su atributo padre plaza. Requiere menos almacenamiento, pero necesita mayor número de juntas para obtener la relación entre el atributo padre presente y su respectivo hijo en un reporte.

Modelo B.

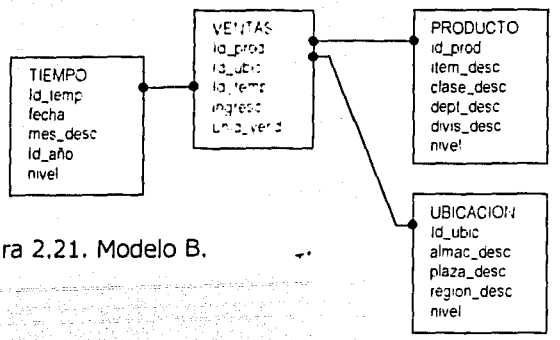



Figura 2.21. Modelo B.

Parte de la estructura del modelo A, donde las tablas dimensión poseen:

- Columnas descriptivas para todos sus atributos.
- Desaparece la columna que describe a la dimensión.
- Las llaves de los atributos desaparecen.
- Se indica el nivel de detalle de los datos contenidos.

La figura 2.22 muestra la normalización del modelo B. Dentro de las tablas de atributo existen llaves que identifican a atributos de nivel más alto, por ejemplo Id_año se encuentra dentro de la tabla día. Estas llaves aparecen en cursivas, señalando la diferencia entre la normalización del modelo A y B.

En el modelo se definen **superclaves** para todas las dimensiones. Se manipulan atributos de bajo nivel por medio de aquellos de nivel más alto. La consulta debe utilizar un campo de tipo caracter, lo que reduce eficacia, pues es más conveniente usar campos numéricos. Su capacidad para desplegar múltiples descripciones de la tabla dimensión en un solo reporte, es más amplia que en el modelo A.

Se desnormalizan las llaves de los atributos de alto nivel. Estas llaves llegan a ser llaves foráneas adicionales de las tablas de atributos de menor nivel. Reduciéndose el número de juntas cuando se manipulan atributos de alto nivel.

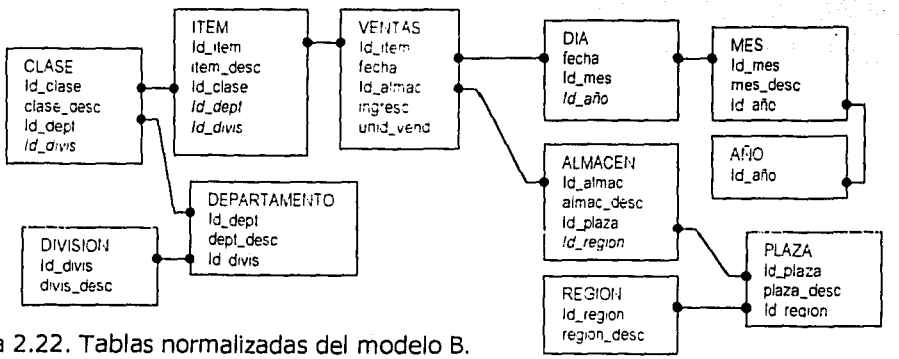



Figura 2.22. Tablas normalizadas del modelo B.

Ejempo, la tabla ítem contiene las llaves foráneas de las tablas departamento y división.

Se utilizan como recurso, identificadores desnormalizados.

Consiste en incluir al atributo padre, dentro de la tabla del atributo hijo. Conlleva almacenamiento redundante, reduciendo juntas para obtener la relación entre un atributo padre y su hijo. Requiere una junta adicional para generar reportes que contengan descripciones de atributos padre e hijo, tal es el caso de nombre de la región y nombre de sus almacenes.

La consulta asociada con las tablas de la figura 2.22 corresponde al siguiente código SQL:

```
Select v.unid_vend, a.almac_desc, r.region_desc
From ventas v, alamcen a, region r
Where v. Id_almac = a. Id_almac and
      a.Id_region = r.Id_region;
```

Las tablas almacen y región se juntan directamente sin utilizar a la tabla plaza como "intermediaria", lo que no sucede para la misma consulta con las tablas de la figura 2.20.

```
Select v.unid_vend, a.almac_desc, r.region_desc
From ventas v, alamcen a, region r, plaza p
Where v. Id_almac = a. Id_almac and
      a.Id_plaza = p.Id_plaza and
      p.Id_region = r.Id_region;
```

Las juntas entre tablas almacén-plaza y plaza-región son necesarias para obtener el atributo region_desc.

Modelo C.

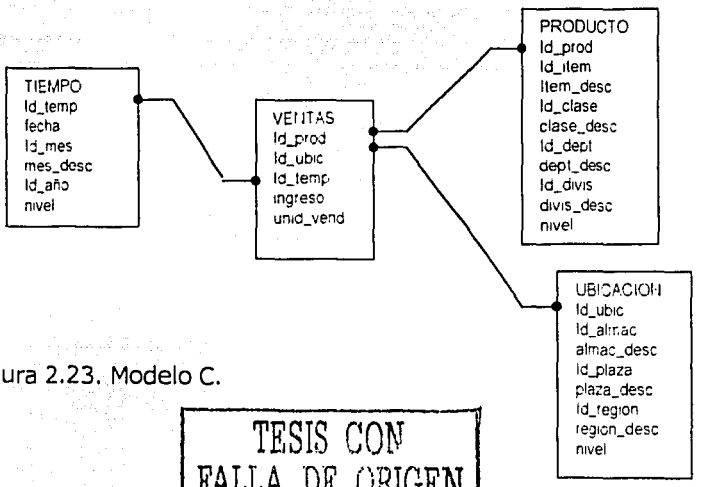



Figura 2.23. Modelo C.

En el modelo C, las tablas dimensión contienen:

- Columnas descriptivas para todos sus atributos.
- Carecen de una columna que describa a la llave primaria.
- Aparecen llaves de atributos.
- Se indica el nivel de detalle de los datos contenidos.
- Utilizan el nivel más alto de desnormalización.

La figura 2.24 muestra tablas dimensión para el modelo C, donde tanto las llaves como las descripciones de los atributos de nivel más alto se encuentra presentes, tal como Id_clase y clase_desc dentro de la tabla ítem.

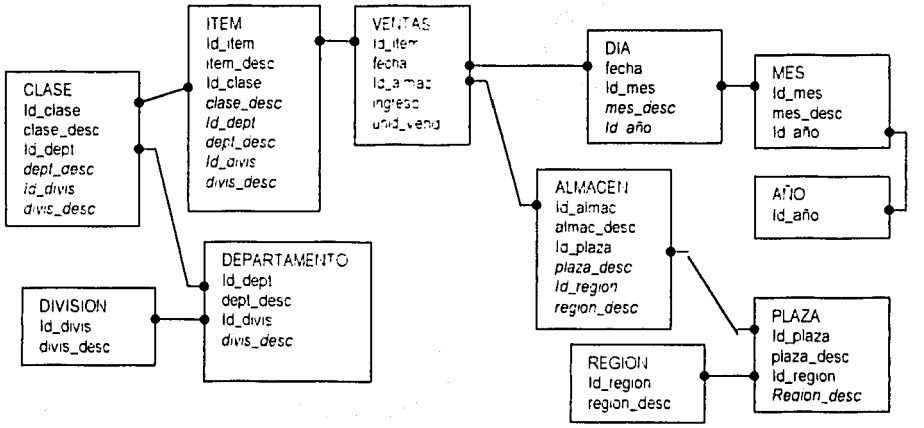



Figura 2.24. Tablas normalizadas del modelo C.

Combina características de los esquemas previos, permite la manipulación de los atributos independientemente de su nivel utilizando identificadores numéricos y múltiples descripciones de éstos campos.

Se utilizan como recurso de desnormalización, descripciones e identificadores desnormalizados.

Consiste en incluir las descripciones de los atributos de mayor nivel, requiere el mayor espacio de almacenamiento, eliminando la necesidad de consultar más de una tabla atributo para cada dimensión, desplegando múltiples atributos por reporte. Relaciona cualquier atributo con uno de nivel más alto.

La consulta ejemplificada anteriormente con las tablas de la figura 2.24, no tiene tablas "intermediarias" siendo más cómoda para una maquinaria **ROLAP**. El código **SQL** correspondiente es:

```
Select v.unid_vend, a.almac_desc, r.region_desc
From ventas v, alamcen a
Where v. Id_almac = a. Id_almac;
```

Solo hay una junta entre la tabla proceso y la dimensión almacen (ubicación para el modelo de la figura 2.23).

Normalización en los esquemas estrella.

Los esquemas estrella típicamente poseen una tabla física por atributo, donde cada uno es identificado por una llave única y tiene su propia columna descripción. Los atributos se encuentran mutuamente relacionados al incluir llaves foráneas en la tabla de atributo correspondiente, como Id_region en la tabla plaza. [Libro # 16]

Vistas materializadas.

Las vistas materializadas son utilizadas en **ROLAP**, como objetos que facilitan la consulta de datos del data warehouse, que no pueden ser actualizados, por el gran trabajo que implican los procesos de: extracción, transformación y carga de los datos.

Son recomendables en las consultas que requieren la junta de muchas tablas, donde el modelo dimensional esta parcialmente desnormalizado. También son utilizadas en consultas con funciones de grupo de **SQL** (AVG, SUM, STDDEV, etc). Estas vistas solamente trabajan sobre una tabla y sobre los valores actuales, después de la última carga de datos

Su utilidad es limitada y depende de las capacidades de los administradores de data warehouse. Tienen problemas, si se quiere hacer juntas con valores nulos incluidos (outerjoin) entre una tabla proceso y una tabla dimensión.

La tabla dimensión más interma requiere restricciones de unicidad sobre las columnas afectadas. Además, se requiere conocer los rowids de los índices de todas las tablas a juntar. Verificando la indexación de todas las tablas y en especial de la tabla proceso.

Las vistas materializadas pueden anidarse reduciendo el número de pasos a realizar en una sola consulta, sin embargo serán creadas en cada proceso de carga de los datos.

La partición de las vistas materializadas, agiliza aun más las consultas. Generalmente se aplica sobre vistas que tienen condiciones, en algún nivel de detalle de la dimensión tiempo. Es necesario determinar la frecuencia de las consultas, a fin de planear la partición, antes de que el usuario final las solicite. Incluso las vistas materializadas se obtienen de tablas particionadas, teniendo el debido cuidado de no modificar la estructura de estas, o bien destruirlas. [Software # 1]

**Consolidación de modelos dimensionales.**

Se conocen dos formatos de consolidación:

Atómicos: Las tablas proceso contienen datos básicos para cada nivel de detalle. Por ejemplo la tabla proceso de ventas que maneja todos los miembros de la jerarquía ubicación: almacen<plaza<región, figura 2.23.

Agregados: Las tablas proceso contienen un nivel único de datos por tabla.

Consolidar datos tiene como ventajas. Utilizar un número reducido de tablas, un esquema de comprensión accesible y consultas **SQL** de poca complejidad. La desventaja radica en la falta de soporte a relaciones muchos a muchos entre atributos. La tabla proceso no es escalable a pesar de su agregación. Las tablas de atributo no son escalables frente al número de atributos dentro de una dimensión. Sobre todo cuando se incrementa el número de elementos dentro de los atributos.

Para consultar la tabla proceso ventas de la figura 2.23, una tabla atómica con cierto nivel de detalle, se utiliza la siguiente sentencia **SQL**.

```
Select prod_desc, ubic_desc, temp_desc, ingreso, unid_vend
From   ventas v, producto p, ubicacion u, tiempo t
Where          v.Id_prod = p.Id_prod and
               v.Id_ubic = u.Id_ubic and
               v.Id_temp = t.Id_temp and
               p.nivel = 1 And
               u.nivel = 2 And
               t.nivel = 3;
```

Se obtiene un reporte para todos los artículos de la región Centro anualmente. [Libro # 9]

**Tabla núcleo y tabla de caracterización.**

Cuando diferentes tipos de productos o clientes son medidos (analizados) en forma distinta, se recomienda la creación una serie de tablas proceso específicas. La primera es conocida como tabla núcleo, la tabla proceso que contiene medidas comunes a todos los tipos. Aparte se construyen otras tablas procesos denominadas tablas de características, con más atributos que una tabla núcleo, pero solamente contienen los renglones asociados a cada tipo, siendo más pequeñas.

Generalmente van acompañadas por una o más dimensiones de características, para describir los aspectos propias de cada tipo. Si hay tipos heterogéneos (sin características comunes) puede requerirse la inclusión de dimensiones de características más no tablas de características. Cada tipo tiene su propia tabla dimensión de característica asociada a una tabla núcleo para su esquema estrella o copo de nieve.

Una dimensión núcleo es usada cuando se cruzan las categorías de un producto en una consulta y una dimensión de características se utiliza para consultas limitadas a una sola categoría. Las tablas núcleo deben ser duplicadas en las tablas de caracterización, añadiéndose solo aquellos atributos que son requeridos para el análisis del tipo de producto a ser tratado.

Por ejemplo, en una institución de crédito se manejan varios tipos de cuentas, al final de cada día se realiza un balance por cada tipo de cuenta. Cada tipo tiene características propias, una cuenta de ahorro requiere una tasa de interés y un certificado de deposito una fecha límite.

Todos los tipos de cuenta necesita dos atributos, tipo de cuenta y número de la misma. Datos incluidos en una tabla dimensión llamada cuenta asociada a una tabla núcleo llamada balance. Atributos específicos como: el interés a invertir en una cuenta o la fecha de termino para un certificado de depósito son incluidos en dimensiones de caracterización del tipo de cuenta que describen y estas tablas acompañan a sus respectivas tablas proceso de caracterización.

El segundo ejemplo, sucede en una compañía telefónica que ofrece servicios de comunicación a negocios y residencias. El departamento de mercadotecnia quiere conocer las necesidades de los clientes para ofrecerles nuevos productos y dar seguimiento a aquellos que cambien de domicilio en una misma área de servicio.

La información relativa a los servicios de tipo comercial y tipo residencial, se encuentra en parte repetida y en algunos aspectos es distinta. Una solución es construir dos data mart, uno para el área comercial y otro para el área residencial, de forma que estos sistemas pueden interactuar entre sí. Con el tiempo esto permitiría la creación de un Enterprise Data Warehouse.

Como los ejecutivos de las dos áreas tienen las mismas necesidades. Se decide la creación de tres modelos dimensionales, uno llamado núcleo con las coincidencias de los dos tipos de clientes y otros dos con su propia tabla de caracterización. En ambos esquemas se repite información referente a la tabla núcleo y se incluye aquella que es propia de cada área.

Al construir los esquemas de caracterización se busca un nivel de detalle y las dimensiones adecuados para maximizar el cruce de las dos áreas.

Data mart de servicio residencial.

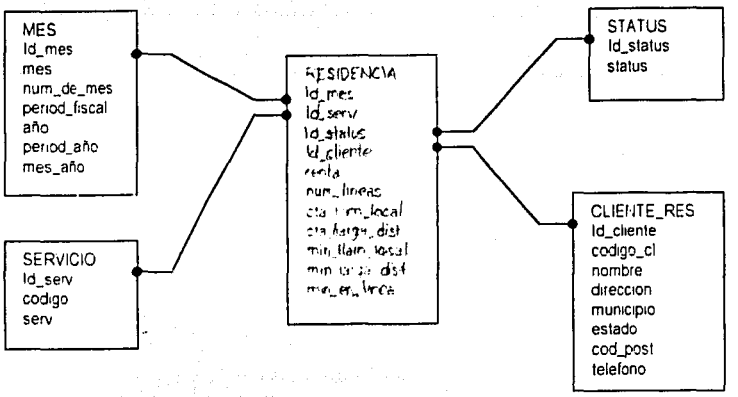



Figura 2.25. Esquema de caracterización para Servicio Residencial.

Un primer diseño, corresponde a la figura 2.25 que maneja la información en base al cliente y servicio, mensualmente. Sin embargo para obtener un reporte de los clientes que poseen correo de voz pero no llamada en espera, se requiere anidar sentencias **SQL** como se muestra en la consulta siguiente.

```
Select cliente_res.nombre
from cliente_res, residencia, servicio
where cliente_res.Id_cliente = residencia Id_cliente
        and servicio.Id_serv = residencia Id_serv
        and servicio.codigo = 'CORR_VOZ'
        and residencia.Id_cliente not in ( select distinct Id_cliente
                                           from residencia, servicio
                                           where residencia.Id_serv = servicio.Id_serv
                                                 and servicio.codigo = 'LLAM_ESP');
```



Conforme el análisis de la información se vuelve más complejo la estructura de las sentencias **SQL** es más sofisticada. En el diseño de la figura 2.25, cada servicio tiene un solo renglón en la tabla residencia y se deben grabar las mismas medidas para cada servicio. Existen medidas como número de líneas que no son aplicables a todos los servicios.

El segundo diseño, figura 2.26, toma en cuenta que el cliente será grabado una vez por mes. Para resolver el problema de los servicios se introducen atributos que funcionan como banderas para cada tipo. No existe una fuerte normalización en la dimensión servicio, sin embargo las banderas permiten observar las combinaciones de estos y el reporte planteado puede obtenerse de una sentencia en **SQL** más sencilla. La variable número de líneas no esta directamente asociada con los tipos de servicio que no lo requieren.

La sentencia en **SQL**, para obtener los nombres de los clientes, con servicio de correo de voz que no tengan contratado llamada en espera para Marzo de 2001 es:

```
Select cliente_res.nombre
From residencia, servicio, mes, cliente_res
Where residencia.Id_cliente = cliente_res.Id_cliente
        And residencia.Id_serv = servicio.Id_serv
        And residencia.Id_mes = mes.Id_mes
        And mes.mes = 'Marzo'
        And mes.año = 2001
        And servicio.band_correo_voz = 'S'
        And servicio.band_ll_espera = 'N'.
```

La dimensión status también sufre un cambio, indica el status para cada tipo de servicio dentro de los siguientes valores: sin cambio, alta y baja. De este modo se pueden sumar y agrupar las altas y bajas para un servicio determinado.

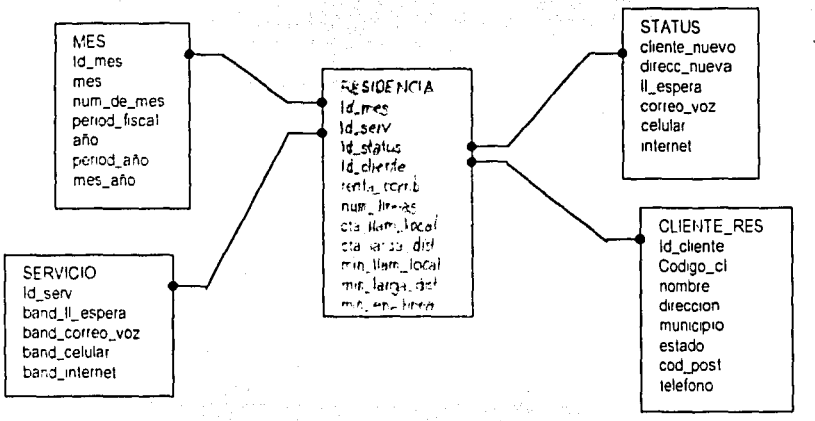



Figura 2.26. Esquema de caracterización para servicio residencial modificado.

Con la sentencia siguiente un atributo de la dimensión status se transforma en ceros y unos que son sumados fácilmente.

```
Select mes.mes, sum (decode(status.correo_voz, 'BAJA', 1, 0)) bajas,
                sum (decode(status.correo_voz, 'ALTA', 1, 0)) altas
from mes, status, residencia
where mes.Id_mes = residencia.Id-mes
        and status.Id_status = residencia.Id_status
group by mes.mes;
```



Esta solución implica más espacio en memoria, un problema que presenta es la necesidad de grabar el número de altas y bajas para cada servicio, entonces se guarda también el total de los servicios juntos. Al construir tablas agregadas se obtendrán valores mayores a uno haciendo más tardado el cálculo. Para evitar esta situación, a la tabla residencia se le asignan los atributos num_altas y num_bajas. [Libro # 3].

**Procesos combinados.**

En muchas ocasiones la respuesta a una pregunta por parte de los tomadores de decisiones, recae en varios esquemas estrella o copo de nieve. Los desarrolladores de data warehouse pueden combinar en un mismo modelo los procesos en cuestión o tenerlos en varios esquemas estrella que trabajarán de forma conjunta para proporcionar los resultados requeridos por los usuarios.

El primer caso implica trabajar sobre el nivel de detalle de los procesos a reunir en un mismo modelo, revisando los atributos de las tablas que integran los esquemas estrella de estos por separado. Lo segundo implica trabajar sobre herramientas Ad hoc, buscando reducir consultas anidadas o demasiadas juntas entre tablas.

Al proceso de obtener resultados de una o más tablas proceso en forma conjunta se le llama *drilling across*. Para realizarlo se requiere que las dimensiones utilizadas por las tablas proceso se encuentren conformadas.

Consiste en consultar cada tabla por separado y combinar los resultados. Se busca minimizar el tiempo de respuesta para obtener datos correctos. El usuario no tiene que construir las consultas sobre el data warehouse, estas se presentan a modo de reportes previamente diseñados.

Las consideraciones necesarias son:

- Cada sentencia **SQL** recupera los mismos atributos.
- Las cláusulas GROUP BY entre las consultas deben ser idénticas.
- Las restricciones descritas en las cláusulas WHERE también deben ser idénticas.
- Los datos obtenidos son resultado de combinar recíprocos *outer-join* sobre dimensiones comunes. Esto implica, dentro de una columna señalada en la operación de junta, colocar un valor nulo, si no se tiene un registro correspondiente en la tabla a la que no pertenece dicha columna.

Para analizar los aspectos que involucran el concepto de *drilling across*, se revisan dos diseños para desarrollo de data mart, uno para comercializar electrodomésticos y otro para administrar el presupuesto de una compañía.

Órdenes y embarques.

Supóngase que aparte de estudiar la cantidades ordenadas, del modelo de la figura 2.10, se pueden analizar a las cantidades embarcadas, en forma conjunta, como se muestra en la figura 2.27. Para este caso, el negocio de electrodomésticos considera la venta terminada al momento del embarque. El proceso embarque trabaja como un espejo del proceso orden, se requiere saber: el ingreso por embarque, el costo de los embarques y el porcentaje de ingresos, incógnitas resueltas con las dimensiones del esquema de embarques.

Aunque son procesos parecidos embarque maneja variables adicionales, datos sobre la compañía encargada de entregar los productos vendidos, la fecha de la orden original y la fecha del embarque.

Ordenes y embarques difieren en el nivel de detalle con el que manejan la dimensión tiempo, por las fechas en que la mercancía es ordenada y embarcada. Aunque siempre se aconseja tener un nivel de detalle específico para cada esquema, mientras los esquemas de embarque y ordenes no se afecten, pueden permanecer en un mismo esquema. Sino sucediera así los procesos tendrán que separarse.

Los cambios entre las figuras 2.10 y 2.27 son: dos llaves foráneas Id_fecha_ord y Id_fecha_emb para conocer las fechas utilizadas por el proceso embarque y la dimensión embarcador vinculada a la tabla proceso de embarque.



