Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Título: Estudio de la factibilidad del Modelo

Objeto Relacional en los sistemas de gestión de información

Autora: Lisseidy Contreras Rodríguez

Tutores: Ing. Hector Manuel Solis Mulet

Lic. Roberto Acosta González

Ciudad de La Habana, Junio de 2009

"Año del 50 aniversario del triunfo de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.
Para que así conste firmamos la presente a los 18 días del mes de Junio del año 2009.

Lisseidy Contreras Rodríguez

Lic. Roberto Acosta González

Ing. Hector Manuel Solis Mulet

La arcilla fundamental de nuestra obra es la juventud, en ella depositamos nuestra esperanza y la preparamos para tomar de nuestras manos la bandera.

Ernesto Guevara de la Serna

DATOS DE CONTACTO

Lic. Roberto Acosta González *(raglz@uci.cu)**:*** Graduado en la especialidad de Ciencia de la Computación en el año 2007. Imparte la asignatura de Programación y ostenta la categoría docente de profesor instructor. Ha presentado ponencias y trabajos en eventos científicos nacionales e internacionales, obteniendo diferentes reconocimientos, publicaciones y premios. Es líder del Proyecto "Colaboración Médica" y dirige el Grupo de Trabajo de Base de Datos del Área Temática Sistema de Apoyo a la Salud en la Facultad 7 de la Universidad de las Ciencias Informáticas.

Ing. Hector Manuel Solis Mulet *(hmsolis@uci.cu)**:*** Recién graduado con Título de Oro de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI). Posee Categoría Docente de Instructor Recién Graduado. Ha participado en proyectos de desarrollo de Sistemas Informáticos para la Salud. Imparte la asignatura de Gestión de Software para estudiantes que cursan el cuarto año de la carrera de Ingeniería en Ciencias Informáticas en la Facultad 7 UCI. Actualmente se desempeña como miembro del Grupo de Trabajo de Base de Datos del Área Temática Sistemas de Apoyo a la Salud.

AGRADECIMIENTOS

A la Revolución y su eterno líder Fidel Castro Ruz por darme la oportunidad de formarme en esta universidad.

A mi familia: *A mi mami linda que la adoro, por ti he llegado hasta aquí y para ti van todos mis logros. A mi hermana Suleidy que es mi vida y mi ranzón de ser. A Danelis y Oslay por su gran ayuda. A mi prima Danaisy. A mi tío Juan por su apoyo incondicional. A mi abuela Dolores por el cariño y sus sabios consejos. A todo el resto de mi familia.*

A los que ya no están: *A mi papá, por dejar en mi todo lo bueno de ti. A mi abuelo materno y a mis abuelos paternos. Muy especial a papito, por su cariño sincero y desmedido.*

A mis tutores: *A mis tutores Roberto y Hector por indicarme el camino, por su constante esfuerzo, sacrificio, preocupación, dedicación, esmero, profesionalidad y entrega en la realización de esta tesis. Sin ustedes no hubiera sido posible. Gracias.*

Especialmente: *A Mónica que es una hermana para mi y a toda su familia que es la mia también. A Javier Ernesto por ser un excelente amigo. A Estrella, por hacer gala de su nombre. A Mercedes por dar los mejor de sí y a su familia, por acogernos siempre. A Juan Manuel por los consejos, la ayuda y momentos que pasamos juntos. A Maykell, gracias por tu paciencia y cariño. A Walfrido (Tito) porque su apoyo siempre fue oportuno.*

A mis compañeras de apartamento: *A todas las que han sido compañeras de apartamento y que hoy ya somos como una familia: Yudmila (Tita), Elaine, Dayrelis, Yunelis, Clenda, Ariagne, Amarilis, Idalmis, Elisa y Marisleydis.*

Amistades de Santa Clara: *A todos con los que comparto muy buenos momentos en Santa Clara.*

A mis profesores: *A todos los profesores que de una forma u otra han contribuido en mi formación profesional. Agradecer especialmente a los que recuerdo con mucho cariño: Erundina, Osmany y Juanita, ustedes fueron la base de mi carrera.*

De manera general a todas las personas que me ayudaron en la realización de esta tesis.

Gracias Dios mío por permitirme llegar.

DEDICATORIA

Hoy se hace realidad un gran sueño para mi .Pero nada hubiese sido posible sin la ayuda de una mujer especial, y a ella le dedico mi tesis:

Por ser primero madre y después mujer.

Por toda la paciencia y el cariño que nos haz dado desde que éramos pequeñas.

Por saber guiarnos por el camino correcto.

Por los buenos y malos momentos que hemos pasado juntas.

Por no cansarte ni rendirte nunca en esta gran batalla que sabes que hemos tenido que llevar.

Por dar lo mejor de ti siempre que te necesitamos.

Por ser ejemplo de madre y amiga.

Por saber tomar justo a tiempo decisiones que no son las mejores para ti, pero si para tus hijas.

Por darnos tanta confianza, respeto y amor.

Por ser siempre la prioridad primera en tu vida.

Por saber buscarle siempre una solución a los problemas, por muy difíciles que sean.

Por tú esfuerzo y desvelo día a día.

Por ayudarnos a enfrentar cada una de nuestras difíciles decisiones que la vida nos ha destinado.

Por que para ti no hay descanso si se trata de tus hijas.

Quiero que sepas que mi hermana y yo, nos sentimos muy orgullosas de ser tus hijas y la vida no alcanza para agradecerte lo que haz hecho por nosotras.

Te queremos.

RESUMEN

Desde los años sesenta los modelos de datos, evolucionaron de forma equivalente, como respuesta a la creciente complejidad que requieren las bases de datos de las aplicaciones. El Modelo Relacional es en la actualidad el más difundido y usado debido a su fuerte fundamento teórico. Como parte de su desarrollo, para aumentar las funcionalidades de las bases de datos, así como su capacidad de modelamiento, debido al creciente progreso que han tenido, surge el Modelo Objeto Relacional.

El objetivo del presente es determinar la factibilidad del Modelo Objeto Relacional para sistemas de gestión de la información. Para la realización del siguiente estudio se utilizó como Sistema Gestor de Base de Datos PostgreSQL 8.3 y como herramienta de modelado el Enterprise Architect en la versión 7.0.

El desarrollo de la siguiente investigación permitirá aumentar las funcionalidades de las bases de datos de los proyectos que se encuentran en la Universidad de las Ciencias Informáticas. Obtener una mejor representación y organización de los datos de sus sistemas, para optimizar el rendimiento de las aplicaciones que se realizan.

Palabras claves: *Base de Datos, Modelo de datos, Sistema Gestor de Base de Datos*

ÍNDICE

INTRODUCCIÓN 1

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA 6

1.1. Conceptos asociados al dominio del problema 6

1.2. Sistema de Gestión de Información 7

1.3. Las bases de datos y los Sistemas Gestores de Bases de Datos 8

1.4. Modelos de Datos 9

1.5. Modelo Objeto Relacional 10

1.6. Sistemas de Gestores de Bases de Datos 13

1.7. Herramientas de modelado visual 15

CAPÍTULO 2. DISEÑO E IMPLEMENTACIÓN 19

2.1. Análisis del problema y metodología utilizada 19

2.2. Implementación del diseño de la base de datos Objeto Relacional 27

2.2.1. Herencia 27

2.2.2. Arreglos 28

2.3. Tipos de datos compuestos 31

CAPÍTULO 3. PRUEBAS 34

3.1. Consultas de selección para el Modelo Objeto Relacional y el Modelo Relacional 35

3.2. Consultas de Inserción para el Modelo Objeto Relacional y el Modelo Relacional 36

3.3. Consultas de actualización para el Modelo Objeto Relacional y el Modelo Relacional 36

3.4. Ambiente de las pruebas 37

3.4.1 Diseño de las pruebas en la herramienta 38

3.5. Resultados de las Pruebas 39

3.5.1 Consultas de selección 39

3.5.2 Consulta de Inserción 41

3.5.3 Consultas de Actualización 43

3.6. Análisis de los Resultados 45

3.7. Conclusiones de los análisis 48

CONCLUSIONES 49

RECOMENDACIONES 50

REFERENCIAS BIBLIOGRÁFICAS 51

BIBLIOGRAFÍA 52

ANEXOS 54

INTRODUCCIÓN

Un sistema de información es la composición de elementos que operan unidos en orden de capturar, procesar, almacenar y distribuir información. Esta información se utiliza generalmente para tomar decisiones, la coordinación, el control y el análisis en una organización. Frecuentemente, el propósito básico del sistema es la gestión de la información. (1)

Los sistemas de Información (SI) están formados por: la base de datos , el Sistema Gestor de Base de Datos (SGBD), los programas de aplicación, los dispositivos físicos (ordenadores, dispositivos de almacenamiento), y el personal que utiliza y desarrolla el sistema. A través de su uso se logran grandes mejoras, pues abastecen una plataforma de información indispensable para la toma de decisiones y su implantación logra reducir la ventaja con los rivales o agrandarla.

El desarrollo de una institución es, en gran medida, el producto de la precisión y efectividad del sistema de información que soporte su funcionamiento. Un sistema con información veraz, confiable, precisa y bien estructurado garantizará el éxito de las decisiones de sus consumidores. Para obtener estos resultados, es preciso un flujo eficiente e ininterrumpido de información que sustente los procesos institucionales para que estos fluyan con libertad, a partir del quehacer de un personal dotado de los conocimientos necesarios para el desarrollo de sus actividades. (2)

Desde los años setenta, los SGBD han reemplazado a los sistemas de ficheros en los SI de las empresas, debido a que en estos la definición de los datos se encuentra codificada dentro de los programas de aplicación y no existe una manipulación y control de los datos más allá de lo impuesto por los programas de aplicación. Su desarrollo ha permitiendo un fácil acceso a los datos, a la información por parte de múltiples usuarios y la manipulación de los datos existentes en la base de datos, como por ejemplo insertar, eliminar y editar. Su adecuada utilización proporciona además eliminar las inconsistencias en los datos, para procurar el ahorro de espacio de almacenamiento, permitir poder compartir los mismos datos entre diferentes aplicaciones con distintas necesidades, mejorar la seguridad de los datos y la creación de un entorno de alta disponibilidad.

En aquella época no existía ningún sistema que permitiera gestionar la inmensa cantidad de información que se necesitaba. La primera empresa encargada del proyecto, NAA (North American Aviation), desarrolló un software denominado GUAM (General Update Access Method), el cual presentaba una estructura en forma de árbol, conocida como estructura jerárquica. Apareciendo así el Modelo Jerárquico.

La principal ventaja del Modelo Jerárquico es la eficiencia de su procesamiento. Un sistema jerárquico puede requerir menos tiempo para manipular datos que otro modelo, pues las relaciones entre los datos son menos complejas. Es apropiado utilizarlo cuando los datos forman una jerarquía natural, son difíciles de cambiar y su principal inconveniente es la poca independencia de los programas respecto a cómo están almacenados los datos, lo que dificulta además la programación de software de acceso a estos sistemas. Incluso con estas ventajas y desventajas muchas importantes organizaciones lo utilizan.

A mitad de los sesenta, se desarrolló la Integrated Data Store (IDS). Trabajo dirigido por uno de los pioneros en los sistemas de base de datos, Charles Bachmann. IDS era un nuevo tipo de SGBD, conocido como Modelo de Red, que produjo un gran efecto sobre los sistemas de información de aquella generación. El modelo se desarrolló, en parte, para satisfacer la necesidad de representar relaciones entre datos más complejos que las que se podían modelar con el jerárquico, y para imponer un estándar de base de datos. A pesas de que ofrecen más flexibilidad que los jerárquicos en terminaos de organización de datos, son más difíciles de desarrollar y usar, debido a la complejidad de las relaciones entre los datos. Aunque algunos se encuentran en funcionamiento no se han utilizado a gran escala.

El modo de ver las bases de datos cambia totalmente cuando Edgar Frank Codd en 1970 postuló las bases del Modelo Relacional, quedando representada así la segunda generación de los SGBD. Un punto fuerte a su favor es la sencillez de su estructura lógica y detrás de ella un importante fundamento teórico del cual carecían los modelos anteriores. Las relaciones es el único elemento utilizado para representar tanto entidades como asociaciones entre ellas. El modelo es una solución simple y elegante para satisfacer las más diversas necesidades de consultas y extracción de datos e información, convertido en un estándar de la industria consolidado, una tecnología confiable y eficiente.

Como respuesta a la creciente complejidad de las aplicaciones, a finales de los ochenta, comienzan a aparecer los SGBD Orientados a Objetos, surgiendo el Modelo Orientado a Objeto. La orientación a objetos ofrece flexibilidad para manejar algunos de estos requisitos y

no está limitada por los tipos de datos y los lenguajes de consulta de los sistemas de base de datos tradicionales. Una característica clave es la potencia que proporcionan al diseñador al permitirle especificar tanto la estructura de objetos complejos, como las operaciones que se pueden aplicar sobre dichos objetos. (3)

Otro motivo para la creación de las bases de datos Orientadas a Objetos es el creciente uso de los lenguajes Orientados a Objetos para desarrollar aplicaciones. Las bases de datos se han convertido en piezas fundamentales de muchos sistemas de información y las bases de datos tradicionales son difíciles de utilizar cuando las aplicaciones que acceden a ellas, están escritas en un lenguaje de programación Orientado a Objetos como C++, Smalltalk o Java. Las bases de datos Orientadas a Objetos se han diseñado para que se puedan integrar directamente con aplicaciones desarrolladas con lenguajes Orientados a Objetos, habiendo adoptado muchos de los conceptos de estos lenguajes. (4)

Debido al gran auge de investigación por perfeccionar el manejo de los SGBD, producirían un nuevo modelo denominado Modelo Objeto Relacional. Este incorpora conceptos del Modelo Orientado a Objetos, por lo que evoluciona a ser una base de datos más extensa y compleja. Combina las ventajas de los dos anteriores: la base de datos es relacional por lo que mantiene su rapidez y eficiencia, pero permite hacer uso de nuevos elementos que modelan los objetos a esta base de datos relacional, con lo que el analista y diseñador ven un Modelo Orientado a Objetos

Las bases de datos relacionales son actualmente las más difundidas y usadas. La Universidad de las Ciencias Informáticas (UCI), institución que ocupa un papel protagónico en la producción de software de Cuba, no está exenta de la utilización de este importante modelo de datos. Esta universidad divide su trabajo productivo en áreas temáticas, encontrándose entre ellas, Sistemas de Apoyo a la Salud (SAS), que se encarga del desarrollo de software que da soporte a diferentes actividades no asistenciales, es decir, no ligadas directamente con la atención médica a pacientes. SAS se encuentra conformada por varios proyectos, entre los cuales resalta por su importancia para el país el Sistema para la Gestión y Control de los Colaboradores de la Salud (Colaboración Médica).

Además del proyecto de Colaboración Médica, gran parte de los sistemas que se llevan a cabo en esta área temática tienen desplegado el Modelo Relacional en sus bases de datos, al proveer a que surja la necesidad de aumentar sus funcionalidades debido al creciente

desarrollo que han tenido los mismo, como es el caso del Modelo Objeto Relacional. Por lo que se desea obtener la mejor representación de la información de sus sistemas, para optimar el rendimiento de las aplicaciones que allí se realizan.

Si bien el Modelo Relacional ha suplantado totalmente el Modelo Jerárquico y el de Red, el Modelo Objeto Relacional es una extensión del Modelo Relacional, existiendo en forma paralela para ampliar las posibilidades a los desarrolladores a usar las soluciones adecuadas para sus propias aplicaciones. La necesidad de programar Orientado a Objeto ha hecho que se lleven a cabo sistemas de gran escala y que sean fáciles de entender, simples de depurar y rápidos de actualizar.

Teniendo en cuenta lo antes expuesto surge la necesidad de la realización de la siguiente investigación, dirigida a solucionar el siguiente **Problema a resolver** La limitada capacidad del modelamiento de datos para sistemas de gestión de la información.

El **objeto de estudio** se define como el uso del Modelo Objeto Relacional para sistemas de gestión de la información. Enfocándose el **campo de acción** en el uso del Modelo Objeto Relacional para sistemas de gestión de la información de la salud.

Por ello se plantea como **objetivo general de la investigación:** Determinar la factibilidad del Modelo Objeto Relacional para sistemas de gestión de la información.

Quedan definidas las siguientes **Tareas de la Investigación** para dar cumplimiento al objetivo general de la investigación.

1. Analizar el Modelo Objeto Relacional.
2. Comparar el Modelo Objeto Relacional y el Modelo Relacional.
3. Asimilar una herramienta CASE (Computer Aided Software Engineering) que permita diseñar utilizando este modelo.
4. Elaborar el diseño del Modelo Objeto Relacional de una base de datos implementada de un proyecto siguiendo el Modelo Relacional.
5. Identificar un Sistema Gestor de Dase de datos libre para la implementación del modelo.
6. Implementar el diseño en el gestor identificado.
7. Plantear los casos de prueba.

El presente documento está estructurado por tres capítulos que se describen a continuación:

Capítulo I: Fundamentación Teórica**.** En este capítulo se realiza un análisis del Modelo Objeto Relacional. Se analiza el estado del arte de Sistemas Gestores de Base de Datos existentes que permitan implementar dicho modelo, así como herramientas que permiten modelar una base de datos utilizando el Modelo Objeto Relacional.

Capítulo II: Diseño e Implementación. Se realiza el diseño e implementación de la base de datos en la herramienta escogida y el Sistema Gestor de Base de Datos respectivamente.

