Universidad Central “Marta Abreu” de Las Villas

Facultad de Matemática, Física y Computación

Trabajo de Diploma

Herramienta Case “Asistente-DW’’ para el Análisis y Diseño de Almacenes de Datos.

Autor: Jesús Alvarez Alfonso.

Tutores: **Dr. Rosendo Moreno Rodríguez.**
M.Sc. Lindsay Gómez Beltrán.

Santa Clara, 2012
“Año 54 de la Revolución”

Dictamen

Hago constar que el presente trabajo fue realizado en la Universidad Central “Marta Abreu” de Las Villas como parte de la culminación de los estudios de la especialidad de licenciado en Ciencias de la Computación autorizando a que el mismo sea utilizado por la institución, para los fines que estime conveniente, tanto de forma parcial como total y que además no podrá ser presentado en eventos ni publicado sin la autorización de la Universidad.

_____

Firma del autor

Los abajo firmantes, certificamos que el presente trabajo ha sido realizado según acuerdos de la dirección de nuestro centro y el mismo cumple con los requisitos que debe tener un trabajo de esta envergadura referido a la temática señalada.

_____ _____

Firma del tutor

Firma del Jefe del Seminario de Bases de Datos

Pensamiento

Con el conocimiento se acrecientan las dudas, *pero* ***las dudas del sabio son más inteligentes*** *porque son dudas que han dejado atrás muchas respuestas que el ignorante aún tiene como horizonte.*

Johann Wolfgang Goethe.

Agradecimientos

Estimo necesario agradecer a todos los que han colaborado con la creación de la herramienta case Asistente-DW, así como los conocimientos que me han sido brindados de la temática de almacenes de datos. Es innegable también lo útil que me ha sido que mis tutores me brindaran la vasta experiencia que tienen en el tema.

Es importante mencionar entonces a quienes me han apoyado:

Dr. Rosendo de Jesús Moreno Rodríguez.
Msc. Lindsay Gómez Beltrán.
Lic. Yoan Martínez López.
Lic. Enrique Alfonso Casanovas Pedre.

Dedicatoria

Este trabajo de diploma que confirma la culminación de los estudios en la facultad M-F-C en la Lic. Ciencias de la Computación, está dedicado a mis padres, mi hermano, mis abuelos y toda mi familia en general.

Resumen

El trabajo de Análisis y Modelado para Almacenes de Datos, específicamente aplicando el Modelo Estrella o el de Constelación de estrellas, en varias ocasiones se torna complicado, puesto que no se poseen herramientas comerciales que generen estos modelos adecuadamente para el almacén a partir de sistemas de información legados, que generalmente existen en base al modelo relacional. Teniendo en cuenta esta dificultad se plantea en este trabajo la creación de una herramienta CASE de Análisis y Diseño de sistemas de información para la toma de decisiones, que facilite el trabajo de analistas, implementadores y usuarios, al revisar de forma automática los metadatos estructurales de las Bases de Datos relacionales existentes, y proponer una estructura dimensional óptima para el Almacén. Este software debe ser simple de usar hasta para los usuarios que no tengan todo el conocimiento y experiencia requerido sobre el tema.

PALABRAS CLAVE

Almacenes de Datos, Modelado Dimensional.

Abstract

Work Analysis and Modeling for Data Warehouses, specifically applying the Model Star or the constellation of stars, repeatedly becomes complicated, since it does not have marketing tools that generate these models adequately for the store from systems information bequests, which are generally, based on the relational model. Given this difficulty arises in this work the creation of a CASE tool for analysis and design of information systems for decision-making, to facilitate the work of analysts, implementers and users to review automated structural metadata Bases existing relational data, and propose an optimal dimensional structure for the Vault. This software should be simple to use even for users who do not have all the knowledge and experience required on the subject.

KEY WORKS

Data Warehouse, Dimensional Model.

Índice

Introducción

Introducción

Con el gran desarrollo de la Informatización, crecen las ventajas para quienes tienen acceso a grandes volúmenes de datos, pero a su vez esto trae un gran problema ¿cómo manejar de manera exitosa estos volúmenes?

Hoy en día existen diversos tipos de sistemas de soporte para la toma de decisiones, pero el que ha tenido más auge a escala mundial en las grandes instituciones sin duda ha sido el *Data Warehouse* o Almacenes de Datos, convirtiéndose en el centro de atención de las organizaciones, puesto que provee un ambiente para hacer un mejor uso de la información administrada por diversas aplicaciones operacionales.

El almacén de datos no es más que una colección de datos orientada a un determinado ámbito (empresa, organización, etc.), integrado, no volátil y variable en el tiempo, que ayuda a la toma de decisiones en la entidad en la que se utiliza. Se trata, sobre todo, de un expediente completo de una organización, más allá de la información transaccional y operacional, almacenada en una base de datos diseñada para favorecer el análisis y la divulgación eficiente de datos. (Inmon, 2005)

En el caso de las empresas cubanas la introducción de tecnologías de la información y las comunicaciones en el manejo del conocimiento tiene como objetivo fundamental el uso más racional de los recursos, el logro de una mayor productividad y la obtención de los productos con una mayor calidad. Aunque no se cuenta con gran experiencia en el uso de los *Data Warehouse* si existen empresas que han tenido experiencias en su uso principalmente en el turismo y el comercio.

En Cuba se ha llevado a cabo un intenso proceso inversionista y de formación de personal, que permite afirmar que existen varios especialistas, no solo graduados de carreras afines a la informática, en diferentes organismos e instituciones dedicadas al desarrollo de sistemas de información, que trabajan o han trabajado en la implementación y explotación de almacenes o mercados de datos a partir de varios sistemas históricos de información y que después aplican diferentes herramientas

estadísticas y de búsqueda de información no revelada tradicionalmente en esos sistemas, para detectar tendencias industriales y comerciales, etc.

En la propia UCLV se han desarrollado varios trabajos aplicados a diferentes esferas, en calidad de tesis de maestrías y de diploma que abordan la creación de almacenes de datos. En todos se nota que el estudio de los metadatos o estructura de los sistemas fuentes se realiza de forma manual, y el analista o desarrollador propone un modelo dimensional (esquema estrella o esquema constelación de estrellas) adecuado para los requerimientos del usuario. A partir de ahí la mayor atención en el desarrollo siempre ha estado en la creación de las herramientas o software de limpieza, transformación y carga de los datos desde los diferentes sistemas de información al almacén; o en lo correspondiente a la explotación de este con la creación de informes generalmente de resúmenes variados, o proponiendo herramientas de creación de estas consultas o procedimientos que permiten generar dichas tablas o gráficos resultantes.

Es entonces que se considera necesario dar alguna solución al primer paso del trabajo con el almacén: el análisis automático de la estructura relacional existente y la propuesta automática de un modelo dimensional, que pueda ser adaptable a los intereses del usuario.

Situación Problémica:

El trabajo de creación de Almacenes de Datos en muchas ocasiones se torna un poco complicado puesto que no se poseen herramientas comerciales de análisis y diseño que generen un modelo dimensional adecuado para el almacén, sin un conocimiento previo de los metadatos y modelo relacional de los sistemas de información que se utilizarán como orígenes de los datos a incorporar al almacén.

Teniendo en cuenta esta dificultad antes expresada, se llegó a la conclusión de que se necesita una herramienta CASE que facilite el trabajo de analistas, implementadores y usuarios en cuanto al análisis automatizado de las Bases de Datos relacionales existentes y proponer una estructura estrella óptima para el Almacén.

Esta herramienta debe permitir el desarrollo visual de diagramas con un modelo estrella, a partir de estructuras Entidad-Relación de sistemas de información heredados, que tributarán datos al almacén, permitiendo generar la estructura de tablas de hechos y dimensionales, así como los procedimientos de carga del almacén.

Objetivo General:

Desarrollar una herramienta CASE que apoye el análisis y diseño de sistemas de información para la toma de decisiones (almacenes de datos), preferentemente basado en el modelo dimensional, y que permita generar propuestas de dicho modelo a partir de la revisión automática del modelo relacional de los sistemas de información heredados.

Objetivos específicos:

1. Estudiar las características propias del modelo dimensional, utilizado mayormente en la confección de Almacenes de Datos y establecer sus diferencias con respecto al Modelo Relacional en que se basan la mayoría de los sistemas de información actuales.
2. Desarrollar el análisis y diseño de sistemas del software a desarrollar en base al UML.
3. Implementar una herramienta CASE, basada en la metodología para diseño de almacenes de datos: “Modelado Conceptual de Almacenes de Datos a partir de Sistemas Operacionales basados en el modelo relacional”, del M.Sc. Lindsay Gómez Beltrán. Ofreciéndole al usuario un esquema conceptual.

Preguntas de Investigación:

1. ¿Cuáles son las diferencias sustanciales entre el modelo relacional y el modelo dimensional, que implican una transformación engorrosa al crear Almacenes de Datos?
2. ¿Se podrá implementar una herramienta CASE que detecte a través del análisis de los metadatos de un sistema de información heredado, y que aplique las pautas de conversión que se establezcan, para poder hacer propuestas automáticas adecuadas de partes del modelo dimensional para la implementación del almacén de datos?
3. ¿Podrá ser probado dicho software con varios formatos de Sistemas de Gestión de Bases de Datos?

Viabilidad de la Investigación:

Una de las temáticas que más desarrollo demanda en los momentos actuales en cuanto a las tecnologías de software es sin dudas la Ingeniería del Software, y dentro de esta, el desarrollo de métodos y algoritmos apropiados que conlleven a la creación y explotación de herramientas que asistan por medios computacionales al desarrollo de otros sistemas (herramientas CASE).

Dentro del campo de los Sistemas de Información, basados en la explotación de Sistemas de Bases de Datos, en los últimos tiempos se ha desarrollado la creación y explotación de Almacenes de Datos como uno de los típicos Sistemas de Ayuda a la Toma de Decisiones. Estos sistemas que por concepto manipulan información histórica recopilada en sistemas de información tradicionales de gestión empresarial, con el objetivo de descubrir nuevas informaciones que permitan avanzar en productividad y logros de todo tipo, se caracterizan entre otras cosas -según varios autores entre los que se pueden mencionar a Immon y Kimbal- por tener una estructura de tablas bastante diferente a la heredada de los sistemas tradicionales (casi siempre relacionales), conocida por Modelo Dimensional o Estrella.

La creación y explotación de Almacenes de Datos pasa entonces por varias fases a saber: Diseño y Creación del Almacén, Proceso de Carga y Limpieza de Datos, Aplicación de Métodos de Descubrimiento de Información (con métodos estadísticos o de minería de datos fundamentalmente), etc.

La primera etapa mencionada requiere hasta ahora del trabajo basado en la experiencia de los analistas de sistemas, pero que parte del estudio del o de los sistemas de información heredados, que ofertarán sus datos al almacén. No obstante hay varias investigaciones incipientes que tienden de una forma o de otra a automatizar de alguna manera esta labor. Para esto se hace necesario contar con las experiencias y hábitos de diferentes analistas dedicados en la práctica a la creación de almacenes y además con una bibliografía adecuada que aporte ideas sobre el desarrollo de esta etapa en la creación de los almacenes. Estas necesidades se cubren parcialmente ya que contamos con la metodología aportada por el M.Sc. Lindsay Gómez Beltrán en la cual nos basamos para realizar esta herramienta case, además de la experiencia en el tema del Dr. Rosendo de Jesús Moreno Rodríguez.

Tareas investigativas:

1. Realización de estudios sobre las técnicas y prácticas empleadas en el desarrollo de una herramienta CASE de Análisis y Diseño para Almacenes de Datos, así como de las características de los modelos relacionales y dimensionales de bases de datos.
2. Estudio de posibles tecnologías y herramientas programáticas a utilizar durante el proceso de desarrollo de una herramienta CASE.
3. Validación del software propuesta con un caso de estudio (sistema de información) determinado.

Hipótesis:

Con el desarrollo de una herramienta que facilite la labor de analistas y programadores en cuanto a proponer la estructura de un modelo dimensional para apoyar la creación de almacenes de datos a partir del modelo relacional de los sistemas de información legados, se logrará mayor productividad y garantía de calidad en la explotación de estos sistemas de toma de decisiones en diferentes tipos de empresas del país.

Estructura de la tesis:

Para lograr una adecuada organización de esta tesis, la misma se ha distribuido en tres capítulos que se refieren a:

- ✓ **Capítulo 1. Fundamentación Teórica**: Marco teórico relativo a los conceptos necesarios para el desarrollo de este trabajo incluyendo Sistemas de Información, Bases de Datos Relacionales, Almacenes de Datos, herramientas CASE, etc.
- ✓ **Capítulo 2. Análisis y Modelado de la Herramienta CASE**: Modelado con UML del software desarrollado.
- ✓ **Capítulo 3. Descripción de Características del CASE**: Breve explicación de las características y opciones de la herramienta CASE desarrollada para permitir su explotación.

Capítulo 1.

“Fundamentación Teórica”

Capítulo 1.”Fundamentacion Teórica”

En este capítulo se tratará todo lo relacionado al estado del arte sobre esta temática, se abordará sobre la metodología utilizada, de las aplicaciones que han servido para la implementación de la herramienta, y se exponen las características y los métodos más utilizados de los almacenes de datos, sus cualidades, ventajas, desventajas, así como, las herramientas de consultas y análisis.