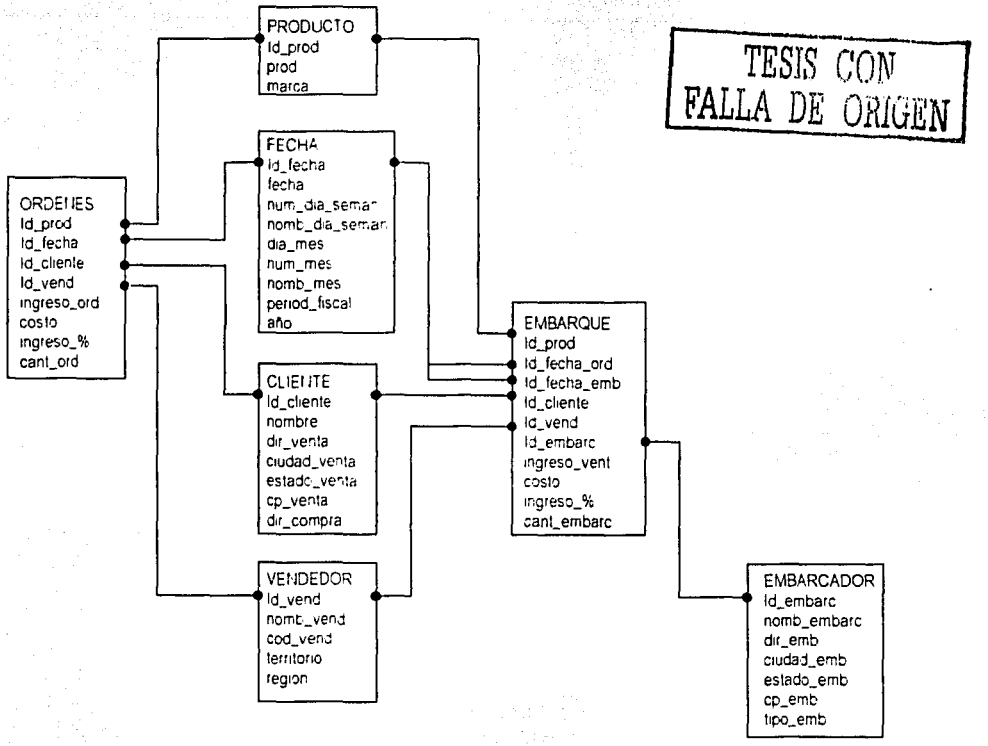



Figura 2.27.
Constelación para Órdenes y Embarques.

Si se quiere obtener el porcentaje de ordenes embarcadas en un mes determinado por producto, es necesario consultar las cantidad ordenada y la cantidad embarcada de su respectiva estrella, consultar la dimensión producto común a ambas estrellas, restringir la fecha de orden y utilizar la fecha de embarque correcta en la tabla embarque.

Un defecto presente en esta solución es que los artículos que todavía no han sido embarcados pero ya han sido vendidos, serían dados por perdidos en el cálculo del porcentaje. Otro es que muchas bases de datos no soportan múltiples ***outer join*** sobre una misma tabla, recurso necesario para las dimensiones fecha y producto en la estrella de embarque. Como consecuencia existe retraso en la ejecución.

Una táctica conveniente es tener los esquemas físicamente separados y consultar cada tabla proceso, combinando los datos obtenidos y cuidando las juntas entre las tablas. Se obtienen registros con información significativa. El proceso de carga del data mart siempre debe generar dimensiones conformadas para cada modelo.

Si se deseará conocer el porcentaje de productos que han sido embarcados en un mes a partir de una fecha determinada, en el esquema de embarques se consultan la fecha de orden y la fecha de embarque. La fecha de embarque debe pertenecer al intervalo comprendido entre la fecha de orden y la misma fecha más 30 días.

Existe independencia en el manejo de la dimensión tiempo entre ordenes y embarques. Las condiciones de consulta sobre embarques no afectan la consulta de ordenes. Se realiza el reporte aunque no cumpla con restricciones idénticas para ambos esquemas como se sugiere en *drilling across*.

Pagos, compromisos y presupuesto.

Un modelo dimensional difícil de diseñar, es aquel relacionado con las finanzas de una organización. El presupuesto, para el siguiente caso de estudio consiste en la asignación de un monto de dinero entre varias líneas de presupuesto para un propósito específico dentro de un tiempo determinado.

Los propósitos se clasifican de formas distintas lo que implica varios niveles de resumen. Los periodos de gasto también varían, por ello los contadores requieren conocer el estado de la cuenta asignada para los gastos de cada área.

El estado de cuenta depende de los compromisos adquiridos. Dinero que ha sido comprometido a un proveedor o socio al momento de la asignación del presupuesto. El pago es la transacción que involucra liquidar el compromiso. Es importante para los contadores conocer el remanente obtenido de restar al monto presupuestado el monto comprometido y los pagos realizados en cualquier momento. Las figuras 2.28, 2.29 y 2.30, exponen los esquemas estrella desarrollados para un data mart perteneciente al departamento de contabilidad de una organización.

Presupuesto.

Dentro del esquema de presupuesto cada renglón en la tabla proceso representa el monto dado a una línea de presupuesto, para una cuenta en particular a lo largo de un mes. Se contemplan presupuestos planeados anualmente y se incluye el atributo presto_anual como el monto presupuestado para todo el año, en la dimensión línea_de_presupuesto.

La variable contemplada en la tabla proceso es el monto del presupuesto, una variable aditiva, las consultas del esquema se dirigen a analizar el estado del monto del presupuesto con respecto a algún periodo de tiempo específico en la dimensión fecha.

En la dimensión línea_de_presupuesto se captura una categoría de dos niveles: categoría y subactegoría. Una categoría de múltiples niveles merece especial atención, grabando cada proceso en el nivel de detalle más bajo.

Si existe una categoría que carezca de subcategoria, se guarda "no aplica" en el atributo subcategoria. Sin embargo si en una categoría dos o más subcategorias tienen "no aplica", un reporte que proporcione los montos presupuestados, mostrará no aplica para una sola subcategoria. Para evitar esta anomalía, se duplican los datos de la categoria dentro de una dimensión subcategoria.

En la dimensión departamento se pueden realizar dos *roll-ups*. de departamento a división y de esta a responsable en la dimensión departamento. No es aconsejable incluir un organigrama, porque la estructura de una organización cambia paulatinamente.

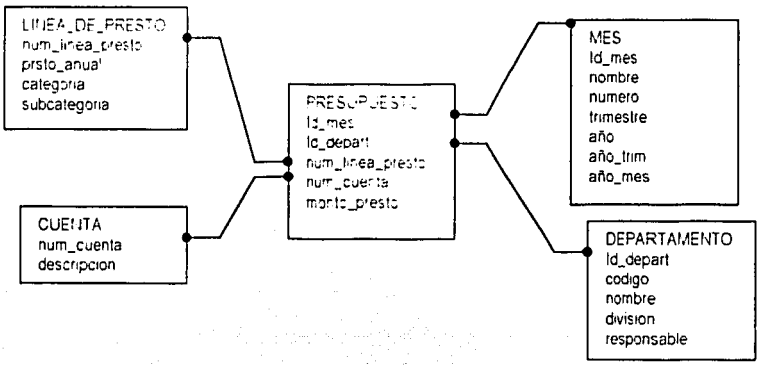



Figura 2.28. Esquema estrella para Presupuesto.



Compromisos.

Cada renglón de la tabla proceso compromisos representa el monto comprometido durante un periodo efectivo, no el total en compromisos del año a la fecha. Cualquier compromiso acoplado para más de una línea de presupuesto será un conjunto de múltiples renglones forzados en la consulta. Esto es similar a lo que sucede con la estrella de presupuestos a quien se le escapan líneas de presupuesto descritas por más de una cuenta.

En esta estrella se añade una nueva dimensión, compromiso que indica el tipo de compromiso. Si este sucede dentro o fuera de al organización y la partida para la cual el comité es hecho.

Cuando en una organización queda un remanente sin comprometer, se aconseja almacenar un renglón para no presupuestado de forma que la posterior asignación del remanente no afecte a las demás líneas de presupuesto.

Existen desembolsos no incluidos como compromisos, tal es el caso de los recibos por servicios públicos, para facilitar el calculo del presupuesto en estos casos se graba "pago fuera de compromiso". Sin embargo para conocer los compromisos a ser pagados es necesario que pagos se relacione en una consulta con presupuesto.

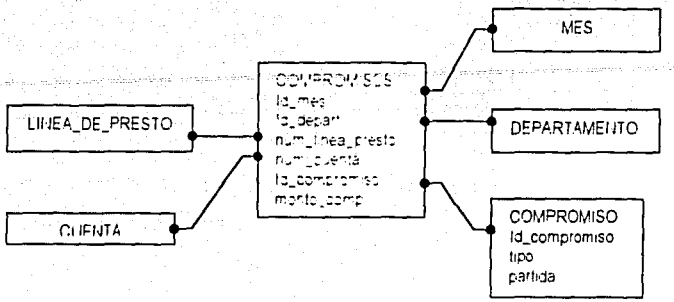



Figura 2.29. Esquema estrella para Compromisos.



Pagos.

El nivel de detalle para la estrella de pagos incluirá el pago ubicado dentro del contexto de cuentas, compromisos y líneas de presupuesto. Al igual que la estrella compromiso tiene un nivel de detalle más amplio que la estrella de presupuestos. La estrella de pagos incluye la dimensión pago que indica si este fue hecho por cheque, transferencia u otra forma.

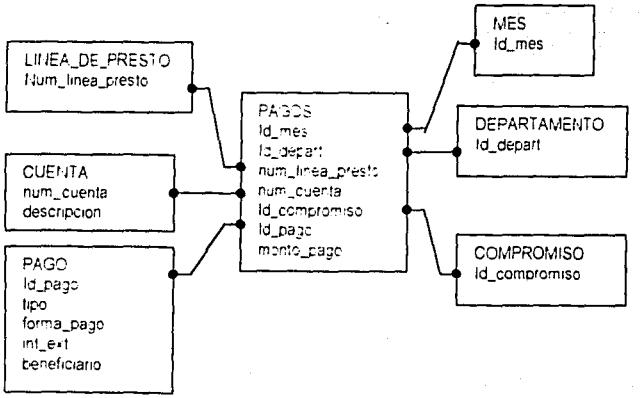



Figura 2.30. Esquema estrella para Pagos.

Los montos para pagos, compromisos y presupuestos representan los cambios para estos procesos a lo largo del mes. Dentro de un mes determinado una variable puede

no tener cambios al grabarse sobre una sola tabla, se graba un cero para la variable que no cambia. Los ceros que aparecen no afectan la eficacia de data mart cuando no se trabaja con esquemas separados.

Analizar la información de cada esquema y contrastarla manualmente es una tarea tediosa para los usuarios finales, por eso se diseñan reportes que auxilian en esta labor. Uno de ellos sería comparar los pagos con los compromisos para obtener el efectivo disponible en los meses siguientes y tener una idea de los compromisos no pagados todavía.

Se combinan dos consultas que obtienen el atributo num_cuenta de la dimensión cuenta y la condición acerca del presupuesto para el año en curso. Separadamente se obtienen el monto de compromisos y el monto de pagos.

Cualquier reporte parecido requiere un tiempo costoso de procesamiento si se realiza sobre datos normalizados hasta la tercera forma normal. El esquema estrella al incluir tablas desnormalizadas agiliza las consultas, facilitando la generación de reportes sobre esquemas combinados.

Los reportes propuestos se realizan sobre herramientas de consulta Ad hoc donde las consultas con **SQL** pueden ser modeladas. Aunque estas no dejan de ser complejas, muy pocas de las herramientas Ad hoc soportan múltiples consultas hechas a una base de datos.

Ceros innecesarios.

En muchas ocasiones los procesos comparten dimensiones. A veces es conveniente distinguir las variables de cada proceso y en otras se pueden omitir. Para el proceso de orden en la figura 2.10, ingreso_ord y cant_ord son variables registradas simultáneamente. Lo mismo sucede en el proceso de embarque para ingreso_vent y cant_embarc. Sin embargo, cant_embarc y cant_ord no son registradas al mismo tiempo. La segunda se registra cuando se efectúa un pedido y la primera cuando el electrodoméstico ha sido entregado.

Conviene tenerlas en tablas procesos separadas, puesto que la fecha de orden no coincidirá con la fecha de embarque y viceversa. Si en una misma tabla proceso se guardan la cantidad embarcada y la cantidad ordenada, se almacena un cero para cant_embarc mientras se registra cant_ord y otro cuando se graba cant_embarc.

Para el caso de pagos, compromisos y presupuesto la presencia de ceros innecesarios se da por la creación de un modelo a nivel de status , mostrado en la figura 2.31. Dicho modelo se diseña por la necesidad de los usuarios finales de obtener la suma de pagos o compromisos en un intervalo de tiempo.

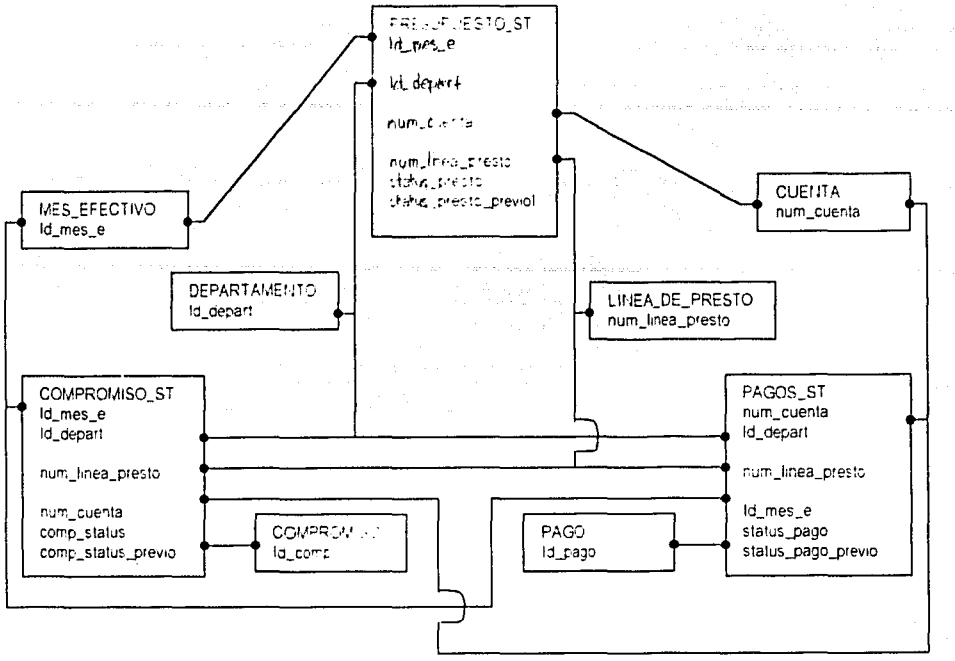



Figura 2.31. Esquema conjunto de Pagos, Compromisos y Presupuesto a nivel de status.

Obtener un reporte que genere por línea presupuestada, el total presupuestado, lo gastado hasta el mes de agosto, el gasto de septiembre y el complemento, requiere de múltiples consultas y múltiples condiciones aplicadas al esquema de compromisos y al presupuesto orientado a transacciones. La mayoría de las herramientas que Ad hoc no soportan esta tarea y por eso se busca un modelo que facilite la obtención de los datos solicitados.

La diferencia en los modelos consiste en la forma en que se guarda la información. Desde el inicio del año en curso hasta el mes actual, más no los cambios ocurridos en ese mes. En el modelo de la figura 2.31 se graban los procesos actuales y los del mes anterior.

Cuando no se presenta actividad esta no queda registrada. Cada renglón en la tabla proceso representa un nivel de status de un mes dado. Para conocer el estado del presupuesto en el departamento de recursos humanos en febrero de 1999. Se localiza el monto del presupuesto en la tabla proceso de presupuesto_st con condiciones sobre el departamento y el mes efectivo.

Las tablas procesos tienen poca densidad de datos, el monto del presupuesto se repite mensualmente. Es necesario restar al monto del mes actual el monto del mes anterior y capturar la diferencia, para conocer el incremento o decremento del presupuesto entre los meses previo y siguiente.

Las tablas de los esquemas estrella pueden estar esparcidas pero aseguran la presencia de meses con actividad, lo que no puede garantizar el modelo a nivel de status.

Semi aditividad.

El modelo de la figura 2.31, registra información resumida desde el inicio del año a la fecha. Las variables de cada mes efectivo son semi aditivas. Por ejemplo, el presupuesto para recursos humanos es de $70,000 y permanece constante a lo largo del año. Al final de cada mes se graba dicha cantidad. Cuando se practique una operación de *roll-up*, se efectuará una suma aritmética que arrojará un total de $840,000, lo que no corresponde a la realidad.

El acceso del data mart a los esquemas estrella evita esta problemática aunque debilita la eficacia la conjunción de las estrellas a nivel de status. Por los problemas de densidad y semi aditividad del modelo al nivel de status, es utilizado solamente para el mes actual. Seis tablas proceso coexisten en la base de datos.

Para satisfacer las necesidades de los usuarios finales se combinan las transacciones mensuales y la información del mes previo. Debido a que el modelo status contiene información para un mes, se evita la semi aditividad y su densidad no llega a ser mayor que el tamaño previsto para su almacenamiento.

Los reportes sobre actividades pasadas requieren las tablas de transacciones. El esparcimiento del modelo status no permite fusionarlo al modelo de transacciones. Los ceros del modelo status no afectan la eficacia del análisis, pero generan registros no deseados en los reportes enfocados sobre las transacciones.

Snapshot.

La necesidad de construir múltiples tablas proceso para monitorear un solo producto es generada por preguntas como: ¿Cuánto se gasto en determinada línea de producción? Para responderla se requieren las transacciones de un mes dado. Usando la tabla proceso que representa la actividad mensual de una serie de transacciones.

En cambio cuando se quiere conocer el presupuesto para artículos de oficina. Se requiere un balance neto de todas las transacciones encabezadas para un tiempo determinado. Mientras se calcula la información en un modelo de transacciones, un modelo dimensional, ofrece un panorama amplio de los presupuestos, con riesgos como la semiaditividad de los porcentajes.

Cuando se trabaja simultáneamente con un sistema de información gerencial y otro de transacciones sobre una misma base de datos se presenta un snapshot. El objetivo de trabajar con estos sistemas es tener una vista de las transacciones y una vista dimensional de los datos. La primera permite evaluar el desempeño de un departamento y la segunda conocer detalladamente las operaciones del mismo.

En realidad se trabaja sobre los mismos datos con enfoques diferentes. El sistema de información gerencial trabaja sobre un data warehouse que se alimenta de la base de datos del sistema de transacciones.

Snapshot es una técnica que combina la tecnología **OLTP** de bases de datos relacionales y data warehousing. En ocasiones se usará la misma tecnología, un sistema de transacciones con **RDBMS** y un sistema de información gerencial con **ROLAP**, o **RDBMS** y **MOLAP**. [Libro # 3]

**Agregación.**

Si en determinado momento para una tabla proceso que almacena la cantidad de producto vendido,es necesario llevar un control por hora, día o semana, se construyen esquemas estrella o copo de nieve, de acuerdo a uno de los niveles requeridos de la dimensión tiempo. Utilizando tablas agregadas o hardware de alto rendimiento y bases de datos afinadas.

Una tabla agregada (*aggregate table*) es mucho más pequeña y rápida que la tabla proceso original. Las principales ventajas que ofrece este recurso son: reducción de operaciones de entrada/salida, utilización del procesador y memoria RAM. También reduce cálculos extras que incrementan el tiempo de respuesta en consultas.

El número de discos físicos a ser leídos disminuye al reducir el número de registros solicitados por el usuario. Al estar ordenados en un nivel de detalle, no es necesario utilizar espacios de memoria para almacenarlos mientras un lote de datos es ordenado, dejando memoria RAM libre para una mejor ejecución.

Pre-Agregación.

Un detalle importante antes de realizar una agregación, consiste en definir cuándo y en qué nivel de detalle se crearán tablas agregadas. Dentro del modelo dimensional no todas las coincidencias o intersecciones entre dimensiones serán candidatas de pre-agregación. Su utilidad consiste en elegir tablas de uso frecuente, para no consumir espacio en memoria imponiendo cargas innecesarias sobre las copias de seguridad, la extracción y carga de la base de datos.

Agregación y relaciones entre atributos.

Cuando una tabla agregada es construida, los registros de un atributo hijo son resumidos en el registro del atributo padre, de acuerdo con una combinación de llaves que se encuentra contenida en la tabla de relaciones.

A partir de este momento la relación desaparece físicamente en las tablas de datos, cualquier modificación requerirá volver a construir las tablas originales con ayuda de la tabla de relaciones. En tablas grandes implicará tiempo y recursos durante el procesamiento de los lotes de datos.

El diseñador de data warehouse, debe contemplar si las relaciones entre atributos que se verán afectadas por una agregación son estáticas o dinámicas. Si la relación es dinámica se debe identificar la frecuencia de cambio, cuando esta es considerable, la agregación no es aconsejable puesto que su creación y mantenimiento implican arduos trabajos para extraer y preparar los datos. Las estáticas permiten operaciones de *roll-up* sobre la dimensión y resúmenes sobre datos agregados.

Radio de compresión.

Las agregaciones implican el uso de funciones como suma o promedio para un conjunto de registros, el promedio de registros que se combinan para generar un registro padre se conoce como radio de compresión y permite estimar la efectividad de una tabla agregada al reducir el número de registros a ser leídos. La agregación es viable si el radio de compresión es significativo.

| Item | Semana | Dia | Ventas |
|---|---|---|---|
| 16 | 1 | 5/2 | 12 |
| 16 | 1 | 6/2 | 10 |
| 16 | 2 | 11/2 | 13 |
| 16 | 3 | 20/2 | 11 |
| 16 | 3 | 21/2 | 17 |
| 16 | 4 | 27/2 | 19 |

Registros base.

| Item | Semana | Ventas |
|---|---|---|
| 16 | 1 | 22 |
| 16 | 2 | 13 |
| 16 | 3 | 28 |
| 16 | 4 | 19 |

Registros finales.

Radio de compresión = número de registros base/ número de registros finales.
= 6/4 = 3:2

Figura 2.32. Para la tabla ítem en un modelo de ventas.

Tomando en cuenta la figura 2.32 el radio de compresión calculado es 3:2 entre las ventas por semana y las ventas mensuales. Si el radio de compresión menor que 3:2 una tabla agregada requerirá más de dos terceras partes del espacio de la tabla base, obteniéndose una reducción de sólo la tercera parte en el número de registros solicitados.

Métodos.

Las metodologías básicas consiste en:

1) Agregar los datos, durante la rutina de extracción del data warehouse.
2) Aplicar *roll-up* a los datos después de ser cargados en el servidor **OLAP**.

El primer método trabaja con lotes de información, aunque implica complejidad en los programas de extracción, incrementando el número de archivos a ser creados para ser transferidos al servidor de l a base de datos, el servidor final no realiza el trabajo.

El segundo método crea tablas agregadas en la base de datos. Utiliza **SQL** y otro lenguaje procedural para resumir la base original después del proceso de carga en la base de datos. El proceso consulta la base de datos, realiza agregación y produce una nueva tabla. El servidor final no se libera de ésta carga de trabajo.

En cuanto a índices al momento de ser extraídos los datos, solamente se requiere un paquete ordenado sobre el índice compuesto maestro de la tabla proceso. Para otros paquetes ordenados basados o fuera de las restricciones dimensionales, será necesario manipularlos por separado en tablas de agregación específicas. [Libro # 9].