Capítulo III: Pruebas. Se le realizan pruebas al sistema para comprobar su efectividad. Se obtienen los resultados de la investigación. Se comprueba si es factible la utilización del modelo. Se realiza un análisis de los beneficios obtenidos.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

El presente capítulo se aborda diferentes conceptos que sirven de base teórica para un mejor entendimiento del tema. Se exponen elementos relacionados con los Sistemas Gestores de Base de Datos y su importancia. Además se incluyen características del Modelo Objeto Relacional, así como el estado del arte de los Sistemas Gestores de Bases de Datos que tienen implementado dicho modelo.

1.1. Conceptos asociados al dominio del problema

Factibilidad: Es el grado en que lograr algo es posible o las posibilidades que tiene de lograrse.

Determinación de la Factibilidad

Factibilidad se refiere a la disponibilidad de los recursos necesarios para llevar a cabo los objetivos o metas señalados, la factibilidad se apoya en 3 aspectos básicos:

- Operativo.
- Técnico.
- Económico.

Factibilidad Operativa: Análisis que estima el grado de aceptación e incorporación del sistema. Se refiere al hecho de que el sistema cumpla con determinadas características de funcionalidad, usabilidad y que sea seguro, si el mismo se llega a desarrollar.

Factibilidad Técnica: Se evalúa la disponibilidad tecnológica utilizada y se mejora el actual (si existiese uno anterior).Con este análisis se determina si es posible desarrollar e implementar el nuevo sistema. (5)

Factibilidad Económica: Un sistema puede ser factible desde los puntos de vista técnico y operacional pero si no lo es económicamente no se podrá llevar a cabo. Este análisis lleva a cabo los costos y beneficios.

La realización de un estudio de factibilidad tiene varios objetivos, entre los que se encuentran:

- Saber si se puede producir algo.
- Definir si se tendrán ganancias o pérdidas.
- Saber cuales son los puntos débiles en producto y reforzarlos.
- Obtener el máximo de beneficios y ganancias.

1.2. Sistema de Gestión de Información

Información: Forma social de existencia del conocimiento consolidada en una fuente determinada. (6)

Gestión de información: Comprende las actividades relacionadas con la obtención de la información adecuada, a un precio apropiado, en el tiempo y lugar más conveniente, para tomar la más óptima decisión.

Sistema de gestión de información: Un Sistema de Información realiza cuatro actividades básicas: entrada, almacenamiento, procesamiento y salida de información. (7)

- **Entrada de información:** Es el proceso mediante el cual el Sistema de Información toma los datos que requiere para procesar la información. Las entradas pueden ser manuales o automáticas. Las manuales son aquellas que se proporcionan en forma directa por el usuario, mientras que las automáticas son datos o información que provienen o son tomados de otros sistemas o módulos.

- **Almacenamiento de la Información:** El almacenamiento es una de las actividades o capacidades más importantes que tiene una computadora, ya que a través de esta propiedad el sistema puede recordar la información guardada en la sección o proceso anterior.

- **Procesamiento de la información:** Es la capacidad del Sistema de Información para efectuar cálculos de acuerdo con una secuencia de operaciones preestablecida. Estos cálculos pueden efectuarse con datos introducidos recientemente en el sistema o bien con datos que están almacenados.

- **Salida de la información:** La salida es la capacidad de un Sistema de Información para sacar la información procesada o bien datos de entrada al exterior. Las unidades típicas

de salida son las impresoras, terminales, diskettes, cintas magnéticas, la voz, entre otros. Es importante aclarar que la salida de un Sistema de Información puede constituir la entrada a otro Sistema de Información o módulo.

Los Sistemas de Gestión de Información constituyen hoy, no sólo soportes de los negocios, sino, además, un instrumento de ventajas competitivas sostenibles al permitir gestionar los activos tangibles e intangibles y convertirse en una herramienta integral de gerencia. Considerar a la información base del conocimiento, la inteligencia y el talento organizacional, como única fuente de ventaja competitiva sostenible, coloca a las organizaciones en condiciones para el aprendizaje y la innovación.

1.3. Las bases de datos y los Sistemas Gestores de Bases de Datos

Una base de datos es un conjunto de datos interrelacionados, no redundantes y persistentes, con una estructura claramente definida basada en un modelo de datos y almacenados en un soporte informático, organizado de forma independiente de su utilización y accesible simultáneamente por distintos usuarios y aplicaciones.

Los Sistemas de Información separan la definición de la estructura de datos de los programas de aplicación y almacenan esta definición en la base de datos, lo que permite que al añadir o modificar estructura de datos, los programas de aplicación no se ven afectados, ya que no dependen directamente de ello.

Una base de datos nunca se accede o manipula directamente sino a través de un Sistema Gestor de Base de Datos. Los SGBD, son aplicaciones que permiten tanto a los usuarios no informáticos como al experto definir, crear y mantener la base de datos y proporciona un acceso controlado a la misma. Es la aplicación que interactúa con los usuarios de los programas de aplicación y la base de datos.

Algunos de los SGBD más conocido son, Oracle, Dbase, Ingres, Microsoft SQL Server, Postgre, Sybase entre otros. Todos se han desarrollado conjunto con las nuevas necesidades y exigencias del perfeccionamiento de las aplicaciones. Cada uno con características propias en sí, pero con un conjunto semejantes de objetivos que permiten maniobrar de forma eficiente, llana y concreta todos los datos. A continuación se exponen algunos de ellos:

Definición de los datos: Mediante el lenguaje de definición de datos, permiten precisar la base de datos, lo cual posibilita especificar la estructura, tipo de datos y restricciones sobre los mismos.

Gestión de datos: Posibilitan la eliminación, inserción, actualización de los datos mediante el lenguaje de manipulación de datos.

Acceso controlado de la base de datos: Proporcionan seguridad e integridad a la base de datos, al igual que el control de recurrencia y recuperación.

Gestión física: Gestionan la estructura física de los datos y su almacenamiento para proporcionar eficacia en las operaciones de la base de datos.

Mecanismo de vistas: Brinda un mecanismo de vistas, permitiendo a cada usuario tener su propia visión de la base de datos. (8)

Además de estos, independizan la estructura de la organización lógica de los datos y la descripción lógica de la base de datos, al igual que permiten una fácil administración de los datos.

1.4. Modelos de Datos

Los modelos de datos aportan la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos, así como la base formal para las herramientas y técnicas empleadas en el desarrollo y uso de Sistemas de Información. Con respecto al diseño de la base de datos, el modelado puede ser descrito dados los procesos y requerimientos de información de una aplicación de un intenso uso de datos, realizar una representación de la aplicación que capture las propiedades estáticas y dinámicas requeridas para poder sustentar todos los procesos que se lleven a cabo. Además de tener en cuenta las necesidades que se requieren para la realización del diseño, la representación debe permitir capturar futuros requerimientos.

Un **modelo de datos** es por tanto un lenguaje orientado a describir los elementos de la realidad que intervienen en el problema dado, y la forma que se relacionan esos elementos entre sí. Típicamente permite describir las estructuras de datos, las restricciones de integridad (conjunto de condiciones que deben cumplir los datos para reflejar correctamente la realidad deseada) y las operaciones de manipulación de datos.

El criterio principal para la clasificación de los SGBD es en cuanto al modelo lógico en que se basan. Los modelos lógicos más utilizados en los SGBD son el Modelo Relacional, el Modelo de Red y el Modelo Jerárquico. Algunos SGBD más modernos utilizan el Modelo Orientado a Objeto y el Modelo Objeto Relacional.

El Modelo Relacional ha sido suficiente para las aplicaciones tradicionales comerciales o de negocios, las mismas se caracterizan por manejar datos muy simples en grandes volúmenes, generalmente se refieren a datos alfanuméricos que, con bastante precisión y facilidad, pueden ser representados en un computador. Estas han evolucionado en cuanto a sus necesidades de manipulación, almacenamiento y análisis de datos y sus características y requisitos difieren en gran mesura de las típicas aplicaciones de gestión. Debido a que las transacciones son de larga duración y la estructura de los objetos es más compleja, se necesitan nuevos tipos de datos para almacenar imágenes, fotos y hace falta definir operaciones no estándar, especificas para cada aplicación.

Un primer acercamiento para proporcionarle solución , fue la de extender el Modelo Relacional para dar lugar a estos nuevos datos y sus necesidades de manejo, lo cual trajo consigo que aparecieran las bases de datos extensibles, desarrollándose hasta lo que se conoce hoy en día como tecnología Objeto Relacional. El término de base de datos Objeto Relacional se utiliza cuando se va a describir una base de datos que ha evolucionado desde el Modelo Relacional hacia una base de datos híbrida, que contiene ambas tecnologías: Relacional y de Objetos.

1.5. Modelo Objeto Relacional

Una base de datos Objeto Relacional o un Sistema Administrador de Base de Datos Objeto Relacional es un sistema administrador de base de datos similar a una base de datos Relacional pero con un modelo de base de datos Orientado a Objetos: los objetos, clases y herencias están soportados directamente en esquemas de base de datos y en el lenguaje de consulta, además, soporta la extensión del modelo de datos con tipos de datos personalizados y métodos.

Reúnen ventajas de ambos modelos. Como sistema Orientado a Objeto son capaces de realizar gestiones muy complejas de componentes, además de adaptarse fácilmente a nuevos tipos y formatos de datos. Por el otro lado en su faceta Relacional pueden soportar aplicaciones para grandes transacciones de datos y lenguajes como SQL; también proporcionan los medios

para ampliar las posibilidades de una base de datos a funciones y tipos de datos definidos por el usuario.

Uno de los objetivos para este tipo de sistema es reducir la brecha conceptual entre las técnicas de modelado de datos, tales como el Diagrama de Entidad Relación (ERD) y Mapeo Objeto Relacional (ORM) y base de datos relacionales, además de cerrar la brecha entre las base de datos relacionales y el Modelado Orientado a Objetos utilizados en las técnicas de lenguajes de programación como Java, C + + o C #.

Se considera que los Sistemas Administradores de Base de Datos Relacional (RDBMS) tradicionales o productos SGBD-SQL se centran en la gestión eficiente de los datos extraídos de un conjunto limitado de tipos de datos, un SGBD Objeto Relacional permite a los desarrolladores integrar sus propios tipos y los métodos que les son aplicables en el SGBD. Los Sistemas Gestores de Base de Datos Objeto Relacional (SGBDOR) tienen por objetivo permitir a los desarrolladores elevar el nivel de abstracción para poder ver el dominio del problema.

En esencia, los conceptos de la tecnología Objeto Relacional son:

- Tipos abstractos de datos (TADs), definidos por el usuario.
- Tipos complejos (o estructurados, o agregados).
- Herencia: definición de tipos de datos como subtipos de otros.
- Tipos referencia. Apuntadores a objetos de otro tipo.
- BLOBs (eran lo único que tenían los RDBMS).

Estos nuevos tipos de datos, necesitan nuevas funciones de manipulación. Estas funciones son:

- Métodos definidos por el usuario: métodos asociados a los TADs.
- Operadores para los tipos complejos (estructurados o agregados).
- Operadores para los tipos referencia.

Con relación al modelo anterior aparecen nuevas características que lo hacen superior. Se extiende la sintaxis de SQL: 92 y se amplía su alcance:

Extensiones en los Tipos de Datos (dominios) Objeto Relacionales:

- Tipo definido por el usuario.
- Tipo fila y referencia.

- Tipo colección (set, bag, list y array).
- Tipo de Objetos Grandes, LOB (Large Objects).

Extensiones en el Control de la Semántica de datos Objeto Relacionales:

- Triggers
- Procedimientos almacenados y funciones definidas por el Usuario

Extensiones en las capacidades Consultivas Objeto Relacionales:

- Consultas recursivas.
- SQL: 99 Incorpora al lenguaje XML y viceversa desde 2006.

Otras Extensiones en las Bases de Datos Objeto Relacionales:

- Extensiones a la ayuda en la toma de decisiones.
- Intercambio de Objetos (OEM), datos multimedia, etc.

Trayendo consigo muchos beneficios entre los que se encuentran:

Extensibilidad: Capacidad de ampliar el sistema de tipos para dar soporte a las nuevas necesidades de las aplicaciones.

- Nuevos tipos de datos: que representen mejor el dominio de la aplicación y gestionar aplicaciones más complejas.
- Nuevas operaciones: para soportar el comportamiento de los tipos.

Poder de expresión: Necesidad de soportar objetos y relaciones complejas.
Reusabilidad: Capacidad de compartir librerías de tipos existentes.
Integración: Del Modelo Relacional y el Orientado a Objetos en un solo lenguaje.
Nuevas consultas: recursivas, multimedia, consultas anidadas, almacenadas, prefabricadas etc.
Mayor capacidad expresiva para los conceptos y asociaciones complejos**.** (9)

1.6. Sistemas de Gestores de Bases de Datos

1.6.1. Oracle

Oracle es un SGBD privativo que desde versión 8.i, ha sido significativamente extendido con conceptos del modelo de base de datos Orientadas a Objetos. Los tipos de objetos definidos por el usuario son el fundamento de su modelo de objetos. Desde la versión Oracle 8 soporta el paradigma Orientado a Objetos en su Servidor de Datos Universal. Implementa las principales características funcionales de las bases de datos Objeto Relacionales en su sistema de gestión de base de datos, a la par que garantiza la compatibilidad con el marco funcional del tradicional modelo de datos Relacional. De este modo, Oracle ofrece el primer producto tecnológico de un servidor de base de datos híbrido.

El modelo de objetos de Oracle es similar al del sistema de clases de C++, Microsoft .NET, y Java. Esto permite implementar una arquitectura Orientada a Objetos real para que forme parte de la implementación de una base de datos. Las entidades complejas del mundo real y la lógica se pueden modelar fácilmente y, a la vez, se permite la reutilización de objetos que permiten desarrollar aplicaciones de la base de datos de una forma más rápida y con mayor eficiencia.

Los programadores de aplicaciones pueden acceder directamente a tipos de objetos, sin necesidad de ninguna capa adicional entre la base de datos y la capa cliente. Las ocurrencias de los tipos de objetos se albergan en la base de datos como filas de una tabla o como valores de una columna y las aplicaciones soportan totalmente las características de abstracción, y encapsulación del comportamiento de los objetos basados en el paradigma Orientado a Objetos.

Los objetos en Oracle son capaces de actuar sobre los datos de distintas formas. Tanto los tipos objeto como sus métodos, están almacenados en la base de datos, y pueden ser accedidos por cualquier aplicación. Es un producto donde sus costos técnicos son muy altos y su licencia es de las más caras. Sólo se utiliza mayormente en empresas grandes y multinacionales.

1.6.2. PostgreSQL

PostgreSQL es un Sistema Gestión de Base de Datos Objeto Relacional basado en POSTGRES, Versión 4.2, desarrollada en el Departamento de Informática de la Universidad de California (Berkeley). POSTGRES fue un proyecto dirigido por el Profesor Michael Stonebraker y contó con el apoyo de diversos organismos y empresas de EE.UU. Es un software libre que proporciona soporte para SQL92 y SQL99.

Con la incorporación de los nuevos conceptos que brinda permite a los usuarios ampliar de forma cómoda sus aplicaciones. Entre estos se destaca la herencia, nuevos tipos de datos y funciones. Dentro de las características que suministran al sistema flexibilidad y solidez, aparecen las restricciones, la integridad de las transacciones y las reglas de integridad.

PostgreSQL trae consigo particulares propias que lo diferencian de un sistema Relacional, entre las que se encuentran, nuevos tipos de datos a parte de los tipos de datos que incluye SQL. Entre ellos se pueden destacar los tipos geométricos, permitiendo representar objetos especiales bidimensionales. El tipo point (punto) es la base del resto. Los tipos soportados son point (), line (), lseg (), box (), path (), polygon (), circle (). Además, proporciona un conjunto de operadores y funciones que permiten realizar operación geométrica como escalada, translaciones, rotaciones y hallar puntos de intersección.

También aparecen los tipos de dirección de red. PostgreSQL aporta tipos para almacenar direcciones IP y direcciones MAC, donde la primera es el número que identifica de manera lógica y jerárquica a las computadoras dentro de una red que utilice el protocolo IP (Internet Protocolo), y la restante un número fijo que es asignado a la tarjeta o dispositivo de red por el fabricante. Ambos son utilizados para evitar el uso del tipo cadena de caracteres para esta tarea, puesto que aporta restricciones y funciones asociadas a estos tipos de datos. Los tipos incluidos son cidr (), inet () y macaddr ().

Dentro del tipo de datos cadena de caracteres aflora el tipo text, no especificado por el estándar SQL, pero si soportado por muchos RDBMS. Este tipo de datos admite una cadena de caracteres de longitud variable e ilimitada. Posibilita definir columnas como arreglos multidimensionales de longitud variable. Los arreglos podrán ser de cualquier tipo predefinido en PostgreSQL o de un tipo definido por el usuario.

1.7. Herramientas de modelado visual

1.7.1. ER/Studio

Es una herramienta de modelado de datos fácil de usar, equipado para crear y manejar diseños de bases de datos funcionales y confiables. Ofrece fuertes capacidades de diseño lógico, sincronización bidireccional de los diseños físicos y lógicos, construcción automática de la base de datos, documentación y fácil creación de reportes. Ayuda a diseñar, generar y mantener aplicaciones de base de datos de calidad y alto rendimiento , y a la toma de decisiones para resolver embotellamientos de los datos, eliminar redundancia y alcanzar en última instancia usos de más alta calidad que entreguen datos más eficientes y exactos. Desde un modelo lógico de sus requerimientos de información y reglas del negocio que definen su base de datos, hasta un modelo físico optimizado por las características específicas de su base de datos de destino. (10)

1.7.2. GNU Ferret

GNU Ferret anteriormente conocido como Gerwin, es un clon del programa privativo ErWin, que sirve para construir modelos de datos mediante diagramas Entidad Relación, y generar el SQL correspondiente al modelo.

