Facultad Regional Mártires de Artemisa

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

Base de datos para los Módulos de las Direcciones de Justicia y Auditoría de la Administración Provincial de Artemisa.

Autora:

Yenisey Torres Vanegas

Tutor:

Ing. Leosmel Zayas Castillo

Co-Tutor:

Lic. Roberto Cruz Moreno

Artemisa, Julio 2012

¨Año 53 de la Revolución¨

DECLARACIÓN DE AUTORÍA

Declaro que soy la única autora de este trabajo y autorizo a la Universidad de Ciencias Informáticas para que le den el uso que estimen conveniente.

Para que así conste firmo la presente a los _____ días del mes _____ del año _____.

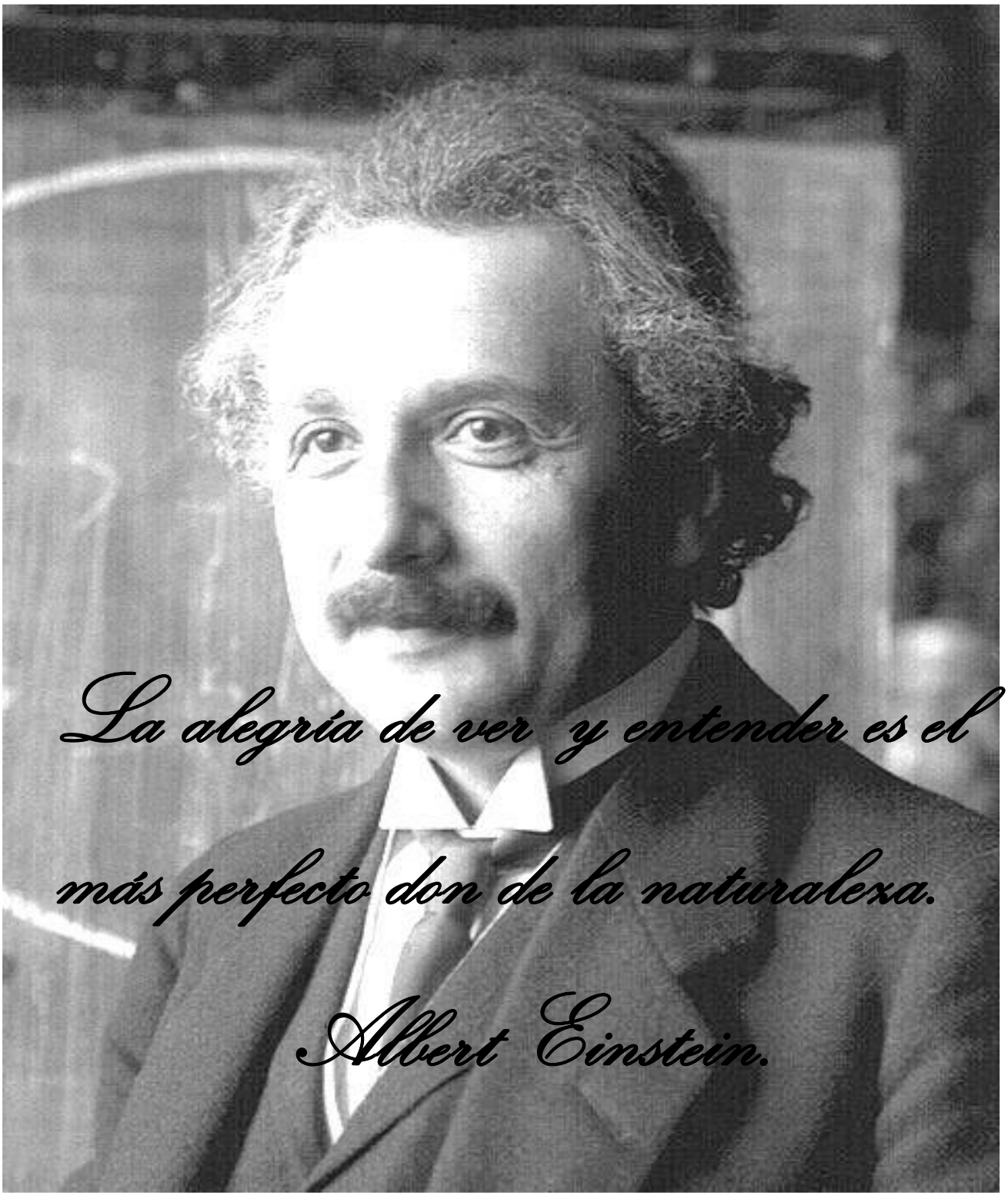
_____ _____

Yenisey Torres Vanegas
Autora

Ing. Leosmel Zayas Castillo
Tutor

_____

Lic. Roberto Cruz Moreno
Co-Tutor


La alegría de ver y entender es el
más perfecto don de la naturalexa.
Albert Einstein.

AGRADECIMIENTOS

A:

Mi familia.

Mis todos los que han cooperado.

DEDICATORIA

A:

Mis padres por ayudarme siempre y encaminarme por el camino correcto.

Mi abuela por su fuerza y sus ganas de tener una ingeniera en la familia.

A mis compañeros y amigos de la universidad.

A mi tutor.

RESUMEN

Los servicios que brindan las nuevas tecnologías de la información y comunicaciones (TIC), posibilitan la creación de modelos, procesos y aplicaciones donde se puede llevar el control informático de toda una serie de operaciones en una determinada entidad. Para ejecutar estas acciones de forma virtual, se propone la elaboración de un proyecto basado en la realización de una Base Datos para la Administración Provincial de Artemisa (APA).

Este trabajo se centra en el diseño y la implementación de una base de datos con una capa de acceso a datos que permita gestionar la información del APA, así como la seguridad del mismo, teniendo como características fundamentales, el soporte de altos niveles de concurrencia y requerimientos estrictos de respuesta temporal; pues el sistema debe permitir el almacenamiento en tiempo real. Se realiza un estudio de los principales gestores de bases de datos que existen, las herramientas más utilizadas para diseñar bases de datos y la capa de acceso a datos, con la selección de las herramientas que más se ajustan a las características del trabajo a desarrollar. Luego se describe la solución propuesta y finalmente se hace una valoración del trabajo realizado.

Palabras Claves: APA, Base de Datos, Capa de Acceso, Herramientas.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Introducción

¨La Proactividad es sinónimo de acción, de ejecución, de tomar la iniciativa, de moverse y de mover al de al lado si es necesario. Es sinónimo de actitud positiva y constructiva, de enfoque didáctico, prefiere ir en lugar de esperar a que vengan, prefiere llamar en lugar de aguardar el "ring" del teléfono, es el opuesto a la pasividad, a la contemplación cansina, a la innecesaria crítica mordaz que no aporta nada, a la lamentación bobalicona o la queja infantil. La Proactividad no es ni siquiera parte de la solución, es la solución. La Pasividad es el problema¨. (Nunes, 2008).

Con la creación de las primeras computadoras electrónicas, surge un nuevo campo dentro de la ciencia: la Informática. Esta ciencia abarca el estudio y aplicación del tratamiento automático de la información por medio de computadoras y se encuentra fuertemente atada al desarrollo de las tecnologías de la informática y las comunicaciones (TIC). Con el paso de los años se han encontrado aplicaciones en numerosas áreas de la sociedad como las telecomunicaciones, la educación, el comercio, la medicina, entre otras. Aún conociendo la finalidad con que cuente cierta aplicación, la necesidad de almacenar y recuperar información es constante, es por ello que en los últimos años el mecanismo más utilizado para transferirla desde su origen hasta la aplicación es la base de datos, lo que en un principio, lo más importante era la posibilidad de guardar los datos para recuperarlos y ya posteriormente, se ha convertido en una exigencia básica que se da por asumida, pasando a primer plano aspectos como la posibilidad de acceder a ellos de manera simultánea desde múltiples puntos y el rendimiento en la ejecución de las operaciones.

Cuba por su parte se encuentra inmersa en el desarrollo de la informatización de sus empresas, ya que se han dado cuenta que el éxito está en la estabilidad, integridad, disponibilidad y seguridad de los datos y para logar todo esto hacen uso de las bases de datos convirtiéndose las mimas en un proceso estratégico de cara a sus competidores.

La Universidad de las Ciencias Informáticas (UCI) fue creada con dos objetivos: informatizar el país y desarrollar la industria del software para contribuir al desarrollo económico del mismo. Como idea de nuestro Comandante en Jefe, de la UCI se derivan 3 facultades regionales siendo una de ellas la Facultad Regional "Mártires de Artemisa (FRA) ubicada en la nueva provincia de Artemisa la que se desarrolla en un proyecto llamado ¨Sistema Informativo de la Administración Provincial de Artemisa¨ (SIAPA), que pretende la informatización del Gobierno Provincial, ya que el mismo no cuenta con un nivel de automatización en los procesos que desarrolla cada dirección que lo conforma.

Se asume la investigación de un problema muy común en el ámbito actual de bases de datos, y es el de lograr alguna variante de integridad y disponibilidad de los datos, que pueda cumplir las funcionalidades que se piden actualmente en la Administración Provincial de Artemisa (APA), cumpliendo con los requisitos de usabilidad, rendimiento, soporte, portabilidad, seguridad y confiabilidad.

En el año 2011 en Cuba se realizó una nueva distribución política administrativa debido a la gran cantidad de municipios que contenía la provincia de La Habana. Con esta nueva distribución surgen las provincias de Mayabeque y Artemisa. En esta última, en el municipio cabecera se encuentra ubicado el órgano del Gobierno Provincial, el cual está dividido en 32 órganos de direcciones entre las cuales se encuentra Justicia y Auditoría.

Entre las tareas fundamentales de estos órganos de direcciones está, la misión de ejecutar y asistir al Gobierno del territorio en la implementación de la política jurídica aprobada para su esfera de atención y por último ejecutar Auditoría y otras acciones de control a la administración y al control de los recursos materiales, humanos y financieros que dispone el Consejo de la Administración y las entidades subordinadas.

El proceso de gestión de información es defectuoso debido a que en las entidades no existe una adecuada forma de almacenar y recuperar la información. Se dificulta el análisis de los reportes ya que los datos se presentan en hojas de cálculo, la información no está íntegra y es inconsistente. Además, es un riesgo la posibilidad real de sufrir un daño que puede contribuir negativamente en el

desarrollo del funcionamiento del gobierno.

Debido a los problemas de organización y seguridad del órgano de dirección la información puede ser afectada de origen accidental o intencionado como es el caso de desastres naturales o domésticos (incendios, inundaciones, cortocircuitos, terremotos, etc.), averías en los equipos informáticos o accesorios (plantas generadoras, instalación eléctrica), errores en la introducción de datos, como borrado indebido de archivos o modificaciones no autorizadas y difusión mal intencionada de información. Cuando se necesita una información urgente se tiene que buscar de forma manual y dentro de grandes volúmenes de información. Además, la información con la que se toman las decisiones muchas veces no se encuentra centrada, sino en varias computadoras.

Esta mala gestión de la información provoca problemáticas para la generación de reportes inmediatos y de la búsqueda de los datos referente los procesos de deficiencias detectadas y medidas disciplinarias aplicadas a los cuadros, dirigentes y trabajadores. Además, el proceso de almacenamiento trae consigo demora en la tramitación de los datos, lo que provoca que en determinados momentos no se tenga constancia real del cumplimiento de asesoramiento jurídico del sector agropecuario, registro civil y registro de la propiedad, así como su pérdida y replicación. También, existen deficiencias en los datos pues es mediante hojas de cálculo y en formato duro donde se traen a la dirección las acciones de control con presuntos hechos delictivos y la actividad notarial, provocando duplicado de estos, causando grandes consecuencias para la toma de decisiones en la provincia.

Una vez conocido el problema que presenta la entidad se da a conocer el:

Problema de investigación:

¿Cómo almacenar la información relacionada con los procesos de almacenamiento de los datos en las direcciones de Justicia y Auditoría de la APA de forma que se garantice la integridad y disponibilidad en el acceso y distribución de los datos?

A partir del problema de investigación planteado se define como:

Objeto de estudio:

La gestión de datos en los sistemas gubernamentales.

Centrado en el:

Campo de acción:

Los procesos de gestión de datos de Justicia y Auditoría en el SIAPA (Sistema Informativo a la Administración Provincial de Artemisa).

Para dar cumplimiento al campo de acción planteado se tiene como:

Objetivo general:

Desarrollar la base de datos para las Direcciones de Justicia y Auditoría del SIAPA, de manera que garantice la integridad y disponibilidad en el acceso y distribución de los datos.

Se tiene como **Idea a Defender:** Con el desarrollo de la base de datos para las direcciones de Justicia y Auditoría se garantiza el correcto almacenamiento y recuperación de la información en la APA.

Una vez definido el problema de investigación, se plantean los **objetivos específicos** que permitan encontrar una solución científicamente fundamentada y obtener una arquitectura apropiada para el almacenamiento distribuido de los datos en el Gobierno Provincial, ellos son:

1. Elaborar la Fundamentación Teórica de la investigación.
2. Estudiar las herramientas, tecnologías y metodologías para el diseño de la BD.
3. Diseñar la base de datos para las Direcciones de Justicia y Auditoría del APA.
4. Implementar la capa de acceso a datos para las Direcciones de Justicia y Auditoría del APA.
5. Validar los resultados obtenidos.

Para dar cumplimiento a los objetivos específicos se plantean ciertas **tareas de investigación** que permiten lograr su alcance, dentro de las que se menciona:

1. Establecimiento de los fundamentos teórico-metodológicos de la investigación.
2. Descripción de los requisitos, la arquitectura y el funcionamiento del SIAPA.

3. Descripción de la tecnología a utilizar.

4. Desarrollo del diseño lógico de la base de datos.

5. Modelación física de la base de datos e implementación de la capa de acceso a datos.

6. Normalización y Validación de los diagramas que componen el Modelo de datos.

7. Validación funcional del diseño de la base de datos.

Los métodos científicos que fueron usados son de dos tipos:

Métodos Empíricos: Observación.

Métodos Teóricos: Analítico-Sintético, Inductivo-Deductivo, y el Modelado.

El método **Analítico-Sintético:** Se analiza y resume la documentación recopilada, a través de los diferentes medios bibliográficos, para de esa forma desarrollar un mejor diseño e implementación del sistema que se propone.

El **Inductivo-Deductivo** se usó para llegar de casos particulares presentes en el ámbito del almacenamiento, a elementos generales, mediante la inducción, mientras que la deducción es el procedimiento de llegar a especificidades desde generalidades.

El **Modelado** 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.

El método de la **Observación**: Es usado en la recolección de la información, es factible y práctico y permite a partir de examinar la situación presente llegar a nuevos conocimientos para lograr una solución práctica del problema planteado, llevando a cabo el registro visual del fenómeno en cuestión.

Este trabajo, como principal resultado obtenido, tiene el de haber propuesto la solución al tema de la integridad y disponibilidad de los datos que se utilizará en el proyecto de informatizar las correspondientes direcciones del APA. Esta aportación teórica tiene la misma significación práctica, al ser utilizada inmediatamente en el objetivo mayor que es informatizar este ministerio.

La novedad de este trabajo no radica en se hayan descubierto metodologías, o modos de interpretar la realidad de la integridad de los datos, sino que se ha utilizado lo que se conoce sobre el tema, tampoco se plantea la búsqueda de soluciones que requirieran nueva infraestructura, solamente la ya existente en el país.

También es posible imaginar que el aporte teórico de este trabajo consiste en haber planteado la solución al tema de bases de datos, que en la práctica, se usará muy pronto.

La estructura en que se desarrolla es la que se muestra a continuación:

Capítulo 1: Fundamentación Teórica. Se abordan los temas relacionados con las bases de datos respaldando el presente trabajo, abarcando definiciones de conceptos, definición de la metodología, herramientas y tecnologías a usar para el desarrollo de la solución propuesta.

Capítulo 2: Diseño y arquitectura de la base de datos. Se describen los patrones de diseño a utilizar y se definen los artefactos correspondientes al rol diseñador de base de datos, como artefactos, modelo entidad relación, la persistencia de los datos, etc.

Capítulo 3: Implementación y Validación de la base de datos. Implementación de los artefactos del diseño de las bases de datos e integración con la aplicación en la capa de persistencia.

CAPÍTULO 1

CAPÍTULO 1: Fundamentación Teórica.

Introducción

En este capítulo se abordarán temas relacionados con el estado del arte de las diferentes herramientas que existen en el mundo para llevar a cabo base de datos relacional de distintas empresas. Se definen las herramientas, metodología y lenguajes a utilizar, argumentando el porqué de su elección.

1.) Historia de las bases de datos.