Ejemplo.

Para las ordenes agregadas mensualmente, las tablas involucran resúmenes de todas las mediciones en dos dimensiones, tiempo y vendedor, aunque de ésta última obtiene vital importancia el atributo, región. A las tablas agregadas que trabajan modificando dos dimensiones se les llama two-way aggregate. Las tablas agregadas pueden resumir mediciones en cualquier número de dimensiones.

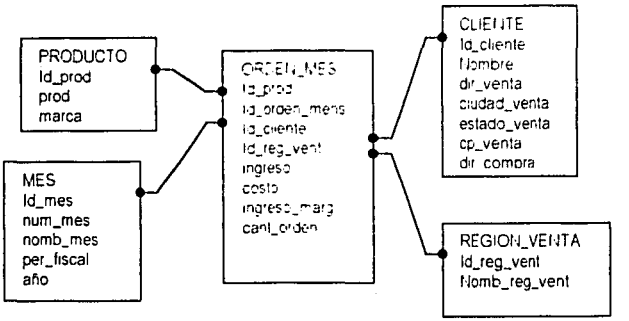



Figura 2.33.
Esquema de Ordenes agregadas por mes.

Observando la figura 2.33, el esquema de la figura 2.10, sufre las siguientes modificaciones. Primero se eliminan las dimensiones degeneradas num_orden y num_linea_orden añadiéndose una nueva dimensión degenerada Id_orden_mes. La llave foránea de fecha es reemplazada por aquella que se relaciona con la tabla mes y aquella que relacionaba a la tabla central con la tabla vendedor es sustituida por Id_reg_vent. La tabla vendedor desaparece y en su lugar se encuentra la tabla region_venta. Las tablas dimensión mes y region_venta reciben el nombre de *dimensional rollups*.

Las tablas agregadas permiten cambios sin afectar la aplicación. En su creación siempre se requerirá crear llaves artificiales para relacionarlas con la tabla proceso, en este caso dichas llaves son: Id_reg_venta y Id_mes. Además, para cada nivel de detalle, se requiere una tabla proceso y un conjunto de tablas dimensión apropiados.

La forma más efectiva para controlar la explosión de una agregación, es asegurarse que cada resumen sea de por lo menos 10 y preferiblemente de mas de 20 registros con un nivel de detalle menor. Sin embargo deben conservar un valor benéfico

para la tabla de agregación. Una restricción holgada, o ausente en una tabla proceso base es una restricción hermética en una tabla de agregación.

Si los datos agregados son representados en el modelo original por medio de un campo constructor, entonces para cada consulta el esquema se deberá restringir el campo constructor a un solo valor o doblarlo. [Libro # 3].

Uso posterior de datos agregados.

Algunos sistemas usan resúmenes para datos históricos. Quizás los datos estarán resguardados en línea por un año. Después de que son guardados, son menos accesibles, si se almacenan en un formato permanente. Las agregaciones son utilizadas para funciones de grupo de **SQL**.

Algunos data warehouses almacenan la información detallada y la información resumida. Esto requiere de más espacio cada vez, pero permite a los usuarios buscar detalles con consultas rápidas. Las agregaciones no pueden realizarse sobre procesos no aditivos, por ejemplo en el caso de una balance diario de una cuenta bancaria, al realizar *roll-up*, se obtendrá un balance para todas las cuentas y no por separado. [Libro # 9]

# Capítulo III.

# Aplicaciones de modelos dimensionales.

En el presente capítulo se presentan varios modelos dimensionales, sobre dos aplicaciones de data warehousing en mercadotecnia, satisfacción del cliente e inventario de distribución.

En el caso de inventario de distribución el modelo planteado para determinar la ocupación hotelera es un ejemplo cuyo desempeño se desconoce en la realidad. Tomando en cuenta la información de un análisis previo se diseño el modelo aunque en la práctica al momento de probar el sistema de información gerencial pueden encontrarse factores como retrasos en el proceso de carga y transformación o consultas con datos inconsistentes que modifiquen dicho modelo.

Lo mismo sucede en el caso de satisfacción del cliente donde el modelo para arrendamiento de autobuses fue diseñado a partir de otros ejemplos estudiados en el libro #3.

**Calidad y servicio.**

La calidad es un concepto que involucra varios procesos, para incrementar la productividad de una empresa, lo cual se ve reflejado en sus ganancias. En la tabla 3.1, se tiene la relación por sector productivo del PIB[1], de los años 1999 a 2001.

| | 1999 | % | 2000 | % | 2001 | % |
|---|---|---|---|---|---|---|
| Servicios financieros, seguros, actividades inmobiliarias y de alquiler | 217,704,364.00 | 0.15 | 228,952,194.00 | 0.15 | 238,324,956.00 | 0.16 |
| Transportes, almacenamiento y comunicaciones | 151,675,934.00 | 0.11 | 166,295,394.00 | 0.11 | 170,963,862.00 | 0.11 |
| Comercio, restaurantes y hoteles | 286,818,399.00 | 0.20 | 322,264,674.00 | 0.21 | 318,097,094.00 | 0.21 |
| Agropecuario, silvicultura y pesca | 80,627,331.00 | 0.06 | 81,128,943.00 | 0.05 | 82,686,903.00 | 0.05 |
| Minería | 18,431,124.00 | 0.01 | 19,133,818.00 | 0.01 | 19,026,571.00 | 0.01 |
| Industria manufacturera | 296,631,276.00 | 0.21 | 316,999,846.00 | 0.21 | 304,655,136.00 | 0.20 |
| Construcción | 60,328,557.00 | 0.04 | 63,381,852.00 | 0.04 | 60,525,042.00 | 0.04 |
| Electricidad, agua y gas | 23,717,887.00 | 0.02 | 23,950,033.00 | 0.02 | 24,365,758.00 | 0.02 |
| Servicios comunales, sociales y personales | 286,213,703.00 | 0.20 | 294,500,704.00 | 0.19 | 296,009,081.00 | 0.20 |
| | | | | | | |
| Total | 1,422,148,575.00 | 1.00 | 1,516,607,458.00 | 1.00 | 1,514,654,403.00 | 1.00 |

Tabla 3.1. PIB de 1999 a 2001.

[1] Producto interno bruto.
Fuente: Banxico.
Nota: Los datos de 2001. se refieren al Producto Interno Bruto trimestral promedio.

Los sectores de servicios son quienes concentran más del 70% del PIB, ver la figura 3.2. Esto muestra la importancia económica de los servicios.

| | 1999 | 2000 | 2001 |
|---|---|---|---|
| Porcentaje del BID para servicio. | 71 | 72 | 73 |

Tabla 3.2. Porcentajes del PIB para servicios.

Debido a esto las empresas involucradas tienen como objetivo sostener y elevar las ganancias anuales, requiriendo para ello de calidad en los servicios prestados.

En general los procesos asociados a la calidad son:

- Inventarios oportunos
- Círculos de calidad
- Control estadístico de procesos
- Operaciones a prueba de errores.

El segundo se aplica a los servicios, mientras que los dos últimos están enfocados mayormente hacia los productos.

Sin embargo los procesos, no describen a la calidad, sino que forman parte de un concepto denominado, administración para la calidad. Esta no es otra cosa que el conjunto de: principios, sistemas, procesos, métodos y técnicas encaminados a mejorar y/o sostener la competitividad de una empresa. Es la administración basada en procesos que vigilan la calidad.

Tipos de servicios.

Un servicio existe mientras el servidor ofrezca alternativas de satisfacción a una determinada necesidad del cliente, con mayores beneficios que los obtenidos por el propio cliente y por un precio razonable para este.

Existen varios tipos de servicios, por su ramo los hay de: comercio, comunicación, construcción, educación, salud, hospedaje, transporte y profesionales, etc.

Sin embargo los servicios se clasifican por su función, siendo esta clasificación sumamente valiosa en la toma de decisiones, pues permite construir estrategias de mercadotecnia. De acuerdo con el cuadro sinóptico de la figura 3.1.

Servicio

- Principal. Es la razón principal de la empresa que lo presta, su misión.
- Periférico. Es complementario o está relacionado con el principal.
- Valor agregado. Libres de costo que acompañan a los dos tipos anteriores incrementando su costo.

Figura 3.1. Tipos de servicios por función.

En un hotel por ejemplo, el servicio principal es el alquiler de una habitación y los servicios periféricos son: restaurantes, agencias de viaje, lavandería, salas de conferencias y discoteque. Los servicios de valor agregado son aquellos que invitan al cliente a quedarse (aunque no son libres de costo) como: alberca, despertador, lociones de baño, jabón, toallas, fax, teléfonos en los cuartos, etc.

Circulo de calidad.

También conocido como círculo de Deming, representa el ciclo de vida de un servicio. Un conjunto sistematizado de procesos llevados a cabo para la creación y prestación del mismo.

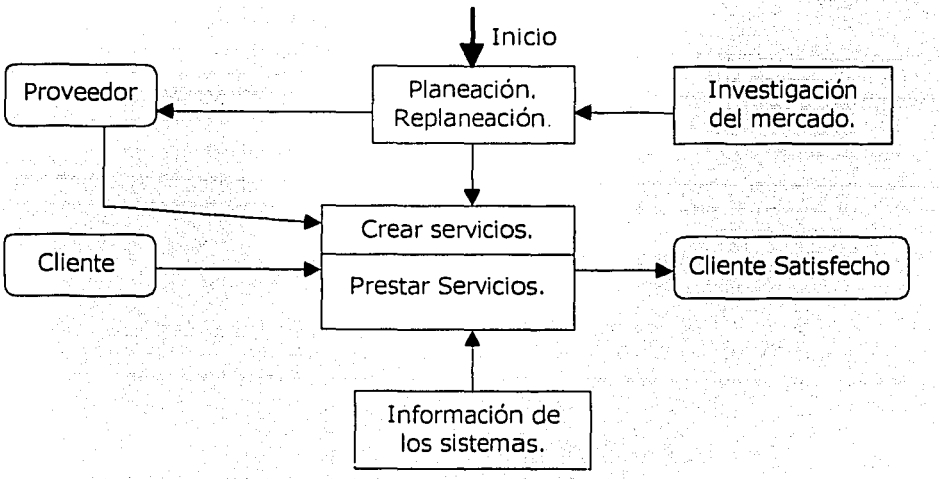



Figura 3.2. Círculo de Calidad o diagrama de Deming.

La figura 3.2, nos muestra el círculo de calidad, que ésta integrado por los siguientes pasos:

1. Determinar ¿Quiénes son los clientes?
   Implica conocer el perfil de las personas que requieren el servicio, por medio de estudios de mercado.

2. Detectar sus necesidades.
   Son los resultados que arrojan los estudios de mercado y análisis sobre la prestación del servicio.

3. Planear el servicio.
   Definir todos los servicios periféricos y de valor agregado que aportarán mayores ganancias en base al perfil y las necesidades del cliente.

4. Crear el servicio.
   Adquirir los recursos necesarios, así como implementar los planes y programas que definen al servicio.

5. Prestar el servicio.
   Darlo a conocer al público.

6. Evaluar el servicio.
   Recurrir (si es necesario) a expertos que transformen a información concisa los datos de los sistemas que monitorean la prestación de un servicio. Tomándose decisiones sobre la información obtenida.

Competitividad.

Consiste en la realización de controles estadísticos para vigilar el cumplimiento de los requerimientos del cliente. Se realiza en cada proceso, equipo, puesto e instalación de la empresa.

Los datos asociados a los sujetos y objetos de evaluación, provienen de los sistemas que registran las actividades efectuadas en cada departamento. Los sistemas pueden ser informáticos o bien, manuales. Estos sistemas deben proporcionar la información necesaria para concretar el quinto paso del ciclo de calidad.

El objetivo de la evaluación es reducir los costos y hacer más eficiente la prestación de los servicios al incluir aquellos de valor agregado que satisfagan al cliente. [Libro # 12]

**Modelos y Data warehousing.**

Como se expuso en el primer capítulo, el desarrollo de sistemas **EIS** y **DSS** y los sistemas de data warehouse fue auspiciado por la toma de decisiones de las empresas. Ante la competencia, se hizo necesario supervisar todos los aspectos que intervienen en la adquisición de un producto o servicio.

En data warehousing, no existe una metodología estándar. Generalmente se adaptan modelos dimensionales operantes, haciéndoles modificaciones acorde a las necesidades propias del cliente. **ROLAP** y **HOLAP** son las tendencias **OLAP** más utilizadas. Parten de software con características relacionales y existe documentación al respecto. Además, el equipo de data warehousing se encuentra más familiarizado con el software relacional.

Se han desarrollado varios modelos, de acuerdo con los problemas que resuelve el departamento para quien fue diseñado. Por ejemplo, los modelos que coordinan los resultados de procesos que vigilan la calidad de la producción.

Además de arte el diseño de un modelo dimensional requiere de experiencia e imaginación. Se busca la mejor solución a los requerimientos condicionada a la capacidad de análisis que los usuarios finales soliciten.

Dicha capacidad, se ve respaldada por la eficiencia del software **OLAP** aunque depende en gran medida de la naturaleza de las variables. Si esta es aditiva, no aditiva o semiaditiva facilitando la implementación de modelos dimensionales.

En muchos casos son los resúmenes o agregaciones quienes simplifican el trabajo del usuario para conseguir información rutinaria de primera mano, o bien, un material valioso sobre el que se realizan análisis estadísticos y otras técnicas.

Satisfacción del cliente.

La satisfacción del cliente es un concepto complejo y relacionado con varios factores. Uno de ellos es la calidad del servicio o producto que se ofrece en el mercado. Otro es la póliza de garantía para cubrir desperfectos del bien a comercializar y en el caso de incorfomidad por parte del comprador, el reemplazo del bien o el reembolso del importe.

El ejemplo de la compra de un automóvil explica el párrafo anterior. Al adquirir un automóvil directamente de agencia, existen dos caminos para efectuar el pago: al contado ó con financiamiento. En cualquier caso, la adquisición se ve respaldada por la garantía de servicio y un seguro en caso de robo. La garantía de servicio incluye: reparación ante desperfectos, revisión y ajuste después de un kilometraje especificado o modificaciones sobre el diseño original para transformar un modelo austero en uno personalizado.

El control de calidad no culmina con la compra, como sucede con productos que limitan la cobertura a la duración de la póliza de garantía. El cliente puede adquirir refacciones o realizar servicio a su unidad con descuentos por un tiempo limitado y en

base a promociones. Si el vehículo es adquirido por medio de un intermediario, la cobertura cambia y se incluyen servicios básicos.

La satisfacción del cliente no solo depende de las políticas de la empresa, sino también del mercado. Cuando una compañía ofrece garantías sobre sus productos después de la compra obtiene para sí, la preferencia del consumidor incrementando sus ganancias. Esto conlleva a que varias compañías consideren proporcionar beneficios más extensos a los consumidores, en la compra de sus productos o servicios.

Esta serie de procesos encaminados a generar bienestar en el poseedor del producto o servicio, no solo involucra al proveedor, sino también a los distribuidores. En el caso de los comerciales de televisión tiendas distribuidoras, ofrecen al adquirir un producto: descuentos y boletos para rifas y algunas veces membresias que proporcionan descuentos adicionales bajo políticas de la empresa.

La satisfacción al cliente es personal y no cuantificable, puesto que puede ser muy sencilla o muy exigente. Lo más recomendable es incluir un diagrama que muestre la distribución de esta desde la insatisfacción extrema hasta la completa satisfacción. Las necesidades de los clientes se conocen por medio de un sistema que archive las quejas y sugerencias de los mismos.

El ejemplo propuesto parte del deseo de incrementar las utilidades. Los analistas consideran como solución a mediano plazo la construcción de un data warehouse para monitorear la información básica para resolver sus requerimientos. Se parte de un modelo base que va siendo refinado. Cada refinamiento ejemplifica una perspectiva distinta de los requerimientos.

Arrendamiento de autobuses.

Se trata de una empresa dedicada a la renta de autobuses. La renta es efectuada por: escuelas, asociaciones civiles o personas físicas para transportarse a: convenciones, eventos especiales, visitas guiadas, excursiones y sepelios.

La empresa cuenta con tres clases de autobuses: estándar, primera clase y lujo. Los autobuses estándar ofrecen 40 asientos para pasajeros, carece de aire acondicionado y sanitario por lo que generalmente es utilizado en tramos cortos.

Los autobuses de primera clase cuentan con: aire acondicionado, sanitario y televisores para la presentación de videos. Son recomendados para sitios relativamente alejados y excursiones que parten y regresan el mismo día.

Los autobuses de lujo cuentan con 20 asientos ajustables a 3 posiciones los servicios de primera clase y cocineta. Presentan menor arrendamiento. El costo por el servicio es calculado en base al número de asientos ocupados (número de pasajeros) y al costo por motivo.

El Gerente General quiere incrementar la renta de autobuses de lujo, sugiriéndolos para eventos especiales y convenciones. Sin embargo no tiene información sobre las preferencias de sus clientes.

La construcción de un data warehouse es un proceso costoso, que pocas empresas pueden solventar. El ejemplo supone que la compañía y un equipo de profesionistas en tecnologías de información, han considerado desarrollar un data mart para el departamento de mercadotecnia. Evaluando periódicamente la preferencia de sus clientes y de ser necesario ajustar los costos por motivo del viaje y por asiento para cada tipo de autobús.

Hacia la construcción del modelo dimensional.

La tarea a realizar es construir un esquema que represente el proceso de arrendamiento de los autobuses para los clientes registrados en la cartera de la compañía. Realizando los siguientes pasos:

- Primero. Definir las dimensiones del esquema considerando la información relacionado con el proceso a tratar.

Se necesitan: la fecha de solicitud, el nombre o razón social del cliente, el motivo de arrendamiento y el tipo de autobús utilizado.

Tomando como base los esquemas del capítulo 2, se tienen dos dimensiones inmediatas, la dimensión cliente y la dimensión tiempo. El siguiente paso es reflexionar acerca del tipo de autobús y el motivo de arrendamiento.

Sabemos que el costo por arrendamiento es calculado por medio de la siguiente fórmula:

$$P = n * a + m$$

Donde:

P: Precio
n: número de pasajeros.
a: costo por asientos.
m: costo por motivo.

El costo por asiento varia en función de los costos por mantenimiento para cada tipo de autobús. El costo por motivo varía en función de la demanda.

El precio es importante puesto que incide en la preferencia de los clientes, entonces el motivo y el tipo autobús son dimensiones del esquema a proponer.

- Segundo. Definir los atributos de la tabla proceso y las tablas dimensión.

Para la tabla proceso de arrendamiento sus atributos son: número de pasajeros, origen y destino. En la dimensión cliente interesa registrar: su nombre o razón social y su

teléfono. Para tipo de motivo, su costo e igualmente el costo de asiento para tipo de autobús.

- Tercero. Definir las jerarquías.

La dimensión tiempo usa una jerarquía con los niveles de detalle: día, mes, trimestre, año, mes-año y trimestre-año. Otras combinaciones tienen baja densidad de datos o no son realmente utilizadas por los usuarios finales.

Para agilizar la búsqueda de información, se añaden: nombre del día y nombre del mes. El esquema propuesto se muestra en la figura 3.3.

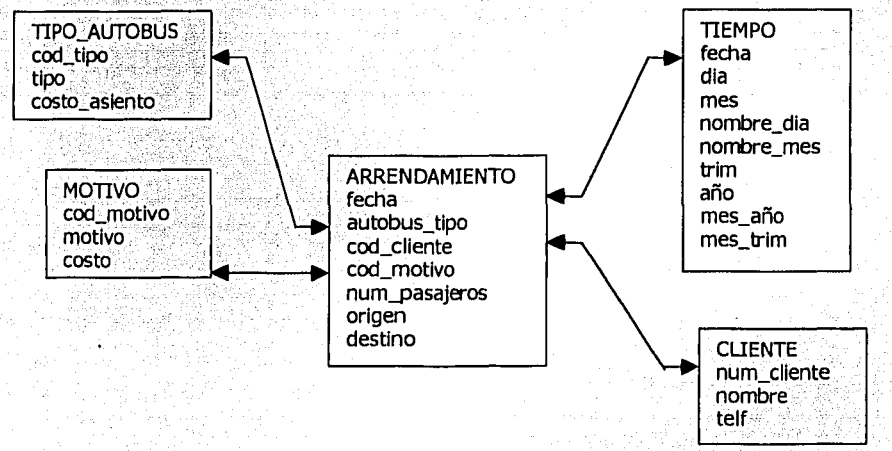



Figura 3.3. Esquema para arrendamiento de autobuses.

Tanto tipo de autobús como motivo son dimensiones de lento cambio tipo II. Los varían con periodicidad distinta en ambas dimensiones. Los analistas pueden utilizar llaves enteras en cada tabla. La preferencia de los clientes depende de la calidad del servicio, de la satisfacción que obtengan de este. Si los autobuses son impuntuales los clientes dudarán antes de volver a contratar el servicio.

Otros factores son: el confort de los asientos, uso apropiado del aire acondicionado, buen funcionamiento del sanitario y la cocineta, botanas y refrescos, por citar algunos.

Puntualidad.

A continuación se ofrece una solución para vigilar la puntualidad del servicio. La fecha del esquema anterior es la fecha del servicio haciendo falta registrar: la fecha y hora de partida, la fecha y hora de llegada y la fecha y hora estimada de llegada.

Los elementos faltantes se vuelven atributos de la tabla proceso, como se muestra en la figura 3.4. Si la hora estimada de llegada, no coincide con la hora real de llegada, se presentó un contratiempo.

El contratiempo puede ser descrito en un campo denominado observaciones, sin embargo, los analistas deben considerar la frecuencia del mismo. Si esta es pequeña, la inclusión de observaciones en la tabla proceso produce datos esparcidos, algo poco conveniente.

Si la frecuencia es considerable, la opción mas viable es crear un modelo que mensual o trimestralmente registre las observaciones.

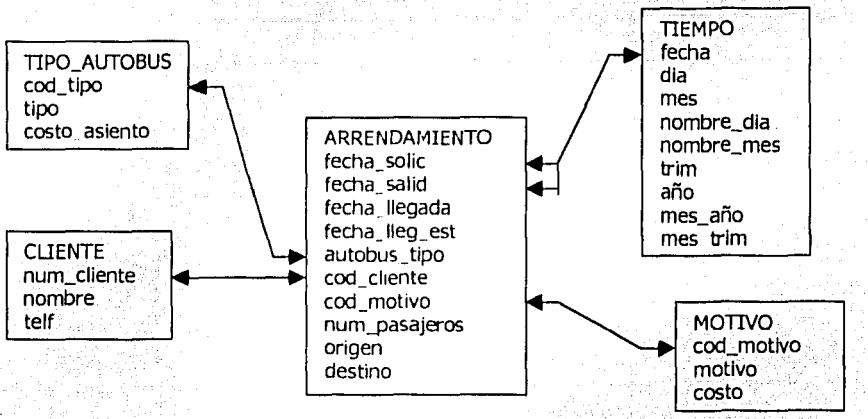



Figura 3.4. Esquema modificado para registrar puntualidad.

Reembolso y cambio.

En la satisfacción al cliente se debe tomar en cuenta la devolución de la mercancía o bien, parte de la inversión hecha por él, al momento de la compra.

Si se vende un producto se realizará cambio y/o reembolso. Cuando las empresas manejan servicios, el reembolso es la única opción para remediar la inconformidad del cliente. Sin embargo no todas las causas ameritan una devolución del 100%. Cada organización determina el porcentaje de reembolso de acuerdo con el motivo que lo solicita. Para obtener información valiosa al respecto se manejan dos procesos diariamente, la transacción base y el reembolso o cambio.