1.7.3. PowerDesigner 6.1

El PowerDesigner es una suite de aplicaciones consignada a la construcción, diseño y modelado de datos a través de diversas aplicaciones. Cuenta con una serie de productos que posibilitan la más firme base de datos para aplicaciones de alto rendimiento. Entre ellos se destacan:

- PowerDesigner DataArchitect: Provee a los diseñadores de las bases de datos, una manera eficiente para la creación inteligente, depuración e ingeniería de reversa del modelado tanto conceptual como físico de los datos.
- PowerDesigner AppModeler: Permite el diseño y ajuste de los componentes de objetos y datos en aplicaciones de uso común como, C++, Visual Basic y Delphi ajustando el modelo de base de datos.
- PowerDesigner WarehouseArchitect: Provee un poderoso datawarehousing para el diseño e implementación de una base de datos. Cuenta con soporte para base de datos

tradicionales y base de datos en plataformas de sistemas analíticos al usarse modelados dimensionales, esquemas, particionamiento y agregación. (11)

1.7.4. Enterprise Architect

Es una herramienta de análisis y diseño de negocio y UML Orientada a Objetos para el desarrollo completo del software. Enterprise Architect (EA) es diseñada para ayudar a construir software robusto y fácil de mantener. Ofrece salida de documentación flexible y de alta calidad. Presenta soporte para los 13 diagramas de UML 2 y más, ayuda a administrar la complejidad con las herramientas para rastrear las dependencias, de la misma manera que provee una generación poderosa de documentos. Soporta ingeniería inversa para muchos de los SGBD, incluyendo Oracle, SQLServer, My SQL, PostgreSQL y otros. Modela tablas, llaves, relaciones complejas y perfiles de modelado de datos incluidos y además permite generar scripts para crear estructuras de base de datos. (12)

1.8. Herramientas de desarrollo

1.8.1 EMS SQL Manager para PostgreSQL

El EMS SQL Manager para PostgreSQL es una herramienta fácil de manejar para la administración de PostgreSQL. Es una de las más completas en la actualidad. Trabaja con cualquiera de sus versiones hasta la 8.i.
Características:

- Soporte completo para PostgreSQL hasta la versión 8.1.
- Administración y navegación rápida de base de datos.
- Administración fácil de todos los objetos PostgreSQL.
- Herramientas de manipulación avanzada de datos.
- Administración efectiva de seguridad.
- Excelentes herramientas visuales y de texto para la construcción de consultas.
- Capacidades de exportación e importación de datos.
- Poderoso diseñador visual de base de datos.
- Modo guiado para labores de mantenimiento.
- Interfaz atractiva. (13)

1.8.2 EMS Data Generator for PostgreSQL

EMS Data Generator for PostgreSQL constituye una poderosa herramienta para generar datos de prueba a tablas de base de datos PostgreSQL de una vez. Permite definir las tablas y campos lo que se le desea generar datos, configurar valores de rangos y obtener listas de valores desde consultas SQL. La interfaz del asistente es muy fácil de manipular. Suministra diferentes tipos de generación de datos por cada campo, lista, azar (random), generación incremental de datos y más. Aporta un control automático sobre integridad referencial para la generación de datos a tablas vinculadas. Ofrece la capacidad para configurar valores nulos para ciertos casos. (14)

1.8.3 EMS SQL Query for PostgreSQL

Constituye una herramienta que permite rápida y sencilla construcción de consultas SQL a base de datos PostgreSQL. Presenta una interfaz gráfica fácil de usar, que admite conectarse a base de datos PostgreSQL, seleccionar tablas y campos para una consulta al servidor PostgreSQL, y establecer los criterios de selección. Permite editar el texto de consulta en el editor, ver los resultados y el tiempo que demoró en realizar la misma. Admite trabajar simultáneamente con varias consultas en ventanas separadas, múltiples conexiones con base de datos, y presenta un historial de las consultas realizadas. (15)

1.8.4 Apache JMeter

JMeter una herramienta Java dentro del proyecto de Jakarta que permite realizar pruebas de rendimiento y pruebas funcionales sobre aplicaciones web y bases de datos. El proyecto Jakarta es el que se encarga de crear y mantener todas las soluciones Open Source (Código Abierto) creadas para la plataforma Java. Se destaca por su versatilidad, estabilidad, y por ser de uso gratuito. JMeter permite realizar pruebas web clásicas, pero también permite realizar test de Protocolo de Transferencia de Archivos (FTP), Conectividad de la Base de Datos de Java (JDBC) y Servicios Web (en Beta) entre otras. También permite la ejecución de pruebas distribuidas entre distintos ordenadores para realizar pruebas de rendimiento además de mostrar los resultados en una amplia variedad de informes y gráficas.

Los elementos jerárquicos del JMeter son:

- Listeners (Elementos de escucha).
- Config Elements (Elementos de configuración).
- Post-processors (Post- procesadores).
- Pre-processors (Pre- procesadores).
- Assertions (Afirmaciones).
- Timers (Cronómetros).

En este capítulo se describieron diferentes conceptos que sirven de base para tener un mejor entendimiento sobre el tema. Se muestra además, la importancia que tienen los Sistemas Gestores de Base de Datos y se detallan características sobre los gestores Oracle y PostgreSQL .Por último se exponen un grupo de herramientas que fueron analizadas para la realización del diseño.

CAPÍTULO 2. DISEÑO E IMPLEMENTACIÓN

En el presente capítulo se describe el diseño del Modelo Objeto Relacional a partir del esquema del Modelo Relacional ya realizado, y las principales tablas del nuevo modelo. Se tienen en cuenta las características que brinda el actual modelo para realizar la implementación y detallar cada una de las funcionalidades que se utilizan. Además, de seleccionar la herramienta y el Sistema Gestor de Base de Datos a utilizar.

2.1. Análisis del problema y metodología utilizada

Para la realización del siguiente estudio se tomo como guía una base de datos Relacional ya implementada en el proyecto de Colaboración Médica del área temática SAS de la facultad 7. El proyecto está dividido en dos módulos, el de gestión y el de pago. Realizados uno y otro con el objetivo de gestión de la información de los colaboradores en misiones médicas y control del pago respectivamente. La base de datos cuenta con un grupo de tablas de las cuales se seleccionaron las principales para el estudio. (Ver figura 1).

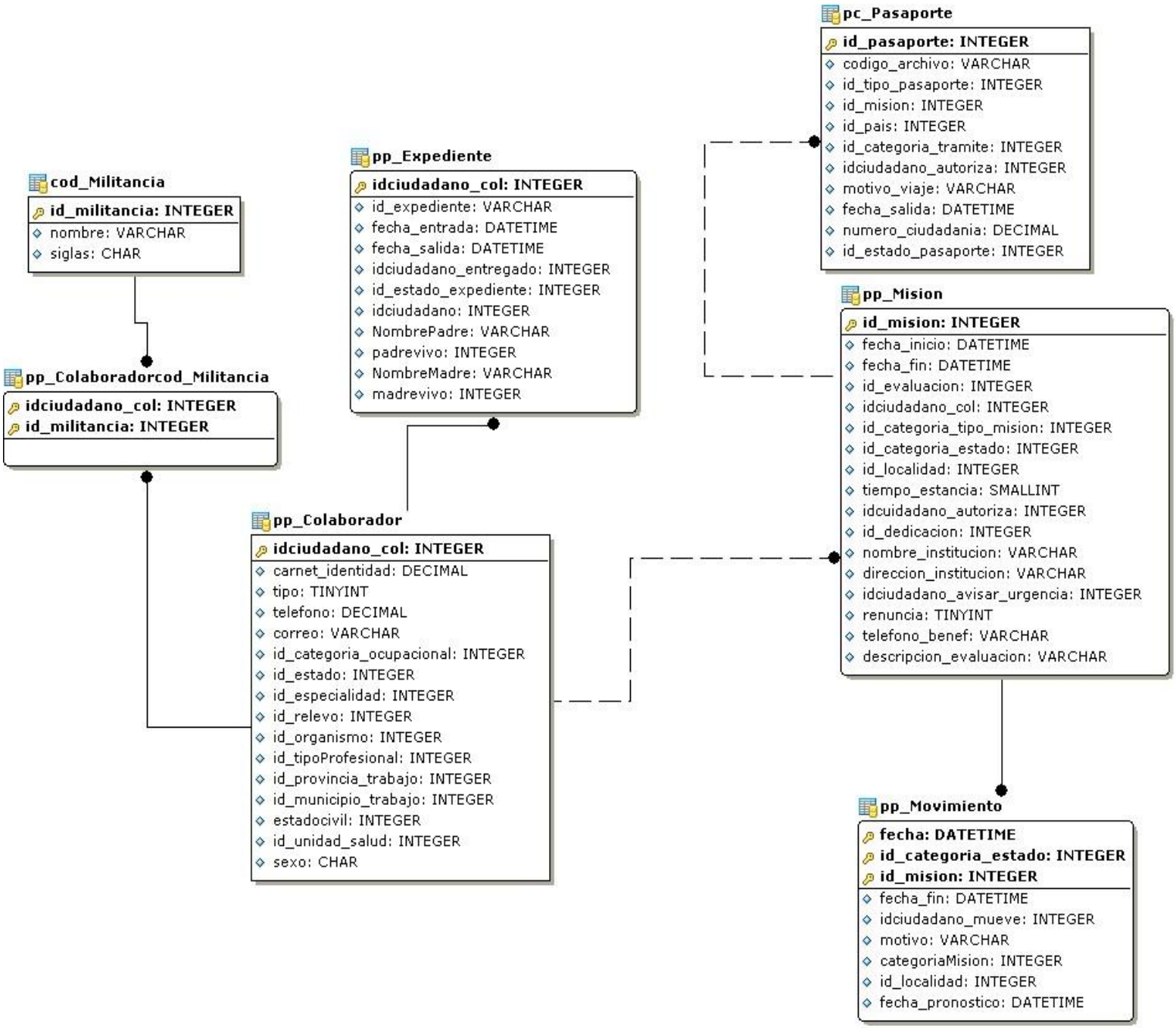

Figura 1. Diagrama de clases del Modelo Relacional

Una vez que se escogieron las tablas fundamentales, se realizó el diseño basado en el Modelo Objeto Relacional pero siguiendo la misma estructura de las anteriores y pudiéndose agregar o modificar los casos necesarios. Una vez reestructurado el diseño, quedó de la siguiente manera:

Colaborador: Tabla donde se almacenan los datos de los colaboradores de manera general.

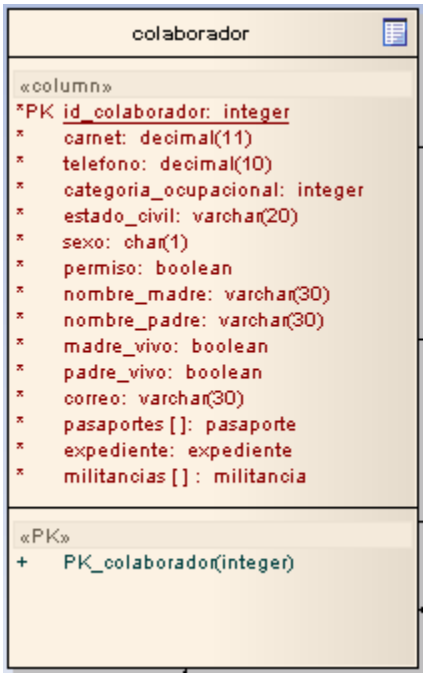

Figura 2. Tabla colaborador

No_salud*:* Tabla donde se almacenan los datos de los colaboradores no médicos. Esta tabla hereda de colaborador.

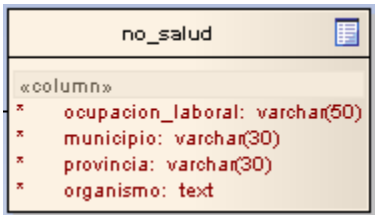

Figura 3. Tabla no_salud

Salud: Tabla donde se almacenan los datos de los colaboradores médicos. Esta tabla también hereda de colaborador.

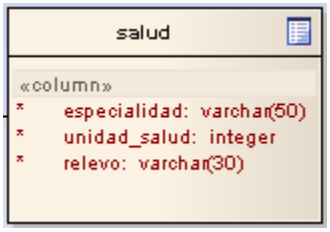

Figura 4. Tabla salud

Misión: Tabla donde se almacenan los datos de la misión que realiza el colaborador.

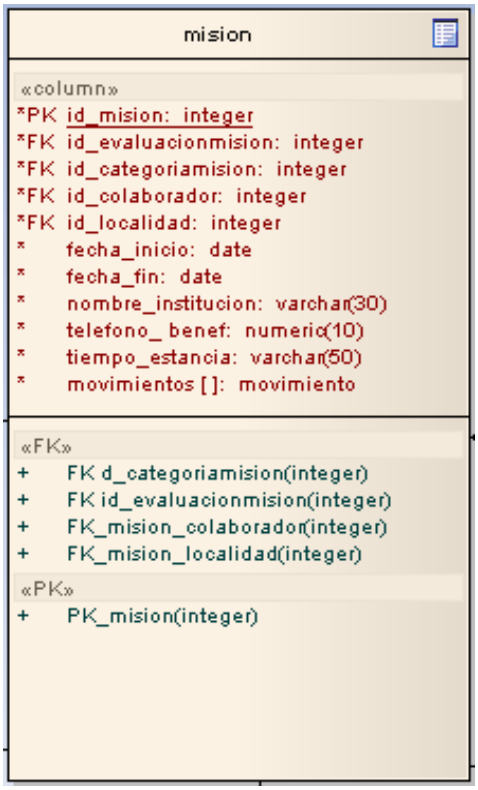

Figura 5. Tabla misión

Pasaporte: Tabla que se recogen en el pasaporte de los colaboradores.

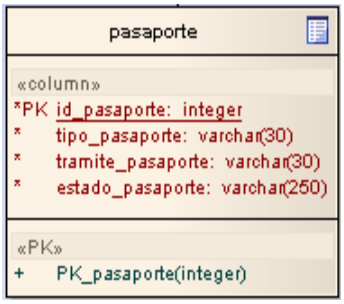

Figura 6. Tabla pasaporte

Localidad: Tabla donde se recogen datos referentes a la localidad y al país.

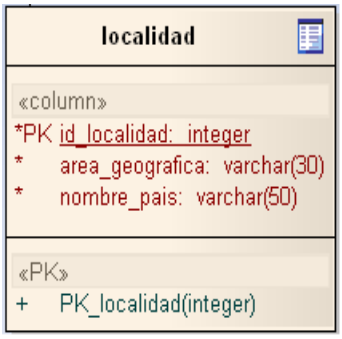

Figura 7.Tabla localidad

Militancia: Tabla que recoge los datos relacionados con las militancias que posee cada colaborador

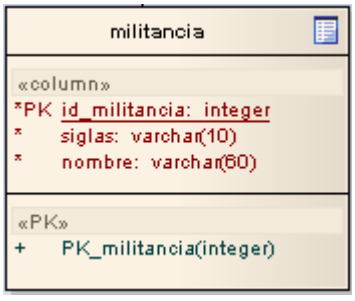

Figura 8.Tabla militancia

Movimiento: Tabla que recoge los datos relacionados con los movimientos que se le realiza a cada colaborador dentro de la misión.

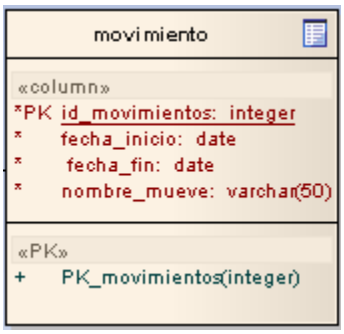

Figura 9. Tabla movimiento

Evaluación misión: Tabla que recoge los datos relacionados con la evaluación que se recibe cada colaborador en la misión.

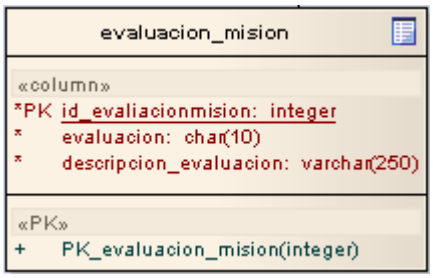

Figura10. Tabla evaluación_misión

Categoría misión: Tabla que recoge los datos relacionados con la categoría que posee la misión de cada colaborador.

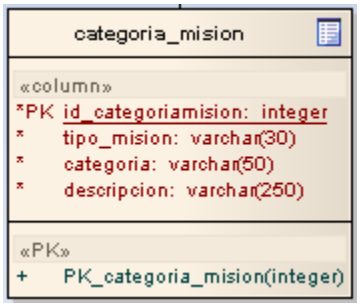

Figura11. Tabla categoría_misión

Debido a que no existe una metodología que guie el proceso de análisis y diseño de una base de datos Objeto Relacional se utilizó el paradigma Orientada a Objeto al esperar una representación de los datos lo más cercano posible al Modelo Objeto Relacional. Para la realización del diseño de la base de datos se utilizó la herramienta CASE Enterprise Architect debido a que esta provee la notación UML (Unified Modelling Lenguage), lenguaje de modelado Orientado a Objeto más difundido en la actualidad, además de permitir modelar relaciones complejas de herencia, asociación y composición, insertar tipos de datos definidos por el usuario y generar script para un conjunto de SGBD.

En la siguiente figura se muestra el diseño de la base de datos donde se visualiza las relaciones antes mencionadas.

