Título: Base de Datos para el Sistema Informativo de la Administración Provincial de Artemisa (SIGOB).

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Edel Pedrol Alvarez

Tutor(as): Ing. Dania Fernández Aguilar
Msc. Isleny Orta Rodríguez

Artemisa, Julio 2012

Declaro ser autor de la presente tesis y reconozco 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 ____ días del mes de _____ del año _____

_____
Edel Pedrol Alvarez
Autor

_____
Ing. Dania Fernández Aguilar
Tutor(a)

_____
MSc. Isleny Orta Rodríguez
Co-Tutor(a)

"Cuando se es joven, se crea. Cuando se es inteligente, se produce. No se adapta, se innova: la mediania copia; la originalidad se atreve".

José Martí

Con este trabajo de diploma quiero agradecerle:

- A mis padres por ayudarme en todo momento con sus consejos, por guiarme siempre por el camino correcto de la vida, por la confianza depositada en mí, por el sacrificio, la dedicación, el cariño y el apoyo incondicional.
- A mi hermana por apoyarme y aconsejarme en los momentos difíciles.
- A Guelmis por toda su dedicación en largas horas de trabajo, por comprenderme y apoyarme en los momentos necesarios, por orientarme y ayudarme en mi preparación como profesional.
- A mis amigos, gracias por haber compartido conmigo tantos momentos durante estos años.
- Le agradezco a mi tutora y co-tutora por su preocupación y orientación.
- A todos los profesores que de una forma u otra han contribuido en mi formación profesional.

Dedico este trabajo de diploma:

- A mis padres por haber depositado toda la confianza en mí.
- A todas aquellas personas que me quieren y puedan sentirse orgullosos.
- A mi familia.

EL presente trabajo de diploma se realizó en conjunto con los proyectos desarrollados en la Facultad Regional Mártires de Artemisa en el periodo 2011-2012, con el objetivo de desarrollar una base de datos (BD) para el Sistema Informativo de la Administración Provincial de Artemisa que garantice la integridad, seguridad y el acceso concurrente a los datos gestionados en cada una de sus direcciones.

Para la creación de la solución se utilizó la metodología SXP para guiar el proceso de desarrollo del software.

Como propuesta de solución se obtuvo una base de datos relacional para el sistema Informativo de la Administración Provincial de Artemisa (SIGOB) haciendo uso de las siguientes herramientas para el desarrollo: PostgreSQL como sistema gestor de base de datos (SGBD), para gestionar y manejar el diseño de la BD el Power Architect, como herramienta para implementar la capa de acceso a datos ¨Hibernate¨, PgAdmin como herramienta administradora del SGBD y NetBeans como IDE de programación.

Una vez elaborada la propuesta de solución se validaron los resultados a través de pruebas teóricas para comprobar la integridad, seguridad y redundancia de los datos; y las pruebas funcionales para observar y comprobar el comportamiento de la BD obteniéndose resultados satisfactorios que responden a los requisitos del cliente.

Palabras claves: Acceso concurrente, integridad, redundancia, seguridad, sistema informativo.

Introducción

A lo largo de los años las tecnologías informáticas se han desarrollado a un ritmo acelerado de modo que en la actualidad la influencia de las mismas es evidente en la vida del hombre, ya sea en sus negocios o en su vida social. Debido a este desarrollo constante las Tecnologías de la Información y las Comunicaciones (TICs) se han convertido en un medio imprescindible para todas las sociedades del mundo puesto que han revolucionado el modo de hacer las cosas.

En todos los países las empresas han reconocido la necesidad de realizar estas transformaciones y aún aquellas que carecen de recursos han tratado de convertir en una prioridad aumentar el uso de estas novedosas tecnologías, debido a que se persigue el objetivo de mejorar el nivel de vida de sus ciudadanos en el ámbito económico, social y cultural.

A partir de estas transformaciones se comenzaron a desarrollar aplicaciones o sistemas informáticos que tendrían como principal objetivo lograr la automatización de toda la población en cada una de las esferas de la vida.

Estos sistemas informáticos que se desarrollan manejan información relevante que necesita ser almacenada o persistida a medida que pase el tiempo, de modo que después pueda ser consultada por quienes lo necesiten. Persistir los datos manejados por un sistema permite mejorar procesos como la obtención de reportes, la toma de decisiones o sencillamente el resguardo de datos históricos.

Al igual que las TIC han evolucionado, también se han desarrollado los software con el paso de los años. Uno de los aspectos en que han evolucionado es en el desarrollo de tecnologías de almacenamiento para diferentes cantidades de información manejadas.

Esta evolución ha permitido la creación de modos de almacenamiento para pequeñas cantidades de información como son los ficheros XML, hasta llegar a las bases de datos que permiten almacenar enormes cantidades de información.

Las bases de datos propician la toma de decisiones a través del uso de un lenguaje que se creó específicamente para las mismas denominado Structured Query Languaje (SQL). A través del uso del mismo se propicia la toma de decisiones y se posibilita manejar los datos a través de consultas donde se analizan volúmenes considerables de información.

Estas bases de datos utilizadas no podían ser mantenidas con la integridad, seguridad y confidencialidad que se necesitaba y es por ello que se crean los Sistemas de Gestión de Bases de datos (SGBD) de modo que facilitaran el proceso de diseño de aplicaciones, proporcionaran tratamientos más eficientes con más rapidez y dieran la mayor flexibilidad posible a los usuarios.

La necesidad de tener una visión global de la empresa y de interrelacionar diferentes aplicaciones que utilizan BD diferentes, junto con la facilidad que proporciona el uso de las redes para la intercomunicación entre ordenadores ha conducido al desarrollo de los SGBD actuales. Estos permiten que un programa pueda trabajar con diferentes BD como si se tratase de una sola, esto es lo que se conoce como base de datos distribuida.

Esta distribución ideal se consigue cuando las diferentes BD son soportadas por una misma marca de SGBD, es decir, cuando existe homogeneidad; sin embargo, esto no es tan sencillo si los SGBD son heterogéneos.

En la actualidad, gracias principalmente a la estandarización del lenguaje SQL, los SGBD de marcas diferentes pueden darse servicio unos a otros y colaborar para proporcionar servicio a un programa de aplicación.

En el mundo actualmente debido a la necesidad de informatizar los procesos de las empresas u organizaciones se utilizan los Sistemas Informativos (SI), puesto que trabajar en ellos permite tener el poder de los datos de una organización y así poder incidir sobre ellos en la gestión oportuna de la información y el conocimiento organizacional, para la toma oportuna de decisiones institucionales.

Estos SI necesitan de sistemas de bases de datos que permitan almacenar las grandes cantidades de información generada de una forma organizada, garantizando que esta sea accesible de modo que ahorre tiempo y esfuerzo acorde con el uso de las nuevas tecnologías de la información.

Por las razones antes mencionadas se puede advertir y comprender la necesidad y el impacto del programa de Informatización de la Sociedad Cubana que se ha venido consolidando en estos últimos años, en el cual ha participado activamente la Universidad de las Ciencias Informáticas (UCI) junto a sus tres facultades regionales ubicadas en las provincias de Artemisa, Ciego de Ávila y Granma.

Como parte de este programa nuestro país se ha trazado como objetivo informatizar los procesos de sus principales instituciones y organizaciones en cada una de sus provincias. La estrategia a seguir consiste en el desarrollo de sistemas de información, tratando de remediar problemas de carácter organizativo para lograr la correcta gestión de la información con el uso de sistemas de bases de datos que faciliten el trabajo en los mismos.

Una de las provincias que se caracterizaba por poseer graves problemas organizativos era La Habana, debido a esto la dirección del país decidió en el año 2010 que a partir del 2011 la misma seria dividida en dos nuevas provincias que

llevarían como nombre Artemisa y Mayabeque con el objetivo de lograr un perfeccionamiento estructural y funcional de la administración y del Gobierno.

Al surgir la provincia de Artemisa se hace necesaria la restructuración de los órganos de dirección políticos y de masas, identificando de este modo la necesidad de crear un organismo que llevara el control sobre la misma. De esta forma se concibe la creación de la Administración Provincial de Artemisa (AP).

La AP tiene como objetivo principal preparar y proponer la política Integral del Estado del Consejo de la Administración Pública y una vez aprobada la misma, se encarga de dirigir, ejecutar, controlar y coordinar el cumplimiento de los procesos asociados a la provincia.

La AP en su estructura organizativa interna cuenta con 32 direcciones que controlan todo lo que acontece en la provincia que posea relevancia para el país, como por ejemplo las estadísticas de la mortalidad infantil, cantidad de estudiantes en enseñanza superior, la cantidad de cabezas de ganado existentes, entre otras.

Debido a las grandes cantidades de información generada por las direcciones se hizo necesario centralizar los datos en seis módulos tales como: Presidencia, Órganos de Dirección, Unidad de Aseguramiento, Sistema Empresarial, Órgano Auxiliar y Órgano Consultorio.

Estos módulos fueron creados con el fin de controlar el manejo de los procesos de las 32 direcciones de la AP para lograr una mayor organización en el trabajo de las mismas. Cada dirección posee una base de datos independiente con la información que le pertenece a la misma, provocando de este modo problemas de organización así como la falta de integridad al realizar actualizaciones.

También se evidencian graves problemas de seguridad, pues no están definidos niveles de accesibilidad a la información que se maneja y esta pudiera ser alterada por fuentes no confiables, además el usuario que necesite en un momento determinado obtener información de varias direcciones simultáneamente tendrá que realizar búsquedas por muchas tablas por lo que se hace difícil generar reportes inmediatos con la calidad requerida, además de que se invierten grandes cantidades de horas hombre en la búsqueda y consulta de estos datos.

Partiendo de la necesidad de lograr el éxito del proyecto “Sistema Informativo de la Administración Provincial de Artemisa” y al ser la base de datos un elemento vital dentro de la gestión de la información, se arriba al siguiente **problema de investigación**:

¿Cómo mejorar los procesos de almacenamiento de datos existentes en la Administración Provincial de manera que se contribuya a la integridad, seguridad y el acceso concurrente de los datos gestionados en cada una de sus direcciones?

Posteriormente de haber realizado un análisis de la situación actual, la investigación enmarca su **objeto de estudio en:** proceso de gestión de datos, delimitando el **campo de acción:** Base de Datos relacional relativa a los procesos de almacenamiento de datos.

Para darle cumplimiento a la investigación se propone, como **objetivo general:** Desarrollar la Base de Datos para el Sistema Informativo de la Administración Provincial de Artemisa que contribuya a la integridad, seguridad y el acceso concurrente de los datos gestionados en cada una de sus direcciones.

Para darle cumplimiento al objetivo general que se propone se definen como **Objetivos específicos**:

1. Elaborar la Fundamentación Teórica de la investigación.

2. Diseñar la Base de Datos para el Sistema Informativo de la AP.
3. Implementar la capa de acceso a datos del módulo de administrativo.
4. Validar los resultados obtenidos.

Teniendo en cuenta la situación descrita se formuló la siguiente **idea a defender:** Mediante el diseño y la implementación de la base de datos relacional del Sistema Informativo de la Administración Provincial de Artemisa se contribuirá a la integridad, seguridad y el acceso concurrente de los datos gestionados en cada una de sus direcciones.

Para darle cumplimiento a los objetivos específicos se definen las siguientes **Tareas de la Investigación:**

1. Definición de la fundamentación teórica de la investigación.
2. Diseño de la Base de Datos relacional para el Sistema Informativo de la AP.
3. Implementación de la capa de acceso a datos del módulo de administración del Sistema Informativo de la AP.
4. Realización de pruebas funcionales a la BD y a la capa de acceso del sistema de gestión.

Para dar cumplimiento a las tareas de la investigación propuestas anteriormente se emplearon métodos científicos de la investigación **Teóricos y Empíricos.**

Como métodos teóricos se utilizaron:

- **Analítico – Sintético:** Este método fue utilizado para realizar un análisis de las tendencias actuales en cuanto al diseño e implementación de bases de datos relacionales, se tuvo en cuenta para esto, los requisitos del sistema y de ahí se sintetizó como realizar la propuesta planteada.

- **Análisis Histórico – Lógico:** Permite analizar la trayectoria completa de los fenómenos de gestión de la información y gestión de datos, que se llevaba a cabo para la creación de los distintos modelos de bases de datos existentes, realizando un estudio histórico de los mismos, lo que permitió definir deficiencias y proponer soluciones acorde a las necesidades de la organización.

- **Modelación:** Se modela el problema planteado, creando una arquitectura y modelo de datos, que cumplen con el objetivo de dar una mejor solución a la situación presente. Se crean abstracciones con el objetivo de explicar la realidad.

Como métodos empíricos:

- **Entrevistas:** Para darle cumplimiento a este método se realizaron entrevistas a especialistas, con el objetivo de profundizar, obtener información y captar ideas para diseñar e implementar la Base Datos.

Se identificó como **población y muestra** a los trabajadores que interactúan con los modelos de los procesos de cada una de las direcciones de la AP que representan un 100% empleando el muestreo intencional.

Variable Independiente: Base de datos relacional para el Sistema Informativo de la AP.

Variable Dependiente: La integridad, seguridad y el acceso concurrente de los datos gestionados en cada una de sus direcciones.

Una vez concluido el proyecto se tendrá como **aporte práctico** un Base de datos relacional para el Sistema Informativo de la AP.

