Universidad de las Ciencias Informáticas
"Facultad 3"

Título: Diseño de la base de datos del procedimiento Ordinario de la materia Penal del proyecto Tribunales Populares Cubanos.

Trabajo de Diploma para optar por el título

de Ingeniero Informático

Autor: Ernesto Alejandro Figueredo Cuesta

Tutor: Ing. Alain Osorio Rodríguez

Co-tutor (a): Ing. Daimí Lamorú Marciel

Ciudad de La Habana, 2011

Declaración de Autoría

Declaro que soy el único autor de este trabajo y autorizo a la Facultad 3 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Ernesto Alejandro Figueredo Cuesta Autor	Ing. Alain Osorio Tutor

Resumen

En los Tribunales Populares Cubanos se procesa un alto volumen de información, la misma se encuentra dispersa en múltiples libros de registro y archivos por lo que la introducción de errores suele ser muy frecuente. Con el objetivo de dar solución a este problema en la facultad 3 de la Universidad de las Ciencias Informáticas se desarrolla un software que cuenta con un subsistema denominado Penal, cuya diseño de la base de datos será propuesta en el presente trabajo. La misma permitirá centralizar y organizar la información tramitada en los tribunales.

Para el diseño de la base de datos fue necesario realizar una investigación previa sobre las tendencias actuales en cuanto a soluciones informáticas existentes en el mundo, relacionada con sistemas jurídicos enfocándose en las bases de datos de los mismos. Además se recopiló información relacionada con las bases de datos y sistemas gestores de bases de datos. Una vez obtenida la información necesaria se procedió a la construcción de la base de datos requerida, la misma se basó en un modelo para el desarrollo de bases de datos, ofreciéndose organización tanto del equipo de desarrollo como del trabajo en general. Se realizó el correspondiente modelado de los datos así como las configuraciones del servidor de base de datos. Fueron tomadas una serie de medidas que garantizaron seguridad e integridad de la información almacenada según el diseño realizado. Además se realizó una validación teórica y funcional del diseño.

Palabras claves: Sistemas de Bases de Datos, Bases de Datos, Arquitectura, Modelo de Datos.

Índice

Introducción

En la actualidad el mundo se ha enmarcado en el desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC´s), las empresas e instituciones automatizan y perfeccionan los procesos y actividades que realizan. Las TIC´s son las tecnologías que se disponen para la gestión de la información, los recursos necesarios para almacenar, proteger y manipular esa información. Actualmente esta revolución científica y tecnológica se está aplicando en todos los campos de la sociedad, se presentan cada vez más como una necesidad, es una realidad y uno de los motores principales de la sociedad actual. (Aportela Rodrígues, Lic. Ivett M. Intranets, 2007)

Mejorar y simplificar los procesos, integrar procesos internos, ahorrar tiempo y dinero a través de los sistemas de información se hace hoy tarea primordial. Cuba, con un proyecto de desarrollo que tiene como principios la igualdad social, la participación popular y la solidaridad, ha diseñado medidas y estrategias que permitan convertir los conocimientos y las tecnologías, en instrumentos a disposición de las profundas transformaciones, logrando más eficiencia en los procesos y por consiguiente, un aumento en la calidad de vida de los ciudadanos. (Valdes, 1987)

La misma ha comenzado por los principales sectores de la sociedad, de forma que los Tribunales Populares Cubanos (TPC) no han sido la excepción.

La Universidad de las Ciencias Informáticas (UCI), nacida bajo el calor de la Batalla de Ideas, fue creada con el objetivo de impulsar el desarrollo económico y la informatización de nuestro país. Entre la diversidad de empresas y organismos que están en convenio con la universidad se encuentra el Tribunal Supremo Popular, que tiene como objetivo estratégico impartir justicia en nombre del pueblo de Cuba de manera pronta y cumplida. Esta importante organización tiene como objetivo principal lograr el orden público, así como establecer las medidas a seguir y a cumplir por el pueblo, razón por la que se impone lograr un ágil y efectivo procesamiento de la información.

Los tribunales populares cubanos están estructurados por instancias y materias. Existen tres instancias: municipal, provincial y supremo, cada uno de los cuales cuenta con diferentes procesos distribuidos en cinco materias que son tratadas a todos los niveles. Las materias anteriormente mencionadas son:

- Materia Administrativa.

- Materia Civil.
- Materia Económica.
- Materia Laboral.
- Materia Penal.

La materia Penal es una de las más importantes por el vínculo directo que tiene con los ciudadanos y porque en gran medida mantiene en una balanza unos de los principales derechos del hombre: la libertad. Esta materia tiene como objetivo dar solución a los conflictos que se originan por la comisión de delitos y realizar todos los actos procesales[1], formas y procedimientos que se hacen en el tribunal para dictar sentencia y ejecutarla conforme dispone la Ley de Procedimiento Penal. (Pírez, 1999)

Actualmente en la sección que se atiende el proceso Penal en los Tribunales Municipales Cubanos se maneja un gran número de causas y de expedientes, todo este trabajo se realiza de forma manual lo que trae como consecuencia intromisión de errores en los datos de las causas y los registros de documentos, violación del orden de ejecución de los actos procesales, morosidad en la tramitación, dificultad en la búsqueda de información producto de la dispersión física de los datos. Además los expedientes se guardan en lugares no idóneos, susceptibles de humedad y por ende de deterioro. Los reportes estadísticos son poco fiables por lo difícil que resulta recopilar gran volumen de información al transcurrir largos periodos de tiempo.

Teniendo en cuenta las dificultades descritas se define como **problema a resolver**: ¿Cómo minimizar la descentralización, duplicidad y deterioro de la información del procedimiento Ordinario de la materia Penal de modo que contribuya a su gestión eficiente?

Luego de un análisis de la situación actual, la investigación enmarca su **objeto de estudio:** sistemas informáticos para la gestión judicial.

Para la solución del problema se plantea como **objetivo general**: realizar el diseño de la base de datos del procedimiento Ordinario de la materia Penal de modo que contribuya a su gestión eficiente.

[1] Cada uno de los trámites por los que transita el expediente hasta que se resuelve el conflicto.

Se delimita el **campo de acción**: Bases de datos para el sistema de tribunales.

La **idea a defender** sobre la cual se sustentó la investigación es: con el diseño de la base de datos del módulo Ordinario del subsistema Penal se contribuirá a su gestión eficiente.

