

Universidad de las Ciencias Informáticas



Facultad 9

Título: Aplicación de Técnicas de Base de Datos para el soporte de indicadores dinámicos en el Sistema Automatizado para el Control de Gestión de Indicadores de Refinación (SACGIR).

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas.

Autor(es): Yaquelin Cintra Almaguer.

Yandy león Núñez.

Tutor: Lic. Yanet Espinal Martin.

Co-Tutor: Lic. David Silva Barrera.

Ciudad de La Habana, Junio del 2008

“Año 50 de la Revolución”

“Cada persona, en su existencia, puede tener dos actitudes: construir o plantar. Los constructores un día terminan aquello que estaban haciendo y entonces les invade el tedio. Los que plantan a veces sufren con las tempestades y las estaciones, pero el jardín jamás para de crecer.”

Paulo Coelho

DEDICATORIA

Le dedico este trabajo y con él todo el empeño que puse, a mi familia y muy particularmente a mi mamita querida, a mi papá, a mi mamá Paquita, a mi papá Juan, a mis hermanitas Lety, Yudy y Betsy, y a mis sobrinitas lindas Claudita y Danielita.

Yaquelin.

A mis queridos padres Caridad y Ramón que con tanto amor y sacrificio me han educado, apoyado y guiado a través de este camino intrincado y lleno de retos, que es la vida. A ellos que han sido, son y siempre serán mi principal escuela y que me inculcaron los valores humanos que han hecho de mí un hombre de bien, va dedicado este trabajo como muestra de mi amor, orgullo y agradecimiento.

Yandy

AGRADECIMIENTOS

Agradecer, con todo el amor del mundo:

A mi madre, que con su ejemplo y sabiduría ha sabido ganarse mi respeto y mi más profunda admiración;

A mi padre, por su constante preocupación por mi formación profesional;

A mi mamá Paquita, por ser mi segunda madre y quererme tanto;

A mi papá Juan, por ser un padre formidable y estar siempre pendiente de mí;

A mis hermanitas Lety y Yudy, por comprenderme, ayudarme y por ser las mejores hermanas del mundo;

A Luis Enrique, por sus atenciones y apoyo;

A Nena, por ayudarme en todo momento;

A Ana, Mailen y Frank, por siempre estar presentes y dispuestos;

A mis amigas y amigos del apartamento y grupo;

A mi novio querido, por ser mi sostén en los momentos más difíciles, ... con tu ayuda el camino se hizo mas corto;

Sepan todos que los quiero con la vida, acojan este trabajo, como parte de mi más sincero agradecimiento.

Yaquelin.

A mi madre y a mi padre por los años de dedicación, esfuerzo, cariño, preocupación y por el apoyo que siempre me brindaron. Muchas gracias por existir y estar presentes.

A mi hermana Yaomara por su constante preocupación y por estar pendiente de cada uno de los pasos importantes de mi vida a pesar de la distancia. Por darme aliento en los momentos más difíciles, por su cariño y su amor.

A mis amigos que son una parte muy importante en mi vida y que de alguna manera han contribuido en este trabajo, a los que han estado más cerca, Henry, Yoelvis, Ana, Mailen, Frank y a los que no lo están, Conrado, Yamil, Amaury.

A mi suegra Isabel por su apoyo incondicional y por acogerme como su familia en todos estos años.

A mi cuñada Leticia que ha sido como mi propia hermana preocupándose y apoyándome en cada momento.

A mis vecinos Migdalia, Elenita, Precedes y Araña que también son como mi propia familia y que siempre se han interesado por mis estudios y por mi bienestar.

A mi novia y compañera de tesis Yaquelin por la paciencia que ha tenido con mis caprichos, por su esfuerzo y dedicación durante la realización del trabajo, por haberme servido de guía e indicarme el camino correcto en cada situación. Por quererme y dejarme quererla.

Yandy

Agradecimientos Compartidos

Estaremos eternamente agradecidos a la Revolución cubana y al compañero Fidel, por crear un proyecto tan maravilloso como la Universidad de las Ciencias Informáticas; que nos ha dado la oportunidad de formarnos como personas de bien.

Agradecer a nuestra tutora Yanet, por ayudarnos cuando se hizo necesario y al profesor Adrian Vieyto, que también aportó conocimiento y experiencia al desarrollo de este trabajo.

DECLARACION DE AUTORÍA

Declaramos ser los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

Yaquelin Cintra Almaguer (Autor)

Yandy león Núñez (Autor)

Lic. Yanet Espinal Martin (Tutora)

Lic. David Silva Barrera (Co-Tutor)

DATOS DEL CONTACTO

Licenciada: Yanet Espinal Martin.

Categoría docente: Instructor

Universidad de Ciencias Informáticas, Habana, Cuba

E-mail: yanete@uci.cu

Licenciado: David Silva Barrera

Categoría docente: Instructor

Universidad de Ciencias Informáticas, Habana, Cuba

E-mail: dsilva@uci.cu

RESUMEN

Para PDVSA se hace indispensable, la gestión de indicadores, como principal fundamento para sus enfoques sistémicos y dinámicos. La implementación de dichos indicadores permite a la empresa la interpretación de la información generada por los sistemas pertinentes en función de la producción del crudo o cualquier otro factor que exista dentro de la misma.

Dichos indicadores estarán clasificados en dependencia de la función que desempeñen y al producto o parámetro al que respondan. Para incluir estos indicadores al sistema, se debe tener en cuenta que el diseño de la Base de Datos debe estar en función de los mismos, garantizando, que se pueda incluir un nuevo indicador, con los respectivos campos o atributos que conformen la fórmula para su posterior cálculo. Esta es precisamente una de las principales respuestas a las que se pretende dar solución con el presente trabajo: realizar el diseño de la Base de Datos de SACGIR de manera que admita la inclusión de nuevos indicadores, con sus respectivos campos. Además se hacen uso de técnicas de Base de Datos (como procedimientos y disparadores) que apoyan y garantizan este proceso.

Es por ello que constituye un elemento de vital importancia la implementación de nuevas funcionalidades a SACGIR, que unidas a un diseño adecuado de la Base de Datos, permitan dar soporte al cálculo de los indicadores de refinación, lo cual representa el objetivo fundamental de este trabajo.

Palabras Claves

Datos, Fórmula, Indicador, Técnicas de Bases de Datos.

ÍNDICE

INTRODUCCIÓN	1
CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA	6
1.1. INTRODUCCIÓN.....	6
1.2. CONCEPTOS ASOCIADOS AL DOMINIO DEL PROBLEMA	6
1.2.1. Conceptos Generales.....	6
1.2.1.1. Software Libre.....	6
1.2.1.2. Base de Datos	7
1.2.1.3. Sistema Gestor de Base de Datos	7
1.2.1.4. Sistema.....	7
1.2.2. Conceptos Particulares.	8
1.2.2.1. Indicador	8
1.2.2.2. Indicador de Refinación	8
1.2.2.3. Refinería	8
1.2.2.4. Refinación.....	8
1.3. OBJETO DE ESTUDIO.....	9
1.3.1. Descripción General.....	9
1.3.2. Descripción actual del dominio del problema	10
1.3.3. Situación Problemática.....	11
1.4. ANÁLISIS DE OTRAS SOLUCIONES EXISTENTES.....	12
1.4.1. Sistema de Información de Manufactura.....	13
1.4.1.1. Sistema Integrado de Movimiento de Petróleo	15
1.4.2. PhpMyAdmin	16
1.4.3. PhpPgAdmin	16
1.4.4. PhpMyAdmin y PhpPgAdmin	16
1.4.4.1. Elementos comunes con la investigación.....	17
1.4.4.2. Elementos no afines a la investigación.....	17
1.4.5. Base de Datos Relacional con Atributos y Registros Variables	18
1.4.5.1. Rasgos.....	19
1.4.5.2. Tipos de Rasgos.....	20
1.4.5.3. Variables Virtuales.....	20
1.4.5.4. Variables Elementales	21
1.4.5.5. Puesta de Trabajo	21
1.4.5.6. Perfiles de Trabajo.....	22
1.5. CONCLUSIONES.....	22
CAPÍTULO 2 TENDENCIAS Y TECNOLOGÍAS ACTUALES A DESARROLLAR	23
2.1. INTRODUCCIÓN.....	23
2.2. SISTEMAS GESTORES DE BASE DE DATOS	23
2.2.1. Oracle.....	23
2.2.2. MySQL	24
2.2.3. PostgreSQL.....	25

2.3.	PL/PGSQL.....	26
2.4.	METODOLOGÍAS DE DESARROLLO DE SOFTWARE.....	27
2.4.1.	RUP (Rational Unified Process)	27
2.4.2.	XP (Extreme Programing)	29
2.5.	UML (UNIFIED MODELING LANGUAGE)	30
2.6.	HERRAMIENTAS PARA EL MODELADO (HERRAMIENTAS CASE: COMPUTER AIDED SOFTWARE ENGINEERING)	31
2.6.1.	Visual Paradigm para UML	31
2.6.2.	Rational Rose.....	32
2.7.	HERRAMIENTA DE DESARROLLO SQL MANAGER 2005 FOR POSTGRESQL	32
2.8.	ANÁLISIS DE LAS HERRAMIENTAS SELECCIONADAS.	33
2.9.	TÉCNICAS DE BASE DE DATOS	35
2.9.1.	Data Warehouse	35
2.9.2.	Data Mart.....	37
2.9.3.	OLAP.....	39
2.9.4.	Data Mining	42
2.9.5.	Bases de Datos Activas	44
2.9.5.1	Triggers.....	45
2.10.	ANÁLISIS DE LAS TÉCNICAS SELECCIONADAS.....	47
2.11.	CONCLUSIONES.....	48
3.	CAPÍTULO 3 ANÁLISIS Y DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA	49
3.1.	INTRODUCCIÓN.....	49
3.2.	DESCRIPCIÓN DE LAS FUNCIONALIDADES.....	49
3.3.	INDICADORES, TIPOS DE INDICADORES, DEFINICIONES Y FÓRMULA	51
3.3.1.	Volumetría Mensual Calculada	51
3.3.2.	Seguridad Industrial	53
3.3.3.	Datos Mensuales Recursos Humanos	54
3.3.4.	Volumetría Mensual Manual.....	55
3.4.	DATOS PRIMARIOS DE LOS INDICADORES	58
3.4.1.	El Cálculo	58
3.4.2.	Tipos.....	59
3.5.	BDARV Y BASES DE DATOS ACTIVAS	59
3.6.	ESTRATEGIA DE INTEGRACIÓN DE BDARV CON SACGIR	60
3.6.1.	N_Tipo_Indicadors	62
3.6.2.	N_Indicadors	62
3.6.3.	Formulas	62
3.6.4.	Datos	63
3.6.5.	Dato_Formula.....	63
3.7.	APLICACIÓN DE TRIGGERS EN SACGIR	63
3.8.	DISEÑO DE LA BASE DE DATOS	65
3.8.1.	Diseño del MER del Módulo Indicadores	66
3.8.2.	Descripción de las Entidades de la Base de Datos.....	67
3.9.	IMPLEMENTACIÓN DE LAS FUNCIONALIDADES	70

3.9.1. Trigger Principal formula_tr	71
3.9.2. Funciones auxiliares utilizadas por la función ejecutar_trigger ().....	72
3.9.3. Función crear_procedimiento (varchar,varchar, text[], text[], varchar)	75
3.9.4. Función add_atributo(text, text, text)	80
3.10. CONCLUSIONES.....	80
CONCLUSIONES GENERALES.....	81
RECOMENDACIONES.....	82
REFERENCIAS BIBLIOGRÁFICAS	83
BIBLIOGRAFÍA.....	86
GLOSARIO DE TÉRMINOS.....	89

Índice de Tablas

TABLA #1. COMPARACIÓN ENTRE LA ESTRUCTURA DE BASE DE DATOS RELACIONAL Y LA TÉCNICA DE BDARV.	19
TABLA #2. FUNCIONALIDADES QUE EL SISTEMA DEBE CUBRIR.	50
TABLA #3. COMPARACIÓN ENTRE LA ESTRUCTURA DE BASE DE DATOS RELACIONAL Y LA TÉCNICA DE BDARV PARA SACGIR.	62
TABLA #4. DESCRIPCIÓN DE LA ENTIDAD N_INDICADORS.	67
TABLA #5. DESCRIPCIÓN DE LA ENTIDAD N_TIPO_INDICADORS.	68
TABLA #6. DESCRIPCIÓN DE LA ENTIDAD FORMULAS.	69
TABLA #7. DESCRIPCIÓN DE LA ENTIDAD DATOS.	70
TABLA #8. DESCRIPCIÓN DE LA ENTIDAD DATO_FORMULA.	70

Índice de Figuras

FIGURA #1. FASES E ITERACIONES DE LA METODOLOGÍA RUP.	28
FIGURA #2. METODOLOGÍA EXTREME PROGRAMING.....	29
FIGURA #3. LOGO DE UML.	30
FIGURA #4. ESQUEMA DE UN DATA WAREHOUSE.	36
FIGURA #5. ESTRUCTURA DE DATOS DE TRES DIMENSIONES.....	40
FIGURA #6. ESQUEMA ESTRELLA.....	41
FIGURA #7. DIMENSIONES Y JERARQUÍA.	42
FIGURA #8. RELACIÓN ENTRE DATO, INFORMACIÓN Y CONOCIMIENTO.	43
FIGURA #9. ESTRUCTURA DE UN TRIGGER.	46
FIGURA #10. MER PARA EL MÓDULO INDICADORES DE SACGIR.....	66

Introducción

PDVSA, una industria vital para Venezuela como quinto mayor productor de petróleo a nivel mundial, incluye, dentro sus gestiones y como enfoques modernos de la Gerencia Operacional y la Gestión de las Áreas Funcionales, el desarrollo de una nueva visión de las mismas, la cual incluye el principio holístico, los procesos, las operaciones flexibles y los enfoques sistémicos y dinámicos.

Esta visión se debe acompañar de un sistema de indicadores, los cuales de manera dinámica permitan a la empresa conocer los estados de desarrollo de sus diferentes estrategias de las Operaciones (de bienes y servicios), planes y procesos, como también las múltiples interrelaciones que permanentemente se están generando al interior de la misma, exigiendo agilidad, transparencia, flexibilidad y sencillez en la interpretación de la información para la toma de decisiones ejecutivas en dicha Operación.

Entre diciembre del 2002 y enero del 2003, tiene lugar un sabotaje cometido contra la petrolera. A raíz del paro petrolero, las operaciones diarias y mensuales de PDVSA se vieron en peligro. Por otra parte, el sistema por el que PDVSA rige sus operaciones, no cumple con todos los requerimientos que son presentados en el negocio actual y el Gobierno Nacional Venezolano orienta a todas las empresas gubernamentales a migrar sus plataformas tecnológicas al software libre y al empleo de estándares abiertos. Por este motivo la petrolera se ve obligada a plantearse la ruta camino a la migración al software libre; lo que le impone proponerse un nuevo sistema, que cumpla con los requerimientos del negocio, y en esencia, queden automatizados los diferentes procesos de acuerdo a las necesidades de la gestión de refinación, como puede ser, la flexibilidad del sistema a la hora de agregar las nuevas informaciones para el cálculo de un indicador.

En PDVSA, los indicadores de refinación permiten conocer el comportamiento de la empresa ante un determinado factor, por lo que son un punto clave en la evaluación y control de los procesos de negocio, posibilitando plantear y planificar la futura evolución de la petrolera; he ahí precisamente la importancia de que los datos básicos para la construcción del indicador sean de fácil obtención. Estos indicadores se conforman de forma dinámica, pues su composición está determinada por diversidad de datos e informaciones almacenadas en la Base de Datos y que puede variar en determinado período de tiempo. Esto quiere decir que todos ellos están sujetos a transformaciones y cambios futuros dependiendo del análisis del personal encargado de su seguimiento y control, pero ¿Qué

pasaría si se determinara que cierto indicador debería estar compuesto por un dato que no se encuentra físicamente en la Base de Datos, o si se crea un nuevo indicador con información que el sistema no maneja de forma persistente? Para resolver este problema se tendría que hacer una reestructuración de la Base de Datos y adaptarla de manera que pueda soportar la incorporación de esta nueva información que necesitan los indicadores, creando nuevas entidades o agregándole nuevos atributos a las ya existentes. Por consiguiente este proceso llevaría consigo nuevos gastos de recursos para PDVSA, ya que se tendrían que hacer estas reestructuraciones cada vez que tenga lugar el problema, resultando esta la solución menos factible para resolver la situación; lo que sin duda alguna da lugar al **problema** que llevó a este trabajo de Diploma, y es que: SACGIR no cuenta con funcionalidades que permitan la integración al sistema, de información que no esté almacenada en el mismo y que sea necesaria para el cálculo de los indicadores.

Con vista a la solución del problema planteado, se ha propuesto como **objeto de estudio**: SACGIR.

Al mismo tiempo, se ha propuesto, implementar nuevas funcionalidades en el sistema SACGIR, de forma tal que permitan la incorporación de los datos necesarios para el cálculo de los indicadores de refinación, representando este, el **objetivo general**, que se persigue con esta investigación.

En este contexto, quedarán definidos entonces, los siguientes **objetivos específicos**:

- ✓ Determinar las técnicas de Base de Datos apropiadas, para la incorporación de los datos necesarios para el cálculo de los indicadores.
- ✓ Realizar un estudio de los indicadores de refinación para conocer los datos que componen la fórmula con que se calculan y poder definir su forma de almacenamiento.
- ✓ Implementar las funcionalidades que permitan la incorporación de los datos necesarios para el cálculo de los indicadores.

Para un mayor control y evaluación del proceso investigativo, y además, para garantizar el cumplimiento de los objetivos propuestos, se han trazado las siguientes **tareas**:

- ✓ Estudiar el sistema SACGIR.
- ✓ Revisar documentación relacionada con técnicas de Base de Datos.
- ✓ Definir las nuevas funcionalidades que se van a implementar.
- ✓ Estudiar el estado del arte referente a aplicaciones que implementen funcionalidades semejantes.

- ✓ Diseñar el MER a partir de la incorporación de estas nuevas funcionalidades.
- ✓ Implementar dichas funcionalidades.

Habiendo definido las situaciones particulares de nuestro objetivo general, y las tareas que darán cumplimiento a los mismos, se puede dirigir la investigación, en esencia, hacia las nuevas funcionalidades que permitan la integración al sistema SACGIR de información para el cálculo de los indicadores, quedando así establecido nuestro **campo de acción**.

A partir del problema anteriormente expuesto se plantea que la implementación de nuevas funcionalidades en el modelo de datos del sistema SACGIR, de manera que este sea capaz de incorporar los datos que se necesiten para el cálculo de los indicadores de refinación, garantizará la flexibilidad del mismo ante nuevos cambios o incorporación de información, y se logrará que sea capaz de incluir nuevos indicadores, resultando esta idea la **hipótesis** de la investigación.

Para facilitar la construcción de modelos e hipótesis de investigación, se utilizan **Métodos Científicos Teóricos**, tales como:

- ✓ *Hipotético-Deductivo*, con el objetivo de inferir conclusiones y establecer predicciones a partir de los conocimientos que se poseen, referentes a Base de Datos.
- ✓ *Histórico- Lógico*, para verificar si hay alguna funcionalidad implementada, que incluya las técnicas necesarias para llevar a cabo la gestión que se propone en el negocio del sistema, para los indicadores. De esta forma se podrá estudiar dicha funcionalidad y hacer un análisis de los resultados obtenidos luego de su aplicación.
- ✓ *Analítico- Sintético*, pues se hace necesario a partir de un previo análisis de las posibles técnicas de bases de datos a implementar, establecer cuáles de ellas serían las apropiadas a partir de sus características.
- ✓ *Modelación*, propiciando facilidades para el estudio y planteamiento de la realidad expuesta, a través del Modelo de Objetos y el MER.

Con el objetivo de evitar que la generalización de los resultados afecte la calidad de la investigación, se define como **población** la Gerencia de Planificación y Gestión de la empresa PDVSA a nivel

corporativo, unida a las Gerencias de Planificación y Gestión de las 7 Refinerías nacionales de PDVSA, haciendo referencia a:

- ✓ Complejo de Refinerías Paraguaná (CRP), integrada por las refinerías Amuay, Cardón y Bajo Grande.
- ✓ Refinería Puerto La Cruz.
- ✓ Refinería El Palito.
- ✓ Refinería Isla.

Para profundizar en el estudio de las características de la población, y con el objetivo de obtener los datos más significativos de la misma, se hizo uso de la técnica de muestreo **no probabilística**, y dentro de ella el **muestreo intencional**, comprendiendo solamente el Área de Gestión y Planificación de la empresa PDVSA a nivel corporativo, ya que en esta se resumen los aspectos más representativos, que pueden brindar mayor información sobre la planificación y control de las diferentes refinerías.

La **muestra** que va a permitir arribar a conclusiones a partir de la población, es la Gerencia de Gestión y Planificación de la empresa PDVSA a nivel corporativo, ya que las Gerencias de Planificación y Gestión de todas estas refinerías tributan directamente a ella; constituyendo la mejor fuente de información con la que se contará.