El *Data Warehousing*, es el encargado de extraer, transformar, consolidar, integrar y centralizar los datos que la empresa genera en todos los ámbitos de su actividad diaria de negocios (compras, ventas, producción, etc.) o la información externa relacionada. Permite de esta manera el acceso y exploración de la información requerida, a través de una amplia gama de posibilidades de análisis multivariables, con el objetivo final de dar soporte al proceso de toma de decisiones estratégico y táctico.

1.1. Características del Data Warehouse.

El *Data Warehousing* posibilita la extracción de datos de sistemas operacionales y fuentes externas, permite la integración y homogeneización de los datos de toda la empresa, provee información que ha sido transformada y resumida, para que ayude en el proceso de toma de decisiones estratégicas y tácticas.

El *Data Warehousing*, convertirá entonces los datos operacionales de la empresa en una herramienta competitiva, debido a que pondrá a disposición de los usuarios indicados la información pertinente, correcta e integrada, en el momento que se necesita.

Pero para que el Data Warehousing pueda cumplir con sus objetivos es necesario, que la información que se extrae, se transforma y se consolida, sea almacenada de

manera centralizada en una base de datos con estructura multidimensional denominada **Data Warehouse** (DW).

Una de las definiciones más famosas sobre DW, es la de William Harvey Inmon, quien define: "*Un Data Warehouse es una colección de datos orientada al negocio, integrada, variante en el tiempo y no volátil para el soporte del proceso de toma de decisiones de la gerencia*" .

Debido a que W. H. Inmon, es reconocido mundialmente como el padre del DW, la explicación de las características más sobresalientes de esta herramienta se basaron en su definición.

Pero antes de pasar al siguiente epígrafe también debemos destacar dos conceptos de Kimball con respecto al modelo a utilizar: *"Los modelos entidad – interrelación, son un desastre para las consultas debido a que ellos no pueden ser comprendidos por los usuarios y ellos no pueden ser útilmente recorridos por el software del SGBD". "Los modelos entidad – interrelación, no pueden ser usados como bases para el almacén de datos empresarial"* . (Kimball et al., 2008)

Inmon defiende una metodología descendente (*Top-Down*) a la hora de diseñar un almacén de datos, ya que de esta forma se considerarán mejor todos los datos corporativos.

Ralph Kimball por su parte define los *Data Warehouse* como: "una copia de las transacciones de datos específicamente estructurada para la consulta y el análisis". También fue Kimball quien determinó que un DW no era más que: "la unión de todos los *Data Marts* (Mercados de Datos: versión especial de almacén de datos) de una entidad". Defiende por tanto una metodología ascendente (*bottom-up*) a la hora de diseñar un almacén de datos.

¿Qué es un Data Warehouse según Ralph Kimball ?

- ✓ Es una fuente de datos de la empresa que puede ser consultada.
- ✓ No debe ser organizada con ayuda del modelo entidad/interrelación.
- ✓ Es frecuentemente modificada, a partir de datos correctos.(Kimball and Ross, 2002)

Las definiciones anteriores se centran en los datos en sí mismos. Sin embargo, los medios para obtener y analizar esos datos, para extraerlos, transformarlos y cargarlos,

así como las diferentes formas para realizar la gestión de datos son componentes esenciales de un almacén de datos.

Un almacén de datos, es una base de datos con características de volumen y explotación distintas a la base de datos operacionales. Una BD operacional se diseña para soportar los procesos básicos de la organización mientras que un AD se diseña para hacer análisis de datos. Esta diferencia de uso, modifica significativamente las aproximaciones al diseño de ADs.

El problema básico del diseño de un AD consiste en obtener un conjunto de esquemas multidimensionales que permitan satisfacer los requisitos de análisis de los usuarios y que puedan ser mantenidos por las bases de datos operacionales existentes en la organización.(Sánchez)

Muchas referencias a un almacén de datos utilizan esta definición más amplia. Por lo tanto, en esta definición se incluyen herramientas para la inteligencia empresarial, herramientas para extraer, transformar y cargar datos (ETL) en el almacén de datos, y herramientas para gestionar y recuperar los metadatos.

Varias han sido las definiciones acerca del término de un Data Warehouse:

Bill Inmon, fue uno de los primeros autores en escribir sobre el tema en términos de las características del almacén de datos:

"Un almacén de datos es una colección de datos

- ✓ ***orientada a temas o materias,***
- ✓ ***integrada,***
- ✓ ***variable en el tiempo y***
- ✓ ***no volátil***

que se usa para el soporte del proceso de toma de decisiones gerenciales. "

Orientado a temas

Los datos en la base de datos están organizados de manera que todos los elementos de datos relativos al mismo evento u objeto del mundo real queden unidos entre sí.

Esta definición del DW clasifica la información en base a los aspectos que son de interés para la empresa. Dicha clasificación afecta el diseño y la implementación de los datos encontrados en el almacén de datos, debido a que la estructura del mismo difiere

considerablemente a la de los clásicos procesos operacionales orientados a las aplicaciones.

En síntesis, la ventaja de contar con procesos orientados a la aplicación, está fundamentada en la alta accesibilidad de los datos, lo que implica un elevado desempeño y velocidad en la ejecución de consultas, ya que las mismas están predeterminadas; mientras que en el DW para satisfacer esta ventaja se requiere que la información no esté normalizada, es decir, con redundancia, duplicidad de los datos y que la misma esté dimensionada, para evitar tener que recorrer toda la base de datos cuando se necesite realizar algún análisis determinado, sino que simplemente la consulta sea enfocada por vectores y variables que permitan localizar los datos de manera rápida y eficaz, para poder de esta manera satisfacer una alta demanda de complejos exámenes en un mínimo tiempo de respuesta.(Bernabue, 2007)

Integrado

La base de datos contiene los datos de todos los sistemas operacionales de la organización, y dichos datos deben ser consistentes.

La integración implica que todos los datos de diversas fuentes que son producidos por distintos departamentos, secciones y aplicaciones, tanto internas como externas, deben ser consolidados en una instancia antes de ser agregados al DW. A este proceso se lo conoce como Extracción, Transformación y Carga de Datos (ETL). (Bernabue, 2007)

Variante en el tiempo

Los cambios producidos en los datos a lo largo del tiempo quedan registrados para que los informes que se puedan generar reflejen esas variaciones.

Debido al gran volumen de información que se manejará en el DW, cuando se le realiza una consulta, los resultados deseados demorarán en originarse. Este espacio de tiempo que se produce desde la búsqueda de datos hasta su consecución es del todo normal en este ambiente y es, precisamente por ello, que la información que se encuentra dentro del depósito de datos se denomina de tiempo variable.

El intervalo de tiempo y periodicidad de los datos debe definirse de acuerdo a la necesidad y requisitos de los usuarios.

Es elemental aclarar, que el almacenamiento de datos históricos, es lo que permite al DW desarrollar pronósticos y análisis de tendencias y patrones, a partir de una base estadística de información, ya que las instantáneas son actualizadas de acuerdo con las actividades del negocio.(Bernabue, 2007)

No volátil

La información no se modifica ni se elimina, una vez almacenado un dato, éste se convierte en información de sólo lectura, y se mantiene para futuras consultas.

La información es útil para el análisis y la toma de decisiones solo cuando es estable. Los datos operacionales varían momento a momento, en cambio, los datos una vez que entran en el DW no cambian.

La actualización, o sea, insertar, eliminar y modificar, se hace de forma muy habitual en el ambiente operacional sobre una base, registro por registro, en cambio en el depósito de datos la manipulación básica de los datos es mucho más simple, debido a que solo existen dos tipos de operaciones: la carga de datos y el acceso a los mismos.

Por esta razón es que en el DW no se requieren mecanismos de control de la concurrencia y recuperación.(Bernabue, 2007)

1.2. Métodos más usados en la Construcción del Data Warehouse

Los modelos propuestos por William H. Inmon y Ralph Kimball para llevar a cabo el diseño de un Data Warehouse son los más aplicados en la actualidad, coincidiendo en que un *Data Mart* o un mercado de datos independiente no satisface las necesidades que tienen las compañías a escala corporativa de acceder inmediatamente y con facilidad a sus datos, pero sus criterios difieren en cuanto al modelo de datos y a las arquitecturas.

El término *Data Mart* es usado para designar a los almacenes de datos cuyo ámbito es más reducido, normalmente un departamento o área específica dentro de la empresa, es definido por Ralph Kimball como bodegas de datos con información de interés particular para un determinado sector de la empresa y aunque su enfoque sea para una sola perspectiva departamental, no lo exime de tener que seguir los lineamientos generales de implementación que posee el *Data Warehouse* (Kimball and Ross, 2002).

Ralph Kimball propone como modelo de datos al modelo dimensional, el más popular en las soluciones que se implementan de manera práctica, el cual facilita a los

usuarios finales las consultas y el análisis. Se caracteriza por ser sencillo de crear, extremadamente estable en presencia de cambios, además de mostrarse muy intuitivo y comprensible; el autor sugiere el uso de este modelo de datos para el desarrollo de los *Data Marts* y del *Data Warehouse* (Kimball and Caserta, 2004).

También William H. Inmon reconoce al modelo dimensional como el mejor para el desarrollo de los *Data Marts* por las ventajas brindadas, pero propone la construcción del *Data Warehouse* basado en el modelo entidad-relación. La idea de Inmon se basa en que el modelo entidad relación es mucho más rico y adaptable que el dimensional.

En cuanto a la arquitectura William H. Inmon en su libro "*Building the Data Warehouse*" (Inmon, 2005) plantea que la construcción del *Data Warehouse* no debe ser sustituida por la implementación de varios *Data Marts*.

Resaltando que la excusa para no desarrollar un almacén de datos la mayoría de las veces es por no contar con un gran presupuesto, la sustitución de este por los *Data Marts* trae desventajas puesto que están diseñados para un área particular de la empresa, lo que trae consigo diferencias entre las estructuras de datos de los mismos, que al integrarlos en el *Data Warehouse* algunos no serán reutilizables, ni flexibles, ni útiles para la reconciliación que se necesita. Inmon manifiesta que el proceso de construcción del *Data Warehouse* parte de los sistemas operacionales existentes, creándose áreas de diferentes temas, cuando existan una cierta cantidad de estas, el *Data Warehouse* inicia el proceso de población de las áreas de una manera integrada, una vez concluido se comienza a dar respuestas a las inquietudes de los usuarios; empezando así el florecimiento del nivel departamental a medida que se tienen más datos en el *Data Warehouse* y es en este punto del desarrollo cuando se centra la atención en las cuestiones de los diferentes departamentos, para definir y crear los almacenes de datos departamentales, los *Data Marts*.

Ralph Kimball en desacuerdo con la arquitectura propuesta por William H. Inmon resalta en su libro "*The Data Warehouse ETL Toolkit, Practical Techniques for Extracting, Cleaning, Conforming and Delivering Data*" (Kimball and Caserta, 2004), que los *Data Marts* están basados en los datos de la fuente y no en la visión departamental, en otras palabras que el *Data Mart* es sólo una parte de un producto orientado a la compañía, los cuales deben consistir en una continua pirámide de estructuras dimensionales idénticas, comenzando siempre con los datos atómicos.

Plantea también que la idea de construcción de un *Data Warehouse* centralizado no es realista, siendo más real construirlo en un ambiente descentralizado e incremental, porque las empresas están en constante cambio, adquiriendo nuevas fuentes de datos y necesitando nuevas perspectivas, propone además, centrarse en trazar estrategias adaptables e incrementales basándose en una idealista visión de controlar toda la información antes de construir el *Data Warehouse*. Por esta razón manifiesta que el proceso de construcción de un almacén de datos parte de los sistemas operacionales existentes, creando los diferentes *Data Marts* basados en la información de dichas fuentes, para luego de tenerlos desarrollados y funcionales se comience con la construcción del *Data Warehouse* basado en la información que éstos contienen.

En la actualidad este método es el más usado, gracias a las diferentes ventajas que proporciona, permitiendo a las empresas acometer los proyectos de manera separada y de esta forma reducir los efectos negativos que tendría fracasar en un intento por construir un *Data Warehouse*.

El funcionamiento de esta herramienta case está basado en la metodología para el 'modelado conceptual de almacenes de datos a partir de Sistemas Operacionales basados en el modelo relacional' del M.Sc. Lindsay Gómez Beltrán:

En la tabla 1 se muestran las pautas metodológicas de trabajo que se proponen, incluyen siete etapas o pasos, las seis primeras etapas se encuentran relacionadas una con otra, y tienen que cumplirse en el orden en que se presentan, y se recomienda una última etapa que interviene en todas las etapas anteriores como forma de mantener una relación integra del trabajo realizado por el diseñador y permite establecer una relación entre el diseño obtenido y las base de datos operacionales que intervienen en el diseño del almacén de datos.

En la concepción metodológica la obtención del modelo conceptual se corresponde con las cuatros primeras etapas de trabajo.

Tabla 1. Etapas de trabajo de la concepción metodológica propuesta.

