MANUAL PARA EL DISEÑO E IMPLEMENTACIÓN DE BASES DE DATOS OLAP Y SU APLICACIÓN EN INTELIGENCIA DE NEGOCIOS

DIEGO ALEJANDRO CALLE SANCHEZ

UNIVERSIDAD EAFIT
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA DE SISTEMAS
MEDELLÍN
2009

MANUAL PARA EL DISEÑO E IMPLEMENTACIÓN DE BASES DE DATOS OLAP Y SU APLICACIÓN EN INTELIGENCIA DE NEGOCIOS

DIEGO ALEJANDRO CALLE SANCHEZ

Trabajo de grado para optar por el titulo
De Ingeniero de sistemas

ASESOR:
Ingeniero Juan Guillermo Lalinde Pulido
Docente de tiempo completo
Departamento de Ingeniería de Sistemas

UNIVERSIDAD EAFIT
ESCUELA DE INGENIERÍA
DEPARTAMENTO DE INGENIERÍA DE SISTEMAS
MEDELLÍN
2009

Nota de aceptación:

_____
_____
_____
_____
_____
_____
_____

Presidente del jurado

Jurado

Jurado

Medellín, Mayo 08 de 2009

AGRADECIMIENTOS

En el desarrollo de este proyecto de grado quisiera agradecer a todas las personas que de uno u otro modo me guiaron para la realización del mismo. La realización de este proyecto no hubiese sido posible sin la ayuda de estos.

Al Ingeniero Juan Guillermo Lalinde, asesor del proyecto, le agradezco la disposición y confianza mostradas en todo momento, así como las recomendaciones en la preparación de este proyecto.

Igualmente agradezco a mi familia por haberme apoyado y ayudado en todo lo posible durante los años de mi carrera.

TABLA DE CONTENIDO

LISTADO DE TABLAS

LISTADO DE FIGURAS

LISTADO DE ANEXOS

INTRODUCCION

En el mundo organizacional actual en donde se hace necesario tomar decisiones estratégicas para el día a día de las empresas de un modo rápido y eficiente, los sistemas de información y las aplicaciones de Bussines Intelligence “BI” en particular, juegan un papel muy importante en la generación de información basada en datos existentes que permita contar con el conocimiento necesario para la toma adecuada de decisiones basadas en un profundo análisis de los datos relevantes del negocio.

En Colombia, actualmente las empresas no tienen por lo general una idea de lo importantes que pueden ser las aplicaciones BI para el proceso de toma de decisiones por lo que se hace necesario dar a conocer estas herramientas, así como sus potencialidades y su eficiencia para el análisis de datos.

Para que una empresa sea competitiva en el mundo actual, las personas que tienen a su cargo la toma de decisiones deben ser capaces de acceder rápida y fácilmente a la información de la empresa y esto se logra con las aplicaciones BI.

Debido a lo anteriormente expuesto, se hace necesario contar con un manual completo, pero a la vez fácil de entender que explique cómo es el proceso de diseño y creación de una aplicación BI que sea de utilidad para estudiantes interesados en el tema.

La importancia de contar con un manual de referencia sobre cubos se acrecienta por el hecho de que es importante que los estudiantes tengan a disposición

información clara sobre el tema de cubos y su aplicación en inteligencia de negocios.

Para suplir esta necesidad, se propuso este documento el cual contiene una metodología para la elaboración de cubos y aplicaciones de BI hecho de un modo sencillo y fácil de entender. .

En el capitulo uno se explicara con mayor detalle cual es el propósito de las aplicaciones BI, en el segundo capítulo se tratará el tema de las diferencias entre una base de datos OLAP y una base de datos OLTP, y se explicará cómo interactúan estas para lograr crear una aplicación BI.

Por último se presentara un manual de cómo es el procedimiento para crear un cubo en SQL Server haciéndolo paso a paso de una forma clara y explícita, explicando cada cosa que se haga a fin de que el lector del presente documento comprenda las razones de los procedimientos a seguir.

Este documento se ha estructurado con el fin de ofrecer al lector un compendio de orden académico y empresarial en el campo de las aplicaciones BI de modo que sea claro tanto la utilidad de las mismas, como la forma de planear crear y ejecutar una aplicación de BI dando las herramientas al lector para que cree un cubo completamente funcional el cual sirva al propósito para el cual fue diseñado.

1. PROPÓSITO DE LAS APLICACIONES DE BI Y SUS USOS.

1.1 CONCEPTOS GENERALES.

Para entender de un modo global cual es la funcionalidad de una aplicación de BI, primero debemos tener en claro que es una base de datos.
Un sistema gestor de bases de datos o DBMS consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a esos datos.

El principal objetivo que se pretende con un DBMS es que este tenga un entorno conveniente para el manejo de la información y sea a la vez eficiente para su utilización en los procesos de extracción y almacenamiento de datos.

Un sistema de base de datos está diseñado para gestionar grandes bloques de información lo que implica que para gestionar estos datos se debe contar con unas estructuras de almacenamiento de información que sirvan como aprovisionadores de los datos requeridos

Es necesario además mencionar que un sistema de bases de datos debe de mantener la seguridad de la información que en el se almacena, dado que estos pueden ser de importancia estratégica y por lo mismo se debe garantizar que no vaya a haber accesos no autorizados a los mismos o datos anómalos, además de garantizar que no haya redundancia o inconsistencia de datos lo que significa que estos deben ser consistentes en todo el modelo.
Un sistema de bases de datos debe garantizar además el acceso fácil a los datos y debe manejar la integridad del modelo, lo que significa que los valores

almacenados en la base de datos deben satisfacer ciertas restricciones de consistencia tales como tipos de datos, valores mínimos y máximos permitidos en un determinado campo etc.
(F.Kort, Slverschatz).

Existen dos tipos de bases de datos las cuales si bien son diferentes en su propósito y en su diseño, se complementan dentro de un modelo de BI. Estas son: bases de datos OLAP y bases de datos OLTP

1.1.1 Bases de datos OLTP:

Una base de datos OLTP “online transaction procesing, es lo que se conoce comúnmente como una base de datos transaccional. Esto significa que estas bases de datos están orientadas al procesamiento de transacciones. Una transacción siempre genera un proceso atómico el cual debe ser validado por un commit o invalidado por medio de un rollback, y el cual puede involucrar operaciones de inserción, modificación y borrado de datos.
Este proceso transaccional es el proceso típico de las bases de datos operacionales.

Las características de una base de datos transaccional son:

- El acceso a los datos se optimiza para tareas de lectura y escritura.
- Existe un estructuramiento de los datos según el nivel de aplicación, esto significa que los datos se estructuran dependiendo si el sistema es un CRM, o un ERP, o un sistema de información departamental etc.
- En muchas ocasiones no hay uniformidad de los datos pudiendo existir falta de compatibilidad entre estos.
- El historial de datos suele limitarse a datos actuales o recientes.

1.1.3 Bases de datos OLAP:

Las bases de datos OLAP “online analitical procesing”, son bases de datos orientadas al proceso analítico, lo que significa que estas suelen utilizarse para hacer análisis sobre grandes cantidades de información. Estos análisis suelen implicar de modo general la lectura de grandes cantidades de datos con el objetivo de extraer algún tipo de información útil de estos. Estas bases de datos son los sistemas típicos de los datamarts.

Las características principales de una base de datos OLAP son:

El acceso a los datos suele ser de solo lectura, siendo la acción más común sobre la misma el realizar consultas sobre los datos habiendo muy pocas inserciones, actualizaciones y eliminaciones sobre los mismos.

- Los datos se estructuran según las áreas del negocio, además los datos están integrados de manera uniforme en toda la organización.
- El historial de los datos es a largo plazo; normalmente de 2 a 5 años.
- Para cargar las bases de datos OLAP con información, se utilizan los sistemas operacionales existentes, utilizando para esto un proceso de extracción, transformación y carga (ETL).

(Arquitectura del data warehouse. Consultado en Noviembre 2008
Disponible en *sinnexus.com/business_intelligence/olap_vs_oltp.aspx*.)

Tabla 1 Diferencias entre las bases de datos OLAP y OLTP

	Sistema OLTP	Sistema OLAP
Fuente de los datos	Datos operacionales, las bases de datos OLTP son las fuentes de los datos	Los datos de una base de datos OLAP provienen de diversas fuentes de datos OLTP
Propósito de los datos	Control de tareas básicas del negocio	Ayuda a la planeación, solución de problemas y soporte de decisiones
Que revelan los datos	Visualización de los procesos actuales del negocio	Vistas multidimensionales de varios tipos de actividades del negocio
Inserciones y actualizaciones	Las inserciones y actualizaciones se realizan de forma corta y rápida siendo estas realizadas por usuarios finales	Se utilizan Jobs periódicos para actualizar los datos
Queries	Relativamente estandarizados los cuales además retornan relativamente pocos datos	Usualmente complejos debido a que envuelven diversas agregaciones
Velocidad de procesamiento	Muy rápido	Esta depende de la cantidad de datos envueltos, siendo las actualizaciones de los datos y los queries complejos tener velocidades de procesamiento de horas
Requerimientos	Puede ser relativamente	Grande, debido a la existencia de

de espacio	pequeño	estructuras de agregación e historiales de datos
Diseño de la base de datos	Altamente normalizado con muchas tablas	Desnormalizada con pocas tablas utilizándose un esquema de estrella
Backup y recuperación de datos	Realización de backups continúa, debido a que los datos son críticos para el negocio	Se recargan los datos de la fuente de datos OLTP como método de recuperación de datos

Fuente tabla: (Differences between OLTP and OLAP. Consultado en Noviembre 2008 Disponible en *it.toolbox.com/wiki/index.php/differences_between_OLTP_and_OLAP*)

Tabla 2 Conceptos básicos de las bases de datos OLAP y OLTP

Característica	OLTP	OLAP
Tamaño BDD	Gigabytes	Gigas a Terabites
Origen de datos	Interno	Interno y externo
Actualización	On-line	Batch
Periodos	Actual	Histórico
Consultas	Predecibles	Ad Hoc
Actividad	Operacional	Analítica

Fuente tabla: (Todo el Business Inteligence. Consultado en Noviembre, 2008. Disponible en ***todo****bi.blogspot.com)*

1.1.4 La aplicación de las herramientas de BI como estrategia competitiva en las empresas:

Si bien en las empresas colombianas no existe aún una conciencia de la importancia de las aplicaciones de BI en los procesos de toma de decisiones, se hace necesario trabajar en la difusión de las aplicaciones de BI como herramienta de gran utilidad para la generación de información que ayude a soportar el proceso de toma de decisiones.

Podemos definir Business Intelligence como el conjunto de herramientas y estrategias enfocadas a la administración y creación de conocimiento mediante el análisis de los datos existentes en una organización o empresa.
Estas herramientas deben de contar con las siguientes características:

- Accesibilidad a la información: se debe garantizar que los usuarios tendrán acceso a los datos con total independencia de la procedencia de estos.
- Apoyo en la toma de decisiones: las herramientas de BI deben servir para algo más que la presentación de la información. Los usuarios deben tener acceso a las herramientas de análisis que les permitan seleccionar y manipular los datos que les sean de interés en determinado momento.
- Orientación al usuario final: las herramientas de BI deben ser de fácil manipulación, de modo que un usuario sin muchos conocimientos técnicos sea capaz de utilizar la herramienta.

De acuerdo al nivel de complejidad de las herramientas de BI estas pueden clasificarse en:

- Consultas en informes simples tales como queries y reportes.
- Cubos OLAP.
- Minería de datos, en el cual las empresas recaban información sobre diversos tópicos importantes para la empresa. Las aplicaciones de minería

de datos se utilizan para identificar tendencias y comportamientos para extraer información que pueda identificar y descubrir relaciones en las bases de datos que revelen comportamientos poco evidentes y que sirvan para intuir cambios o nuevas tendencias.

Las herramientas de BI a la que se refiere este estudio serán únicamente los cubos OLAP por ser el objetivo de este proyecto.

El objetivo de los cubos OLAP es que las personas que tienen a cargo el proceso de toma de decisiones dentro de las empresas tengan a mano una herramienta que les permita acceder rápida y fácilmente a la información de la empresa que sea pertinente para el proceso de toma de decisiones.

Las áreas en las que más comúnmente se utilizan los cubos OLAP son las de ventas, marketing, finanzas y producción.

La herramienta de cubos OLAP lo que permite además de lo anteriormente descrito es minimizar el riesgo que toda toma de decisiones implica, ya que podemos utilizar los datos corporativos y transformarlos en información útil que nos sirva para este propósito.

En resumen lo que buscan las empresas con la utilización de herramientas BI es aumentar su competitividad.

Las herramientas BI permiten esto ya que se centran en destacar los datos relevantes para un adecuado análisis. De este modo se pueden dedicar recursos a los frentes importantes con el objetivo de hacer la empresa más competitiva, y BI logra ser parte importante de esto ya que permite hacer un análisis cuantitativo y cualitativo de los datos visualizando los resultados en informes que permitirán contar con la información relevante para una adecuada toma de decisiones.

1.1.5 Razones por las que las empresas deben utilizar las aplicaciones BI. Las principales razones por las que las empresas deben utilizar BI son:

- Conseguir y mantener una correcta adecuación de las normativas contables, fiscales y legales exigidas, cosa que se logra mediante una adecuada actualización de los datos y una herramienta que nos permita navegar eficazmente a través de ellos.
- Obtener el verdadero valor de los sistemas ERP. Las aplicaciones BI pueden ayudar a sacar un mayor provecho de estos sistemas ya que permiten desarrollar todo el potencial de información que estos pueden contener.
- Crear, manejar y monitorear las métricas fundamentales de la empresa.
- Mejorar la competitividad de la empresa por medio de un adecuado análisis de los datos de la misma.

1.1.6 Aplicaciones de las herramientas BI en las empresas. Las herramientas BI tienen un gran potencial para ser aplicado en las diferentes áreas de la empresa. La forma en que BI puede ser aplicada en las diferentes áreas de la empresa son:

- Departamento de ventas y marketing: Las herramientas BI ayudan a facilitar la comprensión de las necesidades de los clientes y de este modo responder a las necesidades del mercado. BI ayuda en esto por medio del desarrollo de análisis de marketing capaces de medir el impacto de cosas tales como las promociones, precios, análisis de patrones de compra etc.
- Desarrollo de productos: BI facilita el acceso a los datos de los clientes y el mercado lo que permite analizar las relaciones entre coste y beneficio de las características de un determinado producto.
- Operaciones: BI provee las herramientas que permiten analizar el rendimiento de cualquier tipo de proceso operativo ya que comprende desde la planificación de producción pasando por el control de calidad y la administración de inventarios.