Este documento está estructurado por tres capítulos los cuales recogen todo el proceso realizado:

Capítulo 1: Fundamentación teórica: En este capítulo se realiza una fundamentación teórica sobre la que se basa este trabajo de diploma, teniendo en cuenta la tecnología, herramientas y gestores empleados para la realización de la propuesta.

Capítulo 2: Diseño de Bases de Datos Relacionales: En este capítulo se describen los patrones de diseño y metodologías a utilizar en el diseño de la BD. También la creación de varios esquemas, propuesta de integración de nomencladores comunes, normalización, además de una descripción detallada de las entidades para el esquema de administración de la base de datos SIBOB.

Capítulo 3: Implementación de la capa de acceso a datos y validación del diseño realizado: Implementación de la capa de acceso a datos, mostrando el uso de Hibernate y de otros elementos importantes en el proceso. Se realiza la validación teórica y funcional del diseño de la base de datos teniendo en cuenta aspectos como la integridad, la seguridad, el análisis de la redundancia de la información y cómo responde el sistema a las pruebas.

Capítulo 1. Fundamentación Teórica

Introducción

En el siguiente capítulo se abordan aspectos importantes de las Bases de Datos, desde su surgimiento hasta la actualidad, además de conceptos y características asociadas a las mismas. Se realiza un análisis de los diferentes elementos significativos relacionados con la arquitectura y diseño de las BD. Se abordan las tendencias tecnológicas y herramientas más usadas para lograr el correcto funcionamiento de las mismas y sustentar a través de estos conocimientos la solución del sistema Informativo de la AP.

1.1. Conceptos asociados al dominio del problema.

Originalmente las aplicaciones cubrían necesidades muy específicas de procesamiento, se centraban en una tarea específica, el modo en que se almacenaba y trabajaba con la información en las mismas era a través de los sistemas de ficheros.

Los sistemas de ficheros son un conjunto de programas con los que trabajaban los usuarios finales de una aplicación, estos permitían que cada programa controlara sus propios datos. El trabajo con los sistemas de ficheros traía asociadas desventajas considerables, tales como:

- Redundancia e inconsistencia de los datos.
- Dependencia de los datos física-lógica.
- Dificultad para tener acceso a los datos, proliferación de programas.
- Separación y aislamiento de los datos.
- Dificultad para el acceso concurrente.
- Dependencia de la estructura del fichero con el lenguaje de programación.
- Problemas en la seguridad de los datos.
- Problemas de integridad de datos.

Debido a la necesidad de representar de forma eficiente los fenómenos o procesos de la realidad objetiva de un modo en el que se pudieran establecer determinados vínculos entre los elementos u objetos que forman parte de ella, surgen las bases de datos y los sistemas de gestión de bases de datos. Con el surgimiento de las bases de datos se evitan las inconsistencias que se producían por la utilización de los mismos datos lógicos desde distintos archivos a través de procesos independientes.

Bases de Datos

En el mundo existen múltiples definiciones sobre las bases de datos, entre ellas destacan:

¨Una **base de datos** es una colección de información organizada de forma que un programa de ordenador pueda seleccionar rápidamente los fragmentos de datos que necesite. Una base de datos es un sistema de archivos electrónico.¨ (Masadelante.com, 1999-2011)

¨Una Base de Datos (BD) es un conjunto de datos que modelan hechos y objetos de una parcela de la realidad y sirven de soporte a una aplicación informática. Dichos datos deben estar almacenados físicamente en forma de ficheros informáticos y deben estar relacionados entre sí mediante una determinada estructura lógica.¨ (Riscos Núñez, 2010-2011)

¨Conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, que una BD puede considerarse una colección de datos variables en el tiempo.¨ (CUJAE, 2011)

Conjunto de datos almacenados sin redundancias en un soporte de acceso directo. Los datos están interrelacionados y estructurados de acuerdo a un modelo que sea

capaz de recoger el máximo contenido semántico; su finalidad es servir a una o más aplicaciones de la mejor forma posible. Los datos se almacenan de modo que resulten independientes de los programas que los usan; se emplean métodos para incluir nuevos datos y para modificar o extraer los datos almacenados.

El autor de la presente tesis de grado una vez analizados los conceptos mencionados anteriormente realiza la definición operacional de base de datos siguiente:

Una base de datos (BD) es un conjunto de datos agrupados o estructurados que están interrelacionados entre si y pueden ser variables en el tiempo. Su finalidad es ser utilizada en un sistema informativo o de aplicaciones de una entidad u organismo combinando los datos de manera que parezcan estar en una sola ubicación.

Sistemas de Gestión de Bases de Datos

Un sistema de Gestión de la Base de Datos (SGBD) no es más que un conjunto coordinado de programas, procedimientos, lenguajes, etc., que suministra a los distintos tipos de usuarios los medios necesarios para describir y manipular los datos almacenados en la base, garantizando su seguridad. Las operaciones típicas que debe realizar un SGBD pueden resumirse en aquellas que afectan a la totalidad de los datos (o a todos los registros de un determinado tipo) y las que tienen lugar sobre registros concretos.

Entre las principales definiciones que se conocen hoy en el mundo acerca de los sistemas de gestión de bases de datos figuran:

¨Un Sistema de Gestión de Bases de Datos (SGBD) consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a los mismos.¨ (SIGMUR, 2011)

¨Un Sistema de Gestión de Bases de Datos (SGBD) es el conjunto de programas que permiten definir, manipular y utilizar la información que contienen las bases de datos, realizar todas las tareas de administración necesarias para mantenerlas operativas, mantener su integridad, confidencialidad y seguridad. Una BD nunca se accede o manipula directamente sino a través del SGBD. Se puede considerar al SGBD como la interfaz entre el usuario y la BD.¨ (Nc, 2011)

De los conceptos mencionados anteriormente se selecciona para este trabajo de diploma el definido por el autor ¨NC Jeisson, 2011¨.

1.2. Clasificaciones de las bases de datos

Las bases de datos se pueden clasificar de diferentes formas.

Según la forma en que cambia la información almacenada:

- Estáticas: en estas no se puede modificar la información que almacenan o sea, son de sólo lectura permitiendo únicamente consultar datos. Este tipo es utilizado básicamente para almacenar datos acumulativos o históricos sobre los cuales se podrán realizar estudios acerca de su comportamiento.

- Dinámicas: Son aquellas bases de datos en las que la información almacenada cambia constantemente o a lo largo del tiempo. Sobre ellas se pueden realizar operaciones tales como: actualización, borrado y adición de datos, además de las operaciones fundamentales de consulta.

Según la información que almacenan:

- Bibliográficas: contienen un representante de la fuente primaria, que permite localizarla. Una base de datos de este tipo posee como registro típico información sobre el autor, fecha de publicación, etc. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo.

- De texto Completo: **a**lmacenan las fuentes primarias, no fragmentos ni referencias de las mismas.

- Directorios.
- Banco.

1.3. Arquitectura de las bases de datos.

La búsqueda de la eficiencia conduce al diseño de estructuras complejas para usuarios sin conocimientos de computación, para lo cual esta complejidad ha de estar oculta. Existen tres características importantes inherentes a los SBD: la separación entre los programas de aplicación y los datos, el manejo de múltiples vistas por parte de los usuarios y el uso de un catálogo para almacenar el esquema de la base de datos. Para poder lograr lo anterior es necesario definir los distintos niveles de abstracción de una base de datos, lo que constituirá el marco necesario para identificar las diferentes funciones que han de cumplir estos sistemas.

1.3.1. Arquitectura de tres niveles

El objetivo principal de la arquitectura ANSI/SPARC es definir un SGBD con el máximo grado de independencia. Para ello se utilizan tres niveles de abstracción conocidos como interno, conceptual y externo, tal como se muestra en la Fig1.

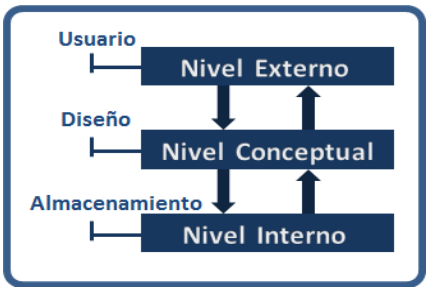

Fig1. Arquitectura en 3 niveles

Nivel Interno: es el nivel más cercano al almacenamiento físico de los datos. Permite escribirlos tal y como están almacenados en el ordenador. En este nivel se diseñan los archivos que contienen la información, la ubicación de los mismos y su organización, es decir se crean los archivos de configuración. (Álvarez, 2007)

Nivel conceptual: Contiene un esquema conceptual, que describe la estructura de toda la base de datos para una comunidad de usuarios. Este esquema hace transparentes los detalles de las estructuras físicas de almacenamiento y se centra en describir entidades, tipos de datos, vínculos, operaciones de los usuarios y restricciones.

Nivel externo: Describe varios esquemas externos o vistas de usuario. Cada uno describe la parte de la base de datos que le interesa y lo oculta al resto de grupo. La información se manipula sin saber cómo está almacenada internamente (nivel interno) ni su organización (nivel conceptual). Es el más cercano al usuario.

1.3.2. Tipos de Arquitectura

Actualmente los sistemas de BD son clasificados según la distribución de sus procesos y el nivel de soporte de datos que posean. Existen dos tipos principales de arquitectura de BD que permiten analizar como quedaran ubicados y organizados los datos, estos son:

Arquitectura Centralizada:

Es una base de datos almacenada físicamente en un solo lugar específico, es decir, es una base de datos almacenada en una sola PC y una sola CPU. Esta arquitectura tiene dos partes fundamentales: la de descripción de datos y la de manipulación de datos, organizadas en torno al diccionario de datos. Estas dos partes se organizan en torno a los tres niveles de la arquitectura.

Ventajas:

- Evita la redundancia e inconsistencia de los datos.
- Puede conservarse la integridad.
- Brinda seguridad a la información almacenada.
- Pueden aplicarse restricciones de seguridad.
- Instalación poco costosa.

Desventajas:

- Cuando un sistema de Base de Datos Centralizada falla, se pierde toda la disponibilidad de procesamiento y sobre todo de la información confiada al sistema.
- En caso de catástrofes o desastres, la recuperación de los datos es difícil de sincronizar.
- No ofrecen mejor proporción precio/rendimiento que los microprocesadores de los sistemas distribuidos.

A continuación la Figura 2 muestra cómo un SBD Centralizado procesa las interacciones entre el usuario final y la base de datos. Dichas actividades de intercambio ocurren de manera transparente para el usuario.

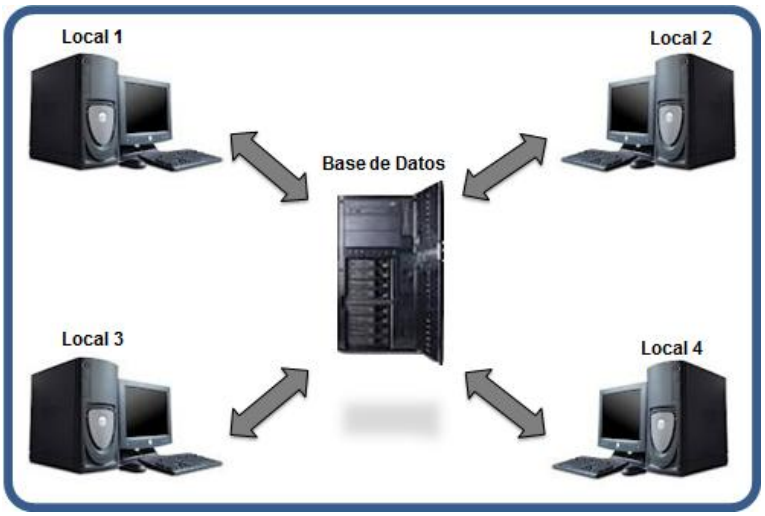

Fig2. Sistema de Base de Datos Centralizado

Arquitectura Distribuida:

Un sistema distribuido de base de datos consiste en un conjunto de localidades, conectados mediante una red de computadoras. Cada localidad puede procesar transacciones locales, o bien transacciones globales entre varias localidades, requiriendo para ello comunicación entre ellas. El diseño de una base de datos distribuida debe tener en cuenta cómo dividir la base de datos en fragmentos, cuál de ellos replicar y dónde localizar esos fragmentos y réplicas.

Ventajas:

- Mayor expansibilidad.
- Mayor fiabilidad y control de los datos.
- Mejora el rendimiento.
- Mayor compartición y disponibilidad de los datos.

Desventajas:

- Mayor complejidad.
- Redundancia de la información.

- Aumento del tráfico de comunicación.
- Mayor Costo.

La figura 3 muestra un ejemplo de SBD distribuido.

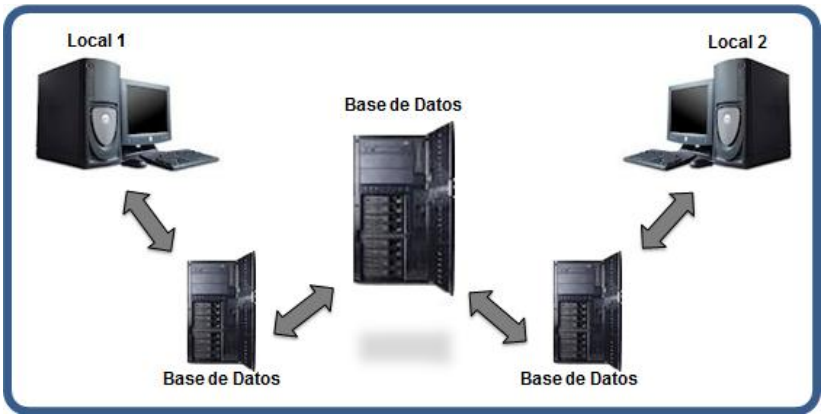

Fig3. Sistema de Base de Datos Distribuida.