Si los sistemas de transacción, contienen en sus bases de datos información referente a los motivos de inconformidad del cliente. Se puede construir un catalogo a partir de la información proporcionada y añadirlo como una dimensión asociada a la tabla proceso reembolso.

En la figura 3.5, se describe un modelo dimensional para reeembolsos y arrendamientos. Como se trata de una cifra, si se desea obtener el porcentaje, se realiza el siguiente cálculo:

Porcentaje de reembolso = reembolso * 100 / (costo_asiento*num_pasajeros + costo)

El valor del reembolso es almacenado en la dimensión razón (motivo del reembolso). Esta operación implica varias consultas al modelo dimensional (una constelación como figura 2.10) que no es conveniente calcular en reportes o guardarlo en tablas, puesto que no es una variable aditiva con respecto al tiempo.

Los analistas para encontrar este porcentaje obtienen el precio total y el precio de reembolso. Posteriormente, los datos son vaciados en una hoja de cálculo, donde la fórmula citada con anterioridad es llevada a cabo obteniéndose el porcentaje requerido.

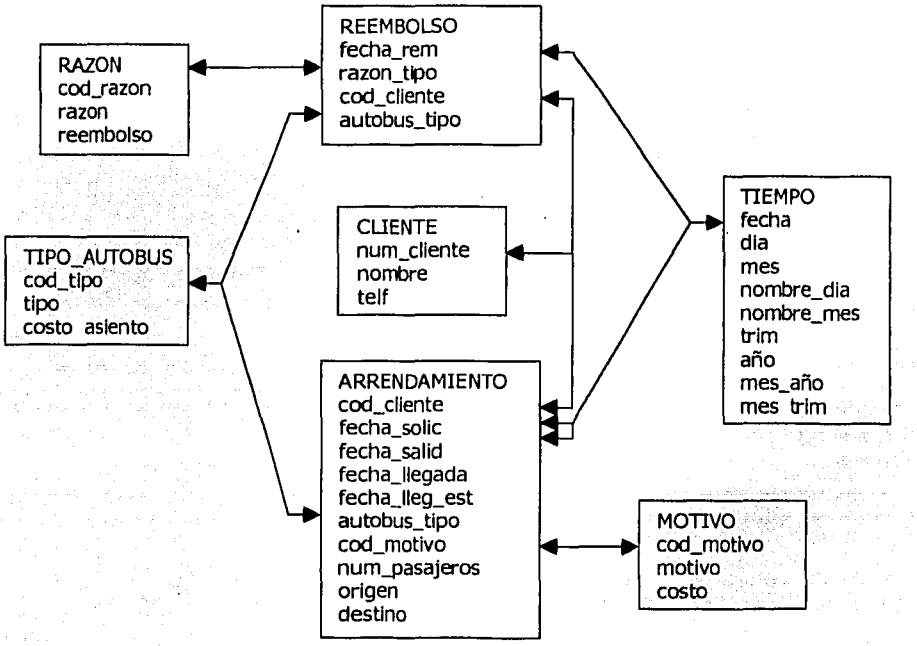



Figura 3.5. Modelo para administrar reembolsos y arrendamientos.

El sistema comprueba la causa de imputualidad, para un reembolso justificado. Para otras causas, es necesario corroborarlas en el sistema transaccional pertinente.

Si se trabaja con productos no perecederos existen dos tablas proceso, una correspondiente a reembolso y otra a cambio. Si los motivos de estos procesos son totalmente ajenos cada una maneja su propia dimensión razón o motivo. Si hay motivos

comunes son incluidos, junto con los propios de cada proceso en una misma tabla tanto a la tabla reembolso, como a la tabla cambio.

Al construir un modelo para satisfacción del cliente, es conveniente.

- Entender el valor del producto. Contemplar un reporte que proporcione: calidad, características y costo.
- Identificar los elementos que forman el valor del producto. [Basado en Libro # 2].

Inventario de distribución.

Un inventario da a conocer aspectos clave, no solo la existencia de determinado artículo, sino también el lugar donde se encuentra y el uso que recibe.

Permite comprar nuevas unidades de este artículo, reemplazarlo ó dejar de utilizarlo. La información proporcionada por los inventarios, influye de manera directa o sutil, sobre las acciones de la empresa. los inventarios son indispensables, para los objetos contenidos en las instalaciones, o para los productos que comercializa la empresa.

La optimización del uso de un artículo inventariado queda a cargo de expertos en materia. Se diseña un modelo con parámetros a evaluar, de forma consistente y confiable. Algunos rubros, requieren, un inventario de distribución, estos inventarios tienen por objetivo, indicar el lugar y el estado de uso de un conjunto de bienes muebles o inmuebles limitados, los cuales generan ganancias directas a la organización.

A continuación se describe un modelo dimensional teórico, diseñado para el departamento de mercadotecnia de una cadena de hoteles. [Libro # 2].

Ocupación hotelera.

El departamento de mercadotecnia de una cadena de hoteles, ha considerado la implementación de un data mart. Los analistas comisionados para el análisis han obtenido el siguiente resumen:

La cadena es dueña de 40 hoteles en el país. Maneja dos líneas de hoteles, la primera conformada por 10 hoteles maneja suites. Sus clientes son en su mayoría turistas que realizan escalas y agentes de negocios. La segunda es solicitada por todo tipo de clientes y procura ofrecer tarifas competitivas en el mercado.

Además de las habitaciones, los hoteles cuentan con restaurantes. En ciudades coloniales o cercanas a zonas arqueológicas, cuentan con tiendas de artesanías y recuerdos.

La utilidad recabada es mayor para la segunda línea. Los costos por mantenimiento y por pago de nómina son altos. La preocupación de la cadena es incrementar la ocupación de sus hoteles.

De la entrevista con el Director de Mercadotecnia de la cadena concluyeron:

1. El número de habitaciones por hotel, varia entre 40 y 70. Existen 6 tipos de habitación básicamente son estándar y suit con tamaños: pequeño, mediano y grande.

2. Si los precios son demasiado bajos, la capacidad es limitada y algunos clientes buscan otras opciones, dejando pocas utilidades. Si están sobrevaluados los hoteles estarán vacíos.

3. Los ejecutivos de mercadotecnia analizan la tasa de ocupación. Dependiendo de la época y del clima, la tasa aumenta o disminuye. Para establecer el precio, comparan el costo marginal con la tasa de ocupación."

Propuesta.

Los analistas definieron los siguientes objetivos:

- Diariamente la ocupación será analizada con respecto a los tipos de habitación.
- El análisis arroja niveles de utilización promedio para hoteles específicos.
- Para cada caso (hotel y tipo de habitación), se captura el valor de la renta.

La tasa de ocupación de un hotel, es el porcentaje de cuartos rentados al momento y se obtiene de la siguiente fórmula:

$$T = o / o + v + n$$

Donde:

T: tasa o porcentaje de ocupación.
o: número de habitaciones ocupadas.
v: número de habitaciones vacantes.
n: número de habitaciones no disponibles.

Las medidas relacionadas a la ocupación son: Número de habitaciones ocupadas, número de habitaciones vacantes, número de habitaciones no disponibles (por reparación o mantenimiento).

De acuerdo con la figura 3.6, las tablas dimensión propuestas son: tipo_hab, tiempo y hotel. La línea hotelera a quien pertenece el inmueble, es un atributo susceptible de normalización. No obstante, el espacio ocupado en memoria y la complejidad de las consultas sugiere un tratamiento de atributo en la tabla hotel. Un programa disparador (*trigger*) se encarga de vigilar su integridad y consistencia durante la carga del data warehouse.

Como realmente los hoteles no se encuentran agrupados por región, no es necesario crear una dimensión para manipular una jerarquía geográfica. En cada consulta, se obtiene el porcentaje de ocupación para cada tipo de habitación por hotel.

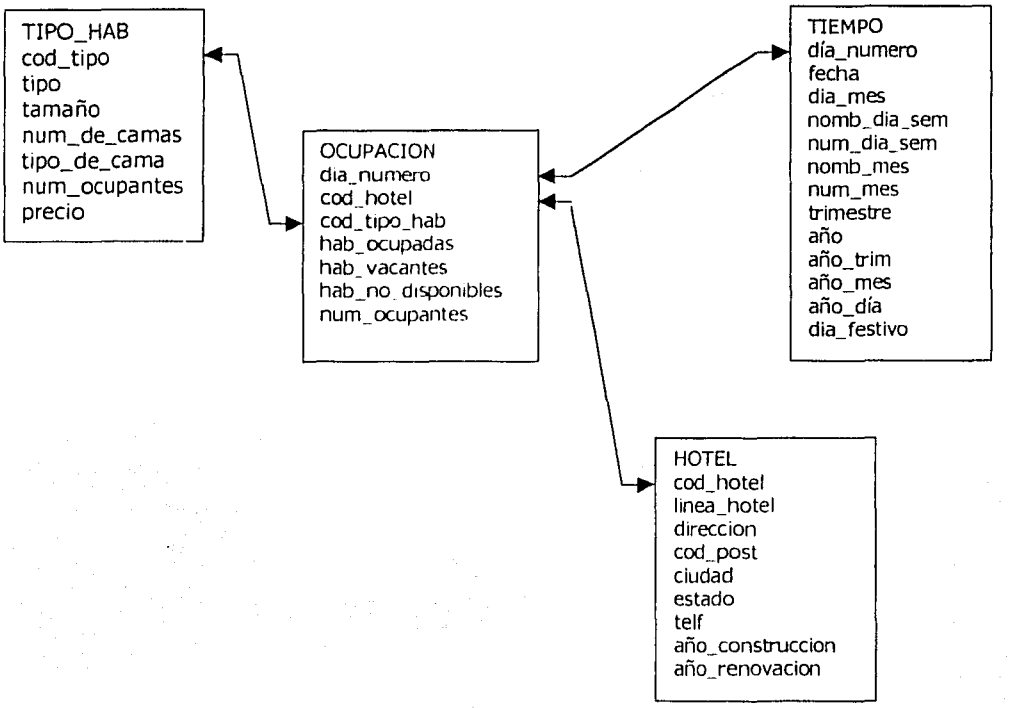



Figura 3.6. Ocupación hotelera.

En la jerarquía de la dimensión tiempo, es importante si el día es festivo. Sirve para efectuar promociones especiales en épocas de puentes vacacionales o días de asueto.

Enfoque detallado.

El precio de cada tipo de habitación es una variable dependiente de: la oferta del cliente, costos operativos, valor total (tomando en cuenta la inversión en mobiliario) e impuestos añadidos de acuerdo con la ubicación del hotel. Se obtiene fácilmente de los sistemas transaccionales.

Cuando una variable es desintegrada en elementos más sencillos, la tarea se dificulta. Muchas veces la información referente a costos operativos no está contenida en la base de datos de un sistema transaccional, sino en reportes realizados en hojas de cálculo.

Cuando los sistemas informacionales permiten obtener esta información, al modelo de la figura 3.6, se le aplican las siguientes modificaciones:

1. Añadir el atributo región con el impuesto relacionado, por hotel. Este puede obtenerse ya sea directamente o mediante cálculos al montar el data warehouse.

2. Incluir como atributos en la dimensión tipo_hab, el valor y un costo operativo asociado con el tipo, quitando precio.

3. Agregar el campo costo_operat para la dimensión hotel.

El costo operativo es distribuido entre el hotel y el tipo del mismo. De acuerdo a sus características cada habitación requerirá para su mantenimiento productos y servicios especiales. El costo del hotel incluye los honorarios del personal, gastos en energía y servicios telefónicos.

La figura 3.7, muestra el modelo de la figura 3.6 con las modificaciones señaladas.

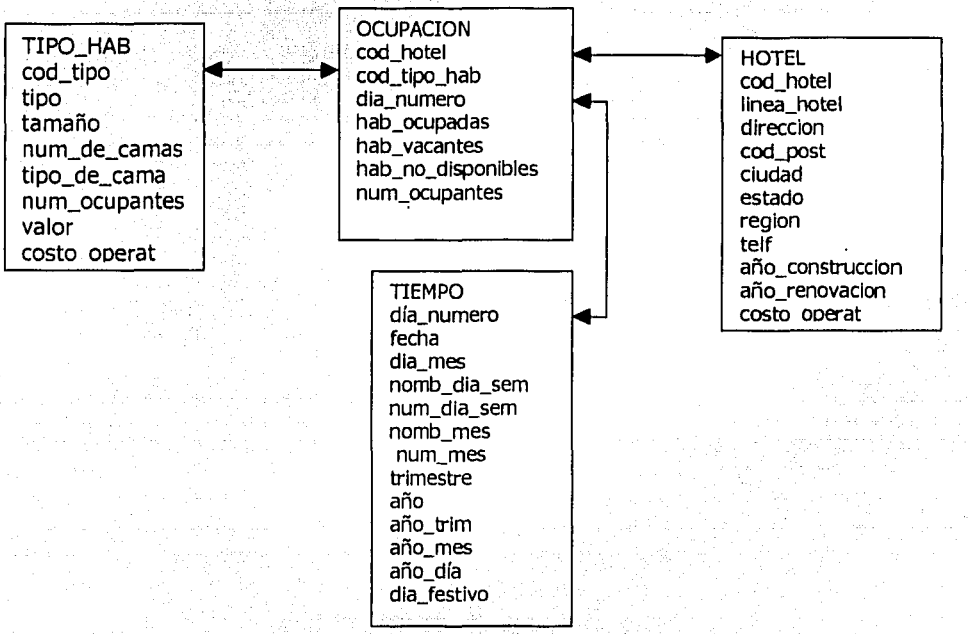



Figura 3.7. Modificaciones para modelo de Ocupación hotelera.

Sus inconvenientes son:

1. El porcentaje de impuesto regional conlleva en cada consulta junta de tablas y el cálculo correspondiente por cada hotel.

2. Si el costo operativo del hotel es desglosado en base a factores provenientes de otros procesos, será necesario relacionar la dimensión hotel con otra(s) tabla(s) proceso además de la tabla ocupación.

3. El impuesto no está contenido en una tabla, consumiendo tiempo en el proceso de carga y transformación del data warehouse.
4. Los porcentajes son medidas no aditivas. Generan errores en operaciones de roll-up con respecto al tiempo, por lo que no son almacenados en las tablas del modelo. Como solución se construyen reportes sobre la tasa de ocupación semanal, trimestral y anualmente.

Volviendo al modelo de la figura 3.6, quizás la compañía quiera conocer la información a nivel del cliente, el modelo de la figura 3.8, muestra el nivel de detalle requerido.

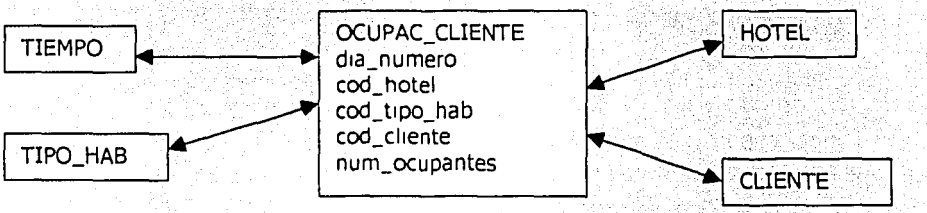



Figura 3.8, Ocupación hotelera por cliente.

Este modelo permite ver la ocupación de un tipo de habitación por hotel y por cliente. Sin embargo el número de habitaciones ocupadas no aparece ya que puede ser contabilizado en la consulta.

El problema radica en las habitaciones no disponibles y vacantes. Las variables al respecto no pueden ser guardadas en tipo_hab ya que se incrementa el número de consultas para obtener la información de todos los hoteles. Tampoco en hotel porque no podría distinguirse el tipo de habitación en las habitaciones no disponibles.

Debido a lo anterior, el modelo de la figura 3.7 ayuda a determinar la tasa de ocupación. El modelo de la figura 3.8 permite realizar análisis en puntos de venta específicos.

Al diseñar de modelos dimensionales para inventarios es conveniente:

- Entender el objetivo del inventario.
- Capturar la información necesaria para obtener las respuestas.

- Diseñar los reportes con suma cautela.
- Evitar cálculos dentro de las consultas.

La primera nos señala que los reportes del modelo son prediseñados y los porcentajes presentados solo tienen validez para el periodo del reporte.

La segunda nos recalca la conveniencia de manejar atributos y no mediciones derivadas de operaciones entre los atributos de las funciones. Las funciones de grupo utilizadas en SQL, son poco recomendables, ya que la dispersión de los datos produce fallas en el promedio aritmético. [Libro #3]

**Ventajas y desventajas de normalización entre los sistemas de transacciones y los sistemas de data warehouse.**

La normalización en sistemas de transacciones permite registrar las actividades de un departamento evitando inconsistencia y gasto en memoria necesaria, para almacenar información. Al realizar una junta sobre las tablas que forman una relación se obtienen los datos requeridos. Estos son guardados en registros identificables de forma única en cada tabla.

La desnormalización en los sistemas de información gerencial busca combinaciones entre los datos. Los modelos dimensionales cumplen con la primera forma normal, sin embargo la segunda y tercera forma normal no son observadas en esquemas estrella donde solo se realiza una operación de junta, entre la tabla dimensión y la tabla proceso. Los esquemas copo de nieve llegan a la tercera forma normal y requieren más operaciones de junta entre tablas.

En el esquema estrella las dependencias, de atributos que no son llave primaria con otro que no lo es está permitida. La tabla dimensión contiene la información de las tablas que utilizaría un esquema copo de nieve, en una sola relación.

Ante la presencia de atributos multivaluados, un sistema de transacciones utilizaría la cuarta forma normal, cuyo desempeño dependería de las combinaciones entre los atributos, sin llegar a una solución única.

Un sistema de data warehouse, puede conservar estas combinaciones. De hecho utiliza llaves derivadas para distinguir entre varias combinaciones, cuando los datos corresponden a una dimensión de lento cambio.

La dimensión tipo de habitación para el caso de ocupación hotelera, podría ser susceptible de normalización si interesará conocer la marca del colchón y si es ortopédico. Por ejemplo, estos atributos dependerían transitivamente de la llave primaria de la dimensión cod_tipo a través del atributo tipo_de_cama. Aplicando la tercera forma normal se crearía otra tabla más para el esquema de la figura 3.6.

Para efectos de consulta es mejor desnormalizar los datos sin aplicar la tercera forma normal, desapareciendo el tipo de colchón y la dimensión tipo de habitación que daría como se muestra en la figura 3.9.

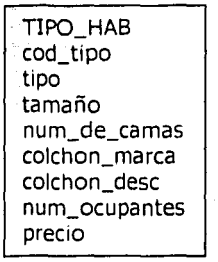



Figura 3.9. Dimensión tipo de habitación modificada.

Las entidades que forman modelo entidad-relación pueden ser atributos dentro de un modelo dimensional. Las tablas dimensión pueden normalizarse en alguno de sus atributos generando una segunda tabla. Esto sucede en el caso de la dimensión departamento para el esquema estrella de presupuesto (ver figura 2.23).

El atributo división al ser tratado como una entidad contiene los atributos: nombre, dirección, código postal, ciudad, estado, región y gerente. La tabla departamento es normalizada colocando en la tabla división los atributos mencionados y relacionarse con la tabla departamento por medio de la llave foránea división, como se muestra en la figura 3.10.

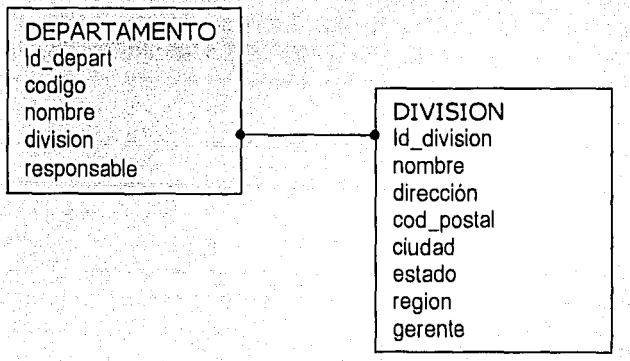



Figura 3.10. Normalización del atributo División en la dimensión Departamento.

La normalización y desnormalización presentadas sobre tablas dimensión muestra su aplicación a modelos dimensionales. Su utilidad para la organización dueña del modelo es percibida al comparar las consultas de un sistema de transacciones y otro de información gerencial.

Normalización vs. Desnormalización.

Los modelos dimensionales ejemplificados durante el capítulo pueden ser utilizados en el mundo real. A continuación se comparan los modelos de datos utilizados por un sistema de transacciones y un sistema de data mart.

El problema a modelar es la venta de automóviles en una agencia automotriz. Los datos utilizados en ambos modelos describen una idea general. El sistema de transacciones registra la factura de un automóvil. Los automóviles pueden ser austeros con características básicas para cada modelo o equipados incluyendo sistema de aire acondicionado y ventana superior (quemacoco).

Se tiene una factura por auto. La agencia se interesa en conocer las ventas realizadas por cada agente para estimar la comisión asociada. Ante una posible reclamación se requiere conocer al jefe del mismo para tomar acciones pertinentes. Sobre los datos del cliente se solicita una dirección estable y su R. F. C.

El tipo de pago determina la emisión de la factura, si el cliente paga en efectivo la obtiene al termino de la transacción. Si lo hace por medio de financiamiento se emitirá factura al terminar de pagar la unidad adquirida. Para la agencia es importante definir las características de la unidad para brindar servicio ante cualquier percance.

El modelo entidad-relación correspondiente se muestra en la figura 3.11. Sin utilizar un identificador único artificial, cada factura es reconocida por la fecha de compra y el número de serie del automóvil vendido. La entidad automóvil presenta dos subtipos austero y equipado. Con estos subtipos se obtiene una mejor descripción del vehículo dentro de la factura o carta factura (financiamiento).

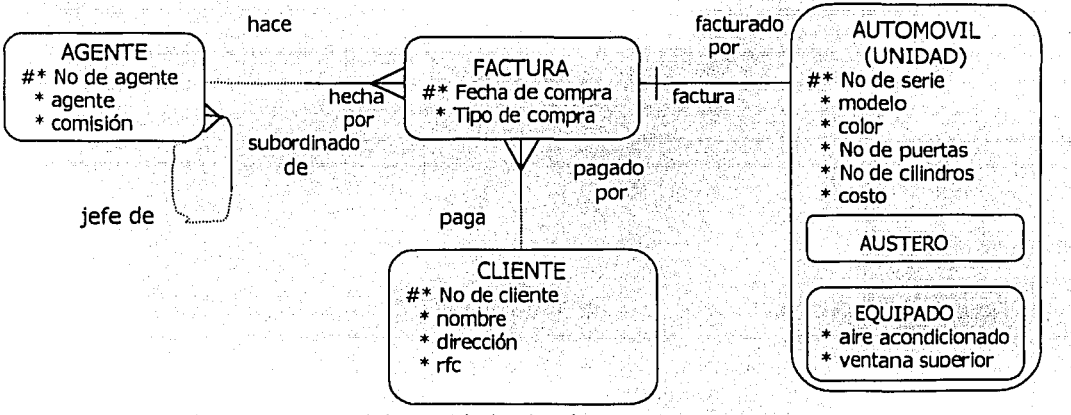



Figura 3.11. Modelo entidad-relación para ventas de autos.