Para dar cumplimiento al objetivo general se realizaron las siguientes **tareas de la investigación:**

- ✓ Estudio del marco teórico que fundamenta el objeto de investigación
- ✓ Selección de la herramienta de modelado
- ✓ Selección del gestor de base de datos
- ✓ Diseño del Modelo de Datos.
- ✓ Refinamiento del documento de Configuraciones.
- ✓ Refinamiento del documento de Entorno de Desarrollo.
- ✓ Validación de la solución propuesta.

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

De los métodos teóricos se emplearon los siguientes:

Histórico-Lógico: Posibilitó determinar las características esenciales de los sistemas jurídicos existentes y realizar el estudio de la trayectoria real de determinados elementos que servirán de guía para el diseño y configuración de una base de datos para el módulo de lo administrativo

Modelación: Permitió la creación de modelos que representan abstracciones, con el objetivo de explicar cómo funciona el proceso.

El método empírico empleado es el siguiente:

Observación: Posibilitó comprender de mejor manera el proceso mediante la percepción planificada y prolongada del fenómeno.

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

En el **Capítulo I**. Fundamentación teórica: Se hace alusión al estado del arte del tema tratado, las tendencias, técnicas, tecnologías, metodologías y software usados para la solución del problema.

En el **Capítulo II**. Se describe el entorno de desarrollo, modelo de datos, las configuraciones, el acceso a datos, la seguridad de la base de datos, la optimización de la misma a través de la normalización y de la utilización de indexadores.

En el **Capítulo III**. Se describen los resultados alcanzados luego de aplicar diferentes pruebas para validar la calidad del sistema. Por otra parte, se realizará la validación funcional con la generación de códigos de programas para un llenado voluminoso de la base de datos y la búsqueda o diseño de herramientas para las pruebas de carga intensiva.

Capítulo 1 Fundamentación Teórica

1.1 Introducción

En el presente capítulo se da una breve reseña de la historia de las Bases de Datos, desde su surgimiento, hasta la actualidad. Se describen aspectos importantes de las mismas, sus tendencias tecnológicas y las metodologías de diseño con sus fases. Se abordan también temas como la seguridad, integridad de la información y normalización. Por último se especificarán las herramientas empleadas para el diseño e implementación de la base de datos (BD) del subsistema Penal.

1.1.1 Bases de datos jurídicas

Las bases de datos jurídicas intentan compendiar la normativa que emana de las distintas administraciones. Son un instrumento de búsqueda que facilita el acceso a la información. El estudio de los distintos textos jurídicos precisa que exista una relación entre los mismos que actualizan, complementan, reforman y amplían cada norma. Las bases de datos permiten "navegar" desde una norma hasta otras a las que afecta o que le afectan, consultar la jurisprudencia asociada, etc. La idiosincrasia de la documentación jurídica determina ciertas particularidades de estas bases de datos, entre las que conviene destacar las siguientes: (Diez, 2007)

a) Precisan recoger el texto completo de los documentos, dado que la información parlamentaria, legislativa y jurisprudencial ha de ser almacenada de forma íntegra para responder a las necesidades reales de los usuarios.

b) Emplean distintas unidades documentales: texto completo del documento, extractos, zonas consideradas especialmente significativas como artículos de leyes, fundamentos jurídicos de sentencias, etc.

c) Contienen, generalmente, un elevado volumen de documentos y, en consecuencia han ido creciendo inevitablemente en el mercado productos temáticos específicos.

1.1.2 Sistemas informáticos jurídicos a nivel internacional

Con el transcurso de los años la creación de software para el campo jurídico se ha incrementado en gran medida, debido a la utilización de las TICs con fines de agilizar y modernizar los procesos llevados a

cabo por las organizaciones y aprovechar todas las ventajas que las tecnologías aportan en cualquier área de la vida. Ejemplo de ello son los sistemas informáticos que se han comenzado a usar en países como España (INFOLEX, SOFT CLASS), con el objetivo de facilitar la tramitación de las causas, de los expedientes y acortar los plazos de duración de los juicios.

A continuación se ofrece una breve descripción de algunos de los sistemas mencionados:

Infolex

Software de Gestión Jurídica (España). Este software es realizado por la empresa Jurisoft la cual es líder en el sector de la Informática Jurídica. Nace en 1995 con una clara vocación de Servicio Integral, para cubrir todas las necesidades surgidas en la actividad diaria del Profesional del Derecho: Bases de Datos Documentales, Bibliografía, Infraestructura Informática, Servicios de Internet, Software de Gestión, Seguridad informática. Es compatible con todos los sistemas operativos Microsoft Windows perteneciente a las versiones XP recomendado, 2000, 2003, Vista, en cualquiera de sus ediciones. (Jurisoft, 1995)

Además con Microsoft SQL Server 2000, 2005 o superior (recomendado en clientes con gran volumen de datos con más de 10.000 expedientes), Oracle 9i o superior. Se puede optar por una versión de uso libre de SQL Server (Microsoft SQL Express 2005 suministrada con la instalación) o por otras versiones de bases de datos SQL de pago. Las versiones de pago Microsoft SQL Server 2000, 2005 (y Oracle 9i) y superior requieren las correspondientes licencias de Servidor y cliente. Microsoft SQL Express 2005 es un motor de datos gratuito Cliente-Servidor basado en SQL Server 2005 que se incluye con la licencia de Office. Las limitaciones fundamentales de SQL Express frente a SQL Server son:

- ✓ Monoprocesador (en el caso de servidores con más de un procesador sólo aprovecha uno)
- ✓ Límite de 1 Gb RAM (en el caso de servidores con más de 1 GB de RAM sólo aprovecha 1)
- ✓ Tamaño máximo de base de datos 4 GB

Es necesario aclarar que aunque SQL Express es un motor potente en la mayoría de los casos, será necesario evaluar por el cliente las previsiones de crecimiento de la Base de Datos.

Soft Class para abogados (España)

Es un sistema integral de gestión para bufetes de abogados y despachos o asesorías jurídicas que permite llevar un completo control de todas las tareas relacionadas con los mismos, además es un software adaptable a otros ámbitos. La principal función de este software es la administración de expedientes, con varios apartados en el que se gestionan independiente o conjuntamente los datos que lo conforman. El programa funciona con una base de datos en formato MS Access (programa, utilizado en los sistemas operativos Microsoft Windows, para la gestión de bases de datos creado y modificado por Microsoft y orientado a ser usado en entorno personal o en pequeñas organizaciones). Eso da garantía de poder recuperar los datos en cualquier momento, y de poder efectuar exportaciones. Del mismo modo, se puede leer los datos de la base de datos para efectuar una combinación o realizar una selección de registros.