Fundamentos de la selección

Para el desarrollo de la solución propuesta se selecciona la arquitectura centralizada debido a que la misma se ajusta a la estructura interna de la AP, dándole al proceso de almacenamiento de datos de cada una de las direcciones una mayor calidad en cuanto a la integridad, seguridad y eliminación de la redundancia de la información.

1.4. Diseño de base de datos.

El correcto diseño de una base de datos garantiza obtener el acceso a información exacta y actualizada. Un diseño correcto es esencial para lograr los objetivos fijados, o sea que la BD termine adaptándose a sus necesidades y pueda modificarse fácilmente.

Diseñar una base de datos es un proceso que lleva una serie de aspectos que se deben cumplir, como evitar la duplicidad de información y garantizar que la información sea correcta y completa.

Para realizar un buen diseño se debe dividir la información en tablas basadas en temas para disminuir la duplicidad de los datos, debe garantizarse la exactitud e integridad de los datos almacenados y satisfacer las necesidades de procesamiento de los mismos así como posibilitar la generación de informes cuando sean solicitados.

1.4.1. Fases del diseño de las bases de datos

Para diseñar una base de datos se debe seguir una metodología que consta de seis fases fundamentales:

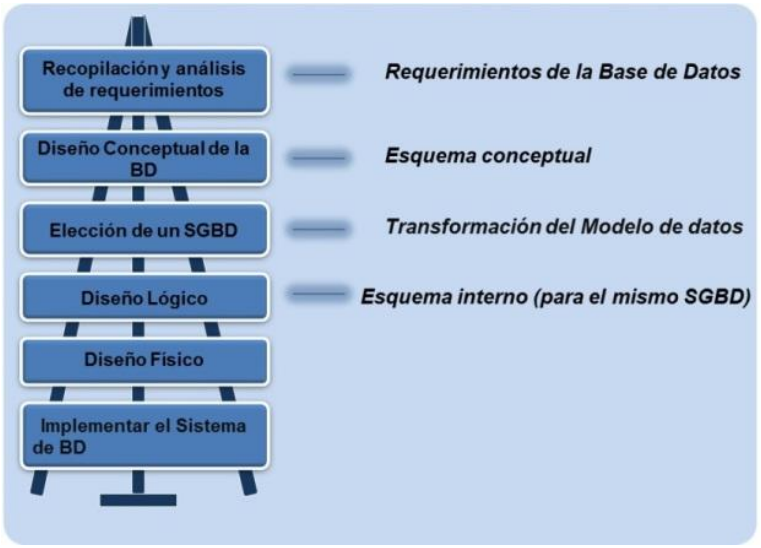

Fig4. Fases del Diseño de una base de datos

Fase 1: Recopilación y Análisis de Requerimientos: Para realizar el levantamiento de los requerimientos necesarios para la base de datos los diseñadores se reúnen con los usuarios finales del sistema para recoger y documentar sus necesidades de información. Además se definirán los requisitos funcionales de la aplicación que se convertirán en transacciones que se aplicarán a la BD para obtener o actualizar datos. En esta fase principalmente se centra todo el trabajo en definir qué es lo que se va a representar.

Fase 2: Diseño Conceptual de la Base de Datos: En esta fase se realiza el diseño del esquema conceptual de la BD y el diseño de las transacciones, o sea es donde se piensa en cómo se va a proceder para representar lo antes definido en la fase 1.

Para crear un esquema conceptual se realiza un modelo de datos conceptual de alto nivel, que no puede utilizarse para implementar directamente la estructura de la base de datos. El esquema conceptual debe contener una descripción detallada de los requerimientos de información de los usuarios e incluir descripciones de los tipos de datos, relaciones entre ellos y restricciones.

Fase 3: Elección de un SGBD: La elección de un SGBD es una fase muy importante en el diseño de la BD pues se deben considerar diferentes factores. Algunos factores a considerar son técnicos, económicos, organizativos, de rendimiento, etc. Sin embargo, resulta difícil la medida y la forma de examinar y considerar los diferentes factores.

Fase 4: Transformación del Modelo de Datos o Fase de Diseño Lógico: El diseño lógico depende de la realización del esquema conceptual. Un esquema lógico no es más que la descripción de la estructura de la base de datos que puede procesarse por el Sistema Gestor de Base de Datos. Esta fase es la más cercana a la implementación de un sistema manejador de bases de datos, con el SGBD seleccionado, transformando el modelo conceptual al modelo de datos empleados por el SGBD (jerárquico, red o relacional). Este diseño persigue como objetivo principal obtener un esquema lógico eficiente en cuanto a operaciones de consulta y actualización.

Fase 5: Diseño Físico: El principal objetivo del diseño físico es conseguir una instrumentación lo más eficiente posible del esquema lógico. Para lograrlo se analizan aspectos como las características del Sistema Operativo, el Sistema Gestor de Base de Datos, la herramienta para realizar el diseño, aspectos

relacionados con el rendimiento y los requisitos de procesos así como las características del hardware, en fin, cualquier factor cercano con la computadora, para con ello lograr optimizar el consumo de recursos, minimizar el espacio de almacenamiento, proporcionar la seguridad máxima, disminuir los tiempos de respuesta y evitar las reorganizaciones.

En esta fase se especifican las estructuras de almacenamiento internas y la organización de los archivos de la base de datos y se persiguen como objetivos principales los siguientes:

- Disminuir los tiempos de respuesta.
- Minimizar el espacio de almacenamiento.
- Evitar las reorganizaciones.
- Conseguir la máxima seguridad de los datos.
- Optimizar el consumo de recursos.

Estos objetivos son fundamentales para optimizar el diseño físico y el ratio costo/beneficio.

Fase 6: Implementación del sistema de base de datos: Esta es la fase final del diseño de la base de datos, pues se implementa la base de datos al crear y compilar uno o varios esquemas y ficheros de bases de datos y definir las transacciones entre las aplicaciones.

1.4.2. Modelos de datos

Hoy en día existen diversos modelos para el diseño de bases de datos. Cada uno de ellos utiliza herramienta matemática para la descripción de la base de datos.

- **Modelo Jerárquico.**

Los modelos de datos jerárquicos se caracterizan por almacenar la información de forma jerárquica. O sea, los datos se organizan en forma similar a un árbol donde un padre tiene uno o muchos hijos denominados nodos. En este modelo el nivel más alto de árbol o nodo padre se denomina raíz y cada nodo representa un registro con sus correspondientes campos.

- **Modelo de Red.**

Este modelo se caracteriza por la representación de sus entidades ya que estas son registros o nodos y las relaciones que se establecen entre ellos se representan como enlaces o punteros. Además utiliza estructuras de datos en red, lo cual significa que un componente es capaz de vincularse con cualquier otro.

- **Modelo Relacional.**

Este modelo se caracteriza por contar con Relaciones como único objeto de tratamiento en el modelo. Se utiliza un grupo de tablas para representar los datos y las relaciones entre ellos, donde cada tabla está compuesta por varias columnas mientras cada columna posee un nombre único. Las relaciones entre datos deben ser representadas explícitamente.

- **Modelo Orientado a Objetos.**

Este modelo fue desarrollado para compensar las deficiencias del modelo relacional en lo referente a la construcción de consultas complejas y estructuras de datos sin la necesidad de dividirlas en una estructura relacional de dos dimensiones. Las bases de datos orientadas a objetos son más bien complicadas para la mayoría de los propósitos y también eran inicialmente mucho más lentas que las bases de datos relacionales.

Fundamentos de la selección

El modelo de datos que se selecciona para propuesta de solución es el modelo relacional, debido a que ofrece grandes ventajas en el diseño de la BD, ajustándose al proceso de gestión de la información de la AP, además de ser el modelo más usado en la actualidad.

1.5. Optimización de bases de datos

Cuando se va a desarrollar un sistema de bases de datos uno de los aspectos más importantes a tener en cuenta es la optimización del mismo puesto que permite mejorar su rendimiento, funcionalidades y eficiencia acorde a sus propósitos y características.

Para llevar a cabo este proceso es necesario contar con un diseño inicial lógico y físico que esté bien elaborado. El objetivo que se persigue con la optimización es minimizar el tiempo de respuesta por petición y maximizar el rendimiento del sistema de modo que se logre disminuir significativamente el tráfico en la red y el tiempo de procesamiento.

La optimización es un proceso continuo que no se debe efectuar solamente cuando el sistema esté implementado, sino sucesivamente a través del ciclo de desarrollo, para lograr el afinamiento de las bases de datos.

Existen dos tipos de afinamiento que se le pueden realizar a las bases de datos:

- Proactivo: se realiza mientras se está desarrollando el sistema.
- Reactivo: se usa para mejorar sistemas que se estén utilizando.

Optimizar no debe ser un proceso que se ejecute cuando exista un problema en el rendimiento o en los tiempos de respuesta de una base de datos pues ya sería

demasiado tarde para implementar alguna estrategia de optimización determinada. Cuando se llega a este punto se pudiera rediseñar completamente la base de datos, pero en caso de que esto sea imposible la única opción que se puede tomar para mejorar el rendimiento es reasignar más memoria y mejorar los parámetros de entrada y salida del disco.

1.6. Características de los SBGD

Estos sistemas deben tener características u objetivos que deben cumplir, entre ellos:

- Independencia de los datos y los programas de aplicación: permite lograr de cierto modo la inmunidad de las aplicaciones ante los cambios que se produzcan en la estructura de almacenamiento y en la estrategia de acceso.

- Minimización de la redundancia: permite reducir la redundancia que se producía en los archivos tradicionales, pero no la elimina del todo el objetivo que se persigue en realidad es el de eliminar la redundancia superflua.

- Integración y sincronización de las bases de datos: permite garantizar que se entreguen al sistema informativo o aplicación los datos que sean solicitados y en la forma que se soliciten independientemente de la estructura o el tipo de representación de los mismos. La sincronización permite garantizar el acceso múltiple y simultáneo a la BD.

- Integridad de los datos: permite garantizar que no exista inconsistencia entre los datos, basándose en la eliminación de la redundancia.

- Seguridad y recuperación: permite garantizar el acceso autorizado a los datos y disponer de métodos que garanticen la restauración de las bases de datos.

- Facilidad de manipulación de la información: permite que se realice una búsqueda rápida por diferentes criterios y que los usuarios planteen sus demandas de forma simple.

- Control centralizado: permite controlar de manera sistemática y única los datos que se almacenan en la BD.

1.6.1. Componentes de un SGBD

Lenguajes

- Lenguaje de definición de datos (DDL).
- Lenguaje de manipulación de datos (DML).

- Diccionario de datos: lugar donde se deposita información sobre todos los objetos que forman la base de datos (estructura lógica y física de los datos, definiciones de todos los objetos de la base de datos)

1.6.2. Clasificación de los Sistemas de Gestión de Base de Datos

Los Sistemas de Gestión de Base de Datos se clasifican según:

Modelo lógico en el que se basan:

- Modelo Jerárquico.
- Modelo de Red.
- Modelo Relacional.
- Modelo Orientado a Objetos.

Número de usuarios:

- Monousuario.
- Multiusuario.

Número de sitios:

- Centralizados.
- Distribuidos: Homogéneos, Heterogéneos.

Ámbito de aplicación:

- Propósito General.
- Propósito Específico.

1.7. Sistemas de Gestión de Bases de Datos Relacionales

Un sistema de gestión de bases de datos relacionales es aquel que sigue el modelo relacional. En 1985 el Dr. Edgar Frank Codd publicó trece reglas para evaluar si un SGBD puede considerarse un SGBDR (Sistema Gestor de Base de Datos Relacional), o dicho más concisamente, si un sistema de bases de datos puede considerarse o no relacional.

Según Codd un SGBD es relacional cuando cumple las siguientes reglas:

- Regla 0: debe ser relacional, una base de datos y un sistema de gestión.
- Regla 1: regla de la información.
- Regla 2: regla del acceso garantizado.
- Regla 3: tratamiento sistemático de valores nulos.
- Regla 4: diccionario dinámico en línea basado en el modelo relacional.
- Regla 5: regla del sublenguaje de datos completo.
- Regla 6: regla de actualización de vistas.
- Regla 7: inserción, actualización y borrado de alto nivel.
- Regla 8: independencia física de datos.
- Regla 9: independencia lógica de datos.
- Regla 10: independencia de integridad.
- Regla 11: independencia de distribución.
- Regla 12: regla de la no subversión.

Existen diversas definiciones acerca de lo que es un sistema gestor de bases de datos relacionales, entre ellas se destacan:

Un SGBD relacional es un modelo de datos que facilita a los usuarios describir los datos que serán almacenados en la base de datos junto con un grupo de operaciones para manejar los datos.