- Departamento financiero: BI permite acceder a los datos de forma inmediata y en tiempo real, mejorando de este modo sus operaciones, incluyendo presupuestos, proyecciones, control de gestión y tesorería.
- Atención al cliente: BI permite evaluar los segmentos del mercado y de los clientes individuales, además permite retener a los clientes más rentables.

En resumen BI permite evaluar diferentes variables que permitan a la empresa reconocer nuevas oportunidades, anticiparse a posibles cambios o problemas y hacer los ajustes necesarios antes que estos afecten el mundo real. Además, la capacidad de tomar las decisiones acertadas en el tiempo preciso se ha vuelto una clave para el éxito empresarial.
(Todo el Business Inteligence. Consultado en Noviembre, 2008. Disponible en ***todo****bi.blogspot.com)*

2. RELACIÓN ENTRE OLTP Y OLAP

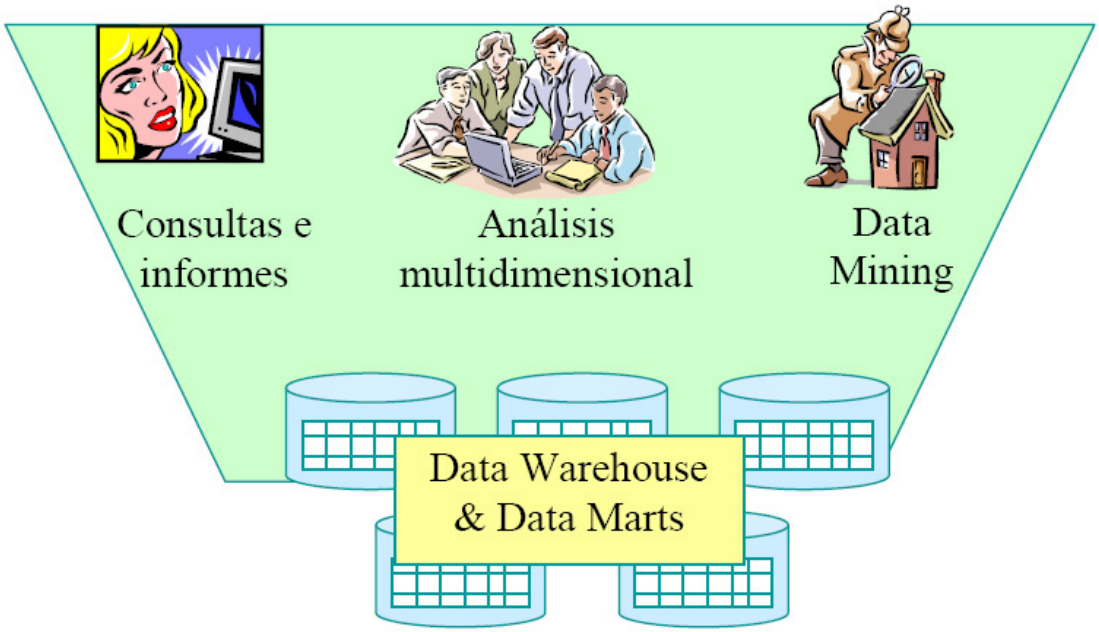
Para comprender como funcionan estos dos tipos de bases de datos y como interactúan dentro de una aplicación BI, primero se debe entender como es la arquitectura de un DataWarehouse.

Debido a que las organizaciones manejan grandes cantidades de datos, los cuales bien pueden residir en distintas bases de datos o pueden estar ubicados en diferentes gestores de bases de datos, tiene como consecuencia dificultades al acceder y utilizar estos datos en aplicaciones de análisis las cuales requieren extraer, preparar e integrar los datos. Para solucionar este problema se creo el Data warehousing el cual es un diseño de procesos e implementación de herramientas que proporcionen información completa, oportuna, correcta y entendible en la toma de decisiones.

Las técnicas utilizadas en el data warehousing son.

- Acceso a fuentes de datos heterogéneas: limpieza de datos, filtrado de datos, transformación de datos.
- Almacenamiento de datos: estructura de datos multidimensional.

Figure 1 Estructura de datos multidimensional



Fuente figura: (Fernando Berzal Data warehousing)

2.1. EL MODELO MULTIDIMENSIONAL.

Los datos de un data Warehouse se modelan en cubos de datos sobre los cuales se realizan las siguientes operaciones las cuales permiten una visión multidimensional de los datos:

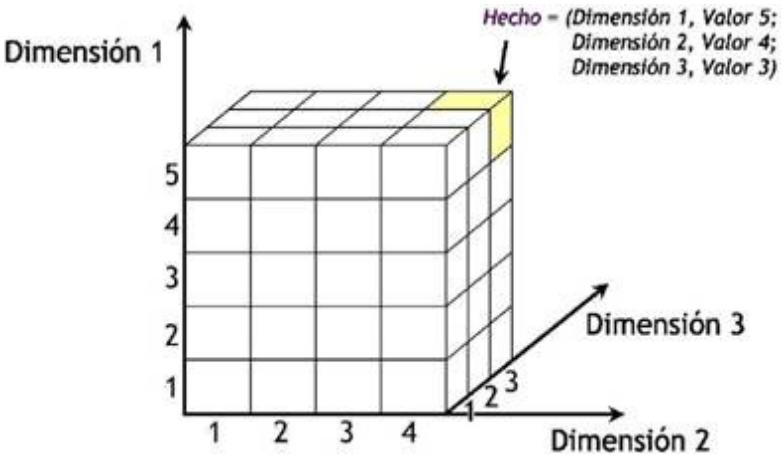
- Roll up: incremento en el nivel de agregación de datos. Este se utiliza cuando no se quiere tener tanto detalle en el nivel de una consulta; el roll up agrupa los datos de una consulta por medio de un criterio de selección de consulta dado.
- Drill down: incremento en el nivel de detalle; lo cual es opuesto al roll up. En este modo se da un mayor nivel de granularidad a una consulta dada utilizando como parámetro un criterio de selección dado.
- Drill across: en esta operación lo que se hace es agregar un nuevo criterio de análisis o dimensión a una consulta.
- Roll across: el roll across es lo opuesto al drill across; con esta operación se elimina un criterio o dimensión a una consulta.
- Pivotaje o rotación: reorientación de la visión multidimensional de los datos. Esto permite reorganizar la forma en que los datos son presentados con el objetivo de evaluar de diferentes formas los mismos.

El objetivo del modelado multidimensional es visualizar un modelo de datos como un conjunto de medidas descritas por dimensiones.
Este modelo es bueno para resumir y organizar datos, trabajar sobre datos de tipo numérico, además es más simple y fácil de visualizar y entender que el modelado E/R.

Las bases de datos multidimensionales poseen una estructura la cual facilita el tener un acceso flexible a los datos con el objetivo de poder analizar sus

relaciones. Los datos se visualizan en un cubo multidimensional en donde las variables asociadas existen a lo largo de varios ejes o dimensiones y la intersección de las mismas representa la medida, o indicador o hecho de lo que se está evaluando.

Figura 2 Cubo



Fuente figura: (Fernando Berzal Data warehousing)

En este ejemplo se tiene un cubo de tres dimensiones y el resultado que se obtiene es el valor 5 de la dimensión 1 mas el valor 4 de la dimensión 2 mas el valor 3 de la dimensión 3. El resultado aquí está dado por los cruces matriciales de acuerdo a los cruces de las dimensiones seleccionadas.

(Data warehousing. Consultado en Diciembre, 2008. Disponible en ***elvex.ugr.es/idbis/db***)

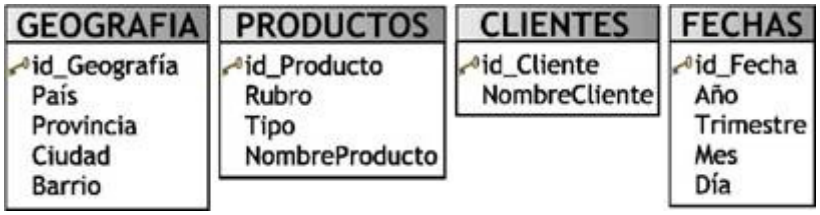
2.1.1 Elementos del modelo multidimensional. El modelo dimensional está compuesto por una BD desnormalizada la cual posee un modelo en estrella en el cual relacionan los hechos con los agentes del negocio o dimensiones.

La razón por la cual la base de datos debe de estar desnormalizada es para evitar el desarrollo de uniones join complejas para acceder a la información, esto con el fin de tener una mayor agilidad a la hora de realizar consultas a la base de datos.

Los elementos de un modelo multidimensional son:

- Dimensiones: estas son las perspectivas o entidades del negocio sobre las cuales una organización desea mantener sus datos organizados; por ejemplo tiempo, localización, clientes, proveedores etc. Estas a su vez se componen de miembros los cuales son nombres o identificadores que marcan una posición dentro de la dimensión. Un ejemplo de esto podrían ser meses, trimestres o años los cuales son miembros de la dimensión tiempo, o ciudades, regiones y países que son miembros de la dimensión localización.

Figura 3 Tabla de dimensiones.



Fuente figura: (Informationmanagement.wolrd press.com).

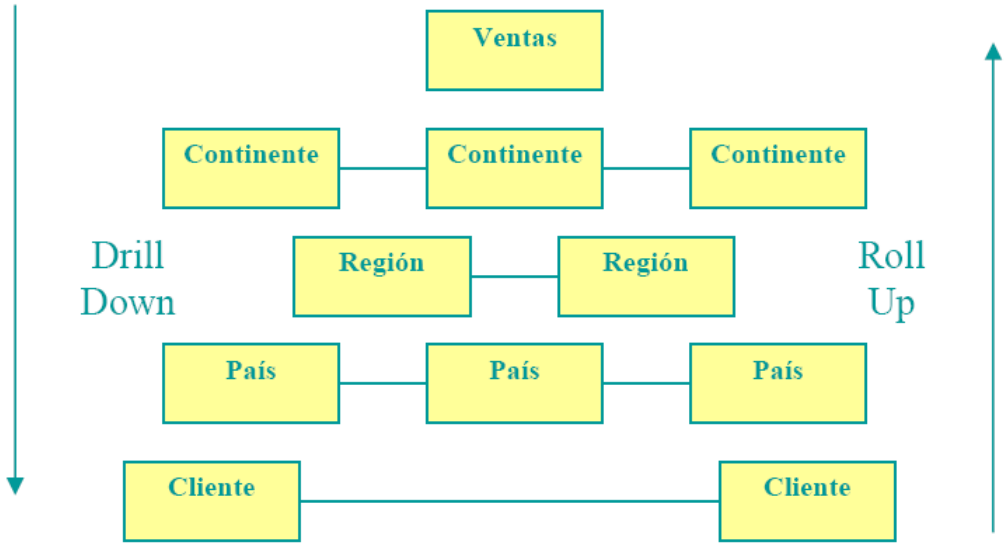
Las tablas de dimensiones le dan el contexto a los datos numéricos.

Los miembros de las dimensiones suelen organizarse en jerarquías, con el propósito de hacer las búsquedas en el cubo más óptimas.

Cada tabla de dimensiones debe poseer un identificador único “clave primaria” y al menos un atributo que describa los criterios de análisis relevantes de la organización.

Existe una dimensión obligatoria la cual es la dimensión de tiempo, siendo la definición de granularidad y jerarquía de la misma dependiente de la dinámica del negocio que se esté modelando. Esta dimensión es muy importante ya que determina el tiempo de ocurrencia y ubicación de las situaciones que se están analizando representando de este modo diferentes versiones de la misma situación.

Figura 4 Organización jerárquica de las dimensiones.



Fuente figura: practical techniques for building dimensional dataware houses

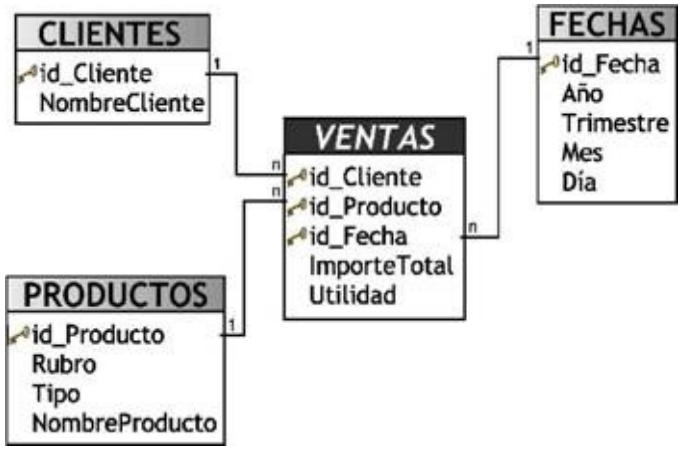
- Hechos: estos son colecciones de datos compuestos por medidas y un contexto, teniendo en cuenta que las dimensiones determinan el contexto de los hechos y que cada hecho en particular está asociado a un miembro de cada dimensión.

  La tabla de hechos almacena hechos relativos a la actividad empresarial como por ejemplo las ventas en pesos y unidades. Esta tabla se compone básicamente de claves foráneas y medidas.
  En lo posible debe tratarse de utilizar solo tipos de datos numéricos y fechas.

La clave primaria de la tabla de hechos consiste en claves foráneas de las tablas de dimensión; además todas las claves foráneas deben ser índices.

- Medidas. Estos son atributos numéricos asociados a los hechos los cuales son lo que realmente se mide. Ejemplos de esto pueden ser volumen de ventas, costo asociado a un producto, número de transacciones efectuadas, porcentaje de beneficios etc.

Figura 5 Tabla de hechos.



Fuente figura: (Informationmanagement.wolrd press.com).

Como se ve en esta figura, la tabla de hechos ventas en este caso está compuesta por las claves primarias de las dimensiones y las medidas para las ventas; en este caso estas se medirán por importe total y utilidad.