Esta investigación ha quedado plasmada en tres capítulos, donde se plantean todos los temas referentes con la misma y con la propuesta final que ha dado como resultado. A continuación se muestra una panorámica de los temas que se abordan en cada uno de los capítulos y que estructuran este documento.

Capítulo 1: En este capítulo se describen los conceptos, que pueden contribuir a un mejor entendimiento de los principales factores que intervienen en el proceso de esta investigación. Además se hace una descripción mucho más exhaustiva de aspectos como el Objeto de Estudio y la Situación Problemática, que se mencionaban anteriormente.

Capítulo 2: En este capítulo se hace una descripción de cada una de las herramientas y técnicas de Base de Datos que fueron estudiadas y que constituyen opciones para dar solución al problema

planteado. Herramientas que pudiesen contribuir al diseño, modelación e implementación de dicha solución, dando lugar a un análisis, que más tarde se verá reflejado en la modelación e implementación de la propuesta.

Capítulo 3: Capítulo donde se describe la solución propuesta. Inicialmente se describen las funcionalidades que se plantean implementar. Además quedarán reflejados en el mismo, el diseño de la Base de Datos, a través del MER y la descripción de cada una de las entidades que responden al nuevo diseño. De igual manera se hace una descripción de cada uno de los procedimientos y triggers implementados.

Capítulo 1 Fundamentación Teórica

1.1. Introducción

En el presente Capítulo se describen los conceptos, que pueden contribuir a un mejor entendimiento de los principales factores que intervienen en el proceso de esta investigación. Se brinda además una visión general de los aspectos relacionados con la misma, incluyendo la situación problemática, y un análisis de los sistemas existentes, que cumplen con algunas de las propiedades y funcionalidades que se pretenden aplicar al Sistema SACGIR.

1.2. Conceptos asociados al dominio del problema

A continuación se mostrarán los conceptos que representan la base para el entendimiento de los principales factores dentro de la investigación.

1.2.1. Conceptos Generales

Con el objetivo de proporcionar un mejor entendimiento de los elementos relacionados con esta investigación se ofrecen un conjunto de conceptos que facilitarán lo anteriormente dicho.

1.2.1.1. Software Libre

El "Software Libre" es un asunto de libertad, no de precio. Para entender el concepto, debes pensar en "libre" como en "libertad de expresión", no como en "cerveza gratis". (1)

"Software Libre" se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software". (1) De modo más preciso, se refiere a cuatro libertades de los usuarios del software:

- ✓ La libertad de usar el programa, con cualquier propósito.
- ✓ La libertad de estudiar cómo funciona el programa, y adaptarlo a tus necesidades. El acceso al código fuente es una condición previa para esto.
- ✓ La libertad de distribuir copias, con lo que puedes ayudar a tu vecino.

- ✓ La libertad de mejorar el programa y hacer públicas las mejoras a los demás, de modo que toda la comunidad se beneficie. El acceso al código fuente es un requisito previo para esto.

1.2.1.2. Base de Datos

Una Base de Datos es un conjunto de información almacenada en memoria auxiliar que permite acceso directo y un conjunto de programas que manipulan esos datos. Base de Datos es un conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquina accesibles en tiempo real y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo. (2)

1.2.1.3. Sistema Gestor de Base de Datos

Un SGBD es un conjunto de programas que permite a los usuarios crear y mantener una BD, por lo tanto, el SGBD es un software de propósito general que facilita el proceso de definir, construir y manipular la BD para diversas aplicaciones. Pueden ser de propósito general o específico (3). Los objetivos fundamentales de los SGBD son:

- ✓ Independencia de los datos y los programas de aplicación.
- ✓ Minimización de la redundancia.
- ✓ Integración y sincronización de las bases de datos.
- ✓ Integridad de los datos.
- ✓ Seguridad y protección de los datos.
- ✓ Facilidad de manipulación de la información.
- ✓ Control centralizado.

1.2.1.4. Sistema

"Sistema es un todo integrado, aunque compuesto de estructuras diversas, interactuantes y especializadas. Cualquier sistema tiene un número de objetivos, y los pesos asignados a cada uno de ellos pueden variar ampliamente de un sistema a otro. Un sistema ejecuta una función imposible de realizar por cualquiera de las partes individuales. La complejidad de la combinación está implícita." (4)

1.2.2. Conceptos Particulares.

Se deben mencionar algunos conceptos que son propios del dominio del problema que ha dado lugar a esta investigación.

1.2.2.1. Indicador

No existe una definición oficial por parte de algún organismo nacional o internacional, sólo algunas referencias que los describen como: "Herramientas para clarificar y definir, de forma más precisa, objetivos e impactos, son medidas verificables de cambio o resultado diseñadas para contar con un estándar contra el cual evaluar, estimar o demostrar el progreso con respecto a metas establecidas, facilitan el reparto de insumos, produciendo productos y alcanzando objetivos". (5) En otras palabras, es una magnitud utilizada para medir o comparar los resultados efectivamente obtenidos, en la ejecución de un proyecto, programa o actividad. Es el resultado cuantitativo de comparar dos variables. Se mide en porcentajes, tasas y razones para permitir comparaciones.

1.2.2.2. Indicador de Refinación

Los indicadores de refinación son un tipo específico de indicadores, que utilizan las empresas petroleras para medir los resultados de alguna operación dentro de un área determinada. En el caso de PDVSA estos indicadores rigen el comportamiento de la empresa y son un punto fundamental en la decisión y planificación económica. Están dados por una fórmula que puede cambiar en dependencia del comportamiento mensual de mismo.

1.2.2.3. Refinería

Centro industrial donde se depura la materia prima hasta conseguir un producto de consumo, como el azúcar o la gasolina. Específicamente, se llama refinería a los centros donde se trata el petróleo para destilarlo y obtener los hidrocarburos. (6)

1.2.2.4. Refinación

Conjunto de procesos industriales empleados para transformar los petróleos crudos en productos derivados; nafta, gas-oil, querosene, solventes, lubricantes, asfalto, entre otros. (7) Por lo que se puede decir, que se trata de un conjunto de procesos que convierten el crudo en unos productos terminados mediante procesos de separación y/o transformación.

1.3. Objeto de estudio

Como un aspecto imprescindible para planificar la investigación, se hace necesaria la descripción del objeto de estudio de la misma.

1.3.1. Descripción General

Con vista a la solución del problema que se plantea en esta investigación, el objeto de estudio estará dirigido a SACGIR, como se había planteado anteriormente.

Producto de la diversidad de procesos que se llevan a cabo en las refinerías de Venezuela y la amplia gama de información que se genera de ellos se obtienen los indicadores financieros y operacionales, los cuales son manejados a distintos niveles dentro de la empresa y se utilizan en la toma de decisiones. Gran parte de la gestión de la información de Refinación se realiza de forma semiautomática. La aparición de la mano del hombre en el proceso, propicia la introducción de errores y ocasiona que los tiempos de respuestas a las demandas de información sean elevados.

SACGIR pretende brindar facilidades de apoyo a los procesos claves de la Gerencia de Planificación y Control de Refinación de PDVSA, liberando a los trabajadores de cálculos pesados y permitiéndoles concentrarse en la elaboración de nuevas estrategias o vías de solución de situaciones para favorecer los objetivos del negocio. De esta forma debe permitir con su puesta en marcha, automatizar la captura de datos de estas actividades al sistema, lograr completitud, disponibilidad y oportunidad en la información, así como evaluar el comportamiento de los indicadores de Refinación.

SACGIR se mostrará como un producto de software amplio, altamente configurable, aplicable según las variables y parámetros que encierra y extensible en todos los sentidos del negocio del control de indicadores de refinación. Dentro de sus objetivos fundamentales se encuentran lo siguientes aspectos:

- ✓ Gestionar de forma automatizada la información proveniente de las Refinerías, tanto de los Resultados Mensuales como de las Operaciones Diarias.
- ✓ Analizar y evaluar los indicadores de la información proveniente de las Refinerías.
- ✓ Generar reportes a partir de los valores mensuales provenientes de las distintas refinerías.
- ✓ Facilitar la planificación de Producción y Paradas de Planta.

- ✓ Permitir el seguimiento de las Operaciones Diarias en las Refinerías.
- ✓ Analizar de forma integral los Resultados Mensuales de Refinación.
- ✓ Mantener la información histórica de Refinación.
- ✓ Gestionar el acceso a la información mencionada y mantener una traza.
- ✓ Facilitar la inclusión de nuevos elementos en el Sistema para el Control de Gestión de Indicadores de Refinación, tales como Refinerías, Unidades de Proceso o Indicadores.

1.3.2 Descripción actual del dominio del problema

PDVSA es la empresa estatal dedicada al sector de los hidrocarburos más sólida de América Latina. Así lo reflejan los indicadores financieros de solvencia y capacidad de endeudamiento de la Corporación, en los cuales se evidencia los resultados exitosos de la estrategia aplicada, así como del destino productivo de los fondos recibidos.

La empresa ha sido víctima de los conflictos entre sus ejecutivos y gerentes con el gobierno de Chávez, debido en gran medida a la reforma petrolera puesta en marcha por el gobierno. Dicha Reforma buscaba corregir la PDVSA que había logrado convertirse prácticamente en un Estado dentro del Estado, y que había escamoteado al pueblo venezolano utilizando su enorme capacidad y potencial económico, siendo capaz hasta entonces de definir el rumbo de la política petrolera venezolana.

Se puede mencionar como un ejemplo de los conflictos anteriormente expuestos, el sabotaje cometido contra la petrolera entre diciembre del 2002 y enero del 2003.

A raíz del paro petrolero, todas las operaciones realizadas por PDVSA se vieron en peligro, y en muchos casos el problema se hacía constar, cuando en algunas de las refinerías los saboteadores cambiaron las claves de acceso, alteraron el código del programa, sustrajeron equipos, destruyeron el computador central ó bloquearon el acceso a alguna de las salas de la refinería, quedando en muchos casos, la administración de alguna de las áreas en manos del saboteador, poniendo en peligro la integridad, confiabilidad y el control sobre la gestión de refinación.

A partir de lo anterior, además de que el sistema por el que PDVSA regía sus operaciones no cumplía con todos los requerimientos que son presentados en el negocio actual del sistema, y unido a que el

Gobierno Nacional Venezolano, en el Decreto 3390, de fecha 23 de febrero del 2004, orienta a todas las empresas gubernamentales a migrar sus plataformas tecnológicas al software libre y al empleo de estándares abiertos, PDVSA tiene que plantearse la ruta camino a la migración al software libre; lo que le impone a la petrolera proponerse un nuevo sistema, que cumpla con los requerimientos del negocio, supla las fallas a las que se hacía alusión anteriormente, y en esencia, quede automatizada, la gestión de diferentes procesos, de acuerdo a las necesidades de la gestión de refinación, quedando establecidos para PDVSA los primeros pasos hacia la independencia y el logro de la plena soberanía tecnológica.

1.3.3 Situación Problemática

Los indicadores son fundamentales dentro del funcionamiento de cualquier entidad, ya que permiten medir cambios en una condición o situación dentro de la misma, posibilitando mirar de cerca los resultados de iniciativas o acciones, lo que los hace un instrumento muy valioso para evaluar y dar seguimiento al proceso de desarrollo de la empresa, o lo que es lo mismo, para orientar sobre cómo se pueden alcanzar mejores resultados.

PDVSA, como cualquier empresa que se interese por un mejoramiento de sus funcionalidades y un incremento de su producción, se ha planteado una serie de indicadores, conocidos como indicadores de refinación, los que representan un punto clave en la evaluación, planificación y control de los procesos de negocio para la Gerencia en la toma de decisiones.

Teniendo en cuenta la importancia que tienen estos indicadores dentro del negocio de PDVSA, surge SACGIR, como una aplicación de control, reporte y consulta, capaz de automatizar la gestión de los indicadores. En la actualidad se planea dar solución a diferentes fallas que SACGIR no ha incorporado dentro de su propuesta de solución, debido en gran medida a:

- ✓ La información referente a los indicadores de refinación es escasa y muy dispersa.
- ✓ No existe conocimiento ni documentación sobre los aspectos que tienen en común y en los que difieren cada uno de los indicadores.
- ✓ La modelación efectuada del módulo de Indicadores es insuficiente para soportar el cálculo de los indicadores de refinación.

- ✓ No se permite agregar a la fórmula de un indicador un dato que no se encuentre almacenado dentro de la Base de Datos del sistema.

Hechos estos planteamientos, ¿qué pasaría si se determinara que cierto indicador debería estar compuesto por un dato que no se encuentra físicamente en la Base de Datos, o si se crea un nuevo indicador con información que el sistema no maneja de forma persistente en la Base de Datos?

Para resolver este problema se tendría que hacer una reestructuración de la Base de Datos y adaptarla de manera que pueda soportar la incorporación de esta nueva información que necesitan los indicadores, creando nuevas entidades o agregándole nuevos atributos a las ya existentes. Sin duda alguna este proceso llevaría consigo nuevos gastos de recursos para PDVSA, ya que se tendrían que hacer estas reestructuraciones cada vez que suceda este problema indefinidamente, resultando esta la solución menos factible para resolver el problema.

Por otra parte, con respecto a la interrogante que se planteaba anteriormente sobre la creación de un nuevo indicador y a su dinámica cambiante, es de suma importancia, por lo que dichos indicadores representan dentro del propio negocio de refinación, definir hasta que punto afectaría esto a la institución y determinar si es sólo una pequeña deficiencia o es en realidad un problema que afecta el buen desempeño y funcionamiento de todo el negocio.

SACGIR tiene como objetivo automatizar todos los procesos relacionados con los indicadores de refinación para monitorizarlos continuamente, buscando una mayor eficiencia en todos los sentidos y en todas las acciones que se llevan a cabo en las refinerías de PDVSA. Esto indica que una vez que alguno de estos procesos falle por las causas ya determinadas con anterioridad, fallará la aplicación por completo, ya que quedarán incompletas las operaciones que dependan de esas funcionalidades, provocando desencanto y decepción con respecto a SACGIR y en el peor de los casos una gran cantidad de pérdidas materiales a la empresa.

1.4. Análisis de otras soluciones existentes

Con el objetivo de hacer más comprensible la situación problemática que se planteó anteriormente, se hará un análisis de las aplicaciones que en la actualidad rigen toda la gestión de información en la empresa petrolera y se mostrará cómo no cumplen con todos los requerimientos del negocio actual de PDVSA. Además se explican algunas de las herramientas o aplicaciones existentes, que incluyan o

que pueden contribuir a dar respuesta a la situación problemática, de acuerdo a los objetivos planteados.

1.4.1. Sistema de Información de Manufactura

El SIM (Sistema de Información de Manufactura), tiene como objetivo consolidar la información relevante para la gestión de las refinerías de PDVSA, apoyando la cadena de valor del Negocio de Refinación, mediante el uso de una Base de Datos única que permite mantener la uniformidad, confiabilidad y consistencia de la información que suministran las refinerías, requerida para la toma de decisiones en la corporación.

El SIM contiene módulos adaptados a la secuencia de la ejecución y control de los procesos del cliente, siendo administrado por el equipo de Control y Gestión - Caracas.

Desarrollado en plataforma web con el lenguaje propietario ASP y con soporte de datos sobre Oracle. No obstante el proceso de Refinación no está totalmente cubierto por sistemas como este, que apoyan el proceso dentro de PDVSA, dando espacio al procesamiento de datos de forma manual y a los inconvenientes que esto conlleva.

El SIM consolida los resultados operacionales, financieros y de personal para la gestión de las refinerías del circuito Refinador Venezuela – Isla. Estos resultados son necesarios para informar a la junta directiva de Refinación, Suministro y Comercio y emitir reportes.

Dispone además de un módulo de trazabilidad, el cual deja registrada la operación que el usuario realiza en el sistema. Esto facilita el control exhaustivo de las actividades y la auditoría de la información, lo cual es imperioso debido a la variedad de roles definidos en el sistema, y la relevancia de la información que maneja para la Gestión de Refinación. Cuenta con ayudas en línea que pueden ser consultadas desde cualquier ventana de la aplicación proporcionando información detallada de las opciones y acciones que pueden realizarse en las mismas. El sistema puede ser trabajado en español o en inglés; a su vez el usuario puede cambiar de sitio web Inglés al Español o viceversa, en cuyo caso, el usuario queda registrado con el último idioma seleccionado.

Las consultas del VIII B¹, sólo serán representadas en Excel.

El SIM lo conforma tres Módulos y dos interfaces:

- ✓ Módulo de Carga/Actualización: En el cual se puede incluir, modificar y eliminar información de la Gestión de Refinación que se maneja en este sistema.
- ✓ Módulo de Consulta: En el cual se puede generar información consolidada de una o más Refinerías o Complejos, en un periodo determinado y selección de los otros criterios correspondientes a cada opción, con la información cargada y confirmada en el Módulo de Carga/Actualización.
- ✓ Módulo de Administración: En el cual los usuarios con el rol de administrador, pueden incluir, modificar y eliminar información de las tablas maestra y administración de usuarios. Así como también permite la consulta de la trazabilidad de las acciones ejecutadas en la carga de los datos transaccionales. Por otra parte, se maneja la consulta de información concerniente a los datos de los usuarios.
- ✓ Interfaz de Volúmenes Reales Mensuales con el SIMP: Trae los datos de Volúmenes Reales Mensuales de las Refinerías Amuay y Cardón, la cual es activada desde la aplicación SIMP y una vez que en el SIM sea confirmada esta información, no se podrá activar nuevamente esta interfaz para dicha refinería mes y año.
- ✓ Interfaz de Mermas con el SIMP: Trae los datos de Mermas de las Refinerías Amuay y Cardón, la cual es activada desde la aplicación SIMP y una vez que en el SIM sea confirmada esta información, no se podrá activar nuevamente esta interfaz para dicha refinería mes y año.

Beneficios:

- ✓ Permite mantener uniformidad, confiabilidad, consistencia y disponibilidad inmediata de la información de gestión operacional y financiera de las refinerías del Sistema Venezuela + Isla.

¹ Documento a través del cual se realiza el reporte de Balance Volumétrico de una Refinería a la Gerencia Corporativa.

- ✓ Minimiza el tiempo de procesamiento de la información para el análisis, evaluación y toma de decisiones.
- ✓ Facilita la disponibilidad de información a otras organizaciones en PDVSA y entes gubernamentales relacionados.
- ✓ Disponibilidad de data histórica.
- ✓ Sirve de plataforma a potenciales sistemas gerenciales-corporativos para consulta y toma de decisiones.
- ✓ Por ser una aplicación en tecnología Web permite a sus usuarios conectarse desde todas las áreas a nivel nacional y refinería Isla.

Desventajas:

- ✓ Actualmente se tiene acceso al sistema pero no al conocimiento, lo cual imposibilita a la empresa de su uso según sus intereses.
- ✓ Dificultad de integración efectiva con otros sistemas actuales. Existen diferentes soluciones no estándares para un mismo problema.
- ✓ Esta aplicación no se adapta completamente a las necesidades de la refinería pues no incluye la gestión de indicadores de refinación, que son un elemento clave dentro de la evaluación y toma de decisiones de la petrolera.
- ✓ La aplicación está provista de una Base de Datos y un lenguaje de programación ASP, ambos propietarios, lo que va en contra de la política de migración a software libre de la empresa.

1.4.1.1. Sistema Integrado de Movimiento de Petróleo

El SIMP (Sistema Integrado de Movimiento de Petróleo) tiene por principales objetivos el control y supervisión de todas las operaciones de movimiento de crudo entre los patios de tanque y los terminales de embarque, el control del movimiento de crudo y productos en refinería, la planificación, ejecución y control de las operaciones de carga y descarga de los tanqueros en la terminal de embarque y la generación y emisión de toda la documentación de embarque.

1.4.2. PhpMyAdmin

PhpMyAdmin es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas webs, utilizando Internet. Es una herramienta muy completa que permite acceder a todas las funciones típicas de la Base de Datos a través de una interfaz web muy intuitiva. La aplicación en si, no es más que un conjunto de archivos escritos en PHP que puede ser copiada en un directorio de cualquier servidor web, de modo que, cuando se acceda a esos archivos, se muestren unas páginas donde se pueden encontrar las bases de datos a las que se tienen acceso en el servidor de bases de datos y todas sus tablas. Dentro de las funcionalidades que ofrece se encuentran:

- ✓ Crear una nueva Bases de Datos.
- ✓ Eliminar una Bases de Datos.
- ✓ Crea y alterar tablas.
- ✓ Eliminar Tablas.
- ✓ Eliminar, añadir y modificar campos.
- ✓ Administrar privilegios y usuarios.
- ✓ Exportar datos en varios formatos
- ✓ Crear un backup (copia de respaldo) de una Base de Datos.
- ✓ Ejecutar consultas SQL.