Los SGBD relacionales son una herramienta efectiva que permite a varios usuarios acceder a los datos al mismo tiempo. Brindan facilidades eficientes y un grupo de funciones con el objetivo de garantizar la confidencialidad, la calidad, la seguridad y la integridad de los datos que contienen, así como un acceso fácil y eficiente a los mismos. (EcuRed, 2012)

PostgreSQL

Es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. (Rafaelma, 2010)

Fundamentos de la Selección

Existen múltiples sistemas gestores de bases de datos tales como Microsoft Access 2010**,** ORACLE , MS SQL Server, dBaseIV y Paradox de Borland, DB2 de IBM, BASE de OpenOffice.org, etc., pero el SGBD seleccionado para este trabajo de diploma es PostgreSQL en su última serie de producción, la 9.1. Se elige al mismo debido a la estabilidad, potencia, robustez, facilidad de administración e implementación de estándares que posee convirtiéndolo en uno de los SGBD más potentes del mercado. También debido a su óptimo funcionamiento al manejar

grandes cantidades de datos, permite alta concurrencia de usuarios accediendo a la vez al sistema y el equipo de trabajo posee experiencia trabajando con este sistema.

1.8. Metodología a emplear para el desarrollo de la solución

Al desarrollar una aplicación informática de cualquier tipo es más que necesario que el proceso de desarrollo esté orientado por una metodología de desarrollo de software. Las metodologías desarrollan un proceso detallado con un fuerte énfasis en planificar que se basa en otras disciplinas de la ingeniería.

Estas se dividen en dos tipos fundamentalmente: robustas o ágiles. Las metodologías robustas o pesadas se emplean básicamente para guiar el proceso de desarrollo de software de grandes dimensiones, cuando un proyecto requiere de gran cantidad de documentación, va a ser realizado en un tiempo considerablemente largo y existe la posibilidad de que pase por las manos de varios equipos de trabajo.

Las metodologías ágiles a diferencia de las robustas trabajan enfocándose en los clientes y los resultados. Estas están basadas en la promoción de iteraciones en el desarrollo del software a lo largo de todo el ciclo de vida del proyecto, logrando que se minimicen los riesgos desarrollando software en cortos períodos de tiempo.

1.8.1. Metodologías Robustas: Proceso Unificado de Desarrollo (RUP)

RUP es un proceso formal que provee un acercamiento disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que satisfaga los requerimientos de los usuarios finales, respetando cronograma y presupuesto. Fue desarrollado por Rational Software, y está integrado con toda la suite de herramientas Rational. Puede ser adaptado y extendido para satisfacer las necesidades de la organización

que lo adopte. Es guiado por casos de uso y centrado en la arquitectura, iterativo e incremental y utiliza UML como lenguaje de notación. (Figueroa, y otros, 2011)

Consta de 4 fases principales:

- Inicio: el objetivo en esta etapa es determinar la visión del proyecto.
- Elaboración: en esta etapa el objetivo es determinar la arquitectura óptima.
- Construcción: en esta etapa el objetivo es llevar a obtener la capacidad operacional inicial.
- Transición: el objetivo es llegar a obtener el despliegue del proyecto.

1.8.2. Metodologías Ágiles: SXP

SXP está compuesta por las metodologías SCRUM y XP, ofreciendo una estrategia tecnológica, a partir de la introducción de procedimientos ágiles que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva. Esta metodología fomenta el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo.

SXP consta de 4 fases principales:

- Planificación-Definición: donde se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.
- Desarrollo: es donde se realiza la implementación del sistema hasta que esté listo para ser entregado;
- Entrega: es la puesta en marcha; y por último.
- Mantenimiento: es la fase donde se realiza el soporte para el cliente.

SXP está especialmente indicada para proyectos con pequeños equipos de trabajo, un constante cambio de requisitos o requisitos imprecisos, donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad. Fomenta el trabajo en equipo, con un objetivo claro, permitiendo el seguimiento y

control de las tareas a realizar. (SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE, 2010)

Fundamentos de la selección

No existe una metodología universal que indique cómo crear todo tipo de software. Cuando un equipo decide construir un producto, se debe escoger la metodología a utilizar teniendo en cuenta las características, complejidad, envergadura del proyecto y el tipo de contrato establecido para el mismo. (SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE, 2010).

Una vez analizadas las características del software a desarrollar y las ventajas que proporciona el uso de las metodologías ágiles se procede a seleccionar a SXP como la metodología de desarrollo que guiará el proceso de desarrollo de software del presente trabajo de diploma.

1.9. Herramientas a emplear para el desarrollo de la solución

1.9.1. Herramientas CASE

Las herramientas CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador) propician un conjunto de métodos y técnicas automatizadas que brindan ayuda y dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todo el ciclo de vida del desarrollo de un software, reduciendo el esfuerzo, el costo y el tiempo. Dichas herramientas se encuentran en una continua evolución, por lo que existe una gran variedad de proveedores y productos, cada uno de ellos con diferentes aplicaciones y especificaciones. (Herramientas Case, 2011) Algunas de estas herramientas son las siguientes:

Visual Paradigm

Es una herramienta CASE que utiliza el lenguaje modelado unificado UML (Unified Modeling Language), que soporta el ciclo de vida completo del desarrollo de software. Esta herramienta multiplataforma ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Brinda excelentes facilidades de interoperabilidad con otras aplicaciones, compatibilidad entre versiones, código inverso, así como construir diagramas UML como son:

Diagramas de Casos de Uso.
Diagramas de Clases.
Diagramas de Secuencia.
Diagramas de Comunicación.
Diagramas de Estado.
Diagramas de Componentes.
Diagramas de Despliegue.
Diagramas de Objetos.
Diagramas de Interacción.

Visual Paradigm tiene licencia tanto libre como comercial, es fácil de instalar y contiene una serie de facilidades que se mencionan a continuación:

- Soporta un conjunto de estándares entre los que se encuentran UML, SysML, BPMN, XML y XMI.
- Soporte de modelado UML, modelado de procesos de negocios y un generador de mapeo de objetos-relacionales para los lenguajes de programación Java, .NET y PHP.
- Integración con herramientas Java (Eclipse/IBM WebSphere, JBuilder, NetBeans IDE, Oracle JDeveloper y BEA Weblogic).
- Permite la generación de código y la ingeniería inversa para un conjunto de lenguajes entre los que se encuentran Java, C++, CORBA IDL, PHP, XML Schema, Ada y Python. Cualquiera de los cambios en el código existente puede reflejarse en el modelo y viceversa. (Visual Paradigm, 2012)

Power Architect

Es una aplicación para generar diagramas entidad-relación que permite generar los scripts de creación de tablas, procedimientos, etc. Proceso que es más conocido como Forward Engineering para los modelos de datos. También permite recuperar dichos modelos de una base de datos existente, proceso también conocido como ingeniería inversa (Reverse Engineering), entre otras posibilidades que brinda tales como: analizar estructuras de datos entre diversos modelos de datos para ver las diferencias y similitudes, creación de perfiles, etc. Todas estas características definen a Power Architect como una herramienta perfecta para administradores de bases de datos, analistas y diseñadores.

Fundamentos de la selección

Aunque existe una gran variedad de herramientas CASE tales como SQL Power Architect, Visual Paradigm, etc., se toma en cuenta que la facultad se rige por las políticas de software libre y por ello se selecciona como herramienta CASE al Power Architect. Esta herramienta a pesar de no ser libre, cuenta con una licencia comercial que posee la universidad. Con ella, el diseñador podrá abrir múltiples conexiones concurrentes a bases de datos, crear y explorar perfiles de datos fuente, arrastrar y soltar esquemas de datos, tablas y columnas dentro del modelo de datos, y confeccionar la base de datos resultante con su plantilla ETL asociada. Hasta el diseño más complicado, la base de datos más grande o el modelo más inabarcable, podrá ser gestionado y manejado con Power Architect.

1.9.2. Herramientas utilizadas al diseñar el acceso a los datos.

Hibernate

Es una herramienta para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones.

Hibernate es una herramienta ORM completa que ha conseguido en un tiempo record una excelente reputación en la comunidad de desarrollo posicionándose claramente como el producto OpenSource líder en este campo gracias a sus prestaciones, buena documentación y estabilidad. Es valorado por muchos incluso como solución superior a productos comerciales dentro de su enfoque, siendo una muestra clara de su reputación y soporte. (Herrera, 2007)

Características Principales:

- En las consultas se pueden definir los campos que se deseen levantar de un objeto, no necesariamente todos.
- Es posible declarar si se desea obtener los objetos relacionados al que estamos obteniendo o no.
- Permite especificar tamaño o límites de los objetos a levantar.
- Se puede hacer uso de la llamada cache de segundo nivel y la cache de consultas.
- Es posible declarar consultas y después usarlas.
- Al momento de hacer pruebas de rendimiento o performance las caches fueron muy útiles, los tiempos fueron reducidos en un 40 o 50 % pero esto es muy discutible, tiene pros y contras hay que probarlo con la base de producción y con las consultas de producción, tiene ventajas pero también tiene desventajas.

Fundamentos de la selección

Existen otras herramientas que permiten diseñar e implementar el acceso a los datos como Doctrine, Propel, etc., pero la herramienta seleccionada para el desarrollo de este trabajo de diploma es Hibernate debido a que la aplicación para la que se va a implementar la capa de acceso a datos estará desarrollada en el lenguaje Java. También se elige este ORM debido a las múltiples ventajas que ofrece, pero fundamentalmente porque es muy cómodo y hace más ágil el

desarrollo, e incluso hay herramientas que ya generan código para Hibernate como AndroMDA y los archivos XML de mapping a partir del modelo xsi.

Conclusiones del capítulo

En el presente capítulo se abordaron diferentes conceptos asociados al dominio del problema que tributan al desarrollo de la investigación, como son las bases de datos, los sistemas gestores de bases de datos, etc. Se realizó un estudio de las fases del diseño de bases de datos que existen actualmente en el mundo de modo que se pudiera lograr un diseño lo más optimizado posible, además de las herramientas y metodología que son utilizadas para un mejor desarrollo de la misma. Como resultado del estudio realizado se obtuvieron los conocimientos teóricos necesarios para continuar con la investigación y se le dio cumplimiento al primer objetivo trazado en este trabajo de diploma.

Capítulo 2. Diseño de Bases de Datos Relacionales

Introducción

En el presente capítulo se describen los patrones de diseño y metodologías a utilizar en el diseño de la BD. Se exponen los procesos principales que tienen lugar en la optimización de la BD, entre ellos se encuentran: la creación de varios esquemas, propuesta de integración de nomencladores comunes, normalización, etc. Se ofrece una descripción detallada de las entidades para el esquema de administración de la base de datos SIBOB y se muestra el modelo físico, así como también Modelo Entidad Relación que compone dicha BD.

2.1. Transformación de un esquema único en varios

El número de tablas con las que inicialmente contaba la base de datos del Sistema Informativo de la Administración Provincial (SIGOB) era una cifra significativa donde se podría prever un difícil y engorroso trabajo con respecto a la organización, seguridad, integridad, confidencialidad y gestión de los permisos sobre cada una de las tablas en el esquema único. Debido al riesgo evidente, se propone convertir el esquema único de la base de datos en varios esquemas, donde los mismos estarán definidos por la lógica del negocio.

De esta forma se podrán obtener grandes ventajas tales como:

- Posibilitar un mayor uso de las base de datos por múltiples usuarios.
- Evitar que las entidades estén ubicadas de forma desorganizada en el esquema único.
- Facilitar la definición de propietarios de esquemas.
- Tener tablas con nombre idénticos en esquemas diferentes evitando así el empleo de prefijos o sufijos para diferenciar las tablas.

El esquema único de la base de datos se Transformará en 25 esquemas independientes.

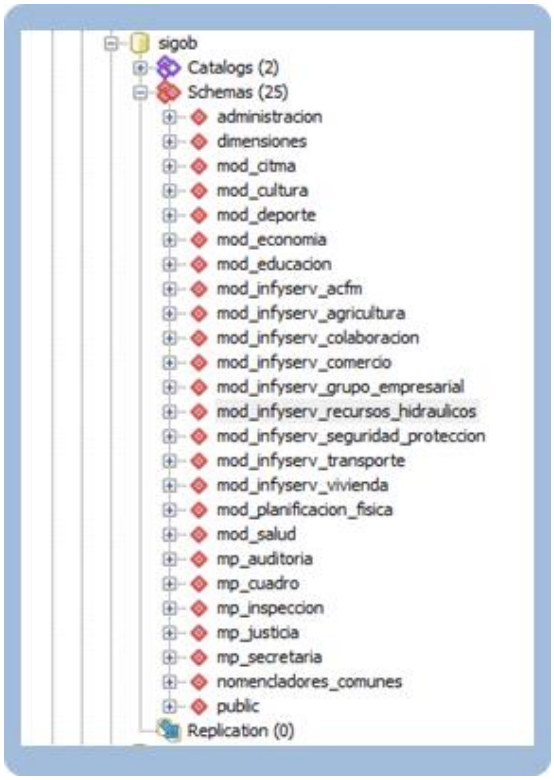

Fig5. Esquemas de la base de datos del SIGOB

2.2. Consideraciones generales de nomenclatura de la BD del SIGOB.

- Los nombres siempre comenzarán con una letra minúscula, nunca con un número o carácter especial.
- El uso de números, acrónimos y abreviaciones serán los mínimos e indispensables, siempre teniendo en cuenta los nombres reales del negocio.
- Eliminar el uso de tildes y espacios entre palabras.
- Se sustituirá la letra "**ñ**" por "**nn**".