Las bases de datos surgieron a partir de la necesidad de almacenar información para su posterior consulta, ya que las nuevas industrias contaban con gran cantidad de información y no poseían máquinas que facilitaran el trabajo manual, siendo así la búsqueda de la información se hacía un poco engorrosa.

Cuando el hombre necesita guardar conocimiento o seguir el rastro de la información, lo escribe, y lo cataloga usando índices de papel. Así el libro fue el primer tipo de base de datos. No eran bases de datos electrónicas, y sin embargo servían para el mismo propósito.

No fue hasta el año 1970 cuando **Edgar Frank Codd** de IBM introdujo la idea de un modelo relacional de Bases de datos, antes de eso la mayoría de las bases de datos estaban basadas en un modelo de red o una simple estructura de archivo plano. Este hecho dio paso al nacimiento de la segunda generación de los Sistemas Gestores de Bases de Datos. Como consecuencia de esto, durante la década de 1970, Lawrence J. Ellison, más conocido como Larry Ellison, a partir del trabajo de Edgar sobre los sistemas de bases de datos relacionales, desarrolló el Relational Software System (Sistema de Software Relacional), o lo que es lo mismo, lo que actualmente se conoce como Oracle Corporation, desarrollando así un sistema de gestión de bases de datos relacional con el mismo nombre que dicha compañía (Historia de la Informática, 2011).

En la década de 1990 la investigación en bases de datos giró en torno a las bases de datos orientadas a objetos. Las cuales han tenido bastante éxito a la hora de gestionar datos complejos en los campos donde las bases de datos relacionales no han podido desarrollarse de forma eficiente. Así se desarrollaron herramientas como Excel y Access del paquete de Microsoft Office que marcan el inicio de las bases de datos orientadas a objetos. Así se creó la tercera generación de sistemas gestores de bases de datos.

En la actualidad las bases de datos son indispensables en cualquier tipo de software o aplicación de gestión de información, sea de escritorio o web, comúnmente estas se comunican con la aplicación a través de un controlador y una conexión escrita en algún lenguaje apropiado para la tarea.

Cuba también forma parte de esa revolución de tecnología ya que en la actualidad, y debido al gran desarrollo tecnológico de campos como la Informática y la Electrónica, la mayoría de las bases de datos están en formato digital (electrónico), que ofrece un amplio rango de soluciones al problema de almacenar datos en las empresas.

La Universidad de las Ciencias Informáticas (UCI) se crea en medio de este desarrollo tecnológico, este centro productivo, cuya misión es producir software y servicios informáticos a partir de la vinculación estudio – trabajo como modelo de formación, es considerado como el mayor productor de software en el país, utilizando en gran medida las bases de datos relacionales.

Por su parte la Facultad Regional Mártires de Artemisa aunque es muy nueva y no cuenta con la experiencia y la masividad de la Sede Central, representa un papel importante en la creación de dichos software y en los adelantos de la tecnología precisamente en la provincia de Artemisa, ya que se ha encaminado en la informatización de la Provincia colaborando en proyectos como la Administración de los datos de determinados sectores empresariales como es el caso de Gobierno Provincial.

1.1) Base de Datos

¿Qué es una Base de datos?

Una Base de datos es un 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 (Mato, 2005).

Una Base de datos es una serie de datos relacionados que forman una estructura lógica, es decir, una estructura reconocible desde un programa informático. Esa estructura no solamente contiene los datos en sí, sino la forma en la que se relaciona. (Marqués A, 2001).

Una base de datos es un conjunto de datos almacenados entre los que existen relaciones lógicas y ha sido diseñada para satisfacer los requerimientos de la información de una empresa u organización. En una base de datos, además de sus datos se almacena su descripción (Codd, 2011).

A modo de resumen se puede decir que las bases de datos son ese espacio o lugar electrónico destinado a almacenar datos que pueden estar interrelacionados lógicamente, para satisfacer las necesidades o demandas de información de un usuario o sistema informático en representación de una persona o cualquier tipo de entidad u organización.

1.1.1) Características de la Base de datos.

Independencia lógica y física de los datos: se refiere a la capacidad de modificar una definición de esquema en un nivel de la arquitectura sin que esta modificación afecte al nivel inmediatamente superior. Para ello un registro externo en un esquema externo no tiene por qué ser igual a su registro correspondiente en el esquema conceptual.

Redundancia mínima: se trata de usar la base de datos como repositorio común de datos para distintas aplicaciones.

Acceso concurrente por parte de múltiples usuarios: control de concurrencia mediante técnicas de bloqueo o cerrado de datos accedidos.

Distribución espacial de los datos: la independencia lógica y física facilita la posibilidad de sistemas de bases de datos distribuidas. Los datos pueden encontrarse en otra habitación, otro edificio e incluso otro país. El usuario no tiene por qué preocuparse de la localización espacial de los datos a los que accede.

Integridad de los datos: se refiere a las medidas de seguridad que impiden que se introduzcan datos erróneos. Esto puede suceder tanto por motivos físicos (defectos de hardware, actualización incompleta debido a causas externas), como de operación (introducción de datos incoherentes).

Consultas complejas optimizadas: la optimización de consultas permite la rápida ejecución de las mismas.

Seguridad de acceso y auditoría: se refiere al derecho de acceso a los datos contenidos en la base de datos por parte de personas y organismos. El sistema de auditoría mantiene el control de acceso a la base de datos, con el objeto de saber qué o quién realizó una determinada modificación y en qué momento.

Respaldo y recuperación: se refiere a la capacidad de un sistema de base de datos de recuperar su estado en un momento previo a la pérdida de datos.

Acceso a través de lenguajes de programación estándar: se refiere a la posibilidad ya mencionada de acceder a los datos de una base de datos mediante lenguajes de programación ajenos al sistema de base de datos propiamente dicho. (ORTIZ, 2008)

1.1.2) Ventajas de las Base de Datos.

Redundancia controlada: Debido al sistema tradicional de archivos independientes, los datos se duplicaban constantemente lo cual creaba mucha duplicidad de datos y creaba un problema de sincronización cuando se actualizaba un dato en un archivo en particular. La redundancia se controla, no se elimina por completo.

Consistencia: Al controlarse la redundancia, cuando actualizas un dato, todos los usuarios autorizados de la Base de Datos pueden ver el cambio independientemente de que estén trabajando en distintos sistemas.

Integridad: La base de datos tiene la capacidad de validar ciertas condiciones cuando los usuarios entran datos y rechazar entradas que no cumplan con esas condiciones. El DBA (Data Base Administrador) es responsable de con esas condiciones. El DBA (Data Base Administrador) es responsable de establecer esas validaciones.

Seguridad: El DBA al tener control central de los Datos, la Base de Datos le provee mecanismos que le permiten crear niveles de seguridad para distintos tipos de Usuarios.

Flexibilidad y rapidez al obtener datos: Aquí el usuario puede fácilmente obtener información de la Base de Datos con tan solo escribir unas breves oraciones. Esto evita el antiguo y burocrático proceso de llenar una petición al Centro de Cómputos para poder obtener un informe.

Aumenta la productividad de los programadores: Debido a que los programadores no se tienen que preocupar por la organización de los datos ni de su validación, se pueden concentrar en resolver otros problemas inmediatos, mejorando de ese modo su productividad.

Mejora el mantenimiento de los programas: Debido a que los datos son independientes de los programas (a diferencia de Cobol), si ocurre un cambio en la estructura de una tabla (archivo), el código no se afecta

Independencia de los Datos: Debido a lo que se mencionó previamente, debido a lo que se mencionó previamente, los datos pueden modificarse para por ejemplo mejorar el "performance" de la Base de Datos y como consecuencia, no se tiene que modificar los programas. (BASE DE DATOS, 2010)

1.1.3) Clasificación de las Bases de datos.

Las bases de datos pueden clasificarse de varias maneras, de acuerdo con el contexto que se esté manejando, o la utilidad de la misma:

Bases de datos estáticas:

Éstas son bases de datos de solamente lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones (Christopher J. ,2003).

Bases de datos dinámicas

Éstas son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización, borrado y adición de datos, además de las operaciones fundamentales de consulta. Un ejemplo de esto puede

ser la base de datos utilizada en un sistema de información de una tienda de abarrotes, una farmacia, un videoclub (Christopher J. ,2003).

Bases de datos bibliográficas

Solo contienen un representante de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, etc. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque si no, se estaría en presencia de una base de datos a texto completo. Como su nombre lo indica, el contenido son cifras o números (Christopher J. ,2003).

Bases de datos relacionales

Éste es el modelo utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Tras ser postulados sus fundamentos en 1970 por Edgar Frank Codd, de los laboratorios IBM en San José (California), no tardó en consolidarse como un nuevo paradigma en los modelos de base de datos. Su idea fundamental es el uso de "relaciones". Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas". Pese a que esta es la teoría de las bases de datos relacionales creadas por Codd, la mayoría de las veces se conceptualiza de una manera más fácil de imaginar. Esto es pensando en cada relación como si fuese una tabla que está compuesta por registros (las filas de una tabla), que representarían las tuplas, y campos (las columnas de una tabla). En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos. La información puede ser recuperada o almacenada mediante "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información.

El lenguaje más habitual para construir las consultas a bases de datos relacionales es SQL, ya que es un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales. Durante su diseño, una base de datos relacional pasa por un proceso al que se le conoce como normalización

de una base de datos (Christopher J. ,2003).

1.1.4) Diseño de la base de datos.

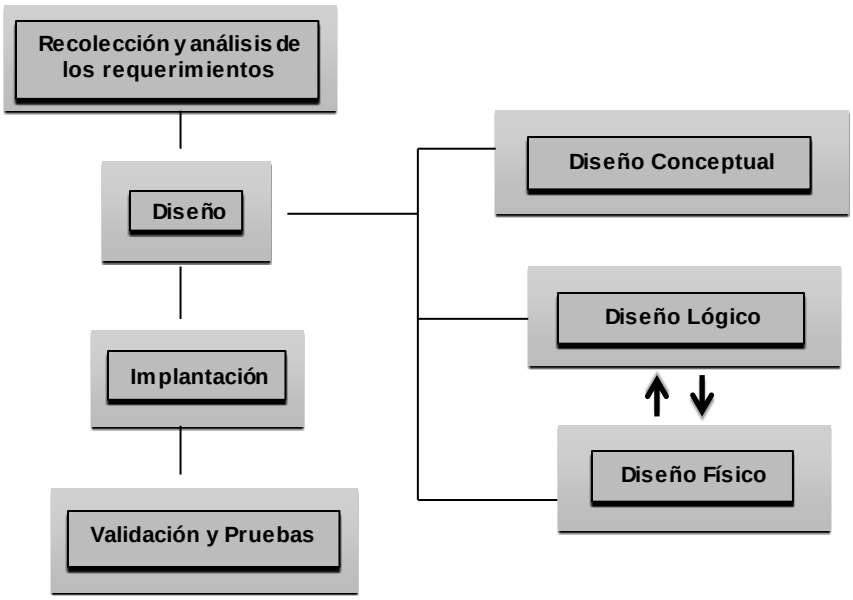


Figura 1: Diseño de la Base de Datos

En el **diseño conceptual**: se construye un esquema de la información del entorno o el sistema donde se implantara la base de datos, independientemente de cualquier consideración física. A este se le denomina **esquema conceptual** y tiene como objetivo lograr la comprensión de la estructura, semántica, relaciones y restricciones de la base de datos, realizar una descripción estable del contenido de la base de datos, lograr la comunicación entre los usuarios, analistas y diseñadores, para dar paso al diseño lógico. Se construye utilizando la información que se encuentra en la especificación de los requisitos de usuarios.

El **diseño lógico**: se obtiene en un esquema físico, a partir del esquema lógico. En el proceso en el cual se realiza la implementación de la base de datos, las estructuras de almacenamiento y los métodos para acceder a la información.

El **diseño físico**: se obtiene un esquema físico, a partir del esquema lógico. En el proceso en el cual se realizan la implementación de la base de datos, la estructura de almacenamiento y los métodos para acceder a la información, para dar comienzo a esta etapa se debe haber decidido cual SGBD se va a utilizar, ya que el sistema físico se adapta a el.

Entre el diseño físico y el lógico hay una retroalimentación ya que algunas de las

decisiones que se toman durante el diseño físico para mejorar las prestaciones, pueden afectar a la estructura del sistema lógico.

Una vez concluida estas fases ya existe prácticamente el sistema de base de datos, pasando así al momento de la evaluación del sistema y posteriormente a una ultima fase de instalación y mantenimiento del mismo. Casi siempre es necesario modificar el diseño de la base de datos tras su puesta en funcionamiento, por lo que se incluye explícitamente esta fase en el proceso de diseño de la base de datos.

Algunas gigantescas Bases de datos.

Alguna de estas gigantescas bases de datos:

AT&T, Google, ChoicePoint (información telefónica, páginas blancas).

1.2) Sistemas Gestores de Base de datos (SGBD)

Un Sistema Gestor de Bases de datos o SGBD (aunque se suele utilizar más a menudo las siglas DBMS procedentes del inglés, Data Base Management System) es el software que permite a los usuarios procesar, describir, administrar y recuperar los datos almacenados en una base de datos.

Un SGBD (sistema gestor de base de datos) es una colección de archivos interrelacionados o un conjunto de programas que permiten a los usuarios acceder y modificar estos archivos. El propósito principal de un sistema de bases de datos es proporcionar a los usuarios una visión abstracta de los datos (Castro, 2005).

1.2.1) Características de los Sistemas Gestores de Base de datos.

Abstracción de la información. Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos. Da lo mismo si una base de datos ocupa uno o cientos de archivos, este hecho se hace transparente al usuario. Así, se definen varios niveles de abstracción (Castro, 2005).

Independencia. La independencia de los datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella (Castro, 2005).

Redundancia mínima. Un buen diseño de una base de datos logrará evitar la aparición de información repetida o redundante. De entrada, lo ideal es lograr una

redundancia nula; no obstante, en algunos casos la complejidad de los cálculos hace necesaria la aparición de redundancias (Castro, 2005).

Consistencia. En aquellos casos en los que no se ha logrado esta redundancia nula, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea (Castro, 2005).

Seguridad. La información almacenada en una base de datos puede llegar a tener un gran valor. Los SGBD deben garantizar que esta información se encuentra segura frente a usuarios malintencionados, que intenten leer información privilegiada; frente a ataques que deseen manipular o destruir la información; o simplemente ante las torpezas de algún usuario autorizado pero despistado Normalmente, los SGBD disponen de un complejo sistema de permisos a usuarios y grupos de usuarios, que permiten otorgar diversas categorías de permisos (Castro, 2005).

Integridad. Se trata de adoptar las medidas necesarias para garantizar la validez de los datos almacenados. Es decir, se trata de proteger los datos ante fallos de hardware, datos introducidos por usuarios descuidados, o cualquier otra circunstancia capaz de corromper la información almacenada (Castro, 2005).

Respaldo y recuperación. Los SGBD deben proporcionar una forma eficiente de realizar copias de respaldo de la información almacenada en ellos, y de restaurar a partir de estas copias los datos que se hayan podido perder (Castro, 2005).

Control de la concurrencia. En la mayoría de entornos (excepto quizás el doméstico), lo más habitual es que sean muchas las personas que acceden a una base de datos, bien para recuperar información, bien para almacenarla. Y es también frecuente que dichos accesos se realicen de forma simultánea. Así pues, un SGBD debe controlar este acceso concurrente a la información, que podría derivar en inconsistencias (Castro, 2005).

1.2.2) Elementos que componen un Sistema Gestor de Base de datos.

Un sistema de bases de datos sirve para integrar los datos. Este está compuesto por los siguientes elementos:

Hardware: Máquinas en las que se almacenan las bases de datos. Incorporan unidades de almacenamiento masivo para este fin.

Software: Es el sistema gestor de bases de datos. El encargado de administrar las bases de datos.

Datos: Incluyen los datos que se necesitan almacenar y los metadatos que son datos que sirven para describir lo que se almacena en la base de datos.

Usuarios: Personas que manipulan los datos del sistema. Existen tres categorías:

Usuarios finales: Aquellos que utilizan datos de la base de datos para su trabajo cotidiano que no tiene por qué tener que ver con la informática. Normalmente, no utilizan la base de datos directamente, si no, que utilizan aplicaciones creadas para ellos a fin de facilitar la manipulación de los datos. Estos usuarios solo acceden a ciertos datos.

Desarrolladores: Analistas y programadores encargados de generar aplicaciones para los usuarios finales.

Administradores: También llamados DBA (Data Base Administrator), se encargan de gestionar las bases de datos.

1.2.3) Arquitectura de los Sistemas de Base de datos.

Existen 3 características importantes relacionadas con los SBD:

1. La separación de los programas de aplicación y de los datos.
2. El poder manejar múltiples vistas por parte de los usuarios.
3. Para almacenar el esquema de la base de datos el uso de un catálogo.