1.1.3 Sistemas informáticos en el ámbito jurídico en Cuba

En el campo de la Informática Jurídica de Gestión el país ha comenzado a dar sus primeros pasos, con el objetivo de automatizar los procesos de las diferentes áreas en cada una de las instancias existentes para ganar en rapidez al tramitar la información. Como parte fundamental de los sistemas de gestión jurídica que están surgiendo se encuentran las bases de datos, que contienen toda la información persistente que utilizan estos procesos de justicia y sin la cual no podrían funcionar dichos sistemas.

"*Es de vital importancia señalar que aún cuando el Sistema Jurídico Cubano, se origina del Sistema Jurídico Romano -Francés, del cual provienen también los de los países que usan hoy esos sistemas de gestión, como es el caso de España, sus realidades actuales en el plano jurídico difieren totalmente de la cubana, razón por la cual resulta de alta complejidad la adaptación de estos sistemas. Todos ellos se rigen por las leyes judiciales de su país por lo que sus módulos no se corresponden con las leyes socialistas establecidas en Cuba*". (Silva, 2009)

Los elevados costos de estos sistemas, además del mantenimiento de sus licencias de uso, hacen que Cuba no los pueda asumir desde el punto de vista económico, cuestión que por las realidades financieras actuales es inevitable que se tome en consideración como punto de partida para cualquier análisis a la hora de proponer un sistema.

Tomando en cuenta todo lo anterior, se considera, que Cuba debe apostar por la realización de un sistema que cumpla con las características requeridas para satisfacer lo concerniente a la gestión de servicios legales que pueda ser utilizable en cualquier despacho legal. Un sistema que se ajuste a la realidad jurídica cubana, desarrollado por especialistas con herramientas de código abierto que permitan su mantenimiento, adaptaciones y ulteriores versiones.

Varios intentos de informatización de los tribunales populares cubanos se han llevado a cabo en el país, entre ellos:

- ✓ SISPRO (Procesos penales, Villa Clara, Cienfuegos).
- ✓ SISECO (Procesos Económicos).

De forma general estos programas presentan las siguientes deficiencias:

- ✓ No ofrecen reportes estadísticos.
- ✓ Nula comunicación entre instancias.
- ✓ Utilización de software propietario.
- ✓ Se puede hacer cualquier acción sobre el expediente sin tener en cuenta el estado del mismo.

Base de datos del Sistema de Gestión Fiscal

Para el diseño de la base de datos del Sistema de Gestión Fiscal los desarrolladores apostaron por el modelo relacional basados en la idea de que es el más conocido y eficiente para el desarrollo de bases de datos operacionales. Como herramientas se utilizaron el Visual Paradigm para el diseño del modelo lógico y el Postgres 8.4 como gestor de base de datos. No se utilizó una metodología específica para el diseño de la base de datos pero si diferentes patrones entre los que se encuentran:

- ✓ Árboles(simples, estructurados y fuertemente codificados)
- ✓ Grafos dirigidos simples
- ✓ Máquinas de estado para identidad y para escenarios
- ✓ Llaves subrogadas

La seguridad de la base de datos es la implementada por defecto en el gestor, basada en el control de acceso por roles (RBAC) y no se maneja las restricciones por ip (Internet Protocol - Protocolo de Internet) debido a que va a existir un servidor en cada fiscalía por lo que esta opción no es eficiente.

Como resultado de lo estudiado anteriormente se decide apostar por un sistema que se ajuste a la realidad jurídica de los tribunales, desarrollado por especialistas con herramientas de código abierto que permitan su mantenimiento, adaptaciones y ulteriores versiones aunque se tuvo en cuenta algunos aspectos de los sistemas antes mencionados.

1.2 Historia de las Bases de Datos

El término de bases de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California, Estados Unidos. Inicialmente una base de datos podía ser una de las antiguas bibliotecas o cualquier tipo de información coherente que fue persistida en papel y seguidamente almacenada en un estante o similar. En la actualidad, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría de las bases de datos están en formato digital lo que ofrece un amplio rango de soluciones al problema de almacenar datos. Existen varios criterios o definiciones de las bases de datos enunciadas por diferentes personalidades en el mundo, entre ellas se encuentran:

"*Un sistema de bases de datos es básicamente un sistema computarizado cuya finalidad general es almacenar información y permitir a los usuarios recuperar y actualizar esa información con base en peticiones*". (Date, 2003)

"*Colección o depósito de datos, donde los datos están lógicamente relacionados entre sí, tienen una definición y descripción comunes y están estructurados de una forma particular. Una base de datos es también un modelo del mundo real y, como tal, debe poder servir para toda una gama de usos y aplicaciones*". (Kybele,2006)

"Conjuntos de ficheros interrelacionados, con estructuras complejas y compartidos por varios procesos de forma simultánea (unos en línea y otros por lotes), recibieron al principio el nombre de Data Banks (Banco de Datos), y después, a inicios de los años setenta, el de Data Bases (Base de Datos)."(Rafael Camps Paré, 2005)

Se puede concluir entonces que una base de datos es una colección de datos interrelacionados, almacenados conjuntamente en uno o más ficheros de computadora, los cuales son usados posteriormente para la obtención de información.

1.2.1 Clasificaciones de las Bases de Datos

Las bases de datos pueden ser clasificadas de múltiples formas teniendo en cuenta algunos criterios existentes.

1.2.1.1 Según la forma en que cambia la información almacenada

Bases de datos estáticas: Son aquellas bases de datos en las que no se puede modificar la información que está guardada, es decir, solo se puede consultar, son utilizadas preferentemente para guardar datos históricos con los que luego se podrán realizar estudios sobre su comportamiento en el transcurso del tiempo, lo cual ayudará a tomar decisiones.

Bases de datos dinámicas: Son aquellas bases de datos en las que la información almacenada está en constante cambio, ya que sobre ella se pueden realizar operaciones tales como: adicionar, actualizar, eliminar, etc., así como las operaciones de consultas.

1.2.1.2 Según la información que almacenan

Bases de datos bibliográficas: Almacenan la información necesaria para poder localizar la fuente primaria. Un registro típico de una base de datos bibliográfica contiene información sobre el título, fecha de publicación, autor, editorial de una determinada publicación, etc. Puede contener un fragmento de la publicación original, pero nunca el texto completo.

Bases de datos de texto completo: Almacenan las fuentes primarias, es decir, guardan toda la información íntegra de las fuentes primarias, no fragmentos, ni referencias.