Una vez vistas las diferentes consideraciones, es importante definir la nomenclatura que se empleará para las **tablas**. A continuación se mostrará un ejemplo de nombres de tablas teniendo en cuenta las consideraciones expuestas previamente.

Ejemplo:

- dpersona.
- dusuario.
- dpublicaciones.

En cuento a las tablas **nomencladoras** se denotarán de la siguiente manera:
¨ n + nombre de la tabla nomencladora ¨**.**

Ejemplo:

- nrol.
- npublicacion.
- ndireccion.

2.3. Propuestas de integración de los nomencladores comunes.

Durante el estudio del negocio, los requisitos y la documentación existente relacionada con la base de datos del SIGOB, uno de los aspectos más importantes encontrados una vez construidos los 25 esquemas independientes, fue el uso repetitivo de tablas nomencladoras en cada uno de los esquemas, trayendo consigo el aumento de entidades para la BD, la duplicación de código, desorganización, etc.

Para la solución de dicho problema se hace necesaria la creación de un esquema independiente denominado “nomencladores_comunes” el cual resuelve dicha problemática, ya que el mismo contiene todas las tablas nomencladoras comunes.

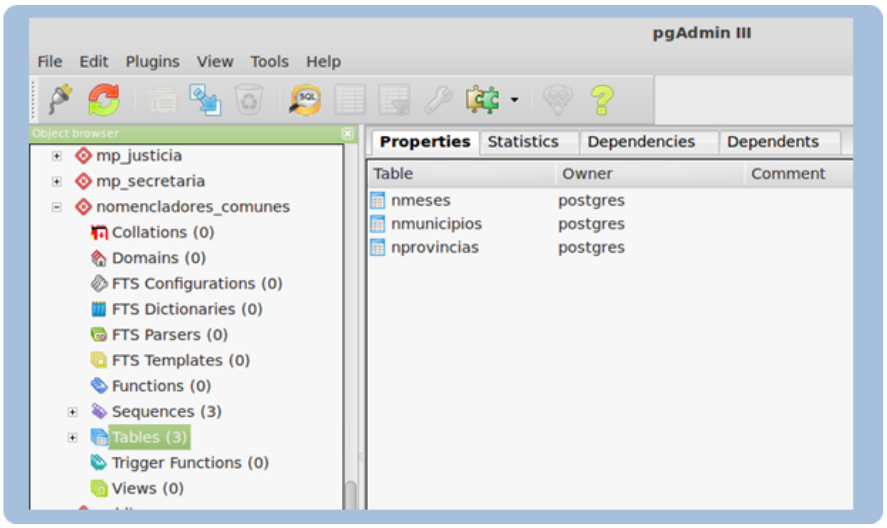

Fig6. Esquema nomencladores_comunes

2.4. Metodología para el diseño de bases de datos

La metodología usada para el diseño de la Base de Datos está determinada por los siguientes puntos:

- Determinación de entidades y atributos.
- Determinación de relaciones.
- Normalización de entidades.
- Obtención del modelo lógico global de los datos.
- Diseño físico de la BD.

2.5. Patrones de Diseño a utilizar

Los patrones constituyen para el diseño de una base de datos una solución estándar para problemas que existen en muchos niveles de abstracción y abarcan las distintas etapas de la implementación. Concluyendo, un patrón de diseño es un grupo de reglas que describen como afrontar diversas tareas para dar solución a problemas que surjan durante el desarrollo de software.

Los Patrones de Diseño ayudan a conseguir diseños optimizados de manera ágil y con poco esfuerzo al proponer buenas prácticas con soluciones pensadas y probadas. Estos aportan ideas reutilizables y adaptables a un amplio campo de problemas. En la actualidad se cuenta con algunos patrones de diseño de base de datos que le sirve de mucha ayuda a los diseñadores para realizar el diseño de una forma óptima. (Navathe, y otros, 2010)

Se utilizaron varios patrones de diseño dentro de los que se encuentran los patrones descritos por Kyle Brown y Bruce G. Whitenack, a continuación se describe la utilización de los mismos.

Representación de objetos como tablas

Problema: ¿Cómo representar un objeto en un esquema de base de datos relacionales?

Solución: Definir una tabla para cada clase de objetos persistentes. Los atributos de la clase que son tipos primitivos serán las columnas de las tablas.

Representación de relaciones como tablas

Problema: ¿Cómo representar una relación en un esquema de base de datos relacionales?

Solución:

Para las relaciones de uno a uno o uno a muchos:

1. Colocar una clave ajena en la tabla de cardinalidad uno, para representar la relación de los objetos.
2. O crear una tabla asociativa para registrar los identificadores de cada uno de los objetos de la relación.

Para las relaciones muchos a muchos:

Crear una tabla asociativa para registrar los identificadores de cada uno de los objetos de la relación.

Identificador de objetos

Problema: ¿Cómo mantener la identidad de un objeto en una base de datos relacional? Cada identidad de objetos individuales debe ser presentada en la base de datos.

Solución: Asignar un identificador independiente (OID) a cada objeto persistente. Se recomienda el uso de un generador de secuencias si hay alguno disponible en la base de datos ya que los identificadores son generalmente enteros largos que garantizan que sean únicos para una clase de objetos en particular.

Referencia de llaves foráneas

Problema: ¿Cómo representar objetos que referencian otros objetos que no son de tipos de datos base?

Solución: Asignar a cada objetos un identificador único. Luego añadir una columna por cada variable de instancia que no tenga un tipo de dato base o sea una colección. En esa columna almacenar el identificador del objeto referenciado y declarar la columna como llave foránea.

Representar una herencia en una base de datos relacional

Problema: ¿Cómo representar una jerarquía de herencia en una base de datos relacional?

Solución: Crear una tabla para cada clase en la herencia que tenga atributos. Cada atributo de la clase será una columna de la tabla, añadir además una columna adicional que represente la llave común entre todas las clases de la herencia.

Claves subrogadas

Estas claves no son más que un identificador único que se asigna a cada registro de una tabla. Son de tipo numérico secuencial sin significado especial y debe ser el único campo que sea clave principal de cada tabla.

2.6. Normalización

Al realizar el diseño de una base de datos relacional, es necesario que las tablas que la componen atraviesen un proceso de normalización que posibilite eliminar la redundancia de los datos, evitar problemas de inserción, eliminación y actualización de los datos, o sea, optimizar el trabajo con la información almacenada. No solo se debe considerar el grado de normalización para determinar si un esquema relacional es eficiente sino que se debe ir ejecutando un proceso que compruebe que cada relación (tabla) cumple una serie de reglas que se basan en la clave primaria y las dependencias funcionales. Cada regla que se cumple aumenta el grado de normalización. Si una regla no se cumple, la relación se debe descomponer en varias relaciones que sí la cumplan.

Existen varios niveles en el proceso de normalización. Cada nivel cuenta con reglas de normalización que se relacionan entre ellas de forma dependiente, lo que indica que para que una base de datos se encuentre en una determinada forma normal, primeramente tiene que cumplir con las reglas de los niveles inferiores de normalización.

Fig7. Formas Normales

Primera Forma Normal: Un esquema de relación está en primera forma normal (1FN) si, y sólo si, los dominios de todos los atributos de la relación son atómicos. Un dominio es atómico si se considera que los elementos del dominio son unidades indivisibles. La primera formal normal se definió para prohibir los atributos multivalorados, los atributos compuestos y sus combinaciones.

Segunda Forma Normal: Un esquema de relación está en segunda forma normal (2FN) si, y sólo si, está en primera forma normal (1FN) y, además cada atributo del esquema de relación que no está en la clave primaria depende funcionalmente de la clave primaria completa y no sólo de una parte de esta. Las dependencias funcionales son restricciones que se aplican sobre el conjunto de relaciones legales de un modelo relacional (una relación es legal si satisface las restricciones impuestas).

Tercera Forma Normal: Un esquema de relación está en tercera forma normal (3FN) si, y sólo si, está en segunda forma normal (2FN) y, además cada atributo del esquema de relación que no está en la clave primaria sólo depende funcionalmente de la clave primaria, y no de ningún otro atributo.

Forma Normal de Boyce-Codd (FNBC): Un esquema de relación está en forma normal de Boyce-Codd (FNBC) si, y sólo si, está en tercera forma normal (3FN) y,

además cada atributo del esquema de relación que determine otros atributos está en una superclave.

Cuarta Forma Normal (4FN): debe estar en *FNBC* y que además existan tres o más atributos formando parte de la llave primaria. Debe cumplir que no existan dos o más atributos independientes con dependencia respecto a un conjunto de atributos, formando parte de la llave junto a dicho conjunto, es decir, no dependencia entre atributos llaves.

Quinta Forma Normal (5FN): cumple que toda dependencia de agregación (*join dependency*) es implicada por las llaves candidatas. Diseñada para reducir la redundancia en relaciones que almacenan hechos multi-evaluados a través del aislamiento de relaciones semánticamente relacionadas. Una relación está en 5FN cuando su contenido no puede ser reconstruido a partir de un conjunto de tablas, se excluye la posibilidad de tablas con los mismos atributos llaves.

2.7. Requisitos funcionales

Un requisito funcional define el comportamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica.

El núcleo del requisito es la descripción del comportamiento requerido, que debe ser clara y concisa. Este comportamiento puede provenir de reglas organizacionales o del negocio, o ser descubiertas por interacción con usuarios, inversores y otros expertos en la organización.

Una vez reunidos con el cliente y analizada la lógica del negocio se obtienen múltiples requisitos funcionales para la base de datos del Sistema Informativo de la Administración Provincial de Artemisa. A continuación se muestran algunos de los requisitos funcionales para el esquema de administración.

RF1- Añadir Usuario.

RF2- Modificar Usuario.

RF3- Eliminar Usuario.

RF4- Listar Usuarios a Modificar y Eliminar.

RF5- Buscar Usuarios.

RF6- Añadir Publicación.

RF7- Modificar Publicación.

RF8- Eliminar Publicación.

RF9- Listar publicaciones a Modificar y Eliminar.

RF10- Buscar Publicaciones.

2.8. Modelos de datos

2.8.1. Modelo-Entidad-Relación

El Modelo-Entidad-Relación (MER), es un tipo de diagrama para el modelado de bases de datos. Su objetivo es representar relaciones que existen en la vida real entendiendo su semántica. Los cuatro elementos fundamentales de un MER son: las entidades, los atributos, las interrelaciones y el dominio.

La base de datos propuesta para el SIGOB cuenta con 25 modelos, de ellos 23 son para el trabajo con las direcciones y departamentos, uno para agrupar los nomencladores comunes y el otro sería para la administración del sistema. A

continuación se mostrará el MER del esquema de Administración del SIGOB, los demás MER se encuentran en el Modelo Canónico de datos de cada una de las direcciones.

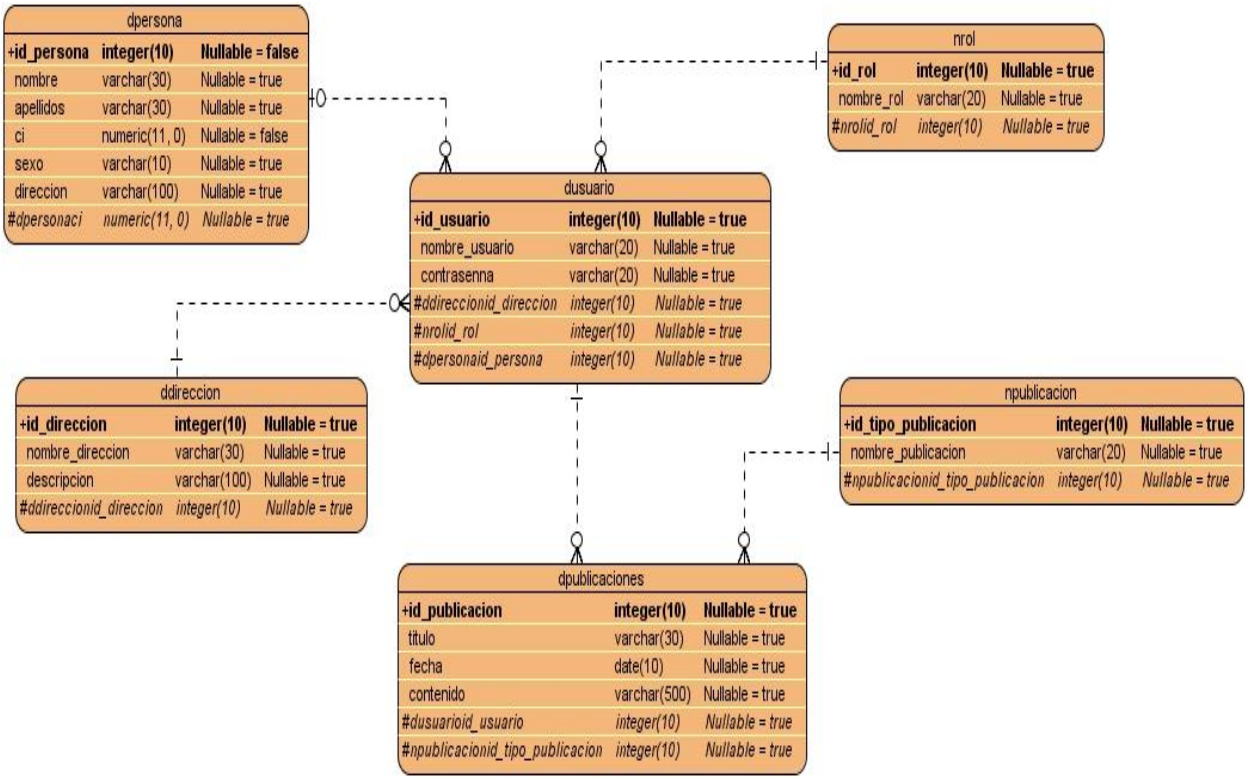

Fig8. MER del módulo de Administración.