La figura 3.12, muestra la relación de datos asociada a la figura 3.11. Los datos mostrados se encuentran en **1FN**. Añadiéndose el atributo jefe que permite la relación recursiva de la entidad vendedor y el atributo tipo que permite implementar los subtipos de la entidad automóvil.

| No de cliente | Nombre | Dirección | RFC | Fecha de compra | No de serie |
|---|---|---|---|---|---|
| 1 | Geovanni Rodríguez | Fresnos #15 | ROLA770808 | 29/06/2002 | 1U2WX19932K445677 |
| 4 | Jenny Galicia | Aldama #11 | GAPJ771119 | 12/10/2002 | 4Y4CH19635P225667 |
| 8 | Verónica Rangel | Esperanza #14 | RAGV761204 | 23/11/2002 | 8B1TJ12231J170089 |

| No de agente | Agente | Jefe | Comisión | Tipo de compra | Modelo | Color | No de puertas | No de cilindros |
|---|---|---|---|---|---|---|---|---|
| 8 | García | 1 | 0.2 | Crédito | Sedan | Vino | 2 | 4 |
| 4 | Noriega | 1 | 0.2 | Crédito | Sedan A3 | Azul | 4 | 4 |
| 9 | Escobar | 2 | 0.2 | Efectivo | SedanG7 | Blanco | 2 | 4 |

| Aire acondicionado | Ventana superior | Tipo | Costo |
|---|---|---|---|
| | | Austero | 105000 |
| | | Austero | 120000 |
| X | X | Equipado | 155000 |

Figura 3.12. Relación de datos para venta de automóviles.

Existen dependencias parciales con respecto a fecha de compra y número de serie del automóvil. Estas son:

Fecha de compra, número de serie ⟶ tipo de compra

Fecha de compra, número de serie ⟶ número de cliente

Fecha de compra, número de serie ⟶ número de vendedor

Aplicando **2FN** la relación original queda compuesta por R1 y R2, como lo muestra la figura 3.13.

| Fecha de compra | No de serie | Tipo de compra | No de cliente | Nombre |
|---|---|---|---|---|
| 29/06/2002 | 1U2WX19932K445677 | Crédito | 1 | Geovanni Rodríguez |
| 12/10/2002 | 4Y4CH19635P225667 | Crédito | 4 | Jenny Galicia |
| 23/11/2002 | 8B1TJ12231J170089 | Efectivo | 8 | Verónica Rangel |

Tabla 3.3. R1.

| Dirección | RFC | No de agente | Agente | jefe | Comisión |
|---|---|---|---|---|---|
| Fresnos #15 | ROLA770808 | 8 | García | 1 | 0.2 |
| Aldama #11 | GAPJ771119 | 4 | Noriega | 1 | 0.2 |
| Esperanza #14 | RAGV761204 | 9 | Escobar | 2 | 0.2 |

Tabla 3.4. R1.

| No de serie | Modelo | Color | No de puertas | No de cilindros |
|---|---|---|---|---|
| 1U2WX19932K445677 | Sedan | Vino | 2 | 4 |
| 4Y4CH19635P225667 | Sedan A3 | Azul | 4 | 4 |
| 8B1TJ12231J170089 | SedanG7 | Blanco | 2 | 4 |

Tabla 3.5. R2.

| Aire acondicionado | Ventana superior | Tipo | Costo |
|---|---|---|---|
| | | Austero | 105000 |
| | | Austero | 120000 |
| X | X | Equipado | 155000 |

Tabla 3.6. R2.

Figura 3.13. Relación Ventas de automóviles normalizada hasta 2FN.

La relación R1 contiene a los atributos que dependen transitivamente del número de serie y fecha de compra. Entre ellos se encuentran dirección, nombre y RFC que dependen de número de cliente. Al aplicarse **3FN** se obtienen las relaciones Compra, Cliente, Agente y Auto de la figura 3.14.

| Fecha de compra | No de serie | Tipo de compra | No de cliente | No de agente |
|---|---|---|---|---|
| 29/06/2002 | 1U2WX19932K445677 | Crédito | 1 | 8 |
| 12/10/2002 | 4Y4CH19635P225667 | Crédito | 4 | 4 |
| 23/11/2002 | 8B1TJ12231J170089 | Efectivo | 8 | 9 |

Tabla 3.7. Relación Compra.

| No de cliente | Nombre | Dirección | RFC |
|---|---|---|---|
| 1 | Geovanni Rodríguez | Fresnos #15 | ROLA770808 |
| 4 | Jenny Galicia | Aldama #11 | GAPJ771119 |
| 8 | Verónica Rangel | Esperanza #14 | RAGV761204 |

Tabla 3.8. Relación Cliente.

| No de agente | Agente | Jefe | Comisión |
|---|---|---|---|
| 8 | García | 1 | 0.2 |
| 4 | Noriega | 1 | 0.2 |
| 9 | Escobar | 2 | 0.2 |

Tabla 3.9. Relación Vendedor.

| No de serie | Modelo | Color | No de puertas | No de cilindros |
|---|---|---|---|---|
| 1U2WX19932K445677 | Sedan | Vino | 2 | 4 |
| 4Y4CH19635P225667 | Sedan A3 | Azul | 4 | 4 |
| 8B1TJ12231J170089 | SedanG7 | Blanco | 2 | 4 |

Tabla 3.10. Relación Auto.

| Aire acondicionado | Ventana superior | Tipo | Costo |
|---|---|---|---|
| | | Austero | 105000 |
| | | Austero | 120000 |
| X | X | Equipado | 155000 |

Tabla 3.11. Relación Auto

Figura 3.14. Relación Ventas de automóviles normalizada hasta 3FN.

Las relaciones presentadas equivalen a las tablas: cliente, auto, vend y compra. Las tablas formarían parte de una base de datos relacional para un sistema de transacciones. El siguiente código **SQL** genera La consulta mostrada en la tabla 3.12.

```
Select nombre, modelo, costo, tipo_compra
From cliente c1, auto a, compra c
Where c.num_serie = a.num_serie and
      c.num_cliente = c1.num_cliente and
      costo <= 130000;
```

| Nombre | Modelo | Costo | Tipo de compra |
|---|---|---|---|
| Geovanni Rodríguez | Sedan | 105000 | Crédito |
| Jenny Galicia | Sedan A3 | 120000 | Crédito |

Tabla 3.12. Consulta sobre nombre del cliente, tipo de compra, modelo y costo del automóvil.

Los datos normalizados de las tablas 3.7 a la 3.11, son utilizados por una agencia. Si una empresa manejará varias agencias, la visión de las transacciones sería global. En vez de consultar datos sobre una factura se analizarían las ventas llevadas a cabo en cada sucursal (agencia).

Las sucursales se agruparían por estado y región. Los datos sobre los agentes serían triviales, siendo más importante conocer el total de las ventas en un determinado periodo.

Nuevamente un departamento de mercadotecnia necesitará un sistema de información gerencial que muestre las ventas de automóviles para todas las agencias. Las dimensiones utilizadas en el modelo dimensional requerido son: tiempo, localización y auto. La dimensión tiempo se forma a partir del atributo fecha_compra.

La jerarquía propuesta para la dimensión tiempo es:

fecha < semana < mes < trimestre < año

La dimensión localización permite sumar los totales por estado o región, presentando la siguiente jerarquía:

número de localidad < sucursal < estado < región

Las figuras 3.15 y 3.16 muestran las relaciones entre los atributos que describen a los dos tipos de automóviles presentes en la dimensión Auto. Estas figuras también definen jerarquías entre los atributos de la tabla Auto. Ambos tipos descritos comparten atributos comunes como color, modelo, número de puertas y número de cilindros. Como el total de ventas no se ve afectado por las características propias de cada tipo, no se requiere contemplar la creación de tablas núcleo y tablas de características.

Una dimensión producto puede manejar ambos tipos. El equipo de data warehousing analizaría la conveniencia de crear tablas agregadas para un tipo y nivel de detalle particular, cuando las consultas para esta combinación fueran recurrentes. Adjuntándolas al modelo dimensional propuesto en la figura 3.17.

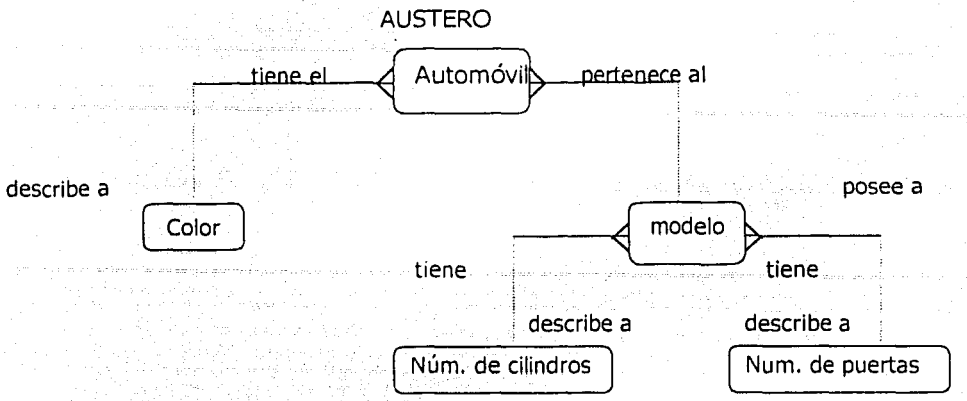



Figura 3.15. Relaciones entre atributos que describen a un vehículo austero.

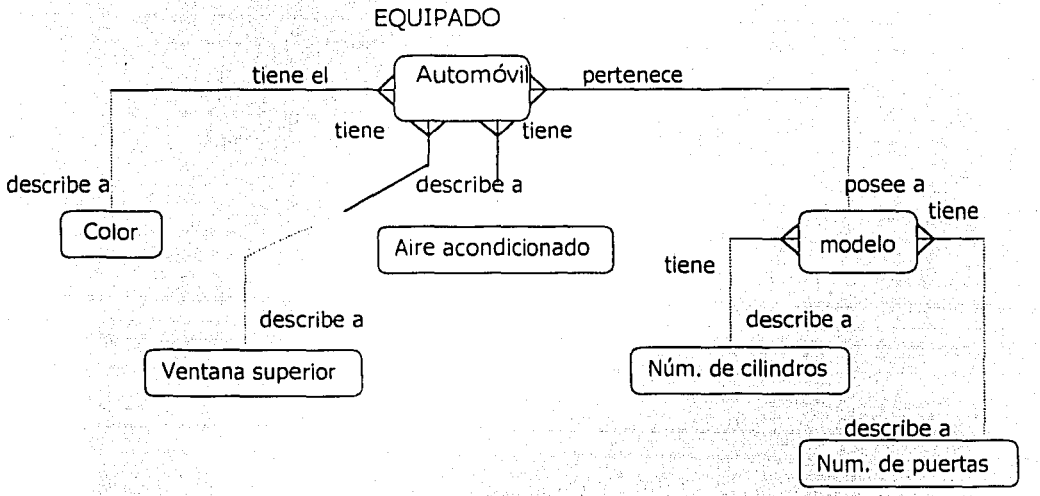



Figura 3.16. Relaciones entre atributos que describen a un vehículo equipado.

La relación Auto tablas 3.10 y 3.11, forman la dimensión producto de la figura 3.17. La consulta propuesta para el sistema e transacciones sobre el esquema de ventas para vehículos austeros corresponde al siguiente código **SQL**.

```
Select nombre, modelo, total, tipo_compra
From cliente, ventas v, auto a, tiempo t
Where v.Num_cliente = cliente.Num_cliente and
      v.Num_serie = a.Num_serie and
      v.Fecha = t.Fecha and
      costo <= 130000;
```

El resultado de la consulta incluirá más registros que en un sistema de transacciones. Se tienen tres juntas que permiten analizar los datos con respecto a las jerarquías de tiempo y localización. Las jerarquías de auto figuras 3.15 Y 3.16, pueden ser utilizadas para construir condiciones al consultar características del automóvil, en vez de operaciones como *roll-up*.

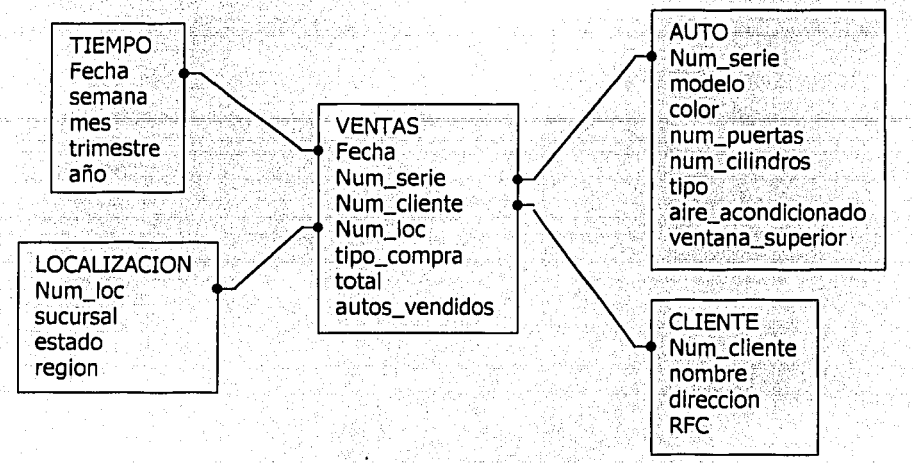



Figura 3.17. Esquema para Ventas de automóviles.

La normalización presente en las tablas de la figura 3.14, beneficia a un sistema de transacciones para almacenar consistentemente la información pertinente a una factura a lo largo de un año. El esquema para ventas de automóviles por su desnormalización disminuye el trabajo realizado por la maquinaria de software **ROLAP** para realizar consultas y obtener un gran número de registros en poco tiempo.

Como total es una variable aditiva, las operaciones de *roll-up* y *drill-down* se realizan sin novedad. La complejidad de las consultas se localiza en las condiciones que se efectúen sobre los atributos de las tablas dimensión y las tablas proceso.

Normalización conlleva actualización constante y reducción de espacio en memoria, se aconseja en sistemas de transacciones por su utilidad en altas, bajas y modificaciones sobre los datos. Las consultas aunque llegarán a ser elaboradas por lo general quedan definidas en aplicaciones de texto o gráficas, enfocadas hacia usuarios finales que no toman decisiones. Su tiempo de ejecución con respecto a sistemas de información gerencial es pequeño puesto que la base de datos es una porción de la que se tiene en un data mart o en un data warehouse.

La normalización se presenta en sistemas de información gerencial dentro de modelos dimensionales utilizados por toda la empresa bajo el enfoque de Entrerprise Data warehouse. Se aplica a modelos dimensionales de consultas poco sofisticadas donde la base de datos desnormalizada consume mucha memoria.

Generalmente para permitir mayor interacción del usuario con la base de datos, se construyen sistemas de data mart (dependientes o independientes) donde uno o varios modelos dimensionales son creados a partir de los datos extraídos de varios sistemas de transacciones. Las tablas de los modelos pueden estar desnormalizadas si fuera conveniente, siendo el tamaño de la base de datos controlable.

La desnormalización no es frecuentemente en sistemas de transacciones. Se presenta ante la no aplicación de **4FN** a atributos multivaluados. Aparte **5FN** tiene aplicación ideal más que práctica. Considerando como grado óptimo de normalización la aplicación de **3FN** se puede decir que la base de datos de un sistema de transacciones no se encuentra desnormalizada. Desnormalizar genera errores de consulta de los datos guardados.

En cambio desnormalizar y normalizar pueden ir de la mano en sistemas de información gerencial, existe cierta flexibilidad ante la aplicación de las formas normales de mayor uso ( de **1FN** a **3FN**). El uso de alguna de ellas, depende de la experiencia que tenga un equipo de data warehouse sobre su eficacia en consultas solicitadas por quienes toman decisiones.

La desventaja de normalizar en data warehouse se manifiesta en la junta de muchas tablas. Su ventaja es eliminar cierta redundancia para modelos cuyas consultas no son afectadas por esta, en la ejecución de sentencias **SQL**.

# Conclusiones.

La presente tesina cumple con el objetivo descrito en la introducción. En sus capítulos II y III se exponen soluciones de consulta modeladas sobre tablas relacionadas desnormalizadas para sistemas de información gerencial.

Para destacar la importancia de la desnormalización (o normalización según convenga) en los modelos dimensionales, se hace una comparación entre un modelo relacional asociado a un sistema de transacciones y un modelo dimensional para implementar un data mart en agencias de automóviles. Se contrastan los niveles de normalización de cada modelo, explicándose cómo afectan la normalización/oesnormalización de datos a los sistemas de transacciones y a los sistemas de información gerencial.

En resumen, los sistemas de data warehouse son sistemas de apoyo a la toma de decisiones, que trabajan sobre datos históricos y se alimentan de sistemas de transacciones. Estos se construyen sobre software **ROLAP** o **MOLAP**.

**ROLAP** trabaja sobre bases de datos relacionales usando un conjunto de tablas desnormalizadas denominado modelo dimensional. **MOLAP** trabaja sobre bases de datos multidimensionales, para consultas elaboradas.

Ambos utilizan herramientas visuales donde el usuario establece criterios sobre las dimensiones de un cubo de datos conceptual. En **ROLAP** se generan consultas **SQL** trasparentes para el usuario y en **MOLAP** los datos son extraídos del arreglo multidimensional localizando la celda indicada.

**HOLAP** es la tendencia de mezclar las mejores características de **ROLAP** y **MOLAP**. Como consecuencia de conjuntar ambas tecnologías su administración es difícil y utiliza recursos extra como cache multidimensional y vistas materializadas.

La importancia de un data warehouse radica en el análisis de grandes volúmenes de información histórica. Obteniéndose respuestas para evaluar presupuestos, conocer estados de ventas y comportamiento de los clientes. Los resultados van encaminados a temas relacionados con la productividad de una organización. De hecho los primeros desarrollos en sistemas de información gerencial se hicieron para los departamentos de mercadotecnia.

¿Porqué utilizan desnormalización los modelos dimensionales? La respuesta implica otra pregunta ¿Por qué la normalización es casi inexistentes en dichos modelos?.

La normalización es una herramienta en el diseño de base de datos, pero no es una varita mágica. Teóricamente es deseable su aplicación hasta **5FN**, sin embargo muchos de los conjuntos de datos son normalizados aceptablemente hasta la **3FN**. Desnormalizar implica, conservar dependencias entre los atributos de una tabla, omitiendo **2FN** y **3FN**, como sucede en el esquema estrella que presenta el mayor grado de desnormalización. El esquema copo de nieve alcanza la tercera forma normal.

La conveniencia de normalizar una relación de datos consiste en evitar su redundancia e inconsistencia. Este último aspecto interesa a los sistemas de transacciones

basados en software **OLTP**, donde los datos requieren estar actualizados constantemente. Con la normalización es posible implementar programas que vigilen la integridad de los datos, para efectuar operaciones de lectura y escritura sobre ellos.

La desnormalización permite observar cambios en las variables de un proceso a través del tiempo. Los sistemas de información gerencial basados en software **OLAP**, requieren desnormalización. Con ella las consultas en **SQL** sobre el modelo dimensional, no utilizan demasiadas juntas entre tablas, las cuales serían consecuencia de aplicar **2FN** y **3FN**. Dichas consultas son extensas en código e incluyen diversas condiciones o cláusulas GROUP BY para el caso de las agregaciones.

Para disminuir el tiempo de ejecución y el consumo de memoria se utilizan modelos copo de nieve. Estos permiten normalizar datos que al conservarse en un esquema estrella ocuparían varios terabytes de espacio. Por otro lado la desnormalización ayuda a que los datos no sean fácilmente actualizados, ya que en los sistemas de información gerencial, se realizan operaciones de lectura sobre información obtenida de los sistemas de transacciones.

Los esquemas estrella son usados para desarrollar data mart, donde la información desnormalizada es pequeña. Los esquemas copo de nieve son implementados para **enterprise data warehouse**, donde la normalización hasta **3FN**, permite reducir el consumo de recursos en el servidor **OLAP**.

Normalizar o desnormalizar un conjunto de datos obedece al uso que se hará de los mismos. Si es necesario leer y actualizarlos se aconseja normalizar. Cuando los datos son leídos únicamente se aconseja desnormalizar o normalizar parcialmente de acuerdo con los requerimientos de consulta y la complejidad de código **SQL** que la normalización lograría simplificar.

En nuestro país, la tecnología de base de datos no se encuentra al alcance de las empresas medianas. Los sistemas de data warehouse son utilizados por sucursales de empresas transnacionales u organismos gubernamentales. Como consecuencia existen pocos analistas con experiencia suficiente para desarrollar modelos dimensionales.

# Apéndice A.

# Modelos lógicos para bases de datos.

Antes de definir los modelos lógicos se explica brevemente el modelo entidad-relación como modelo conceptual que es implementado en cualquiera de los modelos lógicos.

Modelo Conceptual.

*Modelo Entidad-Relación.* Propone la perspectiva de la información referida en una base de datos en base, a entidades y relaciones.

- Una entidad es cualquier cosa u objeto real distinguible de otros objetos y descrita por medio de atributos.
- Una relación es una asociación entre varias entidades.
- Un atributo es un dato específico que requiere ser almacenado.
- Una instancia es una cosa u objeto específico, para la entidad empleado una instancia es Juan Gutiérrez.

Además representa ciertas ligaduras que los contenidos de las bases de datos deben cumplir. La ligadura implica una correspondencia de cardinalidades (uno a uno, uno a varios, varios a varios), que expresa el número de instancias de una entidad, con las cuales se pueden asociar las instancias de una segunda entidad a la primera. Tiene como componentes:

- Rectángulo.- Representa una entidad.
- Elipses.- Representa un atributo.
- Rombos.- Representan relaciones entre entidades.
- Líneas.- Las cuales unen a los atributos con sus entidades y a las entidades con sus relaciones.

Ejemplo.

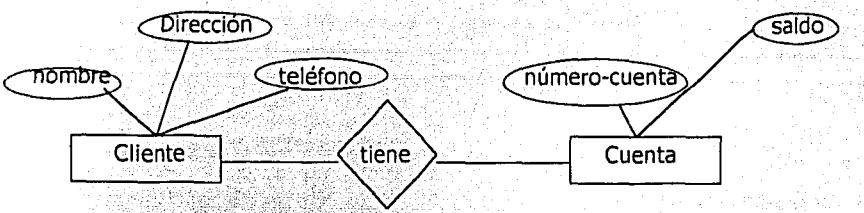



Figura A.12. Modelo Entidad-relación para cliente-cuenta.

Lógicos que utilizan registros.

Llamados así porque emplean una estructura de formato fijo denominada registro, constituida por un conjunto de campos o atributos de longitud fija (la cual puede variar en el modelo relacional).

*Modelo jerárquico.* Se representa la información mediante diagramas de árbol, compuestos por dos componentes fundamentales: las cajas que representan registros y las líneas que representan enlaces entre estos. Los enlaces, establecen correspondencia entre la información contenida en los campos, de los registros.

Se necesita un registro raíz y se permiten representar relaciones, uno a uno, figura A.1. Los diagramas de árbol pueden llegar a ser complicados, sobre todo cuando se tiene la correspondencia de uno a muchos ó muchos a muchos, entonces, se sugiere la creación de dos árboles, uno para cada registro de la relación, sin embargo esto genera duplicidad e inconsistencia, así como saturación de espacio.