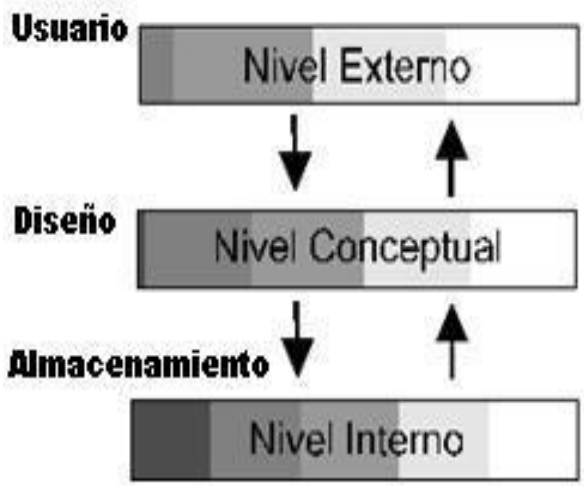


Figura 2: Arquitectura de los Sistemas Gestores de Base de Datos

Nivel interno:

Es el nivel más cercano al almacenamiento físico de los datos. Emplea un modelo físico y los únicos datos que existen están realmente en este nivel (Marqués, 2001). Permite escribir los datos tal cual 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 (Marqués, 2001).

Nivel conceptual:

Describe la estructura de toda la base de datos para una comunidad de usuarios mediante un esquema conceptual. Este esquema oculta los detalles de las estructuras de almacenamiento y describe entidades, atributos, relaciones, operaciones de los usuarios y restricciones. En este nivel se puede utilizar un modelo conceptual o un modelo lógico para especificar el esquema (Marqués, 2001).

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. En este nivel se puede utilizar un modelo conceptual o un modelo lógico para especificar los esquemas. Es el más cercano al usuario. La arquitectura se divide en tres niveles generales: interno, conceptual y externo (Marqués, 2001).

Nivel Interno:

Es el más cercano al almacenamiento físico, o sea, el relacionado con la forma en que los datos están realmente almacenados (Marqués, 2001).

Nivel Externo:

Es el más cercano a los usuarios, o sea, el relacionado con la forma en que los datos son vistos por cada usuario individualmente (Marqués, 2001).

Nivel Conceptual:

Es un nivel intermedio entre los dos anteriores (Marqués, 2001).

Ejemplos de Sistema Gestor de Base de datos.

Oracle y PostgreSQL: Son sistemas de base de datos poderosos. Administra muy bien grandes cantidades de datos y suelen ser utilizadas en intranets y sistemas de gran calibre.

MySQL: es una base de datos con licencia GPL basada en un servidor. Se caracteriza por su rapidez. No es recomendable para grandes volúmenes de datos.

MS SQL Server: es una base de datos más potente que Access desarrollada por Microsoft. Se utiliza para manejar grandes volúmenes de informaciones. (R.ERNANDO, 2008)

1.2.4) Herramientas de un Sistema Gestor de Base de datos.

El éxito del DBMS reside en mantener la seguridad e integridad de los datos. Lógicamente tiene que proporcionar herramientas a los distintos usuarios. Entre las herramientas que proporciona están:

Herramientas para la creación y especificación de los datos. Así como la estructura de la base de datos.

Herramientas para la manipulación de los datos de las bases de datos, para añadir, modificar, suprimir o consultar datos.

Herramientas de recuperación en caso de desastre.

Herramientas para administrar y crear la estructura física requerida en las unidades de almacenamiento.

Herramientas para la creación de copias de seguridad.

Herramientas para la gestión de la comunicación de la base de datos.

1.2.5) Funciones del Sistema Gestor de Base de Datos.

Función de descripción: Sirve para describir los datos, sus relaciones y sus condiciones de acceso e integridad. Además del control de vistas de usuarios y de la especificación de las características físicas de la base de datos. Para poder realizar todas estas operaciones se utiliza un lenguaje de definición de datos o DDL (Data Definition Language).

Función de manipulación: Permite buscar, añadir, suprimir y modificar datos de la base de datos. El DBMS proporciona un lenguaje de manipulación de datos (DML) para realizar esta función.

Función de control: Incorpora las funciones que permiten una buena comunicación con la base de datos. Además proporciona al DBA los procedimientos necesarios para realizar su labor.

1.3) Herramientas y tecnologías y metodologías a utilizar en el desarrollo de la Base de Datos para las Direcciones de Justicia y Auditoría del APA.

1.3.1) Herramientas.

Las herramientas que son utilizadas para el desarrollo de este trabajo son herramientas CASE las cuales no son más que un conjunto de programas que brindan asistencia para el desarrollo del ciclo de vida del proyecto. Con la ayuda de ellas se ha podido confeccionar de manera más cómoda y rápida los diagramas de diseño de la base de datos.

Herramienta SQL Power Architect

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 (herramienta-de-modelado-

de-datos, 2011).

Características del Power Architect:

1. Permite acceder a las bases de datos a través de JDBC(es una API que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java)
2. Puedes conectarte a múltiples bases de datos al mismo tiempo.
3. Compara modelos de datos y estructuras de bases de datos e identifica las discrepancias.
4. Drag-and-drop (Arrastrar y soltar (drag and drop) es una expresión informática que se refiere a la acción de mover con el ratón objetos de una ventana a otra o entre partes de una misma ventana.) de las tablas origen y las columnas en el área de trabajo.
5. Ingeniería directa/inversa para PostgreSQL, Oracle, MS SQL Server y muchas más.
6. Todos los proyectos se guardan en formato XML.
7. OLAP modelos de esquema: cubos, medidas, dimensiones, jerarquías y niveles.

Herramienta Visual Paradigm

Es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software, análisis y diseño orientado a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Es propietario y multilenguaje (Bases de datos, 2011).

Características de Visual Paradigm

Entre la gama de características y/o funcionalidades que soporta se encuentran:

1. Ingeniería inversa.
2. Diagramas de flujo de datos.
3. Generación de BD.

4. Transformación de diagramas ER en tablas en la BD.
5. Ingeniería inversa en BD, desde SGBD existentes a diagramas de Entidad-Relación.
6. Soporta múltiples usuarios trabajando sobre el mismo proyecto.
7. Genera la documentación del proyecto automáticamente en varios formatos como Web o pdf.

Herramienta PgAdmin.

PgAdmin es la más popular y completa plataforma de desarrollo y administración de código abierto (Open Source) para PostgreSQL, la más avanzada base de datos de código abierto en el mundo. La aplicación puede ser usada en Linux y Windows para administrar PostgreSQL 7.3 y anteriores corriendo en cualquier plataforma. PgAdmin está diseñado para responder a las necesidades de todos los usuarios, desde la escritura de simples consultas SQL hasta el desarrollo de bases de datos complejas. La interfaz gráfica soporta todas las funcionalidades de PostgreSQL y hace fácil la administración. La conexión con el servidor debe hacerse a través de TCP / IP, y debe ser encriptado SSL para la seguridad. No son necesarios conductores adicionales para comunicarse con el servidor de base de datos. PgAdmin es desarrollado por una comunidad de expertos de PostgreSQL en todo el mundo y está disponible en más de una docena de idiomas. Es un software libre publicado bajo la licencia de PostgreSQL (PostgreSQL, 2011).

Herramienta PostgreSQL.

PostgreSQL es un poderoso sistema de base de datos objeto-relacional de código abierto. Cuenta con más de 15 años de desarrollo activo y una arquitectura probada que se ha ganado una sólida reputación de confiabilidad, integridad de datos y corrección. Funciona en todos los principales sistemas operativos, incluyendo Linux, y Windows. Tiene soporte completo para claves foráneas, uniones, vistas, disparadores y procedimientos almacenados (en varios idiomas). Se incluye la mayoría de SQL: 2008 tipos de datos, incluyendo INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, y TIMESTAMP. También es compatible con el almacenamiento de objetos binarios, incluyendo imágenes, sonidos o vídeo. Tiene interfaces de programación nativo de C/C++,

Java, .NET, entre otros, y la documentación de carácter excepcional. Una base de datos de clase empresarial, PostgreSQL cuenta con sofisticadas funciones como la versión Multi-Control de Concurrencia (MVCC), punto en el tiempo de recuperación, replicación asincrónica, transacciones anidadas (puntos de retorno) en línea, un planificador de consultas sofisticadas, optimizador, y escribe por delante del registro para la tolerancia a fallos. Es compatible con conjuntos de caracteres internacionales, codificación de caracteres multibyte, y es consciente de la configuración regional para la clasificación, sensibilidad de mayúsculas y minúsculas, y el formato. Es altamente escalable, tanto en la enorme cantidad de datos que puede manejar como en el número de usuarios concurrentes que puede acomodar. No hay sistemas activos de PostgreSQL en entornos de producción que manejen en exceso de 4 terabytes de datos.

PostgreSQL se enorgullece del cumplimiento de las normas. Tiene soporte completo para sub-consultas, lectura de los niveles de aislamiento comprometidos y transacciones serializables. Y mientras que PostgreSQL tiene un catálogo de sistema completamente relacional que a su vez apoya varios esquemas por base de datos, su catálogo es también accesible a través de la información de esquema de cómo se define en el estándar SQL (PostgreSQL, 2011).

1.3.2) Herramientas de Programación.

Entre las herramientas de programación que utilizan Java una de las más indicadas para la realización de este proyecto es Netbeans, porque es una plataforma modular y extensible usada como una estructura de integración para crear aplicaciones. Además de que ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación.

Herramienta NetBeans IDEes un entorno de desarrollo -una herramienta para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java -pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el NetBeans IDE. Es un producto libre y gratuito sin restricciones de uso. NetBeans es un proyecto de código abierto de gran éxito con una gran base de usuarios La plataforma

NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos.

Framework

Se utiliza hibernate y spring porque brinda implementaciones DAO ofreciendo diversas utilidades para acceder a la sesión de Hibernate, lo que simplifica en gran medida el trabajo.

Tecnología Hibernate.

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 récord 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 la reciente integración dentro del grupo JBoss que seguramente generará iniciativas muy interesantes para el uso de Hibernate dentro de este servidor de aplicaciones (Hibernate,2011).

Características de la tecnología Hibernate:

1. No intrusivo (estilo POJO)
2. Muy buena documentación (forums para ayuda, libro)
3. Comunidad activa con muchos usuarios.
4. Transacciones, caché, asociaciones, polimorfismo, herencia, persistencia,
5. Transitiva.

6. Potente lenguaje de consulta (HQL):
7. Fácil testeo.
8. No es estándar.

Tecnología Spring

Es el más popular de estos super-frameworks Java. Nos proporciona varios módulos los cuales abarcan la mayor parte de las cosas que debemos hacer en cualquiera de las capas de nuestras aplicaciones, desde plantillas para trabajar con JDBC o invocación de Web Services, pasando por sus propias soluciones, ORM o MVC (web), hasta integración con otros frameworks. Todo esto de una forma elegante y haciendo uso de muchos buenos principios de programación. Además Spring maneja la infraestructura de la aplicación, por lo que nosotros solo deberemos preocuparnos de la lógica de la misma (y de la configuración de Spring).

El núcleo de Spring está basado en un principio o patrón de diseño llamado Inversión de Control (IoC por sus siglas en inglés). Las aplicaciones que usan el principio de IoC se basan en su configuración (que en este caso puede ser en archivos XML o con anotaciones como en Hibernate) para describir las dependencias entre sus componentes, esto es, los otros objetos con los que interactúa. En este caso "inversión" significa que la aplicación no controla su estructura; permite que sea el framework de IoC (en este caso Spring) quien lo haga (Motores de Persistencia).

Spring está dividido en alrededor de 20 módulos y colocados en los siguientes grupos:

1. Contenedor Central
2. Acceso a Datos / Integración
3. WEB
4. AOP (Programación Orientada a Aspectos)
5. Instrumentación
6. Pruebas

Metodologías

RUP (Rational Unified Procces).

MSF (Microsoft Solution Framework).

Metodologías Ligeras/Ágiles

Esta metodología está orientada a la interacción con el cliente y el desarrollo incremental del software, mostrando versiones parcialmente funcionales del software al cliente en intervalos cortos de tiempo, para que pueda evaluar y sugerir cambios en el producto según se va desarrollando (I, 2011).

Ejemplo de metodologías Ligeras/Agiles:

XP (Extreme Programming).

SXP

Scrum.

1.3.3) Metodología a utilizar

SXP es una metodología de desarrollo de software compuesta por las metodologías SCRUM y XP que ofrece 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 fomentando 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. SCRUM es una forma de gestionar un equipo para que trabaje eficientemente y tenga siempre medidos los progresos. XP más bien es una metodología encaminada para el desarrollo; consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar el éxito del proyecto.

1.3.4) Modelo de acceso a datos

En el modelo de acceso a datos una capa es un nivel lógico en el cual residen componentes o aplicaciones lógicas. Las capas pueden residir en uno a mas equipos o servicios, el numero de capas hace referencia al numero d niveles y no al

numero de equipos en los cuales los servicios son divididos. Las capas que generalmente se incluyen en aplicaciones son:

Capa Cliente

Conocida como capa de presentación es la que contiene las interfaces en las que el usuario interactúa con el sistema.

Capa de la lógica de Negocio

Contiene la lógica que interactúa con el origen de datos. Esta capa intermedia contiene la parte de la aplicación que interactúa con los datos.

1.4.) Conclusiones Parciales.

En este capítulo se realizó una breve reseña del estado del arte del tema a tratar así como también se analizaron los diferentes conceptos necesarios para entender la problemática planteada.

CAPÍTULO 2: Diseño y Arquitectura de la Base de Datos.

Introducción

En el presente capítulo se describen los procedimientos necesarios para la creación de la capa de acceso a datos entre ellos se puede encontrar la descripción de los patrones de diseño, los artefactos correspondientes al rol diseñador de datos, el modelo entidad relación, los requisitos pertenecientes a cada dirección, así como la transformación del modelo entidad relación al modelo relacional.

2.) Diseño de la Base de Datos.

La planificación de la estructura de la base de datos, en particular de las tablas, es vital para la gestión efectiva de la misma. El diseño de la BD consiste en una descripción de cada una de las tablas, así como de los campos que las componen, el registro y los valores que contendrá cada uno de esos campos.

Son muchas las consideraciones a tomar en cuenta al momento de hacer el diseño de la base de datos, quizá las más fuertes sean:

La velocidad de acceso.

El tamaño de la información.

El tipo de la información.

Facilidad de acceso a la información.

Facilidad para extraer la información requerida.

La actividad de diseñar las bases de datos tiene como propósito fundamental asegurar que los datos persistentes sean almacenados consistentemente de manera eficiente y definir el comportamiento que debe ser implementado.

Modelo lógico de los datos

Para una mejor comprensión del diseño de la base datos propuesta se describirá en este epígrafe el modelo lógico de los datos para las dos direcciones, en el que se ilustran las entidades lógica clave y sus relaciones que son necesarias para satisfacer los requisitos del sistema para datos persistentes que sean coherentes con

la arquitectura de la aplicación. Antes de desarrollar este modelo, para crear el diseño físico de la base de datos, se ha perfeccionado aplicando las reglas estándar de normalización.

En la figura 3,4 y 5 se modelan las tablas que almacenan información relacionada con los cuadros de la empresa, las entidades subordinadas, las medidas disciplinarias, los dirigentes y los trabajadores.

Las denominaciones son por cargos y los municipios son por código.