La herencia se muestra al definir la tabla colaborador de la cual heredan salud y no salud. Las tablas hijas van a obtener todas las propiedades y atributos, tantos de ellas mismas como de la tabla colaborador.

Las relaciones de composición se observan en militancia y pasaporte, todas con respecto a la tabla colaborador. De igual forma aparece la tabla movimiento con respecto a misión.

Existen asociaciones en evaluación_misión, categoría_misión y localidad.

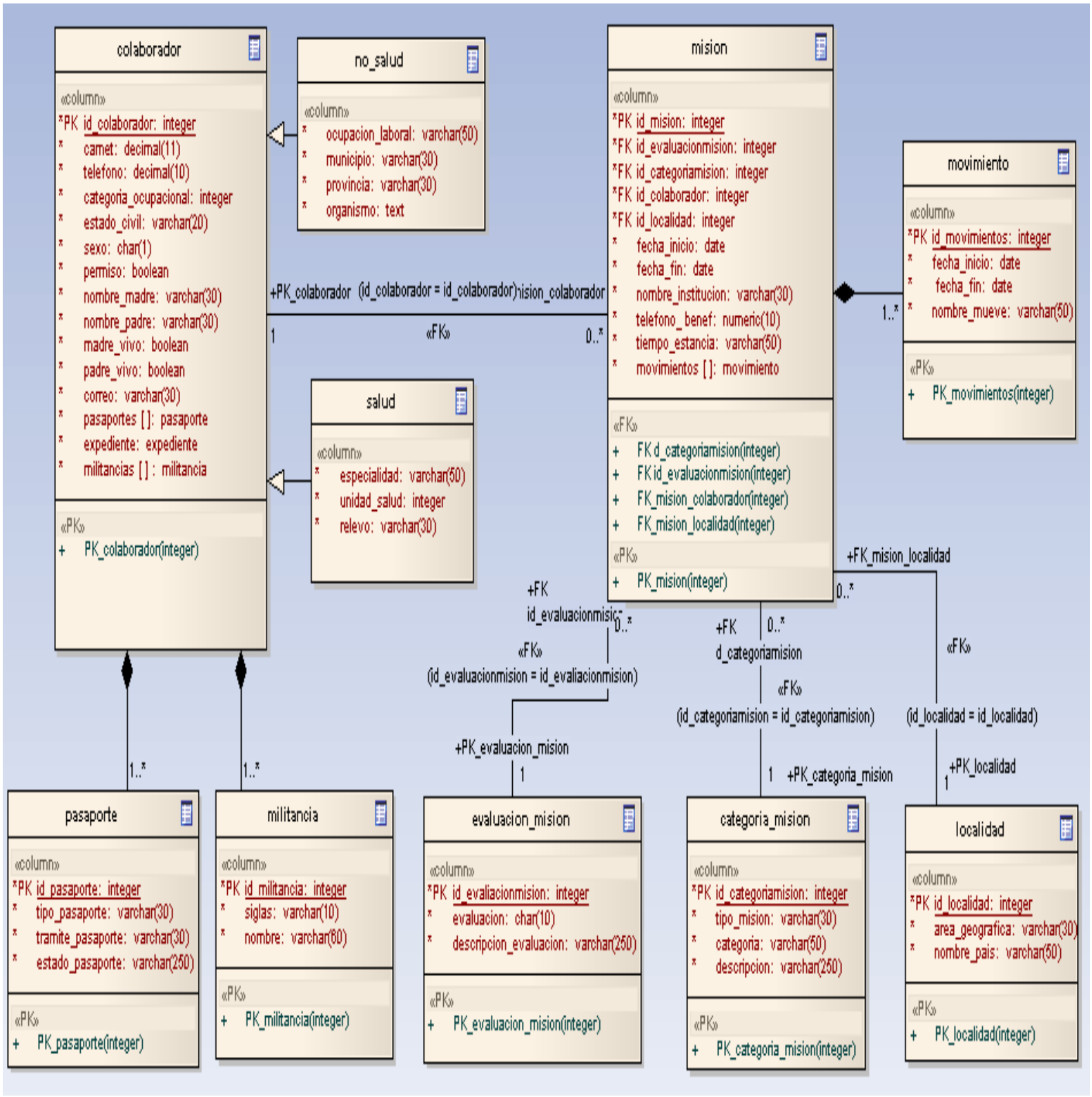

Figura 12. Diagrama de clases del Modelo Objeto Relacional

Para el desarrollo de la base de datos se utilizó PostgreSQL por ser el gestor Objeto Relacional más estable de código abierto y avanzado en estos momentos y soporta un máximo de base de datos ilimitado con un máximo de tamaño de los campos de 1GB. Es un Sistema Gestor de Base de Datos Objeto Relacional que implementa especificaciones para nuevos tipos de datos para el estándar SQL: 1999, para posibilitar la ejecución de datos estructurados como colecciones y arreglos, tipos definidos por los usuarios, de referencia entre otros. Una vez realizado el diseño de la base de datos, se generó el script para luego ser modificado al adoptar las sentencias SQL para PostgreSQL.

2.2. Implementación del diseño de la base de datos Objeto Relacional.

2.2.1. Herencia

Una vez que se obtuvo el script, se implementaron las características propias del modelo, como es el caso de la herencia. La cual mediante el comando INHERITS permite que tablas hijas hereden de las tablas madres. En este caso en particular se le realizó a las tablas Salud y No_salud. Como las tablas ya estaban creadas anteriormente solo se le agregó el fragmento de código INHERITS (colaborador), fuera de los paréntesis de la tabla. El resultado es el siguiente:

CREATE TABLE public.colaborador (id_colaborador integer, carnet numeric(11,0), telefono numeric (10,0),categoria_ocupacional integer, estado_civil character varying(20),sexo character(1), permiso boolean,…….);

CREATE TABLE no_salud (ocupacion_laboral varchar (50), municipio varchar (30),

provincia varchar (30), organismo (text) **INHERITS** (colaborador);

En este caso una instancia de la clase no_salud hereda todos los atributos de la clase colaborador, además de los atributos adicionales que posee. Una clase puede heredar de ninguna o varias clases y una consulta pueden hacer referencia tanto a todas las instancias de una tabla, como a sus descendientes. De esta forma se podría realizar las siguientes consultas:

SELECT carnet, telefono FROM no_salud;

La anterior consulta devuelve todos los carnets y teléfonos de los no_salud.

SELECT carnet, telefono FROM colaborador;

La anterior consulta devuelve todos los carnets y teléfonos de los colaboradores incluyendo todos sus descendientes en la herencia, es decir de los colaboradores de manera en general, los no_salud y los de salud.

2.2.2. Arreglos

Otra de las implementaciones realizadas fue la definición de arreglos. PostgreSQL permite crear tanto arreglos bidimensionales como unidimensionales de longitud fija o variable y utiliza la enumeración basada en uno por defecto, por lo que los arreglos comienzan en 1 y terminan en n elementos. Son utilizados en dependencia de los datos que se utilice. En la base de datos, se crearon tres arreglos, dos de ellos en la clase colaborador y otro en la clase misión.

El arreglo movimientos describe la fecha de inicio y fin del movimiento y el motivo y nombre de quien lo realiza. El que se encuentra en colaborador describe las militancias que poseen cada colaborador con las siglas y el significado de estas.

Para ilustrar su uso se muestra el siguiente ejemplo:

CREATE TABLE mision (id_mision integer, id_evaluacionmision integer, id_categoriamision integer, id_localidad, fecha_inicio date, fecha_fin date, nombre_institucion varchar (30),direcion_institucion varchar (250),telefono_benef varchar (30), **movimientos movimiento [ ]**);

Para escribir elementos en el arreglo se utilizan los paréntesis para encerrar los elementos, y son separados por comas. El arreglo se declaró como un tipo de dato movimiento, que es una tabla, por lo que va a contener dentro los mismos campos que se definieron en la clase movimiento. (Ver Figura 9).

Ejemplo:

{"(1,2008-02-02,2008-02-04, Juan)","(2,2008-02-11,2008-02-12, Maria)","(3,2008-02-10,2008-02-14, Pedro)"}

Los demás arreglos correspondientes a la clase colaborador fueron:

militancias militancia [ ]

{"(6, ACRC, \"Asociacion de Combatientes de la Revolución Cubana\")","(5, CDR, \"Comité de Defensa de la Revolución\")","(1, PCC, \"Partido Comunista de Cuba\")","(2, MTT, \"Milicia de Tropas territoriales\")"}

pasaportes pasaporte [ ]

{"(84, Corriente, Confeccion, Valido)"}

Inserción de datos

Para insertar datos en los arreglos se utilizaron los tipos de datos ARRAY y ROW de PostgreSQL ya predefinidos. Para cada valor del arreglo hay que especificar a que tipo de datos pertenece, separados por coma. La siguiente sentencia muestra como se realiza la inserción de datos en misión.

INSERT INTO public.mision (id_mision, id_evaluacionmision, id_categoriamision, id_colaborador, id_localidad, fecha_inicio, fecha_fin, nombre_institucion, telefono_benef, tiempo_estancia, movimientos)

VALUES (1, 1, 1, 1, 1, '20/02/2009', '20/02/2009', ' Joaquín Lenzina', 2620469, 'dos meses',

ARRAY [row ('1','20/02/2009','02/03/2009','luis')::movimiento])

La consulta siguiente muestra como se insertan datos en la tabla Salud donde se encuentran los otros dos arreglos. De similar forma quedaría para No_salud, pero con los atributos propios de ella.

INSERT INTO public.salud (id_colaborador, carnet, telefono, categoria_ocupacional, estado_civil, sexo, permiso, nombre_madre, nombre_padre, madre_vivo, padre_vivo, correo, pasaportes, expediente, militancias, especialidad, relevo, unidad_salud)

VALUES(5002, 86122212436, 2620468, 1, 'casado', 'F', 'true', 'Ana', 'Juan', 'false', 'false', 'amesa@mighty.nl.cu', ARRAY[ROW(1,'corriente','confeccion','valido')::pasaporte], NULL,ARRAY[ROW(5,'PCC','Partido comunista de Cuba')::militancia, ROW(4,'FMC','Federacion de Mujeres Cubanas')::militancia], 'Cirugía Plástica y Estética', 'Yolanda', 1)

Selección de datos

Una vez insertados los datos se pueden realizar consultas para obtener todos los elementos que se encuentra en el arreglo o cada uno por separado. De igual forma se logran obtener los atributos que le corresponden a cada uno de los elementos.

La siguiente consulta recupera todos los movimientos la tabla misión.

```
SELECT mision. movimientos    FROM  mision;
```

La siguiente consulta obtiene todos los movimientos que se encuentran en la primera posición del arreglo.

```
SELECT mision.movimientos [1]    FROM  mision;
```

En caso que se desee obtener solamente algún elemento específico del arreglo la consulta quedaría:

```
SELECT mision.movimientos [1].fecha_inicio    FROM  mision;
```

Actualización de datos

Para la actualización de los datos que se encuentran en el arreglo se realiza de una forma similar. Se puede actualizar todo el arreglo o simplemente algunos de los elementos del mismo.

```
UPDATE  public.mision SET

movimientos = ARRAY [ROW(1,'02/02/2009','02/02/2009','luis' )::movimiento]

WHERE    public.mision.id_mision=1

UPDATE public.mision  SET

movimientos [1]=ROW('1','02/02/2009','02/02/2009','Liss')::movimiento

WHERE public.mision.id_mision=2
```

2.3. Tipos de datos compuestos

Un tipo de dato compuesto describe la estructura de un registro o fila en una tabla de la base de datos. La sintaxis es comparable con la de CREATE TABLE con la salvedad de que los nombres de los atributos pueden definirse sin necesidad de especificar la sintaxis NOT NULL, además de tener incluido la palabra clave AS. En la base de datos implementada se realizó un tipo de dato compuestos, el expediente en la tabla colaborador.

El expediente esta compuesto por el identificador del expediente, la fecha de entrada y de salida, estado del expediente y nombre del designado.

CREATE TYPE public.expediente AS (id_expediente integer, fecha_entrada date,

fecha_salida date, estado_expediente character (10), nombre_designado character varying (30));

Al definir el tipo de datos compuesto, se puede utilizar en la creación de las tablas.

CREATE TABLE public.colaborador (id_colaborador integer, carnet numeric(11,0), telefono numeric(10,0), categoria_ocupacional integer, estado_civil character varying(20) , sexo character(1), permiso boolean, nombre_madre character varying(30), nombre_padre character varying(30), madre_vivo boolean, padre_vivo boolean, correo character varying(30), pasaportes pasaporte[], **expediente expediente**, militancias militancia[])

Inserción de datos

Para desarrollar la inserción de datos se utiliza ROW como tipo de datos definido en PostgreSQL, entre paréntesis y separados por coma .La consulta queda de la siguiente forma:

INSERT INTO public.salud (id_colaborador, carnet, telefono, categoria_ocupacional, estado_civil, sexo, permiso, nombre_madre, nombre_padre, madre_vivo, padre_vivo, correo, pasaportes, expediente, militancias, especialidad, relevo, unidad_salud) VALUES(5003, 86122212436, 2620468, 1, 'casado', 'F', 'true', 'Ana', 'Juan', 'false', 'false', 'amesa@mighty.nl.cu', NULL, **row(1,'02/02/2009','02/03/2009','incompleto','Luis')**, NULL, 'Cirugía Plástica y Estética', 'Yolanda', 1)

Selección de Datos

Al igual que en los arreglos los tipos de datos compuestos permiten seleccionar cada elemento en particular o toda la estructura. Las siguientes consultas describen como obtener un expediente, el identificador y la fecha de entrada respectivamente de los colaboradores de la salud.

```
SELECT   public.salud. Expediente
FROM  public.salud
SELECT   (expediente).id_expediente
FROM    public.salud
SELECT (expediente).fecha_entrada
FROM   public.salud
```

La siguiente consulta es un ejemplo de la unión de las anteriores

```
SELECT  public.salud. Expediente,  (expediente).id_expediente  FROM   public.salud
```

Actualización de datos

Otra de las acciones que se pueden realizar es la actualización de dicho tipo de datos. La sintaxis es la siguiente:

```
UPDATE  tabla SET nombre_tipodatos = (valores por los cuales se va a cambiar);
```

De manera que si se modificara el tipo de dato expediente de la clase colaborador quedaría de la siguiente forma.

```
UPDATE   public.salud
SET   expediente = row (1,'02/02/2009','02/02/2009','incompleto','Anna')
 WHERE  public.salud.id_colaborador=1
```

La consulta anterior modifica el tipo de dato expediente de la clase colaborador cuyo identificador del colaborador es 1. En caso de que no se especifique ninguna condición entonces serán modificados todos los tipos de datos pasaporte del colaborador, de manera que todos quedan iguales.

```
UPDATE public.salud

SET expediente. Id_expediente =4

WHERE  public.salud.id_colaborador=2
```

Asimismo se pueden actualizar los datos individualmente. La consulta anterior actualiza específicamente el expediente del colaborador, cuyo identificador es 2.Cosecutivamente se pueden realizar actualizaciones según sea conveniente, posteriormente un ejemplo:

```
UPDATE public.salud

SET  expediente. Id_expediente = (expediente).id_expediente + 4

WHERE  public.salud. id_colaborador=2
```

En este capítulo se eligió la herramienta para el diseño de la base de datos que sigue el Modelo Objeto Relacional y el Sistema Gestor de Base de Datos que soporta su funcionamiento. Se realizó una descripción del diseño del Modelo Objeto Relacional, en el cual se representan las relaciones y tablas que intervienen en el mismo. Además se efectuó su implementación, teniendo en cuenta las características propias que este brinda.

CAPÍTULO 3. PRUEBAS

En el presente capítulo se clasifican las sentencias que se le efectuaran a la base de datos. Se describe el ambiente en el cual se desarrollaron las pruebas y su ejecución en la base de datos con la implementación del Modelo Relacional y el Modelo Objeto Relacional sobre el gestor PostgreSQL 8.3. Los resultados obtenidos se analizan y se obtienen los resultados.

Uno de las categorías del lenguaje de consultas estructurado (SQL) es DML (Lenguaje de Manipulación de Datos) el cual se combina con las instrucciones para actualizar y manipular. El mismo esta integrado a su vez por los comandos:

- SELECT: Utilizado para consultar registros de la base de datos que satisfagan un criterio determinado.
- INSERT: Utilizado para cargar lotes de datos en la base de datos en una única operación.
- UPDATE: Utilizado para modificar los valores de los campos y registros especificados.
- DELETE: Utilizado para eliminar registros de una tabla de una base de datos.

Las consultas que se manejaron en las pruebas fueron clasificadas en consultas de Selección (SELECT), de Inserción (INSERT) y de Actualización (UPDATE). A continuación se exponen dos de cada grupo y su implementación en ambos modelos, con el objetivo de facilitar la comprensión del lenguaje SQL y mostrar las diferencias al ser utilizado sobre los modelos. Las descripciones de las consultas mostradas a continuación se encuentran en el Anexo 1.