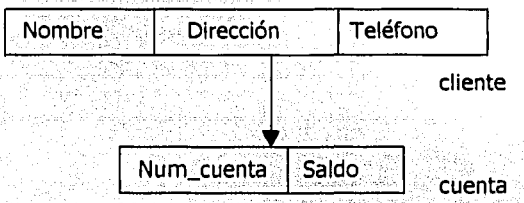



Figura A.1. Modelo jerárquico cliente-cuenta.

La base de datos que pude generarse de este diagrama de árbol, se muestra en la figura A.2.

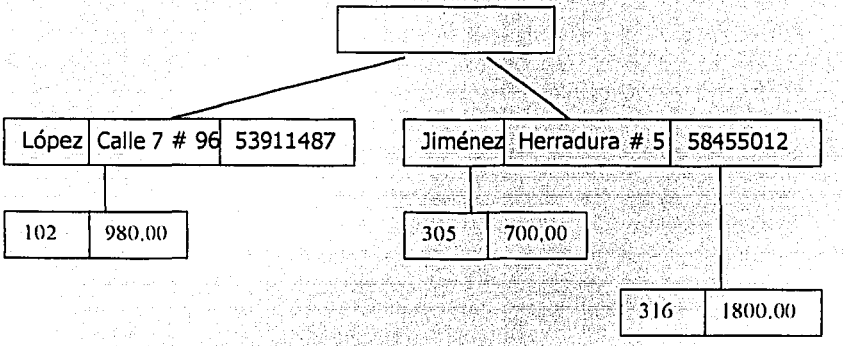



Figura A.2. Base de datos de tipo jerárquico para cliente-cuenta.

El modelo jerárquico señala que una cuenta no puede pertenecer a más de un cliente. Si la relación es muchos a muchos, se crean dos árboles, A y B.

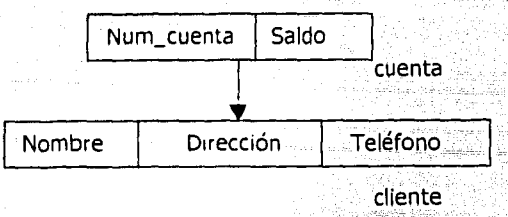



Figura A.3. Diagrama de árbol A.

El diagrama de árbol A, tiene un enlace de varios a uno, donde el registro cliente es el nodo raíz y su nodo hijo es el registro cuenta, figura A.3.

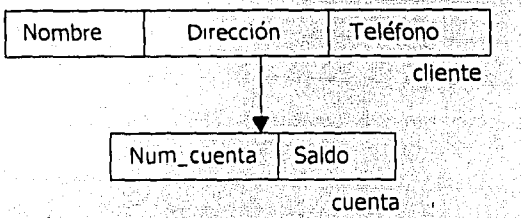



Figura A.4. Diagrama de árbol B.

El diagrama de árbol B, tiene un enlace de varios a uno, donde el registro cuenta es el nodo raíz y su nodo hijo es el registro cliente, figura A.4.

De aquí se desprenden dos posibles bases de datos, mostradas en las figuras A. 5 y A.6. La cuenta x, aparece dos veces en el primer diagrama y el cliente y aparece dos veces en le segundo. Esto demuestra la redundancia que el modelo jerárquico provoca en un momento dado.

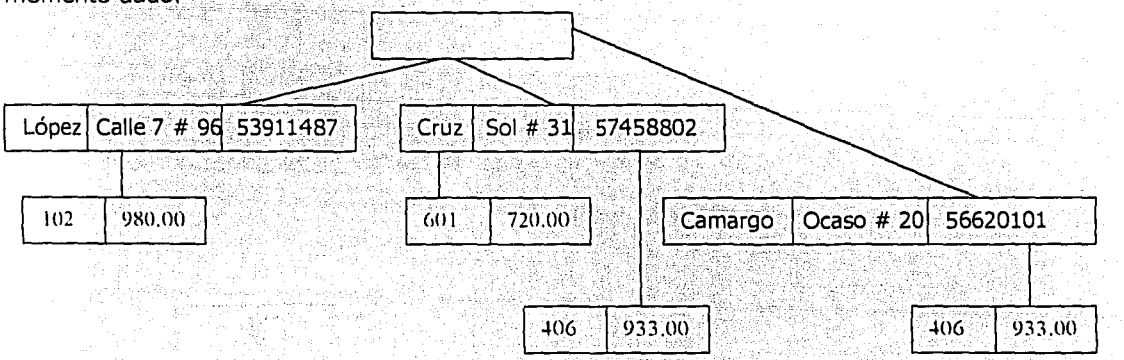



Figura A.5. Base de datos para el diagrama de árbol B.

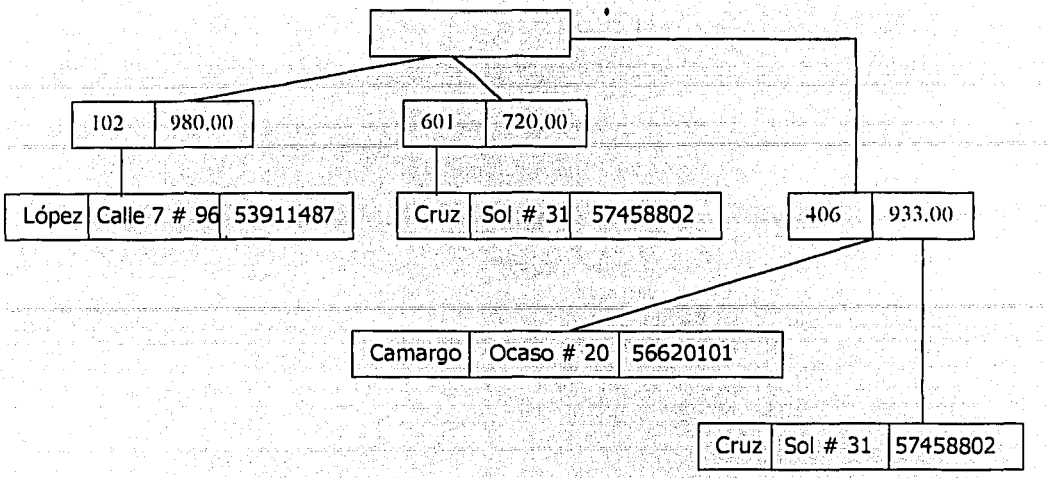



Figura A.6. Base de datos para el diagrama de árbol A.

*Modelo de red.* La información se representa en colecciones de registros (como en la sintaxis del lenguaje de programación PASCAL) y las relaciones entre los registros se denominan enlaces (punteros), dentro de diagramas de estructura de datos (conjunto DBTG), donde se tiene un registro propietario y un registro miembro (a quien se indica con la flecha del propietario).

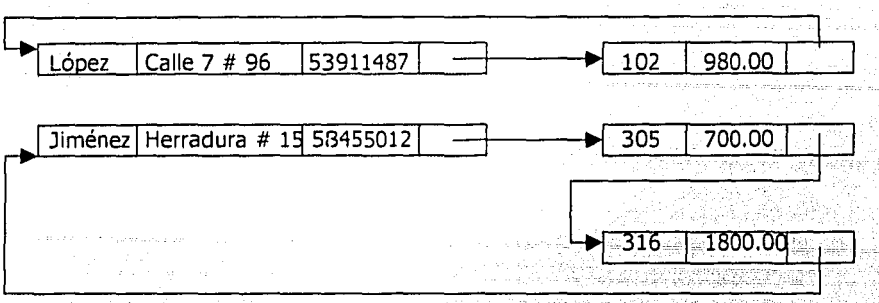

miembro o hijo

propietario o padre

Figura A.7. Conjunto DBTG.

Para definir los enlaces es necesario añadir un campo puntero a los registros miembro, el cual apunta al propietario; para asociar registros propietarios se utilizan estructuras en anillo, en dichas estructuras tanto los miembros como los propietarios se organizan en listas circulares (habiendo una lista circular para cada registro propietario).Los diseñadores de bases de datos tienen que crear datos artificiales para relaciones varios a varios, es necesario navegar la información mediante el uso de punteros por lo que su consulta es complicada.

Figura A.8. Estructura en anillo, para cuenta-cliente.




La figura A.9, muestra el modelo de red, para asociar registros de copia y película.

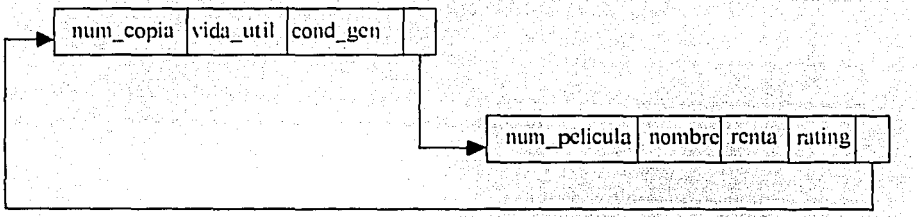



Figura A.9. Modelo en red para copia-película.

En la figura A.9, solo se muestra el caso donde a una película le corresponde una copia, si se tuvieran dos o más sería necesario utilizar una estructura anillo como la descrita en la descrita en la figura A.10.

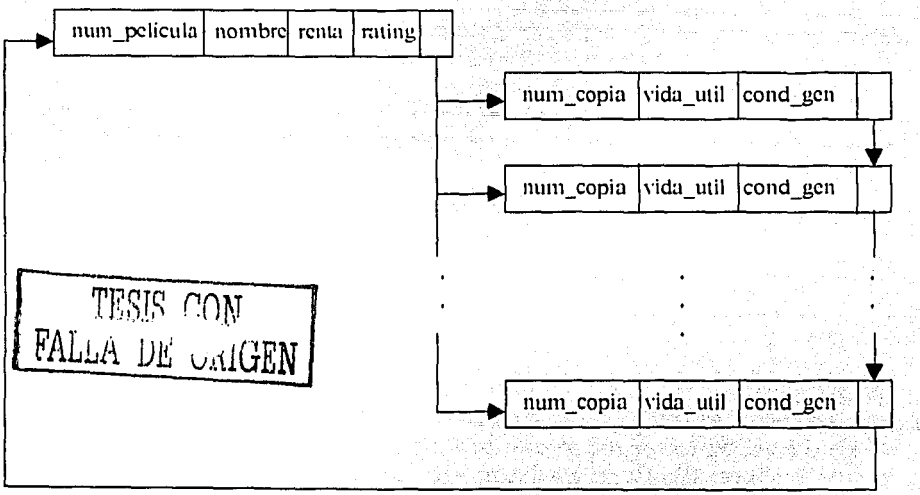





Figura A.10. Modelo en red para copia-película.

*Modelo relacional.* Se utilizan tablas para guardar la información, así como, diagramas entidad – relación que sirven para consultarla. Es implementado en un RDBMS, el cual vigila la integridad y consistencia de los datos, al mismo tiempo que ofrece concurrencia a los usuarios.

Las tablas son un conjunto de registros y columnas, en cuya intersección un dato es almacenado. Existen entre las columnas, dos tipos de interés, la llave primaria y la llave foránea.

La llave primaria identifica cada tupla o registro de forma única, mientras que la llave foránea, es el campo que guarda una correspondencia de los datos contenidos en él, con la llave primaria de otra tabla, de ahí que a través de la llave foránea una tabla A y una tabla B se encuentren relacionadas.

Como lo muestra la figura A.11,donde la tabla A es la tabla cliente y la tabla B es la tabla cuenta. La llave foránea es el campo Número-cuenta dentro de la tabla cliente.

| Nombre | Dirección | Teléfono | Número-cuenta | NIP |
|---|---|---|---|---|
| López | Calle 7 # 96 | 53911487 | 102 | *** |
| Jiménez | Herradura # 15 | 58455012 | 316 | ***** |
| Ortiz | Palmas # 3 | 60081473 | 405 | ** |
| Jiménez | Herradura # 15 | 58455012 | 305 | ***** |

Tabla Cliente

| Número-cuenta | Saldo |
|---|---|
| 316 | 1800.00 |
| 102 | 980.00 |
| 405 | 1050.00 |

Tabla cuenta

Figura A.11. Modelo relacional cliente-cuenta.

E. F. Codd, desarrolló el álgebra relacional y definió operaciones de selección sobre registros por medio de condiciones, junta de tablas, uniones e intersecciones entre consultas.

Lógicos que utilizan objetos.

Se basan en objetos que describen los datos a nivel lógico y de vista.

*Modelo Orientado a objetos*

Basado en una colección de objetos (o variables ejemplo, con fragmentos de código llamados métodos y características distintivas propiedades) los cuales al contener propiedades o métodos semejantes se clasifican en clases. Para acceder a la información de un objeto es necesario invocar a un método del objeto (paso de mensaje). Aquí dos objetos que contienen los mismos valores son completamente diferentes y su distinción

depende de identificadores. La ventaja que ofrecen es que no se necesita alterar varios segmentos de código existente para una modificación, simplemente se modifica al método asociado con el efecto que queremos darle a la información que proporciona el objeto.

Físicos.

Se refieren al implementación en el sistema de bases de datos y solamente se conocen dos: modelo de unificación y modelo de memoria por marcos. [Libro # 7]

# Apéndice B.

# Notación Case*method.

La notación Case*Method se utiliza para la construcción de un modelo relacional. Basado en el modelo entidad-relación, se representan entidades y relaciones bidireccionales, de la entidad A hacia la entidad B y en sentido contrario. También se representa el caso donde la entidad se relaciona consigo misma, relación reflexiva.

Los atributos están contenidos en su entidad y las relaciones tienen dos características: cardinalidad y opcionalidad.

Entidades.

La entidad se describe con rectángulos de esquinas redondeadas, su nombre en singular, es colocado con letras mayúsculas y dentro de paréntesis se colocan sus alias.


Antes de modelar una entidad se aconseja preguntarse lo siguiente:

¿Cuáles son los sujetos u objetos de interés para el negocio?
¿Existe información interesante acerca de la entidad que es necesario conservar?

Relaciones.

Las relación bidireccional se representa como una línea que une a las entidades. La opcionalidad de una relación, se refiere a la existencia de esta, entre todas las instancias de las entidades relacionadas.

Se aconseja verificar la relación de cada par de entidades, luego definir el nombre que llevará en cada dirección, el grado y la opcionalidad de esta.

Existirán instancias de una entidad A, que siempre estarán asociadas a otras en la entidad B, por lo tanto, en la dirección de A hacia B, la relación es obligatoria. En otro caso, existirán instancias de B, que no necesariamente estarán asociadas a las instancias de A, por lo tanto, en la dirección de B hacia A, la relación es opcional. Como lo muestra la figura B.1.

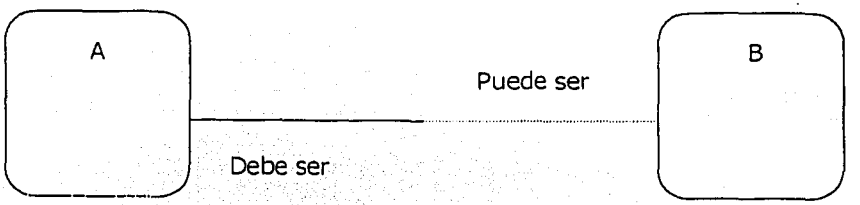
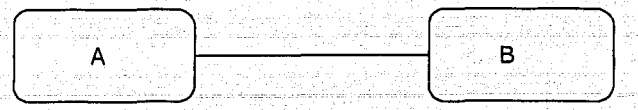
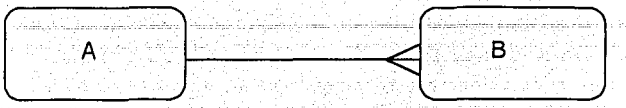



Figura B.1. Opcionalidad de relaciones.

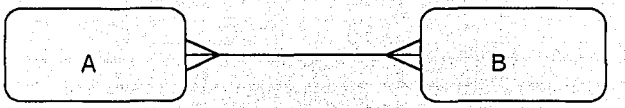
La cardinalidad coincide con los tipos del modelo entidad-relación y se representan en la figura B.2.




Uno a uno.




Uno a muchos.




muchos a muchos.

Figura B.2. Cardinalidad de las relaciones en notación Case*Method.

Atributos.

Los atributos, pueden contener valores nulos. Dentro de una entidad, existen atributos como la llave primaria que no admiten valores nulos, la notación para resaltar esta característica en un atributo es:

* Obligatorio. Cuando el atributo o permite valores nulos.
° Opcionales. Cuando si valores nulos.

Si los atributos necesitan descomponerse, puede tratarse de otra entidad o bien, carece de significado. Es conveniente verificar que un atributo no sea derivado, o calculado de los valores existentes de otros atributos.

Identificadores únicos.

Un identificador único es una combinación de atributos y/o relaciones para identificar de forma única cualquier instancia de una entidad. Estos deben ser obligatorios y para resaltarlos se usa #*.

La figura B.3 muestra a la entidad proyecto y sus atributos obligatorios y opcionales.


Figura B.3. Representación de entidad con sus atributos.

En el caso de las entidades cuenta y banco, la cuenta es identificada por un número de cuenta y el banco específico a la que esta relacionada, entonces se usa una barra para aclarar que la relación es parte del identificador único de la entidad, como lo muestra la figura B.4.

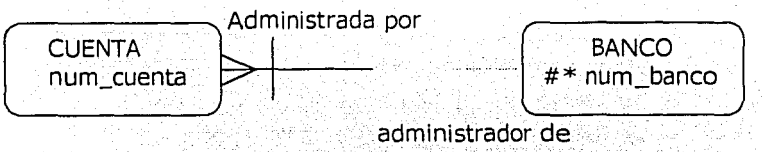



Figura B.4. Relación cuenta-banco.

Relaciones muchos a muchos.

Cuando se tienen relaciones muchos a muchos se utiliza una entidad intersección para modelar relaciones muchos a uno. Estas últimas son obligatorias. Se aconseja resolver las relaciones muchos a muchos, cuando se ha completado el modelo entidad-relación.

En la figura B.5, se tiene una relación muchos a muchos entre la entidades producto y vendedor. La entidad intersección necesaria, se denomina venta y tiene como atributo adicional a fecha.

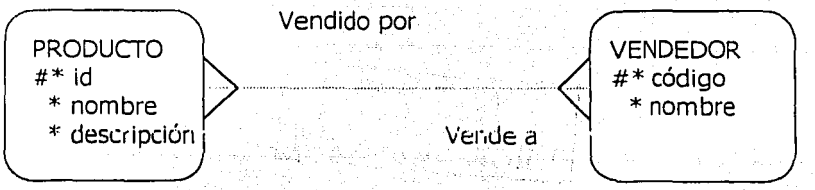



Figura B.5. Relación entre las entidades Producto y Vendedor.

La figura B.6, muestra la transformación de la relación vende a / vendido por.

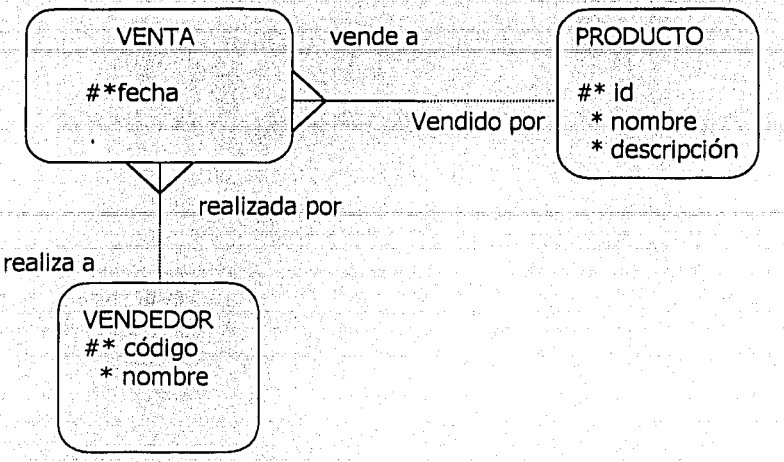



Figura B.6. Modelado con realidad intersección.

Subtipos.

Los subtipos se utilizan para modelar tipos de entidad con atributos o relaciones comunes. Un ejemplo claro sucede con los empleados de una compañía, una porción es asalariada y el resto trabaja por honorarios. Los datos a comunes a ambos tipos son: num_emp, nombre, apellido y departamento. Los atributos específicos son para los empleados asalariados su salario correspondiente y para quienes perciben honorarios: costo por hora, costo por tiempo extra y sindicato.

El supertipo empleado contiene los atributos comunes, cada subtipo define sus propios atributos y relaciones. Dentro de la tabla que representará a la entidad empleado, el subtipo empleado asalariado tendrá valores nulos en los atributos del subtipo empleado por honorarios. Lo mismo sucede en el caso de este último, donde no aplica el atributo salario. Además se incluye un atributo que indica el suptipo. La figura B.6, muestra el modelo entidad-relación asociado al ejemplo.

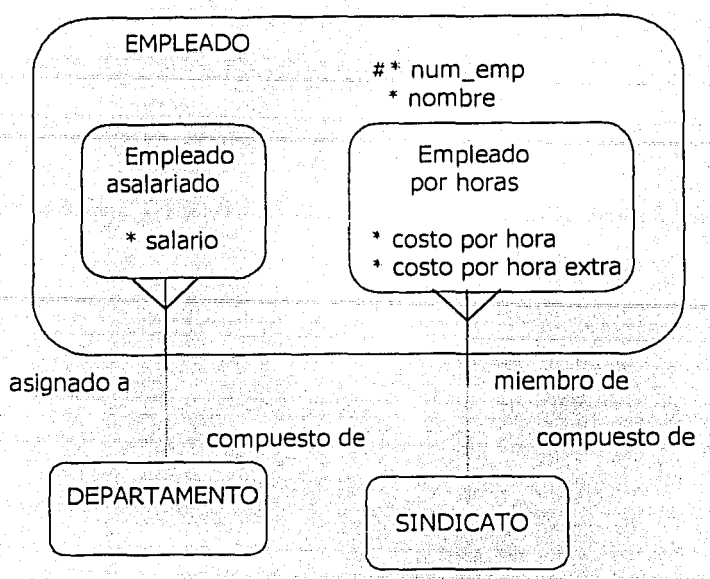



Figura B.7. Supertipo empleado.

Relaciones excluyentes.

Cuando una entidad se relaciona con dos o más entidades de forma mutuamente excluyente, las relaciones son representadas con el uso de un arco, que considere los siguientes lineamientos.

- Todas las relaciones modeladas deben ser obligatorias ú opcionales.
- Cada arco pertenece a una sola entidad y las relaciones incluidas solo deben corresponder a esa entidad.
- Las entidades pueden tener múltiples arcos, sin embargo la relación solo debe ser incluida en un solo arco.

Para ejemplo, se plantea el caso de la entidad cuenta bancaria que debe pertenecer a un cliente ó a una empresa, en la figura B.8.

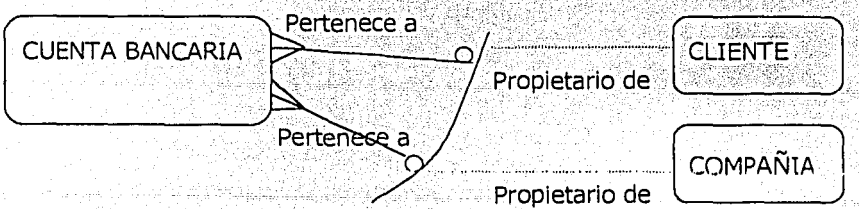



Figura B.8. Relaciones excluyentes entre cuenta bancaria-cliente y cuenta bancaria-compañía.

Roles.