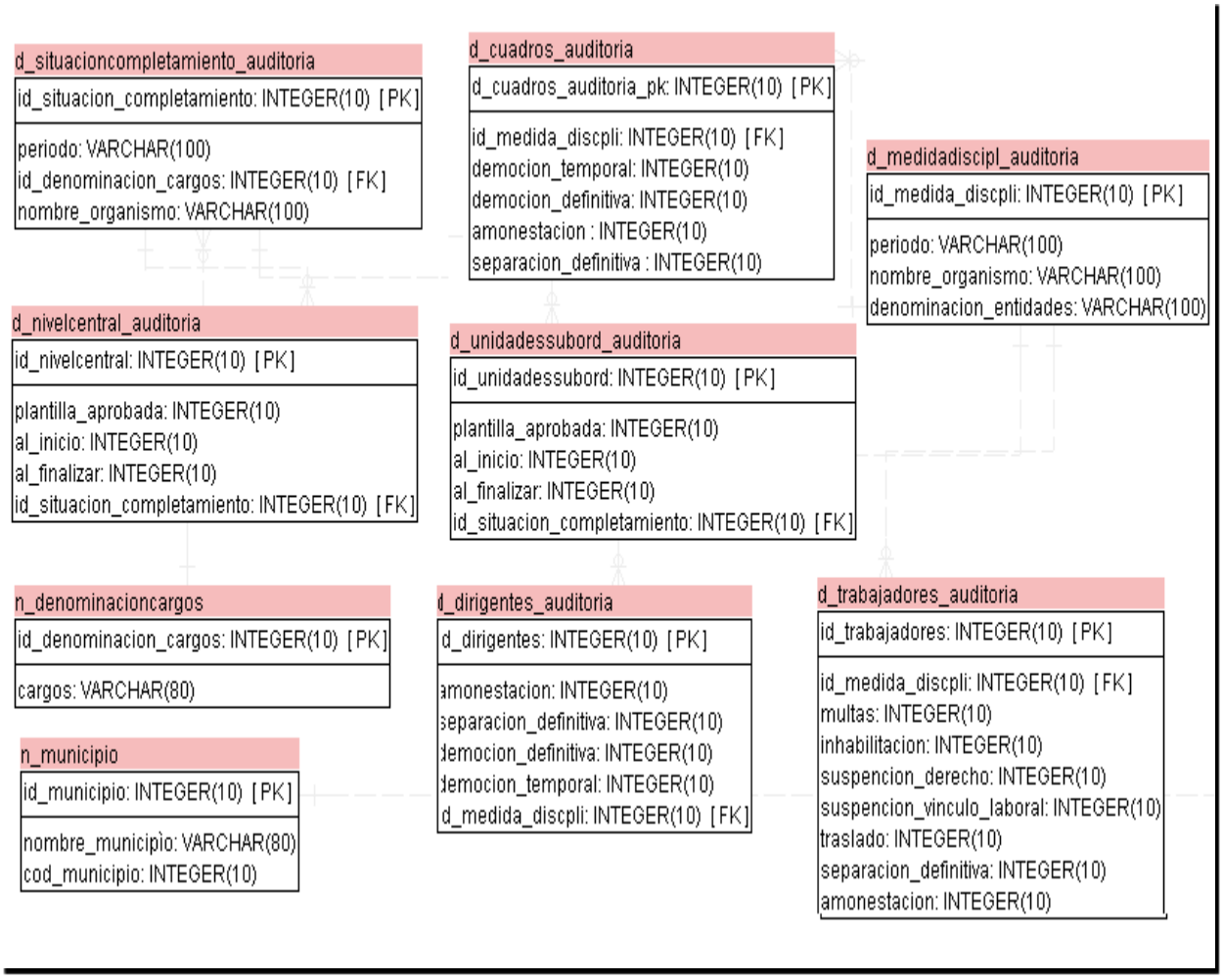


Figura 3: Modelo de datos de Auditoría parte 1

Modelo lógico de los datos para la dirección de Auditoría

En la figura se modelan las tablas con la información relacionada con las deficiencias de los controles realizados, los resultados del perfeccionamiento y las particularidades.

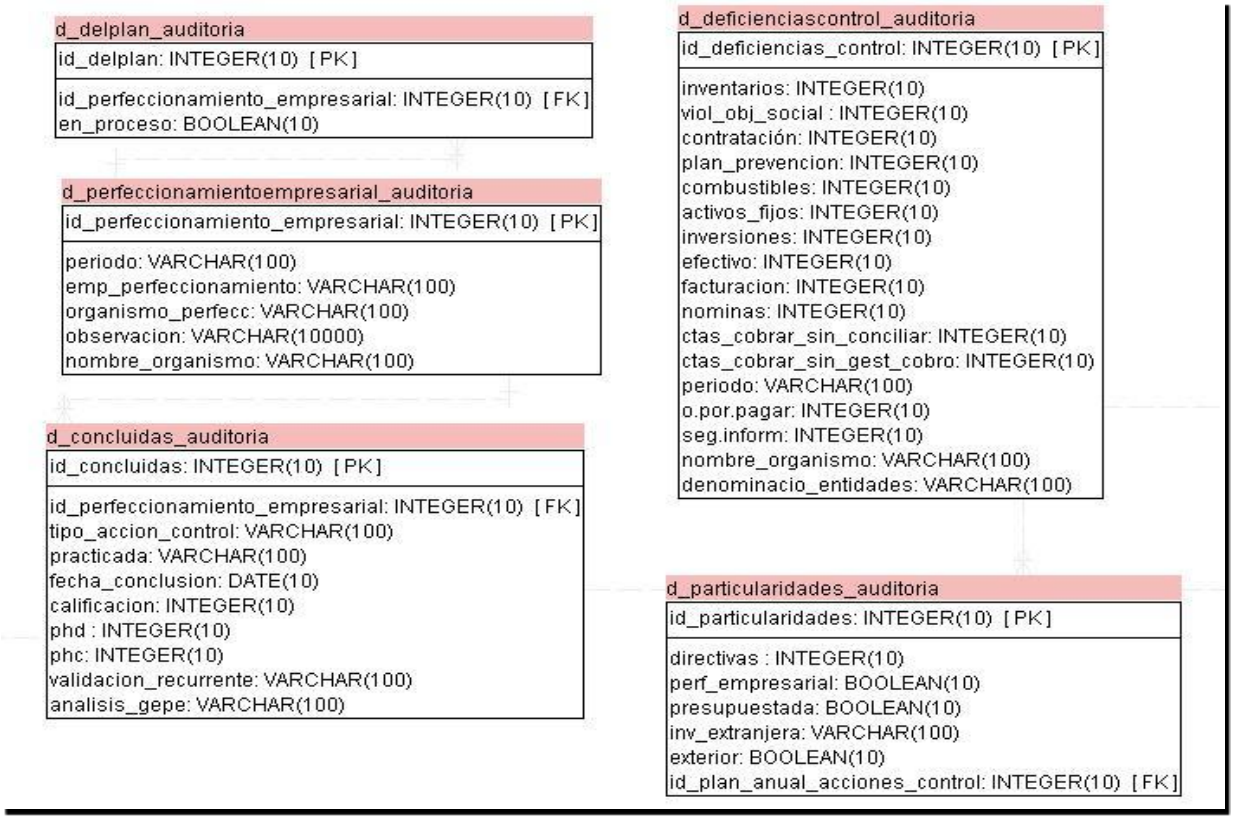


Figura 4: Modelo de datos de Auditoría parte 2

En la figura se modela las tablas con la información relacionada con los hechos delictivos en la provincia, los controles de cumplimientos, los planes por año, los reportes emitidos por cada uno de los resultados y las planificaciones.

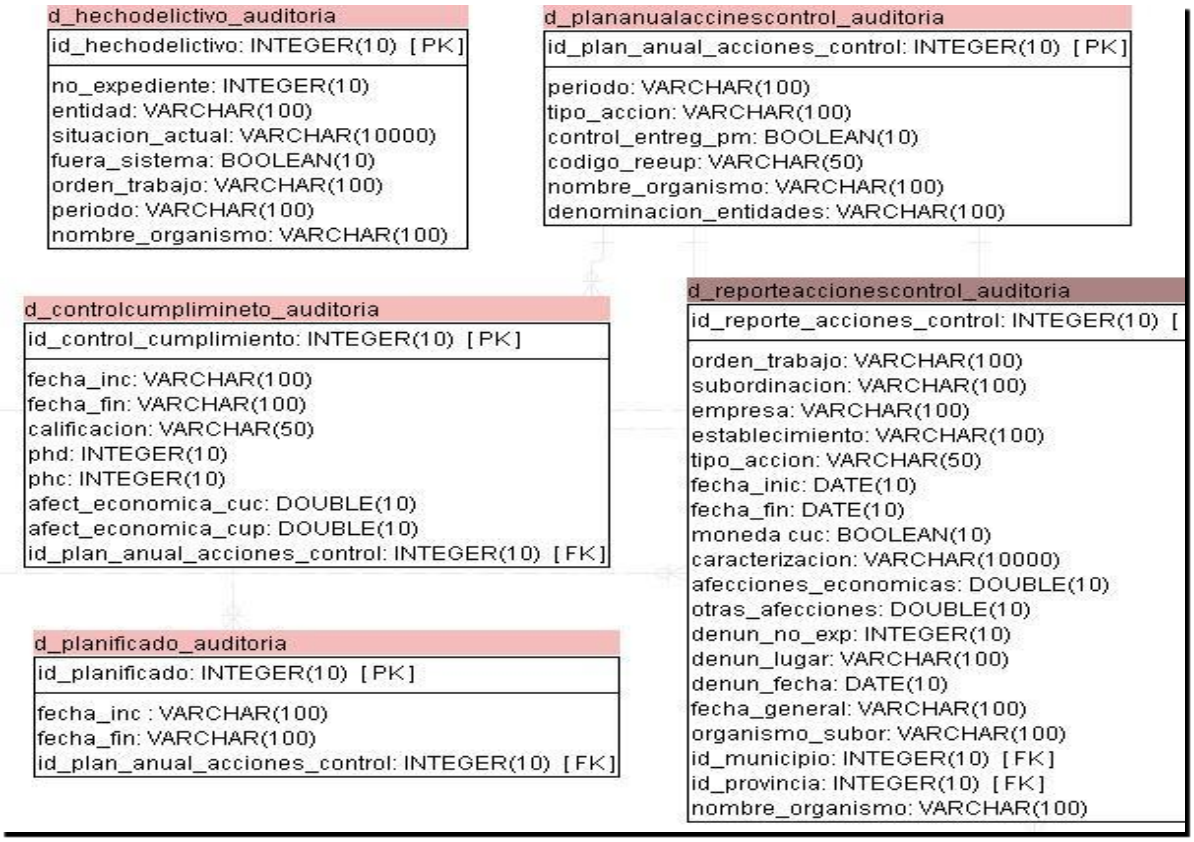


Figura 5: Modelo de datos de Auditoría parte 3

Modelo de datos para la Dirección de Justicia

En la figura se modelan las tablas con la información relacionada con el reporte de consultoría jurídica y sus distintos aspectos como pueden ser los asuntos tramitados en la provincia, los indicadores generales, los contratos de servicio y los asesores propios.

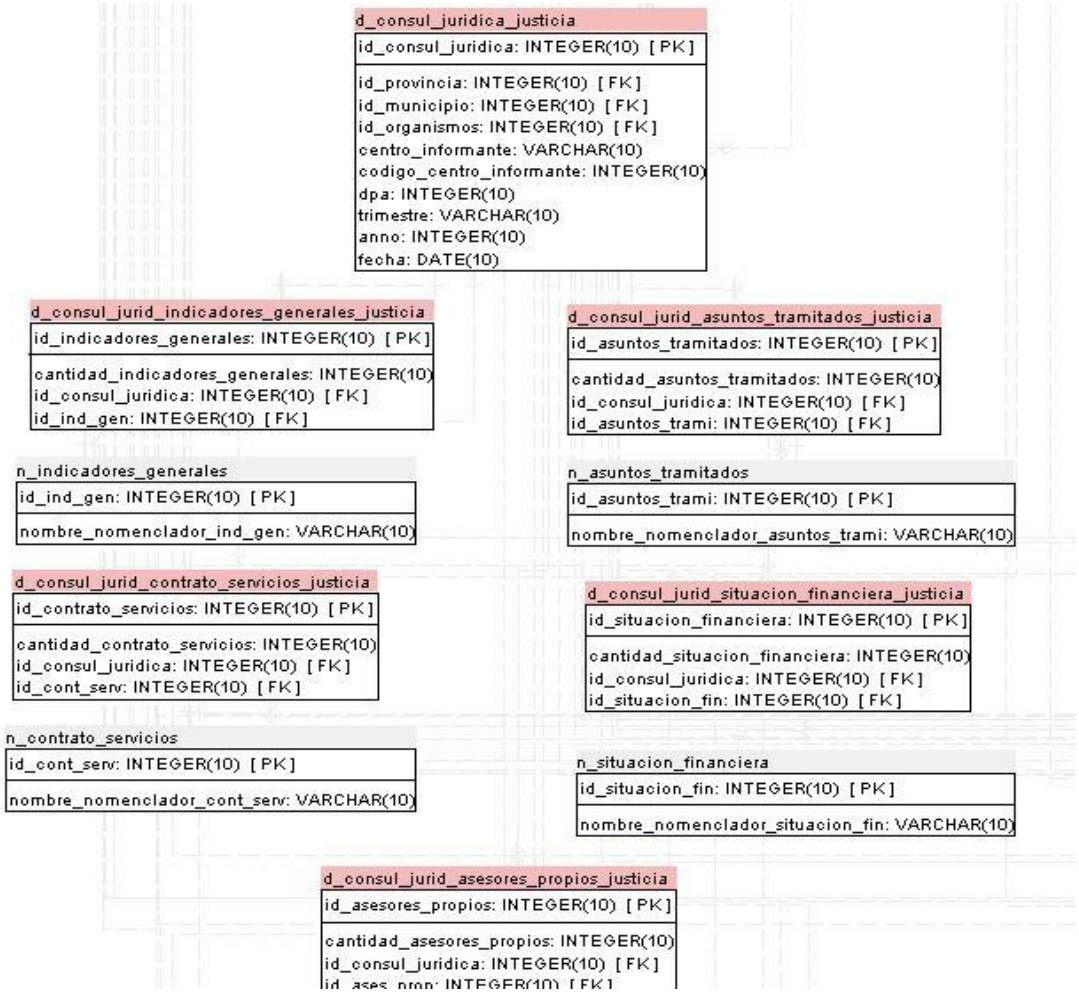


Figura 6: Modelo de datos de Justicia parte 1

En la figura se modelan las tablas con la información relacionada con el reporte de registro civil y sus distintos aspectos como pueden ser los matrimonios en la provincia, nacimientos, actos búsquedas y expedientes.

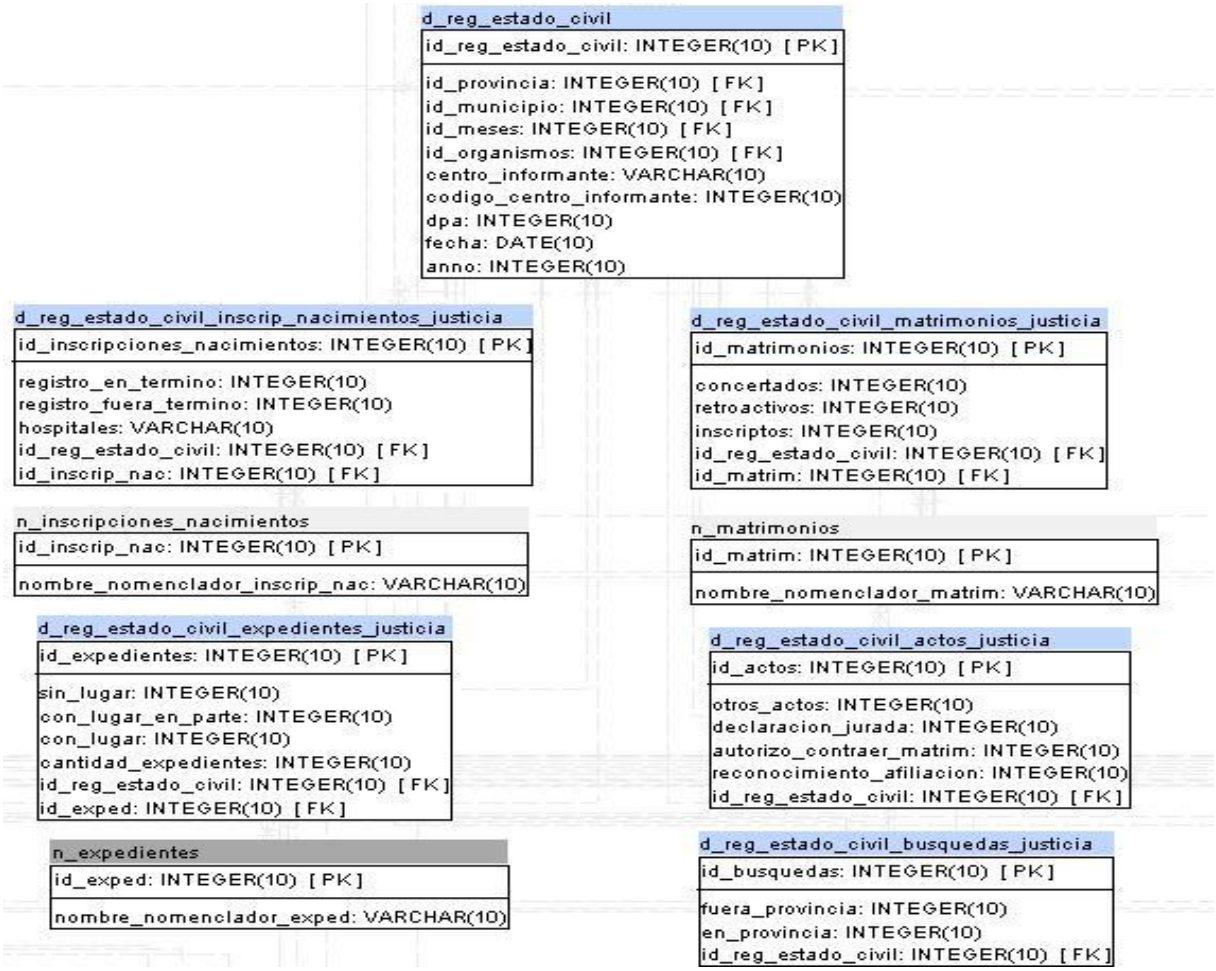


Figura 7: Modelo de datos de Justicia parte 2

En la figura se modelan las tablas con la información relacionada con el reporte de servicios de la computación y sus distintos aspectos como pueden ser los equipamiento, equipamiento roto, entidades y servidores.