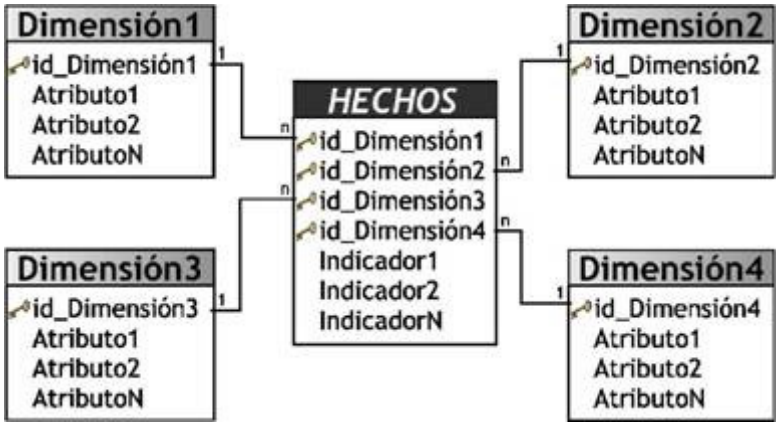
2.1.2 Tipos de modelo multidimensional. Existen tres tipos básicos de esquemas en los cuales se busca por medio de un modelo desnormalizado un mejor tiempo de respuesta en las consultas y una mayor sencillez con respecto a su utilización. La desventaja de estos esquemas es que genera cierto grado de redundancia pero esto se sacrifica en beneficio de una optimización de la respuesta ante las consultas de los usuarios.

Los tipos de modelo son:

Esquema en estrella: este esquema se compone de una tabla de hechos central y de varias tablas de dimensiones que se relacionan con esta a través de sus respectivas claves.

Este esquema es el más simple de interpretar y es el que más optimiza los tiempos de búsqueda de los usuarios, sin embargo es el menos robusto para la carga.

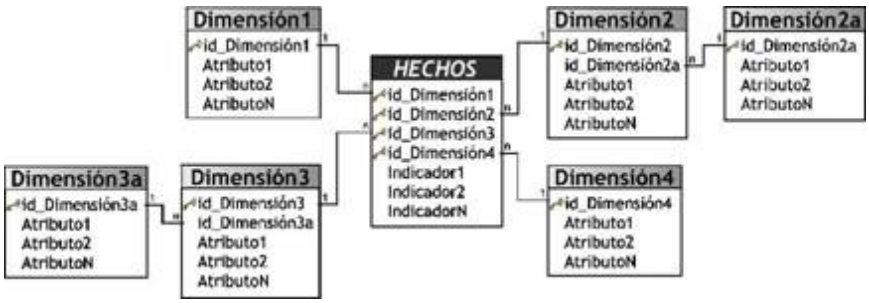
Figura 6 Esquema de estrella.



Fuente figura: (Informationmanagement.wolrd press.com).

Esquema de copo de nieve: este esquema representa una extensión del modelo en estrella cuando las dimensiones se organizan en jerarquías de dimensiones.

Figura 7 Esquema de copo de nieve.

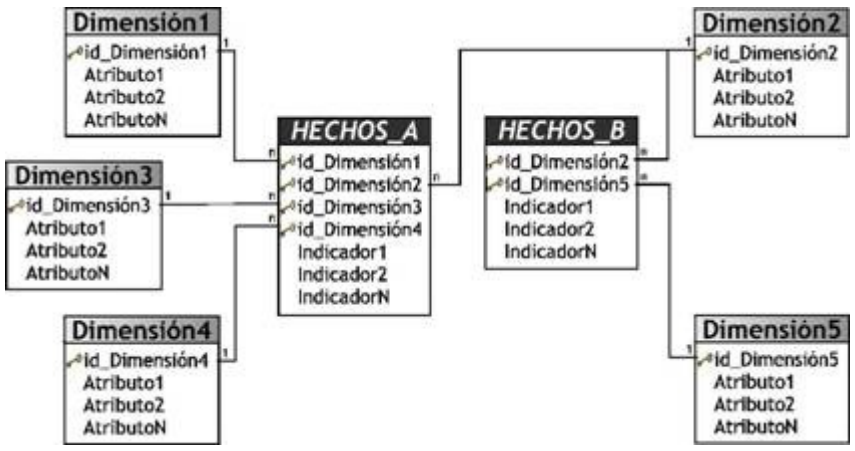


Fuente figura: (Informationmanagement.wolrd press.com).

Este modelo se compone de una tabla de hechos central la cual está relacionada con una o más tablas de dimensiones, las que a su vez pueden estar relacionadas con otras tablas de dimensiones.

Este modelo se utiliza cuando se requiere tener la posibilidad de segregar los datos de las dimensiones para proveer un esquema que sustente los requerimientos de diseño, además posee una mejor utilización del espacio en disco y es muy útil en las tablas de dimensiones de muchas tuplas.
Las desventajas que puede tener este modelo es que se puede crear un número de dimensiones muy grande lo que dificulta su manejo; además el desempeño en las consultas es menor.

Esquema de constelación: Este modelo se compone de una serie de esquemas de estrella. Las Tablas de dimensiones se encuentran en el centro del modelo y se encuentran relacionadas con sus respectivas tablas de dimensiones.
Figura 8 Esquema de constelación.



Fuente figura: (Informationmanagement.wolrd press.com).

La ventaja de este esquema es que permite tener más de una tabla de hechos lo que posibilita analizar un mayor número de aspectos claves del negocio, sin embargo no es soportado por muchas de las herramientas de consulta y análisis.

2.1.3 Arquitectura. Un DataWarehouse está formado por varios componentes que interactúan entre sí.

Los componentes de la arquitectura de un DWH son básicamente los siguientes:

Aplicaciones desde las que se extraen los datos; tipo bases de datos OLTP, si las fuentes de los datos son variadas lo que debe hacerse es una integración de las mismas en una sola base de datos desde la cual se procederá a cargar el cubo.

A pesar de que la fuente más habitual de datos son las bases de datos transaccionales, existen otros tipos de aplicaciones que pueden ser utilizadas como fuente de datos, tales como archivos de texto, hojas de cálculo e informes.

Procedimientos de carga los cuales tienen la función de cargar la información de la base de datos al DWH.

Los ETL sirven a tres propósitos fundamentales los cuales son extraer, transformar y cargar los datos. Básicamente lo que hacen los ETL es extraer los datos desde diversas fuentes, los transforma para resolver posibles inconsistencias entre los mismos y finalmente después de haberlos depurado se procede a su carga en el depósito de datos.

En síntesis las funciones específicas de los ETL son tres:

Extracción: lo que se hace aquí es explorar las diversas fuentes de datos disponibles y se extrae la información relevante de acuerdo a las necesidades del usuario.

Dependiendo de la fuente de datos, los procesos de extracción varían. Si los datos se extraen de una de un SGBD relacional, los procesos de extracción se pueden realizar mediante consultas SQL o rutinas programadas. Si las fuentes son sistemas propietarios o fuentes externas, se deben realizar cambios de formato para que no haya problemas de incompatibilidad de tipos de datos y además utilizar herramientas de volcado de información.

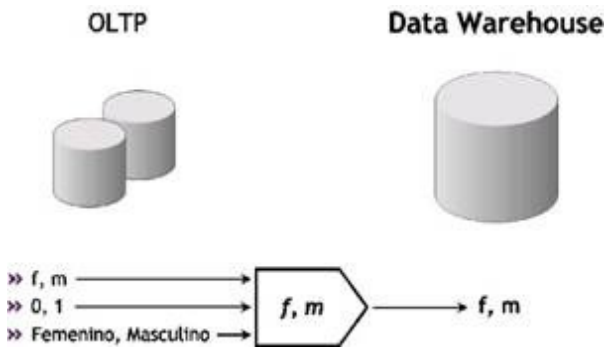
Una vez que se han seleccionado y extraído los datos, estos deben ser guardados en un almacenamiento intermedio con el objetivo de mejorar el rendimiento del DWH ya que no se requiere paralizar los OLTP o los DWH, además se facilita con esta técnica almacenar y gestionar los metadatos que se generan en los procesos ETL y facilita la integración de las diversas fuentes de datos, tanto internas como externas.

Transformación: esta función se encarga de convertir los datos inconsistentes en un conjunto de datos compatibles y congruentes requisito indispensable para utilizar estos en la carga de un DWH. Esto es necesario ya que es vital hacer coincidir formatos, estándares, y formas, de modo que los datos que ingresen al DWH estén integrados.

Las situaciones en las que debe realizarse la transformación de datos son las siguientes:

Codificación: esta inconsistencia surge cuando se encuentran diversas formas para codificar un atributo en común. Un ejemplo de esto es cuando se tienen varias codificaciones para definir el sexo y se codifican todas en un formato en común.

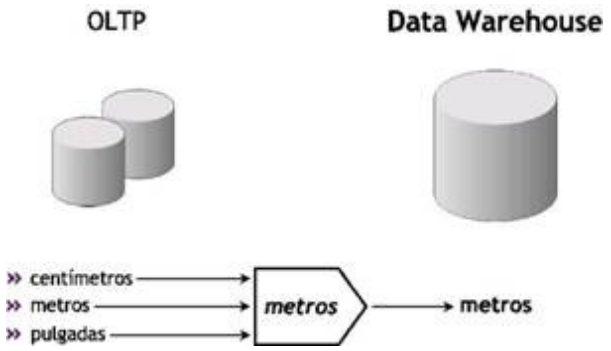
Figura 9 Transformación: Codificación.



Fuente figura: (Informationmanagement.wolrd press.com).

a. Medidas de atributos: Los tipos de unidades de medidas de atributos de una entidad pueden variar entre las diferentes OLTP. Por ejemplo las unidades de medida de cierta entidad pueden estar en cm en una BD y mt en otra.

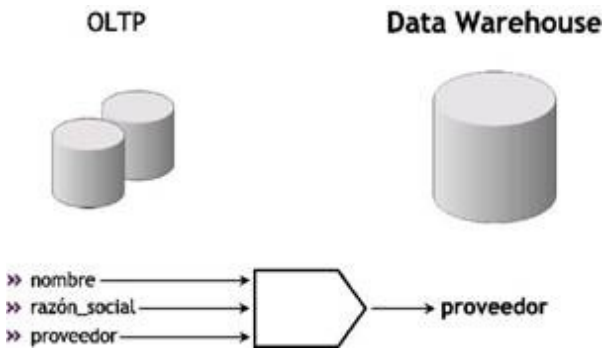
Figura 10 Medidas de atributos.



Fuente figura: (Informationmanagement.wolrd press.com).

b. Convenciones de nombramiento: Un mismo atributo puede ser nombrado de diferentes maneras en las diferentes OLTP; por ejemplo nombre en una BD y razón social en otra.

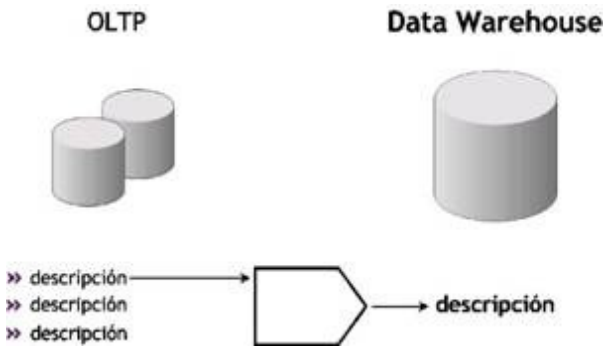
Figura 11 Convenciones de nombramiento.



Fuente figura: (Informationmanagement.wolrd press.com).

Fuentes múltiples: Este caso sucede cuando un mismo elemento puede derivarse de varias fuentes. En este caso se selecciona la fuente más confiable o apropiada.

Figura 12 Fuentes múltiples.



Fuente figura: (Informationmanagement.wolrd press.com).

Limpieza de datos: Su objetivo es el de realizar diferentes tipos de acciones con la finalidad de solucionar problemas con datos erróneos, inconsistentes o irrelevantes.

Las acciones que se realizan sobre estos tipos de datos son: eliminación de la columna que contiene el dato, ignorarlo, filtrado de columna, filtrado de la fila errónea y reemplazo del valor.

Carga: Este proceso se encarga de cargar el DWH con los datos que han sido transformados y que residen en el almacenamiento intermedio o con aquellos datos OLTP que tienen correspondencia directa con el depósito de datos.
Las funciones principales de los ETL son entonces las de cargar el DWH y refrescarlo constantemente con datos recientes los cuales se actualizaran dependiendo de las necesidades del usuario.

En el DWH la información se estructura en cubos multidimensionales los cuales preparan la información para responder a consultas dinámicas que requieran de un buen desempeño.

Las características de un DWH son las siguientes:
Transforma e integra los datos fuente y de almacenamiento intermedio en un modelo adecuado para la toma de decisiones. Realiza la gestión del depósito de datos y lo organiza en torno a una o unas base de datos multidimensionales, las cuales conforman el cubo multidimensional en donde el cruce de los valores de los atributos de cada dimensión a lo largo de las abscisas determinan un hecho específico. Los cálculos aplicados sobre las dimensiones son matriciales, los cuales se procesan dando como resultado reportes tabulares.

Permite realizar todas las funciones de definición y manipulación del depósito de datos con el objetivo de poder soportar los procesos de gestión del mismo.
Es el encargado de ejecutar y dirigir las políticas de particionamiento con el objetivo de conseguir una mayor eficiencia en las consultas al no tener que manejar el grueso de los datos, teniendo en cuenta que estas políticas se ejecutan sobre la tabla de hechos.

Realiza copias de resguardo incrementales o totales de los datos del DWH
Posee un repositorio de datos propio.

Gestiona y mantiene los metadatos los cuales describen o dan información de otros datos existentes en el DWH. Los metadatos dan información sobre localización, estructura y significado de los datos haciendo un mapeo de los mismos.

Las funciones de los metadatos dentro del DWH son:
Facilitar el flujo de trabajo convirtiendo automáticamente los datos de un formato a otro.

Facilitan la búsqueda y descripción de los contenidos del DWH.

Poseen una guía de cómo se transforman e integran los datos fuente operacionales y externos al ambiente del depósito de datos.
Almacenan las referencias sobre los algoritmos utilizados para la esquematización de los datos dentro del DWH.

Contiene las definiciones del sistema de registro desde el cual se construye el DWH.

Los tipos de metadatos son tres; los metadatos de los procesos ETL, los metadatos operacionales que son los que almacenan los contenidos del DWH y los metadatos de consulta los cuales contienen las reglas para el análisis de la información del almacén.

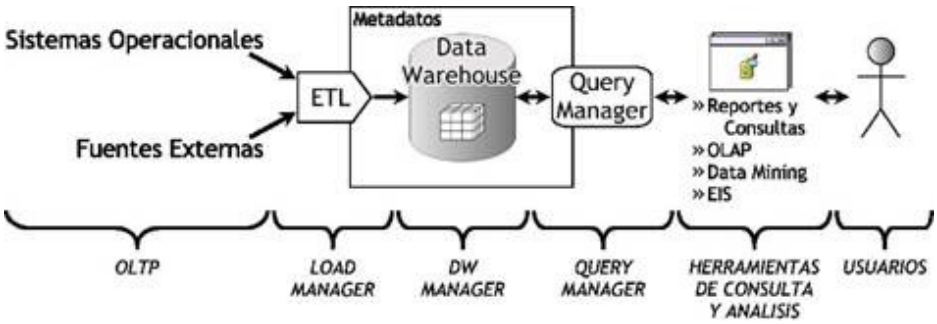
Query Manager. Este componente se encarga de realizar las operaciones necesarias para soportar los procesos de gestión y ejecución de consultas

relacionales, tales como join y agregaciones y de operaciones de consulta para el análisis de datos.