Cuando una entidad en un modelo entidad relación realiza funciones distintas, es necesario representar a la entidad con un nombre distinto de acuerdo a su función. Cuando esto sucede, las instancias de las entidades que representan roles se traslapan. En la figura B.9, se representa el caso donde en una escuela un estudiante puede ser instructor. [Referencia # 9]

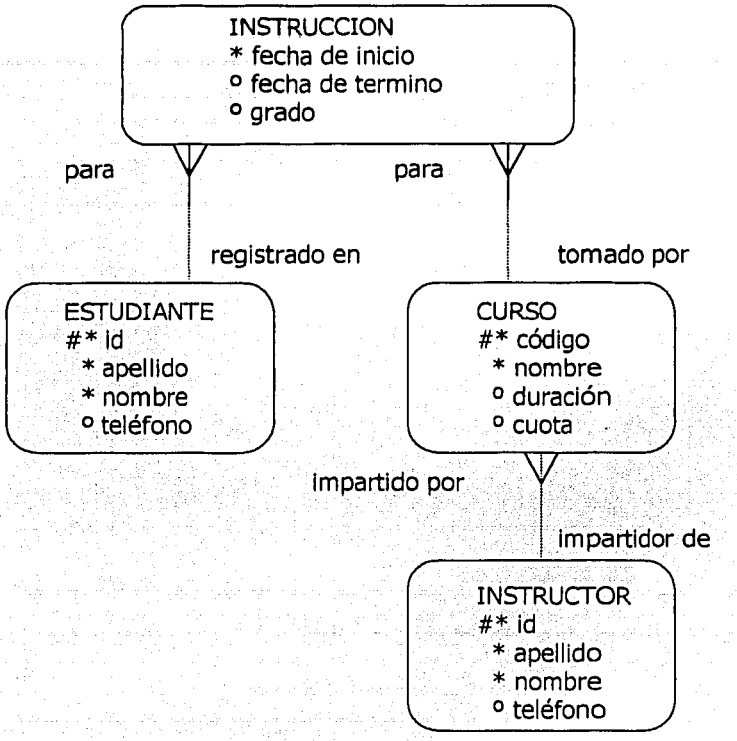



Figura B.9. Entidad estudiante con roles de estudiante e instructor.

# Apéndice C.

# Conversión de Modelo Entidad-Relación a Modelo Dimensional.

Construcción de un modelo entidad-relación.

1. Definir entidades.
2. Identificar atributos.
3. Definir relaciones.
4. Asignar identificadores únicos a las entidades.
5. Si se requiere identificar subtipos y supertipos.

Si se toma como ejemplo, obtener el diagrama asociado a las ventas de una compañía:

1) Las entidades asociadas son:

- Región
- Almacén
- Producto
- Departamento
- Producto
- Inventario
- Ventas

Utilizando la notación Case*method y después de definir las relaciones, identificar atributos y asignar identificadores únicos, se obtiene el diagrama de la figura C.1.

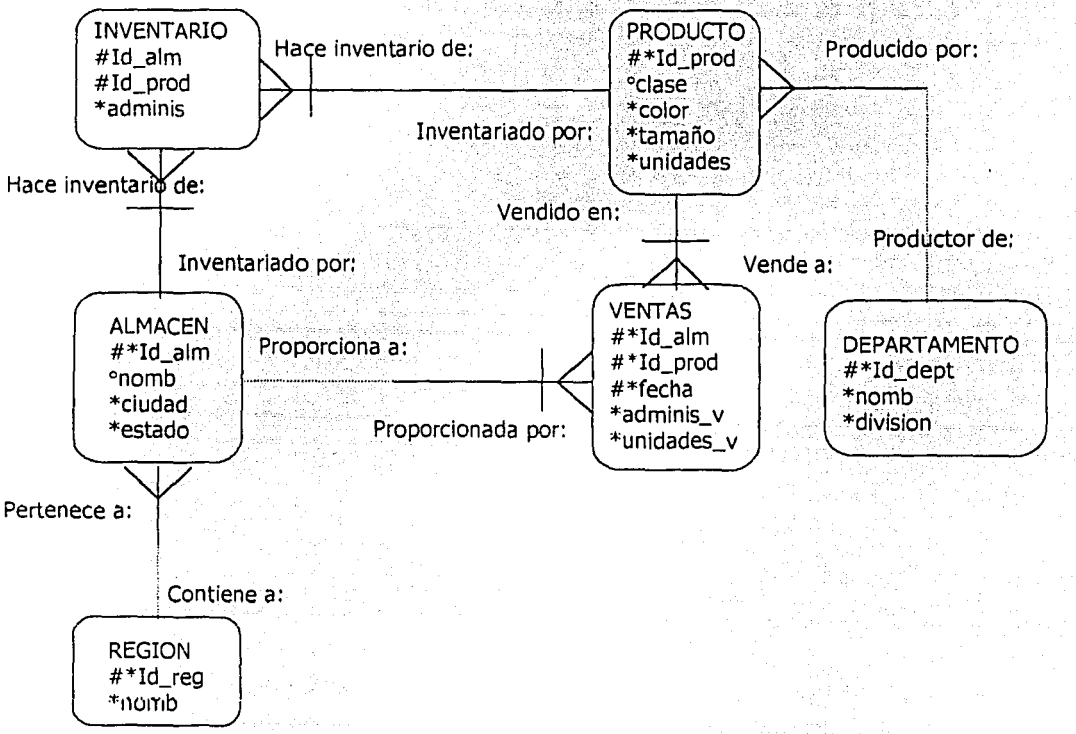



Figura C.1. Modelo entidad-relación para ventas.

5) Los subtipos son utilizados para simplificar un modelo generalizando entidades similares. Este ejemplo no define subtipos o supertipos, por lo que ha concluido su construcción.

Transformar un modelo entidad-relación a modelo dimensional.

1) Definir dimensiones.
2) Transformar las entidades actuales en dimensiones.
3) Identificar las entidades que describen un proceso.
4) Normalizar atributos. Los atributos son las entidades de un modelo dimensional.
5) Organizar los atributos dimensionados jerárquicamente.
6) Asignar identificadores únicos.

Para ejemplificar el modelo entidad-relación de la figura C.1, se transformará en un modelo dimensional.

Paso 1)

Las dimensiones son entidades abstractas en este caso el atributo fecha se transforma en la dimensión tiempo. Otras dimensiones propuestas son ubicación y producto.

Paso 2)

Las entidades como tales no existen en el modelo dimensional, muchos de las entidades y atributos de un modelo entidad-relación se transforman en dimensiones.

En el ejemplo, ubicación es una dimensión compuesta por las entidades almacén y región. La dimensión producto queda conformada por las entidades producto y departamento.

Paso 3)

Las entidades que describen procesos se transforman en tablas proceso. Como es el caso de ventas e inventario.

Paso 4)

Todos los atributos son tratados como si fueran entidades. Ambos atributos y entidades llegan a ser atributos de sus respectivas tablas dimensión. De este modo, los atributos estado, ciudad, junto con las entidades, región y almacén forman la dimensión ubicación mostrada en la figura C.2.

Los atributos clase, color y tamaño de la entidad producto y el atributo división de la entidad departamento, son tratados al mismo nivel. El atributo división es escogido para posible normalización del modelo dimensional. Estos elementos forman la dimensión producto mostrada en la figura C.3.

Se crea la dimensión tiempo y sus atributos, necesarios para realizar reportes de ventas e inventario, estas últimas son entidades que se transformarán en tablas proceso. La figura C.4, como las figuras C.2 y C.3, nos muestran las relaciones entre los atributos y entidades que forman cada dimensión.

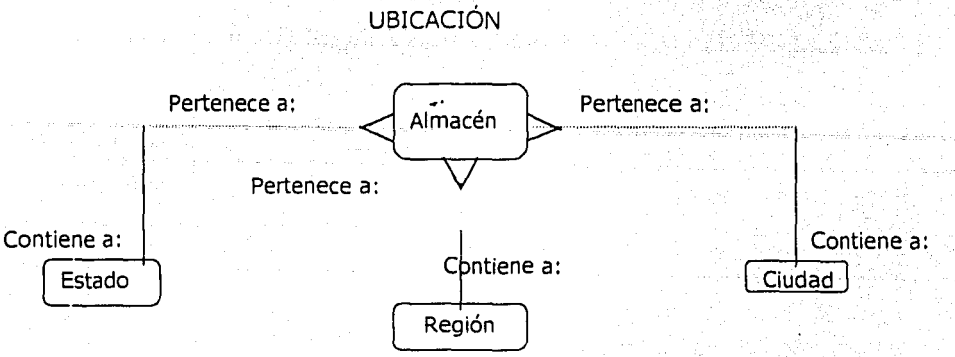



Figura C.2.

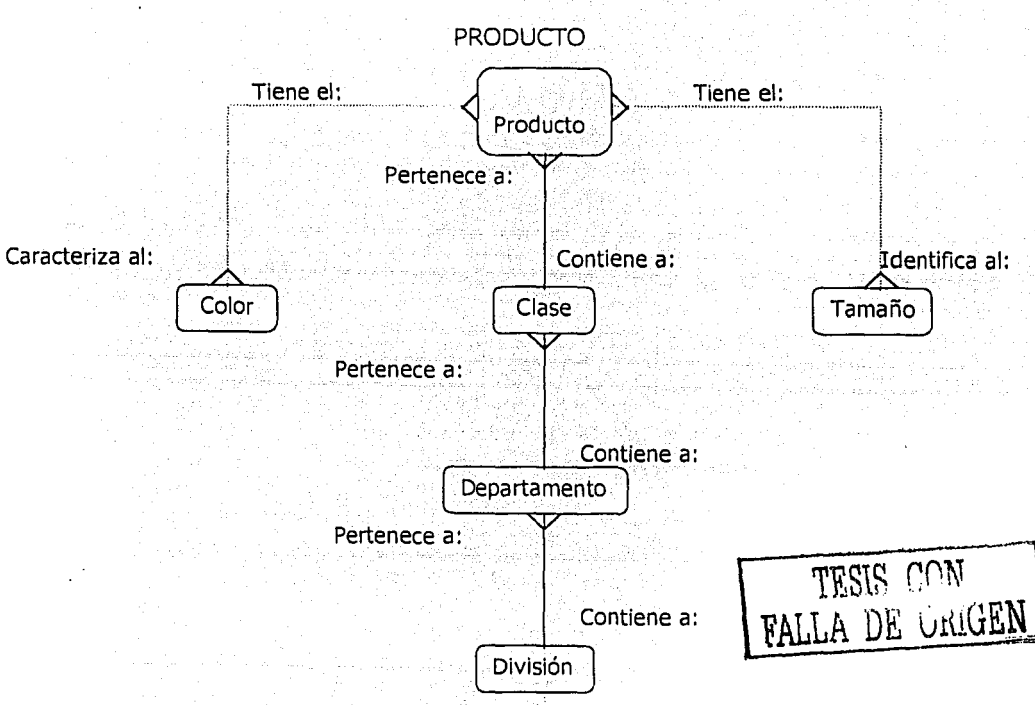



Figura C.3.

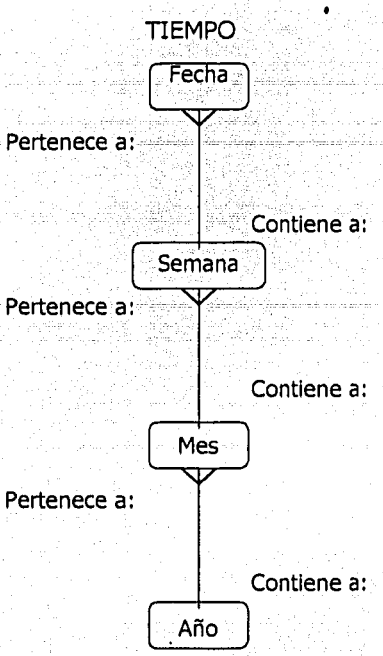



Figura C.4.

Paso 5)

Se establece la jerarquía de cada dimensión, pudiendo ser observadas en la figura C.5.

Paso 6)

Se definen los identificadores únicos pero solamente para las dimensiones. La llave de las tablas proceso es una combinación de las llaves de sus dimensiones. Para el ejemplo solo el atributo departamento es identificado por una combinación entre Id_division y Id_dept. Los demás atributos en las tres dimensiones tienen un identificador, así ciudad es identificado por Id_ciudad y mes por Id_mes.

Luego, se muestra un diagrama con los identificadores propios de los atributos dentro de la jerarquía de cada tabla dimensión, junto con todos los atributos de las tablas proceso.

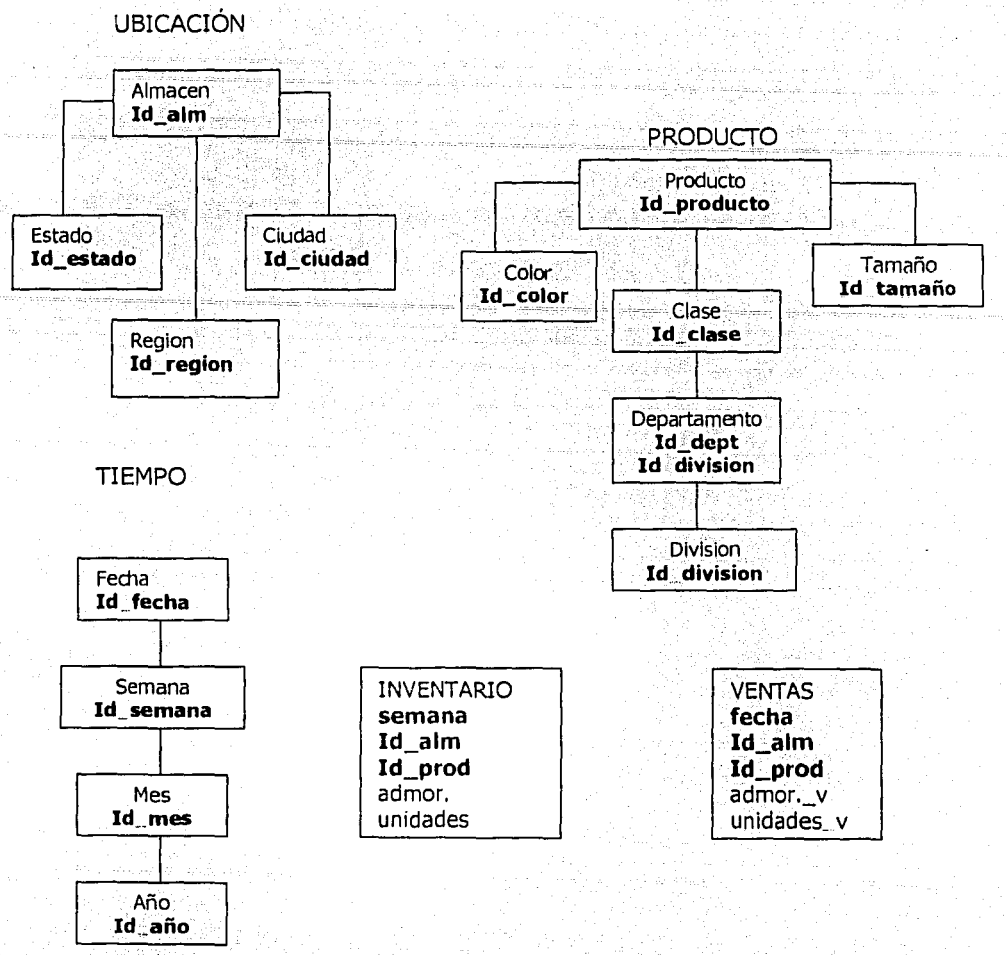



Figura C.5.Jerarquías de las dimensiones para el modelo dimensional de ventas, junto con los identificadores de los atributos de cada dimensión.

Finalmente los modelos dimensional son los siguientes:

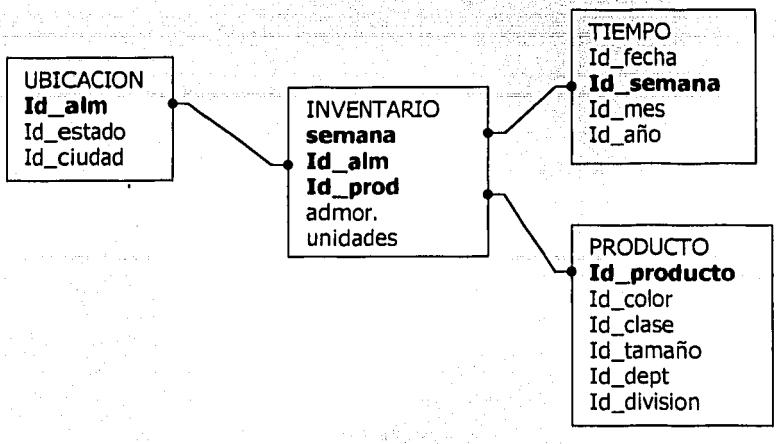



Figura C.6. Esquema estrella para inventario.

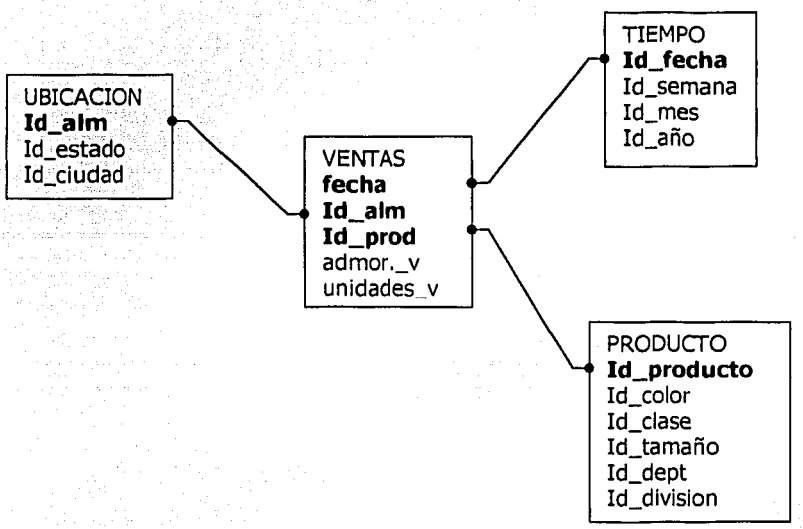



Figura C.7. Esquema estrella para ventas.

Ambos esquemas estrella pudieron quedar en uno solo, sin embargo el nivel de detalle en la dimensión tiempo para inventario es semanal, mientras que para ventas es diario, de ahí que conviene separarlos.

Roles.

Por ejemplo, la entidad estado de la figura C.4, presenta doble rol o función, pues se refiere tanto a la ubicación de un proveedor como de un almacén.

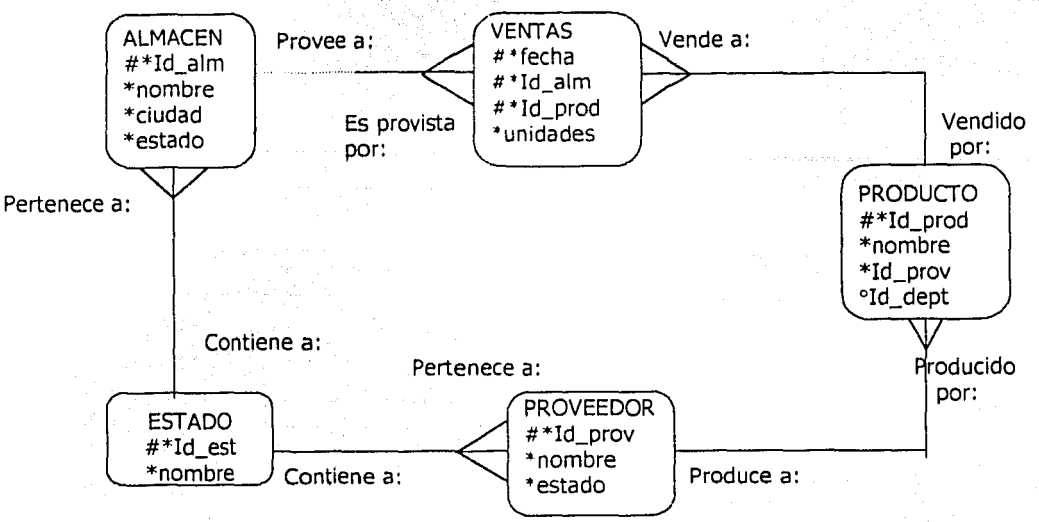



Figura C.8. Modelo de transacciones para ventas.

Dentro de una base de datos relacional, la entidad producto corresponde a la tabla prod, la entidad proveedor a la tabla prov y la entidad almacén a la tabla almac. Las tablas de las otras entidades no alteran el nombre de éstas últimas.

Si se quiere obtener información sobre proveedores y almacenes, se realizaría una consulta SQL como la siguiente:

```
Select a1.estado, sum(ventas.unidades)
From estado a1, almac, ventas, prod, prov, estado a2
        Where a1.est_id = almac.estado and
                almac.alm_id = ventas.alm_id and
                prod.prod_id = ventas.prod_id and
                prov.prov_id = prod.prov_id and
                prov.estado = a2.est_id
                and a2.nombre = "Chihuahua";
```



En un modelo dimensional cada rol implica un atributo con nombre distinto, dentro de cada dimensión afectada. Dentro del depósito de datos se crean vistas de la tabla original cada una con identificadores de nombre distinto.

De este modo para el ejemplo citado en la figura C.4, se crean dos dimensiones ubicación y producto. Cada cual con su atributo estado cuya información proviene de una

tabla única pero almacenada en atributos distintos dentro de un modelo dimensional. Tal y como lo muestra la figura C.9.

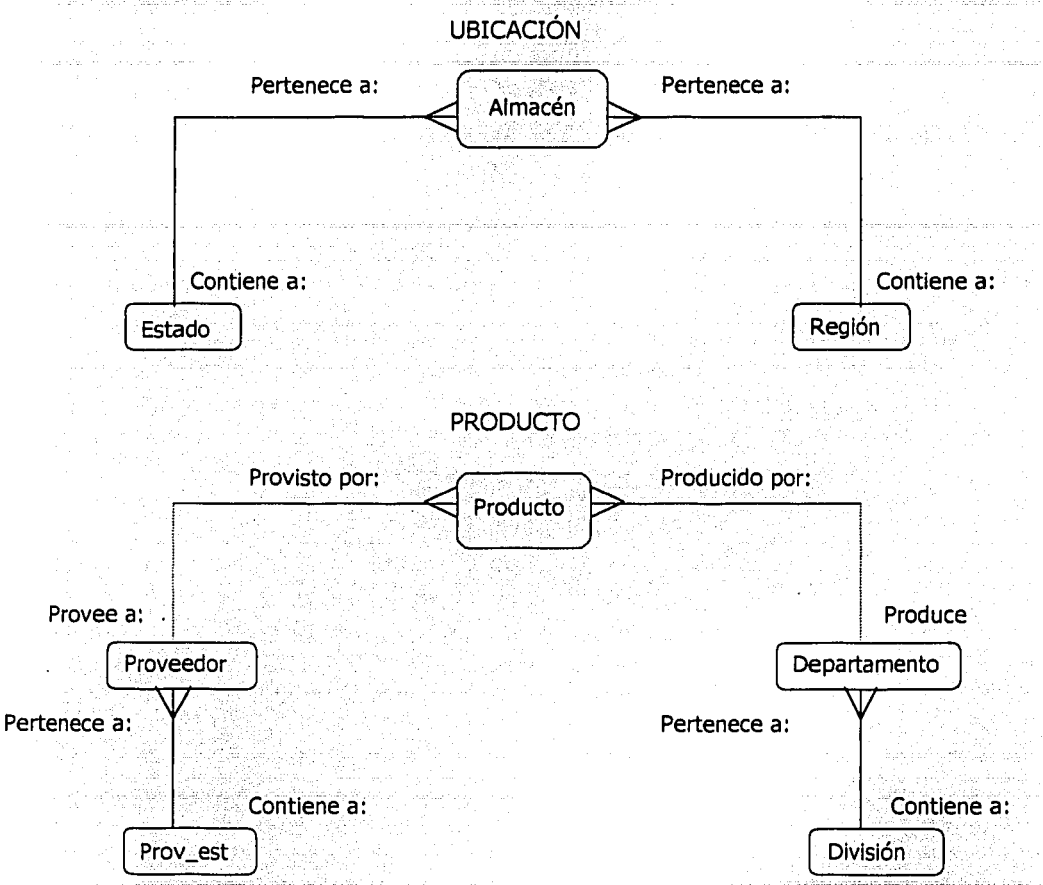



Figura C.9. Relaciones entre los atributos de las dimensiones ubicación y producto.