Etapas	Entradas	Salida	Involucra
Análisis de los sistemas de información (heterogéneos)	Esquema de las bases de datos fuentes	Esquema refinado	Diseñador, gerente de sistema de información y usuario final.
Generación del árbol de dependencia de atributos.	Esquema refinado	Árbol de dependencia de atributos	Sistema
Generación de propuesta de modelo conceptual.	Árbol de dependencia de atributos, reglas de diseño.	Modelo conceptual temporal, hechos y dimensiones	Sistema, diseñador.
Refinamiento del modelo conceptual (estrella)	Modelo conceptual temporal, hechos y dimensiones	Modelo conceptual refinado	Diseñador, usuario final.
Diseño lógico	Modelo conceptual refinado, primitivas de transformación de esquema.	Esquema lógico del AD temporal.	Sistema
Refinamiento del modelo lógico.	Esquema lógico del AD temporal.	Esquema lógico del AD refinado	Diseñador
Trazabilidad	Información desde la primera etapa hasta la última.	Traza del modelo.	Sistema

Una vez recogida la información de las bases de datos de los sistemas operacionales, es importante tener en cuenta que en muchas ocasiones no toda la información provista por las base de datos operacionales es relevante para diseñar el almacén de datos, es por ello que el diseñador debe seleccionar qué información provista por la base de dato operacional es de interés para su diseño. Una vez seleccionadas las relaciones de interés para el diseño, no pueden existir relaciones aisladas. Esta condición es importante debido a que la información escogida será

utilizada para crear el árbol de dependencia de atributo y no tiene sentido tener alguna relación aislada.(Gomez Beltran, 2011)

La información es guardada en dos xml. El xml mapeo contiene las tablas, sus atributos, de cada atributo se tiene si es llave primaria, llave foránea y su tipo; en el xml relaciones se tienen todas las relaciones que hay entre las tablas. Se almacena esta información en los xml debido a que los cambios que el diseñador realice no pueden ser ejecutados en la base de datos operacional en caso de que los pueda hacer. Es importante señalar que en esta etapa no es permitido agregar ningún nuevo atributo ni tabla al diseño que no se encuentre en la base de datos operacional por que con esto devendría un cambio en la base de datos fuente.

Para la obtención de un modelo conceptual y llevar a este a un modelo en estrella, se propone como un paso intermedio, la obtención de un **árbol de dependencia de atributo**, en donde el diseñador tiene que definir una de las tablas de la base de dato como hecho y a partir de este se genera la dependencia de atributos.

El analista debe definir cuáles son los hechos, para cada uno se debe:

- Construir el árbol de dependencia de atributos.
- Podar y unir el árbol de atributos.
- Definir las dimensiones.
- Definir los atributos de hechos.
- Definir la jerarquía.

Después de generado el árbol de dependencia de atributos, se aplican las siguientes reglas:

Regla 1: Si se poda cualquier rama del árbol, los atributos recortados y la jerarquía de este no serán incluidos en el esquema de hechos.

Injertar se usa cuando, aunque un vértice del árbol expresa una información poco interesante, sus descendientes deben conservarse. Por ejemplo, el diseñador puede querer clasificar los productos directamente por la categoría, sin considerar la información sobre su tipo. En ese caso tendría que eliminar a tipo e injertar la categoría al nodo producto.

Regla 2: Si se quiere eliminar un vértice v, y este contiene nodos hijos en su jerarquía, se injertan los hijos de v de la jerarquía en el padre de v y se corta el nodo v.

Regla 3: Cada raíz del árbol de dependencia de atributos se convierte en un hecho y los atributos asociados a él pasan a ser atributos de la tabla de hechos.

Regla 4: Los nodos del árbol de dependencia de atributos que se encuentren más cercanos al nodo raíz y que constituyen llaves de alguna entidad son candidatos a convertirse en dimensiones y los atributos de éstas pasan a ser atributos de la dimensión.

Regla 5: Las llaves de las entidades involucradas pasan a ser llaves de las dimensiones y del hecho en cuestión.

Un punto Importante en la determinación de las dimensiones es el trabajo con la ***dimensión tiempo*** puesto que esta dimensión es una de las más importantes en un almacén de datos y generalmente no se encuentra de forma directa en los sistemas operacionales.

Regla 6: Determinar si existe algún atributo de tipo fecha en la tabla de hechos o en las dimensiones obtenidas. Desagregar el atributo de tipo fecha y convertirlo en una nueva dimensión.

Regla 7: Si no existe explícitamente el atributo de tipo fecha, pero el diseñador considera que éste se encuentra en la base de dato operacional, se le indicará al diseñador que seleccione qué atributo corresponde y su estructura si fuese necesario.

Regla 8: Si no existe ningún atributo que se pueda identificar como fecha, el diseñador debe de tomar la decisión de obtener los valores de la dimensión tiempo en el proceso de carga de los datos.

Existen nodos que conforman las jerarquías y que se encuentran asociadas a las dimensiones obtenidas en las reglas anteriores, se propone realizar un grupo de operaciones con la jerarquía que permita mejorar el diseño.

En ocasiones las jerarquías son necesarias agruparlas en la dimensión que tiene como raíz la jerarquía, en otros casos (menos frecuente), algunos de los nodos pasan a formar una nueva dimensión. A estas operaciones se le llama agrupamiento en la

dimensión y agrupamiento en una nueva dimensión. El diseñador tiene que determinar qué operación aplicar en cada nivel de la jerarquía.(Gomez Beltran, 2011)

Regla 9: Agrupamiento en la dimensión se le llama cuando un nivel de la jerarquía pasa a formar parte de la dimensión como un atributo más de ella.

Regla 10: Agrupamiento en una nueva dimensión se le llama cuando un nivel de la jerarquía constituye una nueva dimensión y el resto de la jerarquía conforma los atributos de la dimensión.

Regla 11: Las dimensiones que solo contienen los atributos llave pueden ser eliminadas si no aportan ninguna información relevante.(Gomez Beltran, 2011)

De manera muy concreta lo anteriormente expuesto es la metodología seguida para la creación de este **case**. De la cual hemos hecho una parte, las reglas que hemos aplicado al árbol de dependencias de atributos son: 3, 4, 5, 6, y 10.

Para versiones posteriores se recomienda que se tomen en cuenta todos los pasos y cada detalle expuesto en la metodología utilizada, esto devendría un mejor resultado. Es importante tener en cuenta que el analista juega un papel fundamental en la obtención del esquema conceptual, puesto que hay decisiones que solo él debe tomar.

1.3. Cualidades del Data Warehouse

Una de las primeras cualidades que se puede mencionar del DW, es que maneja un gran volumen de datos, debido a que consolida en su estructura la información recolectada durante años, proveniente de diversas fuentes, en un solo lugar centralizado. Es por esta razón que el depósito puede ser soportado y mantenido sobre diversos medios de almacenamiento.

Además, el almacén de datos presenta la información resumida y agregada desde múltiples versiones, y maneja información histórica. Organiza y almacena los datos que se necesitan para el procesamiento analítico e informático, con el propósito de responder a preguntas de negocios y brindarles a los usuarios finales una interfaz amigable, comprensible y fácil de utilizar, para que los mismos puedan tomar decisiones sobre los datos sin tener que poseer demasiados conocimientos informáticos. El DW permite un acceso más directo, es decir, la información gira en torno al negocio, y es

por ello que también los usuarios pueden sentirse cómodos al explorar los datos y encontrar relaciones complejas entre los mismos.

El DW no es solo datos, sino un conjunto de herramientas para consultar, analizar y presentar información, que permiten obtener o realizar análisis, reportes, extracción y explotación de los datos, con alta actuación, para transformar dichos datos en información valiosa para la organización. Sin embargo en lo referente a este trabajo nuestro centro de atención está precisamente en la estructura donde deben almacenarse los datos del almacén que después serán tratados por otras herramientas y con diversos fines.

Con respecto a las tecnologías empleadas, en un almacén de datos se pueden encontrar las siguientes:

- Arquitectura cliente/servidor.
- Técnicas avanzadas para replicar, refrescar y actualizar datos.
- Software *front-end*, para acceso y análisis de datos.
- Herramientas para extraer, transformar y cargar datos en el depósito, desde múltiples fuentes muy heterogéneas.
- Sistema de Gestión de Base de Datos (SGBD).

Cabe destacar, que todas las cualidades expuestas anteriormente, son imposibles de saldar en un típico ambiente operacional, y esto es una de las razones de ser del DW.(Bernabue, 2007)

Ventajas del Data Warehouse

- Transforma datos orientados a las aplicaciones en información orientada a la toma de decisiones.
- Integra y consolida diferentes fuentes de datos y departamentos empresariales, que anteriormente formaban islas, en una única plataforma sólida y centralizada.
- Provee la capacidad de analizar y explotar las diferentes áreas de trabajo y de realizar un análisis inmediato de las mismas.
- Permite reaccionar rápidamente a los cambios del mercado.
- Aumenta la competitividad en el mercado.
- Elimina la producción y el procesamiento de datos que no son utilizados ni necesarios, producto de aplicaciones mal diseñadas o ya no utilizadas.

- Mejora la entrega de información, es decir, información completa, correcta, consistente, oportuna y accesible. Información que los usuarios necesitan, en el momento adecuado y en el formato apropiado.
- Logra un impacto positivo sobre los procesos empresariales. Cuando los usuarios tienen acceso a una mejor calidad de información, la empresa puede lograr por sí misma: aprovechar el enorme valor potencial de sus recursos de información y transformarlo en valor verdadero; eliminar los retardos de los procesos empresariales que resultan de información incorrecta, inconsistente y/o inexistente; integrar y optimizar procesos a través del uso compartido e integrado de las fuentes de información; permitir al usuario adquirir mayor confianza acerca de sus propias decisiones y de las del resto, y lograr así, un mayor entendimiento de los impactos ocasionados.
- Aumento de la competitividad de los encargados de tomar decisiones.
- Los usuarios pueden acceder directamente a la información en línea, lo que contribuye a su capacidad para operar con mayor efectividad en las tareas rutinarias o no. Además, pueden tener a su disposición una gran cantidad de valiosa información multidimensional, presentada coherentemente como fuente única, confiable y disponible en sus estaciones de trabajo. Así mismo, los usuarios tienen la facilidad de contar con herramientas que les son familiares para manipular y evaluar la información obtenida en el DW, tales como: hojas de cálculo, procesadores de texto, software de análisis de datos, software de análisis estadístico, reportes, etc.
- Permite la toma de decisiones estratégicas y tácticas.(Bernabue, 2007)

Desventajas del Data Warehouse

- ✓ Requiere una gran inversión, debido a que su correcta construcción no es tarea sencilla y consume muchos recursos, además, su misma implementación implica desde la adquisición de herramientas de consulta y análisis, hasta la capacitación de los usuarios.
- ✓ Existe resistencia al cambio por parte de los usuarios.
- ✓ Los beneficios del almacén de datos son apreciados en el mediano y largo plazo. Este punto deriva del anterior, y básicamente se refiere a que no todos los usuarios

confiarán en el DW en una primera instancia, pero sí lo harán una vez que comprueben su efectividad y ventajas.

- ✓ Además, su correcta utilización surge de la propia experiencia.
- ✓ Si se incluyen datos propios y confidenciales de clientes, proveedores, etc., el depósito de datos atentará contra la privacidad de los mismos, ya que cualquier usuario podrá tener acceso a ellos.
- ✓ Infravaloración de los recursos necesarios para la captura, carga y almacenamiento de los datos.
- ✓ Infravaloración del esfuerzo necesario para su diseño y creación.
- ✓ Incremento continuo de los requerimientos del usuario.(Bernabue, 2007)

El almacenamiento de los datos no debe usarse con datos de uso actual. Los almacenes de datos contienen a menudo grandes cantidades de información que se subdividen a veces en unidades lógicas más pequeñas dependiendo del subsistema de la entidad del que procedan o para el que sea necesario. Los *Data Warehouse* surgen con la promesa del manejo y control de la información. Ellos aseguran una vista única de los datos, que pueden provenir de diversas fuentes. Gracias a esto, los usuarios finales no se ven en la necesidad de aprender y utilizar múltiples sistemas de acceso y manipulación de los datos. Un almacén de datos facilita la comprensión de los datos, transformándolos en información útil, teniendo como bandera el apoyo a la Toma de Decisiones. Permite no solo comprender lo que está pasando, sino predecir lo que va a suceder.

El objetivo del *Data Warehouse* es integrar datos corporativos, residentes en bases de datos operacionales de la organización, en un único repositorio sobre el cual los usuarios puedan realizar consultas o informes y hacer análisis de datos.

La tecnología de almacenes de datos integra las técnicas de bases de datos y las técnicas de análisis de datos.

Las bases de datos multidimensionales implican tres variantes posibles de modelamiento, que permiten realizar consultas de soporte de decisión:

1.4. Variantes de Modelación de Almacenes de Datos.

- ❖ Esquema en estrella (*Star Scheme*).
- ❖ Esquema constelación o copo de estrellas (*Starflake Scheme*).

Estos esquemas pueden ser implementados de diversas maneras, que, independientemente al tipo de arquitectura, requieren que toda la estructura de datos este desnormalizada o semi-desnormalizada, para evitar desarrollar uniones complejas para acceder a la información, con el fin de agilizar la ejecución de consultas. Los diferentes tipos de implementación son los siguientes:

- Relacional – ROLAP.
- Multidimensional – MOLAP.
- Híbrido – HOLAP.