El Query Manager recibe las consultas de los usuarios y las aplica a las tablas correspondientes devolviendo los resultados obtenidos.

Herramientas de consulta y análisis. Para acceder al DWH se pueden utilizar diversas herramientas de consultas, exploración, análisis, reportes etc.
Estas herramientas son sistemas que permiten al usuario realizar la exploración de los datos del DWH.

Por medio de estas herramientas, el usuario genera consultas las cuales son enviadas al DataWarehouse pasando por la herramienta de consulta y análisis y de allí al Query manager el cual realiza la extracción de la información desde el DWH y devuelve los resultados obtenidos a la herramienta de consulta y análisis.
Figura 13 Arquitectura del DWH.



Fuente figura: (Informationmanagement.wolrd press.com).

2.1.4 Conclusiones sobre la relación entre las bases de datos OLTP y OLAP.

Vemos que a diferencia de las bases de datos OLAP, las bases de datos OLTP son utilizadas con el objetivo de soportar la información diaria de las empresas. Esto hace que estas bases de datos se enfoquen en maximizar los procesos transaccionales soportados por los datos.

La estructura de estas bases de datos a diferencia de las OLAP es altamente normalizada con el objetivo de brindar una mayor eficiencia a sistemas que manejen muchas transacciones. Esto significa que estas bases de datos poseen una estructura enfocada a brindar consultas sobre los datos cargados y para la toma de decisiones del día a día; en cambio el modelo OLAP está diseñado para poder llevar a cabo procesos de consultas y análisis para la toma de decisiones estratégicas de alto nivel.

Vemos en conclusión que estos dos modelos de bases de datos se complementan dentro de la arquitectura de un DWH, ya que mientras las bases de datos OLTP se utilizan como fuente de datos, los modelos de bases de datos OLAP se utilizan con el propósito de ser utilizados como herramientas de consulta y análisis.

(The data warehouse toolkit, 1996*)*

3. MANUAL PARA CREACIÓN DEL DWH.

Para la creación del DWH de este manual se utilizará SQL Server 2005.

En este manual se aprenderá como se define la fuente de datos, las vistas de la misma, las dimensiones y la forma de realizar el cubo.

Las herramientas que deben de estar instaladas para poder realizar este proyecto son:

- Motor de bases de datos SQL Server 2005
- SQL Server 2005 Analysis services
- BI development Studio

3.1 DEFINICION DE LA FUENTE DE DATOS.

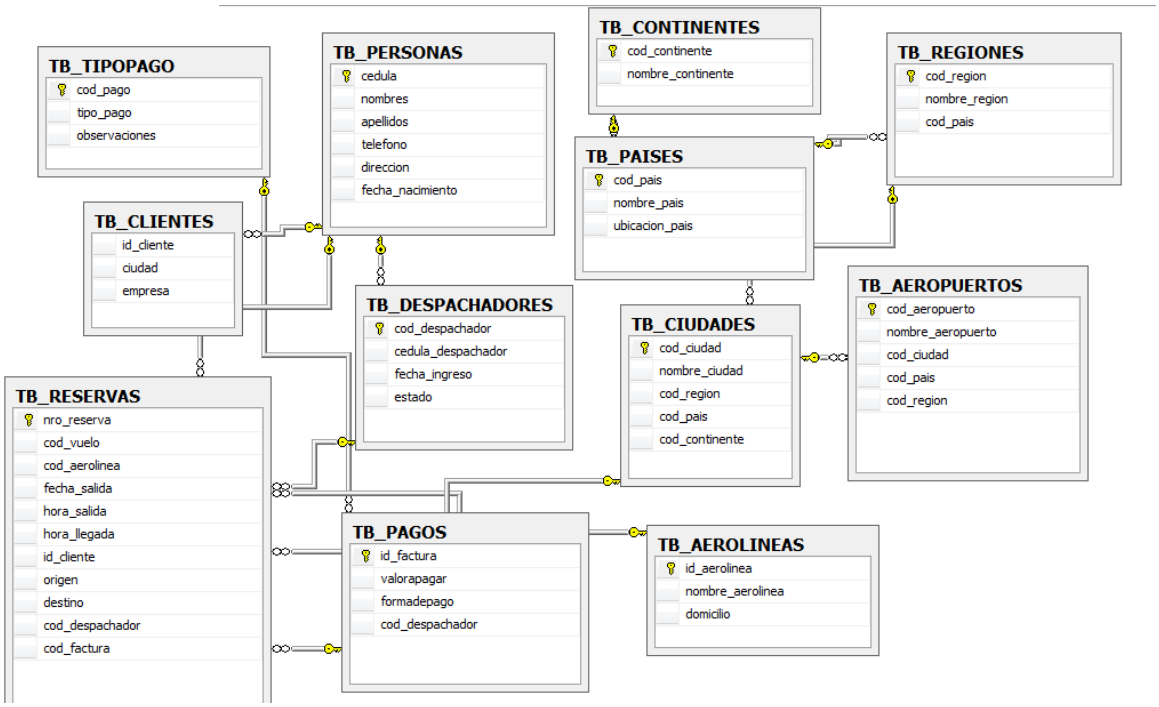
El primer paso para la creación de un cubo es definir de donde se va a extraer la información la cual será de utilidad para modelar el DWH que se piensa crear. Los datos provienen de una o varias bases de datos transaccionales, siendo posible el extraer datos de una o varias tablas y de uno o varios campos de cada una de las tablas.

Dependiendo de la funcionalidad del cubo que queramos crear, se debe seleccionar la información que sirva al objetivo para el cual el cubo va a ser creado.

Para propósitos del ejemplo de este manual, todos los datos se extraerán de una sola base de datos; más adelante se explicará en que varía el proceso si se pretende cargar datos en el cubo provenientes de diferentes bases de datos.

La base de datos creada maneja los datos de las reservas de vuelo de un aeropuerto.

Figure 14 Base de datos transaccional.



Fuente figura: Elaboración propia.

El cubo que se va a diseñar tiene como propósito el servir para estudiar el comportamiento de las reservas de las diferentes aerolíneas que operan en el aeropuerto.

El cubo tiene como objetivo responder a preguntas tales como:

- Cuanto pagan los clientes, sean empresas o personas por viajar diferenciado por ciudades, países o regiones destino.
- Cuáles son las aerolíneas que los clientes más utilizan para viajar.
- Visualizar las formas de pago de los clientes diferenciadas por destinos, clientes y aerolíneas.

Para efectos del cubo, se decidió que la información relevante para el estudio de las reservas proviene de la información relacionada con los clientes, los destinos y las aerolíneas. A continuación se detallarán los campos que contiene cada tabla dimensional del cubo y cuál es el origen de estos en la base de datos transaccional.

Tabla 3 Equivalencias entre tablas del cubo y tablas de la base de datos transaccional –tabla dimensional 1.

Dim_clientes	Tb_personas	Tb_clientes
Id_cliente	Cedula	
Nombre_cliente	Nombre+apellido	
Empresa_cliente		Empresa

Fuente tabla: Elaboración propia.

Tabla 4 Equivalencias entre tablas del cubo y tablas de la base de datos transaccional –tabla dimensional 2.

Dim_aerolineas	Tb_aerolineas
Id_aerolinea	Id_aerolinea
Nombre_aerolinea	Nombre_aerolinea

Fuente tabla: Elaboración propia.

Tabla 5 Equivalencias entre tablas del cubo y tablas de la base de datos transaccional –tabla dimensional 3.

Dim_destinos	Tb_ciudades	Tb_continentes	Tb_paises	Tb_regiones
Cod_destino	Cod_ciudad			
Continente		Nombre_continente		
Pais			Nombre_pais	
Region				Nombre_region
Ciudad	Nombre_ciudad			

Fuente tabla: Elaboración propia.

Después de haber definido las tablas dimensionales del cubo, queda definir la tabla dimensional de tiempo, la cual siempre debe ir en todo modelo de DWH. Los campos que tiene determinada tabla dimensional dependen de las necesidades del negocio. Para este caso se decidió la siguiente división de tiempo.

Tabla 6 Equivalencias entre tablas del cubo y tablas de la base de datos transaccional –tabla dimensional tiempo.

Dim_tiempo
Id_fecha
Fecha
Año
Semestre
Trimestre

Mes
Dia

Fuente tabla: Elaboración propia.

Por último queda definir la tabla de hechos. Esta tabla contiene los id de las tablas dimensionales y las medidas las cuales se utilizarán para la medición que proporcionan los datos del cubo.

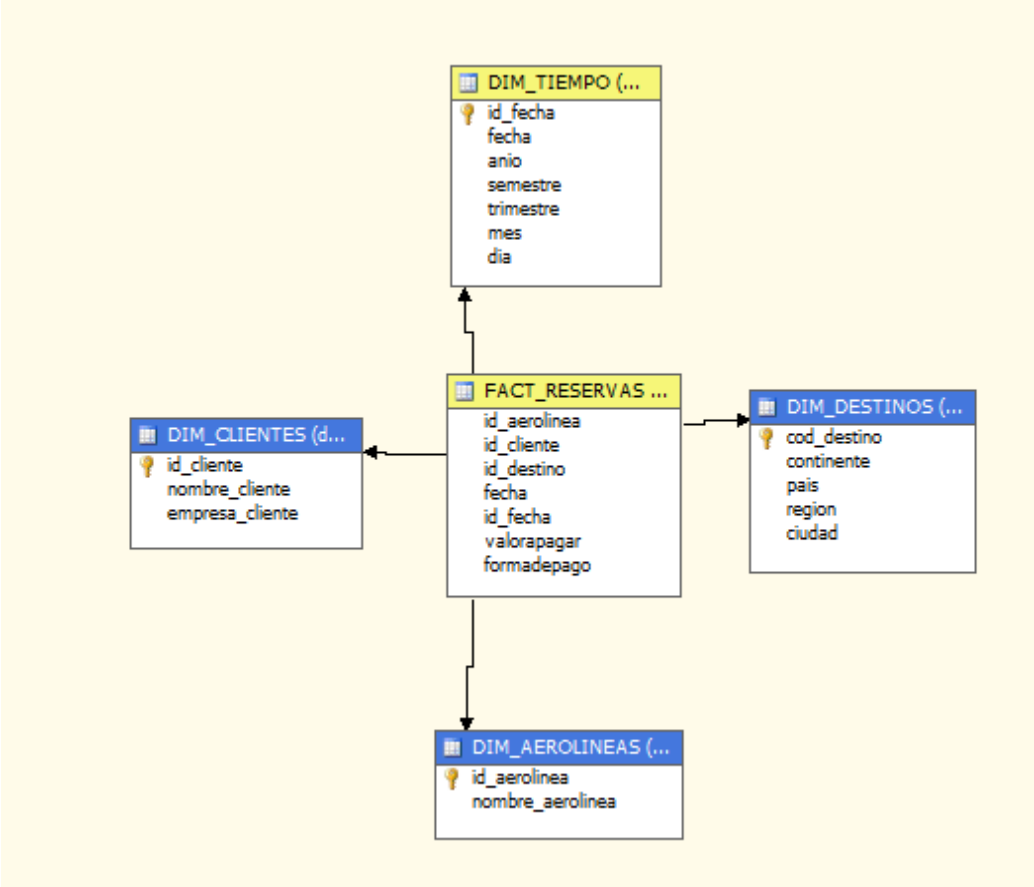
Tabla 7 Equivalencias entre tablas del cubo y tablas de la base de datos transaccional –tabla hechos.

Fact_reservas	Tb_aerolineas	Tb_clientes	Tb_ciudades	Tb_reservas	Tb_pagos
Id_aerolinea	Id_aerolinea				
Id_cliente		Id_cliente			
Id_destino			Cod_ciudad		
Id_fecha				Fecha_salida	
Fecha					
Valor a pagar					Valor a pagar
Forma de pago					Forma de pago

Fuente tabla: Elaboración propia.

El campo fecha el cual es un consecutivo fue creado para que no hubiera problemas de integridad, debido a que en este modelo de cubo, puede haber varias reservas en una misma fecha, por lo que id_fecha no sería una clave apropiada.

Figura 15 Cubo.



Fuente figura: Elaboración propia.

3.2 DEFINICION DE LOS PROCEDIMIENTOS DE CARGA.

Los procedimientos de carga o ETL son una parte muy importante de la arquitectura de un DWH, ya que permiten la extracción de los datos de la base de datos transaccional y su posterior carga dentro del cubo.

La forma de hacer esto es definir un procedimiento de carga por cada tabla del cubo, es decir un procedimiento por cada tabla dimensional y un procedimiento para la tabla de hechos. Hay que tener en cuenta que al igual que en las tablas dimensiónales, los datos de la tabla de hechos, tanto las claves como las medidas son importados desde la base de datos transaccional.

En los anexos del presente trabajo, se explicará en detalle la forma en la que los procedimientos de carga deben ser creados.

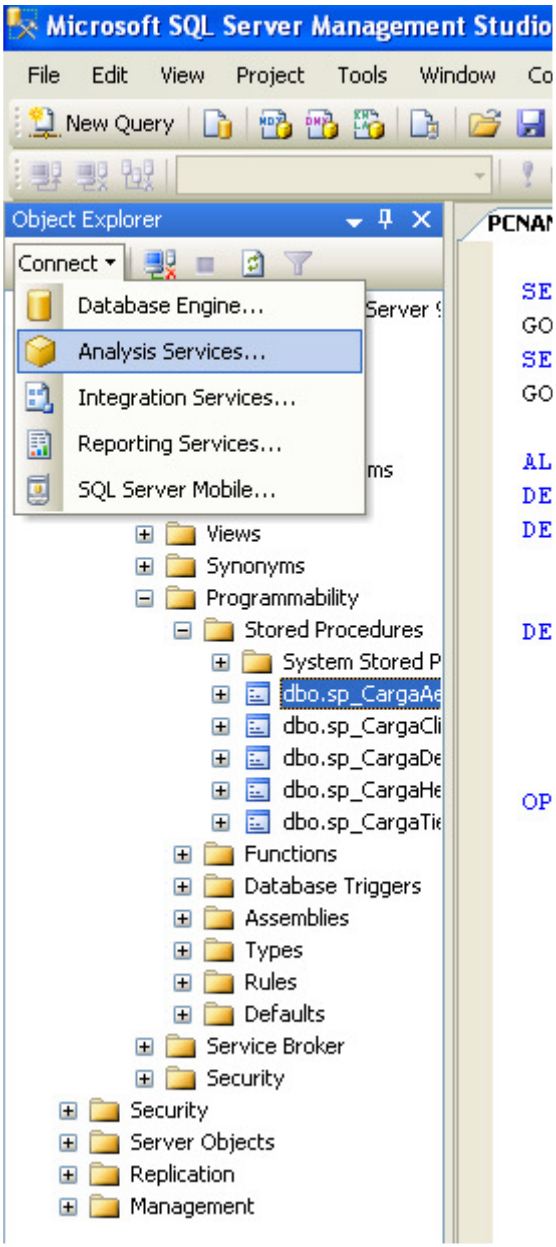
Los procedimientos de carga deben ser creados dentro de la base de datos que contiene el cubo y desde allí deben ser ejecutados cada vez que se vaya a realizar un proceso de carga.