Una de las principales ventajas de la herramienta, es que está disponible en 54 idiomas y sobre todo que se encuentra disponible bajo la licencia GPL. (8)

1.4.3. PhpPgAdmin

Es una herramienta similar a PhpMyAdmin pero en este caso administra bases de datos en PostgreSQL. Entre las características que ofrece se encuentran la posibilidad de administrar varios servidores, soporte a múltiples versiones de PostgreSQL, administración de usuarios, grupos, bases de datos, esquemas, manipulación sencilla de datos, exportar los datos a diferentes formatos, importar sentencias SQL además de las mismas que ofrece PhpMyAdmin. (9)

1.4.4. PhpMyAdmin y PhpPgAdmin

PhpMyAdmin y PhpPgAdmin son dos aplicaciones web de gran uso por los desarrolladores, ya que ambos ofrecen una forma sencilla de interactuar y gestionar todo lo relevante con los servidores de

bases de datos de MySQL y PostgreSQL, de forma gráfica y muy cómoda, ofreciendo una amplia variedad de funcionalidades.

1.4.4.1. Elementos comunes con la investigación

- ✓ Funcionalidad de crear una nueva tabla en la Base de Datos.
Esto se pone de manifiesto cuando ocurre la necesidad de alterar el diseño de la Base de Datos, ya sea por la creación de un nuevo indicador o por la modificación de su fórmula de creación. En este caso si el sistema no está provisto de la capacidad de almacenar la información necesaria para el cálculo de determinado indicador y este necesita de una entidad con atributos, se procede a la creación de una nueva tabla en la Base de Datos.
- ✓ Relacionar una tabla con otra.
Una vez que se haya creado una nueva entidad se necesitará relacionarla, si es preciso, con una o más entidades en el modelo de datos, para que de respuesta a las necesidades del sistema.
- ✓ Funcionalidad de crear un nuevo atributo en una tabla en la Base de Datos.
Al igual que en el caso de crear una nueva tabla en la Base de Datos, esta funcionalidad se lleva a cabo con el mismo propósito, sólo que en este caso no es necesario crear una nueva entidad, sino que se puede agregar un nuevo atributo a una entidad ya existente.
- ✓ Manejar llaves primarias y secundarias.
Las llaves primarias son la forma de identificar una tupla dentro de una entidad, es necesario que cuando se cree una, se le asigne un atributo que las identifique del resto de las tuplas. Por otro lado las llaves secundarias o foráneas se utilizan para relacionar las tablas, por lo que también son de suma importancia.

1.4.4.2. Elementos no afines a la investigación

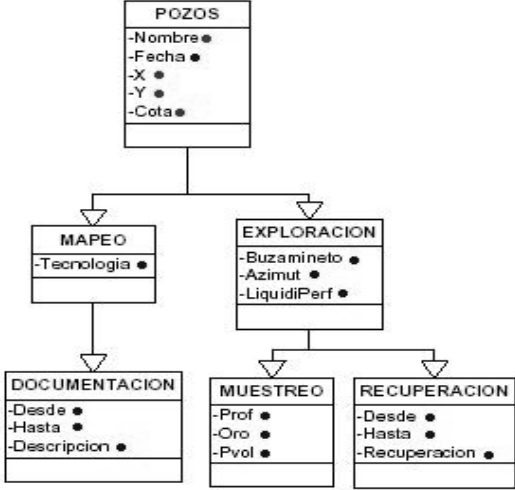
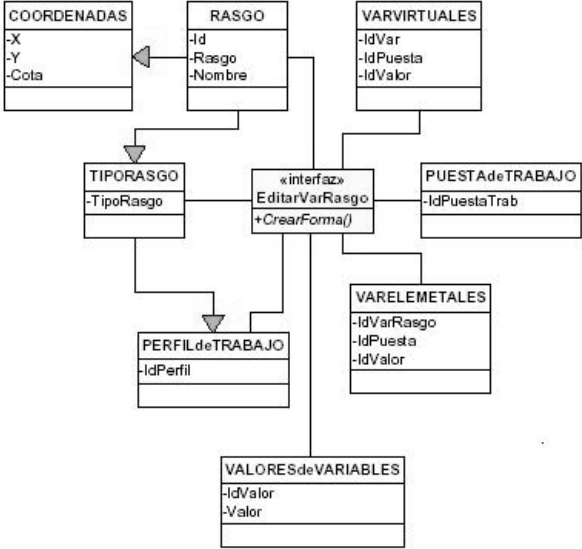
- ✓ Ninguno de ellos es específico para ninguna aplicación, son muy generales.
- ✓ No están sujetos a reglas de ninguna institución o negocio como en el caso de esta investigación.
- ✓ Las funciones de eliminar, crear y modificar se llevan a cabo sin ningún tipo de restricción.

1.4.5 Base de Datos Relacional con Atributos y Registros Variables

La BDARV, es una tecnología desarrollada por el Grupo de Servicios Informáticos de la Empresa Geominera Oriente.

Para lograr un buen entendimiento de esta tecnología se explicará la estructura de una Base de Datos muy simple, propia del sistema geominero, la cual está formada por pozos de mapeo y estructurales, los cuales tienen nombre del pozo (nombre), fecha de perforación (fecha), profundidad total del pozo (Prof.), x, y, cota. Los de mapeo sólo tienen la tecnología de perforación (tecnología) y los estructurales el azimut, buzamiento y tipo de líquido de perforación (liqperf). Los pozos de mapeo tienen la documentación con el desde, hasta y descripción geológica (descripción) y los estructurales el muestreo compuesto por profundidad de muestreo (prof) el contenido de oro (oro) y el peso volumétrico de la mena (pvol) y tiene además la recuperación con el desde, hasta y recuperación (recup). (10)

En la Tabla # 1 se muestra el diagrama de la Base de Datos anterior, y la comparación entre las Base de Datos realizadas por el método clásico y con BDARV.

Clásica	Atributos y Registros Variables
	
<p>-Estructura fija.</p>	<p>-El pozo constituye un rasgo.</p>

<p>-Los campos sólo pueden tener el mismo atributo.</p> <p>(Ej. Unidades de Medidas (U.M.) iguales)</p> <p>-Si aparecieran pozos con datos diferentes a este esquema, tendría que modificarse el Software de captación y la Base de datos.</p>	<p>-Los campos de la estructura clásica pasan a ser datos (variables virtuales).</p> <p>-Las Variables Virtuales tienen asociadas Puestas de Trabajo, las que a su vez tienen variables elementales. Esto permite que tengan diferentes atributos (ej. Unidades de Medidas diferentes).</p> <p>-Aparece el concepto de variables elementales para caracterizar a las variables virtuales.</p> <p>-Tener las coordenadas relacionadas con el rasgo (de uno a varias) permite incluir rasgos con más de una coordenada (ej. Polígonos) con pozos.</p>
--	---

Tabla #1. Comparación entre la estructura de Base de Datos relacional y la técnica de BDARV.

Para poder entender la BDARV y hacer más clara la tabla comparativa entre la misma y una estructura de Base de Datos relacional es necesario explicar algunos conceptos básicos de su estructura, dentro del sistema geológico.

1.4.5.1 Rasgos

Es la entidad que representa un grupo de elementos georeferenciados del mundo real que contiene información geológica y tiene una representación geométrica dada, contiene los atributos de ubicación de acuerdo con el tipo geométrico establecidos en la ISO 19107 (Polígono, Línea, Punto, Polilínea o Segmento) y atributos adicionales que constituyen las llamadas variables del rasgo. Ejemplo: Pozos de Perforación, Puntos de observación, Trincheras, entre otros.

Los atributos que caracterizan al rasgo se nombran Variables Reales, ellos son:

- ✓ Identificador: Valor numérico que identifica al rasgo dentro de la Base de Datos.
- ✓ Nombre: Nombre del rasgo en el mundo real de hasta 12 caracteres.
- ✓ Clase Geométrica: Clase geométrica a que pertenece el rasgo (Punto, línea, polígono).
- ✓ Pertenencia: Identificador de rasgo al que puede pertenecer el rasgo (Un pozo puede pertenecer a un polígono, o un punto de muestreo puede formar parte de una trinchera).

1.4.5.2 Tipos de Rasgos

Es una entidad perteneciente al grupo de entidades representadas por el Rasgo de la cual heredan sus atributos y pueden agregarse otros que constituyen las variables del Tipo de Rasgo. Es decir los Rasgos son meta clases cuyas instancias son los Tipos de Rasgos. Ejemplo: Rasgo: Pozo de perforación y Tipo de rasgo: Pozo estructural.

Los atributos que caracterizan (Variables Reales) al Tipo de Rasgo son:

- ✓ Identificador: Valor numérico que identifica al Tipo de rasgo dentro de la Base de Datos.
- ✓ Tipo de Rasgo: Código identificador del tipo de rasgo.

1.4.5.3 Variables Virtuales

Son Variables Virtuales o simplemente Variables, los atributos normalizados de las clases persistentes que definen las entidades diseñadas en los modelos lógicos. Ejemplo: Fecha de perforado el rasgo Pozo de perforación. Profundidad del muestreo.

Las variables virtuales pueden ser:

- ✓ Simples: Son las que se definen en tiempo de diseño.
- ✓ Codificadas: Son las que toman el nombre de un clasificador de caracteres.

Ejemplo. La variable Sílice está definida como mineral en el clasificador de minerales.

Las variables quedan completamente definidas por los siguientes atributos, que son utilizados en el GeoDato:

- ✓ Código: Es un número entero único para todo el programa GEODATO.
- ✓ Nombre: Nombre corto de la variable con hasta 30 caracteres.
- ✓ Descripción: Descripción de la variable de hasta 70 caracteres.
- ✓ Etiqueta: Etiqueta de hasta 10 caracteres la cual identifica la variable.
- ✓ Tipo: Tipo del la variable.
- ✓ Obligatoriedad: Fija si la variable es obligatoria o no para el GeoDato.

1.4.5.4 Variables Elementales

Las variables elementales son variables que describen a las variables virtuales a diferencias de las virtuales propiamente dichas que describen rasgos, tipos de rasgos y perfiles de trabajo. Sus atributos son los mismos que éstas.

Ejemplo: La variable virtual profundidad para que quede completamente definida tiene a la variable elemental Unidad de medida. El contenido de cobre para que quede completamente definido, tiene por definición en los modelos lógicos a las variables elementales:

- ✓ Laboratorio en que se hizo el ensayo.
- ✓ Método con que se hizo el análisis.
- ✓ Norma de calidad utilizada en la determinación.
- ✓ Unidad de medida en que se da la concentración de cobre.

1.4.5.5 Puesta de Trabajo

Puesta de Trabajo se denomina al grupo de variables elementales con valor, definidas para caracterizar a la variable virtual y es identificada con una etiqueta.

Ejemplo: Variable Virtual: Cu, Variables Elementales: Laboratorio; Norma; UM; etc. (Unidad de medida).Puede tener 2 puestas:

Puesta1= laboratorio Elio Trincado , Norma 160/1978 A, UM %

Puesta2= laboratorio Isac del Corral, Norma 160/1978 B, UM %

1.4.5.6 Perfiles de Trabajo

Es la representación digital dentro del GeoDato de un conjunto de datos en un documento geológico. Contienen el conjunto de variables contenidas en una fuente de información dada dentro de los documentos de soporte físico de los datos primarios (Tablas, informes, registros, etc.). En estos perfiles se recogen las variables por niveles jerárquicos, se establecen sus restricciones, orden de captación y llaves de relación de los niveles jerárquicos. Con ellos puede crearse el formulario de captación en correspondencia con la fuente del dato y asegura la vista de usuario de los datos idéntica a su organización en el soporte original.

Ejemplo: Documentación geológica de pozo. (Desde, Hasta, Documentación), Punto de medición geofísica. (DGBouguer, DeltaZ, GanmaTotal).

1.5 Conclusiones

En este capítulo se profundizó en los conceptos que envuelven todo el proceso de esta investigación, quedando definido el ambiente del mismo, luego de realizar un análisis de las condiciones a las que está sujeto el objeto de estudio, así como la situación problemática. En el capítulo se describe además la solución que existe actualmente en PDVSA, que no cumple con los nuevos requerimientos que impone el negocio, y que constituye una de las motivaciones para esta investigación, a través de la cual han salido a la luz, como aplicaciones que presentan similitud con los propósitos que se persiguen, el PhpPgAdmin, el PhpMyAdmin, y la técnica de BDARV, que también fue descrita en el presente.

Capítulo 2 Tendencias y Tecnologías Actuales a Desarrollar

2.1. Introducción

En el presente capítulo se describen las herramientas a través de las cuales se implementa la solución propuesta, haciendo énfasis en el proceso que se llevó a cabo para su modelación e implementación. Otro de los aspectos que se desarrolla es el referente a las técnicas de Base de Datos que se estudiaron y que conllevaron a la selección de las apropiadas para la implementación. Además se describen herramientas, que aunque en algunos casos no forman parte de la propuesta de solución, si contribuyeron al enriquecimiento del análisis, que dio lugar a la parte de la arquitectura que aquí se propone.

2.2. Sistemas Gestores de Base de Datos

Los SGBD o DBMS, como también son conocidos, son un tipo de software muy específico, dedicado a servir de interfaz entre la Base de Datos, el usuario y las aplicaciones que la utilizan. Se componen de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. A continuación se mencionan algunos de ellos.

2.2.1. Oracle

Oracle es una herramienta cliente/servidor para la gestión de Base de Datos relacional. Esta herramienta hace uso de los recursos del sistema informático en todas las arquitecturas de hardware, para garantizar su aprovechamiento al máximo en ambientes cargados de información. (11) Además, Oracle es una suite de productos que ofrece una gran variedad de herramientas, entre las que se encuentran el SQLForms, por citar un ejemplo, a través de la cual se podrán diseñar pantallas para el ingreso, modificaciones, bajas y consultas de registros. Soporta la mayoría de los lenguajes de computación al igual que 26 idiomas diferentes. A través de este RDBMS se pueden desarrollar aplicaciones de OLTP y de data warehousing mayores y más exigentes.

Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacando su soporte de transacciones, estabilidad, escalabilidad y el hecho de que sea multiplataforma, pues posee igual interacción en todas la plataformas (Windows, Unix, Macintosh y Mainframes), ya que más del

80% de sus códigos internos son iguales a los establecidos en todas las plataformas de Sistemas Operativos. Soporta bases de datos de todos los tamaños, desde severas cantidades de bytes hasta gigabytes. (11)

El sistema ha sido criticado por algunos especialistas en cuanto a la seguridad de la plataforma y las políticas de suministro de parches de seguridad que incrementan el nivel de exposición de los usuarios. (11) Por la gran potencia que tiene y su elevado precio, hace que sólo se vea en empresas muy grandes y multinacionales, por norma general.

2.2.2. MySQL

MySQL es un sistema de gestión de bases de datos relacional de gran uso a nivel mundial. Una de las principales características que le atribuyen su popularidad es que es un producto de código abierto licenciado bajo la licencia GPL de GNU, además de que es muy fácil de usar. Existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su sencilla instalación y configuración. Aunque es software libre, hay una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que ofrece, y la posibilidad de integrar este gestor en un software propietario. Su diseño multihilo le permite soportar una gran carga de procesamiento de información de forma muy eficiente. (12) Dentro de sus principales características se observan:

- ✓ Aprovechamiento de la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
- ✓ Es multiusuario.
- ✓ Gran velocidad a la hora de realizar las operaciones.
- ✓ Bajo consumo de recursos, lo que lo hacen apto para ser ejecutado en una máquina con escasos recursos sin ningún problema.
- ✓ Soporta gran cantidad de tipos de datos para las columnas.
- ✓ Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP y otros).
- ✓ Gran portabilidad entre sistemas.

- ✓ Soporta hasta 32 índices por tabla.
- ✓ Gestión de usuarios y contraseñas, manteniendo un muy buen nivel de seguridad en los datos.

2.2.3. PostgreSQL

El SGBD relacional orientado a objetos conocido como PostgreSQL (y brevemente llamado Postgres95) está derivado del paquete Postgres escrito en Berkeley, distribuida bajo licencia BSD. Con más de una década de desarrollo tras él, PostgreSQL es el gestor de bases de datos de código abierto más avanzado hoy en día, ofreciendo control de concurrencia multi-versión, soportando casi toda la sintaxis SQL (incluyendo subconsultas, transacciones, y tipos y funciones definidas por el usuario), contando también con un amplio conjunto de enlaces con lenguajes de programación como pueden ser C, C++, Java, Perl, TCL y Python. (13)

Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo. Dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group). (13)

Con el pasar del tiempo muchos desarrolladores entusiastas de los motores de Base de Datos se unieron al proyecto y entre todos comenzaron a incorporar muchas características al motor, destacándose entre ellas: (14)

- ✓ Cliente/Servidor. Usa una arquitectura proceso por usuario cliente/servidor. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.
- ✓ Alta concurrencia: Mediante un sistema denominado MVCC (Acceso concurrente multiversión) por sus siglas en inglés, PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo commit (ver Glosario de Términos).

- ✓ Disparadores (Triggers): La ejecución de un procedimiento almacenado basado en una determinada acción (Insertar, Actualizar, Eliminar) sobre una tabla específica. Son además funciones enlazadas a operaciones sobre los datos.
- ✓ Herencia de Tablas: Una tabla puede heredar las funcionalidades de otra tabla especificada.
- ✓ Funciones: Bloques de código que se ejecutan en el servidor. Pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos da, desde las operaciones básicas de programación, tales como bifurcaciones y bucles, hasta las complejidades de la programación orientación a objetos o la programación funcional.
- ✓ Lenguajes Procedurales. Tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/PGSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL.

2.3. PL/PGSQL

PL/PGSQL permite ejecutar comandos SQL mediante un lenguaje de sentencias imperativas y uso de funciones, dando mucho más control automático que las sentencias SQL básicas. Es un lenguaje procedural cargable para el sistema de bases de datos PostgreSQL que tiene como principales objetivos: (15)

- ✓ Crear funciones y procedimientos disparados por eventos (Triggers).
- ✓ Añadir estructuras de control al lenguaje SQL.
- ✓ Realizar cálculos complejos.
- ✓ Heredar todos los tipos definidos por el usuario, las funciones y los operadores.
- ✓ Ser fiable para el servidor.
- ✓ Fácil de usar.

Excepto en el caso de funciones de conversión de entrada/salida y de cálculo para tipos definidos, cualquier cosa que pueda definirse en funciones de lenguaje C puede ser hecho con PL/PGSQL. Es posible crear funciones complejas de cálculo y después usarlas para definir operadores o usarlas en

índices funcionales. Las funciones que están escritas en PL/PGSQL aceptan parámetros y pueden devolver tipos de datos básicos o complejos como son vectores, registros, tablas y funcionan en cualquier plataforma donde PostgreSQL corra. (15)

2.4. Metodologías de desarrollo de Software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software. Estas orientan paso a paso todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben tener, detallando la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla. En la actualidad, con el objetivo de garantizar la mayor calidad posible, no se concibe un proyecto que no emplee alguna metodología que rija su desarrollo, en aras de evitar resultados impredecibles, la detección tardía de errores, y que afecten el trabajo factores como: la introducción de nuevas herramientas y cambios de organización.

2.4.1. RUP (Rational Unified Process)

El Proceso Unificado de Desarrollo es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto. (16)

Está compuesto por nueve flujos de trabajo y cuatro fases, cada una de ellas con un número variable de iteraciones como se muestra en la Figura #1.

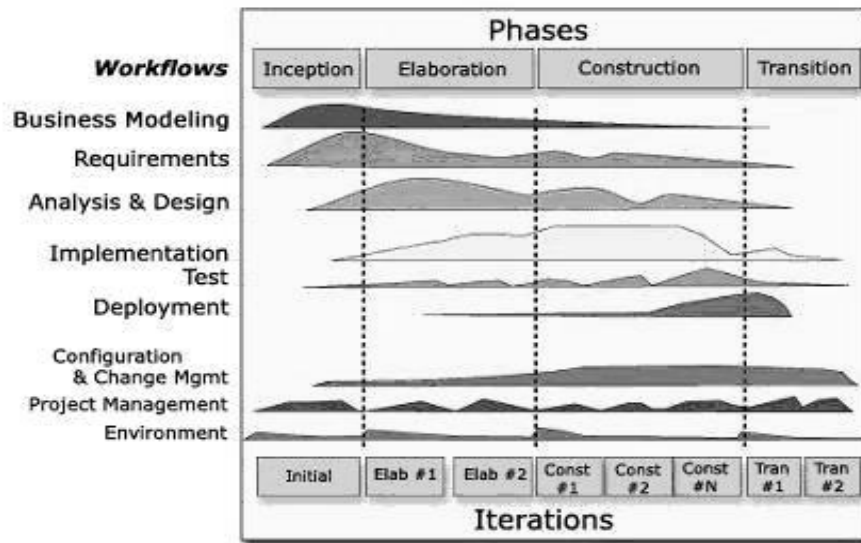


Figura #1. Fases e Iteraciones de la Metodología RUP.

El ciclo de vida de esta metodología tiene las siguientes características:

Dirigido por Casos de Uso: Un caso de uso representa una interacción entre el usuario que va a utilizar el sistema y este, por lo que, para construir un sistema con éxito y que satisfaga las necesidades de sus usuarios hay que conocer lo que ellos necesitan y desean. “Dirigido por casos de uso quiere decir que el proceso de desarrollo sigue un hilo, avanza a través de una serie de flujos de trabajo que parten de los casos de uso.” (16) Dichos casos de uso son especificados, diseñados, llevados a código fuente y probados por los ingenieros hasta formar en su conjunto un producto final.