Esquema en Estrella

El esquema en estrella, consta de una tabla de hechos central y de varias tablas de dimensiones relacionadas a esta, a través de sus respectivas claves, es un paradigma de modelado que tiene un solo objeto en el medio conectado con varios objetos de manera radial. Refleja la visión del usuario final de una consulta empresarial. Un esquema estrella lógico sencillo tiene una sola tabla de hechos y varias dimensiones conectadas a ella. En la figura que sigue se muestra un esquema estrella estándar:

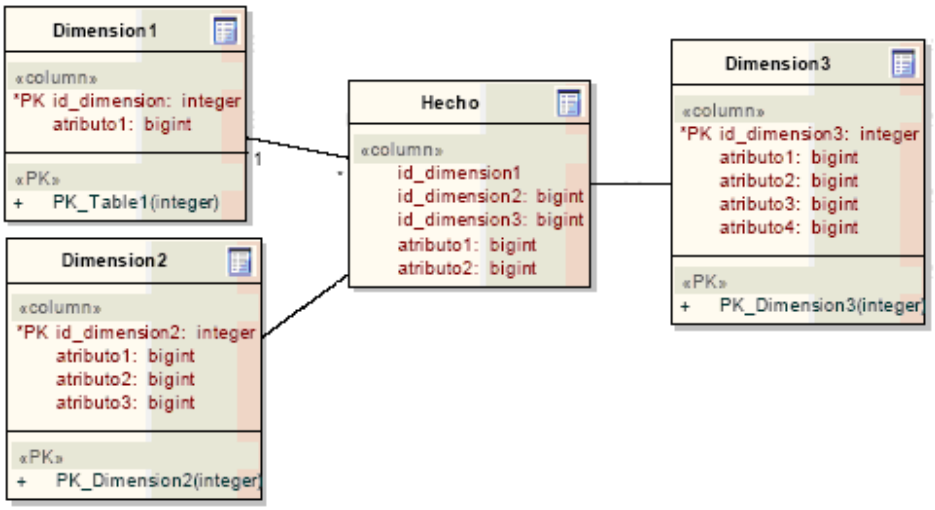


Figura 1. Esquema Estrella estándar

Características del Esquema en Estrella

- ✓ Posee los mejores tiempos de respuesta.
- ✓ Su diseño es fácilmente modificable.
- ✓ Existe paralelismo entre su diseño y la forma en que los usuarios visualizan y manipulan los datos.

- ✓ Simplifica el análisis.
- ✓ Facilita la interacción con herramientas de consulta y análisis.(Bernabue, 2007)

Esquema Constelación

Este modelo está compuesto por una serie de esquemas en estrella, y está formado por una tabla de hechos principal y por una o más tablas de hechos auxiliares, las cuales pueden ser resúmenes de la principal. Dichas tablas yacen en el centro del modelo y están relacionadas con sus respectivas tablas de dimensiones.

No es necesario que las diferentes tablas de hechos compartan las mismas tablas de dimensiones, ya que, las tablas de hechos auxiliares pueden vincularse con solo algunas de las tablas de dimensiones asignadas a la tabla de hechos principal, y también pueden hacerlo con nuevas tablas de dimensiones.

Su diseño y cualidad es muy similar a la del esquema en estrella, pero posee una serie de diferencias con el mismo. Entre ellas se pueden mencionar:

- Permite tener más de una tabla de hechos, por lo cual se podrán analizar más aspectos claves del negocio con un mínimo esfuerzo adicional de diseño.
- Contribuye a la reutilización de dimensiones, ya que una misma dimensión puede utilizarse para varias tablas de hechos.
- No es soportado por todas las herramientas de consulta y análisis.(Bernabue, 2007)

En la figura que sigue se puede apreciar un esquema de Constelación estándar:

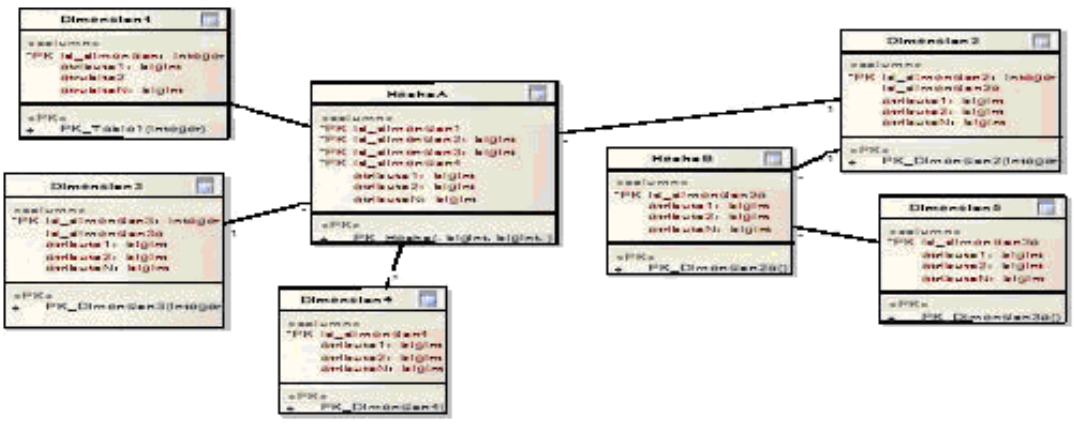


Figura 3. Esquema Constelación estándar

El esquema estrella, contiene una estructura de depósito de datos que se adapta mejor a los requerimientos y necesidades del usuario. Al contar con baja cantidad de dimensiones ofrece respuestas más rápidas a preguntas más complejas. Por lo hemos decidido basarnos en el mismo para el desarrollo de este trabajo. No obstante es de señalar que hay problemas reales donde varias tablas de hechos pueden encadenarse para dar una idea global de una cadena de gestión de información y esto se acomoda al esquema de constelación.

El modelo multidimensional dentro del entorno de las bases de datos, es una disciplina de diseño que se sustenta en el modelo entidad relación y en las realidades de la ingeniería de texto y datos numéricos.(Kimball, 1995)

Dadas las características de los almacenes de datos es ideal la utilización en su diseño de un Modelo Multidimensional (MMD). Este tipo de diseño tiene como ventajas sobre el Modelo Entidad-Relación (MER), que es muy flexible, está desnormalizado y orientado a los intereses de un usuario final, aunque esto no significa que existan inconsistencias en los datos. Mediante la utilización de un MMD se disminuye la cantidad de tablas y relaciones entre ellas, lo que agiliza el acceso a los datos.(Kimball, 1995)

El modelo multidimensional se representa a través de la definición de las tablas de hechos y dimensiones.

1.5. El Modelo Dimensional.

Una dimensión representa una perspectiva de los datos. Las dimensiones son usadas para seleccionar y agregar datos a un cierto nivel deseado de detalle. Podemos definir el concepto de dimensión como el grado de libertad de movimiento en el espacio. Entenderemos esta libertad como el número de direcciones ortogonales diferentes que podamos tomar.

Las dimensiones se relacionan en jerarquías o niveles. Por ejemplo, la dimensión Zona puede tener los siguientes niveles: ciudad, estado, región, país y continente.(Anónimo)

Las tablas de dimensiones definen como están los datos organizados lógicamente y proveen el medio para analizar el contexto del negocio. Representan los ejes del cubo, y

los aspectos de interés, mediante los cuales el usuario podrá filtrar y manipular la información almacenada en la tabla de hechos.(Bernabue, 2007)

Los talleres de sistemas de información, los usuarios finales, y aún las asociaciones de suministradores de datos como A. C. Nielsen, IRI, IMS, y Walsh América han gravitado caso por caso a una simple estructura de "cubo de datos" que se equipara a las necesidades de los usuarios finales por simplicidad. Esta estructura es en definitiva un modelo dimensional. Un Modelo Dimensional típico aparece en la figura 4.

Otro nombre para el modelo dimensional es el de esquema de enlace estrella debido a que los esquemas se parecen a una estrella, con una gran tabla central y un conjunto de pequeñas tablas acompañantes presentadas en un modelo radial alrededor de la tabla central.

A diferencia del modelo entidad –interrelación el modelo dimensional es muy asimétrico.

- ✓ Existe aquí una gran tabla dominante en el centro del esquema.
- ✓ Ella es la única tabla en el esquema con múltiples enlaces conectándola a otras tablas.
- ✓ Las otras tablas tienen todas, un enlace simple que las enlaza con la tabla central.

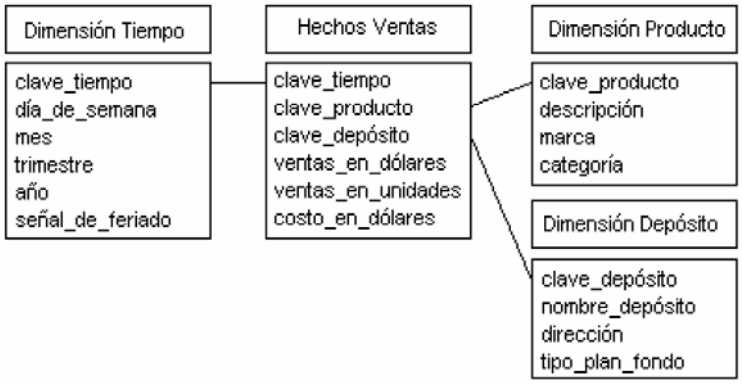


Figura 4. Modelo Dimensional típico

- ✓ Las tablas dimensionales por su parte son aquellas donde las descripciones textuales de las dimensiones del negocio son almacenadas. Cada una de las descripciones textuales ayuda a describir un miembro de la dimensión respectiva. En una base de datos bien diseñada, la tabla dimensional producto tendrá muchos atributos (campos). Sin embargo, en los ejemplos desarrollados

por varios autores como Kimball, resalta la existencia en casi todos los Almacenes de datos de las Dimensiones Tiempo y Localización (geográfica).

- ✓ Las tablas de hechos contienen los hechos, medidas o indicadores que serán utilizados por los analistas de negocio para apoyar el proceso de toma de decisiones. Los hechos son datos instantáneos en el tiempo, que son filtrados, agrupados y explorados a través de condiciones definidas en las tablas de dimensiones.(Bernabue, 2007)
- ✓ En la tabla de hechos es donde las mediciones numéricas del negocio son almacenadas. Cada una de las mediciones es tomada como la intersección de todas las dimensiones, los mejores y más útiles hechos son numéricos, valorados continuamente y aditivos.
- ✓ La razón para los hechos numéricos, valorados continuamente y aditivos, es que en cada consulta hecha contra esa tabla de hechos estamos preguntando por cientos, miles o millones de registros a ser usados por el SGBD para construir el conjunto respuesta.
- ✓ Este gran número de registros será comprimido a unas pocas docenas de filas del conjunto respuesta del usuario. La única forma útil de compactar estos registros en el conjunto respuesta es adicionándolos. Por tanto, si las mediciones son numéricas y si ellas son aditivas, podemos construir fácilmente el conjunto respuesta.
- ✓ En otros casos hay hechos que son semiaditivos, que pueden ser adicionados a lo largo de solo algunas dimensiones. También hay hechos que son no aditivos, los que simplemente no pueden adicionarse del todo. Para los hechos no aditivos, estamos obligados a usar contadores si se desea resumir los registros.

1.6. El Modelo Relacional y los Sistemas de Información.

El modelo relacional fue propuesto en primer lugar por E. F. Codd en su artículo original titulado '*A relational model of data for large shared data banks*'. Este artículo se acepta ahora generalmente como uno de los hitos principales en los sistemas de bases de datos, aunque anteriormente se había propuesto un modelo orientado a conjuntos. Los objetivos especificados del modelo relacional eran:

- ✓ Permitir un alto grado de independencia de los datos. Los programas de aplicación no deben verse afectados por las modificaciones efectuadas en la

representación interna de los datos, y en particular por los cambios efectuados en la organización de los archivos, en la ordenación de los registros o en las rutas de acceso.

- ✓ Proporcionar una base teórica sólida que permitiera tratar con la semántica de los datos y con los problemas de coherencia y de redundancia. En particular, el artículo de Codd introducía el concepto de relaciones normalizadas, es decir, relaciones en las que no haya grupos repetidos (el proceso de normalización incluye cinco formas normales, que no es objetivo explicar acá).
- ✓ Permitir la ampliación de lenguajes de manipulación de datos orientados a conjuntos.

Aunque el interés en el modelo relacional se manifestó desde múltiples direcciones, la investigación más significativa puede atribuirse a tres proyectos que tenían perspectivas bastante distintas. El primero de éstos, en el laboratorio *San José Research Laboratory* de IBM, en California, fue el SGBD relacional prototipo *System R*, que fue desarrollado a finales de la década de 1970. Este proyecto se llevó a cabo para demostrar la posibilidad de implementar el modelo relacional, desarrollando un ejemplo de sus estructuras de datos de operaciones. También demostró ser una fuente excelente de información acerca de problemas de implementación tales como la gestión de transacciones, el control de concurrencia, las técnicas de recuperación, la optimización de consultas, los problemas de seguridad e integridad de los datos, los factores humanos y las interfaces de usuario, y condujo a la publicación de muchos artículos de investigación en el desarrollo de otros prototipos. En particular, el proyecto *System R* condujo a dos desarrollos principales:

- ✓ el desarrollo de un lenguaje de consulta estructurado denominado SQL, que desde entonces se ha convertido en el] lenguaje estándar formal de ISO (*International Organization for Standardization*) y en el lenguaje estándar de facto para los SGBD relacionales;
- ✓ el desarrollo de varios productos comerciales de SGBD relacional a finales de la década de 1970 y principios de la de 1980, como por ejemplo DB2 y SQL/DS de IBM y Oracle de Oracle Corporation.