Descripción de las Entidades

A continuación se mostraran la descripción de las entidades del módulo de Administración.

Entidad:	npublicacion
Descripción:	Entidad nomencladora que contiene el tipo de publicación que se hará en el sistema.

Relaciones:	dpublicaciones		
Campos	**Tipo de Dato**	**Tamaño**	**Descripción**
id_tipo_publicacion	integer	10	Identificador del tipo de publicación
nombre_publicacion	varchar	20	Nombre del tipo de publicación

Fig9. Descripción de la entidad nomencladora de las publicaciones

Entidad:	dusuario		
Descripción:	Esta entidad contiene toda la información relacionada con los usuarios del sistema		
Relaciones:	dpersona, ddireccion, dpublicaciones, nrol		
Campos	**Tipo de Dato**	**Tamaño**	**Descripción**
id_usuario	integer	10	Identificador del usuario
ci	numeric	11	Id que representa a cada una de las persona
id_direccion	integer	10	Id que representa a cada una de las direcciones del CAP
id_rol	Integer	10	Id que representa el rol de los usuarios
nombre_usuario	varchar	20	Nombre del usuario

contrasenna	varchar	500	Contraseña del usuario

Fig10. Descripción de la entidad usuario

Entidad:	dpublicacion		
Descripción:	Esta entidad contiene toda la información referente a las publicaciones que se hagan en el sistema		
Relaciones:	dusuario, npublicacion		
Campos	**Tipo de Dato**	**Tamaño**	**Descripción**
id_publicacion	integer	10	Identificador de las publicaciones
id_tipo_publicacion	integer	10	Id que representa a los tipos de publicaciones que existen
titulo	varchar	30	Título de la publicación
contenido	varchar	500	Contenido de la publicación
fecha	date	10	Fecha en que se hizo la publicación

Fig11. Descripción de la entidad publicación

Entidad:	ddireccion
Descripción:	Esta entidad contiene toda la información referente a las direcciones a las que pertenecen los usuarios del sistema
Relaciones:	dusuario

Campos	Tipo de Dato	Tamaño	Descripción
id_direccion	integer	10	Identificador de las direcciones
nombre_direccion	varchar	30	Nombre de la dirección
descripcion	varchar	100	breve descripción de la dirección

Fig12. Descripción de la entidad dirección.

Entidad:	nrol		
Descripción:	Entidad nomencladora que contiene los nombre de los roles asociados con los usuarios		
Relaciones:	dusuario		
Campos	**Tipo de Dato**	**Tamaño**	**Descripción**
id_rol	integer	10	Identificador del rol
nombre_rol	varchar	20	Nombre del rol del usuario

Fig13. Descripción de la entidad nomencladora de los roles.

Entidad:	dpersona
Descripción:	Esta entidad contiene toda la información referente a las personas

Relaciones:	dusuario		
Campos	**Tipo de Dato**	**Tamaño**	**Descripción**
Id_persona	integer	10	Identificador de la persona.
ci	numeric	11	Carnet identidad .de la persona
nombre	varchar	30	Nombre de la persona
apellidos	varchar	30	Apellidos de la persona
sexo	varchar	10	Sexo de la persona
direccion	varchar	100	dirección particular de la persona

Fig14. Descripción de la entidad personas.

2.8.2. Modelo Físico

El paso de un modelo lógico a uno físico requiere un profundo entendimiento del manejador de bases de datos que se desea emplear, incluyendo características como:

- Conocimiento a fondo de los tipos de objetos (elementos) soportados.
- Detalles acerca del indexamiento, integridad referencial, restricciones, tipos de datos, etc.
- Detalles y variaciones de las versiones
- Parámetros de configuración
- Data Definition Language (DDL)

Los modelos físicos de datos se usan para describir datos en el nivel más bajo. Algunos autores definen estos modelos como "modelos de datos primitivos", el mismo no es más que la representación mediante tablas y relaciones de la Base de Datos. Con una buena confección del mismo, se obtendrán buenos resultados.

La Base de Datos del SIGOB cuenta con 25 modelos físicos, cada uno de estos modelos está vinculado directamente con cada una de las direcciones y departamentos de la Administración Provincial, y estos a su vez están ubicados en distintos esquemas de la BD con el objetivo de agilizar el proceso de acceso a datos y la seguridad de los mismos.

A continuación se mostrará el modelo físico del módulo de administración, los demás modelos los podrán encontrar en cada uno de los Modelos Canónico de las direcciones.

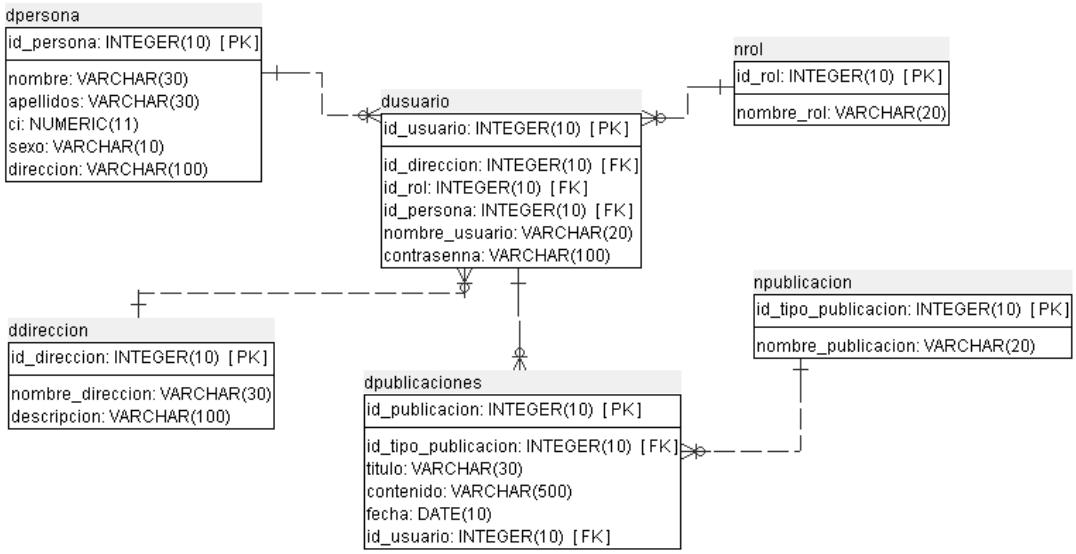

Fig15. Modelo físico del módulo de administración

Conclusiones del capítulo

Al concluir este capítulo se logró obtener un diseño para la BD del sistema Informativo del Gobierno de Artemisa que cumple con las necesidades del mismo. Con el uso de la misma se logra reducir significativamente problemas como la redundancia de datos, evitando inconvenientes en las consultas realizadas para extraer e insertar datos en el sistema, además la utilización de nomencladores comunes logra mejorar considerablemente las operaciones con los datos en cada uno de los esquemas que componen la BD.

También se realizó una descripción detallada de la solución desarrollada. Se describieron las principales tablas del modelo lógico con los patrones de diseño empleados para realizarlas, entre otros aspectos como la optimización de la base de datos que contribuyeron significativamente a que la base de datos cumpliera con todos los requisitos del sistema.

Capítulo 3. Implementación de la Capa de Acceso a Datos y Validación del diseño realizado

Introducción

En este capítulo se abordan aspectos relacionados con la implementación de la capa de acceso a datos mostrando el uso de Hibernate para la conexión de la aplicación al esquema de administración en la BD del SIGOB. Se realiza la validación teórica y funcional del diseño de la base de datos teniendo en cuenta aspectos como la integridad, la seguridad, el análisis de la redundancia de la información y cómo responde el sistema a las pruebas que se le realizan.

3.1. Persistencia de Datos

Actualmente persistir en una BD relacional objetos java es bastante sencillo, ya que existen diversas herramientas que permiten a los implementadores manejar motores de persistencia para convertir objetos Java a columnas/registros de una base de datos y viceversa. Los objetos java son serializados y estructurados en forma de árbol a una base de ratos relacional.

Esencial para este esfuerzo es la necesidad de mapear estos objetos a columnas y registros de la base de datos de una manera optimizada en velocidad y eficiencia. La herramienta Hibernate ha conseguido en un tiempo record una excelente reputación en la comunidad de desarrollo puesto que se enfrenta al problema "Objeto Java a BD" de forma tan eficiente, persistiendo y restaurando viejos objetos Java (POJO) utilizando un modelo de programación muy transparente.

Hibernate en su trabajo con java brinda mecanismos de mapeo objeto/relacional para definir cómo se almacenan, eliminan y actualizan los objetos Java, ofreciendo la recuperación que pueden optimizar los esfuerzos de desarrollo a través de servicios de consulta.

3.2. Implementación de las Entidades

Para la generación de los archivos POJO existen en el IDE “NetBeans” asistentes que permiten crear las clases POJO con solo configurar la conexión a la base de datos.

Para realizar el mapeo de las clases primeramente es necesario realizar la configuración del archivo <hibernate.cgf.xml> el cual define la información sobre la conexión a la BD.