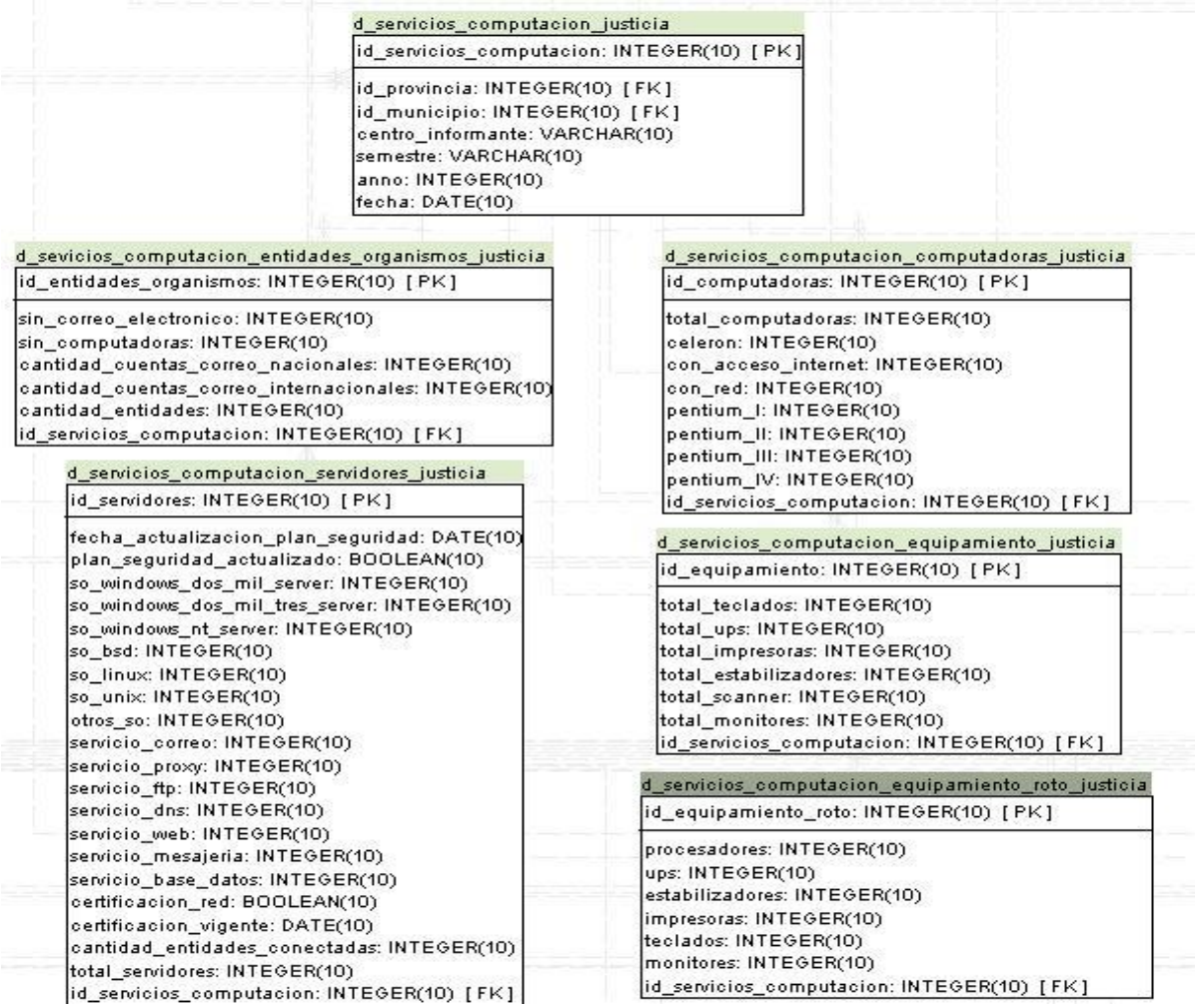


Figura 8: Modelo de datos de Justicia parte 3

2.1) Patrones de Diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Para que una solución sea considerada un patrón debe poseer ciertas características, una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias. (Guzmán, 2011).

2.1.1) Patrones de diseño a utilizar

La mayoría de las aplicaciones, tienen que persistir datos en algún momento, ya sea guardándolos en una base de datos relacional, o en una base de datos orien-

tada a objetos. Para hacer esto, la aplicación interactúa con la base de datos, esta interacción no tiene que ver con la capa lógica de negocio de la aplicación, ya que para ello está la capa de persistencia. Siendo así se puede decir que DAO es el patrón de diseño que se va a utilizar para crear esta capa de persistencia. Este consiste básicamente en una clase que es la que interactúa con la base de datos.

Los métodos de esta clase dependen de la aplicación y de lo que se logre y se quiera hacer, pero generalmente se implementan los métodos CRUD para realizar las operaciones básicas de una base de datos.

2.2) Modelo de Entidad-Relación

El MER está basado en una percepción del mundo real que consta de un conjunto de objetos básicos llamados entidades con sus atributos y de las interrelaciones que existen entre estos objetos. Se desarrolló para facilitar el diseño de Bases de Datos permitiendo la especificación de un esquema del universo de discurso que representa la estructura completa de una Base de Datos.

El MER es uno de los diferentes modelos de datos semánticos que existe; el aspecto semántico del modelo reside en su intento de representar el significado de los datos. Este modelo es extremadamente útil para hacer corresponder los significados e interacciones del desarrollo del mundo real con un esquema conceptual. Los esquemas de MER usan diagramas para representar la estructura natural de los datos, que se nombran Diagrama Entidad Relación (DER). En esos diagramas los rectángulos representan a las entidades y los rombos representan a las interrelaciones. Las interrelaciones son enlazadas con sus entidades constitutivas por arcos, y el grado de la interrelación es indicado en el arco.

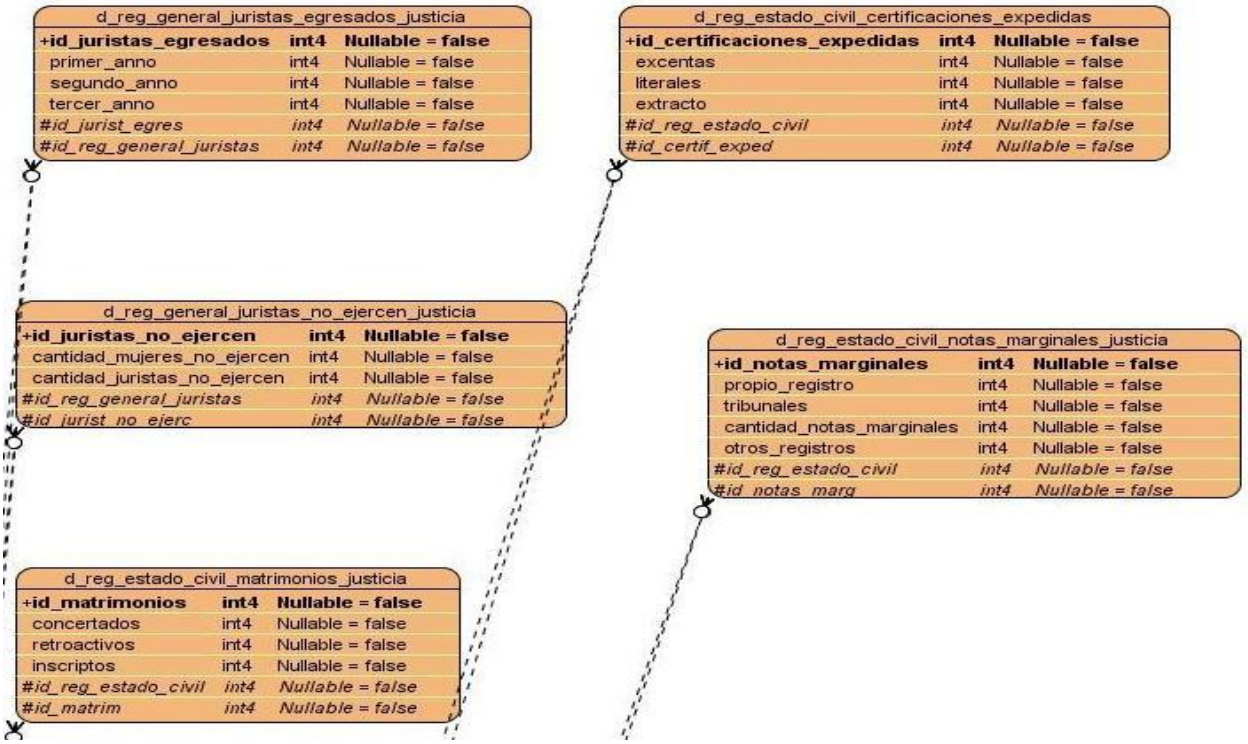


Figura 9: Diagrama de Entidad-Relación de Justicia parte 1

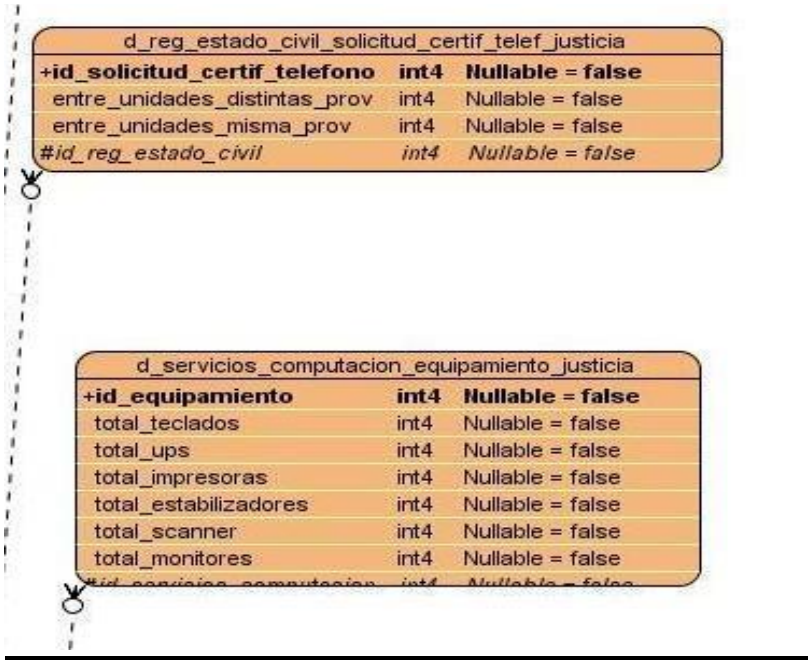


Figura 10: Diagrama de Entidad-Relación de Justicia parte 2

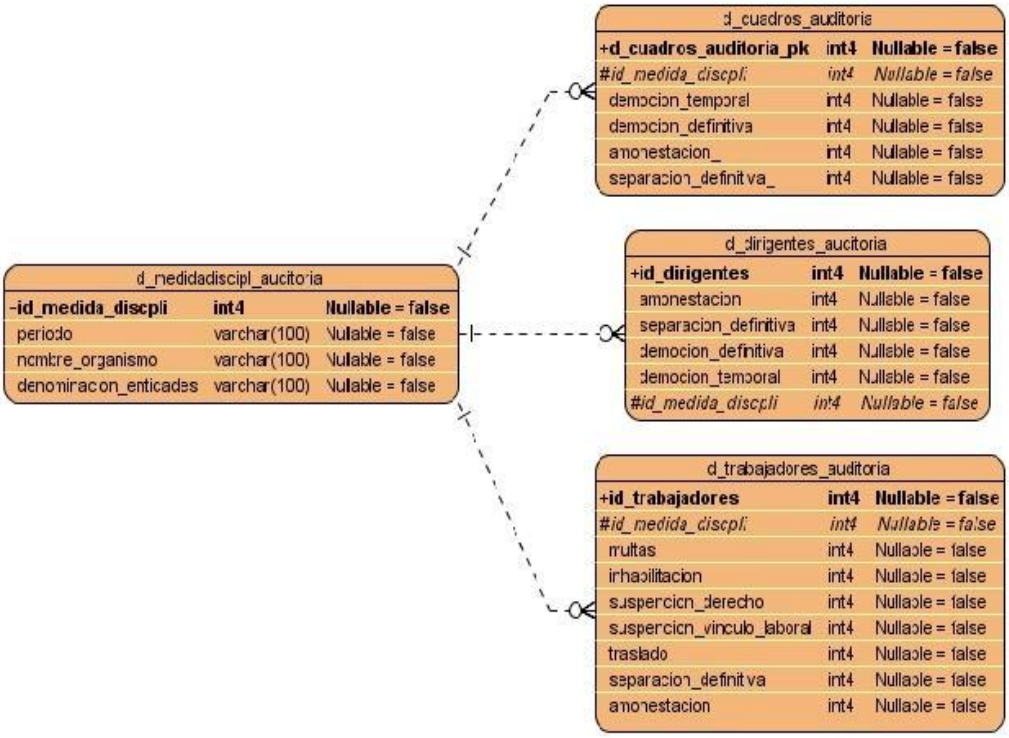


Figura 11: Diagrama de Entidad-Relación Auditoría parte 1

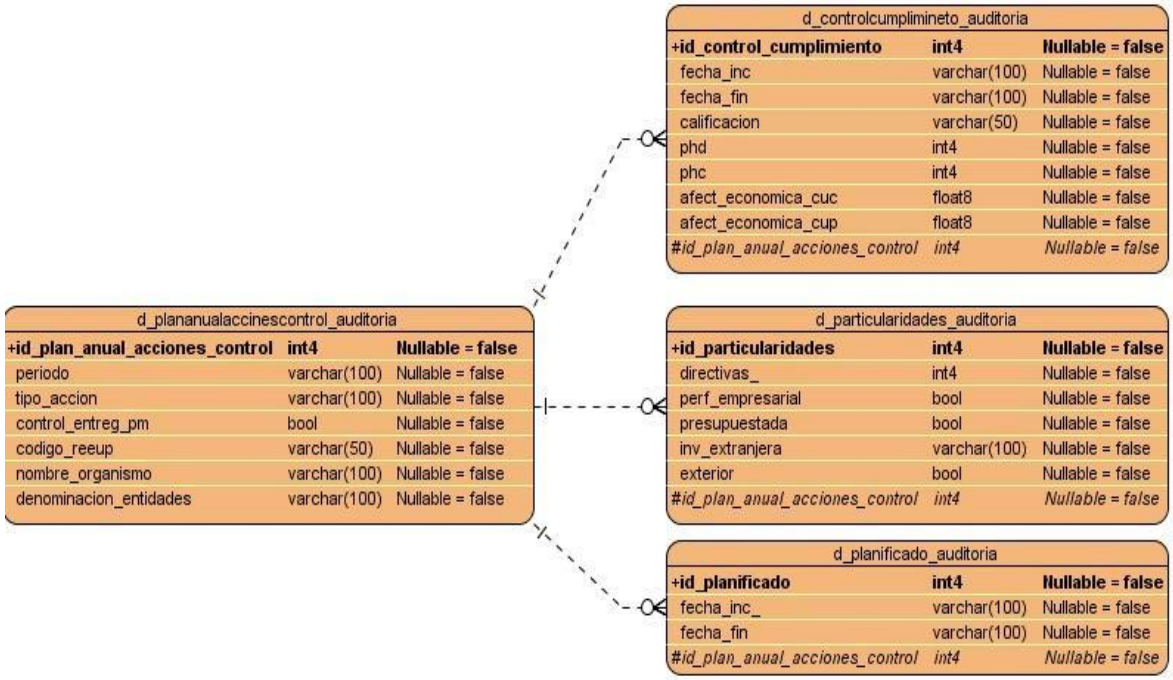


Figura 12: Diagrama de Entidad-Relación Auditoría parte 2

2.3) Descripción de las tablas.

2.3.1.) Descripción de las tablas para la dirección Auditoria.