3.1. Consultas de selección para el Modelo Objeto Relacional y el Modelo Relacional

Modelo Objeto Relacional

- **SELECT** public.mision.id_mision, public.mision.movimientos[1]

  FROM public.salud INNER JOIN public.misionion ON (public.salud.id_colaborador = public.mision.id_mision)

  WHERE (public.salud.categoria_ocupacional = 2)

- **SELECT** count (public.colaborador. id_colaborador) AS cantidad

  FROM public.colaborador INNER JOIN public.mision ON (public.colaborador. id_colaborador = public.mision. id_colaborador) INNER JOIN public.localidad ON (public.mision.id_localidad = public.localidad.id_localidad)

  WHERE (public.localidad.id_localidad = 1) AND (public.mision.id_mision=1) AND (expediente).id_expediente=1

Modelo Relacional

- **SELECT** pp_Mision. id_mision, pp_Movimiento. id_mision

  FROM pp_Colaborador INNER JOIN pp_Mision ON (pp_Colaborador.idciudadano_col=pp_Mision.idciudadano_col) INNER JOIN pp_Movimiento ON (pp_Mision. id_mision= pp_Movimiento. id_mision)

  WHERE pp_Colaborador.id_categoria_ocupacional=2

- **SELECT** COUNT (pp_Colaborador.idciudadano_col)AS cantidad **FROM** pp_Colaborador INNER JOIN pp_Mision ON pp_Colaborador.idciudadano_col=pp_Mision.idciudadano_col INNER JOIN pp_Expediente ON pp_Expediente.idciudadano_col=pp_Mision.idciudadano_col INNER JOIN cod_Localidad ON cod_Localidad. id_localidad=pp_Mision. id_localidad **WHERE** cod_Localidad. id_localidad=1 AND pp_Mision.idciudadano_col=1 AND pp_Expediente.idciudadano_col=1

3.2. Consultas de Inserción para el Modelo Objeto Relacional y el Modelo Relacional

Modelo Objeto Relacional

- **INSERT INTO** public.salud(id_colaborador, carnet,telefono,categoria_ocupacional,

  estado_civil, sexo, permiso, nombre_madre, nombre_padre, madre_vivo, padre_vivo, correo, pasaportes, expediente, militancias, especialidad, relevo, unidad_salud) **VALUES** (5002, 12321456987, 4545454, 1, 'casado', 'F', 'true', 'Alberta', 'Rafael', 'false', 'true', 'ymarti@netscape.sg.cu',array[row('5001','Marino', 'Prorroga','Vencido')::pasaporte], row('7','2009-10-14','2009-10-16','completo','Carlos'), array[row('1',' PCC','Partido Comunista Cubano')::militancia], 'Neurocirugía', 'Elisa', 1)

- **INSERT INTO** evaluacion_mision (id_evaluacionmision, evaluacion, descripcion_evaluacion) **VALUES** (1, 'B', 'Fue una buena mision ')

Modelo Relacional

- **INSERT INTO** pp_Colaborador (idciudadano_col, carnet_identidad, tipo, telefono, correo, id_categoria_ocupacional, id_estado, id_especialidad, id_relevo, id_organismo, id_tipoProfesional, id_provincia_trabajo, id_municipio_trabajo, estadocivil, id_unidad_salud, sexo) **VALUES** (5006,111113011,0,2620492,'Oberliebersbach Str. 9',1,5,2,3,1,4,1,1,2,1,'f')

- **INSERT INTO** pc_Evaluacion_Mision (id_evaluacion, descripcion, id_tipo_evaluacion) **VALUES** (5003,'No Tiene',1)

3.3. Consultas de actualización para el Modelo Objeto Relacional y el Modelo Relacional

Modelo Objeto Relacional

- **UPDATE** public.mision SET movimientos = ARRAY[row('1','2009/02/03','2009/02/05','Luis')::movimiento]

  WHERE public.mision.id_mision= 2

- **UPDATE** public.colaborador SET pasaportes = ARRAY [row(1,'marino','prorroga','vencido')::pasaporte]

WHERE public.colaborador. id_colaborador=1

Modelo Relacional

- **UPDATE** public.pp_movimiento SET motivo = 'Traslado', categoriamision = 141, id_mision = 5000 **WHERE** public.pp_movimiento. id_localidad=149
- **UPDATE** pc_Pasaporte SET motivo viaje = 'CM', fecha_salida = '2009/02/06'

WHERE pc_Pasaporte.id_pasaporte= 1

3.4. Ambiente de las pruebas

Las bases de datos donde se realizaron las pruebas llevan como nombre bd_MOR y dw_pg_colaboracion con la implementación del Modelo Objeto Relacional y Modelo Relacional respectivamente. Se crearon en un servidor con las siguientes características:

- Sistema Operativo: Debian etch 4.0
- Memoria RAM: 1 GB
- Disco duro: 256
- Velocidad CPU: 3.2:GHz

Para el llenado de las mismas se utilizó la herramienta EMS Data Generator 2005 for PostgreSQL, ya que permite generar datos tanto para una o varias tablas a la vez, respeta la integridad referencial al generar los datos que provienen de otras tablas para evitar errores y genera datos para ambos modelos implementados a diferencia de otros generadores probados que sólo generan datos para las bases de datos implementadas siguiendo el Modelo Relacional. Se generó un volumen de 5000 tuplas para cada tabla de ambas bases de datos. Las consultas de SELECT, INSERT y UPDATE poseen diferente complejidad (ver anexo 1) y se elaboraron de acuerdo a cada modelo de datos. La herramienta utilizada para medir el rendimiento de las consultas fue el JMeter.

3.4.1 Diseño de las pruebas en la herramienta

Objetivo: Medir el rendimiento del modelo implementado, ya sea el Modelo Relacional o el Modelo Objeto Relacional, en el gestor de base de datos PostgreSQL.

El proceso de diseñar las pruebas en la herramienta JMeter se realiza de la siguiente manera:
El componente principal es denominado Test Plan o Plan de Pruebas donde se definen todos los aspectos relacionados con la prueba. Este se muestra automáticamente una vez que se abre la aplicación.
Primeramente se crea un Thread Group o Grupo de Hilos, considerado como el grupo de usuarios que se desea simular para la base de datos. Luego se llenan las siguientes opciones:

- Nombre: Se define un nombre.
- Number of Threads (users): Equivale al número de usuarios que se desean simular.
- Ramp-Up Period: Es el lapso de tiempo en segundos que se desea tener entre cada grupo de usuarios se usará, en este caso 1
- Loop Count o Forever: Se utiliza para indicar si la simulación para grupos de hilos será llevada acabo infinitamente, o por un ciclo determinado de veces.

Una vez definidas las características del Grupo de Hilo se pasa a generar las JDBC Connection Configuration (Configuración de la Conexión JDBC) donde se crea una para cada base de datos a probar. Las JDBC Connection Configuration se usan para configurar las conexiones de las bases de datos Esta posee los siguientes aspectos:

- Nombre de la variable: Nombre deseado.
- Database URL: Dirección donde se encuentra la base de datos.
- JDBC Driver class: Para MySQL es com.mysql.jdbc.Driver y para PostgreSQL org.postgresql.Driver.
- El usuario y contraseña de la base de datos.

El resto de los campos que aparecen están predefinidos por defecto.
Luego se generan las JDBC Request (Petición JDBC) utilizadas para definir las peticiones de simulación, en la cual aparecerán las siguientes opciones:

- Name: El que se desee dar a la consulta.
- Variable Name: El mismo que se definió en la variable Name de la JDBC
- Tipo de consulta: Select Statement en caso de ser una consulta SELECT y así sucesivamente.
- Consulta (Query): Donde se define la consulta.

Para mostrar los resultados se pueden generar otros gráficos, a través de los cuales se puede observar en diferentes vías los resultados de las pruebas. Finalmente se guarda el plan de prueba y se ejecuta.

3.5. Resultados de las Pruebas

A continuación se muestra los resultados agrupados por los tipos de consulta.

3.5.1 Consultas de selección

Resultados de las pruebas sobre el Modelo Objeto Relacional y Modelo Relacional

En la tabla 1 se muestra el rendimiento en segundos, así como el número mínimo, máximo y media de peticiones realizadas sobre el Modelo Objeto Relacional por cada una de las 10 consultas.

Modelo Objeto Relacional	# Muestras	Media	Mín	Máx	% Error	Rendimiento/seg
1	500	2046	4	4884	0.0	92,58
2	500	165	4	680	0.0	35,92
3	500	26	3	270	0.0	38,58
4	500	8	2	110	0.0	41,12
5	500	2111	18	4790	0.0	93,84
6	500	7	2	92	0.0	40,49
7	500	187	5	854	0.0	390,63
8	500	13	3	207	0.0	401,93
9	500	19	4	327	0.0	39,37
10	500	58	5	848	0.0	403,88

Tabla 1. Resultados de la consulta de selección sobre el Modelo Objeto Relacional

El Figura 13 muestra un gráfico que representa el comportamiento del rendimiento para cada una de las consultas sobre el Modelo Objeto Relacional.

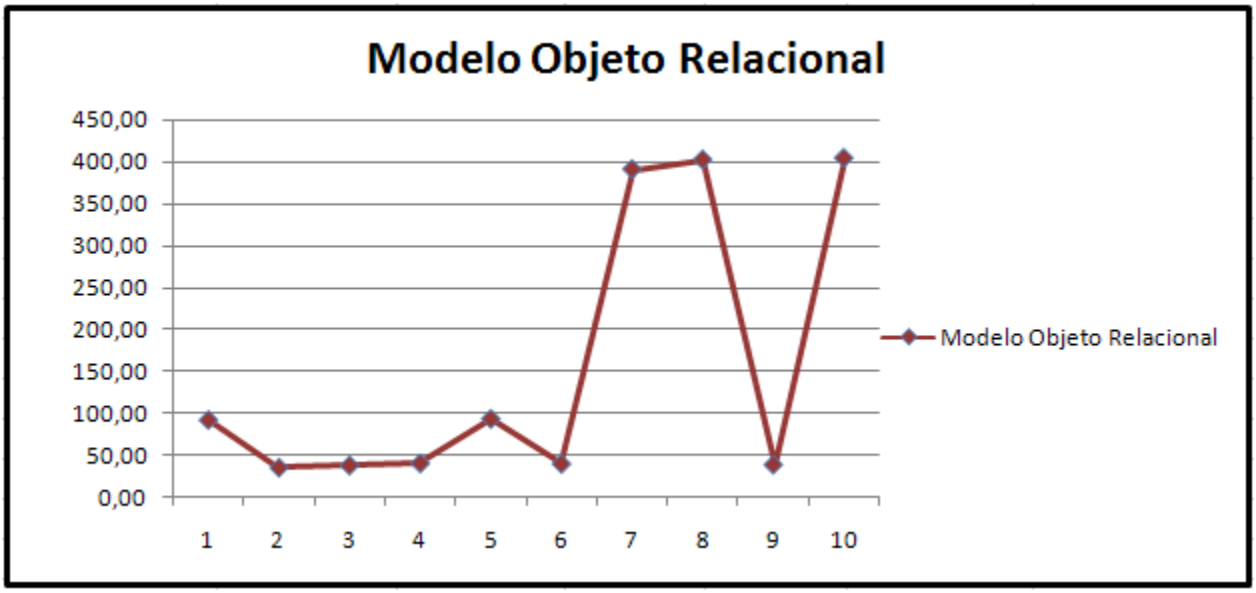

Figura 13. Comportamiento del rendimiento por consultas en el Modelo Objeto Relacional

Los picos del rendimiento están dado por las consultas que al utilizar el tipo de datos arreglo en algún momento de su implementación y procesan un atributo específico en caso de éste ser compuesto, o sea tener varios atributos, el rendimiento se ve afectado, ver consulta # 8 y #10 en Anexo 1. Ocurriendo lo contrario si no se ve implicado en profundidad el uso de arreglos.

A continuación se muestra los resultados obtenidos de las consultas realizadas sobre el Modelo Relacional.

Modelo Relacional	# Muestras	Media	Mín	Máx	% Error	Rendimiento/seg
1	500	1521	24	3852	0.0	12,57
2	500	15	2	175	0.0	4,08
3	500	7	2	104	0.0	40,39
4	500	7	2	89	0.0	408,16
5	500	7114	102	10692	0.31	68,84
6	500	4	1	74	0.0	39,81
7	500	478	11	1967	0.0	241,08
8	500	693	18	1878	0.0	20,11
9	500	1999	35	4804	0.0	98,70
10	500	4476	49	9354	0.0	49,95

Tabla 2. Resultados de la consulta de selección sobre el Modelo Relacional

La siguiente Figura 14 muestra un gráfico que representa el comportamiento del rendimiento para cada una de las consultas sobre el Modelo Relacional.

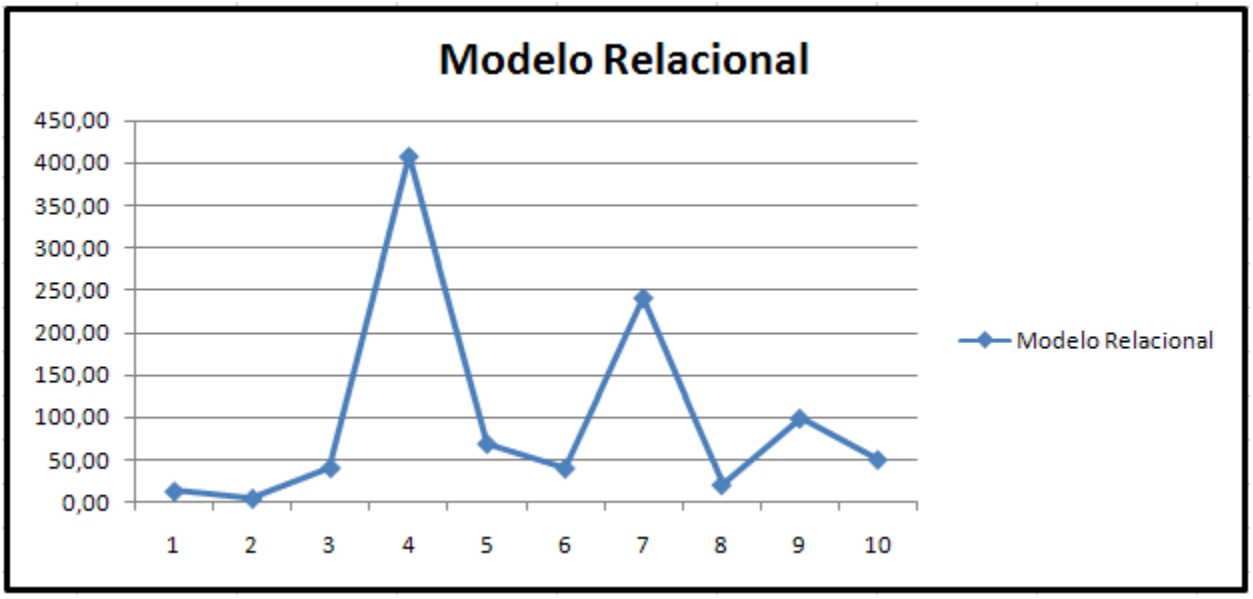

Figura 14. Comportamiento del rendimiento por consultas en el Modelo Relacional

El rendimiento se ve afectado pues en la consultas #4, #7 y #9 la cadena de encuentro entre tablas es mayor y por tanto el rendimiento se ve afectado por la cantidad de tuplas que existen en estas tablas y la realización del encuentro entre las tablas implica una demora en la obtención de los resultados.

3.5.2 Consulta de Inserción

En la Tabla 3 se muestra el rendimiento en segundos, el número mínimo, el máximo y media de las peticiones realizadas sobre el Modelo Objeto Relacional por cada una de las 5 consultas.

Modelo Objeto Relacional	# Muestras	Media	Mín	Máx	% Error	Rendimiento/seg
1	500	0	0	13	0.0	421,59
2	500	0	0	24	0.0	425,53
3	500	0	0	13	0.0	423,01
4	500	0	0	12	0.0	424,09
5	500	0	0	16	0.0	423,72

Tabla 3. Resultados de la consulta de inserción sobre el Modelo Objeto Relacional

El Figura 15 muestra un gráfico que representa el comportamiento del rendimiento para cada una de las consultas sobre el Modelo Objeto Relacional.

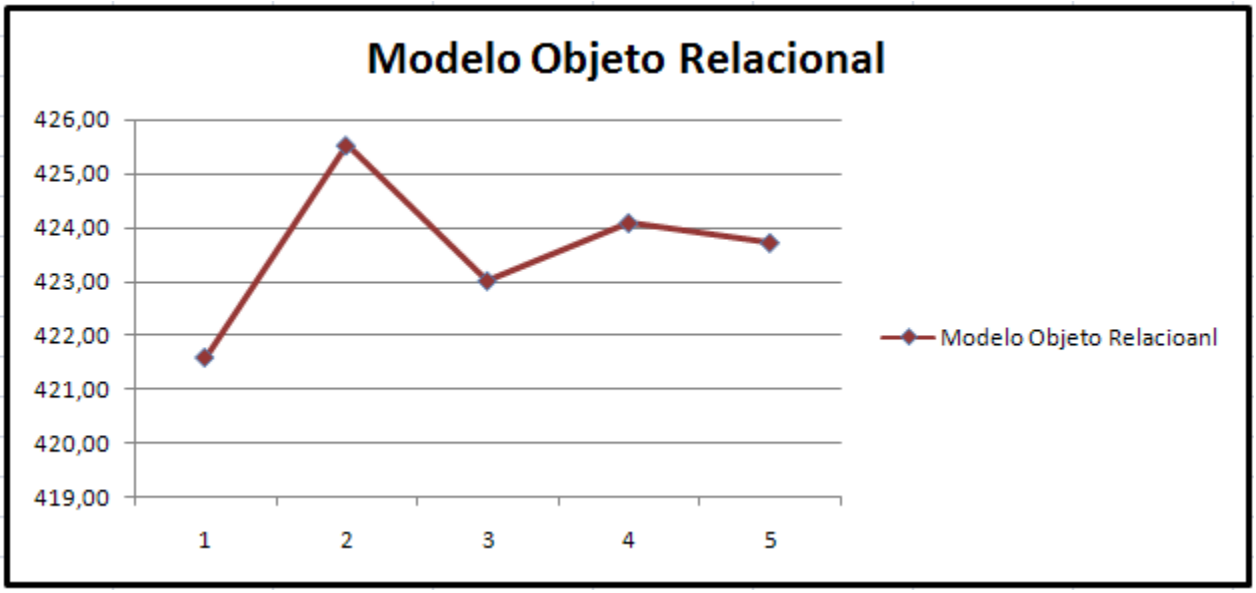

Figura 15. Comportamiento del rendimiento por consultas en el Modelo Objeto Relacional

A continuación se muestra los resultados obtenidos de las consultas realizadas sobre el Modelo Relacional.