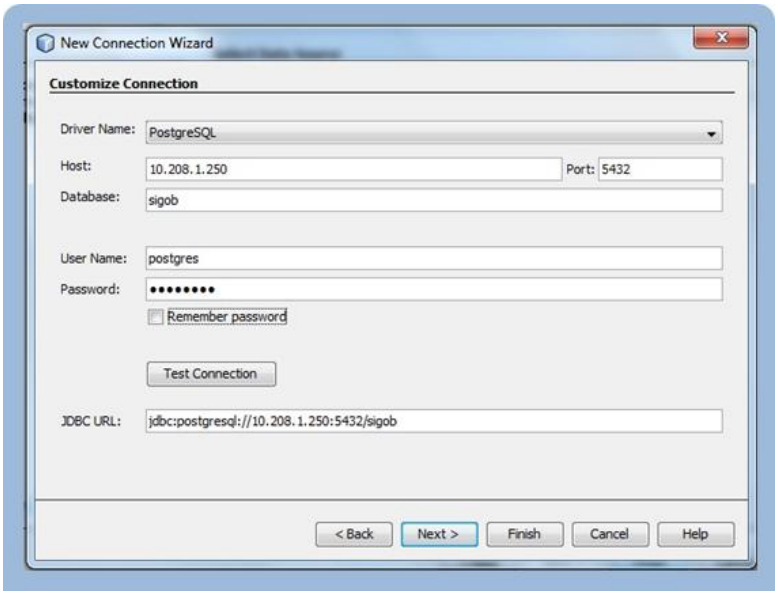

Fig16 Nueva Conexión a la BD

Una vez terminado el proceso de conexión a la BD, se procede a elegir el esquema con el que se va a trabajar y seguidamente se genera el código de la configuración típica del hibernate.cgf.xml.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN" "http://hibernate.
<hibernate-configuration>
  <session-factory>
    <property name="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</property>
    <property name="hibernate.connection.driver_class">org.postgresql.Driver</property>
    <property name="hibernate.connection.url">jdbc:postgresql://10.208.1.250:5432/sigob</property>
    <property name="hibernate.connection.username">postgres</property>
    <property name="hibernate.connection.password">sigobArt</property>
    <property name="hibernate.show_sql">true</property>
    <property name="hibernate.current_session_context_class">thread</property>
    <property name="hibernate.query.factory_class">org.hibernate.hql.ast.ASTQueryTranslatorFactory</property>
  </session-factory>
</hibernate-configuration>
```

Fig17 Código XML de la Configuración de Hibernate

Luego se realiza la generación del archivo de ingeniería inversa <hibernate.reveng.xml> donde se encuentran los nombres de las tablas de la BD que se van a mapear.

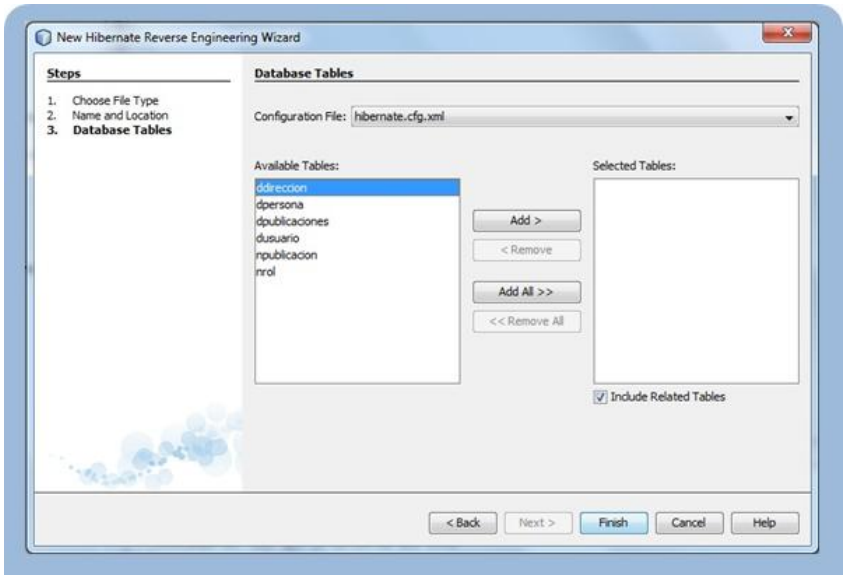

Fig18 Selección de las entidades a mapear.

Para concluir se lleva a cabo el mapeo, donde las tablas de la BD se trabajaran como objetos en Java, para esto se procede a crear los POJO (Plain Old Java Object) de la BD.

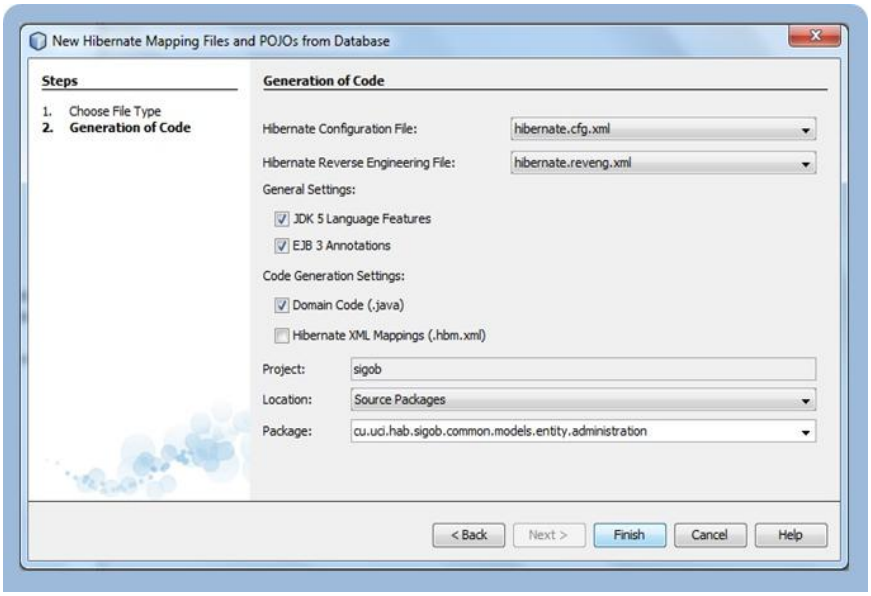

Fig19 Mapeo de las entidades.

En este proyecto del SIGOB se decidió realizar el mapeo de las entidades a través de notaciones donde los atributos de las mismas por medio de metadatos tienen su equivalente en la BD, permitiendo reducir la cantidad de líneas de código, y además la cantidad de archivos necesarios para el mapeo, lo que simplifica tanto el esfuerzo en codificar, mantener, como la probabilidad de cometer errores.

A continuación se muestra la estructura que siguen las clases entidades cuando son mapeadas:

```
@Entity
@Table (name="Nombre de la Tabla"
, schema="Nombre del Esquema"
)

Public class Nombre de la Clase implements java.io.serializable {

/* Atributos de la clase */
private long id;

public Constructor() {
}

/* Métodos Get y Set de cada uno de los atributos junto los metadatos
correspondiente */
}
```

Fig20 Estructura de las entidades mapeada

3.3. Utilización de la clase abstracta "AbstractEntity"

Esta entidad abstracta empleada en la implementación del proyecto del SIGOB tiene como objetivo fundamental proporcionarle a las clases mapeadas que la hereden un atributo identificador auto-incrementable, a continuación en la Figura 16 se muestra la estructura de esta entidad abstracta.

```
Public abstract class AbstractEntity implements Serializable {
   private static final long
   serialVersionUID=5143168877262662951L;
   protected Long id;

// Este método es abstracto porque las anotaciones no son heredables//
public abstract Long getId();

/*Este método es protegido para evitar que un programador pueda poner un identificador en la
instancia, ya que los identificadores deben ser gestionados por la capa de persistencia*/
protected void setId(final Long id){

this.id=id;
   }
}
```

Fig21 Estructura de la clase "AbstractEntity"

Las entidades mapeadas que hereden de la clase AbstractEntity tienen obligatoriamente que redefinir la función **Long getId()** de la clase padre, trayendo consigo cambios para la entidad tales como la eliminación del atributo id que se generó automáticamente cuando se realiza el mapeo de datos con Hibernate, ya que este lo brindará la clase padre "AbstractEntity". Luego se modifican los constructores y por último se definen tres notaciones para la propiedad del atributo id (**Long getId()**), donde se definirá que este atributo será auto-incrementable, tomando para esto el nombre del campo en la BD y el valor de la secuencia en la misma.

```
@Id
@SequenceGenerator(name = "generador", sequenceName = "administracion.dpublicaciones_id_publicacion_seq")
@GeneratedValue(generator = "generador")
@Column(name = "id_usuario", unique = true, nullable = false)
@Override
public Long getId() {
    return this.id;
}
```

Fig22 Función getId () de la entidades dusuario.

3.4. Utilización del DAO Genérico.

Con el fin de tener independencia de la capa de datos para gestionar las entidades en ambas direcciones se implementa el patrón DAO (Objeto de Acceso a Datos) que encapsulará el acceso a la BD. En la aplicación se implementó la clase genérica "GenericDao" con el objetivo de gestionar todas las entidades, suministrando una interfaz entre la aplicación y la BD. Esta clase genérica hereda de la clase HibernateDaoSupport implementada en el framework Spring que ofrece la ventaja de trabajar con las secciones de Hibernate de forma automática y sin ningún tipo de gestión por parte del administrador de BD.

En la clase GenericDao fueron creadas las funciones necesarias para trabajar de forma general con todas las entidades. A continuación se explican algunas de las principales funciones.

- **save**

```
Public void save (AbstractEntity entity){
getHibernateTemplate().save(entity);
}
```

Salva el objeto pasado por parámetro.

- **delete**

```
Public void delete(AbstractEntity entity){
getHibernateTemplate().delete(entity);
}
```

Elimina un objeto pasado por parámetro.

➢ **saveOrUpdate**

```
Public void saveOrUpdate(AbstractEntity  entity){
getHibernateTemplate().saveOrUpdate(entity);
}
```

Este método recibe como parámetro un objeto, verifica que el objeto este almacenado en la base de datos, si es encontrado lo actualiza en caso contrario es salvado.

➢ **find**

```
Public <T> List<T> find(Class<T> entity){
return getHibernateTemplate().loadAll(entity);
}
```

Retorna una lista de todas las tuplas de la base de datos mapeadas como objetos, se le pasa como parámetro el tipo de dato que se desea buscar.

➢ **findByPK**

```
Public <T> object findByPK(Class<T> entityName, serializable id){
return (object) getHibernateTemplate().get(entityName,  id);
}
```

Busca y retorna un objeto a través de la llave primaria y el tipo.

➢ **deleteAll**