El segundo proyecto de importancia en el desarrollo del modelo relacional fue INGRES (*Interactive Graphics Retrieval System*, sistema gráfico interactivo de extracción) en la Universidad de California en Berkeley, que más o menos se desarrolló al mismo tiempo que el proyecto *System R*. El proyecto INGRES implicaba el desarrollo de un prototipo de SGBDR, concentrándose en la investigación en los mismos objetivos globales que el proyecto *System R*. Estas investigaciones condujeron a una versión académica de INGRES que contribuyó a la popularización de los conceptos relacionales y dio como resultado los productos comerciales INGRES de *Relational Technology Inc.* (ahora *Advantage Ingres Enterprise Relational Database* de *Computer Associates*) e *Intelligent Database Machine* de *Britton Lee lnc.*

El tercer proyecto fue el denominado *Peterlee Relational Test Vehicle* en el laboratorio del IBM *Scientific Centre en Peterlee*, Reino Unido. Este proyecto tenía una orientación más teórica que los proyectos *System R* e INGRES y su principal importancia radica en la investigación en cuestiones tales como el procesamiento y optimización de consultas y la ampliación funcional.

Los sistemas comerciales basados en el modelo relacional comenzaron a aparecer a finales de la década de 1970 y principios de la de 1980. Ahora hay centenares de SGBDR para entornas tanto *mainframe* como PC, aunque muchos de ellos no se adhieren estrictamente a la definición del modelo relacional. Como ejemplos de SGBDR basado en PC podemos citar *Office Access* y *Visual FoxPro* de *Microsoft*, *InterBase* y *JDataStore* de *Borland* y *R:Base* de *R:BASE Technologies*.

El modelo relacional está basado en el concepto matemático de relación, la cual se representa físicamente en forma de una tabla. Codd, que tenía formación matemática, utilizó terminología sacada del campo de las matemáticas, principalmente de la teoría de conjuntos y de la lógica de predicados. Un resumen de la terminología utilizada por el modelo relacional es:

- ✓ **Relación**: Una relación es una tabla con columnas y filas.
- ✓ **Atributo**: Un atributo es una columna nominada de una relación.
- ✓ **Dominio**: Un dominio es un conjunto de valores permitidos para uno o más atributos.
- ✓ **Tupla**: Una tupla es una fila de una relación.

- ✓ **Grado**: El grado de una relación es el número de atributos que contiene.
- ✓ **Cardinalidad**: La cardinalidad de una relación es el número de tuplas que contiene.
- ✓ **Base de datos relacional**: Una colección de relaciones normalizadas en la que cada relación tiene un nombre distintivo.

El uso de un SGBD facilita las características de control de redundancia, almacenamiento persistente de las características de los objetos y estructuras de datos, suministro de múltiples interfaces con los usuarios, cumplimiento de restricciones de integridad, así como también menor tiempo de creación de aplicaciones, flexibilidad y disponibilidad de información actualizada.

Cada SGBD posee intrínsecamente un lenguaje de definición de datos (LDD) y un lenguaje de manipulación de datos (LMD), lo que favorece el desarrollo de aplicaciones sin necesidad de complicadas estructuras de programación. Por otra parte con el desarrollo de sistemas de información basados en SGBD es posible implementar adecuadas interfaces de usuario basadas en menúes, formularios e informes principalmente, lo cual es una tendencia actual.

Los SGBDR modernos ofrecen entre otras facilidades, la de contar con un SQL empotrado, lo cual no es más que un conjunto de instrucciones que permite que el lenguaje estructurado de consultas sea utilizado con lenguajes de programación imperativos. A su vez, casi todos estos gestores incluyen dos variantes de creación de las consultas: a través del SQL directamente o a través de módulos de definición de consultas, basado en un lenguaje no procedimental, conocidos como QBE (*Query By Example*, consultas por ejemplo).

El Modelo Entidad - Interrelación (ER) fue introducido por Chen en 1976. En este modelo la información es representada por medio de tres conceptos primitivos:

- ✓ **Entidades**, las cuales representan los objetos a ser modelados.
- ✓ **Atributos**, los cuales representan las propiedades de dichos objetos.
- ✓ **Interrelaciones**, las cuales representan los vínculos entre entidades.

El Modelo Entidad-Interrelación es un modelo conceptual de alto nivel, usado frecuentemente para el diseño de aplicaciones de bases de datos.

El esquema ER correspondiente al diseño de la estructura de una base de datos es presentado de forma gráfica, donde las entidades se dibujan en forma de rectángulos, los atributos se dibujan en forma de elipses, y las interrelaciones se dibujan en forma de rombos.

En cualquier libro de texto sobre bases de datos y en varios artículos, se explica claramente este modelo semántico de diseño de bases de datos. En el desarrollo de un sistema de información para aceros, por ejemplo se deben definir como entidades entre otras las siguientes:

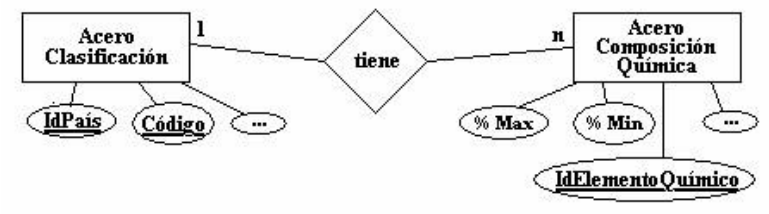


Figura 5. Modelo Entidad-Relación típico

Un Sistema de Información puede definirse técnicamente como un conjunto de componentes interrelacionados que permiten capturar, procesar, almacenar y distribuir la información para apoyar la toma de decisiones, la coordinación y el control en una institución.

Existen diferentes tipos de sistemas de información pero los más característicos para este trabajo son los sistemas de gestión de la información básica de las empresas generalmente clasificados como temporales pues manipulan información relativa a ventas, producciones, costos, etc., para un período de tiempo determinado; y los sistemas de apoyo a la toma de decisiones, que se basan sobre todo en datos permanentes que permiten hacer valoraciones especiales. Estos últimos se desarrollan actualmente mediante las técnicas de almacenes de datos.

1.7. Lenguaje de Modelado Unificado (UML).

El UML que surge a partir del Método de **Booch**, el Método **OOSE** (*Object-Oriented Software Engineering)*, Ingeniería del Software Orientada a Objetos) de *Jacobson* y el Método **OMT** (*Object Modeling Technique)*, Técnica de Modelado de Objetos) de *Rumbaugh*, es un lenguaje estándar para escribir planos de software. UML

puede utilizarse para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software.

UML es independiente del proceso, aunque para utilizarlo óptimamente se debería usar en un proceso que fuese dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental. En particular, UML cubre la especificación de todas las decisiones de análisis, diseño e implementación que deben realizarse al desarrollar y desplegar un sistema con gran cantidad de software.

UML es actualmente la herramienta más utilizada para el desarrollo del software en todo el mundo y está constituido en su primera versión por 9 diagramas (4 estructurales y 5 de comportamiento), mientras que en su versión 2 incluye 13 diagramas.

A los efectos del desarrollo de este trabajo, que consiste en crear una herramienta CASE o software de ayuda al análisis de sistemas relacionales y su adecuación a almacenes de datos, como quiera que era imposible desarrollar todo el software de una vez y acá se propone una primera versión, se estimó útil desarrollar el modelado del mismo con el uso del UML y en especial los diagramas de casos de uso y actores, clases, actividades y componentes o artefactos. Esto garantiza al menos a nivel del análisis y diseño un proceso evolutivo que además gracias a la herramienta a utilizar basado en el UML permitirá dar continuidad al software en versiones futuras, ampliando sus posibilidades.

Diagrama de Clases: Un diagrama de clases presenta un conjunto de clases, interfaces y colaboraciones, y las relaciones entre ellas, son los diagramas más comunes en el modelado de sistemas orientados a objetos y se utilizan para describir la vista de diseño estática de un sistema.

Diagrama de Casos de Uso: Los diagramas de casos de uso se emplean para modelar el contexto del sistema, subsistema o clase, o el modelado de los requisitos de comportamiento de esos elementos. Cada uno muestra un conjunto de casos de uso, actores y sus relaciones.

Diagrama de Actividades: Un diagrama de actividades es fundamentalmente un diagrama de flujo que muestra el flujo de control entre actividades, aunque a diferencia de estos puede mostrar concurrencias y ramas de control. Los diagramas de actividades se utilizan para modelar los aspectos dinámicos de un sistema, lo que generalmente

implica modelar los pasos secuenciales y concurrentes de un proceso computacional, y también para modelar el flujo de un objeto conforme pasa de un estado a otro en diferentes puntos del flujo de control.

Diagrama de componentes: Un diagrama de artefactos muestra la organización y las dependencias entre un conjunto de artefactos. Los diagramas de artefactos se utilizan para modelar la vista de implementación estática de un sistema. Esto implica modelar las cosas físicas que residen en un nodo, tales como ejecutables, bibliotecas, tablas, archivos y documentos.

1.8. Herramientas utilizadas.

Visual Paradigm :

El Visual Paradigm es una herramienta CASE, que permite el diseño detallado del software así como ingeniería inversa (obtención del diseño a partir del código fuente), basado en modelos con soporte UML. Es una forma de ayuda para la comprensión del sistema y de sus distintos componentes. Esta herramienta ha sido utilizada para la creación de los diagramas necesarios para la comprensión del sistema.

Java

Se lo conoce como lenguaje simple porque viene de la misma estructura de c y c++; ya que c++ fue un referente para la creación de java por eso utiliza determinadas características de c++ y se han eliminado otras. Entre sus características está:

- Orientado a Objeto: Toda la programación en java en su mayoría está orientada a objeto, ya que al estar agrupados en estructuras encapsuladas es más fácil su manipulación.
- Distribuido: Permite abrir sockets, establecer y aceptar conexiones con los servidores o clientes remotos; facilita la creación de aplicaciones distribuidas ya que proporciona una colección de clases para aplicaciones en red.
- Robusto: Es altamente fiable en comparación con c, se han eliminado muchas características con la aritmética de punteros, proporciona numerosas comprobaciones en compilación y en tiempo de ejecución.

- Seguro: La seguridad es una característica muy importante en java ya que se han implementado barreras de seguridad en el lenguaje y en el sistema de ejecución de tiempo real.
- Indiferente a la arquitectura: Java es compatible con los más variados entornos de red, cualesquiera sean estos desde Windows 95, Unix a Windows Nt y Mac, para poder trabajar con diferentes sistemas operativos. Java es muy versátil ya que utiliza byte-codes que es un formato intermedio que sirve para transportar el código eficientemente o de diferentes plataformas (Hardware - Software).
- Portable: Por ser indiferente a la arquitectura sobre la cual está trabajando, esto hace que su portabilidad sea muy eficiente, sus programas son iguales en cualquiera de las plataformas, ya que java especifica tamaños básicos, esto se conoce como la máquina virtual de java.
- Interpretado y compilado a la vez: Java puede ser compilado e interpretado en tiempo real, ya que cuando se construye el código fuente este se transforma en una especie de código de máquina.
- Multihebra o Multihilos:Java tiene una facilidad de cumplir varias funciones al mismo tiempo, gracias a su función de multihilos ya que por cada hilo que el programa tenga se ejecutaran en tiempo real muchas funciones al mismo tiempo.
- Dinámico: El lenguaje java es muy dinámico en la fase de enlazado, sus clases solamente actuaran en medida en que sean requeridas o necesitadas con esto permitirá que los enlaces se puedan incluir incluso desde fuentes muy variadas o desde la red.
- Produce Applets: En java se pueden crear aplicaciones independientes y applets. Independientes porque se pueden comportar como cualquier programa escrito en cualquier lenguaje. Por otra parte los applets considerados pequeños programas, tienen la capacidad de ejecutar funciones muy complejas.
- Alto rendimiento: Java es considerado de alto rendimiento por ser tan veloz en el momento de correr los programas y por ahorrase muchas líneas de código.

JDBC

JDBC para consultas en bases de datos. Para trabajar las bases de datos en JCreator se necesita una aplicación de API que es JDBC con el cual se pueden generar ingreso de datos y consultas creando un programa para esto, esto esta explicado a continuación.

Con la ayuda de JDBC, la habilidad de JAVA para integrarse con DBMS comerciales y su naturaleza orientada al manejo de la Red, es posible crear un ambiente ideal tipo cliente-servidor.

Al navegar en el World Wide Web, es fácil darse cuenta de que existe ya mucha información. Muchas compañías están usando bases de datos relacionales para manejar la información en sus sitios del Web. Por ejemplo, la mayoría de las máquinas de búsqueda usan este tipo de base de datos. Las bases de datos relacionales son ideales para el almacenamiento de grandes cantidades de información, la cual puede ser accedida por muchos usuarios.

Hoy en día, la mayoría de los gestores de bases de datos relacionales tienen soporte para la utilización de interfaces HTML. Para algunas aplicaciones, las páginas HTML favorecen al interfaz, pero, en aplicaciones más complejas se presentan ciertas limitaciones que no permiten generar un buen trabajo. En estos casos, es conveniente la utilización de lenguajes de programación como JAVA, que permitan elaborar aplicaciones para generar el interfaz con la base de datos.