3.3 VISUALIZACION DEL CUBO.

Para el despliegue del cubo se utilizará la herramienta de SQL Server Business Inteligence Development Studio. Esta herramienta viene incluida en la suite de SQL server profesional edition y tiene la ventaja de que permite desplegar el cubo creado en una arquitectura distribuida, simplemente se deben crear las conexiones hacia la maquina que contiene el cubo.

Una vez se ha creado el cubo, se procede a crear una conexión a la herramienta de Analisis Services dentro del motor de bases de datos. Esta herramienta es un paquete para análisis de aplicaciones BI la cual viene como parte del motor de bases de datos SQL server.

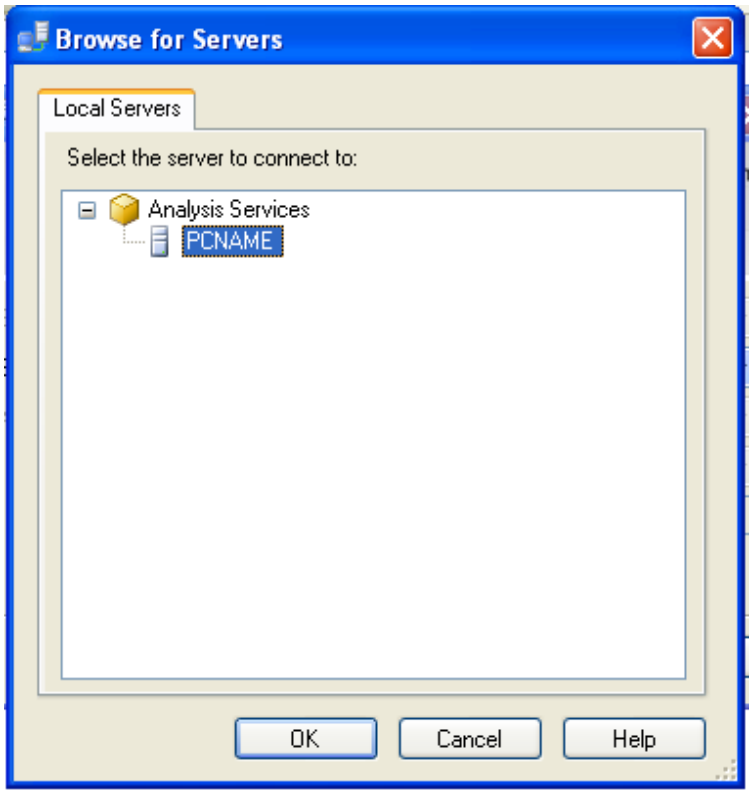
Figura 16 Conexión a analysis services.



Fuente figura: Elaboración propia.

A continuación en nombre de servidor se selecciona la opción buscar otro servidor y por último se despliega la entidad Analysis services y se selecciona el servidor.

Figura 17 Selección del servidor para la conexión.



Fuente figura: Elaboración propia.

Una vez realizado esto se puede proceder a realizar los pasos para el despliegue del cubo.

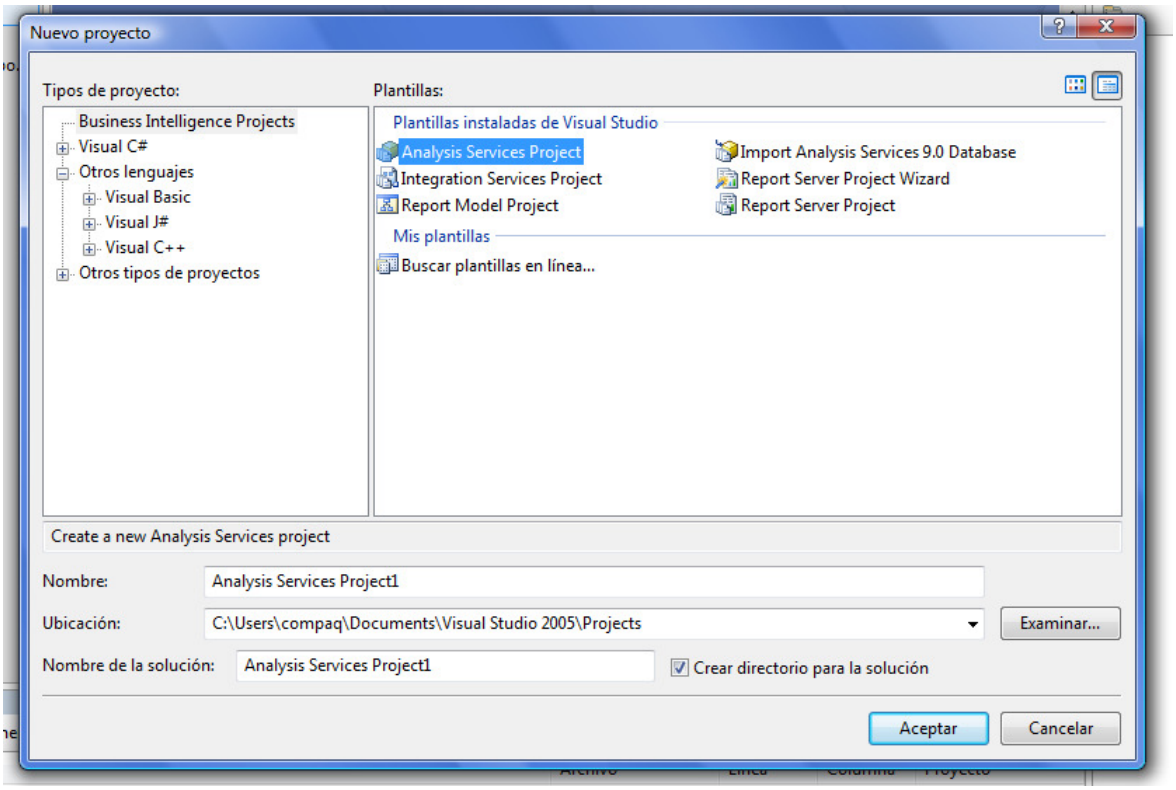
Los pasos a seguir son los siguientes:

- Creación de un proyecto de Analisis Services.

  En todos los programas dentro de la carpeta de Microsoft SQL Server 2005 se selecciona la opción SQL Server Business intelligence development Studio, luego se selecciona la opción archivo en el menú del visual Studio y

se selecciona la opción nuevo -> proyecto. En la caja de dialogo tipos de proyecto que aparece a continuación se selecciona la opción Business Intelligence Projects -> Analysis Services Project y a continuación se procede a dar nombre al proyecto.

Figura 18 Creación del proyecto.



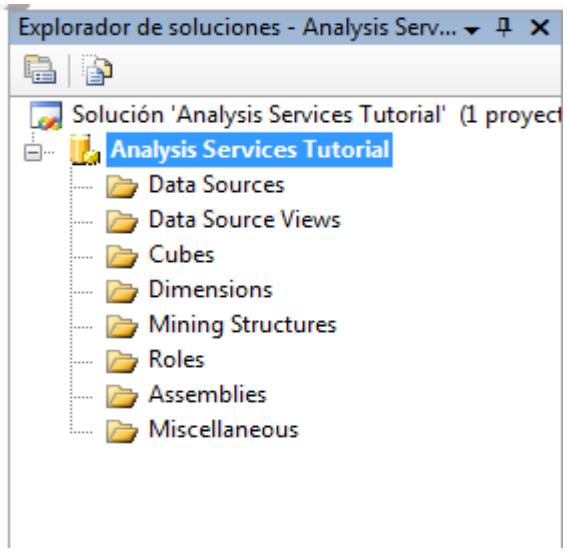
Fuente figura: Elaboración propia.

- Definición de una fuente de datos.

  En esta parte lo que se hace es definir una cadena de conexión la cual será utilizada para conectarse a la fuente de datos.

  En el explorador de soluciones a la derecha del SQL Server Business intelligence development Studio se da click derecho sobre la opción Data Sources -> New Data Source.

Figura 19 Explorador de soluciones.

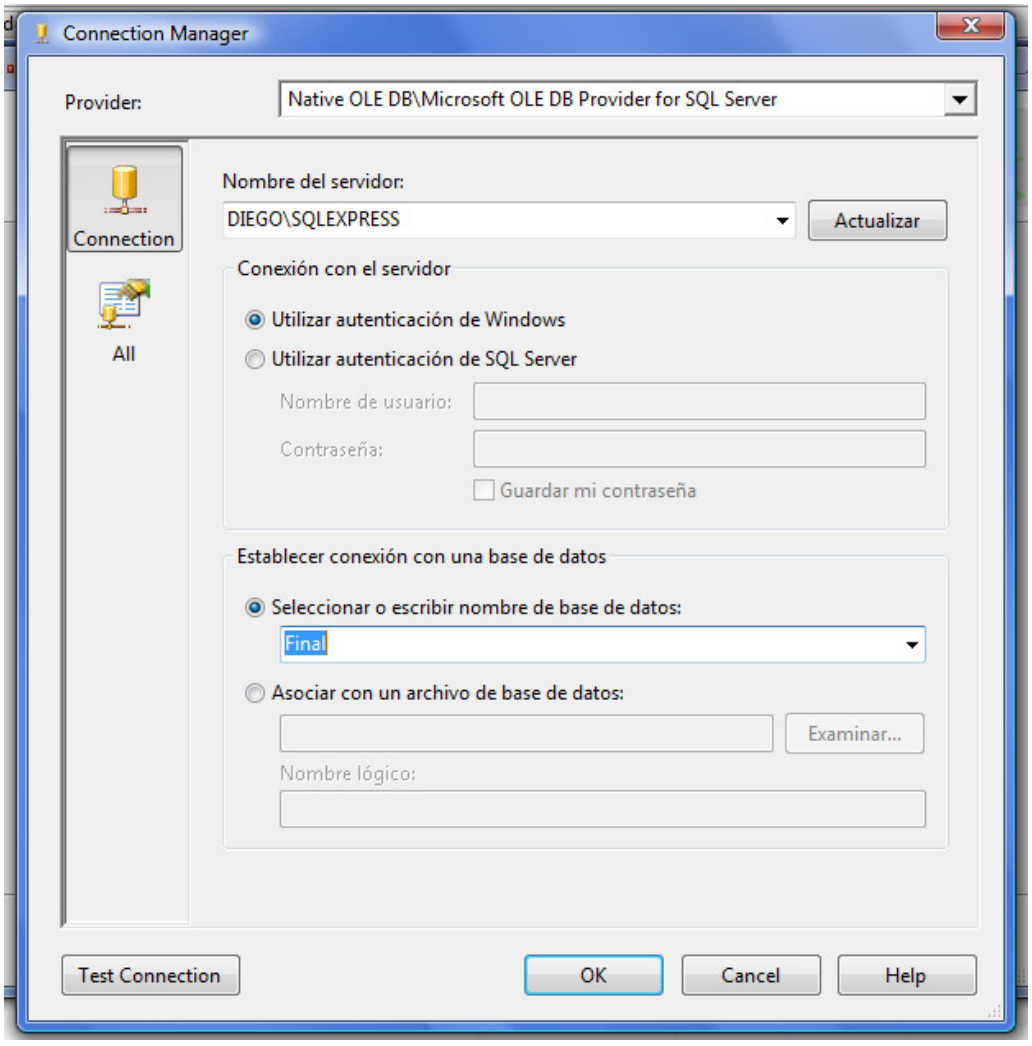


Fuente figura: Elaboración propia.

Después de abierto el asistente se da click en siguiente y aparece la página de cómo definir una conexión, se selecciona la opción new.

En la caja de dialogo del administrador de conexiones se verifica en la parte superior que la opción NativeOLE DB\ Microsoft DB OLE Provider for SQL Server esté seleccionada. Después de esto se selecciona el nombre del servidor el cual es usualmente el nombre del motor de la base de datos “localhost”, se selecciona la base de datos a la que nos conectaremos y damos click en Ok, luego next y por ultimo finish.

Figura 20 Conection manager.



Fuente figura: Elaboración propia.

- Definición de la vista de datos.

  En este paso lo que se busca es crear una vista unificada para los metadatos la cual puede provenir de tablas o de vistas definidas en el proyecto. El guardar los metadatos en la fuente de datos, permite al desarrollador trabajar con los mismos durante el desarrollo de la aplicación sin necesidad de abrir conexiones.