Modelo Relacional	# Muestras	Media	Mín	Máx	% Error	Rendimiento/seg
1	500	0	0	17	0.0	421,23
2	500	0	0	17	0.0	432,53
3	500	0	0	17	0.0	418,41
4	500	0	0	25	0.0	423,01
5	500	0	0	32	0.0	433,28

Tabla 4. Resultados de la consulta de inserción sobre el Modelo Relacional

El Figura 16 muestra un gráfico que representa el comportamiento del rendimiento para cada una de las consultas sobre el Modelo Relacional

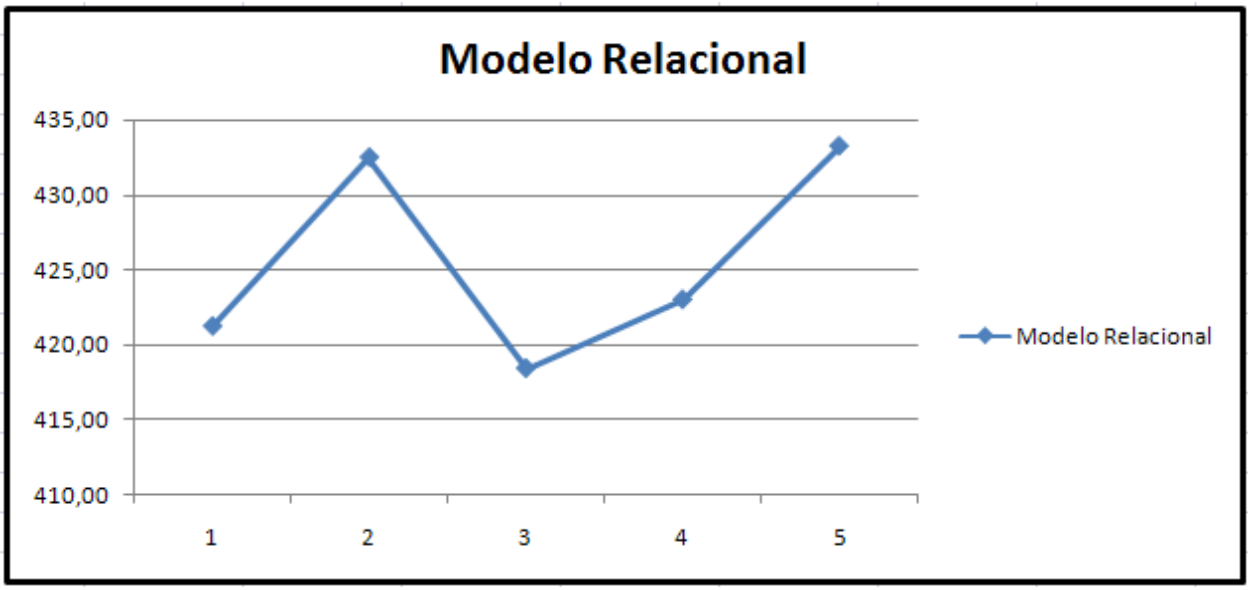

Figura 16. Comportamiento del rendimiento por consultas en el Modelo Relacional

3.5.3 Consultas de Actualización

En la Tabla 5 se muestra el número mínimo, máximo, media y el rendimiento en segundos de las peticiones realizadas sobre el Modelo Objeto Relacional por cada una de las 4 consultas.

Modelo Objeto Relacional	# Muestras	Media	Mín	Máx	% Error	Rendimiento/seg
1	500	20	1	190	0.0	372,30
2	500	204	11	625	0.0	351,37
3	500	4	0	89	0.0	393,70
4	500	36	2	284	0.0	436,68

Tabla 5. Resultados de la consulta de actualización sobre el Modelo Objeto Relacional

El Figura 17 muestra un gráfico que representa el comportamiento del rendimiento para cada una de las consultas sobre el Modelo Objeto Relacional

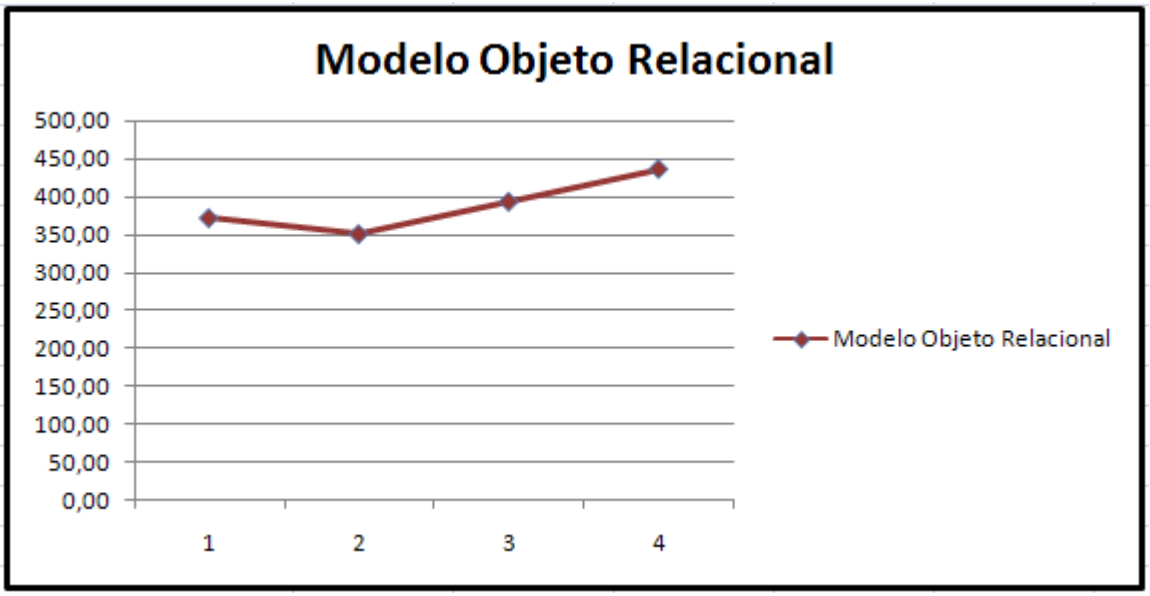

Figura 17. Comportamiento del rendimiento por consultas en el Modelo Objeto Relacional

La siguiente tabla muestra los resultados obtenidos de las consultas realizadas sobre el Modelo Relacional.

Modelo Relacional	# Muestras	Media	Mín	Máx	% Error	Rendimiento/seg
1	500	10	2	94	0.0	401,61
2	500	4	0	90	0.0	400,00
3	500	4	0	70	0.0	397,77
4	500	2	0	60	0.0	420,54

Tabla 6. Resultados de la consulta de actualización sobre el Modelo Relacional

El Figura 18 muestra un gráfico que representa el comportamiento del rendimiento para cada una de las consultas sobre el Modelo Relacional

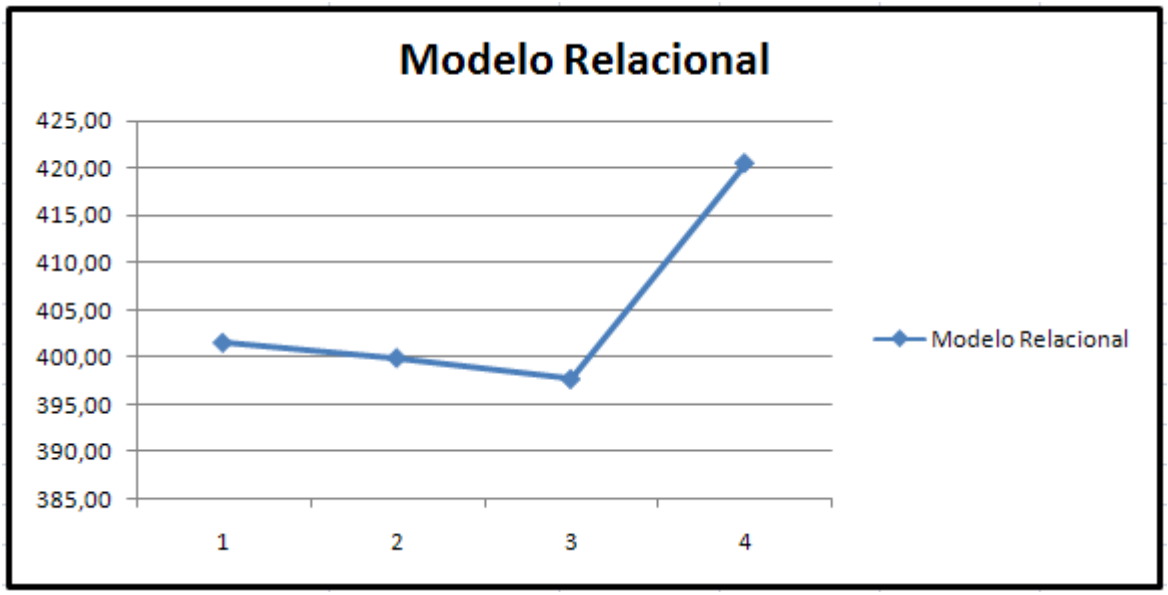

Figura 18. Comportamiento del rendimiento por consultas en el Modelo Relacional

3.6. Análisis de los Resultados

Selección

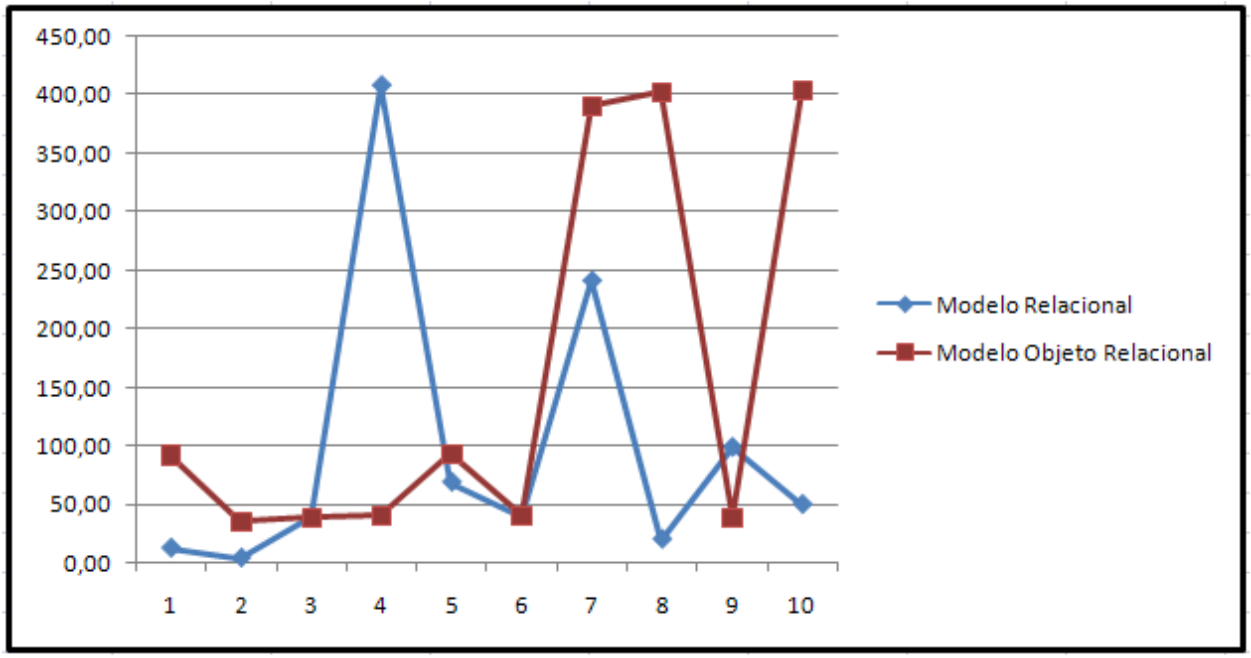

Figura 19. Resultados de las consultas de selección

Al efectuar un análisis de los resultados obtenidos de las consultas probadas en ambos modelos, se puede concluir lo siguiente:

Para las consultas de tipo SELECT en el Modelo Objeto Relacional, donde se utilizan los tipos de datos compuestos, el rendimiento es similar al Modelo Relacional, tal es el caso de las consultas #3 y #6. Otras consultas del tipo SELECT, el rendimiento es mucho mejor al aplicarse sobre el Modelo Objeto Relacional, pues no hacen un gran uso de las particularidades que ofrece el Modelo Objeto Relacional. Siendo diferente con aquellas como son las consultas #7, #8 y #10 que si utilizan en mayor medida las características del Modelo Objeto Relacional, como son arreglos, en la cláusula WHERE o en el SELECT.

Inserción

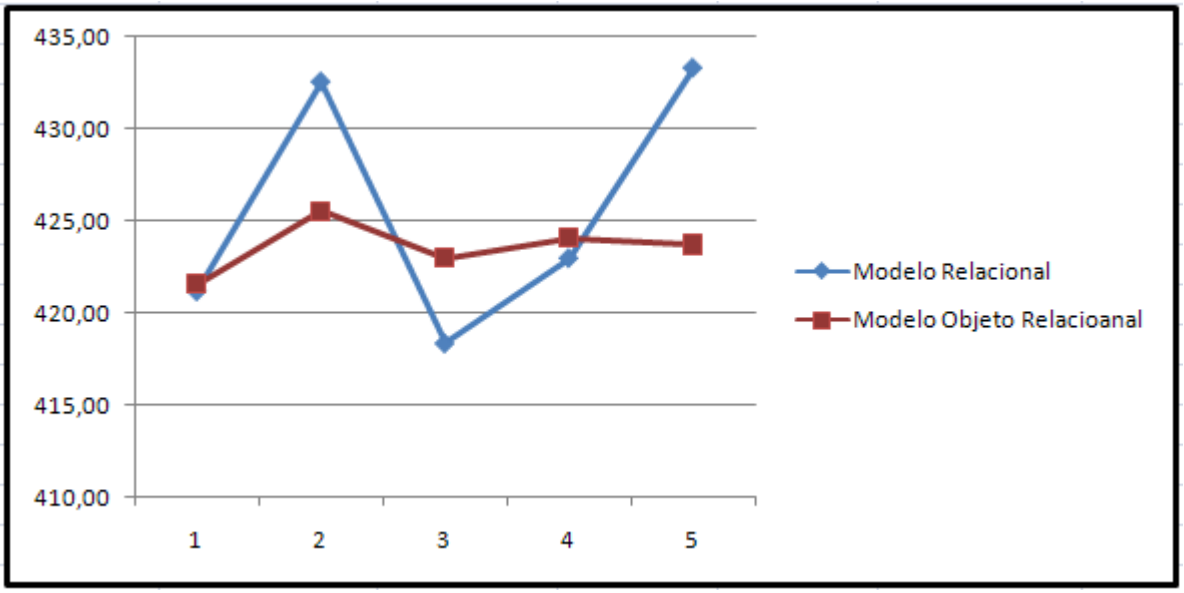

Figura 20. Resultados de las consultas de inserción

Al efectuar un análisis de los resultados obtenidos sobre las consultas de inserción, se puede concluir:

En las consultas no presentan ninguna característica propia del Modelo Objeto Relacional tales como arreglos o tipos de datos compuestos, el Modelo Relacional exhibe un peor rendimiento. En los casos donde se utilizan los arreglos y los tipos de datos compuestos, ambos a la vez, el Modelo Objeto Relacional muestra un comportamiento igual al Modelo Relacional, como por ejemplo la consulta #1.