Nótese que la entidad proveedor forma parte de la dimensión producto, la dimensión tiempo corresponde a la diseñada en las figuras C.6 y C.7

El modelo planteado para la figura C.8 corresponde a la figura C.10, donde los atributos de estado se encuentran subrayados dentro de las tablas dimensión producto y ubicación.

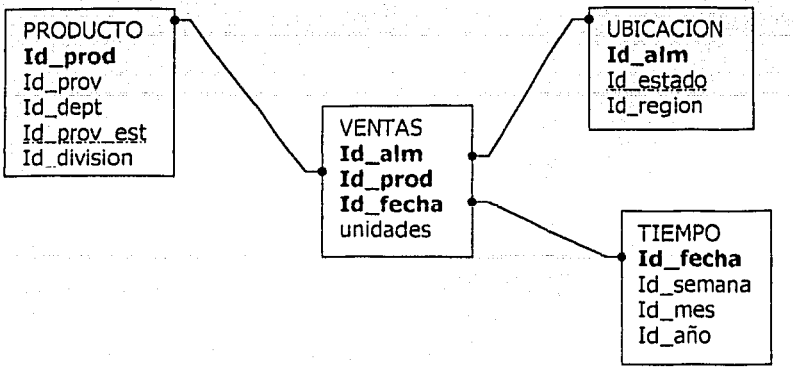



Figura C.10. Esquema estrella para ventas con dos atributos estado.

Para subtipos pueden construirse tablas núcleo y de caracterización cuando los tipos son analizados de forma particular o quedar incluidos en un modelo dimensional. La dimensión asociada puede estar normalizada o desnormalizada según convenga a la ejecución de una consulta. [Libro # 1]

# Glosario.

**API (Application Program Interface).** Conjunto de rutinas, protocolos y herramientas para construir aplicaciones de software.

**Atributo de dimensión.** Información adicional incluida con una dimensión, que no es usada en la definición de los niveles de la dimensión. Las dimensiones llegan a ser más útiles cuando existen muchos atributos que pueden ser utilizados para analizar los datos.

**Autentificación.** Proceso verificador de la clave de acceso o *password* para un sistema de encriptación.

**Clave candidata.** Sea K un conjunto de atributos de una relación, K es una clave candidata si y solo, posee las dos propiedades siguientes:

Unicidad: Jamás, ningún valor de R contiene dos registros distintos con el mismo valor de K.

Irreductibilidad: Ningún subconjunto propio de K tiene la propiedad de unicidad.

**Consistencia.** Obtener los mismos datos desde distintas terminales de un Sistema de Información en un momento dado.

**CORBA (Common Object Request Broker Architecture).** *Middleware* utilizado por grandes empresas, integra diversas plataformas (mainframe y minicomputadoras). Su aplicación más socorrida es el comercio electrónico.

**Cubo.** Conocido también como cubo multidimensional. Es la estructura fundamental para datos en MOLAP, contiene dimensiones, jerarquías, niveles y medidas. Cada punto individual en un cubo es referido como una celda.

**Cubo virtual.** Aquel que está formado a partir de otros cubos, similar a una vista en bases de datos relacionales. Usado con propósitos de seguridad, dando acceso a los usuarios a algunas de las dimensiones y medidas. También puede ser usado para mostrar información de varios cubos a la vez. Son más útiles cuando se comparten las dimensiones.
Dimensión virtual (Virtual Dimension).- Dimensión creada a partir de otras dimensiones.

**Data mining.** Una clase de aplicaciones sobre base de datos, que utilizando procedimientos estadísticos localiza desviaciones o patrones significativos en un grupo de datos. Los datos arrojados permiten predecir la conducta de los mismos y apoyar la toma de decisiones.

**Data Warehousing.** Proceso de visualizar, planear, construir, utilizar, mantener el data warehouse y/o los data marts. Lo cual implica un proceso complejo. Consiste de diseñar un sistema que tendrá como base un data warehouse.

**Dato.** Es cualquier percepción, cifra o texto que por sí solo carece de significado.

**Dependencia funcional.** Sea r una relación y sean X y Y subconjuntos arbitrarios del conjunto de atributos de r. Entonces decimos que Y es dependiente funcionalmente de X. En otras palabras, siempre que dos registros de r coincidan en algún valor X, también lo harán en algún valor Y. Como ejemplo se tiene la dependencia entre Id_articulo y color.

Id_articulo ⟶ color

**Dependencia multivaluada.** Sea R una relación y sean A, B y C subconjuntos de los atributos de R. Decimos que B es multidependiente de A, en símbolos.

A ⟶⟶ B

Si y solamente si en todo valor válido posible de R, el conjunto de valores B que coinciden con un determinado par (valor a, valor C) depende del valor de A y es independiente del valor C.

**DES (Data Encryption Standard).** Método de encriptación simétrica desarrollado en 1975 y estandarizado por ANSI en 1981 como ANSI X.3.92. Si se carece de los requerimientos BXA no se puede utilizar fuera de E.U.A. y Canadá. Usa claves de 56 bits y un mismo código para encriptar y decodificar el mensaje.

**Desnormalización.** Proceso donde los datos de una relación no cumplen estrictamente con una serie de formas normales. Generalmente las relaciones desnormalizadas utilizan la primera forma normal. De acuerdo a la conveniencia de las consultas a la relación le son aplicadas las siguientes formas normales. En una base de datos completamente denormalizada la información será redundate facilitando la ejecución de consultas. La desnormalización se recomienda en el desarrollo de data mart o data warehouse (sistemas que utilizan software OLAP).

**DML (Data Manipulation Language).** Subconjunto de sentencias SQL que permiten modificar, insertar y borrar datos de una base de datos relacional.

**Dominio.** Conjunto de valores que asume un atributo en un modelo entidad-relación.

**DRDA.** Arquitectura propia de IBM que permite la interoperabilidad entre bases de datos relacionales. Utilizada para redes homogéneas, donde el cliente está consciente de los datos que subyacen en el servidor. El modelo de codificación mejora la interacción entre plataformas similares. Si las plataformas son distintas se tienen que implementar formatos de codificación.

**EBCDIC.** *Extended Binary Coded Decimal Interchange Code.* Código de IBM para representar caracteres como números. Aunque es utilizado ampliamente en computadoras IBM grandes, la mayoría de las otras computadoras, inclusive computadoras personales y computadoras Macintosh usan código ASCII.

**Encriptación.** Traducción de datos a código secreto. Los datos encriptados se encuentran en texto cifrado que al ser decodificados se transforman al texto original sea numérico o carácter.

**GUI (Graphical User Interface).** Interfaz de programa que utiliza las capacidades gráficas de las computadoras para facilitar el uso de un programa. La primera GUI fue diseñada por Xerox Corporation a principios de los 70's y dada a conocer por las computadoras Apple Macintosh.

**Hipercubo.** Conocido como cubo multidimensional. Es un cubo con más de tres dimensiones.

**Información.** Es un conjunto de datos significativos, un dato puede ser: carácter, sonido, o imagen, utilizada para trasmitir emociones, ideas o actividades de la cultura humana. El texto y las imágenes son datos no estructurados, aquellos que tienen un formato definido para su utilización, se denominan datos estructurados y un conjunto de ellos, forma una base de datos.

**Integridad.** Característica de los datos de un Sistema de Información, en la cual cumplen con las especificaciones o reglas del negocio que éste necesita de ellos.

**Inteligencia artificial.** Rama de la ciencia de la computación que busca hacer a las computadoras similares a los humanos. Termino acuñado en 1956 por John McCarthy en el MIT. Incluye áreas como:

1. Sistemas expertos. Programación para toma de decisiones sobre situaciones reales.
2. Redes neuronales. Sistemas que simulan inteligencia intentando reproducir los tipos de conexiones físicas que ocurren dentro de los cerebros de seres vivos.
3. Robótica. Programación para reproducir capacidades motrices.
4. Juegos. Programación para simular en base a algoritmos el desarrollo de un juego.
5. Lenguaje natural. Programación para entender lenguajes humanos naturales.

En lenguaje natural y robótica todavía las interfaces están limitadas al control humano. Los sistemas expertos usados en medicina e ingeniería son costosos y solo auxilian en casos limitados. El área de desarrollo más moderna son las redes neuronales que están siendo aplicadas hacia reconocimiento de voz y procesamiento de lenguaje natural.

**IP (Internet Protocol).** Especifica el formato de los paquetes de datos enviados entre redes. Permite enviar los paquetes a una dirección y depositarlos en el sistema.

**IPX (Internetwork Packet Exchange).** Protocolo de red utilizado por sistemas operativos Novell Netware. Al igual que **UDP** y **TCP** es un protocolo de datagramas para comunicaciones sin conexión.

**Kerberos.** Sistema de autenticación desarrollado en el Instituto de Tecnología de Massachusetts MIT. Habilita a dos computadoras a intercambiar información privada como si se tratará de una red abierta.

**Knowlegde Worker.** Trabajadores de conocimiento. Personas que toman decisiones en una organización a través de herramientas de negocio inteligente.

**LAN (Local Area Network).** Red de área local o restringida a un sitio geográfico específico.

**Latice (lattice).** Conjunto parcialmente ordenado en base a una relación binaria reflexiva, antisimétrica y transitiva. Donde $x \vee y = y$ es equivalente a $x \wedge y = x$. Delimitado por $x \cup y$ como límite inferior y $x \cap y$ como límite superior.

**Llave foránea.** Columna de referencia a una llave primaria de otra tabla.

**Llave primaria.** Clave candidata elegida por cada tabla en una base de datos relacional.

**Mainframe.** Sistemas de cómputo basados en consolas receptoras de información procesada desde una computadora central de gran capacidad.

**MDBMS (Multidimensional Database Management System).** Un sistema de gestión de datos que organiza datos multidimensionalmente.

**Metadato.**- Es la información acerca del data warehouse y los datos que contienen, de modo que el administrador del data warehouse pueda dar mantenimiento al mismo, incluye descripción de las tablas y campos origen, con un mapeo de campos origen hacia el data warehose, así como: formato, conversiones monetarias y formulas.

Un metadato es un dato acerca del dato y se utilizan tres tipos de ellos.

Metadatos del negocio. Se refieren a la información acerca del negocio para atributos de datos específicos, utilizados antes y después de las consultas.

Metadatos de la base de datos. Corresponden al diccionario de datos y describen los objetos que ésta contiene.

Metadatos de aplicación. Explican términos acerca de la ejecución de una aplicación y van dirigidos a los usuarios finales.

**Middleware.** Es un software para comunicar sistemas de forma transparente, ocultando las complejidades de conectividad entre ellos; su objetivo es evitar que los desarrolladores escriban código varias veces para aplicaciones usadas por una red de computadoras. Ejemplos de ellos son las herramientas ODBC (Open Database Conectivity) y OLE-DB (Open Linking and Embedding for Database) de Microsoft y JDBC (Java Database Conection).

**Modelado de base de datos.** Consiste en diseñar el modelo conceptual y físico de una base de datos partiendo del análisis de sistemas o bien, ingeniería de software.

**Normalización.** Proceso que organiza datos de una relación conforme a preceptos denominados formas normales. Se conocen cinco formas normales utilizándose con frecuencia las tres primeras (veáse la sección de normalización en el capítulo II). En una base de datos normalizada los datos se almacenan una sola vez, evitando redundancia e inconsistencia. La normalización se recomienda en el desarrollo de sistemas OLTP.

**OLAP (On line Analythical Processing).** Procesamiento de información para sistemas de información gerencial. Dichos sistemas son alimentados por sistemas de transacción en línea,

siendo contenedores de datos históricos. A partir de estos se realizan análisis que auxilian la toma de decisiones.

***MOLAP*** *(Multidimensional On line Analythical Processing).OLAP que trabaja sobre bases de datos multidimensionales.*

***ROLAP*** *(Relational On Line Analythical Processing). OLAP que trabaja sobre bases de datos relacionales.*

**OLE (Object Linking and Embedding).** Un estándar de documento compuesto desarrollado por Microsoft. Permite crear objetos en una aplicación para vincularlos o incrustarlos en una segunda aplicación. Los objetos incrustados conservan su formato original vinculándolos a la aplicación que los creo originalmente.

**OLTP (OnLine Transaction Processing).** Procesamiento de información para sistemas de transacción en línea. Los datos utilizados describen transacciones (compras, ventas, etc.) y su vigencia está determinada por la conveniencia del negocio para preservarlos. Estos se encuentran normalizados y enfocados en la visión de los usuarios involucrados directamente con las transacciones.

**Outer Join.** Operación de junta entre tablas donde alguna de las columnas utilizadas posee valores nulos. En el caso de consultar el nombre del presidente de una compañía, este carecerá de jefe. La columna Id_jefe debe contener un valor nulo. La sentencia asociada de outer join asociada es:

```
Select ap_pat "NOMBRE", e.ap_pat "JEFE"
From emp, emp e
Where Id_jefe = e.Id_emp (+);
```

**PL/SQL.** Lenguaje de programación que permite crear procedimientos con sentencias de **SQL**.

**Red.** Conjunto de computadoras y dispositivos electrónicos conectados para compartir recursos entre usuarios. Posee tres características:

Topología. Arreglo de interconexión (Bus, Anillo o híbridos).
Protocolo. Reglas y codificación específica para enviar datos.
Medio transmisor. Material que transmite la información entre los miembros de la red como UTP (Unshielded Twisted Pair, pares de alambres de cobre aislados dentro de un tubo de plástico), coaxial y fibra óptica.

**Reembolso.** Un reembolso consiste de regresar parte o todo el capital invertido en la adquisición de un producto o servicio.

**PIB (Producto Interno Bruto)**. Se utiliza en contabilidad nacional y representa de forma global, el resultado final de la actividad productiva, o el valor de los bienes y servicios finales generados por una economía en su territorio.

No es fácil de calcularse, sin embargo existen tres fórmulas que son las más usuales:

a) PIB = Σ Va + IVA de los productos + impuestos netos ligados a la importación

Va: Valor añadido por sector económico.

b) PIB = Σ emp + exportaciones – importaciones
emp: Empleos finales interiores de bienes y servicios (formación bruta de capital).

c) PIB = Σ rem + excedente bruto de explotación + impuestos sobre producción + importaciones – subvenciones.

rem: Remuneración de los asalariados.

**RAID (Redundant Arrays of Inexpensive Disks).** Es un arreglo de pequeños discos que simulan ser un solo disco. El concepto surgió en la Universidad de California Berkeley en 1987. Existen cinco tipos de arquitectura RAID.

RAID 0. Nivel no redundante, los datos están repartidos en un arreglo. La ejecución es buena mientras uno de los discos no falle en la recuperación de datos.

RAID 1. Produce redundancia conteniendo los datos para dos o más discos. La ejecución es más rápida en lectura y más baja en escritura comparada con un solo disco. Si alguno de los discos falla los datos no se pierden. El sistema resulta costoso puesto que un disco es utilizado para almacenar datos o duplicarlos.

RAID 2. Utiliza código de corrección de errores Hamming, se usa para discos que no tienen detección de errores.

RAID 3. Desmantela los datos en bytes para varios discos.

RAID 4. Es un sistema de discos externos, todo el sistema está conectado vía controladores SCSI. Simulando en la computadora anfitriona un solo disco. De acuerdo con el número de canales las operaciones de lectura y escritura son más eficientes.

**RDBMS (Relational Database Management System).** Un sistema de gestión de bases de datos basado en la teoría relacional (Oracle, Sybase y Microsoft SQL Server) apoyados con SQL.

**RSA.** Una tecnología de encriptación pública basada en algoritmo del mismo nombre para claves de gran tamaño. RSA es el acrónimo de los apellidos de sus inventores Rivest, Shamir y Adelman.

**Servicio.** Es el trabajo remunerado realizado para otra persona.

**Sistema de legado**. Sistema Operacional o Transaccional. Algunas veces las organizaciones tienen varios sistemas que han sido desarrollados en tiempo diferentes y por desarrollares distintos para una variedad de propósitos. Los datos en dichos sistemas generalmente son incompatibles. Uno de los retos más grandes es reunir toda la información en un depósito de datos único.

**SNA (Systems Network Architecture).** Un conjunto de protocolos de red desarrollados por IBM. Diseñado en 1974 para sistemas mainframe de IBM, ha evolucionado hacia arquitectura cliente servidor.

**SNAPI (Structured N-dimensional Application Program Interface).** Un programa que trabaja sobre una arquitectura cliente-servidor de varios niveles, encargado de apoyar el desarrollo de aplicaciones gráficas de software **OLAP**.

**SQL (Structured Query Language).** Lenguaje estructurado de consulta para bases de datos relacionales.

**Superclave.** Un conjunto de atributos que contienen una clave candidata.

**TCP (Transmission Control Protocol).** Permite a dos computadoras cliente no necesariamente en la misma red, establecer una conexión e intercambiar flujos de datos. Garantiza la entrega de datos y también que los paquetes serán liberados en la misma forma en que fueron enviados.

**Trigger.** Objeto de una base de datos relacional codificado en **PL/SQL**, que permite vigilar la violación de la integridad de algún atributo dentro de una tabla, cuando se presenta una operación de inserción, modificación o borrado de datos.

**UDP (User Datagram Protocol).** Similar a **TCP**, solo que provee pocos servicios de recuperación ante errores en transferencia de archivos por **IP**.

**UNIX.** Sistema Operativo multiusuario y multitarea programado en lenguaje C hacia principios de los años 70's por los laboratorios Bell. Multiusuario significa que varias computadoras cliente estén conectadas a un servidor para compartir software y dispositivos periféricos. Es multitarea porque varias aplicaciones pueden ser ejecutadas simultáneamente en un cliente.

De UNIX se desprenden dos familias de sistemas operativos System V desarrollada por AT&T y BSD4.x iniciada en Berkeley University.

# Referencias.

## Referencia Bibliográfica.


1. **Advanced DSS Functionality & Architecture.**
   Versión 5.1, Microstrategy, 1998.

2. **Data mining: Concepts & techniques**
   Jiawei Han & Micheline Kamber
   Sn. Diego, California, USA, Academic Press, 2001.

3. **Data warehouse Design Solutions**
   Christopher Adamson, Michael Venerable
   John Wiley & Sons, USA, 1998.

4. **Data warehouse from architecture to implementation.**
   Barry Devlin
   Reading, Massachusetts, Addison-Wesley, 1997.

5. **Data warehousing. Concepts, technologies, implementions and management.**
   Harry S. Singh.
   Prentice Hall, Upper Saddle River, NJ, USA, 1998.

6. **Data warehousing. The how-to Guide for implementing your own data warehousing**
   Harjinder S. Gill & Prakash C. Rao
   Indianapolis; Que Corporation, 1996.

7. **E-data Turning Data into Information with Data Warehousing**
   Addison Wesley Information Technology Series
   Jill Dyché
   Adisson Wesley, March 2000.

8. **Fundamentos de base de datos. Tercera Edición.**
   Abraham Silberschatz, Henry F. Korth, S. Sudarshan.
   Madrid, España; Mc Graw-Hill, 1998.

9. **Fundamentals of DSS Agent & Architecture.**
   Versión 5.1, Microstrategy, Education, 1998.

10. **Introducción a los sistemas de bases de datos.**
    C. J. Date
    México, Pearson Educación (Prentice Hall), 7ª. Ed., 2001.

11. **Introducción a Oracle. Parte 1: Diseño relacional de base de datos.**
    Oracle de México
    México, Oracle, 1994.

**12. La calidad en el servicio**
Carlos Dunga Dávila
México, Panorama editorial, 1ª. Ed., 1995.

**13. Oracle Data warehousing. Guía sobre Oracle data warehousing**
Michael J. Corey, Michael Abbey
Madrid, España, Osborne/Mc Graw Hill, 1997.

**14. Oracle 8 Data warehousing.**
Gary Dode, Tim Gorman
John Wiley & Sons, 1998.

**15. Sistemas de bases de datos. Conceptos fundamentales**
Rammaez Elmasrii, Shamkat B. Navathe
Wilmington, Delaware; Addison Wesley iberoamericana, 1997.

**16. SQL Server 7 Data warehousing**
Michael Corey, Michael Abbey, Ian Abramson, Larry Barnes, Benjamín Taub & Rajan Venkitachalam; Berkeley, California; Osborne McGraw Hill, 1999.

**17. The data warehouse toolkit: practical techniques for building dimensional Data warehouse.**
Ralph Kimball
New York, John Wiley & Sons, 1996.

## Referencia de Artículos de Revista.

**1.-** Sánchez Paredes José Antonio. Una receta para construir un repositorio analítico de Información (data warehouse). Soluciones Avanzadas, Junio 1996, año 4, número 34.

**2.-** Bernardo Miramón Commons. Data warehousing estrategias generales de implantación. Soluciones Avanzadas, Junio 1996, año 4, número 34.

**3.-** Ulises Castillo, *Middleware*: No salga a hacer Data warehousing sin él. Soluciones Avanzadas, Junio 1996, año 4, número 34.

**4.-** Bruce Jenks. Data warehouse de Niveles (Tired Data Warehouse). Soluciones Avanzadas, Junio 1996, año 4, número 34.

**5.-** Adolfo Guzmán Arenas. Uso y diseño de mineros de datos. Soluciones Avanzadas, Junio 1996, año 4, número 34.

**6.-** Federico Hernández Alvarez. Sistema Gerencial Administrativo para el soporte de decisiones. Soluciones Avanzadas, Junio de 1997, año 5, número 46.

## Referencia de Páginas Web.

**Páginas de instituciones educativas:**

**1.** http://misdb.bpa.arizona.edu/~mis696g/Reports/olap/sld001.htm
OLAP/MOLAP/ROLAP
Kevin Rasmussen Yesim Tabanoglu

**2.** http://scis.acast.nova.edu/~lookjoe/CISD_794P.html
What's OLAP?
última modificación Enero 09, 2000.

**3.** http://engpub1.bu.edu/BE500DB/02.03.98/tsld029.htm
Biological Database Analysis.

**Páginas de revistas:**

**4.** http://www.dbmsmag.com/9804d141.html
OLAP's Place in the Warehouse Architecture
By Steven B. Elkins,
DBMS, April 1998.

**Otras páginas:**

**5.** http://www.ibm.com

## Referencia de Software.

**1.** Oracle8i Data warehousing Guide. Oracle8i Documentation 2001.