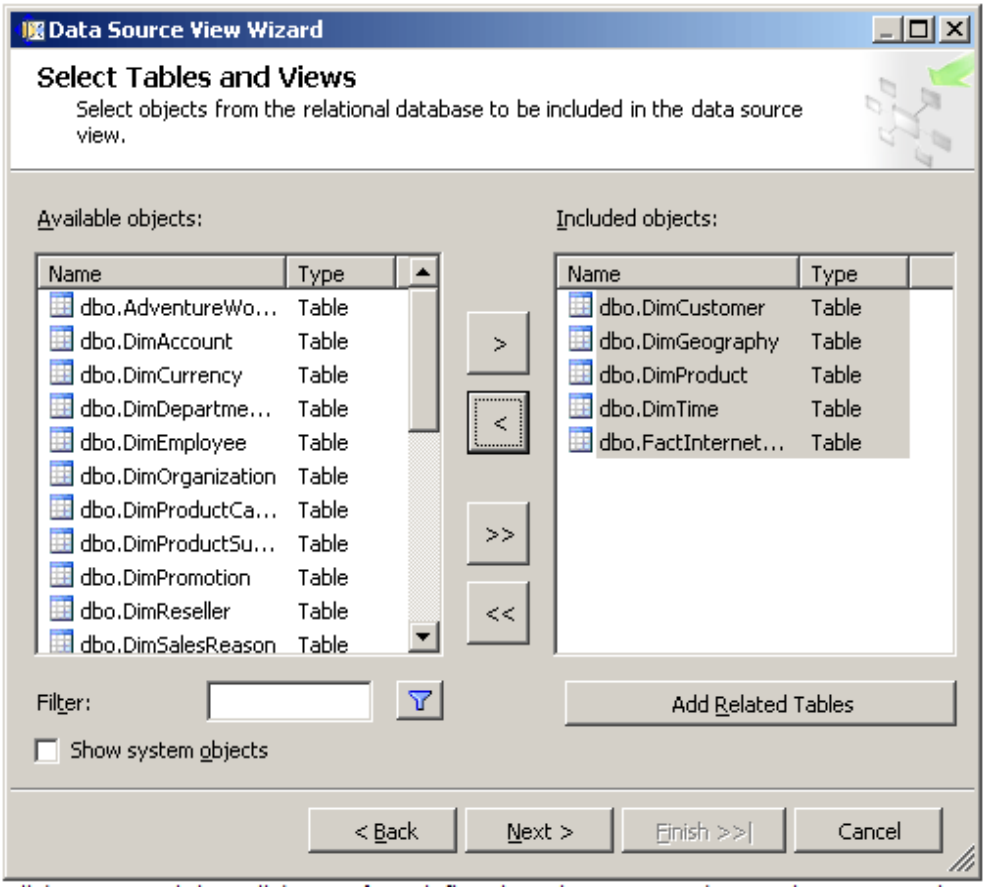
  Para la definición de la fuente de datos lo que se hace es ir al explorador de soluciones dentro del SQL Server Development Studio y se da click derecho en data source views -> new data source view.

  Al abrirse el asistente se da click en next. Bajo el campo relational data sources se selecciona el modelo de DWH con el que estemos trabajando y se da click en siguiente.

  Al aparecer las tablas y las vistas se seleccionan los objetos que se requieran, tales como las tablas de dimensiones y hechos y se da click en next -> finish.

  Es recomendable cambiar los nombres de las tablas por unos mas nemotécnicos eliminando el prefijo Dim, dando click derecho sobre la tabla a la cual se le desea cambiar el nombre.

Figura 21 Vistas del data source.



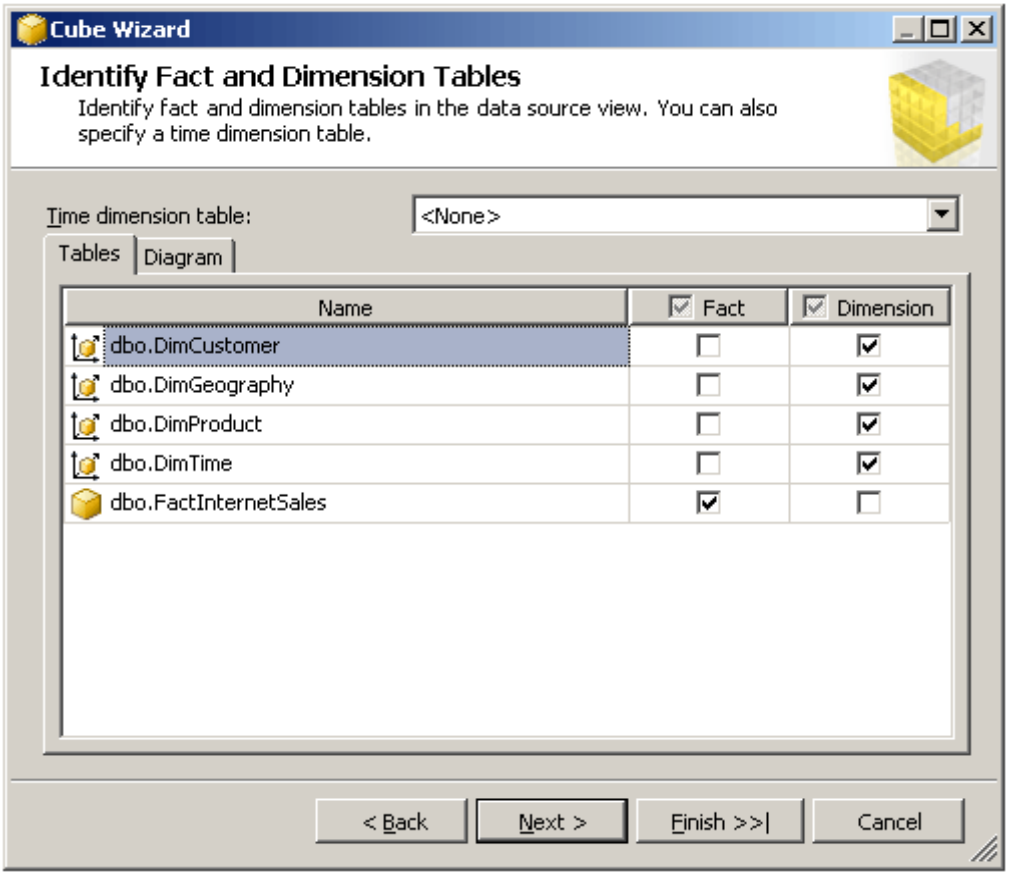
Fuente figura: Elaboración propia.

3.4 DEFINICIÓN Y DESARROLLO DEL CUBO.

Después de las vistas de la fuente de datos, se puede proceder a definir el cubo de analysis services. Lo que se hace en este paso es seleccionar las dimensiones que se utilizarán y el cubo del cual estas formarán parte.
Para hacer esto se harán los siguientes pasos.

- Definición del cubo.
  Para la definición de un cubo se ubica en el explorador de soluciones la opción cubes; se da click derecho en ella y después se selecciona la opción new cube. Al abrirse el asistente, se da click en siguiente, En la página Select Build Metod, se seleccionan las opciones Build the cube using a data source y Auto Build y se da click en siguiente; luego se verifica que la data source que se está utilizando es la de nuestro DWH y se da click en siguiente.

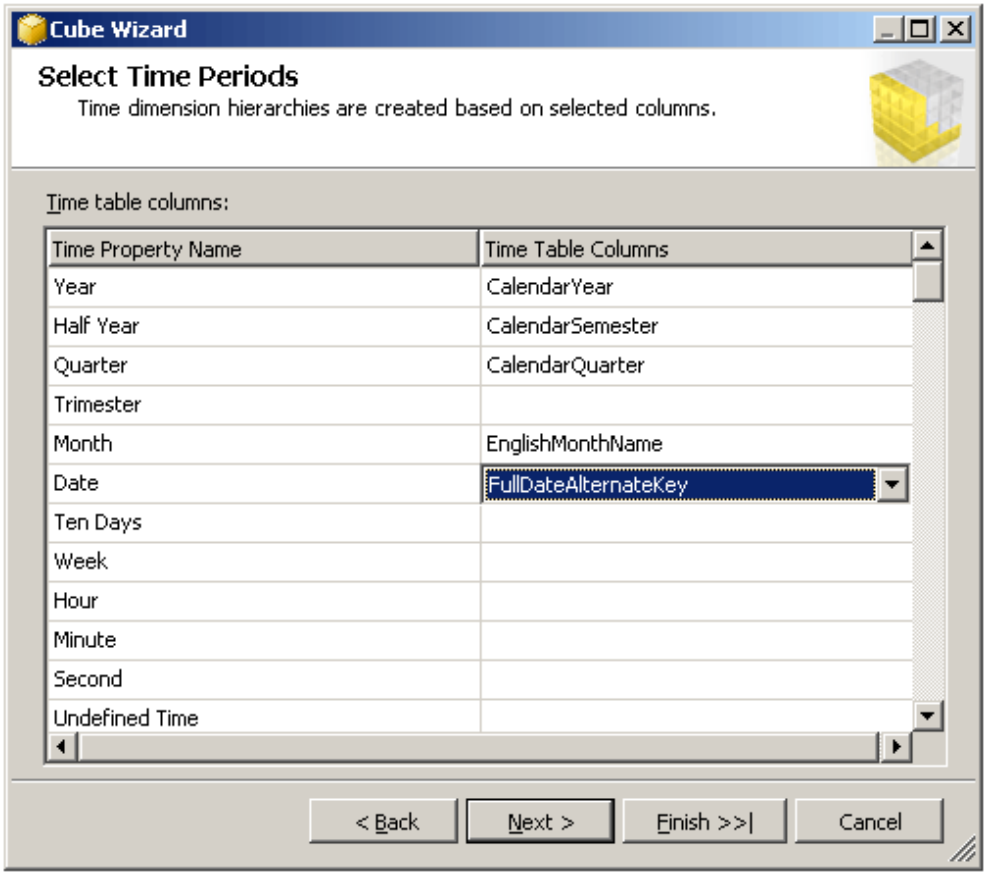
Figura 22 Asistente para cubos.



Fuente figura: Elaboración propia.

Se selecciona la dimensión de tiempo y se da click en Next., aquí se mapean las propiedades de tiempo.

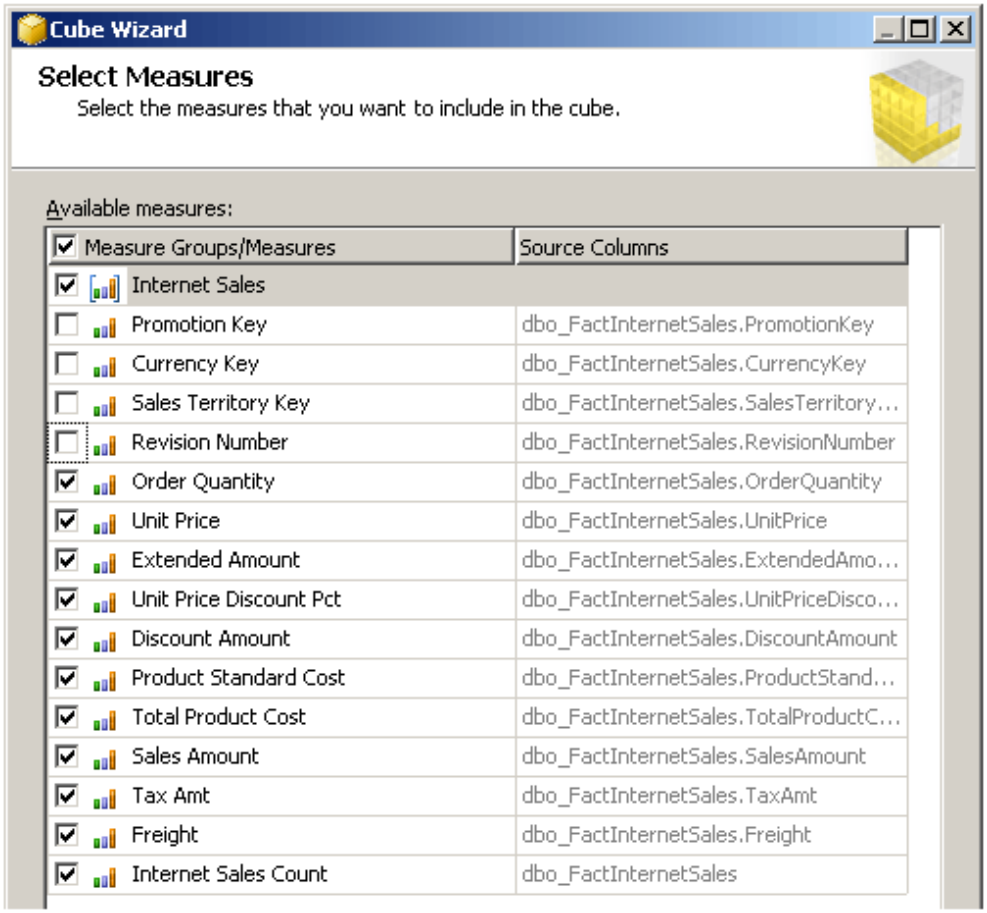
Figura 23 Configuración de las propiedades de tiempo.



Fuente figura: Elaboración propia.

Después de esto se da click en siguiente y se muestra a continuación la página de selección de medidas; se seleccionan las medidas a utilizar y se da click en Next.

Figura 24 Configuración de las medidas:



Fuente figura: Elaboración propia.

Después de esto se chequea que las dimensiones sean las seleccionadas y se da click en Next -> Finish.

- Como desplegar un proyecto de Analysis Services.

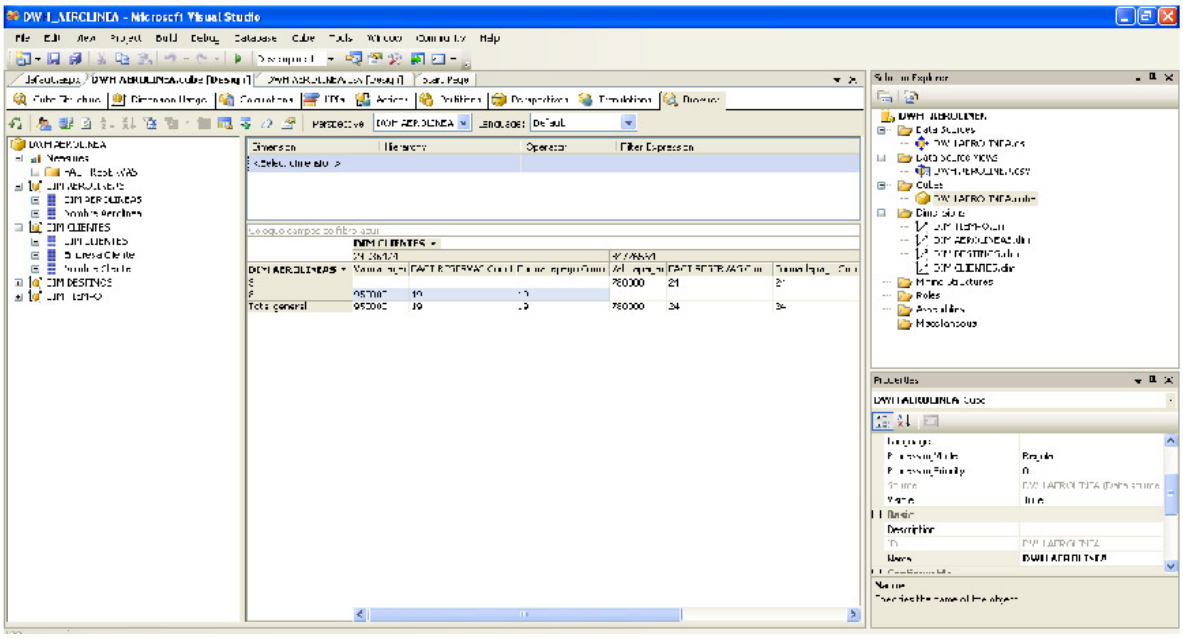
  En este paso lo que se hace es desplegar el proyecto creado dentro de una instancia específica de Analisis Services con el objetivo de procesar el cubo y sus dimensiones. Esto tiene como objetivo que los datos se copien desde la fuente de datos a los objetos del cubo.