Clase	d_deficienciasdetectadas_auditoria			
Descripción:	Clase que describe las principales deficiencias detectadas en esta dirección			
Tipo	Superclase			
Roles o Propiedades	**Descripción**	**Valor requerido**	**Llave**	**Relación**
id_deficiencia_detectadas	Número consecutivo que mostrara la numeración de las deficiencias por entidades controladas	Int	X	
periodo	Muestra el período en que se estáá analizando dicho modelo	String		
id_organismo	Muestra el id del organismo al que pertenece este modelo	Int		

Tabla 1: Descripción de la tabla d_deficienciasdetectadas_auditoría

Clase	d_entidadcontrolada_auditoria			
Descripción:	Clase que describe los parámetros que deben tener la entidad a controlar			
Tipo				
Roles o Propiedades	**Descripción**	**Valor requerido**	**Llave**	**Relación**
id_deficiencia_detectadas	Número consecutivo que mostrara la numeración de las deficiencias por entidades controladas	Int	X	
inventarios	Número de inventario	Int		

viol_obj_social	Número de viol_obj_social	Int		
contratación	Número de contratación	Int		
plan_prevención	Número de plan_prevención	Int		
combustibles	Número de combustible	Int		
activos_fijos	Número de activos_fijos	Int		
inversiones	Número de inversiones	Int		
efectivo	Número de efectivo	Int		
facturación	Número de facturación	Int		
nominas	Número de nominas	Int		
ctas_cobrar_sin_conciliar	Número de ctas_cobrar_sin_conciliar	Int		
ctas_cobrar_sin_gest_cobro	Número de ctas_cobrar_sin_gest_cobro			
id_defidetec	Identificador del nomenclador de la tabla deficiencias detectadas	Int		n_deficienciasdetectadas

Tabla 2: Descripción de la tabla d_entidadcontrolada_auditoría

Clase	n_deficienciasdetectadas
Descripción:	Clase que nomencladora de la clase Deficiencias Detectadas se muestra el nombre de las entidades
Tipo	

Roles o Propiedades	Descripción	Valor requerido	Llave	Relación
id_defidetec	Número consecutivo que mostrara la numeración de las entidades	Int	X	
nombre_entidad	Muestra el nombre de las entidades	String		

Tabla 3: Descripción de la tabla n_deficienciasdetectadas

Clase	d_perfeccionamientoempresarial_auditoria			
Descripción:	Clase que describe los requisitos de las empresas en perfeccionamiento empresarial			
Tipo	Superclase			
Roles o Propiedades	**Descripción**	**Valor requerido**	**Llave**	**Relación**
id_perfeccionamiento_empresarial	Número consecutivo que mostrara la numeración de los modelos	Int	X	
fecha	Muestra la fecha en que se elaboró el modelo	Date		
emp_perfeccionamiento	Muestra la empresa a la que se le está haciendo el perfeccionamiento	Varchar		
id_organismo	Muestra el organismo al que pertenece	Int		n_organismo

Tabla 4: Descripción de la tabla d_perfeccionamientoempresarial_auditoria

Clase	d_empresaenperfeccionamiento_auditoria

Descripción:	Clase que muestra las observaciones			
Tipo	Superclase			
Roles o Propiedades	**Descripción**	**Valor requerido**	**Llave**	**Relación**
no	Número consecutivo que mostrara las observaciones	Int	X	
observaciones	Muestra las descripción de las observaciones	String		

Tabla 5: Descripción de la tabla d_empresaenperfeccionamiento_auditoria

2.3.2.) Descripción de las tablas de la dirección Justicia.

Clase	d_control_libros_justicia			
Descripción:	Clase que muestra el control por cada municipio de los nacimientos, fallecidos etc.			
Tipo	Superclase			
Roles o Propiedades	**Descripción**	**Valor requerido**	**Llave**	**Relación**
id_control_libros	Número consecutivo que mostrara el id de los libros	Int	X	
trimestre	Muestra el trimestre en se realizó este análisis	String		
encuadernados_forrados	Muestra la cantidad de encuadernados y forrados que existen	Int		
reconstruidos	Muestra la cantidad de reconstruidos que existen	Int		
transcriptos	Muestra la cantidad de	Int		

	transcriptos existentes			
tipo_libro	Muestra la cantidad de cada tipo de libros	Int		
ciudadanía		Int		
id_municipio	Muestra los municipios	Int		n_municipio
total_libro_nacim	Muestra el total de cada tipo de libros que existen	Int		
total_libros_matrim	Muestra el total de cada tipo de libros que existen	Int		
total_libros_defuc	Muestra el total de cada tipo de libros que existen	Int		
total_libro_cuidad	Muestra el total de cada tipo de libros que existen	Int		
libros_blanco_nacim	Muestra el total de cada tipo de libros que existen en blanco	Int		
libros_blanco_matrim	Muestra el total de cada tipo de libros que existen en blanco	Int		
libros_blanco_defuc	Muestra el total de cada tipo de libros que existen en blanco	Int		
pendiente_encuad	Muestra el total de los que están pendientes por encuadernar	Int		
pendiente_transc	Muestra el total de los que están pendientes por transcribir	Int		
pendientes_reconst	Muestra el total de los que están pendientes por reconstruir	Int		
anno	Muestra el año para el que se realizó este modelo	Int		

Tabla 6: Descripción de la tabla d_control_libros_justicia

Clase	d_regciv_incripciones_defuncion_justicia			
Descripción:	Clase que muestra el registro de inscripciones defunción			
Tipo	Superclase			
Roles o Propiedades	**Descripción**	**Valor requerido**	**Llave**	**Relación**
id_regciv_inscripciones_defusion	Número consecutivo que mostrara el id de cada descripción	Int	X	
centro_informante	Muestra el centro que da la información	String		
codigo_centro_informante	Muestra el código del centro	Int		
fecha	Muestra la fecha en la que se realiza el modelo	Date		
dpa		Int		
id_indicador	Muestra los indicadores pertenecientes a esa modelo	Int		n_indicadores
id_municipio	Muestra los municipios pertenecientes al modelo	Int		n_municipio
id_organismos_	Muestra los organismos pertenecientes al modelo	Int		n_organismo
id_provincia	Muestra las provincias pertenecientes al modelo	Int		n_provincia
cantidad		Int		

Tabla 7: Descripción de la tabla d_regciv_incripciones_defuncion_justicia

Clase	d_regciv_ind_nacimientos_justicia
Descripción:	Clase que muestra en el registro civil los nacimientos

Tipo	Superclase			
Roles o Propiedades	**Descripción**	**Valor requerido**	**Llave**	**Relación**
id_regciv_ind_nacimientos	Número consecutivo que mostrara el id del cada nacimiento	Int	X	
centro_informante	Muestra el centro que da la información	String		
codigo_centro_informante	Muestra el código del centro	Int		
fecha	Muestra la fecha en la que se realiza el modelo	Date		
dpa		Int		
id_indicador	Muestra los indicadores pertenecientes a esa modelo	Int		n_indicador
id_municipio	Muestra los municipios pertenecientes al modelo	Int		n_municipio
id_organismos_	Muestra los organismos pertenecientes al modelo	Int		n_organismo
id_provincia	Muestra las provincias pertenecientes al modelo	Int		n_provincia
hospitales		Int		
registro_dentro_termino		Int		
registro_fuera_termino		Int		

Tabla 8: Descripción de la tabla d_regciv_ind_nacimientos_justicia

Clase	d_revision_penal_justicia			
Descripción:	Clase que muestra las revisiones penales			
Tipo	Superclase			
Roles o Propiedades	**Descripción**	**Valor requerido**	**Llave**	**Relación**
id_revision_penal	Número consecutivo que mostrara el id de la revisión	Int	X	
sub_indicador		String		
id_indicador	Muestra los indicadores pertenecientes a esa modelo	Int		n_indicador
fecha_	Muestra la fecha en que se realizo	Date		
id_organismos_	Muestra los organismos pertenecientes al modelo	Int		n_organismo
id_municipio	Muestra los municipios pertenecientes al modelo	Int		n_municipio
cantidad		Int		

Tabla 9: Descripción de la tabla d_revision_penal_justicia

Clase	d_consult_jurid_asuntos_tramitados_justica			
Descripción:	Clase que muestra los asuntos tramitados en el dpto. de consultoría jurídica			
Tipo	Superclase			
Roles o Propiedades	**Descripción**	**Valor requerido**	**Llave**	**Relación**
id_consult_jurid_asuntos_tra	Número consecutivo que	Int	X	

mitados	mostrara el id del modelo			
fecha	Muestra fecha del modelo	Date		
cant_asuntos_tramitados		Int		
trimestre	Muestra el trimestre se realizó el modelo	String		
id_municipio	Muestra los municipios pertenecientes al modelo	Int		n_municipio
id_organismos_	Muestra los organismos pertenecientes al modelo	Int		n_organismo
id_indicador	Muestra los indicadores pertenecientes a esa modelo	Int		n_indicador
id_provincia	Muestra las provincias pertenecientes al modelo	Int		n_provincia

Tabla 10: Descripción de la tabla d_consult_jurid_asuntos_tramitados_justicia

2.4) Requisitos Funcionales y no Funcionales.

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir.

Los no funcionales son propiedades o cualidades que el producto debe tener, permitiendo que el mismo sea atractivo, usable, rápido y confiable.

Es de vital importancia para la creación de un sistema informático llevar a cabo una correcta realización de requisitos para llegar a acuerdos positivos entre los usuarios finales y los desarrolladores del sistema, asegurando además la calidad del producto.

Requisitos no Funcionales

1. La aplicación web debe contar con una interfaz profesional, siguiendo una arquitectura de información, permitiendo que los usuarios finales de la misma sean capaces de interactuar con esta, aún cuando solo posean conocimientos básicos en el manejo de las computadoras.
2. La aplicación podrá ser utilizada por personas que tengan un conocimiento básico en el manejo de las computadoras.
3. Las PC clientes, deberán estar equipadas con: tarjeta de red, con una capacidad mínima de 256 MB de RAM, un disco duro de 60 GB como mínimo.
4. El servidor deberá estar equipado con: tarjeta de red, el servidor de base de datos debe tener 1GB de RAM y 60 GB de disco duro como mínimo, el servidor web debe tener 1GB de RAM.
5. Tener un navegador compatible o superior con Chromium, Chrome 4, Opera 11, Firefox 4, o Safari 5.
6. Necesita un servidor de bases de datos que soporte una capacidad alta de datos.
7. Crear diferentes cuentas de usuario y asignarle a cada una solo los permisos que le corresponden.
8. Mostrar a cada usuario sólo las funcionalidades de la aplicación sobre las cuales tiene permiso de acceso.
9. Ofrecer mensajes de verificación antes de ejecutar acciones irreversibles (eliminación de datos).
10. El servidor donde se encuentre instalada la aplicación debe estar situado en un local protegido, donde no esté expuesto a desastres naturales o robo.
11. La aplicación está diseñada para su funcionamiento constante, permitiendo el acceso a los servicios que brinda durante los siete días de la semana y 24 horas del día.

Requisitos Funcionales

Requisitos funcionales pertenecientes a la dirección de Justicia.

RF1 Insertar información sobre el Registro general de juristas.

RF2 Modificar información sobre el Registro general de juristas.

RF3 Eliminar información sobre el Registro general de juristas.

RF4 Buscar información sobre el Registro general de juristas.

RF5 Insertar información sobre el Registro de Potencial científico.

RF6 Modificar información sobre el Registro de Potencial científico.

RF7 Eliminar información sobre el Registro de Potencial científico.

RF8 Buscar información sobre el Registro de Potencial científico.

RF9 Insertar información sobre el Registro mercantil para el perfeccionamiento empresarial.

RF10 Modificar información sobre el Registro mercantil para el perfeccionamiento empresarial.

RF11 Eliminar información sobre el Registro mercantil para el perfeccionamiento empresarial.

RF12 Buscar información sobre el Registro mercantil para el perfeccionamiento empresarial.

RF13 Insertar información sobre el Registro de las Consultorías jurídicas.

RF14 Modificar información sobre el Registro de las Consultorías jurídicas.

RF15 Eliminar información sobre el Registro de las Consultorías jurídicas.

RF16 Buscar información sobre el Registro de las Consultorías jurídicas.

RF17 Insertar información sobre el Registro del Asesoramiento jurídico del sector agropecuario.

RF18 Modificar información sobre el Registro del Asesoramiento jurídico del sector

agropecuario.

RF19 Eliminar información sobre el Registro del Asesoramiento jurídico del sector agropecuario.

RF20 Buscar información sobre el Registro del Asesoramiento jurídico del sector agropecuario.

RF21 Insertar información sobre el Registro de Sistema de pago a los consultores.

RF22 Modificar información sobre el Registro de Sistema de pago a los consultores

RF23 Eliminar información sobre el Registro de Sistema de pago a los consultores.

RF24 Buscar información sobre el Registro de Sistema de pago a los consultores.

RF25 Insertar información sobre el Registro de la actividad notarial.

RF26 Modificar información sobre el Registro de la actividad notarial.

RF27 Eliminar información sobre el Registro de la actividad notarial.

RF28 Buscar información sobre el Registro de la actividad notarial.

Requisitos Funcionales pertenecientes a la dirección de Auditoria

RF1 Insertar información sobre la Situación de presuntos hechos delictivos.

RF2 Modificar información sobre la Situación de presuntos hechos delictivos

RF3 Eliminar información sobre la Situación de presuntos hechos delictivos

RF4 Insertar información sobre los reportes de la acción de control con presuntos hechos delictivos.

RF5 Modificar información sobre los reportes de la acción de control con presuntos hechos delictivos.

RF6 Eliminar información sobre los reportes de la acción de control con presuntos hechos delictivos.

RF7 Insertar información sobre el Resumen de las acciones de control.

RF8 Modificar información sobre el Resumen de las acciones de control.

RF9 Eliminar información sobre el Resumen de las acciones de control

RF10 Insertar información sobre el Análisis de la situación del completamiento de

la Plantilla.

RF11 Modificar información sobre el Análisis de la situación del completamiento de la Plantilla.

RF12 Eliminar información sobre el Análisis de la situación del completamiento de la Plantilla.

RF13 Insertar información sobre la aplicación de las medidas disciplinarias.

RF14 Modificar información sobre la aplicación de las medidas disciplinarias.

2.5) Diagrama de Clases Persistentes

Un diagrama de clase muestra un conjunto de clases, interfaces, y colaboraciones y sus relaciones entre ellos. Los diagramas de clase se usan en el diseño del modelo estático para ver un sistema. Para las demás partes, este modelado involucra el vocabulario del sistema, el modelado de colaboraciones, o modelado de esquemas. Los diagramas de clase son también la base para un par de diagramas relacionados: Diagramas de Componente y Diagramas de Instalación (Diplomen).Los diagramas de clase son importantes no solo para la visualización, especificación y documentación del modelo estructural, pero también para la construcción de sistemas ejecutables. Ingeniería hacia adelante e ingeniería inversa (Java Database Connectivity, 2005).