Centrado en la Arquitectura: La arquitectura en un sistema software es una vista global del sistema con las características más relevantes resaltadas, incluyendo los aspectos dinámicos y estáticos más significativos. Mediante la arquitectura se le da forma al sistema en cuestión, permitiendo que evolucione con el pasar de los años y que los casos de uso encajen en el sistema de manera perfecta aportando las funcionalidades requeridas al software.

Iterativo e Incremental: Debido al esfuerzo y a la duración que puede estar sometido un producto software, es sumamente conveniente dividir el trabajo en partes más pequeñas y manejables y haciéndolo de una forma controlada. Estas partes pequeñas constituyen una iteración y su culminación un incremento del producto a construir.

Los elementos del RUP son: (17)

- ✓ Actividades: Son los procesos que se llegan a determinar en cada iteración.
- ✓ Trabajadores: Son las personas o entes involucrados en cada proceso.
- ✓ Artefactos: Puede ser un documento, un modelo, o un elemento del modelo.

Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software. (17)

2.4.2. XP (Extreme Programming)

“Es una de las metodologías de desarrollo de software más exitosas en la actualidad utilizada para proyectos de corto plazo, equipo pequeño y cuyo plazo de entrega era ayer. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto”. (17)

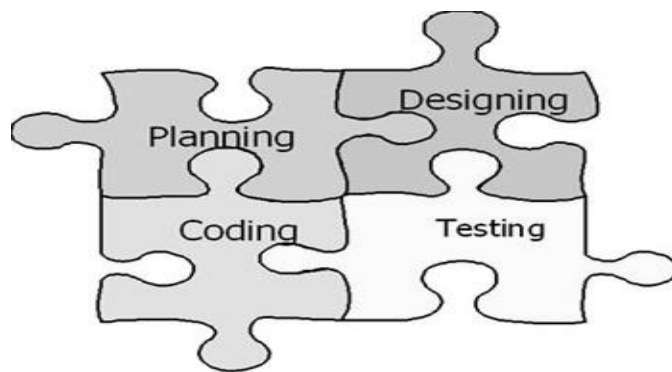


Figura #2. Metodología Extreme Programming.

La metodología se basa en: (17)

- ✓ Pruebas Unitarias: Se basa en las pruebas realizadas a los principales procesos, de tal manera que se adelante en algo hacia el futuro; se puedan hacer pruebas de las fallas que pudieran ocurrir.
- ✓ Refabricación: Se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.

- ✓ Programación en pares: Una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. Es como el chofer y el copiloto: mientras uno conduce, el otro consulta el mapa.

Los elementos fundamentales de esta metodología son: (17)

- ✓ La comunicación, entre los usuarios y los desarrolladores.
- ✓ La simplicidad, al desarrollar y codificar los módulos del sistema.
- ✓ La retroalimentación, concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales.

2.5. UML (Unified Modeling Language)

“UML es ya un estándar de la industria, pero no sólo de la industria del software sino, en general, de cualquier industria que requiera la construcción de modelos como condición previa para el diseño y posterior construcción de prototipos”. (18)

“UML es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos. Se ha convertido en el estándar de facto de la industria, debido a que ha sido impulsado por los autores de los tres métodos más usados de orientación a objetos: Grady Booch, Ivar Jacobson y Jim Rumbaugh”. (19)



Figura #3. Logo de UML.

El Lenguaje Unificado de Modelado es una herramienta que sirve como enlace entre quien tiene la idea y el desarrollador. Esto es posible gracias a un conjunto de símbolos y diagramas, que tienen

fines distintos dentro del proceso de desarrollo. En la actualidad es el lenguaje de modelado de software más conocido y utilizado. A través de él se puede visualizar, especificar, construir y documentar un sistema de software, donde se especifica, no se describen los métodos y procesos. Se debe añadir a esta descripción que UML no especifica en sí mismo qué metodología o proceso usar.

Uno de los aspectos que resaltan en este modelado visual es su independencia del lenguaje de implementación, de tal forma que los diseños realizados se pueden implementar en cualquier lenguaje que soporte las posibilidades de UML (principalmente lenguajes orientados a objetos). (20) Permite generar código a partir de los modelos y a la inversa, permitiendo que el modelo y el código estén estrechamente relacionados. Es un lenguaje de modelado muy fácil de aprender y utilizar.

La primera versión de UML 1.1 fue adoptada por la OMG (Object Management Group) como estándar en noviembre de 1997. (20) Desde entonces, UML atravesó por varias revisiones y refinamientos hasta llegar a la versión actual: UML 2.1.

2.6. Herramientas para el Modelado (Herramientas CASE: Computer Aided Software Engineering)

Las herramientas Case permiten organizar y manejar la información de un proyecto informático, tornándolo más flexible y comprensible para los participantes de un proyecto y mejorando además la comunicación entre los mismos. Por esta razón se puede decir que representan una forma a través de la cual se pueden modelar los Procesos de Negocio de las empresas.

2.6.1. Visual Paradigm para UML

Visual Paradigm para UML es una herramienta profesional para el modelado UML considerada como muy completa y fácil de usar, es multiplataforma y proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. Soporta el ciclo de vida completo del desarrollo de software: captura de requisitos, análisis y diseño orientados a objetos, implementación, pruebas y despliegue. Permite realizar todos los tipos de diagramas de clases, generar código desde diagramas y generar documentación. (21)

Dentro de sus principales características se pueden citar: (21)

- ✓ Soporte de UML versión 2.1.
- ✓ Diagramas de Procesos de Negocio, Actor de negocio, Documento.

- ✓ Ingeniería inversa.
- ✓ Generación de código.
- ✓ Editor de Detalles de Casos de Uso.
- ✓ Diagramas de Flujos de Datos.
- ✓ Soporte ORM (Generación de objetos Java desde la Base de Datos).
- ✓ Generación de bases de datos (Transformación de diagramas de Entidad-Relación en tablas de Base de Datos).
- ✓ Generador de informes para generación de documentación.
- ✓ Incorpora soporte para trabajo en equipo, que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros.

2.6.2. Rational Rose

El Rational es un software propietario, de desarrollo de modelado visual que integra y completa todo el ciclo de vida del desarrollo del software. A través del Rational Rose muchas organizaciones en el mundo han garantizado la producción de software en mucho menos tiempo. (22)

El Rational Rose es una ayuda invaluable para aminorar los esfuerzos de desarrollo, ya que unifica todos sus equipos de desarrollo a través del modelamiento, el cual está basado en UML. Esto significa que con Rational Rose, todo el equipo corporativo puede comunicarse con un lenguaje y una herramienta.

Este software acelera la implementación para soluciones que se apoyen en Java, Web y XML, ya que automatiza los modelos arquitectónicos para cada solución. Además establece una plataforma para automatizar arquitecturas de mejores prácticas hechas a la medida de soluciones tecnológicas específicas.

Se debe destacar además que a través de esta herramienta de modelado se puede visualizar, entender, y refinar sus requerimientos y arquitectura antes de enfrentarse al código, lo que le evita esfuerzos al equipo de desarrollo. En la actualidad el Rational domina el mercado de herramientas para el análisis, modelamiento, diseño y construcción orientada a objetos. (22)

2.7. Herramienta de desarrollo SQL Manager 2005 for PostgreSQL

EMS SQL Manager for PostgreSQL es una poderosa herramienta gráfica para la administración y desarrollo del servidor de bases de datos PostgreSQL. PostgreSQL Manager funciona con cualquier

versión de PostgreSQL, hasta la 8.1, y soporta todas las nuevas características de PostgreSQL incluyendo espacios de tablas (tablespaces), argumentos nombrados (named arguments) en funciones y otras más. Ofrece una gran variedad de herramientas poderosas para usuarios avanzados, tales como Visual Database Designer (diseñador visual de Base de Datos), Visual Query Builder (constructor visual de consultas), y un poderoso editor de objetos binarios (BLOB) para satisfacer todas sus necesidades. (23)

PostgreSQL Manager cuenta con una nueva y avanzada interfaz gráfica de usuario con un sistema asistente bastante descriptivo.

Características principales:

- ✓ Soporte completo de PostgreSQL hasta la versión 8.1;
- ✓ Nueva interfaz gráfica de usuario último modelo;
- ✓ Ágil navegación y administración de Base de Datos;
- ✓ Administración sencilla de todos los objetos PostgreSQL;
- ✓ Herramientas de manipulación de datos avanzada;
- ✓ Administración efectiva de seguridad;
- ✓ Excelentes Herramientas visuales y de texto para elaboración de consultas;
- ✓ Acceso al servidor PostgreSQL a través del protocolo HTTP;
- ✓ Conexión vía reenvío de puerto local a través del túnel SSH;
- ✓ Impresionantes opciones de exportación e importación de datos;
- ✓ Poderoso diseñador visual de Base de Datos;
- ✓ Asistentes fáciles de usar que realizan mantenimiento de tareas de PostgreSQL.

2.8. Análisis de las Herramientas seleccionadas.

A partir de la descripción de cada una de las herramientas anteriores, se puede concluir que con el objetivo de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante para el proyecto en cuestión, se trabajará con el Gestor de Base de Datos PostgreSQL, teniendo en cuenta esencialmente, además de las múltiples facilidades que brinda, y que es un SGBD libre, que se puede definir como una Base de Datos objeto-relacional, ya que incluye funcionalidades como pueden ser: clases, herencias, tipos y funciones, las que brindan múltiples facilidades para el manejo de la información.

Teniendo en cuenta el alcance del proyecto y de los objetivos que se proponen, se ha elegido como metodología para la implementación de la solución propuesta: RUP. Esta metodología está enfocada al cliente y es la más adaptable para proyectos a largo plazo, dado que requiere un equipo de trabajo capaz de administrar un proceso complejo en varias etapas, que en el caso de esta investigación radica en la etapa de elaboración, donde su mayor peso se encentra en el flujo de trabajo de Análisis y Diseño, según plantea RUP donde se construye el diagrama de clases persistentes y modelo de datos para el desarrollo de la Base de Datos. El rol desempeñado dentro de esta metodología es el de Diseñador de Base de Datos, el cual es el responsable de la definición de los detalles del diseño de la Base de Datos incluyendo:

- ✓ Tablas.
- ✓ Índices.
- ✓ Vistas.
- ✓ Restricciones.
- ✓ Disparadores.
- ✓ Procedimientos almacenados.

Este rol también incluye tener conocimientos sólidos sobre:

- ✓ Modelado de datos.
- ✓ Diseño de bases de datos.
- ✓ Técnicas de análisis y diseño orientado a objetos.
- ✓ Administración de bases de datos.

El Propósito fundamental de la actividad Diseñar la Base de Datos es asegurarse de que los datos persistentes son almacenados consistente y eficientemente y definir el comportamiento que debe ser implementado en la Base de Datos.

Para el modelado, unido a la metodología elegida, se plantea UML, que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos, garantizando la relación entre el cliente y el desarrollador de software.

Para el aseguramiento de la construcción de un sistema correcto, se utiliza una herramienta de modelado, en este caso, el Visual Paradigm, herramienta Case que cuenta con una versión gratuita denominada Community Edition, con vista a un posible exportación del software a herramientas completamente libres. Esta herramienta permite el modelamiento del MER, con posibilidad de conexión al Gestor de Base de Datos y exportación de las entidades hacia el mismo. Además se puede aprovechar el hecho de que la herramienta genera la documentación que se le indique a partir del diseño realizado.

Con el objetivo de la implementación de las funcionalidades que se han propuesto en esta investigación, se utiliza el EMS SQL Manager 2005 for PostgreSQL. Aplicación de alto desempeño para la administración y desarrollo de los servidores de BD para PostgreSQL, unido por supuesto a cada una de las funcionalidades que incluye y que se mencionaron anteriormente.

2.9. Técnicas de Base de Datos

Las técnicas de Base de Datos surgen al intentar superar problemas como el bajo rendimiento y fiabilidad de las aplicaciones. El objetivo prioritario es unificar toda la información del sistema para evitar redundancias, sin perder las distintas perspectivas que de la misma tienen los usuarios; agregando al sistema independencia, concurrencia y privacidad. (3)

2.9.1. Data Warehouse

Una de las principales problemáticas al implantar un sistema en una empresa es que la información es dispersa, heterogénea y muchas veces incoherente. Las empresas emplean muchas aplicaciones distintas para cada uno de sus departamentos y seguramente cada una guarda diferentes datos de los clientes, con distintos formatos y en diferentes bases de datos siendo incluso posible que haya incoherencias, como el caso de un cliente que haya cambiado de dirección y dicha información sólo esté actualizada en uno de los departamentos. El paso necesario para combinar la información de las diferentes aplicaciones departamentales de cada empresa es tomar dicha información, depurarla, y traspararla a un lugar donde sea almacenada de igual manera para cada uno de ellos, garantizando la uniformidad y consistencia de la misma. Este lugar es llamado *Data Warehouse*, y no es más que el

almacén de datos que reúne la información histórica generada por todos los distintos departamentos de una organización, orientada a consultas complejas y de alto rendimiento (24). Pretende conseguir que todos los departamentos puedan acceder a la información que manejan otros de una forma centralizada, controlada y uniforme y garantizar que todas las operaciones tengan el mismo significado para cada uno de ellos.

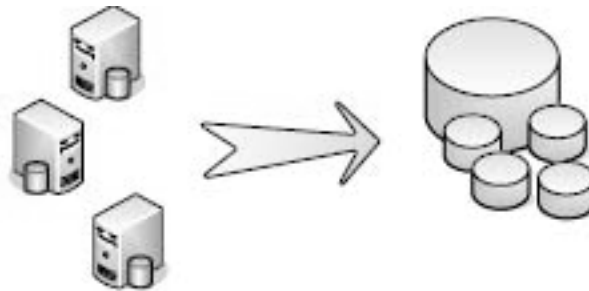


Figura #4. Esquema de un Data Warehouse.

Los Data Warehouse proporcionan una visión global, común e integrada de los datos de la organización, independiente de cómo se vayan a utilizar posteriormente por los consumidores o usuarios, con las propiedades siguientes (25):

- ✓ Integración: Integra datos provenientes de distintos sistemas operacionales. Los datos se almacenan en un formato consistente y existe un único esquema de representación.
- ✓ Orientada a sujetos: Los datos se organizan y se resumen por temas, tales como ventas, finanzas y transportación, para cada uno de los cuales se contiene sujetos, tales como productos, compradores y regiones. Por tanto, un Data Warehouse se enfoca a las entidades del negocio, lo cual contrasta con los sistemas operacionales que se orientan a los procesos.
- ✓ Variante en el tiempo: Los datos se asocian con un punto en el tiempo o con un período. La toma de decisiones se apoya en diferentes modelos (estadísticos o de otro tipo) que necesitan información histórica. Esta característica básica de los datos en un Data Warehouse difiere del comportamiento en el ambiente operacional donde los datos reflejan exactamente el momento actual.

- ✓ No volátil: Los datos no se modifican una vez introducidos (sólo lectura). Ello permite la optimización del acceso a los datos, puesto que el sistema no tiene que efectuar frecuentemente los chequeos de integridad requeridos por las operaciones de modificación. Además, se garantiza la disponibilidad de datos históricos.

Los lineamientos de la implementación de un almacén de datos están dados por las siguientes características (26):

- ✓ Recolección y análisis de requerimientos.
- ✓ Creación del modelo de datos y su diseño físico.
- ✓ Definición de los orígenes de los datos.
- ✓ Selección de la tecnología de Base de Datos y hardware a utilizar.
- ✓ Extracción de los datos desde sistemas operacionales, su limpieza, transformación y carga al Data Warehouse.
- ✓ Selección de las formas de acceso, herramientas de análisis, reporte y presentación.
- ✓ Desarrollo de los reportes y aplicaciones necesarias.
- ✓ Actualización del Data Mart.

2.9.2. Data Mart

A pesar de las grandes ventajas del Data Warehouse, parecen existir unas importantes barreras para su utilización en empresas de tamaño mediano. Los productos Data Warehouse han nacido para resolver problemas de análisis de grandes masas de información, en empresas donde una pequeña diferencia en el valor de una variable, puede afectar la cuenta de resultado con unas diferencias de millones de dólares. Para resolver este tipo de necesidades han surgido los Data Mart, productos que utilizan la tecnología Data Warehouse adaptada a las necesidades de las empresas medias. A diferencia de los Data Warehouse, un Data Mart es un almacén de datos históricos relativos a un departamento de una organización, así que puede ser simplemente una copia de parte de un Data Warehouse para uso departamental. (24)

La estrecha relación estructural que existe entre el diseño de los Data Warehouse y los Data Marts, permite que se empleen en estos últimos los mismos lineamientos que para los Data Warehouse. Pese a esta relación, los Data Marts no se pueden considerar como un Data Warehouse en escala inferior, ya que ellos están diseñados para satisfacer las necesidades específicas de los departamentos o divisiones en las empresas, esto permite asegurar que sin lugar a duda los Data Marts utilizan un planteamiento de “divide y vencerás”, que a menudo es la solución, cuando la Data Warehouse crece desmedidamente, a tal punto que se hace incontrolable su operación. (26)

Los Data Marts por las consideraciones mencionadas, vienen a ser una excelente herramienta de análisis de datos y soporte para la toma de decisiones para las pequeñas y medianas empresas, por su versatilidad, corto tiempo de desarrollo y bajo costo económico, así como la obtención de los resultados esperados a un corto plazo. Para tener un panorama general sobre los ellos, es preciso pasar a especificar de forma clara y detallada cada uno de los tres modelos de patrones de desarrollo así como sus variaciones. (26)

Modelo Top Down

Todos los datos necesarios para el apoyo a la toma de decisiones se encuentran en el Data Warehouse después de que ha sido implementado con la información procesada, solamente es necesario distribuir los datos según las necesidades de información por departamento, generando los Data Marts como subgrupos de datos específicos para los requerimientos particulares de cada departamento.

Los Data Marts en este modelo son derivados del Data Warehouse y debido a esta relación se da una inconsistencia al querer definir una nueva necesidad de información para un departamento, ya que en este caso es necesario modificar primero el Data Warehouse para que ocurra el respectivo cambio en el Data Mart.

Modelo Bottom Up

En este modelo los Data Marts se construyen a partir de los datos dispersos y la Data Warehouse se construye a partir de los Data Marts existentes. Esta construcción se realiza a través de dos procesos diferentes de extracción, transformación y transportación.

En el primer proceso cada Data Mart se construye con los datos aislados que son necesarios para satisfacer las funciones del departamento por medio de los procesos de extracción, transformación y transportación. Es necesario enfatizar que por la naturaleza de su diseño no existe ninguna relación entre un Data Mart y cualquier otro.

Un segundo proceso de tratamiento de datos (Extracción, Transformación y Transportación) ocurre en el sentido de los Data Marts hacia la Data Warehouse. Este Data Warehouse que ha sido construido a través de los diversos Data Marts, contendrá toda la información que la empresa requiera de acuerdo a la necesidad o análisis que quiera realizar. (26)

Modelo Paralelo

El diseño del modelo paralelo se basa en dos alternativas, en la primera se tratan los Data Marts con entidades independientes de los Data Warehouse y en la segunda, esta independencia se trata de forma temporal.

En el primer caso se enfoca la construcción de los Data Marts como entidades independientes de la Data Warehouse, utilizando el modelo de construcción de esta. Esto consiste en utilizar el mismo modelo de datos que se emplea en la construcción del Data Warehouse propuesto por la empresa. La ventaja de que este modelo funcione de esta forma es que permite detectar y controlar problemas, como la falta y la redundancia de información presentes en el modelo de la Data Warehouse. Esto permite el mejoramiento del modelo de construcción en los futuros Data Marts y en el mismo Data Warehouse.

Este modelo puede verse desde otro punto de vista donde los Data Marts se construyen con una independencia temporal de la Data Warehouse, lo que indica que una vez que están implementados pasan a ser parte de ella, como un subconjunto de datos que conforma el Data Warehouse que existe en la empresa. (27)

2.9.3. OLAP

OLAP hace referencia al proceso de análisis de información útil para la toma de decisiones desde fuentes de datos especialmente estructuradas para este objetivo como son los Data Warehouse y los Data Marts; dichas fuentes de datos tienen una concepción multidimensional, que las hace muy difícil de implementar en los actuales Sistemas de Bases de Datos que poseen un enfoque relacional. Por

este motivo, ROLAP aparece como una alternativa de implementación y análisis de estructuras multidimensionales sobre modelos relacionales. La modelación multidimensional de datos es una forma de facilitar el análisis empresarial en línea y de mejorar el rendimiento de las consultas. El Administrador de OLAP permitirá convertir los datos almacenados en bases de datos relacionales en información empresarial significativa y fácil de explorar con sólo crear un cubo de datos. Una estructura multidimensional de tres dimensiones se podría representar como se muestra en la Figura #5, donde cada dimensión representa una de las características que debe poseer el valor a medir, siendo en este ejemplo, producto, sucursal y mes. El punto de intersección de dichas dimensiones o características determinaría un valor específico de ventas de un producto, para un mes particular, en una sucursal determinada, que en definitiva es el hecho que se está midiendo.