En el explorador de soluciones se da click derecho sobre el icono del proyecto creado y luego se da click en propiedades. En propiedades de configuración se da click en Show Deployment Progress; aquí se podrá verificar que el cubo se halla construido, desplegado y creado sin errores. Al dar click sobre cada dimensión a la derecha de la pantalla se mostrará en detalle el proceso de desplegado de cada dimensión.

- Visualización del cubo desplegado.

  Se selecciona la pestaña browser situada a la derecha la cual mostrará las medidas y las dimensiones del cubo.

Figura 25 Visualización del cubo.

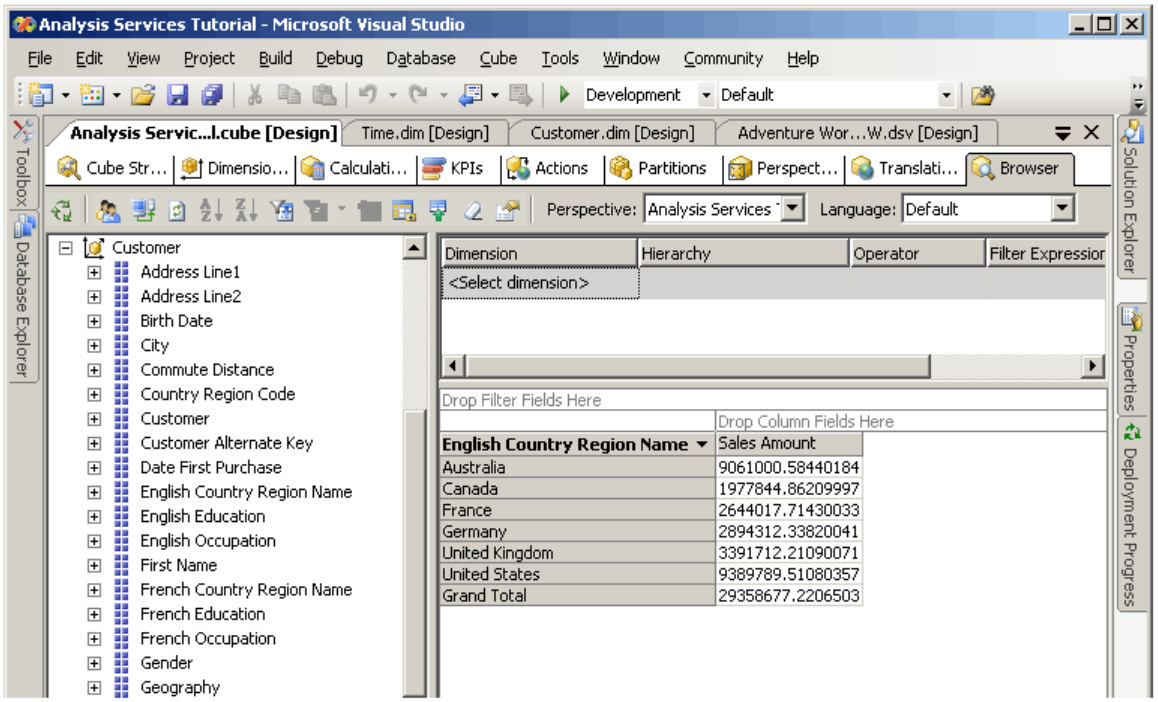


Fuente figura: Elaboración propia.

Se selecciona la opción medidas y estas se desplegarán; las medidas que se vayan a utilizar se seleccionarán y se arrastran al panel de datos “Drop Filter Fields Here”.

Después de esto se expanden las dimensiones que se desee visualizar.

Figura 26 Visualización de las dimensiones.



Fuente figura: Elaboración propia.

Se procede a seleccionar el objeto que se desee visualizar arrastrándolo al panel de datos; repitiendo el procedimiento con cualquier objeto o grupo de objetos que se desee visualizar.

(Tutorial de analysis services. Consultado en Enero, 2009. Disponible en *http://msdn.microsoft.com/es-es/library/ms166713(SQL.90).aspx)*

4. CONCLUSIONES

Podemos ver que las aplicaciones de BI son utilizadas en el mundo empresarial con el objetivo de tener una herramienta poderosa para el proceso de toma de decisiones. Las aplicaciones BI permiten un mejor uso de los datos de la empresa para su utilización en la toma de decisiones.

Las aplicaciones BI han tomado gran importancia dentro del mundo de los negocios actual, ya que la información es el recurso que utilizan las empresas para apoyarse en la toma de decisiones y BI provee la solución a esta necesidad, ya que permite utilizar información para generar un valor agregado al negocio.
Las aplicaciones de BI a diferencia de las bases de datos transaccionales, están enfocadas no al proceso de transacciones sino a la consulta y el análisis, o lo que es mas exacto, una DWH es una copia de las transacciones de datos específicamente estructuradas para la consulta y el análisis.

Las aplicaciones de BI permiten que basados en la información almacenada de las empresas, podamos utilizar esta como un apoyo en la toma de decisiones.
Debido a que en la actualidad las organizaciones pueden llegar a manejar grandes volúmenes de información, se hace necesario contar con herramientas que permitan manipular esta información de modo que se minimice el riesgo que implica el proceso de toma de decisiones.

Las aplicaciones BI, posibilitan el poder contar con información proveniente de diversas fuentes de datos visualizando esta en una forma estructurada y fácil de analizar.

Debido a que la información de las organizaciones es administrada por medio de las bases de datos transaccionales, el paso más fácil y lógico es utilizar estas para

la extracción de la información la cual pueda ser utilizada en el proceso de toma de decisiones. El problema aquí yace en que estos modelos de bases de datos están diseñados para el manejo de transacciones pero no para el análisis de la información. Aquí es donde entran los modelos OLAP o cubos, los cuales a diferencia de las bases de datos OLTP, tienen un diseño enfocado al proceso analítico.

En la arquitectura DWH, lo que se hizo fue integrar estos dos modelos de bases de datos para que fueran útiles al propósito de servir como fuente de datos confiable, enfocada a la toma de decisiones aprovechando las fortalezas de ambos modelos.
Así, mientras que las bases de datos transaccionales funcionan como fuente de datos, las bases de datos OLAP estructuran la información proveniente de las anteriores para que pueda ser utilizada para el análisis y la toma de decisiones.
La ventaja de los modelos DWH es que los datos pueden provenir de diferentes fuentes de datos o bases de datos distribuidas y esta información puede ser estructurada dentro de un cubo el cual posibilite el contar con esta información para la toma de decisiones.

Los otros elementos importantes en la arquitectura DWH son los ETL los cuales incluyen procedimientos almacenados y las herramientas de visualización o despliegue del cubo. Los primeros tienen como función la extracción, transformación y carga de los datos, sirviendo como enlace entre la base de datos transaccional y la BD OLAP, y los segundos tienen como finalidad el ser utilizados para la visualización del cubo. Existen diversas herramientas para esto, tales como Pentaho, Excel, Analyisis services etc.

Tomando en cuenta lo anteriormente expuesto, se vio la necesidad de tener un documento que explicara de un modo claro y conciso, el método para construir

una aplicación de BI explicando los conceptos detrás del modelo DWH, con el objetivo de que cualquier persona que posea conocimiento en bases de datos, pero que no posea el mismo en la arquitectura DWH tenga a mano una herramienta que le permita crear sus propios cubos, finalidad última de este trabajo.

Este proyecto sirvió en lo profesional para poner en práctica todos los conocimientos adquiridos durante la carrera en el área de bases de datos tales como procedimientos almacenados, normalización, desnormalización y diseño de bases de datos OLAP y OLTP.

Business Inteligence, tiene una amplia aplicabilidad ya que esta permite crear soluciones enfocadas al análisis de los datos y por ende al mejor uso de estos. Este conocimiento me genera un valor agregado como profesional en ingeniería de sistemas.

Este proyecto puede servir de base para el desarrollo de proyectos futuros enfocados a la utilización de BI en bases de datos distribuidas, como se puede aplicar BI en el mundo empresarial en distintos tipos de organizaciones, la creación de nuevos modelos OLAP o el estudio de la eficiencia de los diferentes modelos OLAP, motores de bases de datos etc.

5. GLOSARIO

BASE DE DATOS TRANSACCIONAL: Una base de datos transaccional es una base de datos orientada a la realización de transacciones; lo que significa que la información se procesa descomponiéndola en forma unitaria e indivisible. Las bases de datos transaccionales manejan Querys para consultar la información

NORMALIZACIÓN: Es el proceso que se realiza sobre una base de datos con el fin de aumentar la integridad y disminuir la redundancia y las dependencias funcionales de la estructura de la BD. Esto se logra clasificando la información en una forma restrictiva siendo más elevada la forma normal mientras mas restrictivas sean estas reglas

DESNORMALIZACION: En el proceso de desnormalización lo que se busca es aumentar el rendimiento de la base de datos en detrimento de la integridad de la base de datos. Sin embargo la desnormalización es necesaria en el diseño de los DWH debido a que estos están enfocados a reducir el tiempo de las consultas.

6. BIBLIOGRAFIA

REFRENCIAS BIBLIOGRAFICAS PARA LIBROS, FOLLETOS E INFORMES

SILBERSCHATZ Abraham, KORT Henry, SUDARSHAN S. Fundamentos de diseño de bases de datos. University of Minnesota. Mac Graw Hill 2007

AKEEL I Din. Structured Query Language. Ncc Blackwell 2004

VAN DER LANS Rick. Introduction to SQL. Pearson 2006

KIMBALL Ralph. The Data Warehouse Tool kit: Practical techniques for building dimensional data warehouses. John Wiley and Sons 1996

JIAWEY Han, KAMBER Micheline. Data Minning Concepts and Techniques. Morgan Kaufmann 2000

CJ Date. Introducción a los sistemas de bases de datos. Pearson 2002

SILBERSCHATZ Abraham, KORT Henry. Fundamentos de sistemas de bases de datos. Addison Wesley 2002

JONSON James L, Bases de datos, modelos, lenguajes, diseño. Alfaomega 2000

DATE, C.J, Bases de datos, una guía práctica. Addison Wesley 1987

DE MIGUEL CASTAÑO Adoración, PIATINNI VELTHUIS Mario Gerardo. Addison Wesley 1993

REFERENCIAS DOCUMENTALES PARA FUENTES DE INFORMACION ELECTRONICA

Arquitectura del data warehouse. Consultado en Noviembre 2008
Disponible en *sinnexus.com/business_intelligence/olap_vs_oltp.aspx*.

Differences between OLTP and OLAP. Consultado en Noviembre 2008 Disponible en *it.toolbox.com/wiki/index.php/**Differences_between_OLTP**_and_**OLAP***

Modelamiento multidimensional. Consultado en Noviembre, 2008. Disponible en *www.inf.udec.cl/~revista/ediciones/edicion4/modmulti.PDF*

Todo el Business Inteligence. Consultado en Noviembre, 2008. Disponible en ***todo**bi.blogspot.com/*

Data warehousing review. Consultado en Noviembre, 2008. Disponible en *www.dwreview.com*

Microsoft SQL Server Performance Guide. Consultado en Octubre, 2008. Disponible en *download.**microsoft**.com/download/8/5/e/85eea4fa-b3bb-4426-97d0-7f7151b2011c/SSAS**2005**Perf**Guide**.doc*

Tutorial de analysis services. Consultado en Enero, 2009. Disponible en *http://msdn.microsoft.com/es-es/library/ms166713(SQL.90).aspx*

REFERENCIAS PROGRAMAS DE COMPUTACION (SOFTWARE)

MS SQL SERVER 2005 PROFESSIONAL EDITION [Programa de computación]

ANEXOS

ANEXO A SCRIPTS DE CREACION DE LAS TABLAS DEL CUBO.

```
/**TABLA DIMENSIONAL AEROLINEAS**/
USE [DWH_AEROLINEA]
GO
/****** Object:  Table [dbo].[DIM_AEROLINEAS]    Script Date: 02/02/2009
11:58:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[DIM_AEROLINEAS](
      [id_aerolinea] [int] NOT NULL,
      [nombre_aerolinea] [nvarchar](50) COLLATE Modern_Spanish_CI_AS
NULL,
 CONSTRAINT [PK_DIM_AEROLINEAS] PRIMARY KEY CLUSTERED
(
      [id_aerolinea] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]


/**TABLA DIMENSIONAL CLIENTES**/
USE [DWH_AEROLINEA]
GO
/****** Object:  Table [dbo].[DIM_CLIENTES]    Script Date: 02/02/2009
12:01:12 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[DIM_CLIENTES](
      [id_cliente] [int] NOT NULL,
      [nombre_cliente] [nvarchar](50) COLLATE Modern_Spanish_CI_AS NULL,
      [empresa_cliente] [nvarchar](50) COLLATE Modern_Spanish_CI_AS NULL,
 CONSTRAINT [PK_DIM_CLIENTES] PRIMARY KEY CLUSTERED
(
      [id_cliente] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

```
/**TABLA DIMENSIONAL DESTINOS**/
USE [DWH_AEROLINEA]
GO
/****** Object:  Table [dbo].[DIM_DESTINOS]    Script Date: 02/02/2009
12:02:44 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[DIM_DESTINOS](
      [cod_destino] [nvarchar](50) COLLATE Modern_Spanish_CI_AS NOT NULL,
      [continente] [nchar](20) COLLATE Modern_Spanish_CI_AS NOT NULL,
      [pais] [nvarchar](50) COLLATE Modern_Spanish_CI_AS NULL,
      [region] [nvarchar](50) COLLATE Modern_Spanish_CI_AS NULL,
      [ciudad] [nvarchar](50) COLLATE Modern_Spanish_CI_AS NULL,
 CONSTRAINT [PK_DIM_DESTINOS] PRIMARY KEY CLUSTERED
(
      [cod_destino] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]