Directorios: Este es el caso de directorios digitales como los que edita la Empresa de Telecomunicaciones de Cuba (ETECSA) con los números telefónicos de sus usuarios, ya sean privados o estatales, donde se puede encontrar el nombre al que pertenece dicho número y la dirección dónde se encuentra.

Banco: es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.

1.2.2 Componentes de una Base de Datos

- ✓ **Hardware**: constituido por dispositivos de almacenamiento como discos, tambores, cintas, etc.
- ✓ **Software**: el Sistema Gestor de Bases de Datos (SGBD).
- ✓ **Datos**: están almacenados de acuerdo a la estructura externa y van a ser procesados para convertirse en información.

1.2.3 Definición de Integridad

Son restricciones que definen los estados de consistencia de la Base de Datos, esto significa que la BD o los programas que generaron su contenido, incorporen métodos que aseguren que el contenido de los datos del sistema no se rompa, así como las reglas del negocio. La integridad de los datos puede verse afectada añadiendo datos no válidos, modificando datos existentes, que se les asigne un valor incorrecto o eliminando datos que violen alguna regla. La integridad de datos presenta varias restricciones que posibilitan la declaración y comprobación de condiciones para expresar la consistencia, corrección y exactitud de datos almacenados, estas son: Integridad de dominio, de clave, de entidad, referencial, entre otras.

1.2.3.1 Integridad de los datos

La integridad de los datos no es más que la corrección y completitud de la información en una base de datos. Cuando los contenidos se modifican con sentencias INSERT (insertar), DELETE (eliminar) o UPDATE (actualizar), la integridad de los datos almacenados puede perderse de muchas maneras diferentes. Pueden añadirse datos no válidos a la base de datos, tales como un pedido que especifica un producto inexistente. Los cambios en la base de datos pueden perderse debido a un error del sistema o a un fallo en el suministro de energía. Una de las funciones importantes de un sistema gestor de base de datos relacional es preservar la integridad de sus datos almacenados en la mayor medida posible.

Existen cuatro restricciones fundamentales que sustentan la integridad de los datos, las mismas se mencionan a continuación:

- ✓ **Datos Requeridos:** establece que una columna tenga un valor no nulo. Se define efectuando que la declaración de una columna es no nula cuando la tabla que contiene las columnas se crea por primera vez, como parte de la sentencia crear tabla.

- ✓ **Chequeo de Validez**: Cuando se crea una tabla donde en cada columna tiene un tipo de datos y el sistema gestor de base de datos asegura que solamente los datos del tipo especificado sean ingresados en la tabla.
- ✓ **Integridad de entidad**: Establece que la clave primaria de una tabla debe tener un valor único para cada fila de la tabla; sino, la base de datos perderá su integridad. Se especifica en la sentencia crear tabla. El sistema gestor de base de datos comprueba automáticamente la unicidad del valor de la clave primaria con cada sentencia INSERT (insertar) Y UPDATE (actualizar). Un intento de insertar o actualizar una fila con un valor de la clave primaria ya existente fallará.
- ✓ **Integridad Referencial:** La integridad referencial es un sistema de reglas que utilizan la mayoría de las bases de datos relacionales para asegurarse que los registros de tablas relacionadas son válidos y que no se borren o cambien datos relacionados de forma accidental produciendo errores de integridad.

1.2.4 Fases del diseño de base de datos

El proceso de diseño de una base de datos cuenta con tres fases fundamentales: Diseño Conceptual, Diseño Lógico y Diseño Físico.

1.2.4.1 Diseño Conceptual

El objetivo del diseño conceptual, también conocido como modelo conceptual, y que constituye la primera fase de diseño, es obtener una buena representación de los recursos de información, con independencia de usuarios y aplicaciones en particular y sin considerar aspectos como eficiencia de la computadora. Esta primera fase consta de dos momentos: análisis de requisitos, donde se centra el trabajo en definir qué es lo que se va a representar, y la conceptualización, donde se piensa en cómo se va a proceder para representar lo antes definido.

1.2.4.2 Diseño Lógico

El diseño lógico parte del esquema conceptual. Un esquema lógico es la descripción de la estructura de la base de datos que puede procesarse por el Sistema Gestor de Base de Datos. El objetivo principal del diseño lógico es obtener un esquema lógico eficiente en cuanto a operaciones de consulta y actualización.

1.2.4.3 Diseño Físico

En este paso se especifican las estructuras de almacenamiento internas y la organización de los archivos de la base de datos. El principal objetivo del diseño físico es conseguir una instrumentación lo más eficiente posible del esquema lógico. Para lograrlo se analizan aspectos como las características del Sistema Operativo, el Sistema Gestor de Base de Datos, la herramienta para realizar el diseño, aspectos relacionados con el rendimiento y los requisitos de procesos así como las características del hardware, en fin, cualquier factor cercano con la computadora, para con ello lograr optimizar el consumo de recursos, minimizar el espacio de almacenamiento, proporcionar la seguridad máxima, disminuir los tiempos de respuesta y evitar las reorganizaciones.

1.3 Modelos para el desarrollo de Bases de Datos

1.3.1 Modelo de datos Jerárquico

Los modelos de datos jerárquicos almacenan la información de forma jerárquica. Los datos se organizan en forma similar a un árbol donde un padre tiene muchos hijos. Este árbol está compuesto de unos elementos llamados nodos. El nivel más alto del árbol se denomina raíz. Cada nodo representa un registro con sus correspondientes campos.

Este modelo es especialmente útil en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos, permitiendo crear estructuras estables y de gran rendimiento.

"El modelo de datos jerárquico presenta importantes inconvenientes, que provienen en gran medida de su rigidez, por la falta de capacidad de las organizaciones jerárquicas para representar sin redundancias ciertas estructuras muy difundidas en la actualidad. La poca flexibilidad de este modelo puede obligar a la introducción de redundancias cuando es preciso instrumentar, mediante el modelo jerárquico, situaciones del mundo real que no responden a una jerarquía". (Moraga, 2001)

La representación gráfica de este modelo se realiza mediante la creación de un árbol invertido, los diferentes niveles quedan unidos mediante relaciones. Ejemplo:

Además en este modelo solo se pueden representar relaciones 1: M, lo que provoca que ocurran problemas como son:

- ✓ No se admiten relaciones N:M

- ✓ Un segmento hijo no puede tener más de un padre.
- ✓ No se permiten más de una relación entre dos segmentos.
- ✓ Para acceder a cualquier segmento es necesario comenzar por el segmento raíz.
- ✓ El árbol se debe de recorrer en el orden designado.