```
Public <T> void deleteAll(Collection<T>  collection){
getHibernateTemplate().deleteAll(collection);
}
```

Elimina todas las tuplas de una tabla recibiendo como parámetro el tipo de esta.

3.5. Validación teórica del diseño realizado.

Es de suma importancia obtener un correcto diseño de la BD ya que el mismo es un punto significativo en el desarrollo de un sistema de gestión, por lo que se deben

tener presentes diversos aspectos que garanticen un eficiente diseño. Estos aspectos que se van a tratar en este acápite se relacionan con la integridad, privacidad, confidencialidad, la redundancia de la información y la seguridad de los datos, garantizado de esta manera el acceso autorizado a los datos y que la información no se altere por operaciones inconsistentes.

3.5.1. Integridad de los datos

La integridad en una BD trata acerca de la corrección y exactitud de la información contenida. Cualquier BD puede contar con un conjunto de restricciones de integridad de múltiples complejidades. La integridad de los datos hace alusión a la validez, corrección, consistencia y completitud de los datos almacenados en la BD, que tiene como función proteger la BD contra operaciones que introduzcan inconsistencias en los datos, además puede verse afectada de diferentes formas cuando se realizan acciones como la inserción de datos inválidos o incorrectos, la modificación de los datos con valores incorrectos y la eliminación de datos que produzcan la violación de alguna restricción en la BD.
La integridad de los datos se agrupa en diferentes categorías como son:

- **Integridad de entidad:** La clave primaria de las entidades no puede aceptar valores nulos y siempre deberá ser única, debido a que la llave primaria es un identificador irreducible que se utiliza para identificar de modo único las tuplas.

- **Integridad de clave:** La clave primaria de una entidad no puede tomar valores iguales en tuplas diferentes, es decir las llaves primarias no pueden repetirse. En la BD propuesta para el SIGOB las claves primarias son de tipo numeric, generándose incrementalmente garantizando así que no se cometan errores, uniformidad en la creación de las llaves, etc.

- **Integridad de Dominio:** La integridad de dominio es definida por el conjunto de valores que son posibles para un atributo determinado. Se pueden restringir los

valores de un atributo mediante tipos de datos, reglas y restricciones de verificación, definiciones por defecto y definiciones no nulas. Para mantener una buena integridad de dominio en la BD, no se definieron atributos null y se definieron correctamente cada uno de los tipos de datos para cada uno de los atributos

A continuación se muestra algunas de las restricciones de dominio utilizadas en la BD del SIGOB.

Dominio	Tipo de Dato	Campo	Restricción
Entero	INTEGER	No. enteros positivos	@var> 0
Nombre	VARCHAR(30)	Nombre hasta 30 caracteres	
Bool	BOOLEAN	Verdadero o Falso	@var = true or @var =false
FechaHora	DATATIME	Fecha y Hora	FechaInicio <= FechaFin
NumeroCI	VACHAR	Numéricos y decimales	@var> 0
Texto	TEXT	Texto en general	

Fig23 Diccionario de Datos

- **Integridad Referencial**

Garantiza que las relaciones que se establezcan entre las tablas de la BD sean coherentes, donde no deban existir valores de claves foráneas sin concordancia. Si se tiene en una tabla una columna declarada como llave foránea, sus valores deben coincidir con los valores presentes en la llave primaria de la tabla a la cual se hace referencia.

La integridad referencial implica que en todo momento todos los datos involucrados en la BD sean correctos y sin repeticiones innecesarias. Permite además que las relaciones permanezcan sincronizadas durante las operaciones de actualización y eliminación, asegurando que las modificaciones y eliminaciones que se realicen en una tabla se reflejen en la otra.

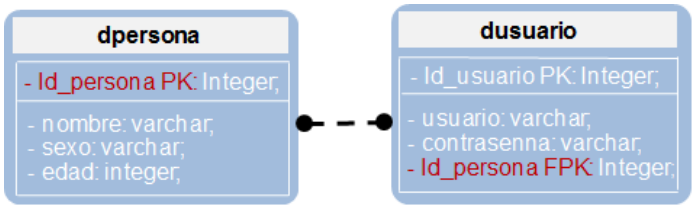

Fig24 Relación entre entidades

Existen reglas que se deben cumplir para mantener la integridad referencial:

- No se podrá introducir un valor en la tabla relacionada si antes no ha sido introducida en la tabla principal.
- No se puede eliminar un registro de una tabla principal si existen registros coincidentes en la tabla relacionada.
- No se puede cambiar un valor de la clave primaria en la tabla principal si el registro tiene registros relacionados.

3.5.2. Análisis de redundancia de la información

La redundancia de la información es uno de los problemas que en la actualidad son muy comunes en las BD, debido al cúmulo innecesario de información que se produce. Ante la más mínima introducción de redundancia es necesaria la realización de un análisis exhaustivo para eliminarla y de esta manera evitar inconvenientes en el trabajo con los datos, aumentando la velocidad de almacenamiento, rendimiento y el acceso a los mismos, minimizando al máximo el espacio de almacenamiento. De esta manera los beneficios que se obtengan serán evidentes.

En la BD diseñada para el SIGOB se realizó un estudio acerca de la redundancia de la información encontrándose existencia de la misma a través de datos comunes en cada uno de los esquemas independientes. Posteriormente se

eliminaron tales redundancias con la creación de un nuevo esquema que contiene los nomencladores que son comunes en cada esquema de las direcciones.

3.5.3. Análisis de la seguridad de la BD

En la actualidad las BD son elementos primordiales para almacenar todo tipo de información, por lo que es necesario garantizar que la misma sea consultada por los usuarios autorizados. Con una buena seguridad de la BD se lograría evitar ataques que puedan provocar eliminación o modificación no deseada de la información, manteniendo tres aspectos importantes a tener en cuenta, la integridad, confidencialidad y disponibilidad. Es importante además garantizar la recuperación de la información en caso de que ocurra alguna anomalía en la cual se produzca algún fallo con la BD que provoque la pérdida de información. Para la BD del SIGOB se definieron una serie de reglas que garantizan la seguridad de la información:

- Se establecieron contraseñas para el acceso a la BD.
- Se le realizan copias de respaldo diariamente a la BD.
- El administrador de la BD es el único que tiene todos los permisos para modificar la estructura de la BD.
- Los usuarios de la BD solo podrán insertar, modificar, eliminar y acceder a la información.

3.6. Validación funcional

Es necesario realizar pruebas funcionales en la BD con el objetivo de comprobar y observar cómo se comporta la misma bajo determinadas circunstancias, estas pruebas aseguran que la misma cumpla con los requisitos definidos sin violar la integridad de los datos.

3.6.1. Prueba de volumen

Esta prueba de volumen se centra en analizar el comportamiento de la BD a través de volúmenes de datos almacenados lo más similar posible a los esperados, verificando posibles fallas reales del sistema. Para la realización de esta prueba se implementó una función en java encargada de generar cientos de tuplas en un grupo entidades de la BD del SIGOB.

Una vez ejecutada la función no se obtuvieron problemas de límite de capacidad, desbordamiento de columnas, atributos o tipos de datos. La utilización de esta función permitió verificar la integridad de los datos, además de garantizar que el diseño de las estructuras de la BD y el gestor utilizado para el desarrollo soportan el volumen de información requerida para el correcto funcionamiento de la BD.

3.6.2. Pruebas a la capa de acceso a datos.

Las pruebas realizadas a la capa de acceso a datos tienen como objetivo fundamental comprobar que los datos sean accesibles desde la BD. En el módulo de administración se realizaron pruebas en conjunto con el cliente y el servidor donde se trabajó con una serie de datos arbitrarios buscando posibles errores en el sistema. La gestión con esta información fue correcta, demostrando eficiencia y rapidez en la capa de acceso a datos.

3.7. Aporte social y económico

SIGOB es una BD que mejorará significativamente el flujo de trabajo del Gobierno de Artemisa debido a que almacenará gran parte de la información que allí se procesa actualmente de forma manual. Esta BD brindará grandes ventajas al trabajo de este organismo garantizando la integridad, confidencialidad y seguridad de la información manejada, contribuyendo a administrar y controlar los recursos con mayor efectividad. La misma en conjunto con la aplicación forman parte del

proceso de informatización de las empresas que se está llevando a cabo en nuestro país contribuyendo de este modo al perfeccionamiento empresarial, ahorrando tiempo y esfuerzo innecesario del personal del centro llegando finalmente a la satisfacción del usuario como cliente final.

Conclusiones del capítulo

Al concluir este capítulo se puede afirmar que se han cumplido con todos los objetivos propuestos para el mismo. La implementación del módulo administrativo culminó con resultados satisfactorios, lográndose de esta manera la correcta integración entre cliente-servidor-BD. La validación del diseño concluyó con la valoración de aspectos importantes tales como la integridad, redundancia y seguridad de la información almacenada. Se demostró a través de las pruebas funcionales que se cuenta con una BD robusta y segura para brindar los servicios requeridos.

Conclusiones Generales

Concluido el presente trabajo de investigación se puede afirmar que se ha cumplido con el objetivo de Desarrollar la Base de Datos para el Sistema Informativo de la Administración Provincial de Artemisa.

- Se realizó un estudio acerca del desarrollo de datos que permitió al autor adueñarse de los conocimientos necesarios para elaborar la fundamentación teórica del presente trabajo.
- Se diseñó la base de datos relacional para el SIGOB, la cual cumple con todos los requisitos funcionales establecidos, contribuyendo a la integridad, seguridad y el acceso concurrente de los datos gestionados en la AP.
- Se implementó la capa de acceso a datos con la calidad requerida para el módulo de administración del SIGOB, lográndose una correcta gestión de la información administrativa.
- Se realizaron las pruebas necesarias a la BD para comprobar y observar cómo se comporta la misma bajo tales circunstancias obteniéndose resultados satisfactorios.

Recomendaciones

- Continuar el análisis de la existencia de entidades nomencladoras que son comunes en las direcciones de la AP, que pueden ser agregadas en el esquema que contiene a dichas entidades, evitando de esta forma la desorganización y duplicación de datos.

- Se recomienda que con la implantación de la BD en la AP se realice un análisis y revisión periódica de su rendimiento durante el funcionamiento.

- Establecer técnicas de respaldo de la BD para prevenir fallos ajenos a la misma mediante la elaboración de backups periódicos que permitan realizar una restauración en caso de ser necesario.

Referencias Bibliográficas

1. **CUJAE. 2011.** [En línea] 2011. [Citado el: 28 de Noviembre de 2011.] http://teleportal.cujae.edu.cu/dcomputacion/pregrado/plan-d/programacion-ii/conferencias/Conferencia%233%20Creacion%20de%20Bases%20de%20Datos%20.doc.

2. **EcuRed. 2012.** [En línea] 2012. [Citado el: 7 de Febrero de 2012.] http://www.ecured.cu/index.php/Sistema_Gestor_de_Base_de_Datos.

3. **Navathe, Sham y Blaha, Michael. 2010**. Patterns of Data Modeling. s.l. : Georgia Institute of Technology. College of Computing. Atlanta, Georgia, U.S.A.., 2010.

4. **Nc, Jeisson . 2011.** SlideShare Inc. [En línea] 2011. [Citado el: 1 de Diciembre de 2011.] http://www.slideshare.net/jeissonlarry/sistema-gestin-de-bases-de-datos-2657624.

5. **Riscos Núñez, Agustín . 2010-2011**. Sistemas de Gestión de Bases de Datos. [En línea] 2010-2011. [Citado el: 30 de Noviembre de 2011.] http://www.cs.us.es/cursos/bd/tema1.pdf.

6. **SIGMUR. 2011.** SIG y Teledetección en la Universidad de Murcia. [En línea] 2011. [Citado el: 4 de Diciembre de 2011.] http://www.um.es/geograf/sigmur/sigpdf/temario_9.pdf.

7. SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE. **Peñalver, G, Meneses, A y García, S. 2010.** Chile : 1er congreso Iberoamericano de Ingeniería de Proyectos, 2010.

8**. Visual Paradigm. 2012.** Visual Paradigm. [En línea] 2012. [Citado el: 11 de Diciembre de 2011.] http://www.visual-paradigm.com

9. **Álvarez, Sara. 2007**. Desarrollo Web. [En línea] 15 de Junio de 2007. Citado el: 21 de Abril de 2012.] http://www.desarrolloweb.com/articulos/arquitectura-base-de-datos.html.

Bibliografía

1. Blaha, Michael. 2010. EcuRed. *EcuRed.* [En línea] 2010. [Citado el: 28 de 2 de 2012.] http://www.ecured.cu/index.php/Patrones_de_dise%C3%B1o_de_bases_de_datos.

2. BVSCuba. 1999-2010. Centro de Ayuda BVSCuba. [En línea] Infomed, 1999-2010. [Citado el: 3 de Diciembre de 2011.] http://bvsayuda.sld.cu/ayudas/faq/cumed.

3. C.J.Date. 2003. *Introducción a los Sistemas de bases de datos.* Ciudad de la Habana : Felix Varela, 2003. 2.

4. Carrasco, Yailin Fundora. 2011. *Diseño e implementación de la base de datos del Sistema de Gestión Académica de Postgrado.* 2011.

5. Carrero, Angel. 2011. Conceptos básicos de ORM (Object Relational Mapping). Programación en Castellano. [En línea] 2011. [Citado el: 6 de 12 de 2011.] http://www.programacion.com/articulo/conceptos_basicos_de_orm_object_relational_mapping_349.

6. Colectivo de Autores CUJAE. 2011. *CUJAE.* [En línea] 2011. [Citado el: 28 de Noviembre de 2011.] http://teleportal.cujae.edu.cu/dcomputacion/pregrado/plan-d/programacion-ii/conferencias/Conferencia%233%20Creacion%20de%20Bases%20de%20Datos%20.doc..

Diseño de bases de datos relacionales. [En línea] http://usuarios.multimania.es/cursosgbd/UD4.htm.

7. Domínguez Vaillant, A. E., & Miranda Gutiérrez. 2007. *SIMDEC Sistema de Manejo de Datos de Ensayos Clínicos: Diseño e implementación de la Base de Datos.* Ciudad de La Habana (Universidad de las Ciencias Informáticas) : s.n., 2007.

8. Duran, Lore. 2011. BuenasTareas.com. [En línea] 2011. [Citado el: 6 de Diciembre de 2011.] http://www.buenastareas.com/ensayos/Ventajas-y-Desventajas-De-Bases-De/2563701.html.

9. E.V.A. UCI, I. D. S. Conferencia #13. Base de Datos, ISW 2. . 2012. EcuRed. *EcuRed.* [En línea] 8 de 2 de 2012. [Citado el: 28 de 2 de 2012.]

10. 2010. ecured. *ecured.* [En línea] 14 de 5 de 2010. [Citado el: 24 de 02 de 2012.] http://www.ecured.cu/index.php/Bases_de_datos.

11. EcuRed. 2012. EcuRed. [En línea] 2012. [Citado el: 7 de Febrero de 2012.] http://www.ecured.cu/index.php/Sistema_Gestor_de_Base_de_Datos.

12. Figueroa, Roberth G, Solís, Camilo J y Cabrera, Armando A. 2011. Entorno Virtual de Aprendizaje. [En línea] 2011. [Citado el: 13 de 12 de 2011.] http://eva.uci.cu/mod/resource/view.php?id=9303&subdir=/Metodologias/RUP.

13. Frazer, A. 1992. *"Reverse Engineering- hype, hope or here?" En Software Reuse and Reverse Engineering in Practice.* s.l. : Chapman & Hall, 1992.

14. Gavin King, Christian Bauer. 2010. *Documentación de referencia de Hibernate 3.6.3.Final .* 2010.

15. Grueso, Cesar David Fernandez. 2009. slideshar. *slideshar.* [En línea] 2 de 12 de 2009. http://www.slideshare.net/senaticscesar/bases-de-datos-conceptos-basicos.

16. *Herramientas Case.* **Alfaro, Félix Murillo. 2011.** 2011, COLECCION CULTURA INFORMATICA.

17. 2012. Hibernate.org. *Hibernate.org.* [En línea] 2012. http://www.hibernate.org/.

18. Herrera, Cristhian. 2007. Adictos al Trabajo. [En línea] 16 de Agosto de 2007. [Citado el: 6 de Febrero de 2012.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=hibernateVSEJB3.

19. Kiral. 2011. BuenasTareas.com. [En línea] 2011. [Citado el: 6 de Diciembre de 2011.] http://www.buenastareas.com/ensayos/Base-De-Datos-Unidad-1-Introducci%C3%B3n/326383.html.

20. Korth, Henry F, Silberschatz, Abraham. 2007. *Database System Concepts.* New York, NY : Fifth Edition, 2007. ISBN-13: 978-0072958867.

—. 1991. *Database System Concepts.* New York : McGraw-Hill, 1991. 0071008047.

21. 2007. Mapeo Relacional de Objetos (ORM) en PHP | El Blog de Leech. [En línea] 2007. [Citado el: 7 de 12 de 2011.] http://www.dr-leech.com.ar/2007/10/08/mapeo-relacional-de-objetos-orm-en-php.

22. Marqués, Mercedes. 2011. *Bases de datos.* s.l. : Col·lecció Sapientia, 2011. 978-84-693-0146-3.

23. Martinez Bravo, Indira . 2011. Informatica. [En línea] 2011. [Citado el: 29 de Noviembre de 2011.] http://indira-informatica.blogspot.com/2007/09/qu-es-un-sistema-de-gestin-de-base-de.html.

24. Masadelante.com. 1999-2011. masadelante.com. [En línea] Masadelante.com , 1999-2011. [Citado el: 1 de Diciembre de 2011.] http://www.masadelante.com/faqs/base-de-datos.

25. Mato, Rosa Maria. 2006. *Sistemas de Bases de Datos.* Ciudad de la Habana, Cuna : Felix Varela, 2006. 1.

26. Merono, Alex. 2011. slideshare. *Sistemas Gestores de Bases de Datos.* [En línea] 2011. [Citado el: 3 de Diciembre de 2011.] http://www.slideshare.net/alexmerono/sistemas-gestores-de-bases-de-datos.

27. Montero, César Luza. 2010. *Modelado y Diseño de Bases de Datos.* Lima : s.n., 2010.

28. 2011. MySQL. *About MySQL.* [En línea] 2011. http://www.mysql.com/about.

29.Rafaelma. 2010. PostgreSQL-es. [En línea] 2 de Octubre de 2010. [Citado el: 17 de Febrero de 2012.] http://www.postgresql.org.es/sobre_postgresql.

30. Romero Ciccone, Wilder J. 2011. Wilder J. Romero Ciccone. [En línea] 21 de Abril de 2011. [Citado el: 5 de Diciembre de 2011.] http://cicconew.wordpress.com/2011/04/21/base-de-datos-relacional/.

31. 2010. Tu informática f@cil. *Tu informática f@cil.* [En línea] 19 de 10 de 2010. http://www.tuinformaticafacil.com/herramientas-desarrollo/sql-power-architect-herramienta-de-modelado-de-datos.

32. Verschoor, Rob. 2011. Sybase. [En línea] Sybase Iberia, 16 de 12 de 2011. [Citado el: 24 de 2 de 2012.] http://www.sybase.es/manage/embedded-databases. CIF B-80506629.

33. Visual Paradigm. [En línea] [Citado el: 11 de 12 de 2011.] http://www.visual-paradigm.com.

34. W.Hansen, Gary and Hansen, James V. 2006. *Diseño y Administración de Base de Datos 2da Edición.* 2006.