Actualización

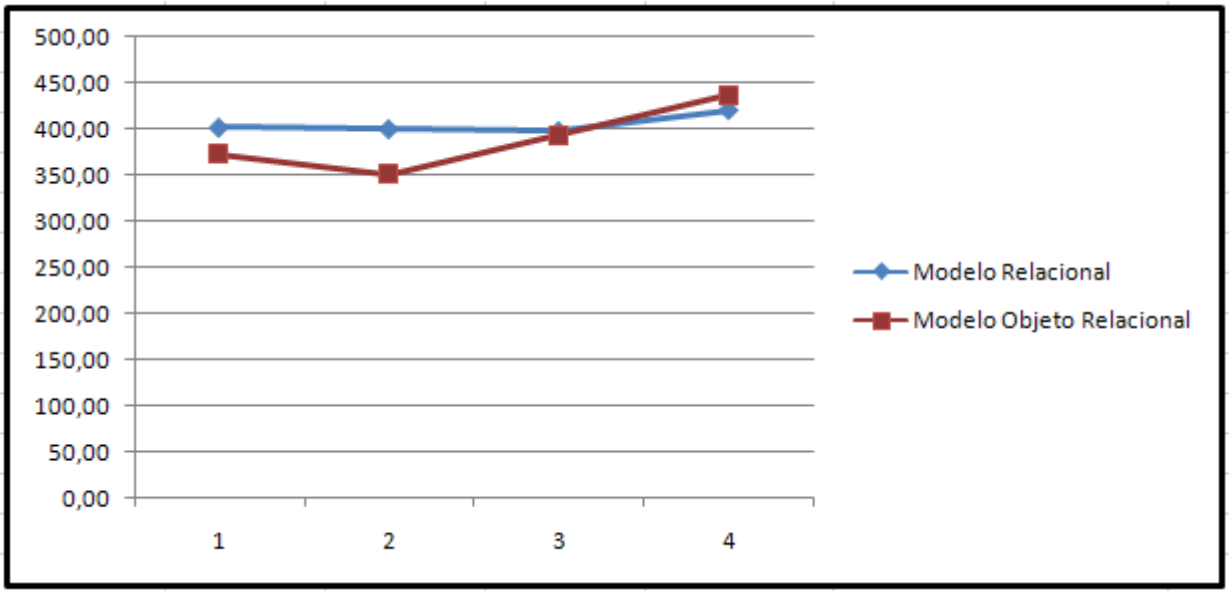

Figura 21. Resultados de las consultas de actualización

Al realizar un análisis de los resultados obtenidos sobre las consultas de actualización, se puede concluir:

En las consultas de actualización el Modelo Objeto Relacional se comporta mayormente con un mejor rendimiento que le Modelo Relacional, teniendo implementadas, como es el caso de la #1 y #2, características del Modelo Objeto Relacional. Al utilizarse los arreglos de mayor complejidad, el comportamiento es bastante parejo, ejemplo la #4. Al existir semejanza en la implementación de las consultas donde mayormente están implicadas característica del Modelo Relacional respecto al Modelo Objeto Relacional, el rendimiento de ambos modelos es muy similar, ejemplo en la #3. Esto significa que el Modelo Objeto Relacional como promedio el rendimiento para las acciones de actualización es mejor respecto al Modelo Relacional a pesar de trabajar con estructuras de datos más complejas.

3.7. Conclusiones de los análisis

Teniendo en cuenta según lo planteado por otros especialistas las operaciones mayormente realizadas sobre una base de datos en un 90% son de selección otro 8% es de inserción y el resto de actualización y/o eliminación. Asumiendo como premisa lo antes expuesto, el Modelo Objeto Relacional respecto al Modelo Relacional para las consultas de selección en la mayoría de los casos el rendimiento es mucho menor, esto implica que quede por debajo en una operación realizada frecuentemente por los usuario de las bases de datos.

No obstante es importante destacar la importancia que ofrecen la posibilidad de tener un registro con atributos que tengan tipos de datos compuesto, que sólo es posible tenerlo, al utilizarse el Modelo Objeto Relacional, y cabría preguntarse a la hora de realizar un diseño, ¿Es conveniente sacrificar un poco de rendimiento por organización? .Entiéndase que se habla de organización desde el punto de vista del diseño, al hacer uso de la herencia, de la agregación y/composición, combinado con las característica del Modelo Relacional.

En este capítulo se clasificaron las consultas que fueron utilizadas en la realización de las pruebas, así como se precisaron las condiciones sobre las cuales las pruebas fueron realizadas. Se mostraron los resultados obtenidos explicando en cada caso, cuando es factible la utilización del Modelo Objeto Relacional.

CONCLUSIONES

A partir de los resultados obtenidos por el cumplimiento de las tareas, se puede concluir lo siguiente:

- La elaboración del diseño de la base de datos siguiendo el Modelo Objeto Relacional expuso cómo coexisten en el diseño las relaciones de agregación, herencia y las del Modelo Relacional.

- Fueron analizadas varias herramientas para la realización del diseño de la base de datos finalmente se utilizó Enterprise Architect ya que ofrece un mayor número de prestaciones respecto a las otras herramientas.

- Los resultados de las pruebas mostraron que las fortalezas del modelo vienen dada por la capacidad de insertar y modificar registros cuyas estructuras sean complejas y no verse afectado el rendimiento en estas operaciones.

- Las debilidades del modelo aparecen para las consultas de selección, donde el rendimiento del Modelo Objeto Relacional fue menor de forma general comparado con el Modelo Relacional.

- Se considera que para aquellos sistemas de gestión cuyo modelo de datos implementado siguiendo el Modelo Relacional sea extenso, o sistemas que requieran de registros que tengan campos compuestos y se quiera una mejora desde el punto de vista de diseño, es factible el uso del Modelo Objeto Relacional.

- Para aquellos sistemas que la estructura de los datos que manejan no sean complejos desde el punto de vista de su composición, es recomendable el uso del Modelo Relacional.

RECOMENDACIONES

Es recomendable dar seguimiento a la investigación y para ello se propone lo siguiente:

- Realizar pruebas sobre el gestor de base de datos Oracle a una base de datos que esté implementada siguiendo el Modelo Objeto Relacional.
- Realizar una investigación que trate el tema de la normalización para las bases de datos que sigan el diseño del Modelo Objeto Relacional.
- Implementar una Sistema que utilice una base de datos que sea implementada siguiendo el Modelo Objeto Relacional, y analizar el comportamiento del rendimiento.

REFERENCIAS BIBLIOGRÁFICAS

1. Moisés, Daniel Toledo Díaz. *The Architecture of Enterprise Information Systems. A View based on patterns.* [En línea] [Citado el: 5 de 12 de 2008.] http://www.moisesdaniel.com/wri/eisarc.html.

2. Yanetsys Sarduy Domínguez y Pedro Urra González. *Revista Cubana de los Profesionales de la información y comunicación en Salud Acimed.* [En línea] 4 de Abril de 2006. http://www.bvs.sld.cu/revistas/aci/vol14_4_06/aci11406.htm.

3. Marqués Merche. Diseño de Sistemas de Bases de Datos. *Bases de datos orientadas a objetos.* [En línea] [Citado el: 10 de Diciembre de 2008.] http://www3.uji.es/~mmarques/e16/teoria/cap2.pdf..

4. *Idem a la referencia 3.*

5. Departamento de Sistemas y Computación del Instituto Tecnológico de La Paz. [En línea] [Citado el: 10 de Diciembre de 2008.] http://sistemas.itlp.edu.mx/tutoriales/desproyectos/tema3_1.htm.

6. *Aspectos técnicos.* **A Cruz Paz y VM García Suárez.** 1994.

7. Manuel Peralta. *Sistema de Información.* [En línea] [Citado el: 15 de Enero de 2009.] http://www.monografias.com/trabajos7/sisinf/sisinf.shtml.

8. SISTEMAS DE GESTIÓN DE BASE DE DATOS. [En línea] 14 de Febrero de 2005. [Citado el: 7 de Enero de 2009.] https://intranet.facyt.uc.edu.ve/.

9. Sistemas de Información y Bases de Datos. [En línea] 2007-2008. [Citado el: 12 de Febrero de 2009.] http://sinbad.dit.upm.es.

10. EMBARCADERO, Power To Your Platforms. [En línea] [Citado el: 15 de Febrero de 2009.] http://www.embarcadero.com/products/er_studio/.

11. SYBASE DE COLOMBIA. [En línea] [Citado el: 18 de Febrero de 2009.] http://www.mtbase.com/productos/modelamientometadatos/powerdesigner.

12. Sparx Systems. [En línea] [Citado el: 02 de Marzo de 2009.] http://www.sparxsystems.com.au/products/ea/index.html.

13. SQL Manager.net. [Online] [Cited: Marzo 20, 2009.] http://sqlmanager.net/.

14. EMS Data Generator for PostgreSQL. *Database Mangement Solutions.* [En línea] 2007. [Citado el: 19 de Marzo de 2009.] http://www.sqlmanager.net/en/products/postgresql/datagenerator.

15. EMS SQL Query for PostgreSQL. *Database Mangement Solutions.* [En línea] 2007. [Citado el: 19 de Marzo de 2009.] http://www.sqlmanager.net/en/products/postgresql/query .

BIBLIOGRAFÍA

C.J.Date. *Introducción a los Sistemas de Base de Datos.*

[En línea] [Citado el: 20 de 11 de 2008.] http://elies.rediris.es/elies9/4-2.htm.

Sección 2:**Los sistemas de base de datos y los SGBD**. [En línea] 2000. [Citado el: 25 de 11 de 2008.] http://tramullas.com/documatica/2-3.html.

Sistemas de Información y Bases de Datos. [En línea] [Citado el: 03 de 12 de 2008.] http://sinbad.dit.upm.es/docencia/grado/curso0708/BDOR%202%C2%AA%20Parte%20slices%20Modelo%20de%20Datos%20O-R%20Nov%202007.pdf.

Asociation for Computing Machinery. [En línea] [Citado el: 03 de 12 de 2008.] www.acm.org/crossroads/espanol/xrds7-3/ordbms.html .

Universidad Nacional del Nordeste. [En línea] [Citado el: 13 de 01 de 2009.] www.unne.edu.ar/Web/cyr/com2004/6-Biologia/B-057.pdf.

Departamento de Lenguajes y Sistemas Informaticos. [En línea] [Citado el: 15 de 01 de 2009.] www.kybele.etsii.urjc.es/docencia/BD/2008-2009/Material/%5BBD-2008-2009%5DTema3-ModeloOR.pdf.

University of Cambrigde. [En línea] www.cl.cam.ac.uk/teaching/2003/DBaseThy/or-manifesto.pdf .

El profesional de la informacion. *Revista Internacional Científica y profesional.* [En línea] http://www.elprofesionaldelainformacion.com/contenidos/1998/marzo/sistemas_de_bases_de_datos_relacionales_orientados_a_objetos_e_interfaces_vrml_para_el_www.html..

Bases de Datos Orientadas a Objeto y el estándar ODMG. Autores: Clara Martín Sastre y Enrique Medarde Caballero. http://tejo.usal.es/~fgarcia/docencia/poo/02-03/trabajos/S1T3.pdf

Grupo de Base de Datos. [En línea] [Citado el: 01 de 02 de 2009.] http://www.bd.cesma.usb.ve/ci5311/apuntesOR.pdf.

Postgrado en Informática. [En línea] [Citado el: 15 de 02 de 2009.] http://www.postgradoinformatica.edu.bo/enlaces/investigacion/pdf/INGSW3_68.pdf?PHPSESSID=323667a56c227b92ab6d9e52a3c88404.

Universidad de Carabobo. *Facultad Eperimental de Ciencias y Tecnologías.* [En línea] http://alfa.facyt.uc.edu.ve/computacion/pensum/cs0347/download/exposiciones/1/SGBD.pdf

Oracle Buys Sun. [En línea] http://www.oracle.com

Sybase. [En línea] http://www.sybase.com/products/powerdesigner.

Danysoft. [En línea] 1990. http://info.danysoft.com/free/model2.pdf.

Lockhart, Thomas. *Manual del usuario de PostgreSQL.*

PostgreSQL. [En línea] [Citado el: 03 de 03 de 2009.] http://www.postgresql.org/.

Sun Microsystems. [En línea] [Citado el: 8 de 04 de 2009.] http://sunsite.unam.mx/servidores/docs/bd/ANSI%20SQL.pdf .

[En línea] [Citado: Marzo 15, 2009.] http://www.sqlmanager.net/.

Ernesto Quiñones Arcárate. Asociación Peruana de Software Libre. [En línea] [Citado el: 10 de Febrero de 2009.] http://www.apesol.org.pe.

ANEXOS

Anexo1. Implementación de las consultas Sobre el Modelo Objeto Relacional y Modelo Relacional

Diseño de Casos de Prueba

Descripción General de las consultas de Selección

1: Obtener el primer movimiento y la misión sobre la cual se realizó el movimiento de los Colaboradores de No Salud cuya categoría ocupacional es 2.

2: Obtener cantidad de colaboradores en la misión, cuyo identificador de la localidad, misión y expediente es 1.

3: Obtener en nombre de la institución de los colaboradores cuyo tipo de misión es ATC-CD arreglar

4: Obtener el estado civil de los colaboradores cuyo id de localidad es 2

5: Obtener el id de expediente y la fecha de salida de los colaboradores de la salud ordenados descendentemente.

6: Obtener la fecha de entrada y de salida como fechas de los colaboradores cuyo identificador de misión es 1.

7: Obtener la especialidad y el identificador de los colaboradores de la salud evaluados de bien.

8: Obtener la fecha de inicio y de fin del primer movimiento de los colaboradores cuya categoría de misión es Funcionario MINSAP y el área geográfica de la localidad es América del sur.

9: El correo y el estado del expediente de los colaboradores que no son de la salud que están evaluados de bien y que son militantes del PCC.

10: El identificador de los pasaportes que este deteriorados y de los colaboradores que no son de la salud que pertenecen al INDER y que cumplen misión en Bolivia.

Descripción General de las consultas de Inserción

1: Insertar un colaborador**.**

2: Insertar una evaluación de la misión.

3: Insertar una misión.

4: Insertar un movimiento

5: Insertar una categoría de la misión.

Descripción General de las consultas de Actualización

1: Actualizar un movimiento con un identificador de condición.

2: Actualizar un pasaporte cuando el identificador es 1.

3: Actualizar una localidad con un identificador de condición.

4: Actualizar una militancia cuando el identificador es 1.

Consultas de Selección para el Modelo Objeto Relacional

Id del escenario	Escenario	Identificador de la misión	Primer movimiento de la misión	Resultado de la Prueba
1	SELECT public.mision.id_mision, public.mision.movimientos[1] FROM public.salud INNER JOIN public.mision ON (public.salud.id_colaborador = public.mision.id_mision) WHERE (public.salud.categoria_ocupacional = 2);	4	(16,2008-08-07,2008-08-08,Elsa)	*Satisfactorio*

Id del escenario	Escenario	Cantidad	Resultado de la Prueba
2	SELECT count(public.colaborador. id_colaborador) AS cantidad FROM public.colaborador INNER JOIN public.mision ON (public.colaborador. id_colaborador = public.mision. id_colaborador) INNER JOIN public.localidad ON (public.mision.id_localidad = public.localidad.id_localidad) WHERE (public.localidad.id_localidad = 1) AND (public.mision.id_mision=1) AND (expediente).id_expediente=1	2	Satisfactorio.

Id del escenario	Escenario	Nombre de la institución	Resultado de la Prueba
3	SELECT mision.nombre_institucion FROM misión INNER JOIN categoria_mision ON mision.id_categoriamision=categoria_mision.id_categoriamision WHERE categoria_mision.tipo_mision='ATC-CD'	Joaquin Lenzina	*Satisfactorio.*

Id del escenario	Escenario	Estado civil	Resultado de la Prueba
4	SELECT salud.estado_civil FROM salud INNER JOIN misión ON salud.id_colaborador=mision.id_colaborador INNER JOIN localidad ON mision.id_localidad=localidad.id_localidad WHERE localidad.id_localidad= 2	soltera/o	Satisfactorio.

Id del escenario	Escenario	Identificador del expediente	Fecha de salida	Resultado de la Prueba
5	SELECT (expediente).id_expediente, (expediente).fecha_salida FROM public.salud ORDER BY (expediente).id_expediente DESC	100	16/10/2009	Satisfactorio.

Id del escenario	Escenario	Fecha de entrada	Fecha de salida	Resultado de la Prueba
6	SELECT (expediente).fecha_entrada, (expediente).fecha_salida FROM public.no_salud INNER JOIN public.mision ON public.no_salud.id_colaborador= public.mision.id_colaborador WHERE public.mision.id_mision= 1	04/02/2009	05/03/2009	Satisfactorio.

Id del escenario	Escenario	Identificador del colaborador	Especialidad	Resultado de la Prueba
7	SELECT public.salud.id_colaborador, public.salud.especialidad FROM public.salud INNER JOIN public.mision ON public.salud.id_colaborador= public.mision. id_colaborador INNER JOIN public.evaluacion_mision ON public.mision. id_evaluacionmision =public.evaluacion_mision.id _evaliacionmision WHERE public.evaluacion_mision. evaluacion= 'B'	98	Ginecología y Obstetricia	Satisfactorio.

Id del escenario	Escenario	Fecha de inicio	Fecha de fin	Resultado de la Prueba
8	SELECT public.mision.movimientos[1].fecha_inicio, public.mision.movimientos[1].fecha_fin FROM public.mision INNER JOIN public.categoria_mision ON public.mision. id_categoriamision = public.categoria_mision. id_categoriamision INNER JOIN public.localidad ON public.mision.id_localidad= public.localidad.id_localidad WHERE public.localidad.area_geografica='América del Sur' AND public.categoria_mision. categoria= 'Funcionario MINSAP'	14/05/2008	16/05/2008	Satisfactorio.

Id del escenario	Escenario	expediente	Correo electrónico	Resultado de la Prueba
9	SELECT ((expediente).estado_expediente) AS Estado, public.no_salud. correo FROM public.no_salud INNER JOIN public.mision ON public.no_salud. id_colaborador =public.mision. id_colaborador INNER JOIN public.localidad ON public.mision.id_localidad= public.localidad.id_localidad INNER JOIN public.evaluacion_mision ON public.mision. id_evaluacionmision =public.evaluacion_mision.id_evaliacionmision WHERE public.localidad.nombre_pais= 'Eritrea' AND public.no_salud.militancias [2].siglas= 'PCC'AND public.evaluacion_mision. evaluacion = 'B'	completo	wserrano@mighty.nl.cu	Satisfactorio.

Id del escenario	Escenario	Identificador del colaborador	Estado del pasaporte	Resultado de la Prueba
10	SELECT public.no_salud. id_colaborador, public.no_salud.pasaportes [1].estado_pasaporte FROM public.no_salud INNER JOIN public.mision ON public.no_salud. id_colaborador= public.mision. id_colaborador INNER JOIN public.localidad ON public.mision.id_localidad= public.localidad.id_localida d WHERE public.no_salud.organismo ='INDER' AND public.localidad.nombre_pais= 'Bolivia'	96	Deteriorado	Satisfactorio.