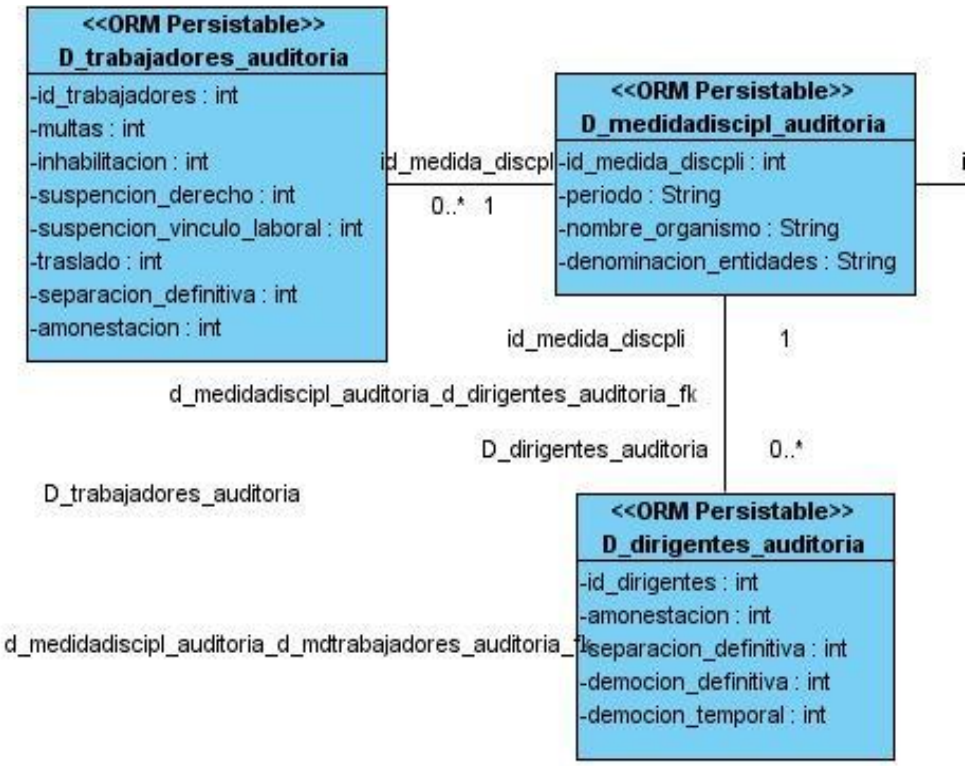


Figura 13: Diagrama de Persistencia Auditoría

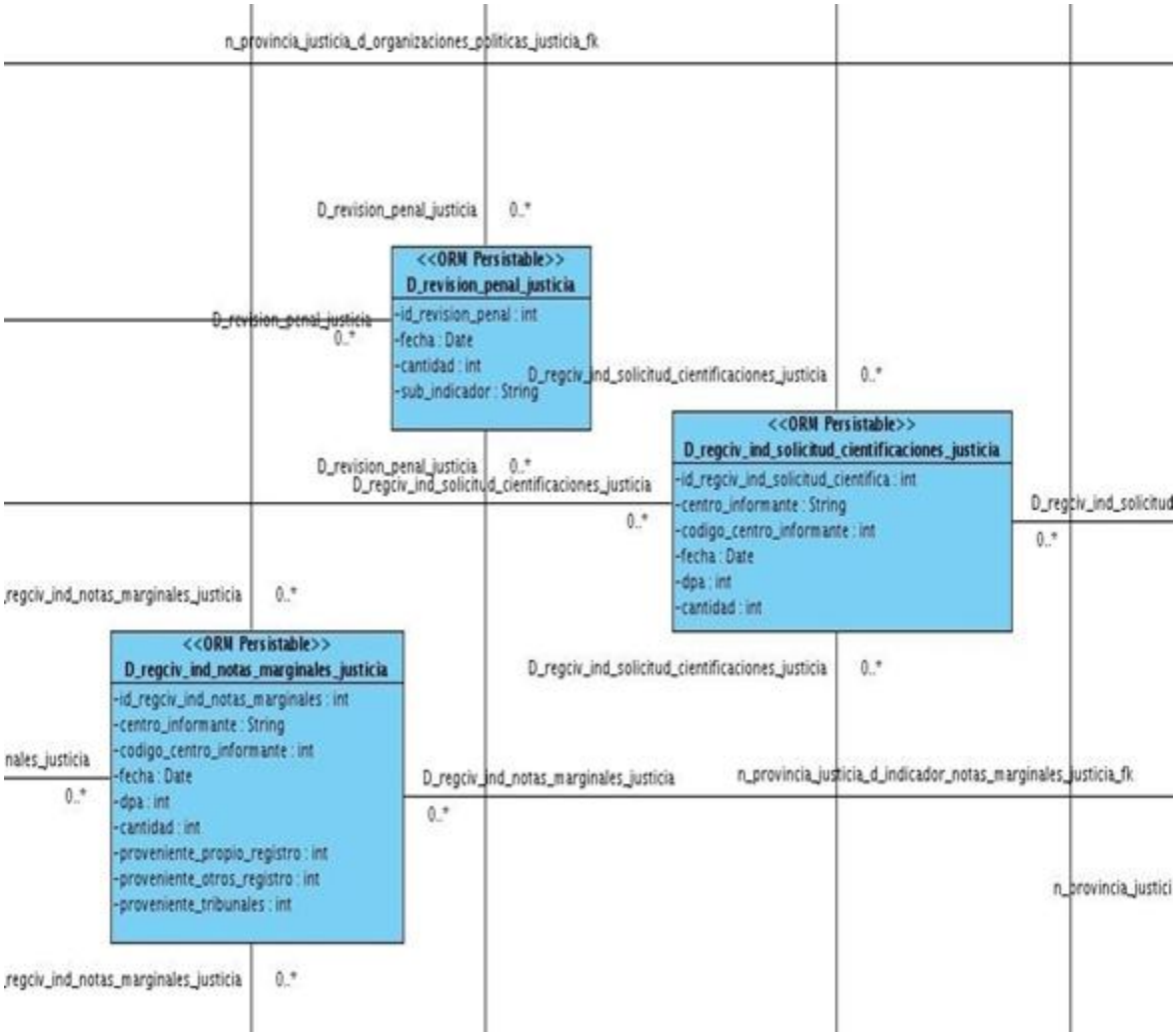


Figura 14: Diagrama de Persistencia Justicia

2.7) Conclusiones Parciales

En este capítulo se desarrolló de manera general la estructura de la capa de acceso a datos, los diagramas correspondientes al rol de Diseñador de la Base de Datos así como se generaron los artefactos correspondientes a dicho rol.

CAPÍTULO 3

CAPÍTULO 3: Implementación y Validación de la Base de datos.

Introducción

En el presente capítulo se hará referencia a la implementación y validación de la base de datos, teniendo en cuenta la descripción de la integridad y análisis de la redundancia de los datos respectivamente. Por último se realizarán pruebas que permitan comprobar la implementación propuesta.

3.) Aspectos a tener en cuenta antes de la validación e integración de los Datos.

Integridad

La integridad de una BD se refiere a la consistencia de los datos almacenados de acuerdo a un conjunto de restricciones semánticas.

Integridad de Entidad: Establece que la llave primaria de una tabla debe tener un valor único para cada fila de la tabla, sino la base de datos perderá su integridad. Para lograr satisfacer este parámetro se hace uso de secuencias, las cuales garantizan que en cada inserción en una tabla de la base de datos, la llave del registro insertado sea única.

Datos Requeridos: Establece que al realizar una operación de INSERT sobre una fila, una columna marcada como NOT NULL no debe recibir valores nulos.

Chequeo de Validez: Al crear una nueva tabla cada columna se describe con un tipo de datos y el SGBD garantiza que sólo los datos del tipo especificado sean ingresados en la tabla.

Integridad Referencial: Garantiza que una entidad (fila o registro) siempre se relaciona con otras entidades válidas, es decir, que existen en la base de datos. Implica que en todo momento dichos datos sean correctos, sin repeticiones innecesarias, datos perdidos y relaciones mal resueltas. Todas las bases de datos relacionales gozan de esta propiedad gracias a que el software gestor de base de datos vela por su cumplimiento.

Redundancia

El diseño de una BD debe ser realizado cuidadosamente, procurando permitir un fácil acceso a la información, facilitando un alto rendimiento, una buena velocidad de respuesta y una gran consistencia de los datos. Al diseñar una BD se debe tener en cuenta que la información almacenada ocupará irremediablemente un espacio en memoria; es de vital importancia eliminar la posibilidad de almacenar datos repetidos que podrían conducir a que exista inconsistencia en la información. A este almacenamiento de información repetida se le conoce como redundancia de la información. (Avendaño, 2009)

Un buen diseño de una base de datos logrará evitar la aparición de información repetida o redundante. De entrada, lo ideal es lograr una redundancia nula. En algunos casos la complejidad de los cálculos hace necesaria la aparición de redundancias que en ocasiones resulta mejor para ganar en simplicidad de las consultas y lograr un mejor tiempo de respuesta.

Seguridad

Las BD están bajo constante amenaza de sufrir ataques de personas o sistemas no autorizados que ponen en riesgo su integridad y confidencialidad; por eso deben tomarse una serie de medidas que impidan estos sucesos y permitan conservar salvas de la BD para restaurar los datos si se pierden o corrompen por alguna razón.

Los SGBD permiten definir autorizaciones o derechos de acceso teniendo en cuenta los usuarios, ubicaciones desde donde se puede acceder, así como asignar privilegios que tendrán los usuarios una vez autentificados. Lo primero que debe garantizarse es que sólo puedan acceder a los datos los usuarios autorizados.

PostgreSQL, permite configurar los permisos de los usuarios utilizando tres niveles de acceso. En el primer nivel se configuran los permisos de conexión para los host y los usuarios a la o las BD en el archivo pg_hba.conf, se define qué dirección o direcciones IP tendrán acceso a cuál o cuáles BD, y en qué modo podrán conectarse: conexión sin contraseña, validando el usuario y la contraseña o que rechace cualquier conexión desde el IP o rangos IP y usuarios seleccionados. En el segundo nivel, se configuran a qué BD pueden acceder determinados usuarios,

utilizando las opciones del archivo pg_ident.conf. En el tercer nivel, permite configurar los accesos de los usuarios a las tablas de la BD, utilizando las sentencias GRANT y REVOKE se pueden asignar o denegar respectivamente los permisos para insertar, eliminar y actualizar datos, entre otros.

En la BD propuesta se establecen controles de seguridad para los datos almacenados, garantizando que sólo los usuarios autorizados puedan solamente tener acceso a su dirección, para evitar la pérdida de los datos en caso de ocurrir alguna falla, PostgreSQL permite realizar salvas de la BD mediante el volcado (dump), pueden realizarse de forma automática o manualmente.

PostgreSQL corriendo, se puede realizar a una o varias BD y mantiene compatibilidad entre versiones; este proceso puede ser un poco lento cuando se hacen salvas a BD de gran volumen. Para restaurar las salvas, se utiliza el comando pg_restore indicando la salva que deseamos restaurar entre otros parámetros.

3.1) ¿Cómo lograr un correcto diseño de una Base de datos?

Validación teórica del diseño.

Para realizar la validación teórica de la base de datos se incluye fundamentalmente un análisis detallado de la integridad de la información, característica que asegura la calidad del almacenamiento y la disponibilidad de los datos. Con su tratamiento se evitan errores de entrada introducidos por usuarios descuidados o cualquier otra circunstancia de intento de violar la información existente en la base de datos.

Integridad

La integridad es un de los aspectos mas importantes a la hora de realizar el diseño de la base de datos. Se divide en varios aspectos como son:

Integridad referencial: garantiza interrelaciones validad entre entidades. Implica que los datos sean correctos, sin duplicaciones, perdida de datos, o relaciones mal resueltas. Todas las bases de datos relacionales incluyen esta propiedad, pues el software gestor es el encargado de su cumplimiento.

Integridad de dominio: viene dada por la validez de entrada de los datos para una columna determinada. Se puede exigir la integridad de dominio para restringir el

tipo mediante tipos de datos, el formato mediante reglas y restricciones, o el intervalo de valores posibles mediante restricciones.

Integridad de la entidad: define una fila como entidad única para una tabla, exige la integridad entre las columnas de los identificadores o la clave principal de una tabla, mediante índice y restricciones.

Normalización de la base de datos

La teoría de la normalización se ha desarrollado para obtener estructuras de datos eficientes que evidentes las anomalías de actualización.

"La normalización es una expresión formal del modo de realizar un buen diseño. Provee los medios necesarios para describir la estructura lógica de los datos en un sistema de información" (Mato, 2005).

Ventajas:

1. Evita anomalías en la actualización.
2. Mejora la independencia de los datos, permitiendo realizar extensiones de la base de datos afectando muy poco o nada a los programas de aplicación existente que acceden a la base de datos.

Involucra varias fases que se realizan en orden. Tras completar cada fase se dice que la información está en:

Primera Forma Normal (1FN): una relación está en 1FN si cumple con la propiedad de que sus dominios no contengan elementos que a su vez sean conjuntos.

Segunda Forma Normal (2FN): está basada en el concepto de dependencia funcional total, es decir que los atributos no primos dependen totalmente de los atributos llaves y debe cumplir con lo establecido en la 1FN. Se aplica solamente a los esquemas de relación que tengan llaves primarias compuestas por dos o más atributos. Si un esquema de relación está en 1FN y su llave es simple entonces está en 2FN.

Las relaciones que no están en 2FN pueden sufrir anomalías cuando se realizan actualizaciones.

Para la trasformación de 1FN a 2FN hay que eliminar las dependencias parciales de su clave primaria. Para ello se eliminan los atributos que son funcionalmente dependientes y se descomponen en una nueva relación con una copia de los atributos de la clave primaria de los que dependen.

Tercera Forma Normal (3FN): está basada en el concepto de dependencias transitivas. Se considera que está en 3FN si:

Si y solo si, la relación está en 2FN y los atributos no llaves son independientes de cualquier otro atributo no llave primaria.

Esto es lo mismo que decir que se deben eliminar las dependencias transitivas de atributos no llave respecto a la llave primaria, con la relación en 2FN.

La base de datos para los módulos de las direcciones de Justicia y Auditoría de la APA actualmente se encuentra en la 3FN, porque cumple con todas las especificaciones antes expuestas, es decir en ella no existen atributos multivaluados, dependencias transitivas entre las relaciones, ni redundancia de la información. La 3FN es la usada casi en la totalidad de los productos que utilizan base de datos, puesto que estas garantizan poca o una redundancia casi nula de la información, además logra eficacia en las consultas factor fundamental en el diseño e implementación de la bases de datos.

Análisis de la redundancia

Es aquella información duplicada o almacenada varias veces en la misma base de datos. Esto dificulta la tarea de modificación de datos y es el motivo más frecuente de inconsistencia de los mismos. Además requiere un mayor espacio de almacenamiento que influye en un mayor costo y tiempo de acceso a los datos. Los procesos de normalización mejoran en buena medida el comportamiento de este parámetro.

Después de haber llevado la base de datos para los módulos de las direcciones de justicia y auditoría de la APA a 3FN, es decir al normalizarla, queda libre de redundancia ya que se elimina completamente la presencia de datos repetidos innecesariamente.

Análisis de la seguridad

La seguridad es un punto importante en las bases de datos para evitar ataques e impedir cualquier acceso no autorizado, con la intensión de modificar, usar y/o difundir información almacenada, ya sea por error del usuario o un ataque intencional.

Los sistemas gestores garantizan en cierta medida que el control de acceso por parte de los usuarios a la base de datos se realice de acuerdo a los privilegios que tengan asignados y los lugares en que se encuentren ubicados y además los datos con los que se esta trabajando deben estar encriptados garantizando su seguridad.

En el caso de PostgreSQL permite realizar configuraciones para controlar el permiso de los usuarios. Tal es el caso del archivo pg_hba.conf que permite configurar los permisos de la conexión para los host y los usuarios de la base de datos, definiéndose que direcciones de IP tendrán acceso y en que modo se conectarán (conexión sin contraseña o con validación de user y contraseña) del archivo pg_indent.conf permite configurar que usuarios tendrán acceso a determinada base de datos, así como configurar los accesos a las tablas.

Para el desarrollo de la base de datos para los módulos de la direcciones de Justicia y Auditoria de la APA , se definió solamente dar permiso de administrador al encargado de modificar los campos de la tablas, las tablas y la utilización de contraseña, quedando conformado así un sistema seguro en el no cualquier usuario pueda acceder a la información de forma rápida, lo que provocaría problemas en la Administración Provincial porque no todas las personas que trabajen en distintos locales tienen el porque tener acceso a toda la información relacionada con la provincia.

3.2) Persistencia de los datos.

Actualmente persistir en una BD relacional objetos java es bastante sencillo, ya que existe una cierta cantidad de 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 (POJOs) 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 desarrollos a través de servicios de consulta.

3.3) Capa de Acceso a Datos (CAD).

Una capa es un conjunto de subsistemas que comparten el mismo grado de generalidad y de volatilidad en las interfaces: las capas inferiores son de aplicación general a varias aplicaciones y deben poseer interfaces más estables, mientras que las capas más altas son más dependientes de la aplicación y pueden tener interfaces menos estables [Jacobson et al...2000].

La arquitectura en capas se refiere a una visión de los componentes implementados en una capa que solo pueden hacer referencia a los componentes en capas inferiores, simplificando de esta forma la organización del sistema y reduciendo las dependencias, de manera que las capas inferiores no son participes de la implementación de las capas superiores. Su principal ventaja es la modularidad que le confiere a los sistemas de software.

En la CAD es donde se implementan los componentes que interactúan con la BD y donde se encapsula el acceso a datos con estos componentes de una manera fácil, para lograr un rendimiento óptimo. Permite que las demás capas, componentes e interfaces no interactúen directamente con el servidor de BD y así excluir las complejidades del acceso a datos en el código fuente de otras capas.

La CAD juega un importante papel dentro de la arquitectura de las aplicaciones, ella provee muchas ventajas claves entre las que se tienen:

Separación entre la capa de negocio y los gestores de BD: los componentes de la CAD implementan aquellas funciones especializadas en el acceso a los diferentes

servidores de BD y brindar esos datos, en un formato adecuado a la capa de negocio de la aplicación. Este nivel de aislamiento protege a los componentes de las otras capas, específicamente de la capa de negocio, de los cambios potenciales que puedan ocurrir en el servidor de BD.

Mejoras en el mantenimiento de la aplicación: encapsula toda la gestión de acceso a datos en una sola capa, que ofrece la ventaja de reutilizar componentes, reduciendo la cantidad de código fuente a desarrollar y mantener. (García, Plasencia. 2007)

3.3.1) Estructura de la capa de acceso a datos.