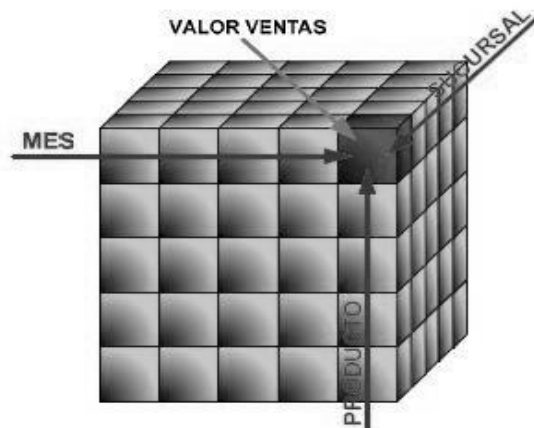


Figura #5. Estructura de datos de tres dimensiones.

Este cubo multidimensional, permite analizar la información de la manera que se desee. Se podrán cruzar todas las dimensiones para obtener nueva información que responda a las preguntas que se quieran hacer y permitirá tomar mejores decisiones.

Existen dos operaciones básicas que se pueden realizar en un cubo OLAP:

Rotar y Rebanar: Esta operación permite cambiar las dimensiones del cubo que está viendo y obtener una nueva vista de información. Por ejemplo, 'Ventas por producto' puede cambiarse fácilmente a 'Ventas por vendedor'. Rebanar es cambiar el valor de una dimensión por otro valor, por ejemplo, de las ventas de enero a las ventas de febrero. Rotar es aventar el cubo como si fuera un dado para obtener una nueva cara del cubo.

Taladrar o Drilling: Los datos de las dimensiones se pueden abrir para obtener más detalle. Una especie de taladro que se hunde más en la información. Si usted ve información geográfica, puede pasar de un continente a un país y luego a una ciudad en particular.

Con esta simple combinación de cosas, se puede extraer la información generada por un negocio o información corporativa para todo el personal tomador de decisiones, en formas que antes no era posible realizar.

Existen distintos modelos para el diseño de cubos multidimensionales como son el Esquema Estrella, Copo de Nieve y el Híbrido. Dentro de estos el más utilizado es el Esquema Estrella debido a que es el más sencillo de implementar. Está orientado a la representación de dimensiones que convergen sobre un hecho común o valor cuantificable, logrando implementar en un sistema de bases de datos tradicional el concepto de cubos multidimensionales. La implementación, como se mencionaba anteriormente es bastante simple. Existe una tabla central denominada tabla de hecho que representa el atributo cuantificable, alrededor de la cual se definen otras tablas ligadas a esta mediante claves foráneas, estas tablas se denominan tablas de dimensiones y son aquellas que definen cualidades del atributo representado por la tabla de hechos Figura #6. Finalmente la clave primaria de la tabla de hecho será una clave compuesta por todas las claves foráneas que representan las claves primarias de cada una de las dimensiones asociadas. Las dimensiones pueden tener una distribución jerárquica dependiendo del nivel de detalle deseado para cada hecho, como por ejemplo una dimensión tiempo que se distribuya jerárquicamente en año, semestre, mes y semana, siendo año el nivel más resumido y semana el nivel de mayor detalle.

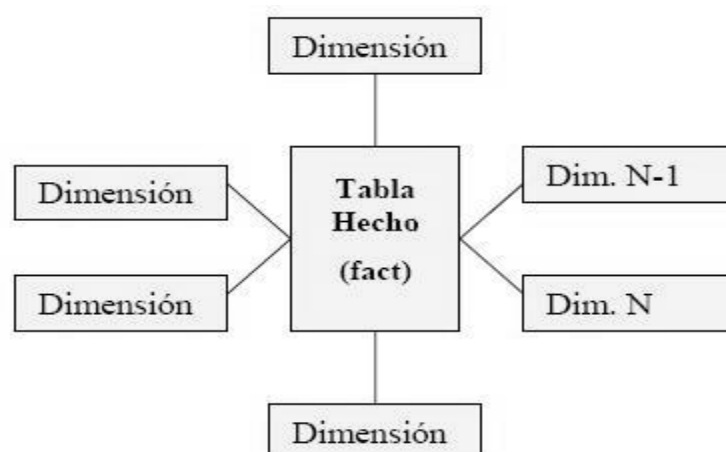


Figura #6. Esquema Estrella.

Las jerarquías de dimensiones están agrupadas en niveles que constan de los miembros de una dimensión Figura #7. Podrá unir los niveles de una dimensión para formar los valores de los que constará el siguiente nivel superior. Por ejemplo, en una dimensión temporal, los días se unen en meses y los meses forman trimestres. (28)

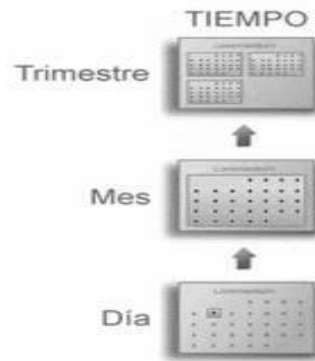


Figura #7. Dimensiones y Jerarquía.

2.9.4. Data Mining

El Data Mining surge como una tecnología que intenta ayudar a comprender el contenido de una Base de Datos. De forma general, los datos son la materia prima bruta. En el momento que el usuario les atribuye algún significado especial pasan a convertirse en información. Cuando los especialistas elaboran o encuentran un modelo, haciendo que la interpretación del confronto entre la información y ese modelo represente un valor agregado, entonces esto se refiere al conocimiento. En la Figura #8 se ilustra la jerarquía que existe en una Base de Datos entre dato, información y conocimiento.

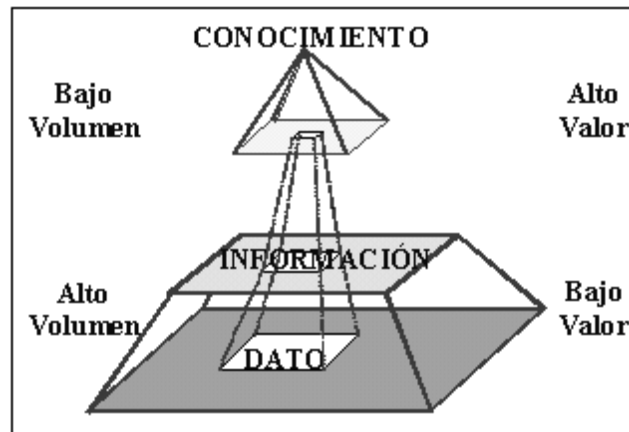


Figura #8. Relación entre dato, información y conocimiento.

Conceptualmente, Data Mining es "un proceso no trivial de identificación válida, novedosa, potencialmente útil y entendible de patrones comprensibles que se encuentran ocultos en los datos". (29)

Es una de las principales herramientas que se utilizan dentro de los programas de gestión del conocimiento como soporte a la toma de decisiones. El fin es la extracción de información oculta o análisis de datos mediante técnicas estadísticas de grandes bases de datos. Las herramientas de Data Mining pueden responder a preguntas de negocios empresariales a priori no planteadas o que pueden consumir demasiado tiempo para ser resueltas. Conceptualmente Data Mining es el conjunto de técnicas y herramientas usadas para encontrar y entender relaciones en una gran cantidad de datos y presentarlas en una forma útil y ventajosa. (30)

O sea, cuando una empresa ha acumulado una gran cantidad de datos en su interacción con los clientes, podría usar estos datos para conocer mejor a estos clientes y tomar mejores decisiones. Y ¿Cómo haría esto? La mejor manera, sobre todo, si la cantidad de datos es muy grande, sería usar programas especiales que le permitan no sólo obtener promedios y sumas, sino también que sean capaces de encontrar relaciones útiles. Por ejemplo, si el porcentaje de clientes que se cambia a la competencia es preocupante, sería conveniente conocer cuáles son los factores que influyen en sus decisiones para poder actuar de manera que se logre disminuir las bajas.

Actualmente los proyectos de Data Mining se utilizan para responder diversas cuestiones. Por ejemplo:

- ✓ ¿Cuáles son los mejores productos para venderles a mis clientes?

- ✓ ¿Cuál es la probabilidad de que un cliente compre un determinado producto?
- ✓ ¿Cuál es la mejor manera de agrupar o segmentar a mis clientes?
- ✓ ¿Qué características comparten mis mejores clientes?
- ✓ ¿A cuáles clientes puedo otorgarles crédito y a cuáles no?
- ✓ ¿Cuál es la probabilidad de que un cliente esté cometiendo un fraude?
- ✓ ¿Cuál es el riesgo de perder este cliente?

Cada una de estas técnicas es de gran importancia en los negocios actuales de las grandes y medianas empresas, fundamentalmente para el apoyo y soporte a la toma de decisiones, contribuyendo favorablemente al desarrollo de las empresas que las ponen en práctica.

2.9.5. Bases de Datos Activas

El paradigma de bases de datos activas planteado por Morgenstern² en 1983, describe la noción de que una Base de Datos sea activa, como una metáfora de su comportamiento, el cual se concentra en: la dinámica de la interacción con los usuarios, unido a la inteligencia de la Base de Datos para lidiar con las consecuencias e implicaciones de esa interacción. (31) A partir de lo que se puede definir un sistema de Bases de Datos activa, como:

Un sistema de gestión de bases de datos que contiene un subsistema que permite la definición y la gestión de reglas de producción (reglas activas). En muchas aplicaciones, la Base de Datos debe evolucionar independientemente de la intervención del usuario como respuesta a un suceso o una determinada situación. En los sistemas de gestión de bases de datos tradicionales (pasivas), la evolución de la Base de Datos se programa en el código de las aplicaciones, mientras que en los sistemas de gestión de bases de datos activas esta evolución es autónoma y se define en el esquema de la Base de Datos.

² 1902-1976. Nacido en Gorlitz, Silesia, estudia en las universidades de Viena, Harvard y Nueva York. Miembro de la Escuela Austriaca y avezado matemático.

El poder especificar reglas con una serie de acciones que se ejecutan automáticamente cuando se producen ciertos eventos, es una de las mejoras de los sistemas de gestión de bases de datos que se consideran de gran importancia desde hace algún tiempo. (31)

Se ha hecho mucha investigación sobre lo que debería ser un modelo general de bases de datos activas desde que empezaron a aparecer los primeros disparadores. El modelo que se viene utilizando para especificar bases de datos activas es el modelo evento–condición–acción (Modelo ECA).

A través de las bases de datos activas al encontrarse las reglas definidas como parte del esquema de la Base de Datos, se comparten por todos los usuarios, en lugar de estar replicadas en todos los programas de aplicación. Cualquier cambio sobre el comportamiento reactivo se puede llevar a cabo cambiando solamente las reglas activas, sin necesidad de modificar las aplicaciones. Además, mediante los sistemas de bases de datos activas se hace posible el integrar distintos subsistemas.

El problema principal en el diseño de las bases de datos activas está en entender el comportamiento de conjuntos complejos de reglas. Las propiedades principales de estas reglas son terminación, confluencia e idéntico comportamiento observable.

- ✓ Un conjunto de reglas garantiza la *terminación* cuando, para cada transacción que puede activar la ejecución de reglas, esta ejecución produce un estado final en un número finito de pasos.
- ✓ Un conjunto de reglas garantiza la *confluencia* cuando, para cada transacción que puede activar la ejecución de reglas, la ejecución termina produciendo un estado final único que no depende del orden de ejecución de las reglas.
- ✓ Un conjunto de reglas garantiza un *comportamiento observable idéntico* cuando, para cada transacción que puede activar la ejecución de reglas, esta ejecución es confluyente y todas las acciones visibles llevadas a cabo por la regla son idénticas y producidas en el mismo orden.

2.9.5.1 Triggers

Casi todos los sistemas relacionales incorporan reglas activas simples denominadas disparadores (*Triggers*), que se define también como rutinas autónomas asociadas con una tabla o vista que automáticamente realiza una acción cuando una fila en la tabla o la vista se inserta (INSERT), se actualiza (UPDATE), o se borra (DELETE).

Un Trigger nunca se llama directamente. En cambio, cuando una aplicación o usuario intenta insertar, actualizar, o anular una fila en una tabla, la acción definida en el disparador se ejecuta automáticamente (se dispara).



Figura #9. Estructura de un Trigger.

Hay diferencias importantes en el modo en que cada sistema define la activación, consideración y ejecución de disparadores. Los disparadores relacionales tienen dos niveles de granularidad: a nivel de fila y a nivel de sentencia. En el primer caso, la activación tiene lugar para cada tupla involucrada en la operación y se dice que el sistema tiene un comportamiento orientado a tuplas. En el segundo caso, la activación tiene lugar sólo una vez para cada sentencia SQL, refiriéndose a todas las tuplas invocadas por la sentencia, con un comportamiento orientado a conjuntos. Un disparador puede activar otro disparador. Esto ocurre cuando la acción de un disparador es también el evento de otro disparador. En este caso, se dice que los disparadores se activan en cascada. Los triggers constituyen las herramientas más potentes para el mantenimiento de la integridad de la Base de Datos, ya que pueden llevar a cabo cualquier acción que sea necesaria para mantener dicha integridad.

Las ventajas de usar los Triggers son:

- ✓ La entrada en vigor automática de restricciones de los datos, hace que los usuarios entren sólo valores válidos.
- ✓ El mantenimiento de la aplicación se reduce, los cambios a un Trigger se refleja automáticamente en todas las aplicaciones que tienen que ver con la tabla sin la necesidad de recompilar o relinquear.
- ✓ Logs automáticos de cambios a las tablas. Una aplicación puede guardar un registro corriente de cambios, creando un Trigger que se dispare siempre que una tabla se modifique.
- ✓ La notificación automática de cambios a la Base de Datos con alertas de evento en los Triggers.

2.10. Análisis de las técnicas Seleccionadas

Después de haber efectuado un análisis minucioso sobre las principales técnicas de Base de Datos que podrían dar respuesta al problema científico de esta investigación, sólo resta seleccionar cuál o cuáles de ellas son realmente factibles. Como se había visto en los epígrafes anteriores, Data Warehouse, Data Mart, OLAP y Data Mining son un conjunto de técnicas empleadas principalmente en el ámbito empresarial para la recuperación, homogenización, almacenamiento y consulta de la información que se maneja en un negocio, constituyendo herramientas fundamentales para la toma de decisiones y el mejor funcionamiento de la institución. Los Data Warehouse, por ejemplo, son los encargados de almacenar grandes volúmenes de datos provenientes de los distintos departamentos (Data Marts) de manera que puedan ser accedidos de igual forma por todos, garantizando que los datos no sean redundantes o incoherentes.

En principio se había pensado en el estudio de estas técnicas con el objetivo de almacenar de forma efectiva los datos que integran la fórmula por la que están compuestos los indicadores, de forma que se pudiera verificar que cada uno de ellos estuviese registrado en el sistema. En otras palabras, se necesita un mecanismo de almacenamiento que permita la incorporación de nuevos datos para la fórmula de los indicadores y que monitoree y reaccione ante situaciones que pudieran presentarse en la Base de Datos, como pudiera ser la inexistencia de un dato necesario para conformar la fórmula de un indicador insertado o modificado por el usuario. Como se puede apreciar mediante el análisis de lo que es un Data Warehouse y un Data Mart, y el uso que ambos tienen en el campo del almacenamiento de la información, no es difícil inferir que las funcionalidades que ellos brindan no cubren las necesidades planteadas, o bien no es recomendable su utilización en este caso. El alcance y valor real de estas técnicas no reside precisamente en el hecho de que puedan o no almacenar convenientemente la información sobre algún objeto en específico, sino en el alto rendimiento que tienen en el manejo de enormes volúmenes de información para su consulta en la toma de decisiones empresariales. Por este motivo también se descarta el uso de OLAP que no es más que el proceso en el cual se analiza toda esta información que se encuentra en el Data Warehouse o el Data Mart con el objetivo de lograr la mayor eficiencia posible en el momento en que es consultada.

Al igual que OLAP, Data Mining es utilizado mayormente a nivel empresarial para la ayuda en la toma de decisiones, sólo que el segundo busca patrones en los datos con el objetivo de predecir algún tipo de comportamiento en determinado aspecto u objeto del negocio. Además de esto, tomando como base el estudio efectuado sobre los usos actuales de los proyectos Data Mining, ninguno de ellos

puede ser puesto en práctica en esta situación, lo cual hace que esta técnica no sea apropiada para dar solución a la problemática planteada.

Por otra parte el uso de Base de Datos Activas, constituye la única variante para conseguir que el sistema esté activo o en constante monitoreo en busca de situaciones adversas, como podría ser la incorporación de nueva información a la fórmula de los indicadores y tomar medidas correctivas en cada caso. Las principales funcionalidades que podría incorporar a SACGIR el uso de este tipo de técnicas son las siguientes:

- ✓ Verificar la existencia de todos los datos que componen la fórmula del indicador en SACGIR.
- ✓ Responder ante un evento iniciado por el usuario, como puede ser la inserción o actualización de la fórmula de los indicadores.

2.11. Conclusiones

En este capítulo se han analizado y concretado las herramientas y software que serán utilizados a lo largo de la investigación para la elaboración del código y respectivos modelos necesarios en la implementación de las funcionalidades que van a dar solución al problema planteado. Otro de los aspectos de relevante importancia ha sido el análisis de las técnicas de Base de Datos que el equipo de desarrollo ha llevado a cabo y que se ha concretado en la selección de la técnica a través de la cual se podrá dar solución al problema.

Capítulo 3 Análisis y Descripción de la Solución Propuesta

3.1. Introducción

En el presente capítulo se describe la solución propuesta, quedando plasmado en el mismo el diseño de la Base de Datos, que incluye, el DER y la descripción de aquellas entidades, que se agregaron y que complementan la propuesta de solución. Además se hace un estudio exhaustivo de los indicadores de refinación que están en vigencia en PDVSA, también se describe de forma explícita cada una de las funcionalidades a las que se plantea dar solución como parte de los objetivos que inicialmente quedaron expuestos.

3.2. Descripción de las funcionalidades

SACGIR no está preparado para la modificación o inserción de un nuevo indicador con características diferentes a los ya definidos con anterioridad por la dirección de la empresa, más aún cuando algún dato o información que lo conforma no está almacenado en la Base de Datos. Existen funcionalidades, sobre cuya base se ha planeado la propuesta de solución, esas funcionalidades, deben responder a los objetivos que se plantearon y que guardan estrecha relación con la concepción inicial de este trabajo.

A partir de lo anterior, se plantea que a través del diseño adecuado de la Base de Datos y con la implementación de triggers, se garantizará, que la propuesta planteada cubra los elementos descritos en la Tabla #2.

Nombre de la Funcionalidad	Descripción de la Funcionalidad
1- Buscar en la Base de Datos actual los elementos que conforman la fórmula de un indicador.	Una vez que se haya actualizado o insertado una fórmula de un indicador se procederá a revisar cada uno de los elementos que la conforman, para verificar si es información que el sistema maneja, o si hay que incorporarla a este.

2- Crear un nuevo dato cuando la fórmula de un indicador lo requiera.	Cuando se crea un nuevo indicador o cuando la fórmula de uno ya existente varía, y alguno de los datos que la forman no está en la Base de Datos, entonces se procede a crear un nuevo dato o variable de la fórmula, que se corresponda con el elemento en cuestión.
3- Adicionar un nuevo atributo dentro de la Base de Datos.	El usuario avanzado, en este caso el especialista, que es el que se encarga de la gestión de los indicadores incluyendo el trabajo con sus fórmulas correspondientes, es la persona que sabe en qué lugar de la Base de Datos es más conveniente adicionar este nuevo dato que necesita la fórmula del indicador, de acuerdo con las conveniencias del negocio. Este nuevo campo se puede adicionar en cualquiera de las entidades existentes en la Base de Datos.
4- Actualizar los datos de la fórmula del indicador una vez que es modificada.	Una vez que la fórmula de un indicador es modificada y comprobados cada uno de los datos que la conforman, se deben actualizar los nuevos datos de la fórmula del indicador actualizado.
5- Diseñar la Base de Datos de manera que pueda soportar la incorporación de datos de la fórmula de los indicadores que no estén almacenados en ella.	Se hace necesario modelar la Base de Datos, de manera que permita la incorporación de nuevos campos para conformar la fórmula de un indicador determinado, sin necesidad de que cuando haya que adicionar un nuevo elemento, haya que modificar todo el sistema. El nuevo elemento debe poderse adicionar, según la consideración del usuario, en una de las entidades ya existentes o en la entidad Ind_Formula_Datos.

Tabla #2. Funcionalidades que el sistema debe cubrir.

3.3. Indicadores, Tipos de Indicadores, Definiciones y Fórmula

El estudio de los indicadores es un aspecto de fundamental importancia en el marco de este trabajo debido a la dinámica cambiante de ellos y a la importante repercusión que tienen dentro del negocio de refinación para la empresa PDVSA. Para llevar a cabo esta tarea, primeramente se hace necesario organizar convenientemente y por cada una de sus clasificaciones todos los indicadores de refinación que se tienen hasta el momento en funcionamiento. Como se ha explicado con anterioridad, un indicador está dado por un nombre que lo identifica y una fórmula mediante la cual se pueden sacar o deducir resultados para cada uno de ellos. Cada uno de los datos que componen estos indicadores, con excepción de los valores numéricos, están almacenados en la Base de Datos del SACGIR o pueden ser obtenidos de ella sin ningún tipo de inconveniente, por tanto también se hace necesario identificar todos los datos que conforman las respectivas fórmulas de cada uno de ellos dentro de la Base de Datos.