1.3.2 Modelo de datos Orientado a Objetos

Las bases de datos orientadas a objetos se crearon para tratar de satisfacer las necesidades de las nuevas aplicaciones. La orientación a objetos ofrece flexibilidad para manejar algunos de estos requisitos y no está limitada por los tipos de datos y los lenguajes de consulta de los sistemas de bases de datos tradicionales. Una característica clave de las Bases de Datos Orientadas a Objetos (BDOO) es la potencia que proporcionan al diseñador al permitirle especificar tanto la estructura de objetos complejos, como las operaciones que se pueden aplicar sobre dichos objetos. Otro motivo de la creación de estos modelos es el creciente uso de los lenguajes orientados a objetos.

Una base orientada a objetos es una base de datos que incorpora todos los conceptos importantes del paradigma de objetos:

- ✓ **Encapsulación:** Propiedad que permite ocultar información al resto de los objetos, impidiendo así accesos incorrectos o conflictos.
- ✓ **Herencia:** Propiedad a través de la cual los objetos heredan comportamientos dentro de una jerarquía de clases.
- ✓ **Polimorfismo:** Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

Algunas de sus ventajas son:

- ✓ Combinación de los procedimientos de una entidad con sus datos, permite modelar tipos de datos complejos y definir operaciones entre ellos.
- ✓ La cantidad de información que puede modelarse en una base datos orientada a objetos se incrementa, y es más fácil modelar esta información.

- ✓ Los Sistemas de Bases de Datos Orientados a Objetos son capaces de tener mayores capacidades de modelado por medio de la extensibilidad.
- ✓ En una BDOO, el manejo de versiones está disponible para ayudar a modelar cambios diversos a los sistemas.
- ✓ La reutilización de clases juega un rol vital en el desarrollo y mantenimiento más rápido de aplicaciones. Las clases genéricas son potentes, pero más importante es que ellas pueden ser usadas nuevamente.

Entre sus principales desventajas se pueden mencionar:

- ✓ Cuentan con cierto nivel de inmadurez en el mercado de BDOO.
- ✓ Inexistencia de estándares en la industria orientadas a objetos.
- ✓ Estos sistemas apenas permiten flexibilidad para modificaciones, y el sistema debe desactivarse cuando se requiere modificar estructuras de objetos y métodos.
- ✓ Su arquitectura no es flexible, lo cual motiva, por ejemplo, el uso del modelo relacional.

Características mandatorias o reglas de oro

Un sistema de Base Datos Orientada a Objetos (BDOO) debe satisfacer dos criterios:

Mandatorias: Son las que el Sistema debe satisfacer a orden de tener un sistema de BDOO y estos son: objetos complejos, identidad de objetos, encapsulación, tipos ó clases sobre paso combinado con unión retardada, extensibilidad, persistencia y manejador de almacenamiento secundario, concurrencia, recuperación y facilidad de respuesta.

Opcional: Son las que pueden ser añadidas para hacer el sistema mejor pero que no son mandatorias estas son de: herencia múltiple, chequeo de tipos e inferencia distribución y diseño de transacciones y versiones.

Abiertas.- Son los puntos donde el diseñador puede hacer un número de opciones y estas son el paradigma de la programación la representación del sistema ó el tipo de sistema y su uniformidad.

1.3.3 Modelo de datos de red

Una base de datos en red consiste en un conjunto de registros conectados entre sí mediante punteros. Los registros son en muchos aspectos parecidos a las entidades del modelo entidad-relación (E-R). Cada registro es un conjunto de campos (atributos), cada uno de los cuales sólo contiene un valor de datos. Los punteros son asociaciones entre exactamente dos registros. Por tanto, los punteros pueden considerarse una forma restringida (binaria) de relación en el sentido del modelo E-R (Entidad Relación).

A diferencia del modelo jerárquico, en este modelo, un hijo puede tener varios padres. Esto ofrece una solución eficiente al problema de la redundancia de datos. Este modelo de datos permite representar relaciones N: M (mucho a mucho) sin embargo es poco utilizado.

Los conceptos básicos en el modelo en red son:

- ✓ El tipo de registro, que representa un nodo.
- ✓ Elemento, que es un campo de datos.
- ✓ Agregado de datos, que define un conjunto de datos con nombre.

Los modelos anteriores presentan deficiencias como son:

- ✓ Difíciles de administrar.
- ✓ Ejecución compleja.
- ✓ Carencia de independencia estructural

1.3.4 Modelo de datos Relacional

Una base de datos relacional es una base de datos donde todos los datos visibles al usuario están organizados estrictamente como tablas de valores, y en donde todas las operaciones de la base de datos operan sobre estas tablas. Estas bases de datos son percibidas por los usuarios como una colección de relaciones normalizadas de diversos grados que varían con el tiempo. En 1970 Edgar Frank Codd introdujo el modelo relacional, todos los datos están estructurados a nivel lógico como tablas formadas por filas y columnas, aunque a nivel físico pueden tener una estructura completamente distinta. Un punto fuerte del modelo relacional es la sencillez de su estructura lógica.

Características principales de los ``archivos'' relacionales:

- ✓ Cada ``archivo'' contiene solo un tipo de registros
- ✓ Los campos no tienen un orden específico, de izquierda a derecha
- ✓ Los registros no tienen un orden específico, de arriba hacia abajo
- ✓ Cada campo tiene un solo valor
- ✓ Los registros poseen un campo identificador único (o combinación de campos) llamada clave primaria.

Ventajas:

- ✓ **Operaciones de escritura intensivas**: las bases de datos relacionales están preparadas para hacer un uso constante de operaciones orientadas a transacciones, que implican la modificación o borrado constante de los datos almacenados.
- ✓ **Esquema específico para cada aplicación**: las bases de datos relacionales son creadas para cada aplicación específica, siendo complicado adaptar los esquemas a nuevas aplicaciones.
- ✓ **Modelo de datos complejo**: permiten manejar complejos modelos de datos que requieren muchas tablas.
- ✓ **Integridad de datos**: todos sus componentes están desarrollados para mantener la consistencia de la información en todo momento. Esto incluye operaciones de roll-back, integridad referencial y operaciones orientadas a transacciones.
- ✓ **Transacciones aisladas:** de tal forma que si dos transacciones están ejecutándose de forma concurrente los efectos de la transacción A son invisibles a la transacción B y viceversa, hasta que ambas transacciones han sido completadas.
- ✓ **Operaciones de roll-back (vuelta atrás):** hasta el final de la transacción ninguna de las acciones llevadas a cabo pasa a un estado final. Si el sistema falla antes de finalizar una transacción todos los cambios realizados son eliminados (roll-back)

Desventajas:

- ✓ Se dice que la fundamental consiste en la dificultad de lograr productividad adecuada de los sistemas, ya que no se emplean los medios técnicos idóneos, tales como las memorias

asociativas, siendo necesario simular este proceso, pero, en realidad, la eficiencia y productividad de los sistemas actuales resultan realmente muy satisfactorias (Mato García, 2005).