La verdad es que todo lo que se pueda hacer en C++ se puede hacer con JAVA. Y mejor aún, al combinar JAVA con JDBC, se presentan nuevas expectativas para comunicarse con las bases de datos a través de un esquema similar al de las aplicaciones en C y C++. Muchos usuarios siguen confundidos en cuanto a la naturaleza de JAVA y piensan que es solo útil para animaciones y applets sencillos, para el Web. Vendedores de productos comerciales para bases de datos han puesto su atención en la nueva metodología de integración JAVA-DBMS. Una parte de este interés es impulsado por la posibilidad de obtener aplicaciones con una mejor imagen. Pero en la mayoría de los casos, estas compañías lo ven como una nueva oportunidad para que los programadores puedan tomar ventaja de esta tecnología en un futuro próximo. JDBC, es una API de JAVA para permitir ejecutar instrucciones SQL (Lenguaje estructurado de consultas), que es un lenguaje de alto nivel para crear, manipular, examinar y gestionar bases de datos relacionales.

Esto es lo que hace JDBC:

- Establece una conexión con una BD, que puede ser remota o no.
- Envía sentencias SQL a la BD.
- Procesa los resultados obtenidos de la BD.

JDBC es un API incluido dentro del lenguaje Java para el acceso a bases de datos. Consiste en un conjunto de clases e interfaces escritos en Java que ofrecen un completo API para la programación de bases de datos, por lo tanto es la una solución 100% Java que permite el acceso a bases de datos, la primera aparición de JDBC (JDBC 1.0) se encuentra dentro del paquete java.sql que ha fue incorporado en la versión del JDK. No será necesario escribir un programa para cada tipo de base de datos, una misma aplicación escrita utilizando JDBC podrá manejar bases de datos Oracle, Sybase, o SQL Server. Además podrá ejecutarse en cualquier sistema que posea una Máquina Virtual de Java, es decir, serán aplicaciones completamente independientes de la plataforma.

FUNCIONES DEL JDBC

- Establecer una conexión con una base de datos.
- Enviar sentencias SQL.
- Manipular los datos.
- Procesar los resultados de la ejecución de las sentencias.

CARACTERISTICAS DEJ JBDC

JDBC es independiente de la plataforma al estar escrito en Java. “JDBC es una API de bajo nivel ya que hace llamadas SQL directas”, Sun desea que JDBC pueda ser llamado desde otra API de más alto nivel que pueda simplificar la labor del programador, aunque la utilización de JDBC es sencilla y potente. Se tiene noticia de que ya existen diversos proyectos en marcha que intentan crear estas APIs de alto nivel. Aquí el término API hace referencia a un conjunto de clases e interfaces. Una forma de ver las características de JDBC es enfrentarlo con otro API que permita también el acceso a bases de datos, uno de los más usados y extendidos es el API de Microsoft ODBC (Open Data Base Connectivity).

ODBC permite la conexión a casi todo tipo de bases de datos en casi todas las plataformas, por lo tanto ¿por qué no podemos simplemente usar ODBC desde Java?. El

hecho es que se puede utilizar ODBC desde Java, pero a través de JDBC con lo que se denomina el puente JDBC-ODBC (JDBC-ODBC Bridge, desarrollado por Sun e Intersolv).

Importancia DEL JBDC

Usar ODBC directamente desde Java no es apropiado ya que usa un interfaz en C, y las llamadas desde Java a código nativo de C pueden ocasionar diversos problemas de seguridad y en la portabilidad de las aplicaciones.

Una traducción literal del API de ODBC escrito en C no es adecuado, ya que, Java no utiliza punteros y sin embargo ODBC hace un uso bastante frecuente de ellos. Se puede considerar que JDBC es una traducción de ODBC a un interfaz de programación orientada a objetos que es natural para los programadores de Java.

ODBC es más complicado de aprender, mezcla características sencillas con avanzadas y tiene opciones complejas incluso para las consultas más sencillas.

JDBC es necesario para disponer de una solución Java pura (100% puro Java). Cuando se utiliza ODBC el gestor de drivers y los drivers deben ser instalados manualmente en cada máquina cliente. Mientras que JDBC está escrito completamente en Java y su código se instala automáticamente.

ACCESO DE LA API JDBC A LAS BASES DE DATOS

La API JDBC soporta dos modelos distintos de acceso a las BD:

- Modelo de dos capas.
- Modelo de tres capas.

MODELO DE DOS CAPAS

En este modelo la aplicación JAVA o el Applet, se conectan directamente con la BD. Esto significa que el driver JDBC específico para conectarse con la BD estará instalado en el sistema local. La BD puede estar en otra máquina y se accede a ella mediante red. Esta configuración también se llama Cliente/Servidor. El programa

cliente envía instrucciones SQL a la BD, y esta las procesa y envía los resultados de vuelta al usuario.

MODELO DE TRES CAPAS

En este modelo, las instrucciones son enviadas a una capa intermedia que se encarga de enviar las sentencias SQL a la BD .El manejador de BD procesa las sentencias y retorna los resultados a la capa intermedia que se encarga de enviarlos al usuario

1.9. Conclusiones parciales.

En este capítulo se ha tratado todo lo relacionado sobre el tema objeto de estudio. Se ha expuesto la metodología usada en la creación del case. Se abordaron los conceptos y características principales de los almacenes de datos y del modelo dimensional que es más afín a estos, incluyendo los métodos más utilizados en la construcción de los mismos, sus cualidades y las variantes de modelado de estos.

A su vez se detallaron algunas consideraciones sobre los modelos relacionales y los gestores de datos, muy utilizados en el desarrollo y explotación de los sistemas de información tradicionales, que son la fuente de los datos que poblaran a los almacenes. Por otra parte se hizo un breve esbozo de algunos CASE usados en el modelado de bases de datos, de los cuales se consideraron algunas características típicas que debía cumplir el software que se presenta. También hace referencia al lenguaje de modelado del software utilizado (UML), especificando que diagramas se exponen en el capítulo correspondiente; y las herramientas que se usaron en este trabajo para la implementación del CASE propuesto.

Capítulo 2.

"Análisis y Modelado de la Herramienta Case Asistente-DW"

Capítulo 2. "Análisis y Modelado de la Herramienta CASE Asistente-DW"

En este capítulo se describen diversos requerimientos que debe cumplir la aplicación. También se planteará el análisis del sistema utilizando el Lenguaje Unificado de Modelado (UML), que permitirá representar diferentes diagramas.

2.1. Especificación de los requisitos de software.

Requerimientos funcionales

Un Requerimiento Funcional define el comportamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica. Los requerimientos funcionales que debe cumplir este sistema se describen como pautas a continuación:

- ✓ El software debe ser capaz de captar los metadatos de una base de datos (tablas, atributos, tipos de atributos, etc.) de diferentes SGBDR. En específico para esta versión se logra la implementación de la lectura de bases de datos en postgres SQL.
- ✓ El software se asegura de que todas las tablas que fueron seleccionadas están relacionadas.
- ✓ El software debe ser capaz de, a partir de una tabla propuesta como de hechos marcada, generar el modelo dimensional esquema estrella.

No fue posible brindarle al analista las tres tablas más promisorias a ser hechos, tampoco se logró que el analista pudiera podar e injertar en el árbol de dependencias de

atributos. En el trabajo de la jerarquía el usuario no tiene el control de las dimensiones que se están creando, puesto que se implementó de manera automática la regla 10.

Pero es necesario que se tenga en cuenta en versiones posteriores el esquema constelación de estrellas.

El software debe ser capaz de guardar de cierta forma el resultado alcanzado por el analista para su posterior uso como almacén de datos.

Requerimientos no funcionales

Los requerimientos no funcionales especifican propiedades o cualidades que el producto de software debe tener, como restricciones del entorno o de la implementación, rendimiento, dependencias de la plataforma y facilidad de mantenimiento.

En el presente trabajo se considera que el software debe ser de tipo monousuario, a ser explotada en una microcomputadora convencional INTEL compatible, a partir de un ejecutable, sin requerimientos especiales de rendimiento, velocidad.

Requerimientos de Seguridad

- La información que se encuentre en las bases de datos relacionales que se lean para estudiar sus características, no será revelada por el software, de manera que algún usuario-analista no pueda tener conocimientos que no le correspondan. Solo se analizarán los metadatos.
- La propuesta de clasificación de las tablas como dimensionales o de hechos, así como las adecuaciones que de forma controlada (manual) el usuario-analista establezca, será almacenada en un archivo XML.
- El uso y manejo del sistema se supone que sea controlado por el usuario-analista que lo explote. No se requieren otros niveles de seguridad específicos.

Requerimientos de Confiabilidad

- Todas las salidas del sistema tendrán 100% de precisión, acorde con los pasos establecidos o las decisiones del analista-usuario.

Requerimientos de Rendimiento

- No existen restricciones específicas de este tipo.

Requerimientos de apariencia o interfaz externa

- La interfaz será agradable, sencilla y sugerente como corresponde a una aplicación de escritorio, asumiendo por defecto el color establecido por el ambiente Windows del usuario-analista.
- Los textos que muestran las acciones a realizar en el menú o en la ventana central de resultados tendrán un tipo de letra y tamaño adecuados.

Requerimientos de Hardware

La PC utilizada debe tener como mínimo las siguientes características:

- Computador PENTIUM III o superior.
- 256 MB de memoria RAM o más (según exija el Sistema Operativo instalado).

Requerimientos de Software

- Las estaciones de trabajo clientes utilizarán como sistema operativo cualquier versión de Windows superior a Windows XP con MDAC (*Microsoft Data Access Components*) instalado.

Requerimientos de Soporte

- El sistema contará con una documentación apropiada para agilizar su explotación, mantenimiento y configuración.

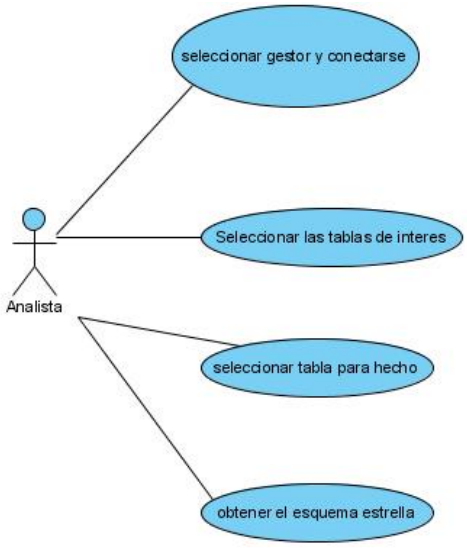
Requerimientos de Facilidad de Uso

- La interfaz será fácil de usar para los diversos usuarios que interactúen con ella.
- El sistema estará bien documentado, con el fin de lograr un mejor uso de los servicios que este ofrecerá, para ello se realizará una ayuda que explique paso a paso cada una de las funcionalidades del software.

Requerimientos legales

- Los módulos de desarrollo del proyecto *DW-asistente* así como la documentación del mismo, son propiedad de la UCLV y específicamente de la facultad de MFC.

2.2 Casos de uso.



Descripción de los Casos de Uso

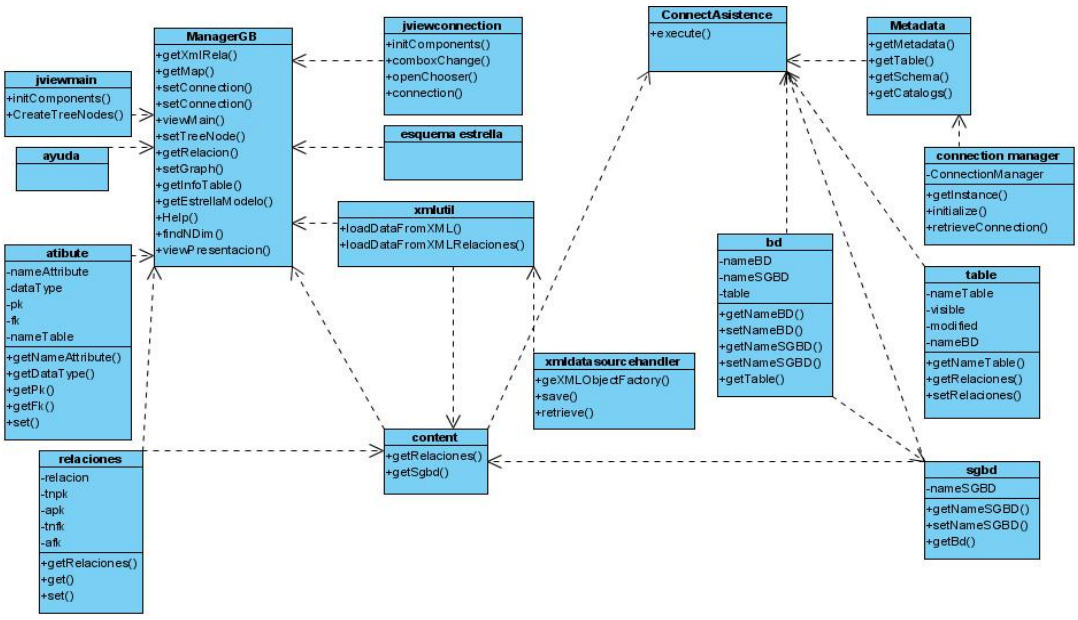
Nombre del CU	Elegir gestor de bases de datos y conectarse.
Actor(es)	Analista
Propósito	Seleccionar el gestor adecuado para la base de datos de la que queremos obtener información, para que el software obtenga los metadatos luego de conectarse.
Descripción:	El caso de uso se inicia cuando el actor selecciona el gestor, luego introduce el puerto, servidor, login, password según haga falta; o la dirección del Access. Después hace clip en aceptar.
Referencia	Es necesario tener instalado el proveedor correspondiente para cada Base de Datos
Precondiciones	- Tener el Data Access Component instalado(MDAC) - Que la Base de Datos sea válida.