3.3.1. Volumetría Mensual Calculada

La volumetría mensual calculada se refiere a todos aquellos volúmenes de crudo utilizados en la refinería (producidos y consumidos) en un mes o período de tiempo determinado.

Volumen Producido

El volumen producido es el resultado final de la suma de los volúmenes de todos los productos obtenidos en el proceso de refinación en un período determinado. El resultado final del cálculo de este indicador se expresa en m³.

Fórmula:

$$\text{Volumen Producido} = \text{VECNI} + \text{VE} + \text{EDV} + \text{CR} + \text{OC} + \text{OS} + \text{PR} + \text{EF} - \text{EI} - \text{VR}$$

VECNI = Volumen entregas Combustible Naves Internacionales.

VE = Volumen Exportado.

EDV = Entregas al Departamento de Ventas.

CR = Consumos Refinerías.

OC = Otros Consumos.

OS = Otras Salidas (Transferencias).

PR = Productos Reprocesados.

EF = Existencia Final.

EI = Existencia Inicial.

VR = Volumen Recibido.

Insumos

Los insumos no son más que productos utilizados en una refinería que se alimentan a plantas de proceso o que son incorporados directamente a mezcla con el objetivo de la obtención de productos finales con las especificaciones requeridas. El resultado final del cálculo de este indicador se expresa en m³ o en MBD³.

Fórmula:

Insumos = Volumen Exportado + Volumen Recibido

Insumos a Mezcla

Son aquellos productos que se utilizan para obtener los productos finales con una calidad requerida o específica. El resultado final de este indicador se expresan en m³ o en MBD (miles de barriles/día).

Fórmula:

Insumos a Mezcla = Productos Reprocesados + Insumos

Insumos a Proceso

Los Insumos a Proceso se refieren aquellos productos que se alimentan a plantas de procesos en una refinería. Estos pueden ser productos reprocesados o de procedencia externa a la refinería. La cantidad de estos insumos consumidos en un periodo de tiempo se expresan en m³ o en MBD (miles de barriles/día).

Fórmula:

Si: (Productos reprocesados – Volumen Recibido) > 0

Insumos a Procesos = Volumen Recibido

Si (Productos reprocesados – Recibos externos) < 0

Insumos a Procesos = Productos Reprocesados.

³ Unidad de medida para los volúmenes de productos manejados en PDVSA. Las siglas significan Miles de Barriles Diarios.

Productos Obtenidos

Se refiere a la producción obtenida por grupo de productos. La frecuencia de cálculo es mensual y acumulada. Se expresa en MBD.

Fórmula:

$$\text{Productos Obtenidos} = VP + VR + VI - CR - PR - OC$$

VP = Volumen Producido

VR = Volumen Recibido

VI = Volumen Importado

CR = Consumo Refinería

PR = Productos Reprocesados

OC = Otros Consumos

3.3.2. Seguridad Industrial

Los indicadores de seguridad industrial son aquellos que miden las ocurrencias de accidentes dentro de la empresa en un período de tiempo.

Fuerza Bruta

El indicador Fuerza Bruta expresa la cantidad de accidentes ocurridos en un período de tiempo determinado dentro de la empresa.

Fórmula:

$$\text{Fuerza Bruta} = \text{accidentes} * 1000000 / \text{horas_hombre_trabajadas}$$

Fuerza Neta

Este indicador expresa la cantidad de accidentes que han dejado incapacitados a los trabajadores en un período de tiempo determinado.

Fórmula:

$$\text{Fuerza Neta} = \text{accidentes_incapacitados} * 1000000 / \text{horas_hombre_trabajadas}$$

Severidad

La Severidad se refiere a los días perdidos por trabajadores incapacitados por accidentes de trabajos en un período de tiempo determinado.

Fórmula:

$$\text{Severidad} = \text{DPAI} * 1000000 / \text{horas_hombre_trabajadas}$$

DPAI = días_perdidos_accidentes_incapacitados.

3.3.3. Datos Mensuales Recursos Humanos

Los indicadores de datos mensuales de recursos humanos son aspecto importante dentro de PDVSA sobre todo para tener un control mensual preciso de la mano de obra y fuerza de trabajo con que se dispone.

Fuerza Laboral Contratada

La fuerza laboral contratada es aquel personal que PDVSA contrata por un período de tiempo pero que no son plantilla de la empresa.

Fórmula:

$$\text{Fuerza Laboral Contratada} = \sum \text{fuerza_laboral_contratada}$$

Fuerza Laboral Propia

A diferencia de fuerza laboral contratada este indicador hace referencia al personal que pertenece a PDVSA, es decir, que son plantilla fija de la empresa.

Fórmula:

$$\text{Fuerza Laboral Propia} = \sum \text{fuerza_laboral_propia}$$

Accidentes

El indicador Accidentes está definido como la cantidad de accidentes que ocurren dentro de la empresa en determinado período de tiempo.

Fórmula:

Accidentes = cantidad de accidentes

Accidentes, Incapacitados

El indicador Accidentes Incapacitados está dado por la cantidad de accidentes que ocurren dentro de la empresa en determinado período de tiempo y han dejado incapacitado al personal.

Fórmula:

Accidentes = accidentes_incapacitados

Accidentes, Horas Perdidas

Se refiere a las horas de trabajo que se han perdido debido a la ocurrencia de cualquier tipo de accidente.

Fórmula:

Accidentes Horas Perdidas = cantidad de horas perdidas.

3.3.4. Volumetría Mensual Manual

Volumen Exportado

Este indicador se refiere a los volúmenes de productos exportados en una fecha. Se expresa en m³.

Fórmula:

Volumen Exportado = \sum volumen exportado del producto.

Volumen Recibido

Los productos recibidos en una refinería pueden ser de procedencia externa al sistema de refinación o transferencia de alguna refinería del sistema. Se expresa en m³.

Fórmula:

Volumen Recibido = \sum VRPRC + Otros orígenes del producto.

VRPRC = Volúmenes recibidos del producto de las refinerías del circuito

Crudos Procesados

Es la cantidad total de crudo, expresada en m³ o en MBD, que ha procesado una refinería en un período de tiempo determinado, incluye los diferentes tipos de crudos: livianos, medianos y pesados, es decir, es la sumatoria de estos tres tipos de crudo.

Fórmula:

Crudos procesados = \sum crudos livianos + medianos + pesados.

Nave Tránsito Internacional

Es el volumen de combustible entregado a los buques de tránsito internacional.

Fórmula:

Naves Tránsito Internacional = \sum Volúmenes del producto entregados a naves en TI

TI = Tránsito internacional

Productos Reprocesados

Los productos reprocesados son productos que se vuelven a alimentar a la planta donde se obtuvo para continuar degradándolo y sacarle más rendimiento al mismo. Un producto se puede reprocesar siempre y cuando cumpla con los requerimientos o especificaciones para la entrada a la planta. Se expresa en m³.

Fórmula:

Productos Reprocesados = \sum Productos Reprocesados

Existencia Inicial

Es el volumen de productos que se encuentran en inventario en una refinería. Se expresa en m³.

Fórmula:

Existencia Inicial de productos = \sum Existencia inicial del Producto

Existencia Final

Volumen de productos que se encuentran en inventario en una refinería. Se expresa en m³.

Fórmula:

$$\text{Existencia final de productos} = \sum \text{Existencia final del producto}$$

Otras Salidas

Son las transferencias de productos entre las refinerías del sistema de refinación nacional. Se expresa en m³.

Fórmula:

$$\text{Otras Salidas} = \sum \text{Otras salidas del producto}$$

Otros Consumos

Son aquellos productos necesarios que apoyan al proceso productivo y que son consumidos en un período determinado. Se expresa en m³.

Fórmula:

$$\text{Otros Consumos} = \sum \text{Otros consumos del producto}$$

Consumo Refinería

Son aquellos productos que se producen en un determinado período con el objetivo de ser consumidos por la propia refinería. Constituye una producción propia para insumo. Se expresa en m³.

Fórmula:

$$\text{Consumo Refinería} = \sum \text{Consumo refinería del producto}$$

Entrega Departamento Ventas

Son los volúmenes de productos entregados al departamento de ventas para el Mercado Nacional. Se expresa en m³.

Fórmula:

Entrega Departamento Venta = \sum *Volumen entrega departamento ventas del producto*

Volumen Importado

Se refiere a todos los volúmenes de productos que se recibieron desde el exterior del país. Se expresa en m³.

Fórmula:

Volumen Importado = \sum *Volumen importado del producto*

3.4. Datos Primarios de los indicadores

Una vez analizado los tipos de indicadores, sus definiciones y fórmulas, es necesario hacer algunas acotaciones acerca de los datos que los componen, encontrar diferencias y aspectos que tienen en común y puedan ser usados como un factor decisivo en su modelación.

3.4.1. El Cálculo

Los indicadores de refinación que se utilizan en PDVSA son siempre calculados en un período mensual o diario. Esto restringe el cálculo de cada uno de los datos primarios mediante los cuales se obtiene el resultado final del indicador, pues ellos tendrán que ser obtenidos estrictamente para el período requerido del indicador al que pertenecen.

Existen además algunos indicadores como es el caso de los indicadores de Volumetría Mensual Manual que en algunos casos deben ser calculados u obtenidos para un tipo de producto en específico.

Por tanto, estos datos no pueden almacenar valores o resultados fijos, puesto que su obtención está dada por el período en que se quiera obtener el indicador y en ocasiones para un producto determinado.

Para dar solución a este inconveniente se utilizarán procedimientos almacenados en el gestor de Base de Datos que efectúen su cálculo. Dicho procedimiento recibirá un período y producto determinado para realizar la búsqueda dentro de la Base de Datos utilizando el período como restricción, devolviendo el valor resultante para el producto en cuestión (si lo requiere).

3.4.2. Tipos

La fórmula de los indicadores de refinación involucra tanto a datos que están en la Base de Datos como a otros indicadores. Un indicador puede estar perfectamente compuesto por un dato que represente un indicador, como es el caso del indicador Volumen Producido. Por esta razón, los datos por los que están compuestos los indicadores se clasifican en dos categorías.

✓ Simples

Un dato es clasificado como *simple* cuando su obtención no depende de ningún otro indicador. Es decir, cuando el dato no representa un indicador.

Ejemplo: En el Indicador Fuerza Neta el dato horas_hombre_trabajadas es simple ya que no representa un indicador, es un dato que se obtiene de una selección dentro de una o varias tablas.

✓ Compuestos

En el caso de los datos compuestos, son aquellos que para obtenerlos necesitan que se calcule un indicador, es decir que representan un indicador al mismo tiempo.

Ejemplo: En el Indicador Volumen Producido, el Volumen Exportado es un indicador también, por lo que para calcular el resultado final del Volumen Producido hay que calcular el indicador Volumen Exportado. En este caso el Volumen Exportado es un dato que a la vez es un indicador.

3.5. BDARV y Bases de Datos Activas

En aras de garantizar el diseño adecuado de la Base de Datos de manera que este brinde el soporte requerido para el cálculo de los indicadores dinámicos de refinación y al mismo tiempo garantice la incorporación al sistema de las funcionalidades antes explicadas, se llevará a cabo dicho diseño, teniendo en cuenta la tecnología Base de Datos Relacional con Atributos y Registros Variables.

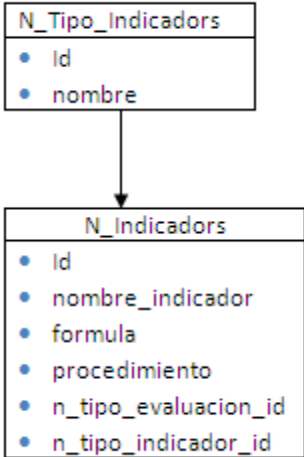
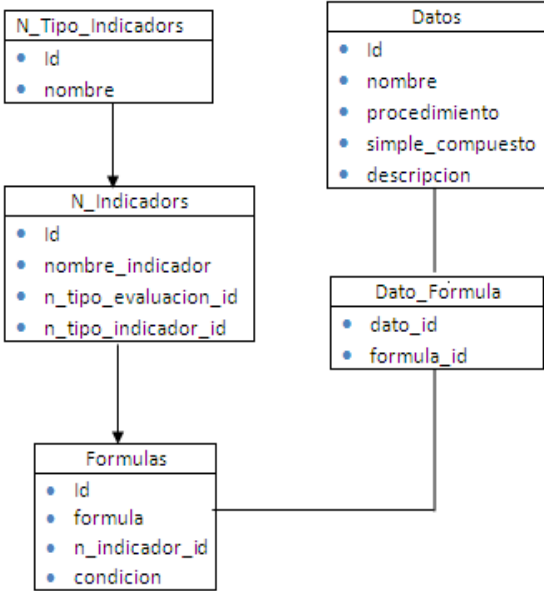
Luego del estudio realizado en el capítulo anterior acerca de las técnicas de Base de Datos y la selección de las bases de datos activas como la principal técnica para la solución del problema científico planteado, se perciben las ventajas que esta técnica ofrece unida a la BDARV, logrando cubrir el cien por ciento de las funcionalidades definidas. Dichas ventajas quedan reflejadas en:

- ✓ Disponibilidad de la Información requerida: Cuando se haga necesario el cálculo de la fórmula de un indicador, no será un problema que la misma cuente con atributos que no se encuentran almacenados en la Base de Datos, si no que por el contrario, estos campos serán agregados al sistema de forma automática.
- ✓ Mayor flexibilidad del sistema ante cambios: El sistema estará en condiciones de asimilar cualquier cambio referente a un indicador o a la fórmula que lo representa.
- ✓ Incorporación de nuevos indicadores al negocio de refinación: Mediante la modelación efectuada y la incorporación de reglas activas (Triggers) se garantiza la incorporación de un nuevo indicador con datos que no se encuentren almacenados en la Base de Datos.

3.6. Estrategia de Integración de BDARV con SACGIR

Tras haber estudiado la técnica de BDARV desarrollada por el Grupo de Servicios Informáticos de la Empresa Geominera Oriente y saber de qué se trata, se puede observar que existe mucha semejanza entre dicha tecnología y el objetivo de esta investigación con respecto a la modelación de la Base de Datos de forma que pueda adaptarse ante la inclusión de objetos que no estén modelados inicialmente en el esquema. Para BDARV en el sistema geológico, esto se evidencia en el caso de que pudieran aparecer pozos con características o datos que no se encuentran en el esquema modelado, y por otro lado, en SACGIR cuando se componga un indicador con un dato que no se encuentra almacenado en la Base de Datos. La aplicación de esta tecnología en SACGIR, adaptada a las necesidades en cuestión, brindará soporte al diseño de la Base de Datos, teniendo en cuenta la posible inclusión al sistema de nuevos indicadores, con información que inicialmente no estaba registrada.

La Tabla #3 muestra la diferencia existente entre el uso de la técnica BDARV para SACGIR con respecto a una modelación tradicional.

Clásica SACGIR	Atributos y Registros Variables SACGIR
	
<p>-Estructura fija.</p> <p>-No existe forma alguna de verificar la existencia de un dato de una fórmula para un indicador en la Base de Datos.</p> <p>-No soporta la conformación de la fórmula de un indicador, que incluya elementos que no estén almacenados previamente en la Base de Datos.</p>	<p>-Estructura dinámica.</p> <p>-Los datos de la fórmula de un indicador pasan a ser Datos (similar a las Variables Virtuales).</p> <p>-Los datos de una fórmula pueden ser buscados dentro de la entidad Datos y verificar su existencia en la Base de Datos.</p>
<p>-Un indicador sólo puede tener asociada una fórmula.</p>	<p>-Si algún dato no existiese, puede ser insertado dentro de la tabla Datos.</p> <p>-La fórmula de los indicadores puede ser modificada incluyéndole datos que no se encuentren almacenados.</p> <p>-La relación (de uno a muchos) entre los indicadores y la fórmula permite que un indicador pueda tener más de una</p>

	fórmula.
--	----------

Tabla #3. Comparación entre la estructura de Base de Datos relacional y la técnica de BDARV para SACGIR.

Las entidades expuestas en la tabla anterior son las principales dentro del módulo de indicadores que se incluyeron para efectuar una modelación que ofrezca soporte para el cálculo de los indicadores. Para poder comprender de una manera más exacta como funciona este modelo, se explicará con un nivel mayor de profundidad cada una de ellas.

3.6.1. N_Tipo_Indicadors

N_Tipo_Indicadors es la entidad que representa los distintos tipos de indicadores existentes en el negocio de refinación.

Ejemplo: Volumetría Mensual Manual, Volumetría Mensual Calculada.

3.6.2. N_Indicadors

La entidad N_Indicadors representa una instancia de un tipo de indicador (similar a la entidad Rasgos de BDARV). Contiene el nombre del indicador y el tipo de indicador al que pertenece y está relacionada con la entidad Formula, debido a que un indicador puede tener más de una fórmula.

Ejemplo: Indicador Insumos a Proceso

Tipo Indicador Volumetría Mensual Calculada

Fórmula Si Condición1 Fórmula 1 ó Si Condición2 Fórmula 2

3.6.3. Formulas

La entidad Formulas representa la fórmula a través de la cual se calculará un indicador dado. El atributo *formula* es un texto que permite que esta pueda ser obtenida y calculada una vez que se hayan buscado los datos que la componen de la Base de Datos.

*Ejemplo: (accidentes *1000000 / horas_hombre_trabajadas) es la fórmula del indicador Fuerza Bruta que pertenece al tipo de indicadores de Seguridad Industrial.*

3.6.4. Datos

La entidad Datos representa los atributos de la fórmula de los indicadores (variables virtuales). Teniendo los datos centralizados en esta tabla se garantiza que cuando se efectúe algún cambio en la fórmula de algún indicador o se inserte uno nuevo, se pueda verificar la existencia del dato en la Base de Datos.

Ejemplo: horas_hombre_trabajadas es un dato que pertenece a la fórmula del indicador Fuerza Bruta.

3.6.5. Dato_Formula

La entidad Dato_Formula es el resultado de componer la fórmula de un indicador, es decir, es la entidad que implementa la relación de mucho a mucho que se establece entre *Formulas* y *Datos* donde se tiene el identificador de la fórmula del indicador con los identificadores de los datos que lo componen.

*Ejemplo: Si $(accidentes * 1000000 / horas_hombre_trabajadas)$ es la fórmula del indicador Fuerza Bruta, la entidad Datos_Formula reflejaría que esa fórmula está compuesta por los datos accidentes y horas_hombre_trabajadas.*

3.7. Aplicación de Triggers en SACGIR

Los Trigger no pueden ser invocados directamente, sino que están asociados al comportamiento de una tabla o vista, y son ejecutados automáticamente cuando ocurre una acción del tipo INSERT, UPDATE o DELETE sobre la misma (Ver Capítulo 2). La implementación de los triggers conlleva el proceso que se muestra a continuación.

- ✓ Seleccionar la tabla sobre la cuál se va a aplicar.
- ✓ Determinar el o los eventos para los que se va a ejecutar el Trigger.
- ✓ Seleccionar la condición sobre la cual se va a ejecutar.
- ✓ Seleccionar el nivel sobre el cuál se ejecutará
- ✓ Determinar la acción que se quiere realizar cuando el Trigger se dispare.

Aplicando el modelo especificado para las bases de datos activas (evento–condición–acción) en cada uno de los pasos descritos anteriormente, el Trigger principal a implementar tendría las siguientes características:

Tabla: Formulas

La tabla Formula es la que contiene todas las fórmulas pertenecientes a los indicadores, por lo que el Trigger va a estar asociado a esta entidad y va a ser disparado cada vez que ocurran los eventos seleccionados.

Eventos para ejecutarse: Insert y Update

Al efectuarse cualquiera de los eventos de inserción o modificación de la fórmula se disparará el Trigger para ejecutar la operación requerida.

Condición: After

Las condiciones de ejecución para un Trigger pueden ser solamente de dos tipos:

Before: Cuando la función debe ser llamada antes de que ocurra el evento

After: Cuando la función debe ser llamada después de que ocurra el evento.

En este caso se usa la condición AFTER debido a que se va a trabajar una vez se hayan actualizado o insertado las fórmulas.

Nivel: Row

Existen dos niveles sobre los cuales se puede ejecutar un Trigger:

A nivel de fila (ROW): Se ejecuta para cada una de las filas afectadas, es decir, si se insertan diez filas, el Trigger se ejecutará para cada una de ellas. De esta forma se pueden acceder a todas las filas afectadas en el proceso mediante el uso del objeto NEW que es creado para los triggers disparados a nivel de fila.

A nivel de sentencia (STATEMENT): Se ejecuta una vez que se haya completado la operación. En este nivel no se pueden acceder a cada una de las tuplas afectadas.

Acción: EXECUTE PROCEDURE "public"."ejecutar_trigger"();

ejecutar_trigger () es la función llamada una vez que se haya ejecutado cualquiera de los eventos mencionados anteriormente. Es la función que engloba el tratamiento que se le va a dar a los datos insertados. Aquí se efectúan las validaciones pertinentes para cada uno de ellos.