1.4 Modelo de Desarrollo de la Base de Datos

El desarrollo de una base de datos dentro del desarrollo de software es un aspecto importante y requiere atención. Los desarrolladores de software en ocasiones toman decisiones cuestionables respecto a base de datos desde un punto de vista arquitectónico. Al mismo tiempo, encontramos especialistas de base de datos que tienen poca o ninguna experiencia con las técnicas modernas de desarrollo de software.

Para el diseño de la base de datos se utilizará el modelo relacional pues ofrece un grupo de ventajas importantes en comparación con otros modelos existentes, de ahí que la dirección del proyecto apostara por su puesta en práctica.

Con el fin de lograr mayor organización en el trabajo se utilizará como guía el Modelo de Desarrollo de Bases de Datos para Procesos de Desarrollo de Software propuesto en el trabajo de diploma del Ingeniero Alain Osorio Rodríguez. En el mismo se contemplan 3 fases fundamentales:

1. Inicio o Definición.
2. Desarrollo o Construcción.
3. Transición o Despliegue.

En la fase de inicio el mayor esfuerzo está en desarrollar la primera estructura de la base de datos y a su vez las configuraciones iniciales, lo cual tributa a los primeros pasos del desarrollo. En la fase de desarrollo suele ocupar el mayor tiempo las tareas de implementación y acceso a datos, con determinados cambios en el modelo de datos, y en la fase de transición suele ocupar mayor importancia las configuraciones para la puesta en marcha del software a su entorno real de ejecución.

El proceso está divido en cuatro actividades fundamentales las cuales incluyen tareas generales.

1. Modelado de Datos.
2. Configuraciones.
3. Implementación.

4. Acceso a Datos, Optimización, Prueba (ADTP).

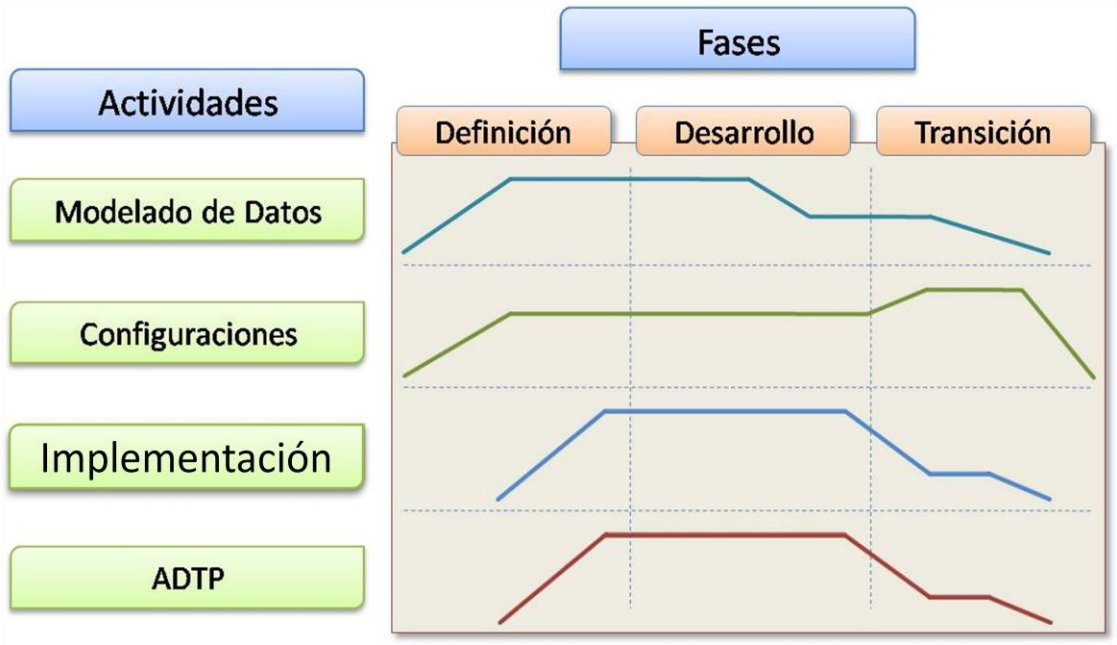


Figura 1 Modelo de desarrollo

1.4.1.1 Modelado de Datos

La actividad tiene como tarea general la estructuración del Modelo de Datos y su respectiva documentación. Los objetivos de esta actividad son:

1. Estructurar iterativamente el Modelo de Datos.
2. Documentar el Modelo de Datos a la par del desarrollo del mismo.

Se debe ir refinando la documentación asociada a la estructura de la base de datos a medida que evoluciona el Modelo de Datos.

El artefacto que se genera es el Modelo de Datos.

1.4.1.2 Configuraciones

La actividad tiene como tareas generales el establecimiento de las configuraciones del servidor de base de datos, así como la estructuración de una estrategia de instalación y configuración de la arquitectura de la BD para diferentes entornos de desarrollo, como son el propio desarrollo, prueba y despliegue.

El objetivo de esta actividad es:

1. Definir y aplicar las configuraciones del sistema operativo – sistema gestor de bases de datos – base de datos para diferentes entornos de desarrollo mediante automatizaciones.

Los artefactos que se generan en esta actividad son:

1. Documento de configuraciones del sistema operativo – sistema gestor de bases de datos – base de datos.

2. Estructura automatizada de aplicación de las configuraciones.

Ejemplo de automatización de las configuraciones puede ser crear una imagen del sistema operativo y sus configuraciones para los servidores de base de datos.

1.4.1.3 Implementación

Esta actividad se centra en la implementación de todas las funcionalidades requeridas en la BD.

Los objetivos de esta actividad son:

1. Implementar las funcionalidades de la base de datos.

2. Documentar las funcionalidades.

Se debe ir documentando las funcionalidades a medida que se van implementando.

El artefacto que se genera es la Codificación de las Funcionalidades. La documentación debe formar parte de este artefacto.