Técnicas para crear Capas de Acceso a Datos
Existen múltiples formas para crear una capa de acceso a datos:

1. Una forma es crear una clase por cada tabla en la base de datos y mapearlas una a una. Esta es la forma más sencilla y es muy válida cuando se tiene una estructura de base de datos que sea muy parecida al modelo de dominio (que no se haya optimizado con particiones horizontales o uniones).
2. Otra forma es respetar un modelo orientado a objetos que represente los objetos del dominio del negocio. Esta es una forma más correcta de trabajar pero se hace más complejo mapear esas clases a sus tablas correspondientes.

Model: contiene los paquetes DAO y Entity.

DAO: Se encuentra la clase genérica generic Dao.

Entity: contiene todas las clases necesarias para la generación de los datos.

(Ver anexo # 1).

DAO

Un DAO define la relación entre la lógica de presentación y empresa por una parte y por otra los datos. El DAO tiene un interfaz común, sea cual sea el modo y fuente de acceso a datos.

Algunas características:

No es imprescindible, pero en proyectos de cierta complejidad resulta útil que el

DAO implemente un interfaz. De esta forma los objetos cliente tienen una forma unificada de acceder a los DAO.

El DAO accede a la fuente de datos y la encapsula para los objetos clientes. Entendiendo que oculta tanto la fuente como el modo (JDBC) de acceder a ella.

El TransferObject encapsula una unidad de información de la fuente de datos. El ejemplo sencillo es entenderlo como un "bean de tabla", es decir, como una representación de una tabla de la base de datos, por lo que representamos las columnas de la tabla como atributos del TransferObject. El DAO crea un TransferObject (o una colección de ellos) como consecuencia de una transacción contra la fuente de datos. Por ejemplo, una consulta sobre ventas debe crear tantos objetos (TransferObject) de la clase Venta como registros de la consulta; el DAO devolverá la colección de TrasnferObject de la clase Venta al objeto Cliente. También puede ocurrir que el objeto Cliente mande un TransferObject para parametrizar una consulta o actualización de datos por parte del DAO.

Ventajas del DAO

1. Cualquier objeto no requiere conocimiento directo del destino final de la información que se manipula.
2. Se baja el nivel de acoplamiento entre clases, reduciendo la complejidad de realizar cambios.
3. Se aísla las conexiones a la fuente de datos en una capa fácilmente identificable.
4. Se oculta los detalles de implementación a la fuente de datos.
5. Simple ya puede ser entendido por la mayoría de los desarrolladores.
6. No requiere tiempo de ejecución de contenedores (código DAO puede ser una unidad de prueba en el cliente).
7. El código es muy eficiente, si el DAO está diseñado para aprovechar las capacidades de base de datos.

Paquete entity

Este paquete posee todas las tablas de la base de datos mapeadas y llevadas al NetBeans. A través del DAO y los métodos de acceso de estas clases se podrá acceder a los datos que estas poseen y utilizarlos según convenga al usuario.

3.4) Descripción de la arquitectura.

Para el diseño se hizo uso del estilo arquitectónico en capas como nivel de abstracción más alto de la estructura del sistema, que corresponden con las capas presentación negocio y acceso a datos.

Capa de Presentación

Es la que interactúa directamente con el usuario, captura la información de entrada por éste (realiza un filtrado previo para comprobar que no hay errores de formato) y hace las peticiones a la capa inferior mostrando al usuario la respuesta proveniente de ésta. Únicamente se comunica con la capa de negocio. (García, Plasencia. 2007)

Capa de Negocio

Está conformada por subsistemas, que se ajustan a los requisitos y casos de uso arquitectónicamente significativos. Desde el punto de vista de diseño, esta capa es contenedora de las clases entidades y controladoras. Únicamente se comunica con la capa de Acceso a Datos. (García, Plasencia. 2007)

3.5) Mapeo de las Entidades.

1. Para mapear las tablas en hibernate primero deben agregar un paquete en la sección “Source Package” del proyecto.

2. Haga clic derecho en “Source Package” aparecerán las siguientes opciones New > Other”. (Ver anexo # 2).

3. En categoría seleccione “Hibernate” y en File Types “Hibernate Configuration Wizard”, luego “Next >”. (Ver anexo # 3).

4. En File Name dejar por defecto como lo genera, luego “Next >”. En esta sección se creara el archivo hibenate.cfg.xml en el cual contiene la conexión de base de datos.

5. En File Name dejar por defecto como lo genera, en Database Connection seleccionar la cadena de conexión que ha configurado previamente en el netbeans y presionar “Finish”, en la ruta indicada en el punto 4 se creará el archivo de configuración de hibernate. (Ver anexo # 4).

6. El paso siguiente antes de mapear los archivos de la base de datos es generar el archivo de ingeniera inversa, para eso debe seleccionar “Source Package > New > Other” (Ver anexo # 5).

7. En categoría seleccione “Hibernate”, en Files Types “Hibernate Reverse Engineering Wizard”, luego “Next >” (Ver anexo # 6).

8. Dejar por defecto el nombre que le coloca la herramienta y el folder, ruta donde se insertará el archivo, luego presione “Next >”.

9. La herramienta escaneará las tablas de la base de datos y automáticamente las agregara en “Available Tables”, seleccione toda las tablas y presiona “Add >” (Ver anexo # 7).

10. Luego de haber agregado todas las tablas, aparecerán en selected tables, en caso que la opción “Include Related Tables” no se encuentre seleccionada, selecciónela; luego presione “Finish”.

11. El archivo generado mostrará esta lista de table-filter, en la ruta indicada en el punto 8 (Ver anexo # 8).

12. El tercer paso importante para finalizar y lograr el mapeo de las tablas es ejecutar el pojo en los archivos de configuración de la base de datos y el de ingeniería inversa efectuado en los puntos anteriores (Ver anexo # 9).

Seleccionar “Source Package > New >Other”.

13. En categorías seleccionar “Hibernate” y en File Types “Hibernate Mapping Files and POJOs from Database”, luego “Next >” (Ver anexo # 10).

14. Seleccionar “JDK 5 Language Features”, luego en Package el paquete donde se agregarán las tablas mapeadas, luego “Finish” (Ver anexo # 11).

15. En la carpeta seleccionada en el punto anterior se muestra las tablas mapeadas, con extensión “xml” y “java.

3.6) Pruebas.

Para garantizar el manejo y persistencia de la información de forma correcta se realizaron un conjunto de pruebas.

Pruebas de Conexión

Se realizaron comprobaciones al archivo de configuración (*hibérnate-persistence*) que necesita los *frameworks Spring* e *Hibernate* para la conexión con la base de datos.

```
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
  http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

    <bean id="myDataSource" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
        <property name="driverClassName" value="org.postgresql.Driver"/>
        <property name="url" value="jdbc:postgresql://10.208.1.250:5432/sigob"/>
        <property name="username" value="user"/>
        <property name="password" value="user"/>
    </bean>

    <bean id="sessionFactory"
        class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">
        <property name="dataSource" ref="myDataSource" />
        <property name="packagesToScan">
            <list>
                <value>cu.uci.hab.sigob.common.models.entity.administration</value>
                <value>cu.uci.hab.sigob.common.models.entity.common_nomenclatures</value>
                <value>cu.uci.hab.sigob.common.models.entity.directorates</value>
                <value>cu.uci.hab.sigob.common.models.entity.presidency</value>
            </list>
        </property>
        <property name="hibernateProperties">
            <props>
                <prop key="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</prop>
```

Figura 15: Prueba de Conexión

Pruebas de Persistencia

Las pruebas realizadas a la base de datos mediante la capa de acceso a datos fueron realizadas para identificar posibles problemas, las mismas arrojaron resultados satisfactorios, logrando que el cliente tenga las funcionalidades requeridas

Insertar los datos a la tabla antecedentes penales.

```
GenericDao dao=new DaojusticiaOK ();

D_antecedentes_penales antecedentes= new D_antecedentes_penales (5, 4, 2011, 2011-05-05, "san cristóbal", 4, 3);

Dao.save (antecedentes);
```

	id_anteceden [PK] serial	id_provincia integer	id_mes integer	anno integer	fecha date	registro character var	exterior integer	nacionales integer
1	11	2	4	2011	2011-02-04	kk	52	41
2	14	5	4	2011	2011-05-05	candelaria	2	2
3	15	5	4	2011	2011-05-05	san cristob	4	3
*								

Figura 16: Prueba de Persistencia Insertar Tabla d_antecedentes_penales

Eliminar los datos a la tabla antecedentes penales.

```
GenericDao dao=new DaojusticiaOK ();

D_antecedentes_penales antecedentes= new D_antecedentes_penales (5, 4, 2011, 2011-05-05, san cristóbal, 4, 3);

Dao.delete (antecedentes);
```

	id_anteceden [PK] serial	id_provincia integer	id_mes integer	anno integer	fecha date	registro character var	exterior integer	nacionales integer
1	11	2	4	2011	2011-02-04	kk	52	41
2	14	5	4	2011	2011-05-05	candelaria	2	2
*								

Figura 17: Prueba de Persistencia Eliminar Tabla d_antecedentes_penales

Modificar los datos en la tabla antecedentes penales.

```
GenericDao dao=new DaojusticiaOK ();

D_antecedentes_penales antecedentes= new D_antecedentes_penales (5, 4, 2012, 2012-05-05, san cristóbal, 4, 34);

Dao.update (antecedentes);
```

	id_anteceden [PK] serial	id_provincia integer	id_mes integer	anno integer	fecha date	registro character var	exterior integer	nacionales integer
1	11	2	4	2011	2011-02-04	kk	52	41
2	14	5	4	2011	2011-05-05	candelaria	2	2
3	15	5	4	2012	2012-05-05	san cristob	4	34
*								

Figura 18: Prueba de Persistencia Modificar Tabla d_antecedentes_penales

Estas pruebas fueron realizadas a la base de datos en su completo funcionamiento.

Resultados obtenidos.

Como resultado del trabajo Base de datos para los módulos de las direcciones de Justicia y Auditoria de la Administración Provincial de Artemisa (APA) en su versión 1.0. Se obtuvo una base de datos con su respectiva capa de acceso, la cual conjuntamente con su cliente y servidor concretaran un sistema que cumpla con todas las especificaciones planteadas por el cliente logrando la integridad y disponibilidad de los datos almacenados en las direcciones.

Funcionalidades Obtenidas.

Entre las principales funcionalidades que posee el proyecto en su versión 1.0 se pueden mencionar:

1. Lograr almacenar y recuperar lo datos de forma correcta.
2. Centralizar la información.
3. Una correcta generación de reportes para contribuir a una adecuada toma de decisiones.

Aporte social y económico.

La base de datos realizada para las Direcciones de Justicia y Auditoría en su versión 1.0 se utiliza en la APA.

El desarrollo de esta base de datos en versiones posteriores puede ser utilizado no solo en otras entidades de Artemisa sino también a nivel de país, ya que sus requisitos pueden ser modificados de forma fácil, erradicando así la mala gestión de la información existente. Este hecho posibilita además una disminución considerable en el tiempo y la complejidad del desarrollo del sistema. Las herramientas utilizadas para el desarrollo de esta base de datos son libres. En consecuencia, los costes de desarrollo y mantenimiento de dicha base de datos se reducen extraordinariamente.

3.7) Conclusiones Parciales

En este capítulo se expuso la implementación y la validación propuesta para el problema a resolver, logrando la creación de la capa de acceso a datos para los módulos de la direcciones evidenciando el correcto funcionamiento de los componentes para la persistencia de los datos en las direcciones del CAP.

CONCLUSIONES GENERALES

Conclusiones generales

Una vez concluida el diseño e implementación de la capa de acceso y la base de datos para los módulos de las direcciones de justicia y auditoria de la administración provincial de artemisa, se dan por resueltos todos los objetivos planteados

Se logro un diseño lógico de la base de datos minimizando la redundancia de los datos en la entidad.

Se implementó el modelo físico de la base de datos con restricciones de integridad que garantizan en gran medida que los datos modificados sean correctos

Se implemento la capa de acceso a datos garantizando la correcta disponibilidad e integridad de los datos almacenados.

GLOSARIO DE TÉRMINOS

Glosario de Términos

Escalabilidad (En ingeniería informática, la escalabilidad es la propiedad deseable de un sistema, una red o un proceso, que indica su habilidad para extender el margen de operaciones sin perder calidad, o bien manejar el crecimiento continuo de trabajo de manera fluida).

SQL lenguaje de consulta estructurado (*structured query language*).

BIBLIOGRAFÍA CONSULTADA

Bibliográficas Consultada

Base de datos. Enciclopedia libre.2011 [Consultado el 1 de Noviembre del 2011];
Disponible en: http://dajees.wordpress.com
http://www.slideshare.ne

Página Historia de la Informática. Blog sobre Historia de las base de datos. 2011 [Consultado el 5 de Noviembre del 2011];
Disponible en: http://www.cubaminrex.cu

Conferencia de Ingeniería de Software. (2011). Introducción a la Disciplina de Requisitos de RUP (pág. 5). La Habana: Universidad de las Ciencias Informáticas.

http://html.rincondelvago.com
http://ict.udlap.mx

Marqués A. María M. Diseño de bases de datos, 2001.
Disponible en:
http://www3.uji.es/~mmarques/f47/apun/node68.html.
http://www.maestrosdelweb.com

Damián Pérez Valdés. (2007) ¿Que son los sistemas de bases de datos?
http://www.cavsi.com

Date, Christopher J. Introducción a los Sistemas de Bases de Datos. Ciudad de La Habana: Editorial Félix Varela, 2003; Disponible en:
http://es.kioskea.net

Silverio Castro, Rogelio S. La problemática de las Bases de datos Distribuidas. 2005.

Marqués Andrés, María Mercedes. Apuntes de Ficheros y Bases de Datos. 2001.
Disponible en:
http://www3.uji.es/~mmarques/f47/apun/apun.html

"SQL Power Architect herramienta de modelado de datos." [Online].
Disponible en:
http://www.tuinformaticafacil.com/herramientas-desarrollo/sql-power-architect-herramienta-de-modelado-de-datos.
[Consultado el 5 de Noviembre de 2011].
http://www.dataprix.com

"PostgreSQL - Guía Ubuntu." [Online]. [Consultado el 5 de Noviembre del 2011]
Disponible en:
http://www.guia-ubuntu.org/index.php?title=PostgreSQL.
http://www.pgadmin.org.es.mk.gd/
"Bases de datos." [Online]. [Consultado 5 de Noviembre del 2011];
Disponible en:
http://www.hipertexto.info/documentos/b_datos.html
http://www.softpedia.es
"Hibernate - Wikipedia, la enciclopedia libre." [Online]. [Consultado el 5 de Noviembre del 2011];
Disponible en:
http://es.wikipedia.org/wiki/Hibernate
http://www.davidmarco.es

Anón. Motores de Persistencia. Programación en Castellano.

http://www.programacion.com/articulo/motores_de_persistencia_231#joa_persistencia_motores "Patrones de diseño de bases de datos. [Consultado el 5 de Noviembre de 2011]
Disponible en:
http://eva.hab.uci.cu
¨Java Database Connectivity - Wikipedia, la enciclopedia libre." [Online].
Disponible en:
http://es.wikipedia.org
http://c.unam.mx
Anon. Ha Muerto el Diseño? [Cited 16 February 2012].
Disponible en:
http://www.programacionextrema.org/articulos/designdead.es.html.

Página Mapeo de tablas con hibernate. Blog sobre el mapeo de tablas
[Consultado el 21 de abril 2012];
Disponible en:

http://elizabethsaenz.blogspot.com/2011/04/configuracion-hibernate-en-netbeans.html

Peñalver G, Meneses A, Garcías S. “SXP, METODOLOGÍA ÁGIL PARA EL DESARROLLO DE SOFTWARE”. 1Er Congreso Iberoamericano de Ingeniería de Proyectos. Mayo 2010. Universidad de las Ciencias Informáticas. Ciudad de la Habana. Cuba

“PostgreSQL - Guía Ubuntu.” [Online]. Disponible en: http://www.guia-ubuntu.org/index.php?title=PostgreSQL.

“Características de PostgreSQL | Manuales gratis.” [Online]. Disponible en: http://www.manualesdeayuda.com/manuales/bases-dedatos/postgresql/caracteristicas-de-postgresql-01844.html.

Mato G. Lic. Rosa M. Diseño de Bases de Datos. 1999.

Date, Christopher J. Introducción a los sistemas de bases de datos. 7ma Ed, México: Editorial Pearson Educación S.A, 2001, ISBN 968-444-419-2.

Marqués A. María M. Diseño de bases de datos, 2001. Disponible en: http://www3.uji.es/~mmarques/f47/apun/node68.html