3.8. Diseño de la Base de Datos

Como principio, unido a lo anteriormente expuesto se debe dejar establecido entonces como quedaría el diseño de la Base de Datos de SACGIR aplicando la técnica de BDARV, esencialmente en el módulo Indicadores, que es donde se realizan los cambios al diseño inicialmente establecido; estos cambios por supuesto deberán quedar reflejados en el MER General de SACGIR.

3.8.1. Diseño del MER del Módulo Indicadores

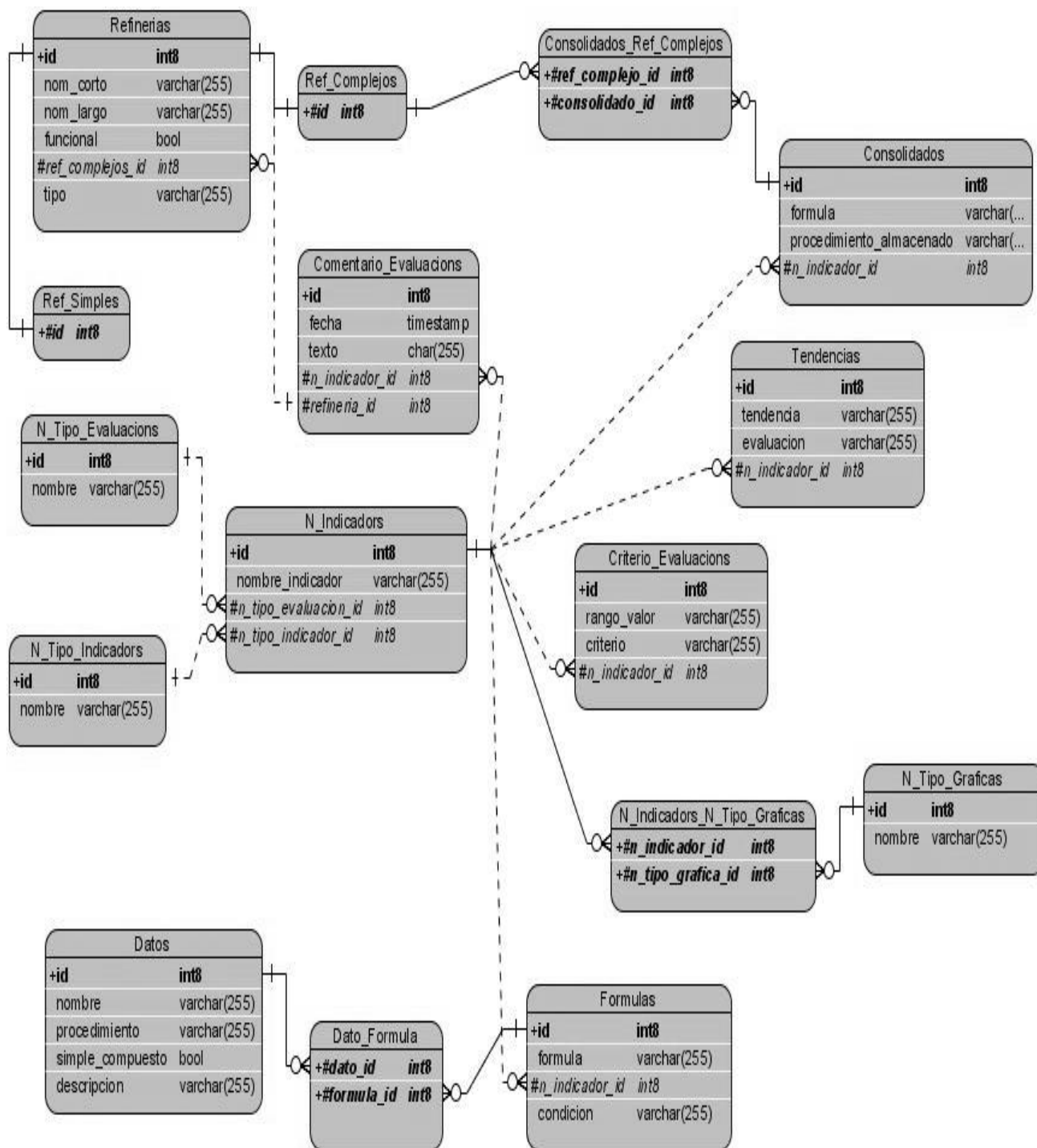


Figura #10. MER para el Módulo Indicadores de SACGIR.

3.8.2. Descripción de las Entidades de la Base de Datos

A continuación se describe en detalle cada una de las principales entidades agregadas o modificadas en el módulo de Indicadores de SACGIR, que brindan el soporte requerido para la inclusión de nuevos indicadores al negocio de PDVSA.

Nombre de la Entidad	N_Indicadors
Tipo de Clase	Clase Entidad
Descripción	Entidad que almacena todos los indicadores de refinación que se utilizan en el negocio actual de PDVSA.

Atributos			
Nombre Atributo	Tipo de Atributo	Tipo de Dato del Atributo	Descripción del Atributo
id	llave primaria	int8	Campo llave de la tabla, auto incrementable.
nombre_indicador	simple	varchar(255)	Representa el nombre del indicador.
n_tipo_evaluacion_id	llave foránea	int8	Representa el tipo de evaluación efectuada sobre el indicador, refiriéndose a su nombre e identificador.
n_tipo_indicador_id	llave foránea	int8	Representa el tipo del indicador al que se hace referencia, refiriéndose a su nombre e id.

Tabla #4. Descripción de la entidad N_Indicadors.

Nombre de la Entidad	N_Tipo_Indicadors
Tipo de Clase	Clase Entidad
Descripción	Tabla para almacenar los nombres de los tipos de indicadores existentes.

Atributos			
Nombre Atributo	Tipo de Atributo	Tipo de Dato del Atributo	Descripción del Atributo
id	llave Primaria	int8	Campo llave de la tabla, auto incrementable.
nombre	simple	varchar(255)	Nombre del tipo de indicador.

Tabla #5. Descripción de la entidad N_Tipo_Indicadors.

Nombre de la Entidad	Formulas
Tipo de Clase	Clase Entidad
Descripción	Representa la ecuación matemática que se utiliza para medir y determinar los resultados obtenidos en un complejo de refinerías. Es la entidad que almacena las fórmulas de los indicadores.

Atributos			
Nombre Atributo	Tipo de Atributo	Tipo de Dato del Atributo	Descripción del Atributo
id	llave Primaria	int8	Campo llave de la tabla, auto incrementable.
formula	simple	varchar(255)	Campo donde se almacena la fórmula en cuestión.
n_indicador_id	llave foránea	int8	Campo a través del cual se identifica el indicador al que

			pertenece dicha fórmula.
condicion	simple	varchar(255)	Representa la condición para la cual se calcularía el indicador al que pertenece la fórmula, ya que un indicador puede tener más de una fórmula pero sólo una de ellas puede ser calculada.

Tabla #6. Descripción de la entidad Formulas.

Nombre de la Entidad	Datos
Tipo de Clase	Clase Entidad
Descripción	Entidad que almacena todos los datos por los que están compuestos los indicadores.

Atributos			
Nombre Atributo	Tipo de Atributo	Tipo de Dato del Atributo	Descripción del Atributo
id	llave Primaria	int8	Campo llave de la tabla, auto incrementable.
nombre	simple	varchar(255)	Campo donde se almacena el nombre del dato de la fórmula.
procedimiento	simple	varchar(255)	Campo que almacena el nombre del procedimiento mediante el cual se obtiene el valor final del dato en cuestión.
simple_compuesto	simple	bool	Campo que almacena un valor booleano que es verdadero cuando el dato es simple o falso cuando es compuesto.(Ver epígrafe 3.4.2)
descripcion	simple	varchar(255)	Almacena una descripción del dato

			para ofrecer ayuda en el momento de conformar un indicador.
--	--	--	---

Tabla #7. Descripción de la entidad Datos.

Nombre de la Entidad	Dato_Formula
Tipo de Clase	Clase Entidad
Descripción	Tabla que surge como resultado de la relación de las Entidades Datos y Formulas. En esta entidad se almacenará cada fórmula, con los elementos que la conforman.

Atributos			
Nombre Atributo	Tipo de Atributo	Tipo de Dato del Atributo	Descripción del Atributo
id	llave Primaria	int8	Campo llave de la tabla, auto incrementable.
dato_id	llave foránea	int8	Campo donde se especifica el identificador del dato correspondiente a una fórmula determinada.
formula_id	llave foránea	Int8	Campo donde se especifica el identificador de la fórmula a la que pertenece un dato determinado.

Tabla #8. Descripción de la entidad Dato_Formula.

3.9. Implementación de las funcionalidades

Las funciones implementadas en este epígrafe, unidas a la modelación realizada para el módulo de los indicadores son aquellas que dan solución al problema científico de la investigación. Para efectuar estas implementaciones dentro del gestor de Base de Datos PostgreSQL se utilizó el lenguaje procedural PL/PGSQL descrito en el capítulo 2. (Ver epígrafe 2.3)

3.9.1. Trigger Principal formula_tr

El Trigger formula_tr se ejecuta cada vez que se inserte o actualice la fórmula de un indicador. Es el Trigger fundamental que propicia las validaciones de cada uno de los datos por los que está compuesta la nueva fórmula haciendo la llamada a la función ejecutar_trigger ().

```
CREATE TRIGGER "formula_tr" AFTER INSERT OR UPDATE ON "public"."formula" FOR EACH  
ROW  
EXECUTE PROCEDURE "public"."ejecutar_trigger"();
```

Función ejecutar_trigger ()

Esta función es la encargada de verificar la existencia de cada uno de los datos de la fórmula del indicador. Es la responsable del almacenamiento de los nuevos datos no numéricos de la fórmula dentro de la entidad Dato_Formula.

```
CREATE OR REPLACE FUNCTION "public"."ejecutar_trigger" () RETURNS SETOF trigger AS  
$body$  
DECLARE  
    dato RECORD;  
    d_f RECORD;  
    r RECORD;  
    form TEXT;  
    nuevo BOOL = false;  
BEGIN  
form = (SELECT formulas.formula FROM formula WHERE formulas.id = NEW.id);  
DELETE FROM dato_formula WHERE dato_formula.formula_id = NEW.id;  
FOR dato IN SELECT * FROM datos_formula (form)  
LOOP  
    IF(dato.datos_formula is not null)THEN  
        SELECT INTO r * FROM datos WHERE nombre = dato.datos_formula;  
        IF FOUND THEN  
            INSERT INTO dato_formula (dato_id,formula_id)  
            VALUES(SELECT * FROM id_dato(dato.datos_formula)), NEW.id);
```

```
ELSE
  IF(not es_numerico(dato.datos_formula))
  THEN
    INSERT INTO datos(nombre, procedimiento,simple_compuesto,descripcion) VALUES
(dato.datos_formula, NULL, (SELECT * FROM simple_o_compuesto(dato.datos_formula)),NULL);
    INSERT INTO dato_formula (dato_id,formula_id)
VALUES((SELECT * FROM id_dato(dato.datos_formula)), NEW.id);
  END IF;
END IF;
END IF;
END LOOP;
RETURN NULL;
END;
$body$
LANGUAGE 'plpgsql' VOLATILE CALLED ON NULL INPUT SECURITY INVOKER;
```

3.9.2. Funciones auxiliares utilizadas por la función ejecutar_trigger ()

Función datos_formula (text)

Función que recibe como parámetro la fórmula de un indicador y devuelve cada uno de los datos que la componen por separados.

```
CREATE OR REPLACE FUNCTION "public"."datos_formula" (formula text) RETURNS SETOF text
AS
$body$
DECLARE
  formula alias for $1;
  dato TEXT = "";
BEGIN
  FOR i IN 1..LENGTH(formula)
  LOOP
    IF(substring(formula,i,1)!='+' AND substring(formula,i,1)!='-')
```

```

AND substring(formula,i,1)!='*' AND substring(formula,i,1)!='/'
AND substring(formula,i,1)!='^' AND substring(formula,i,1)!='{'
AND substring(formula,i,1)!='}' AND substring(formula,i,1)!='('
AND substring(formula,i,1)!=')' AND substring(formula,i,1)!='['
AND substring(formula,i,1)!='']
THEN
    dato = dato || substring(formula,i,1);
ELSE
    IF(LENGTH(dato)>0) THEN
        RETURN NEXT trim(dato);
        dato = "";
    END IF;
END IF;
END LOOP;
RETURN NEXT trim(dato);
END
$body$
LANGUAGE 'plpgsql' VOLATILE CALLED ON NULL INPUT SECURITY INVOKER;

```

Función es_numerico (text)

El lenguaje procedural PL/PGSQL no cuenta con una función que evalúe una cadena y verifique que es un número. Esta función valida que todos los caracteres del dato que recibe sean números. Retorna verdadero si esto ocurre o falso de lo contrario.

```

CREATE OR REPLACE FUNCTION "public"."es_numerico" (dato text) RETURNS boolean AS
$body$
DECLARE
flag BOOL = FALSE;
caracter TEXT = '';
BEGIN
FOR i IN 1..LENGTH(dato)
    LOOP

```

```
caracter = substring(dato,i,1);
IF(caracter ='0' OR caracter='1' OR caracter='2' OR caracter='3' OR caracter ='4'
  OR caracter='5' OR caracter='6' OR caracter='7' OR caracter='8' OR
  caracter='9' OR caracter=',' OR caracter='.') THEN
flag = TRUE;
ELSE RETURN FALSE;
END IF;
END LOOP;
IF(flag = TRUE) THEN
RETURN TRUE;
ELSE RETURN FALSE;
END IF;
END
$body$
LANGUAGE 'plpgsql' VOLATILE CALLED ON NULL INPUT SECURITY INVOKER;
```

Función simple_o_compuesto (text)

Función que recibe un dato y verifica la existencia del mismo en la entidad n_indicadores. Si el dato es encontrado implica que es compuesto, de lo contrario es simple.

```
CREATE OR REPLACE FUNCTION "public"."simple_o_compuesto" (dato text) RETURNS boolean
AS
$body$
DECLARE dato alias for $1;
myrec RECORD;
BEGIN
SELECT INTO myrec * FROM n_indicadores WHERE n_indicadores.nombre_indicador = dato;
IF NOT FOUND THEN
  RETURN TRUE;
ELSE
  RETURN FALSE;
END IF;
```

```
END
```

```
$body$
```

```
LANGUAGE 'plpgsql' VOLATILE CALLED ON NULL INPUT SECURITY INVOKER;
```

3.9.3. Función crear_procedimiento (varchar,varchar, text[], text[], varchar)

Esta función es fundamental para la inserción de nuevos datos en el sistema, ya que permite crear el procedimiento mediante el cual se va a obtener el resultado o valor del dato adicionado. Este procedimiento recibe cinco parámetros, los cuales son:

- ✓ Nombre de la función resultante.
- ✓ Atributo a seleccionar.
- ✓ Arreglo de tabla(s) involucradas para la selección del atributo, donde la primera posición representa la tabla a la que pertenece dicho atributo.
- ✓ Arreglo de condiciones para la selección del atributo.
- ✓ Operador de selección (count, sum).

Una vez creado el procedimiento que obtiene de la Base de Datos el valor del dato, el nombre de la función resultante para el dato en cuestión, es añadido a la entidad Datos, en el atributo procedimiento para que el mismo pueda ser obtenido y calculado.

```
CREATE OR REPLACE FUNCTION "public"."crear_procedimiento" (nombre_fun varchar, atributo  
varchar, tablas text [], condiciones text [], operador varchar) RETURNS "pg_catalog"."void" AS
```

```
$body$
```

```
DECLARE
```

```
cant_tablas NUMERIC; cant_condiciones NUMERIC;
```

```
consulta TEXT; consulta1 TEXT; seleccion TEXT; seleccion1 TEXT; op TEXT;
```

```
BEGIN
```

```
consulta = 'CREATE OR REPLACE FUNCTION ' || nombre_fun || ' (fecha_d date, mes bool, producto  
varchar)
```

```
RETURNS text AS $$
```

```
DECLARE
```

```
give_back text;
```

```
begin ';
```



```
IF(operador is NULL) THEN
op = 'SUM';
ELSE op = operador;
END IF;
seleccion = ' give_back = (SELECT '|| op ||'(' ||tablas[1]||'. '|| atributo ||')FROM ';
SELECT array_upper(tablas,1) INTO cant_tablas;
SELECT array_upper(condiciones,1) INTO cant_condiciones;
FOR i in 1..cant_tablas LOOP
  IF(i=1) THEN seleccion = seleccion || tablas[i];
  ELSE seleccion = seleccion || ' , ' || tablas[i];
  END IF;
END LOOP;
IF(condiciones is not null) THEN seleccion = seleccion || ' WHERE ';
FOR i in 1..cant_condiciones LOOP
  IF(i=1) THEN
    seleccion = seleccion || condiciones[i];
  ELSE seleccion = seleccion || ' AND ' || condiciones[i];
  END IF;
END LOOP;
END IF;
seleccion1 = 'IF(producto is null) THEN
  IF(mes = true)THEN ';
  IF(condiciones is not null) THEN
    seleccion1 = seleccion1 || seleccion || ' and ' || 'date_part("MONTH",'||tablas[1]||'.fecha) =
date_part("MONTH", fecha_d ) and date_part("year", '|| tablas[1]||'.fecha) = date_part("year", fecha_d)';
  ELSE
    seleccion1 = seleccion1 || seleccion || ' where ' || 'date_part("MONTH",'||tablas[1]||'.fecha) =
date_part("MONTH", fecha_d )
    and date_part("year", '|| tablas[1]||'.fecha) = date_part("year", fecha_d)';
  END IF;
  seleccion1 = seleccion1||' );
  ELSE ';
```

if(condiciones is not null) THEN

seleccion1 = seleccion1 || seleccion ||' and '|| tablas[1]||'.fecha = fecha_d';

ELSE

seleccion1 = seleccion1 || seleccion ||' where '|| tablas[1]||'.fecha = fecha_d';

END IF;

seleccion1 = seleccion1||');

END IF;

ELSE

IF(mes = true)THEN ';

IF(condiciones is not null) THEN

seleccion1 = seleccion1 || seleccion ||' and '|| 'date_part("MONTH",||tablas[1]||'.fecha) =
date_part("MONTH", fecha_d) and date_part("year",|| tablas[1]||'.fecha) = date_part("year",
fecha_d) and productos.n_producto_id = (select id from n_productos where nombre = producto)';

ELSE

seleccion1 = seleccion1 || seleccion ||' where '|| 'date_part("MONTH",||tablas[1]||'.fecha) =
date_part("MONTH", fecha_d)

and date_part("year",|| tablas[1]||'.fecha) = date_part("year", fecha_d) and
productos.n_producto_id = (select id from n_productos where nombre = producto)';

END IF;

seleccion1 = seleccion1||');

else ';

IF(condiciones is not null) THEN

seleccion1 = seleccion1 || seleccion ||' and '|| tablas[1]||'.fecha = fecha_d and
productos.n_producto_id = (select id from n_productos where nombre = producto)';

ELSE

seleccion1 = seleccion1 || seleccion ||' where '|| tablas[1]||'.fecha = fecha_d and
productos.n_producto_id = (select id from n_productos where nombre = producto)';

END IF;

seleccion1 = seleccion1||');

end if;

end if; ';

```
consulta = consulta || seleccion1 || RETURN give_back;
END;
$$
LANGUAGE plpgsql VOLATILE; ;
EXECUTE consulta;
END
$body$
LANGUAGE 'plpgsql' VOLATILE CALLED ON NULL INPUT SECURITY INVOKER;
```

Para garantizar un mejor entendimiento de esta función, se muestra a continuación el siguiente ejemplo de la llamada a la función crear_procedimiento:

```
SELECT crear_procedimiento ('nombre_procedimiento ', 'cant_accidentes', ARRAY['accidentes'],
ARRAY['id = 1'],'SUM');
```

- ✓ nombre_procedimiento: Nombre del procedimiento.
- ✓ cant_accidentes: atributo a seleccionar dentro de la tabla accidentes.
- ✓ ARRAY['accidentes']: arreglo de tablas involucradas donde accidentes es la tabla a la que pertenece el atributo cant_accidentes.
- ✓ ARRAY['id = 1']: Arreglo de condiciones para la cual se va a obtener el valor. (WHERE)
- ✓ SUM: Operador de selección del atributo cant_accidentes.