/**TABLA DIMENSIONAL TIEMPO**/
USE [DWH_AEROLINEA]
GO
/****** Object:  Table [dbo].[DIM_TIEMPO]    Script Date: 02/02/2009
12:04:19 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[DIM_TIEMPO](
      [id_fecha] [int] IDENTITY(1,1) NOT NULL,
      [fecha] [smalldatetime] NOT NULL,
      [anio] [int] NULL,
      [semestre] [int] NULL,
      [trimestre] [int] NULL,
      [mes] [int] NULL,
      [dia] [int] NULL,
 CONSTRAINT [PK_DIM_TIEMPO] PRIMARY KEY CLUSTERED
(
      [id_fecha] ASC
)WITH (PAD_INDEX  = OFF, STATISTICS_NORECOMPUTE  = OFF, IGNORE_DUP_KEY =
OFF, ALLOW_ROW_LOCKS  = ON, ALLOW_PAGE_LOCKS  = ON) ON [PRIMARY]
) ON [PRIMARY]
```

```
/**TABLA DE HECHOS**/
USE [DWH_AEROLINEA]
GO
/****** Object:  Table [dbo].[FACT_RESERVAS]    Script Date: 03/07/2009
13:41:53 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[FACT_RESERVAS](
      [id_aerolinea] [int] NOT NULL,
      [id_cliente] [int] NOT NULL,
      [id_destino] [nvarchar](50) NOT NULL,
      [fecha] [int] IDENTITY(1,1) NOT NULL,
      [id_fecha] [datetime] NOT NULL,
      [valorapagar] [money] NULL,
      [formadepago] [int] NULL
) ON [PRIMARY]

GO
ALTER TABLE [dbo].[FACT_RESERVAS]  WITH CHECK ADD  CONSTRAINT
[FK_FACT_RESERVAS_DIM_AEROLINEAS] FOREIGN KEY([id_aerolinea])
REFERENCES [dbo].[DIM_AEROLINEAS] ([id_aerolinea])
GO
ALTER TABLE [dbo].[FACT_RESERVAS] CHECK CONSTRAINT
[FK_FACT_RESERVAS_DIM_AEROLINEAS]
GO
ALTER TABLE [dbo].[FACT_RESERVAS]  WITH CHECK ADD  CONSTRAINT
[FK_FACT_RESERVAS_DIM_CLIENTES] FOREIGN KEY([id_cliente])
REFERENCES [dbo].[DIM_CLIENTES] ([id_cliente])
GO
ALTER TABLE [dbo].[FACT_RESERVAS] CHECK CONSTRAINT
[FK_FACT_RESERVAS_DIM_CLIENTES]
GO
ALTER TABLE [dbo].[FACT_RESERVAS]  WITH CHECK ADD  CONSTRAINT
[FK_FACT_RESERVAS_DIM_DESTINOS] FOREIGN KEY([id_destino])
REFERENCES [dbo].[DIM_DESTINOS] ([cod_destino])
GO
ALTER TABLE [dbo].[FACT_RESERVAS] CHECK CONSTRAINT
[FK_FACT_RESERVAS_DIM_DESTINOS]
GO
ALTER TABLE [dbo].[FACT_RESERVAS]  WITH CHECK ADD  CONSTRAINT
[FK_FACT_RESERVAS_DIM_TIEMPO] FOREIGN KEY([fecha])
REFERENCES [dbo].[DIM_TIEMPO] ([id_fecha])
GO
            ALTER TABLE [dbo].[FACT_RESERVAS] CHECK CONSTRAINT
                        [FK_FACT_RESERVAS_DIM_TIEMPO]
```

ANEXO B PROCEDIMIENTOS ALMACENADOS PARA CARGA DEL CUBO.

```
/*PROCEDIMIENTO CARGA HECHOS*/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE  PROCEDURE [dbo].[sp_CargaHechos] AS

DECLARE @idaerolinea nchar(10)
DECLARE @idcliente int
DECLARE @iddestino nvarchar(50)
DECLARE @idfecha datetime
DECLARE @valorapagar money
DECLARE @formadepago int



DECLARE cu_hechos SCROLL CURSOR  FOR

       SELECT Aerolinea.dbo.TB_AEROLINEAS.id_aerolinea,
              Aerolinea.dbo.TB_CLIENTES.id_cliente,
              Aerolinea.dbo.TB_CIUDADES.cod_ciudad,
              Aerolinea.dbo.TB_RESERVAS.fecha_salida,
              Aerolinea.dbo.TB_PAGOS.valorapagar,
              Aerolinea.dbo.TB_PAGOS.formadepago
       FROM   Aerolinea.dbo.TB_RESERVAS
       INNER JOIN Aerolinea.dbo.TB_PAGOS ON
Aerolinea.dbo.TB_RESERVAS.cod_factura = Aerolinea.dbo.TB_PAGOS.id_factura
AND
                  Aerolinea.dbo.TB_RESERVAS.cod_despachador =
Aerolinea.dbo.TB_PAGOS.cod_despachador
       INNER JOIN Aerolinea.dbo.TB_CLIENTES ON
Aerolinea.dbo.TB_RESERVAS.id_cliente =
Aerolinea.dbo.TB_CLIENTES.id_cliente
       INNER JOIN Aerolinea.dbo.TB_AEROLINEAS ON
Aerolinea.dbo.TB_RESERVAS.cod_aerolinea =
Aerolinea.dbo.TB_AEROLINEAS.id_aerolinea
       INNER JOIN Aerolinea.dbo.TB_CIUDADES ON
Aerolinea.dbo.TB_RESERVAS.origen = Aerolinea.dbo.TB_CIUDADES.cod_ciudad

OPEN  cu_hechos

      FETCH NEXT FROM cu_hechos

      INTO  @idaerolinea, @idcliente, @iddestino, @idfecha, @valorapagar,
@formadepago

            WHILE  @@fetch_status =  0

      BEGIN
```

```
            IF  NOT EXISTS (SELECT
DWH_AEROLINEA.dbo.FACT_RESERVAS.id_fecha
                              FROM       DWH_AEROLINEA.dbo.FACT_RESERVAS
                              WHERE
DWH_AEROLINEA.dbo.FACT_RESERVAS.id_aerolinea = @idaerolinea and

DWH_AEROLINEA.dbo.FACT_RESERVAS.id_fecha = @idfecha and

DWH_AEROLINEA.dbo.FACT_RESERVAS.id_cliente = @idcliente and

DWH_AEROLINEA.dbo.FACT_RESERVAS.id_destino = @iddestino and

DWH_AEROLINEA.dbo.FACT_RESERVAS.valorapagar = @valorapagar and

DWH_AEROLINEA.dbo.FACT_RESERVAS.formadepago = @formadepago)
               BEGIN

                        INSERT INTO DWH_AEROLINEA.dbo.FACT_RESERVAS
(id_aerolinea,id_cliente, id_destino, id_fecha, valorapagar, formadepago)
                        VALUES (@idaerolinea, @idcliente, @iddestino,
@idfecha, @valorapagar, @formadepago)

               END

            FETCH NEXT FROM cu_hechos

            INTO  @idaerolinea, @idcliente, @iddestino, @idfecha,
@valorapagar, @formadepago
      END
CLOSE cu_hechos

DEALLOCATE cu_hechos
```

```
/**PROCEDIMIENTO CARGA DIMENSION AEROLINEAS**/
CREATE  PROCEDURE sp_CargaAerolineas AS

DECLARE @idAerolinea int
DECLARE @nombreAerolinea  nvarchar(50)


DECLARE cu_aerolineas SCROLL CURSOR  FOR
      SELECT     Aerolinea.dbo.TB_AEROLINEAS.id_aerolinea,
                   Aerolinea.dbo.TB_AEROLINEAS.nombre_aerolinea
        FROM       Aerolinea.dbo.TB_AEROLINEAS

OPEN  cu_aerolineas

      FETCH NEXT FROM cu_aerolineas

      INTO  @idAerolinea, @nombreAerolinea

            WHILE  @@fetch_status =  0

      BEGIN

             IF  NOT EXISTS (SELECT
DWH_AEROLINEA.dbo.DIM_AEROLINEAS.id_aerolinea
                              FROM       DWH_AEROLINEA.dbo.DIM_AEROLINEAS
                              WHERE
DWH_AEROLINEA.dbo.DIM_AEROLINEAS.id_aerolinea = @idAerolinea)

               BEGIN

                        INSERT INTO DWH_AEROLINEA.dbo.DIM_AEROLINEAS
(id_aerolinea, nombre_aerolinea)
                        VALUES (@idAerolinea, @nombreAerolinea)

               END

            FETCH NEXT FROM cu_aerolineas

            INTO  @idAerolinea, @nombreAerolinea
      END
CLOSE cu_aerolineas

DEALLOCATE cu_aerolineas

GO
```

```
/**PROCEDIMIENTO CARGA DIMENSION CLIENTES**/
CREATE  PROCEDURE sp_CargaClientes AS

DECLARE @nombreEmpresa nvarchar(50)
DECLARE @idCliente int
DECLARE @nombreCliente  nvarchar(50)


DECLARE cu_clientes SCROLL CURSOR  FOR
      SELECT     Aerolinea.dbo.TB_CLIENTES.empresa,
                   Aerolinea.dbo.TB_PERSONAS.cedula,
                   Aerolinea.dbo.TB_PERSONAS.nombres +
Aerolinea.dbo.TB_PERSONAS.apellidos AS nombreCompleto
        FROM       Aerolinea.dbo.TB_CLIENTES
        INNER JOIN Aerolinea.dbo.TB_PERSONAS ON
Aerolinea.dbo.TB_CLIENTES.id_cliente = Aerolinea.dbo.TB_PERSONAS.cedula

OPEN  cu_clientes

      FETCH NEXT FROM cu_clientes

      INTO  @nombreEmpresa, @idCliente, @nombreCliente

            WHILE  @@fetch_status =  0

      BEGIN

             IF  NOT EXISTS (SELECT
DWH_Aerolinea.dbo.DIM_CLIENTES.id_cliente
                             FROM       DWH_Aerolinea.dbo.DIM_CLIENTES
                             WHERE
DWH_Aerolinea.dbo.DIM_CLIENTES.id_cliente = @idCliente)

               BEGIN

                        INSERT INTO DWH_Aerolinea.dbo.DIM_CLIENTES
(id_cliente, nombre_cliente, empresa_cliente)
                        VALUES (@idCliente, @nombreCliente,
@nombreEmpresa)

               END

            FETCH NEXT FROM cu_clientes

            INTO  @nombreEmpresa, @idCliente, @nombreCliente
      END
CLOSE cu_clientes

DEALLOCATE cu_clientes

GO
```

```
/*PROCEDIMIENTO CARGA DIMENSION DESTINOS***/
CREATE  PROCEDURE sp_CargaDestinos AS

DECLARE @idDestino nvarchar(50)
DECLARE @continente nchar(20)
DECLARE @pais nvarchar(50)
DECLARE @region  nvarchar(50)
DECLARE @ciudad nvarchar (50)


DECLARE cu_destinos SCROLL CURSOR  FOR
      SELECT         Aerolinea.dbo.TB_CIUDADES.cod_ciudad,
                  Aerolinea.dbo.TB_CONTINENTES.nombre_continente,
                  Aerolinea.dbo.TB_PAISES.nombre_pais,
                  Aerolinea.dbo.TB_REGIONES.nombre_region,
                  Aerolinea.dbo.TB_CIUDADES.nombre_ciudad
        FROM       Aerolinea.dbo.TB_CIUDADES
        INNER JOIN Aerolinea.dbo.TB_REGIONES ON
Aerolinea.dbo.TB_CIUDADES.cod_region =
Aerolinea.dbo.TB_REGIONES.cod_region AND
                  Aerolinea.dbo.TB_CIUDADES.cod_pais =
Aerolinea.dbo.TB_REGIONES.cod_pais
        INNER JOIN Aerolinea.dbo.TB_PAISES ON
Aerolinea.dbo.TB_CIUDADES.cod_pais = Aerolinea.dbo.TB_PAISES.cod_pais
        INNER JOIN Aerolinea.dbo.TB_CONTINENTES ON
Aerolinea.dbo.TB_CIUDADES.cod_continente =
Aerolinea.dbo.TB_CONTINENTES.cod_continente


OPEN  cu_destinos

      FETCH NEXT FROM cu_destinos

      INTO  @idDestino, @continente, @pais, @region, @ciudad

            WHILE  @@fetch_status =  0

      BEGIN

             IF  NOT EXISTS (SELECT
DWH_AEROLINEA.dbo.DIM_DESTINOS.cod_destino
                           FROM       DWH_AEROLINEA.dbo.DIM_DESTINOS
                           WHERE
DWH_AEROLINEA.dbo.DIM_DESTINOS.cod_destino = @idDestino)

               BEGIN

                        INSERT INTO DWH_AEROLINEA.dbo.DIM_DESTINOS
(cod_destino, continente, pais, region, ciudad)
                        VALUES (@idDestino, @continente, @pais, @region,
@ciudad)
```

```
                END

            FETCH NEXT FROM cu_destinos

            INTO  @idDestino, @continente, @pais, @region, @ciudad
      END
CLOSE cu_destinos

DEALLOCATE cu_destinos
```

```
/**PROCEDIMIENTO CARGA DIMENSION TIEMPO**/
CREATE  PROCEDURE sp_CargaTiempo AS

DECLARE @idfecha smalldatetime
DECLARE @ano int
DECLARE @semestre int
DECLARE @trimestre int
DECLARE @mes int
DECLARE @dia int
DECLARE @fechaIni smalldatetime
DECLARE @fechaFin smalldatetime
DECLARE @fechaAct smalldatetime

        SET @fechaIni = '01/01/1996'
        SET @fechaFin = '01/01/2010'
        SET @fechaAct = @fechaIni

            WHILE  @fechaAct <  @fechaFin

      BEGIN

             SET @ano = YEAR(@fechaAct)
             SET @mes = MONTH(@fechaAct)
             SET @dia = DAY(@fechaAct)

             IF (@mes <= 6)
            SET @semestre = 1
             ELSE
                SET @semestre = 2

             IF (@mes <= 3)
            SET @trimestre = 1
             ELSE
                IF (@mes <= 6)
                    SET @trimestre = 2
                ELSE
                   IF (@mes <= 9)
                       SET @trimestre = 3
                   ELSE
                       IF (@mes <= 12)
                           SET @trimestre = 4

             IF  NOT EXISTS (SELECT DWH_AEROLINEA.dbo.dim_tiempo.fecha
                             FROM   DWH_AEROLINEA.dbo.dim_tiempo
                             WHERE
DWH_AEROLINEA.dbo.dim_tiempo.fecha = @fechaAct)
             BEGIN

                INSERT INTO DWH_AEROLINEA.dbo.DIM_TIEMPO(fecha, anio,
semestre, trimestre, mes, dia)
```

```
                VALUES (@fechaAct, @ano, @semestre, @trimestre, @mes,
@dia)

            END

              SET @fechaAct = DATEADD(Day,1,@fechaAct)

      END

GO
```