1.4.1.4 ADTP

El nombre de esta actividad viene dado por las tareas generales que se realizan:

1. Acceso a Datos.
2. Tuning (Optimización).
3. Pruebas Generales.

La tarea general Acceso a Datos se refiere a la creación y modificación de la interfaz de comunicación entre la aplicación de negocio y la base de datos. La tarea general Tuning se refiere a las optimizaciones de E/S (Entrada/Salida) vinculadas con las funcionalidades implementadas en la base de datos. Por último la tarea general de Pruebas Generales se refiere a las pruebas en el marco de las tareas de Acceso a Datos y Tuning, y puede definirse como una tarea intermedia de comprobación y terminación.

En esta actividad, las tareas generales suelen estar muy relacionadas y solo en determinadas ocasiones se realizan aisladamente.

Los objetivos de esta actividad son:

1. Estructurar el acceso a datos.
2. Optimizar las operaciones de E/S vinculadas a la codificación de las funcionalidades de la base de datos.
3. Probar las funcionalidades hasta obtener un resultado eficiente en la medida de los requerimientos del sistema.
4. Garantizar un acceso a datos óptimo y seguro.

El artefacto que se genera es el Acceso a Datos.

1.4.2 Normalización de Base de Datos

La normalización es el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas, que además de ser más simples y más estables, son más fáciles de mantener. También se puede entender la normalización como una serie de reglas que sirven para ayudar a los diseñadores de BD a desarrollar un esquema que minimice los problemas de lógica. Cada regla está basada en la que le antecede. La normalización se adoptó porque el viejo estilo de poner todos los datos en un solo lugar, como un archivo o una tabla de la BD, era ineficiente y conducía a errores de lógica

cuando se trataban de manipular los datos. Los seres humanos tienen la tendencia de simplificar las cosas al máximo. Lo hacen con casi todo, desde los animales hasta con los automóviles. Ven una imagen de gran tamaño y la hacen más simple agrupando cosas similares juntas. Las guías que la normalización provee, crean el marco de referencia para simplificar una estructura de datos compleja. Otra ventaja de la normalización de BD es el consumo de espacio. Una BD normalizada ocupa menos espacio en disco que una no normalizada. Hay menos repetición de datos, lo que tiene como consecuencia un mucho menor uso de espacio en disco.

Consiste en aplicar una serie de reglas a las relaciones obtenidas tras el paso del modelo entidad relacional al relacional. Las BD relacionales se normalizan para:

- Evitar la redundancia de los datos.
- Evitar problemas de actualización de los datos en las tablas.
- Proteger la integridad de los datos.

Las formas normales son aplicadas a las tablas de una BD. Decir que una BD está en la forma normal **N** es decir que todas sus tablas están en la forma normal **N**. En general, las primeras tres formas normales son suficientes para cubrir las necesidades de la mayoría de las BD. El creador de estas 3 primeras formas normales (o reglas) fue Edgar Frank Codd. Las formas normales establecen los siguientes planteamientos:

El proceso de Normalización define tres Formas Normales (FN) principalmente:

- ✓ Primera Forma Normal (1FN)
- ✓ Segunda Forma Normal (2FN)
- ✓ Tercera Forma Normal (3FN)
- ✓ Forma Normal de Boyce-Codd (FNBC)

Primera Forma Normal (1FN)

Una tabla está en Primera Forma Normal si:

Todos los atributos son atómicos. Un atributo es atómico si los elementos del dominio son indivisibles, mínimos. La tabla contiene una clave primaria, la clave primaria no contiene atributos nulos, no posee ciclos repetitivos, no debe de existir variación en el número de columnas, una columna no puede tener múltiples valores. Los datos son atómicos (Si a cada valor de X le pertenece un valor de Y, entonces a cada valor de Y le pertenece un valor de X). Esta forma normal elimina los valores repetidos dentro de una BD.

Segunda Forma Normal (2FN)

Independencia Funcional. Una relación está en 2FN si está en 1FN y si los atributos que no forman parte de ninguna clave dependen de forma completa de la clave principal. Es decir que no existen dependencias parciales.

Tercera Forma Normal (3FN)

La tabla se encuentra en 3FN si es 2FN y si no existe ninguna dependencia funcional transitiva entre los atributos que no son clave.

Las formas normales someten el esquema de relación a una serie de pruebas para certificar si pertenecen a una cierta forma normal, cuando el esquema relacional esté en FNBC, es que ya el esquema se llevó a 3FN, y para estar en 3FN el esquema se tuvo que haber llevado a 2FN y así sucesivamente, por lo que se puede decir que la FNBC es una de las FN más deseadas para un buen diseño de BD. No obstante, no siempre que el diseño de BD esté en la FNBC podemos garantizar que sea un diseño idóneo de los datos.

Ventajas de la normalización

- ✓ Evita anomalías en la actualización.
- ✓ Mejora la independencia de los datos, permitiendo realizar extensiones de la base de datos, afectando muy poco, o nada, a los programas de aplicación existentes que permiten el acceso a la BD.
- ✓

1.4.3 Desnormalización de Bases de Datos

La desnormalización es el proceso al cual se someten los diseños de bases de datos con el fin de ganar en rendimiento. Una base de datos con la medida justa de desnormalización reduce el número de tablas que deben combinarse sin dificultar en exceso el proceso de actualización. Entonces, tendrá consecuencias de redundancia de datos, pero mejorará el rendimiento ya que se evitarían uniones y subconsultas, pero es importante tener en cuenta que sólo se debería implementar si presenta notables ventajas de funcionamiento. El punto de vista teórico, marca la premisa en general, de evitar la redundancia de datos para obtener un diseño normalizado y optimizado. Pero si se parte desde la realidad

cotidiana en la cual se desenvuelve el negocio, en la práctica, dependiendo del contexto en que se esté trabajando a veces se debe desestimar parte de la normalización, para mejorar el comportamiento de un sistema y para lograr un mejor desempeño en el funcionamiento del sistema. Se adiciona la variable costos en un ambiente real, que muchas veces tiene un impacto más ponderante que el que un buen diseñador podría desear. Cabe destacar que a decisión de incluir redundancia en los datos (por más mínima que esta sea) debe ser perfectamente estudiada y analizada, y por otro lado los beneficios deberían ser claramente notorios para que justifique la acción.

1.5 ***Seguridad de las Bases de Datos***

Consiste en las acciones que toma el diseñador de base de datos al momento de crearla, tomando en cuenta el volumen de las transacciones y las restricciones que tiene que especificar en el acceso a los datos; esto permitirá que el usuario adecuado sea quién visualice la información adecuada.