Consultas de Selección para el Modelo Relacional

Id del escenario	Escenario	Identificador de la misión	Identificador del movimiento	Resultado de la Prueba
1	SELECT pp_Mision.id_mision, pp_Movimiento.id_mision FROM pp_Colaborador INNER JOIN pp_Mision ON(pp_Colaborador.idciudadano_col=pp_Mision.idciudadano_col) INNER JOIN pp_Movimiento ON (pp_Mision.id_mision= pp_Movimiento.id_mision) WHERE pp_Colaborador.id_categoria_ocupacional=2	4	4	*Satisfactorio*

Id del escenario	Escenario	Cantidad	Resultado de la Prueba
2	SELECT COUNT(pp_Colaborador.idciudadano_col)AS cantidad FROM pp_Colaborador INNER JOIN pp_Mision ON pp_Colaborador.idciudadano_col=pp_Mision.idciudadano_col INNER JOIN pp_Expediente ON pp_Expediente.idciudadano_col=pp_Mision.idciudadano_col INNER JOIN cod_Localidad ON cod_Localidad.id_localidad=pp_Mision.id_localidad WHERE cod_Localidad.id_localidad=1 AND pp_Mision.idciudadano_col=1 AND pp_Expediente.idciudadano_col=1	2	Satisfactorio.

Id del escenario	Escenario	Nombre de la institución	Resultado de la Prueba
3	SELECT pp_Mision.nombre_institucion FROM pp_Mision INNER JOIN cod_Tipo_Mision ON cod_Tipo_Mision.id_tipo_mision= pp_Mision.id_categoria_tipo_mision WHERE cod_Tipo_Mision.tipo_mision='ATC-CD'	Joaquin Lenzina	*Satisfactorio.*

Id del escenario	Escenario	Estado civil	Resultado de la Prueba
4	SELECT pp_Colaborador.estadocivil FROM pp_Colaborador INNER JOIN pp_Mision ON pp_Colaborador.idciudadano_col= pp_Mision.idciudadano_col INNER JOIN cod_Localidad ON pp_Mision.id_localidad= cod_Localidad.id_localidad WHERE cod_Localidad.id_localidad='2'	Slotero/a	Satisfactorio.

Id del escenario	Escenario	Identificador del expediente	Fecha de salida	Resultado de la Prueba
5	SELECT pp_Expediente.id_expediente, pp_Expediente.fecha_salida FROM pp_Expediente INNER JOIN pp_Colaborador ON pp_Expediente.idciudadano_col=pp_Colaborador.idciudadano_col ORDER BY pp_Expediente.id_expediente DESC	100	16/10/2009	Satisfactorio.

Id del escenario	Escenario	Fecha de entrada	Fecha de salida	Resultado de la Prueba
6	SELECT pp_Expediente.fecha_salida,pp_Expediente.fecha_entrada FROM pp_Expediente INNER JOIN pp_Colaborador ON (pp_Expediente.idciudadano_col = pp_Colaborador.idciudadano_col) INNER JOIN pp_Mision ON pp_Mision.idciudadano_col = pp_Colaborador.idciudadano_col WHERE pp_Mision.id_mision=1	05/02/2009	11/03/2009	Satisfactorio.

Id del escenario	Escenario	Id del colaborador	Especialidad	Resultado de la Prueba
7	SELECT pp_Colaborador.idciudadano_col, p_Colaborador.id_especialidad FROM pp_Colaborador INNER JOIN pp_Mision ON pp_Colaborador.idciudadano_col= pp_Mision.idciudadano_col INNER JOIN pc_Evaluacion_Mision ON pp_Mision.id_evaluacion= pc_Evaluacion_Mision.id_evaluacion INNER JOIN cod_Tipo_Evaluacion ON cod_Tipo_Evaluacion.id_tipo_evaluacion= pc_Evaluacion_Mision.id_tipo_evaluacion WHERE cod_Tipo_Evaluacion.evaluacion='B'	1	Ginecología y Obstetricia	Satisfactorio.

Id del escenario	Escenario	Fecha de inicio	Fecha de fin	Resultado de la Prueba
8	SELECT pp_Movimiento.fecha, pp_Movimiento.fecha_fin FROM pp_Movimiento INNER JOIN pp_Mision ON pp_Movimiento.id_mision= pp_Mision.id_mision INNER JOIN pp_Colaborador ON pp_Mision.idciudadano_col= pp_Colaborador.idciudadano_col INNER JOIN cod_Categoria_Mision ON pp_Mision.id_categoria_tipo_mision= cod_Categoria_Mision.id_categoria_mision INNER JOIN cod_Localidad ON pp_Mision.id_localidad= cod_Localidad.id_localidad INNER JOIN cod_Pais ON cod_Localidad.id_pais= cod_Pais.id_pais INNER JOIN cod_Area_Geografica ON cod_Pais.id_area_geografica = cod_Area_Geografica.id_area_geografica WHERE cod_Area_Geografica.nombre=' América del Sur' AND cod_Categoria_Mision.categoria_mision='Funcionario MINSAP'	10/05/2008	12/05/2008	Satisfactorio.

Id del escenario	Escenario	expediente	Correo electrónico	Resultado de la Prueba
9	SELECT pp_Colaborador.correo, (cod_Estado_Expediente.estado) AS estado FROM pp_Colaborador INNER JOIN pp_Expediente ON pp_Colaborador.idciudadano_col= pp_Expediente.idciudadano_col INNER JOIN cod_Estado_Expediente ON pp_Expediente.id_estado_expediente= cod_Estado_Expediente.id_estado_expediente INNER JOIN pp_Colaboradorcod_Militancia ON pp_Colaborador.idciudadano_col= pp_Colaboradorcod_Militancia.idciudadano_col INNER JOIN cod_Militancia ON cod_Militancia.id_militancia= pp_Colaboradorcod_Militancia.id_militancia INNER JOIN pp_Mision ON pp_Colaborador.idciudadano_col= pp_Mision.idciudadano_col INNER JOIN pc_Evaluacion_Mision ON pp_Mision.id_evaluacion= pc_Evaluacion_Mision.id_evaluacion INNER JOIN cod_Tipo_Evaluacion ON pc_Evaluacion_Mision.id_tipo_evaluacion= cod_Tipo_Evaluacion.id_tipo_evaluacion INNER JOIN cod_Localidad ON pp_Mision.id_localidad= cod_Localidad.id_localidad INNER JOIN cod_Pais ON cod_Localidad.id_pais= cod_Pais.id_pais WHERE cod_Militancia.siglas= 'PCC' AND cod_Tipo_Evaluacion.evaluacion= 'B'	incompleto	wserrano@mighty.nl.cu	Satisfactorio.

Id del escenario	Escenario	Identificador del colaborador	Estado del pasaporte	Resultado de la Prueba
10	SELECT pc_Pasaporte.id_pasaporte, pp_Colaborador.idciudadano_col FROM pp_Colaborador INNER JOIN pp_Mision ON pp_Colaborador.idciudadano_col=pp_Mision.idciudadano_col INNER JOIN pc_Pasaporte ON pp_Mision.id_mision= pc_Pasaporte.id_mision INNER JOIN cod_Localidad ON pp_Mision.id_localidad= cod_Localidad.id_localidad INNER JOIN cod_Pais ON cod_Localidad.id_pais= cod_Pais.id_pais WHERE pp_Colaborador.id_organismo='1' AND cod_Pais.nombre='Bolivia'	1	Deteriorado	Satisfactorio.

Consultas de Inserción para el Modelo Objeto Relacional

Id del escenario	Escenario	Resultado de la Prueba
1	INSERT INTO public.salud(id_colaborador, carnet, telefono, categoria_ocupacional, estado_civil, sexo, permiso, nombre_madre, nombre_padre, madre_vivo, padre_vivo, correo, pasaportes, expediente, militancias, especialidad, relevo, unidad_salud) VALUES (5002, 12321456987, 4545454, 1, 'casado', 'F', 'true', 'Alberta', 'Rafael', 'false', 'true', 'ymarti@netscape.sg.cu', array[row('5001','Marino', Prorroga','Vencido')::pasaporte], row('7','2009-10-14','2009-10-16','completo','Carlos'), array[row('1',' PCC','Partido Comunista Cubano')::militancia], 'Neurocirugia', 'Elisa', 1)	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
2	INSERT INTO evaluacion_mision (id_evaliacionmision, evaluacion, descripcion_evaluacion) VALUES (1, 'B', 'Fue una buena misión ');	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
3	INSERT INTO misión (id_mision, id_evaluacionmision, id_categoriamision, id_colaborador, id_localidad, fecha_inicio, fecha_fin, nombre_institucion, telefono_benef, tiempo_estancia, movimientos) VALUES (5005, 5005, 5000, 5000, 5000, '2008-03-10', '2010-01-25', 'Simón Bolívar', '2620494', 'un mes', ARRAY [ROW('56','2009-08-06','2009-08-08','Rafael')::movimiento,('57','2009-05-13','2009-05-15','Juan')::movimiento])	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
4	INSERT INTO movimiento (id_movimientos, fecha_inicio, fecha_fin, nombre_mueve) VALUES (93, '2010-07-30', '2010-08-07', 'Barbara')	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
5	INSERT INTO public.categoria_mision(id_categoriamision, tipo_mision, categoria, descripcion) VALUES (5003, 'PIS', 'PIS1 ', 'Programa Integral de Salud')	Satisfactorio.

Consultas de Inserción para el Modelo Relacional

Id del escenario	Escenario	Resultado de la Prueba
1	INSERT INTO pp_Colaborador (idciudadano_col, carnet_identidad, tipo, telefono, correo, id_categoria_ocupacional, id_estado, id_especialidad, id_relevo, id_organismo, id_tipoProfesional, id_provincia_trabajo, id_municipio_trabajo, estadocivil, id_unidad_salud, sexo) VALUES (5006,111113011,0,2620492,'Oberliebersbach Str. 9',1,5,2,3,1,4,1,1,2,1,'f')	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
2	INSERT INTO pc_Evaluacion_Mision (id_evaluacion, descripcion, id_tipo_evaluacion) VALUES (5003,'No Tiene',1)	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
3	INSERT INTO pp_Mision (id_mision, fecha_inicio, fecha_fin, id_evaluacion, idciudadano_col, id_categoria_tipo_mision, id_categoria_estado, id_localidad, tiempo_estancia, idcuidadano_autoriza, id_dedicacion, nombre_institucion, direccion_institucion, idciudadano_avisar_urgencia) VALUES (5009,'2009-05-29 07:12:30','2009-07-21 18:43:30',200,13,10,12,3,38,3,28,'Ignacia Zeballos','Am-Bruchweiher 175',5)	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
4	INSERT INTO pp_Movimiento (fecha, fecha_fin, idciudadano_mueve, motivo, id_categoria_estado, categoriaMision, id_localidad, fecha_pronostico, id_mision) VALUES ('2009-05-15 12:28:36','2009-05-23 15:58:32',51,'CKW818CONSTANT',52,51,51, '2009-02-25 01:50:45',5000)	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
5	INSERT INTO cod_Categoria_Mision (id_categoria_mision, categoria_mision, id_tipo_mision, Cod_clue) VALUES (6001,'Residente de Medicina Física y Rehabilitación',8,1665)	Satisfactorio.

Consultas de Actualización para el Modelo Objeto Relacional

Id del escenario	Escenario	Resultado de la Prueba
1	UPDATE public.mision SET movimientos = ARRAY[row('1','2009/02/03','2009/02/05','Luis')::movimiento] WHERE public.mision.id_mision= 2	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
2	UPDATE public.colaborador SET pasaportes = ARRAY [row(1,'marino','prorroga','vencido')::pasaporte] WHERE public.colaborador.id_colaborador=1	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
3	UPDATE public.localidad SET area_geografica = 'El Caribe', nombre_pais = 'Haiti' WHERE public.localidad.id_localidad=2	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
4	UPDATE public.salud SET militancias = ARRAY[row (1,'PCC','Partido Comunista de Cuba')::militancia,row (2,'UJC','Union de Jovenes Cominustas')::militancia] WHERE public.salud.id_colaborador= 1	Satisfactorio.

Consultas de Actualización para el Modelo Relacional

Id del escenario	Escenario	Resultado de la Prueba
1	UPDATE public.pp_movimiento SET motivo = 'traslado', categoriamision = 141, id_mision = 5000 WHERE public.pp_movimiento.id_localidad=149	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
2	UPDATE pc_Pasaporte SET motivo_viaje = 'CM', fecha_salida = '2009/02/06' WHERE pc_Pasaporte.id_pasaporte= 1	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
3	UPDATE public.cod_localidad SET descripcion = 'Bermejo', nombre = 'Bermejo', id_pais = 1 WHERE public.cod_localidad.id_localidad=1	Satisfactorio.

Id del escenario	Escenario	Resultado de la Prueba
4	UPDATE cod_Militancia SET id_militancia = '6000', nombre ='Partido Comunista de Cuba', siglas ='PCC' WHERE cod_Militancia.id_militancia=1	Satisfactorio.

Anexo2. Descripción de las tablas de la base de datos

Nombre: colaborador		
Descripción: Contiene los atributos comunes para ambos tipos de colaboradores.		
Atributo	**Tipo**	**Descripción**
id_colaborador	integer	Identificador del colaborador
carnet	decimal (11)	Número de carnet
teléfono	decimal (10)	Número de teléfono
categoría_ocupacional	integer	Categoría de la ocupación
estado_civil	varchar (20)	Estado Civil
sexo	Char(1)	Sexo
permiso	boolean	Si tiene permiso para cobrar
nombre_madre	varchar (30)	Nombre de la madre
nombre_padre	varchar (30)	Nombre del padre
madre_vivo	boolean	Si la madre esta viva
padre_vivo	boolean	Si el padre esta viva
correo	varchar (30)	Correo electrónico
pasaportes []	pasaporte	Arreglo de pasaportes que posee
expedientes	expediente	Expediente del colaborador
militancias []	militancia	Arreglo de militancias a las que pertenece

Nombre: mision		
Descripción: Contiene las misiones de cada uno de los colaboradores		
Atributo	**Tipo**	**Descripción**
id_misión	integer	Identificador de la misión
id_evaluacionmisión	integer	Identificador de la evaluación misión
id_categoriamisión	integer	Identificador de la categoría de la misión
id_colaborador	integer	Identificador del colaborador
id_localidad	integer	Identificador de la localidad
fecha_inicio	date	Fecha de inicio de la misión
fecha_fin	date	Fecha de fin de la misión
nombre_institución	Varchar(30)	Nombre de la institución donde realiza la misión
dirección_institución	Varchar(250)	Dirección de la institución donde realiza la misión
teléfono_benef	Varchar(30)	Teléfono del beneficiario
tiempo_estancia	Varchar(50)	Tiempo de estancia en la misión
movimientos[]	movimiento	Arreglo de movimientos que se le realizan en la misión

Nombre: movimiento		
Descripción: Contiene los atributos de cada movimiento		
Atributo	**Tipo**	**Descripción**
id_movimiento	integer	Identificador del movimiento
fecha_inicio	date	Fecha de inicio del movimiento
fecha_fin	date	Fecha del fin del movimiento
nombre_mueve	Varchar(50)	Nombre de el que realiza el movimiento

Nombre: no_salud		
Descripción: Contiene los atributos de los colaboradores que no pertenecen a la salud		
Atributo	**Tipo**	**Descripción**
ocupación_laboral	Varchar(50)	Ocupación laboral
municipio	Varchar(30)	Municipio donde vive
provincia	Varchar(30)	Provincia donde vive
organismo	text	Organismo al que pertenece

Nombre: salud		
Descripción: Contiene los atributos de los colaboradores que pertenecen a la salud		
Atributo	**Tipo**	**Descripción**
especialidad	Varchar(50)	Especialidad a la que se dedica
unidad_salud	integer	Unidad de Salud
relevo	Varchar(30)	Nombre del relevo en la misión

Nombre: militancia		
Descripción: Contiene los atributos de las militancias de cada colaborador		
Atributo	**Tipo**	**Descripción**
id_militancia	integer	Identificador de la militancia
siglas	Varchar(10)	Siglas de la militancia
nombre	Varchar(60)	Nombre completo de la militancia

Nombre: pasaporte		
Descripción: Contiene los atributos de los pasaportes de cada colaborador		
Atributo	**Tipo**	**Descripción**
id_pasaporte	integer	Identificador del pasaporte
tipo_pasaporte	Varchar(30)	Tipo de pasaporte
trámite_pasaporte	Varchar(30)	Tramite del pasaporte
estado_pasaporte	Varchar(250)	Estado en que se encuentra el pasaporte

Nombre: categoria_mision		
Descripción: Contiene los atributos de la categoría que tiene cada misión		
Atributo	**Tipo**	**Descripción**
id_categoriamision	integer	Identificador de la categoría de la misión
tipo_misión	Varchar(30)	Tipo de misión
Categoría	Varchar(50)	Siglas de la categoría de la misión
descripción	Varchar(250)	Descripción de la categoría

Nombre: evaluacion_mision		
Descripción: Contiene los atributos de la evaluación que tiene cada misión		
Atributo	**Tipo**	**Descripción**
id_evaluacionmision	integer	Identificador de la evaluación de la misión
evaluacion	Char(10)	Evaluación de la misión
descripcion_evaluacion	Varchar(250)	Descripción sobre la evaluación

Nombre: localidad		
Descripción: Contiene los atributos de la localidad donde se realiza la misión		
Atributo	**Tipo**	**Descripción**
id_localidad	integer	Identificador del la localidad de la misión
area_geografica	Varchar(30)	Área geográfica
nombre_pais	Varchar(50)	Nombre del país donde se realiza la misión