El resultado de la llamada a la función produce un procedimiento, con la siguiente estructura:

```
CREATE OR REPLACE FUNCTION "public"."nombre_procedimiento" (fecha_d date, mes boolean,
producto varchar) RETURNS text AS
$body$
DECLARE
give_back TEXT;
BEGIN
IF(producto is null) THEN
IF(mes = TRUE)THEN
```

```
give_back = (SELECT SUM(cant_accidentes) FROM accidentes WHERE id = 1 AND
date_part('MONTH',accidentes.fecha) = date_part('MONTH', fecha_d )AND
date_part('year',accidentes.fecha) = date_part('year', fecha_d) );
ELSE
give_back = (SELECT SUM(cant_accidentes) FROM accidentes WHERE id = 1 AND accidentes.fecha
= fecha_d);
END IF;
    ELSE
    IF(mes = TRUE)THEN
give_back = (SELECT SUM(cant_accidentes) FROM accidentes WHERE id = 1 AND
date_part('MONTH',accidentes.fecha) = date_part('MONTH', fecha_d )
AND date_part('year',accidentes.fecha) = date_part('year', fecha_d) AND productos.n_producto_id =
(SELECT id FROM n_productos WHERE nombre = producto) );
ELSE
give_back = (SELECT SUM(cant_accidentes) FROM accidentes WHERE id = 1 AND accidentes.fecha
= fecha_d AND productos.n_producto_id = (select id FROM n_productos WHERE nombre =
producto));
    END IF;
END IF;
RETURN give_back;
END;
$body$
LANGUAGE 'plpgsql' VOLATILE CALLED ON NULL INPUT SECURITY INVOKER;
```

Como puede observarse en ejemplo anterior, el procedimiento generado recibe tres parámetros, de ellos:

- ✓ fecha_d: Indica la fecha para la cual se quiere obtener el valor resultante.
- ✓ mes: Este campo le indica al procedimiento si la búsqueda se va a efectuar para el mes la fecha anteriormente introducida. Representa un valor booleano que de ser verdadero implica que es mensual y diario para el caso en que sea falso.
- ✓ producto: Si el valor del dato se desea calcular para un producto en específico, esta variable tendrá el nombre del producto a buscar, de lo contrario recibe un valor de tipo null.

3.9.4. Función add_atributo(text, text, text)

```
CREATE OR REPLACE FUNCTION "public"."add_atributo" (columna text, tabla text, tipo text)  
RETURNS "pg_catalog"."void" AS  
$body$\br/>DECLARE  
BEGIN  
EXECUTE 'alter table ' || tabla || ' add column ' || columna || ' ' || tipo || ';;'  
END  
$body$\br/>LANGUAGE 'plpgsql' VOLATILE CALLED ON NULL INPUT SECURITY INVOKER;
```

3.10. Conclusiones

En el presente capítulo se han descrito las principales funcionalidades a las cuales se pretende dar solución. Dicha solución se plantea basando el análisis esencialmente en la descripción de los indicadores y componentes de sus fórmulas, que rigen actualmente el funcionamiento de PDVSA y que forman parte del negocio actual planteado por SACGIR. Otro aspecto de relevante importancia es la descripción de la aplicación de la Tecnología: Base de Datos de Atributos y Registros Variables aplicada a SACGIR, donde se explican a profundidad los cambios necesarios en el modelo de datos actual, para cumplir los aspectos planteados por dicha técnica, haciendo una descripción de las entidades que son modificadas y de las que surgen como resultado de haberla aplicado. En el capítulo se analiza además, cómo la técnica BDARV, unida a la técnica de Base de Datos Activas, con la implementación de Triggers y otros procedimientos, que también son explicados, complementan la propuesta de solución.

Conclusiones Generales

Teniendo en cuenta que con el desarrollo de este trabajo se propone la implementación de nuevas funcionalidades en SACGIR, de manera que permitan la incorporación de los datos necesarios para el cálculo de los indicadores de refinación, se puede afirmar que:

- ✓ Fueron determinadas las técnicas de Base de Datos apropiadas, para la incorporación de los datos necesarios para el cálculo de los indicadores, en este caso, la Técnica de Bases de Datos Activas y la técnica implementada por el Grupo de Servicios Informáticos de la Empresa Geominera Oriente: Bases de Datos Relacionales con Atributos y Registros Variables.
- ✓ Se realizó un estudio minucioso de los indicadores de refinación, con el objetivo de conocer los datos que componen la fórmula con que se calculan y poder definir su forma de almacenamiento, teniendo en cuenta la futura aparición de nuevos indicadores, así como nuevos datos para la conformación de su fórmula.
- ✓ Luego de haber realizado el diseño correspondiente, teniendo en cuenta la incorporación de la técnica de BDARV, en el módulo de Indicadores de SACGIR, se procedió a la implementación de las funcionalidades, a través de la definición de triggers, que son el fundamento de las Bases de Datos Activas.

Por los factores antes explicados, se puede concluir, que cada uno de los objetivos trazados fue cumplido.

Recomendaciones

A lo largo del proceso de desarrollo de este trabajo, se hizo necesaria la consulta a documentos, donde se explican algunas de las técnicas de Bases de Datos, que aunque no forman parte de la propuesta de solución, si se explican muy detalladamente en cada epígrafe correspondiente, por lo que se recomienda:

- ✓ Que el presente trabajo sirva de consulta para el estudio de las Técnicas de Bases de Batos: Data Warehouse, Data Mining, Data Marts, OLAP y Bases de Datos Activas.

En el presente además se plantea la solución, teniendo en cuenta el alcance del trabajo, sólo para el Módulo de Indicadores del sistema SACGIR, por lo que se propone:

- ✓ Extender el modelado de toda la aplicación teniendo en cuenta la técnica de BDARV.

Teniendo en cuenta los fundamentos de la investigación, se propone por último:

- ✓ Aplicar y hacer extensiva la solución propuesta, no sólo al sistema SACGIR, si no a cualquier institución, cuyo negocio incluya la gestión de indicadores.

Referencias Bibliográficas

1. Software Libre. *www.gnu.org*. [En línea] <http://www.gnu.org/home.es.html>
2. **members.fortunecity.es**. fortunecity. *Informática Jurídica.Base de Datos*. [En línea] <http://members.fortunecity.es/infojuridico/inicio.html>
3. **Gómez Ballester, Eva, y otros**. Bases de Datos 1. *www.alu.ua.es*. [En línea] <http://www.alu.ua.es/ijmr36/Conectate/Base%20Datos/Apuntes2006.pdf>
4. **Terms, IEEE Standard Dictionary of Electrical and Electronic**. daedalus. *Qué es un sistema*. [En línea] <http://www.daedalus.es/AreasISSistema-E.php>
5. **definicion.org**. definicion.org. *Definición de Indicador*. [En línea] <http://www.definicion.org/indicador>
6. **telepolis**. telepolis. *Glosario de Geografía R*. [En línea] <http://club.telepolis.com/geografo/glosario/r.htm>
7. Glosario de Términos Hidrocarburíferos . *www.ypfb.gov.bo*. [En línea] <http://www.ypfb.gov.bo/glosario.shtml>
8. **phpmyadmin.net**. phpmyadmin. *The phpMyAdmin Project*. [En línea] http://www.phpmyadmin.net/home_page/index.php
9. phpPgAdmin: administra PostgreSQL desde la web. *sentidoweb.com*. [En línea] <http://sentidoweb.com/2007/03/29/phppgadmin-administra-postgresql-desde-la-web.php>
10. **Servicios Informáticos Geominera Oriente**. Programa GeoDato. *www.geominera.co.cu*. [En línea] <http://www.geominera.co.cu/GeoDato.asp?show=ifisica>
11. **debubuntu.wordpress.com**. debubuntu. *Debubuntu's Weblog*. [En línea] 2008. <http://debubuntu.wordpress.com/2008/02/09/oracle-relational-data-base-management-system/>
12. **Pecos, Daniel**. PostGreSQL vs. MySQL. *www.netpecos.org*. [En línea] http://www.netpecos.org/docs/mysql_postgres/x57.html

13. **Equipo de desarrollo de PostgreSQL.** Manual de Usuario de PostgreSQL. *es.tldp.org*. [En línea] <http://es.tldp.org/Postgresql-es/web/navegable/user/user.html>
14. **Worsley, John y Drake, Joshua.** PostgreSQL Práctico. *www.sobl.org*. [En línea] 2001. <http://www.sobl.org/traducciones/practical-postgres/practical-postgres.html>
15. **Equipo de Desarrollo PostgreSQL.** PL/pgSQL. *www.ibiblio.org*. [En línea] <http://www.ibiblio.org/pub/Linux/docs/LuCaS/Postgresql-es/web/navegable/todopostgresql/x19283.html>
16. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. *biblioteca.uci.cu*. [En línea] 2000. <http://biblioteca.uci.cu/titdigitales.htm#igs>
17. **Sánchez, María A. Mendoza.** Metodologías De Desarrollo De Software. *www.informatizate.net*. [En línea] 2004. http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html
18. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** Lenguaje Unificado de Modelado.Manual de Referencia. [En línea] <http://biblioteca.uci.cu/titdigitales.htm#igs>
19. **Ferré Grau, Xavier y Sánchez Segura, María Isabel.** Desarrollo Orientado a Objetos con UML. *clikear.com*. [En línea] 2004. <http://www.clikear.com/manuales/uml/>
20. **Hernández Orallo, Enrique.** El lenguaje Unificado de Modelado(UML). *www.acta.es*. [En línea] http://www.acta.es/articulos_mf/26067.pdf
21. **Paradigma Visual para UML.** *Free Download Manager*. [En línea] [http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_\(M%C3%8D\)_1472_0_p/](http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_1472_0_p/)
22. **Rational Rose.** *indudata*. [En línea] http://www.indudata.com/1rational_rose.htm
23. **Free Download Manager.** Free Download Manager. *Administrador EMS SQL 2005 para PostgreSQL*. [En línea] 2006. http://www.freedownloadmanager.org/es/downloads/SME_Gerente_de_SQL_2005_para_PostgreSQL_36037_p/

24. **Castellar Busó, Vicent y de la Herrán Gascón, Manuel.** Cómo diseñar grandes variables en bases de datos multidimensionales. *www.redcientifica.com.* [En línea] <http://www.redcientifica.com/doc/doc200104190004.html>
25. **Tamargo, MSc. Lic. Lourdes Cerezal.** Primeros pasos para adentrarnos en el mundo de los Data Warehouse. *BETSIME.* [En línea] http://www.betsime.disaic.cu/secciones/tec_ia_04.htm
26. **Sánchez S, Kenneth y Martínez V, Rafael.** Generalidades y patrones de desarrollo de Data Marts. *www.intersedes.ucr.ac.cr.* [En línea] 2004. http://www.intersedes.ucr.ac.cr/07-art_14.html
27. **Firestone, J.** Data Warehouses and Data Marts: A Dynamic. *www.dkms.com.* [En línea] 1997. <http://www.dkms.com/DWDDV.html>
28. **aprendedynamics.com.** Aprende Dynamics AX. [En línea] <http://www.aprendedynamics.com/olap.html>
29. **Molina Félix, Luis Carlos.** Data mining: torturando a los datos hasta que confiesen. *www.uoc.edu.* [En línea] <http://www.uoc.edu/web/esp/art/uoc/molina1102/molina1102.html>
30. **pti.com.** ¿Qué significa Data Mining? *www.pti.com.* [En línea] <http://www.pti.com/espanol/Datamining.htm>
31. **Merche, Marqués.** *Diseño de Sistemas de Bases de Datos.* 2002.

Bibliografía

1. **ABCdatos.com.** ABCdatos. *ABCdatos.* [En línea]
<http://www.abcdatos.com/tutoriales/programacion/basesdedatos/varios.html>
2. **angelfire.com.** angelfire. *Glosario.* [En línea]
http://www.angelfire.com/crazy4/ana24caro0/index_glosario2.html
3. **baquia.com.** BAQUIA. *BAQUIA Knowledge Center.* [En línea]
<http://blogs.baquia.com/todobi/post/2006/04/06/aque-es-olap->
4. **batiburrillo.net.** batiburrillo.net. *Instalar y Configurar phpMyAdmin.* [En línea]
http://www.batiburrillo.net/adivina/adivina_sw13.php
5. **csi.map.es.** OLAP, ROLAP, MOLAP. *csi.map.es.* [En línea]
<http://www.csi.map.es/csi/silice/DW2251.html>
6. **daedaus.es.** daedalus. *DAEDALUS.* [En línea] <http://www.daedalus.es/mineria-de-datos/>
7. **Equipo de Desarrollo de PostgreSQL.** Guía del Administrador de PostgreSQL. *www.unoweb-s.uji.es.* [En línea] https://www.unoweb-s.uji.es/IS24/lista2/theList/guia_administrador.pdf
8. **euitio178.ccu.uniovi.es.** euitio178. *Bases de Datos-Tercer curso de Ingeniería Técnica Informática.* [En línea] 2006. <https://euitio178.ccu.uniovi.es/foros/download.php?id=1011>
9. **Grau, Xavier Ferré.** Desarrollo Orientado a Objetos con UML. *pub.ufasta.edu.ar.* [En línea] 2004.
http://pub.ufasta.edu.ar/fernandos/jaiio36/Simposios/SIS%202007/06%20-%20Romero%20-%20Mie%2017%20hs%20-%20Aplicacion_de_firma_digital_a_las_prestaciones_de_un_Sistema_de_Informa.pdf
10. **LSI.** Isisa. *Ejemplo Data Warehouse.* [En línea] 2000-2003.
<http://www.lsisa.com/gxintelligent/verpag.asp?pagina=dwe03&titulo=Ejemplo%20Data%20Warehouse&indice=Data%20Warehouse&imagen=dw>
11. **Masip, David.** Qué es Oracle. *www.desarrolloweb.com.* [En línea]
<http://www.desarrolloweb.com/articulos/840.php>

12. **Menéndez, Dr. Eugenio Santos.** Diseño y Optimización de Bases de Datos: Data Warehouse. *oei.eui.upm.es*. [En línea] http://www.oei.eui.upm.es/Asignaturas/BD/DYOBDEjemplo_DW.pdf
13. **Microsoft Corporation.** MSDN. *msdn.microsoft.com*. [En línea] 2008. <http://msdn.microsoft.com/es-es/library/ms174861.aspx>
14. **Mota, Soraya Abad.** Bases de Datos Activas,(apuntes de CI-5311). *www.bd.cesma.usb.ve*. [En línea] <http://www.bd.cesma.usb.ve/ci5311/sd05/apuntesParadigmasB.pdf>
15. Panorámica del sistema de gestión de base de datos MySQL. *dev.mysql.com*. [En línea] <http://dev.mysql.com/doc/refman/5.0/es/what-is.html>
16. Paradigma Visual para UML. *www.freedownloadmanager.org*. [En línea] [http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_\(M%C3%8D\)_1472_0_p/](http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(M%C3%8D)_1472_0_p/)
17. **PARIS Technologies.** OLAP. *olap.com*. [En línea] <http://www.olap.com/>
18. phpPgAdmin - Default branch. *freshmeat.net*. [En línea] <http://freshmeat.net/projects/phppgadmin/>
19. Rational Rose. *www.indudata.com*. [En línea] http://www.indudata.com/1rational_rose.htm
20. **Rubio, Gabriel Buades.** Datawarehouse. *dmi.uib.es*. [En línea] mayo de 1999. <http://dmi.uib.es/~bbuades/datawarehouse/sld001.htm>
21. **Salgado, Carlín, y otros.** VALORIZACIÓN DE LAS BASES DE DATOS DEDUCTIVAS Y DE LAS BASES DE DATOS ACTIVAS. *www.quadernsdigitals.net*. [En línea] http://www.quadernsdigitals.net/datos_web/hemeroteca/r_1/nr_502/a_6850/6850.htm
22. **Schmuller, Joseph.** Aprendiendo UML en 24 horas. *biblioteca.uci.cu*. [En línea] 2000. <http://biblioteca.uci.cu/titdigitales.htm#igs>
23. SME_Gerente_de_SQL_2005_para_PostgreSQL. *www.freedownloadmanager.org*. [En línea] 2000. http://www.freedownloadmanager.org/es/downloads/SME_Gerente_de_SQL_2005_para_PostgreSQL_36037_p/

24. **Soto, Prof Lauro.** Tecnológico. *www.MiTecnologico.com.* [En línea]
<http://www.mitecnologico.com/Main/CaracteristicasGestorBaseDeDatos>
25. **Strange, Kevin.** Data Warehouse Vr. Data Marts. *www.gartner.com.* [En línea] 1998.
<http://www.gartner.com/webletter/ibmbusint/article4/article4.htm>
26. **swgreenhouse.com.** swgreenhouse.com. *Data Mart.* [En línea]
<http://www.swgreenhouse.com/Productos/Hi-Spins/DataMart.html>
27. **Tabares, Marta Silvia.** Técnicas y Tecnologías de Bases de Datos. *unalmed.edu.co.* [En línea]
<http://www.unalmed.edu.co/~mstabare/>
28. **Velthuis, Mario Piattini.** Líneas de evolución de las bases de datos. *ati.es.* [En línea] 2000.
<http://www.ati.es/novatica/2000/145/marpia-145.pdf>
29. **W. Hansen, Gary y V. Hansen, James.** Diseño y Administración de Base de Datos. *bibliodoc.uci.cu.* [En línea] <http://bibliodoc.uci.cu/pdf/reg00071.pdf>
30. **Wolff, Carmen.** Implementando un Data Warehouse. *inf.udec.c.* [En línea]
<http://www.inf.udec.cl/revista/ediciones/edicion5/datawh.PDF>
31. **WordPress.** informationmanagement. *Information Management/Data Warehousing, Data Warehouse y Datamart.* [En línea] 7 de octubre de 2007.
<http://informationmanagement.wordpress.com/2007/10/07/data-warehousing-data-warehouse-y-datamart/>

Glosario de Términos

BD: (Base de Datos). Conjunto de datos pertenecientes al mismo contexto y almacenados sistemáticamente para su posterior uso.

BDARV: (Base de Datos Relacional con Atributos y Registros Variables). Técnica para el diseño de una Base de Datos, desarrollada por el Grupo de Servicios Informáticos de la Empresa Geominera Oriente.

BLOB: (Binary Large Objects). Abreviatura en inglés de: Grandes Objetos Binarios. Son elementos utilizados en las bases de datos para almacenar datos de gran tamaño que cambian de forma dinámica. Generalmente, estos datos son imágenes, archivos de sonido y otros objetos multimedia.

BSD: (Berkeley Software Distribution). Abreviatura en inglés de: Distribución de Software Berkeley. Se utiliza para identificar un sistema operativo derivado del sistema Unix nacido a partir de las aportaciones realizadas a ese sistema por la Universidad de California en Berkeley.

Commit: El comando COMMIT en SQL indica la finalización de una transacción de Base de Datos.

Data Mart: Es una versión especial de un Data Warehouse. Son subconjuntos de datos con el propósito de ayudar a que un área específica dentro del negocio pueda tomar mejores decisiones.

Data Mining: (Minería de datos). La minería de datos prepara, sondea y explora los datos para sacar la información oculta en ellos.

Data Warehouse: También conocido como almacén de datos, se puede definir como 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.

DBMS: (Database Management System). Abreviatura en inglés de: Sistema Gestor de Base de Datos.

DER: (Diagrama Entidad Relación). Lenguaje gráfico para describir conceptos, cuyos elementos son: entidades, relaciones y atributos.

EMS: (Electronic MicroSystems). Compañía de Tecnología de la Información, uno de sus campos de actividades es el desarrollo de software.

Gestión: Es el proceso mediante el cual se obtiene, despliega o utiliza una variedad de recursos básicos para apoyar los objetivos de la organización.

Lenguaje Procedural: Es una serie de instrucciones que se ejecutan una por una de principio a fin, a menos que una instrucción mande el control a alguna otra parte.

Log: Registro de actividad de un sistema, que generalmente se guarda en un fichero de texto, al que se le van añadiendo líneas a medida que se realizan acciones sobre el sistema.

MER: (Modelo Entidad Relación). Herramienta para el modelado de datos de un sistema de información.

OLAP: (On Line Analytical Process). Abreviatura en inglés de: Procesamiento Analítico en Línea. Solución utilizada en el campo de la llamada Inteligencia empresarial, cuyo objetivo es agilizar la consulta de grandes cantidades de datos.

OLTP: (OnLine Transaction Processing). Abreviatura en inglés de: Procesamiento de Transacciones en Línea. Es un tipo de sistemas que facilitan y administran aplicaciones transaccionales, usualmente para entrada de datos y recuperación y procesamiento de transacciones.

PDVSA: (Petróleos de Venezuela S.A). Empresa estatal venezolana que se dedica a la explotación, producción, refinación, petroquímica, mercadeo y transporte del petróleo venezolano.

PL/PGSQL: (Procedural Language/PostgreSQL Structured Query Language). Lenguaje imperativo provisto por el gestor de Base de Datos PostgreSQL.

RDBMS: (Relational Database Management System). Abreviatura en inglés de: Sistema de gestión de bases de datos relacionales. Proporciona el ambiente adecuado para gestionar una Base de Datos.

ROLAP: (Relational On Line Analytical Process). Abreviatura en inglés de: Procesamiento Analítico en Línea Relacional. Sistemas y herramientas OLAP construidos sobre una Base de Datos relacional.

SACGIR: (Sistema Automatizado para el Control de Gestión de Indicadores de Refinación).

SGBD: (Sistema Gestor de Base de Datos). Tipo de software muy específico, dedicado a servir de interfaz entre la Base de Datos, el usuario y las aplicaciones que la utilizan.

SQL: (Structured Query Language). Abreviatura en inglés de: Lenguaje de Consulta Estructurado. Un lenguaje estándar de consulta a gran cantidad de bases de datos.

Técnicas de Base de Datos: Son la herramienta informática actual para la gestión de grandes volúmenes de datos almacenados en memoria externa.