Nombre del CU	Seleccionar las tablas de interés
Actor(es)	Analista
Propósito	El analista selecciona las tablas que le son relevantes.
Descripción:	El usuario visualiza todas las tablas y a través del botón de seleccionar elige las que desea.
Precondiciones	La acción de obtener los metadatos fue satisfactoria. El usuario puede ver las tablas.

Nombre del CU	Seleccionar un hecho
Actor(es)	Analista
Propósito	El analista selecciona el o los hechos que él quiera teniendo como referencia los hechos promisorios que le brinda el software.
Descripción:	El usuario selecciona de las tablas que el eligió la o las que desee.
Precondiciones	Debe haber seleccionado antes las tablas de interés y el software dado los hechos promisorios.

Nombre del CU	Obtener esquema estrella
Actor(es)	Analista
Propósito	Obtener del software el esquema conceptual
Descripción:	El usuario selecciona del menú modelo el submenú que corresponda con el esquema deseado.
Precondiciones	Debe haber seleccionado antes uno o más hechos promisorios

2.3 Clases.



En este diagrama se han representado las clases más significativas para la comprensión del funcionamiento del CASE, así como los métodos y atributos más importantes.

ManagerGB: En esta clase se vincula la interfaz gráfica con las operaciones de obtener los metadatos y el esquema dimensional estrella (hechos y dimensiones) .Trabaja con objetos de tipo jviewMain , jviewConnection, las clases Content, XMLUtil los métodos getRelaciones(),loadDataFromXMLRelaciones(),loadDataFromXML().Se usan además las clases SGBD , BD , Relaciones,Table. Se controla en esta clase la conexión y la información que se muestra al usuario.

ConnectAsistence: En esta clase se toman los metadatos necesarios y se almacenan en dos xml mapeo y relaciones .Usamos objeto de tipo DatabaseMetaData ,los métodos getMetadata, getCatalogs, getDatabaseProductName, getTable, getPrimaryKeys, getIndexInfo, getColumns,getImportedKeys y getCrossReference este último nos brinda las relaciones que existen entre todas las tablas.

```
XMLDatasourceHandler.save(c, new File("data/mapeo.xml"));
XMLDatasourceHandler.save(r, new File("data/relaciones.xml"));
```

Con esto guardamos los metadatos en los xml.

Metadata: en esta clase se implementan los métodos getCatalogs, getSchema, getTable, getMetadata. Con el uso de objeto tipo Connection conn : return conn.getMetaData ().getSchemas().getMetaData();

XMLUtil: En esta clase tomamos la información que guardamos usando loadDataFromXML(),un objeto tipo Content-(sc).

```
sc = XMLDatasourceHandler.retrieve(new File("data/mapeo.xml"));
sc = XMLDatasourceHandler.retrieve(new File("data/relaciones.xml"));
```

XMLDatasourceHandler: Esta clase es como un manejador, usamos objeto

ObjectFactory

Y el método geXMLObjectFactory.Importamos:

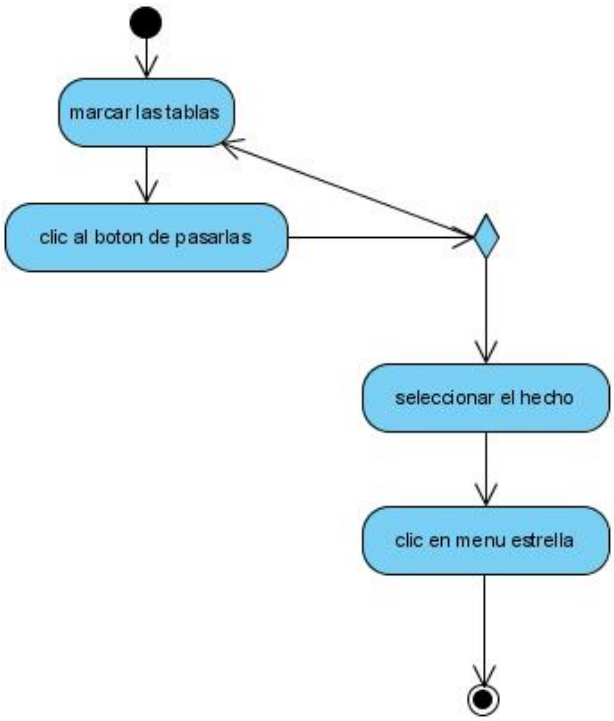
```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.OutputStream;

import javax.xml.bind.JAXBContext;
import javax.xml.bind.JAXBException;
import javax.xml.bind.Marshaller;
import javax.xml.bind.Unmarshaller;
```

Connection Manager: en esta clase se trata todo lo referente a la conexión.

Content: en esta clase se trabaja con los XML. Los objetos de tipo content c para el xml mapeo y r para xml relaciones son usados en la clase ManagerGB.

2.4 Actividades.

Este es el diagrama relacionado con los casos de uso: seleccionar las tablas, seleccionar el hecho y obtener esquema estrella.

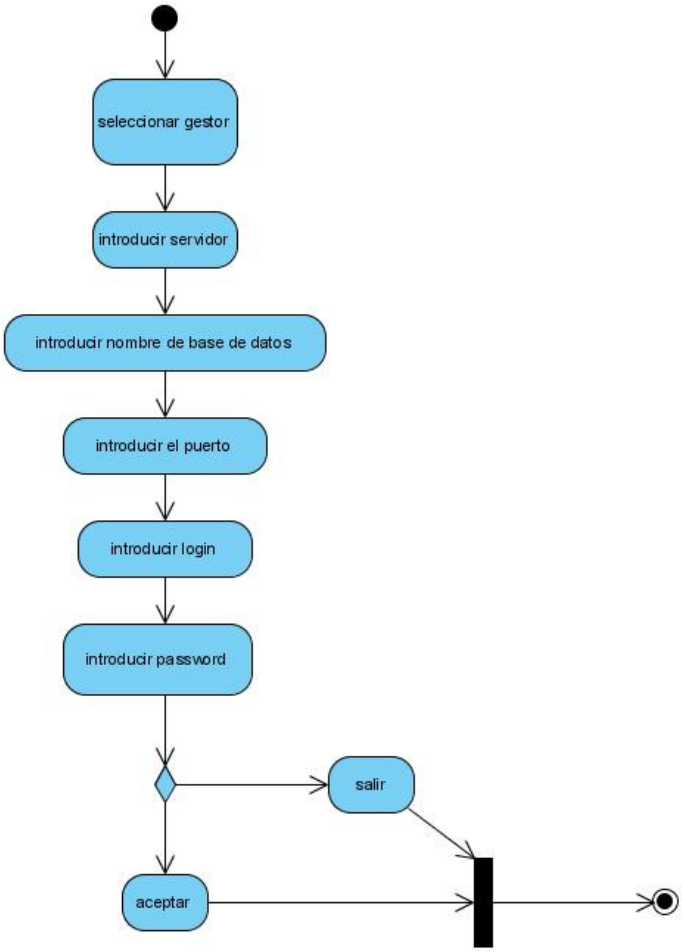


El analista marca todas las tablas que desea, luego hace clic en el botón de pasarlas a la derecha donde puede visualizar los atributos de cada tabla, de ellos su tipo, si es llave primaria y si es llave foránea .Si alguna tabla no fue seleccionada, puede hacerlo. Elige después el atributo candidato a ser considerado hecho y hace clic en el menú estrella para obtener el diagrama.

En el diagrama siguiente se hace referencia al caso de uso: elegir gestor de base de datos y conectarse. El usuario elige a cual gestor se va a conectar, introduce cual es el servidor al que se va a conectar, el nombre de la base de datos, el login y el password.

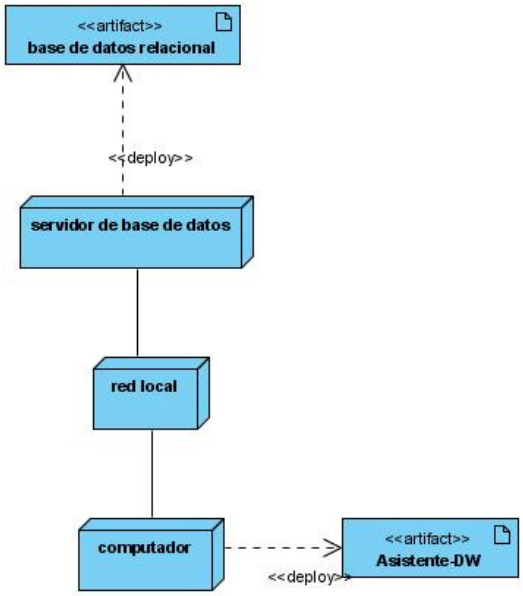
Para conectarse solo debe hacer clic en aceptar, si no desea continuar debe hacer clic en salir.

El diagrama siguiente es el relativo a la conexión.



Como indica el diagrama el usuario debe elegir el gestor postgres sql y luego introducir los datos necesarios para conectarse a la base de datos. Pulsando el botón de aceptar el software se conecta, extrae los metadatos, y los guarda en dos xml.

2.5 Despliegue.



En este esquema podemos apreciar el funcionamiento del software desde el punto de vista físico. Se necesita una computadora, el servicio de red en caso de que el servidor no sea la propia pc (local host).

2.6 Conclusiones Parciales.

En este capítulo se ha abordado todo lo referente a la modelación del software, mediante los diagramas uml se ha brindado la información necesaria para comprender el funcionamiento del software. Los requerimientos funcionales y no funcionales fueron expuestos esclareciendo como debe y que debe hacer el software. También queda claro mediante la descripción de los casos de uso que acciones puede hacer el analista.

Capítulo 3.

"Descripción de las Características del CASE."

Capítulo 3. "Descripción de las Características del CASE"

3.1Aspectos de la Implementación

El sistema diseñado tiene como nombre Asistente-DW, este tiene relación con su función fundamental de ayudar al analista en el diseño del *data warehouse*, extrayendo los metadatos de un sistema de información en modelo relacional, y posteriormente la creación de la estructura dimensional de un Almacén de Datos.

El software desarrollado como resultado de este trabajo fue programado en lenguaje java, utilizando el jdbc para la conexión y extracción de los metadatos.

El programa le permite al usuario seleccionar las tablas que son de su interés, luego verifica de manera automática que todas estén relacionadas, el usuario elige la tabla promisoria a ser hecho, y el programa le devuelve el esquema estrella asociado a su selección.

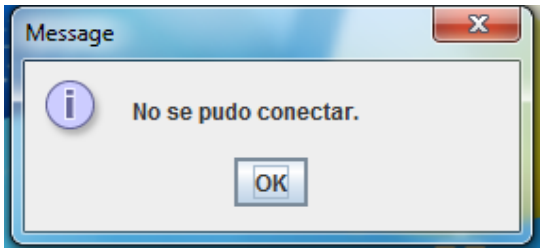
A modo de pasos seria:

- ✓ seleccionar BD, servidor, gestor — analista
- ✓ extracción de los metadatos — asistente-dw
- ✓ guardar datos en xml — asistente-dw
- ✓ mostrar datos al usuario — asistente-dw
- ✓ seleccionar las tablas — analista
- ✓ seleccionar un hecho — analista
- ✓ brindar esquema dimensional — asistente-dw

3.2Explotacion de la herramienta.

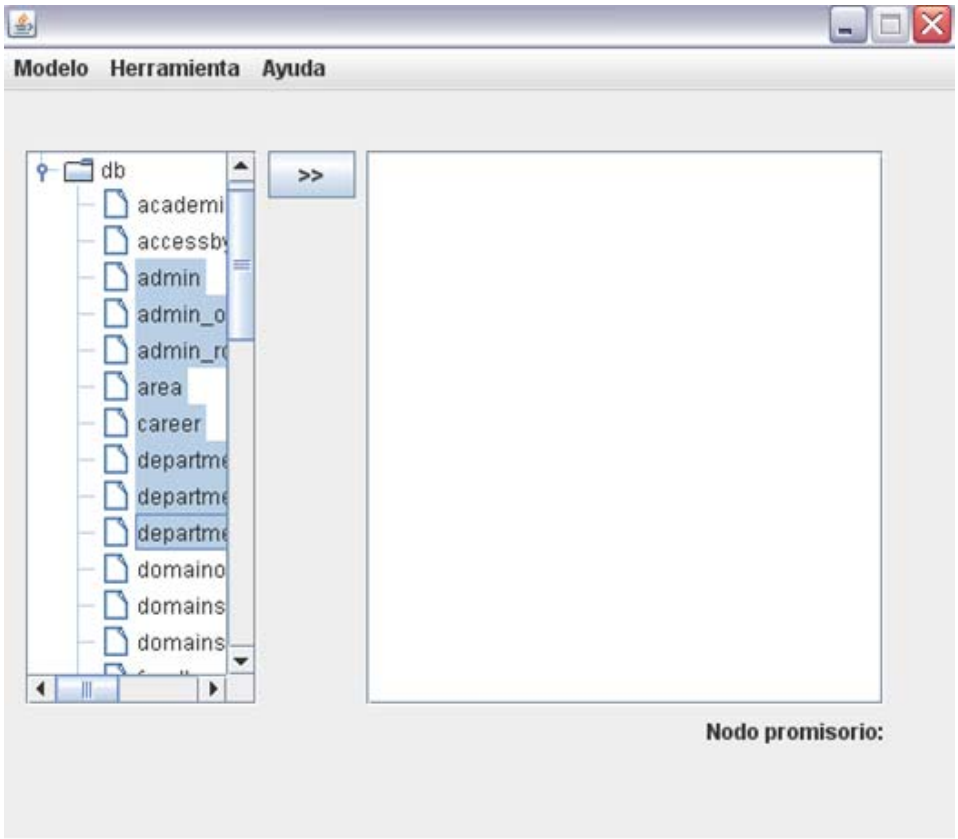
Esta es la ventana donde el usuario elige el gestor, la base datos y el servidor por el cual se va a conectar, para esta versión debe ser postgres.

Haciendo clic en el botón aceptar se pasa a la ventana principal si la conexión tuvo éxito, si no se muestra el siguiente mensaje:

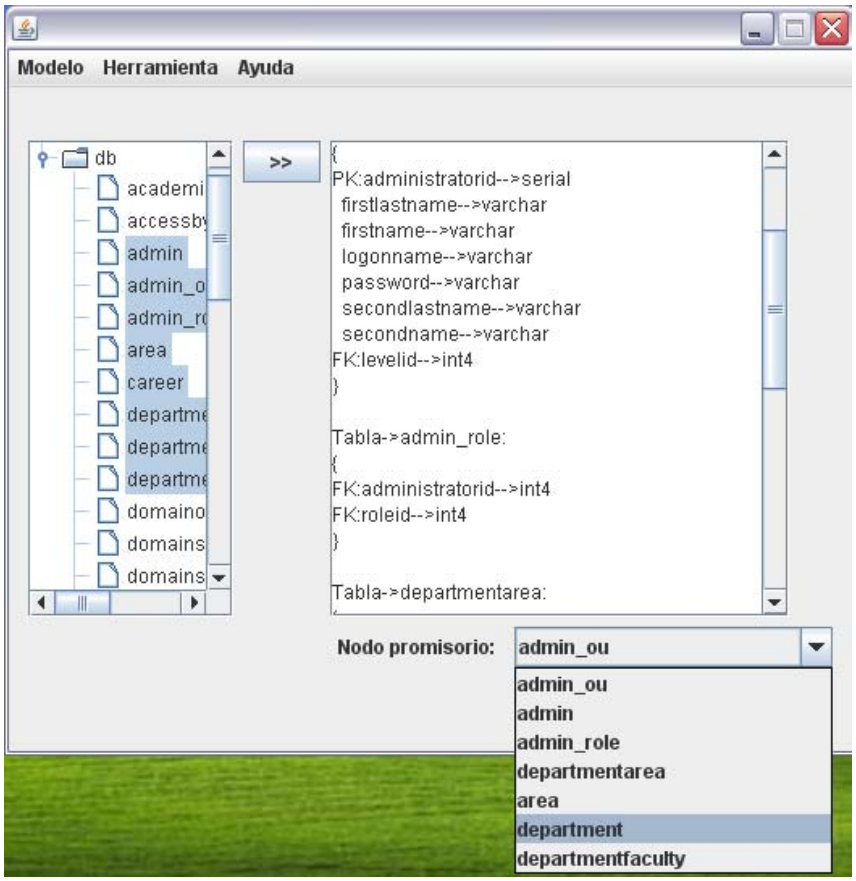


Si se hace clic en salir se cierra la ventana y termina el trabajo.

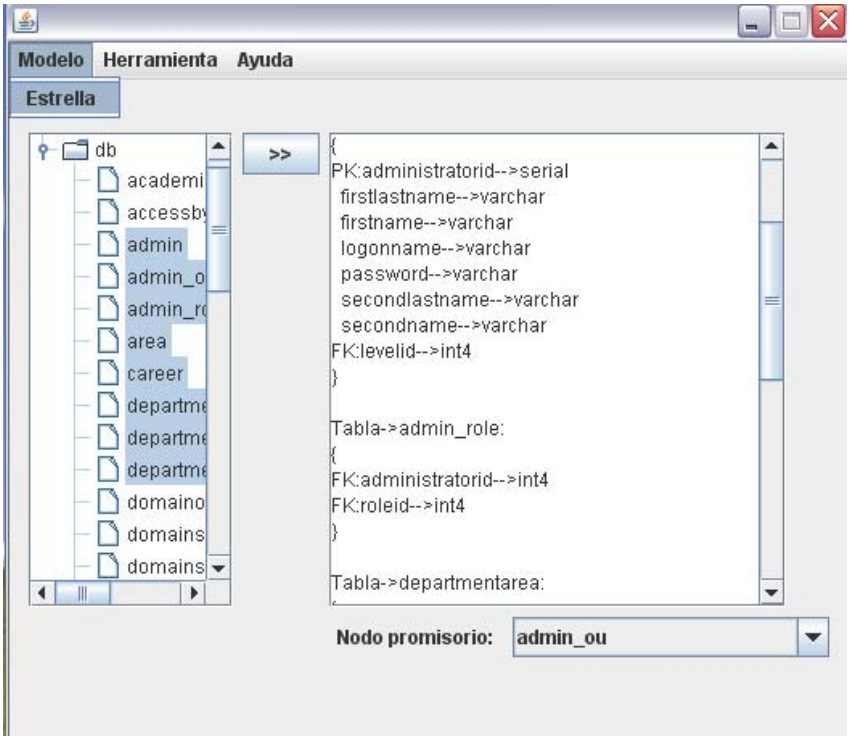
En esta ventana el usuario selecciona las tablas: marca todas las que le interesan y luego hace clic en el botón de pasar a la derecha. En el menú herramienta lo que se brinda es la posibilidad de volver a la ventana de conexión.



Aquí se muestra la acción de pasar las tablas, el software verifica que todas están relacionadas. Además el analista puede ver todos los atributos de estas tablas, su tipo si es primari key(PK) y si es foreng key(FK). El usuario escoge la tabla promisoria a ser hecho.



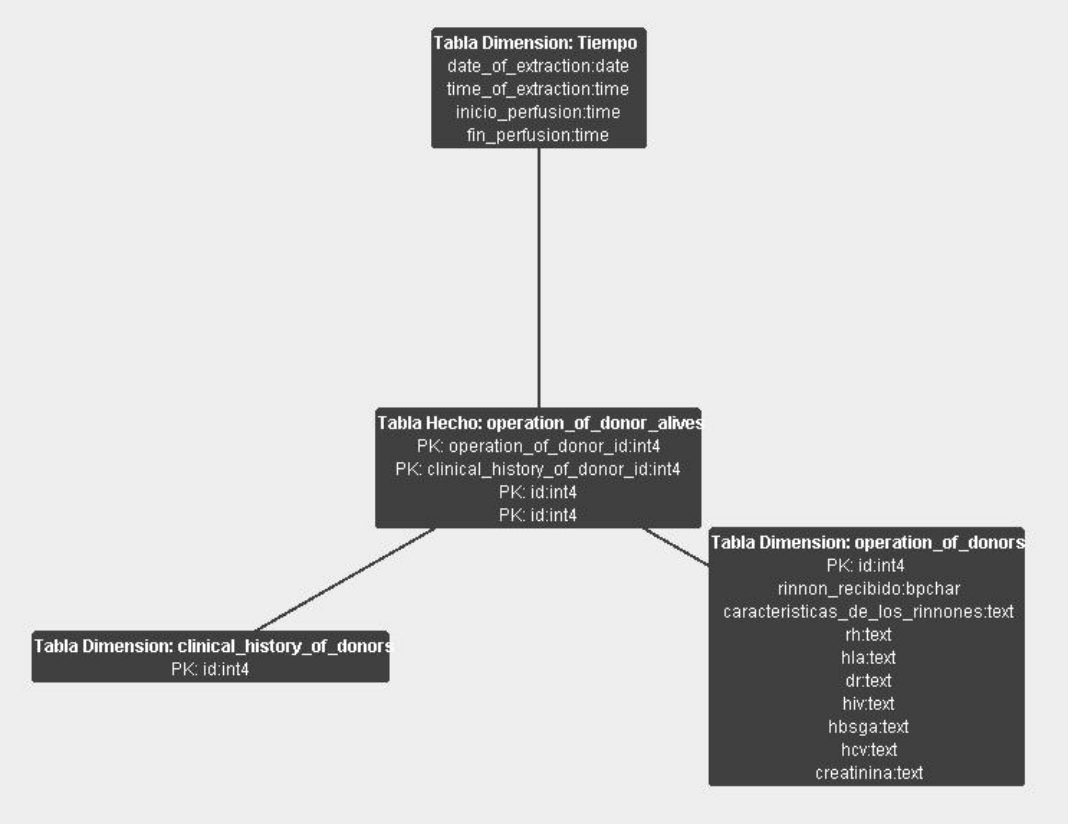
Luego se selecciona el menú estrella



Luego de elegida la tabla para hecho y haciendo clic en el menú estrella se obtiene el esquema conceptual que se muestra a continuación donde la tabla en el centro del diagrama es la de hechos y las demás que están conectadas a la misma son las dimensiones.

Este esquema conceptual es la razón de ser de este CASE, es el resultado de todo el trabajo del software. Si bien no es necesariamente el esquema conceptual definitivo para el analista, está cerca de la solución óptima aun cuando quedaron muchos aspectos a tener en cuenta en posteriores versiones.

Se puede observar en el esquema los atributos de cada tabla y su tipo, las llaves primaria y foránea.



3.3Conclusiones Parciales

En este capítulo se trató acerca de las características de la aplicación, dicha aplicación cuenta con una ventana inicial donde el usuario tiene la posibilidad de conectarse a una base de datos y automáticamente el software extrae los metadatos y los guarda en los xml .En la ventana principal el usuario puede ver las tablas de dicha base de datos y seleccionar las que considere importantes para el almacén ,luego puede visualizar de dichas tablas sus atributos así como su tipo, llaves primarias y llaves foráneas .Tiene la opción de seleccionar la tabla promisoria a ser hecho y seleccionar el menú estrella para obtener el esquema relativo a dichas tablas.

Conclusiones

Tras la terminación de este trabajo se puede concluir que:

1. Se estudiaron las características propias del modelo dimensional (esquemas estrella) utilizado mayormente en la confección de Almacenes de Datos y se establecieron sus diferencias con respecto al Modelo Relacional en que se basan la mayoría de los sistemas de información actuales, sobre todo en lo concerniente a la existencia de una gran tabla de hechos central que puede incluir información redundante, pero principalmente con datos mensurables y claves, y rodeada de tablas dimensionales relacionadas a partir de las claves. Se determinó como una característica esencial la existencia de las Dimensión Tiempo siempre que aparecieran atributos fechas en las tablas.
2. Se aplicaron las pautas metodológicas para llegar al esquema conceptual del almacén de datos definidas por el M.Sc. Lindsay Gómez Beltrán.
3. Se implementó una herramienta CASE (Asistente-DW) en su primera versión, que aplicando la metodología anteriormente especificada, presenta al usuario un esquema conceptual estrella .La cual le permite seleccionar las tablas de interés y la promisoria a ser hecho.
4. Se probó la explotación del CASE Asistente-DW tomando como caso de estudio las bases de datos de estudiantes investigadores del grupo de Bases de Datos del Centro de Estudios de Informática perteneciente a la facultad MFC, probando todas las opciones explicadas del sistema.

Recomendaciones

Se recomienda que en versiones posteriores se implementen otras funcionalidades que debe cumplir un software de este tipo. Estas son:

- ✓ establecer la conexión con otros gestores de base de datos (Oracle, sql server, my sql y Access), usando el jdbc.
- ✓ brindarle al analista los tres mejores candidatos a tabla de hecho.
- ✓ brindarle la opción de visualizar el árbol de dependencia para que pueda hacer operaciones como podar e injertar explicadas en el capítulo 2, además de decidir cómo trabajar con la jerarquía, en cada caso aplicando la regla 9 ó 10 según se quiera.
- ✓ brindarle al usuario el esquema conceptual Constelación de Estrellas.

Referencias Bibliográficas

ANÓNIMO. *Data Wareahousing* [Online]. Available: <http://porgramacion.com/bbdd/tutorial/warehouse/>.

BERNABUE, I. R. D. 2007. "Data Warehousing: Investigación y Sistematización de Conceptos – Hefesto: Metodología propia para la Construcción de un Data Warehouse". Córdova.

GOMEZ BELTRAN, L. 2011. Modelado Conceptual de Almacenes de Datos a partir de Sistemas Operacionales basados en el modelo relacional.pdf>.

INMON, W. H. 2005. Building the Data Warehouse - Fourth Edition. Wiley Publishing.

KIMBALL, R. 1995. "Fact Tables and Dimension Tables".

KIMBALL, R. & CASERTA, J. 2004. The Data Warehouse ETL Toolkit. Wiley Publishing, Inc.

KIMBALL, R. & ROSS, M. 2002. The Data Warehouse Toolkit Second Edition The Complete Guide to Dimensional Modeling. *In:* ELLIOTT, R. (ed.). Robert Ipsen.

KIMBALL, R., ROSS, M., THORNTHWAITE, W., MUNDY, J. & BECKER, B. (eds.) 2008. *The Data Warehouse Lifecycle Toolkit*: Wiley, 2a. Edición.

SÁNCHEZ, L. Z. Z. <Metodología para el Diseño Conceptual de Almacenes de Datos .pdf>.