La seguridad en las bases de datos abarca varios temas:

- ✓ Cuestiones éticas y legales relativas al derecho a tener acceso a cierta información.
- ✓ Cuestiones de política en el nivel gubernamental, institucional o corporativo relacionadas con la información que no debe estar disponible para el público.
- ✓ Cuestiones relacionadas con el sistema.

Las bases de datos hoy en día son de gran importancia en cada una de las aplicaciones que la requieran, es por ello que la seguridad para su acceso y manejo de la información se restringe en una jerarquía de usuarios. Los sistemas gestores de bases de datos (SGBD) permiten mantener la seguridad mediante el establecimiento de claves para identificar al personal autorizado a utilizar la base de datos. Las autorizaciones se pueden realizar a nivel de operaciones, de modo que un usuario puede estar autorizado a consultar ciertos datos pero no a actualizarlos.

La protección de los datos deberá llevarse a cabo contra fallos físicos, fallos lógicos y fallos humanos (intencionados o no). Estos fallos alteran indebidamente los datos, los corrompen; por lo que la base de datos ya no puede servir a los fines para los que fue creada.

Pudieran tomarse en cuenta algunos aspectos de importancia para lograr mayor seguridad en las bases de datos, algunos de ellos se enuncian a continuación:

- ✓ **Protección en profundidad**: entre más acciones se tomen para incrementar la protección de la base de datos, menor será la probabilidad de que un atacante tenga éxito, y exponga o abuse de cualquier información secreta que estuviera almacenada.
- ✓ **Protección de los ficheros de la base de datos:** ya que todos los ficheros almacenados en la base de datos están protegidos contra escritura por cualquier cuenta que no disponga de los permisos necesarios.
- ✓ Las conexiones de los clientes se deben restringir por dirección IP y/o por nombre de usuario.
- ✓ Las conexiones de los clientes pueden ser autentificadas mediante otros paquetes externos.
- ✓ A cada usuario se le asigna un nombre de usuario y (opcionalmente) una contraseña. Por defecto, los usuarios no tienen permiso de escritura a bases de datos que no hayan creado.
- ✓ Los usuarios pueden ser incluidos en grupos, y el acceso a las tablas puede restringirse en base a esos grupos.
- ✓ Evitar que los puertos de la base de datos se encuentren publicados hacia el exterior.
- ✓ Usar las herramientas de seguridad que proporcione el SGBD. Perfiles de usuario, vistas, restricciones de uso de vistas, etc.

1.6 Sistemas Gestores de Base de Datos

Ante la notable demanda de soluciones informáticas para la progresiva informatización de todas las organizaciones con la necesidad de optimizar servicios y productos, han surgido diferentes gestores de bases de datos; estos son programas que permiten manejar la información de modo sencillo y que prestan servicios para el desarrollo y el manejo de bases de datos. Los mismos sirven de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

Los SGBD ofrecen un control centralizado de la información teniendo como objetivos evitar la redundancia de los datos, mejorar los mecanismos de seguridad de los mismos y la privacidad, mantener la integridad

de los datos realizando las validaciones necesarias y mejorar la eficacia del acceso a los datos. En los textos que tratan este tema, o temas relacionados, se mencionan los términos SGBD y DBMS, siendo ambos equivalentes, y acrónimos, respectivamente, de Sistema Gestor de Bases de Datos y Administrador de Sistemas de Base de Datos (Data Base Management System), por sus siglas en inglés.

Diccionario de datos: Es una base de datos donde se guardan todas las propiedades de la base de datos, descripción de la estructura, relaciones entre los datos, etc.

El diccionario debe contener:

- ✓ La descripción externa, conceptual e interna de la base de datos
- ✓ Las restricciones sobre los datos
- ✓ El acceso a los datos
- ✓ Las descripciones de las cuentas de usuario
- ✓ Los permisos de los usuarios
- ✓ Los esquemas externos de cada programa

1.6.1 Oracle

Oracle es básicamente una herramienta cliente/servidor para la gestión de Bases de Datos. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales, por norma general. En el desarrollo de páginas web pasa lo mismo: como es un sistema muy caro no está tan extendido como otras bases de datos, por ejemplo, Access, MySQL y SQL Server. Para su utilización primero sería necesario la instalación de la herramienta servidor (Oracle 8i) y posteriormente se podría atacar a la base de datos desde otros equipos con herramientas de desarrollo como Oracle Designer [2]y Oracle Developer[3], que son las herramientas básicas de programación sobre Oracle.

[2] Herramienta para el diseño en el sistema gestor de base datos Oracle.
[3] Herramienta para el desarrollo e implementación en el sistema gestor de base de datos Oracle.

Oracle es sin duda una de las mejores bases de datos que existen en el mercado, es un sistema gestor de base de datos robusto, tiene muchas características que garantizan la seguridad e integridad de los datos; que las transacciones se ejecuten de forma correcta, sin causar inconsistencias.

Ventajas:

- ✓ Oracle ofrece soporte mundial a través de sus centros de soporte y sus sitios web donde pueden encontrarse desde scripts hasta documentos de instalación.
- ✓ Ayuda a administrar y almacenar grandes volúmenes de datos; estabilidad, escalabilidad y es multiplataforma.
- ✓ Es el motor de base de datos relacional más usado a nivel mundial.
- ✓ Ofrece mayor seguridad de la BD con Oracle Virtual Database (Base datos virtual de Oracle) y Oracle Portal.
- ✓ Puede ejecutarse en todas las plataformas, desde una computadora personal hasta un supercomputador.
- ✓ Permite el uso de particiones para la mejora de la eficiencia, de replicación e incluso ciertas versiones admiten la administración de bases de datos distribuidas.

Inconvenientes:

- ✓ El mayor inconveniente de Oracle es su precio, incluso las licencias de la edición personal son excesivamente caras.
- ✓ Necesidad de ajustes. Un error frecuente consiste en pensar que basta instalar el Oracle en un servidor y conectar directamente las aplicaciones clientes.
- ✓ El elevado coste de la información. Sólo últimamente han comenzado a aparecer buenos libros sobre asuntos técnicos distintos de la simple instalación y administración.

1.6.2 Microsoft SQL Server

Microsoft SQL Server es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional. SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle, PostgreSQL o MySQL.

Es uno de los más populares sistemas de gestión de bases de datos relacional, utiliza como lenguaje de programación el Structured Query Language [4](SQL) por sus siglas en inglés. Es capaz de almacenar una enorme cantidad de datos de gran variedad y de distribuirlos para cubrir las necesidades de cualquier tipo de organización. MySQL es un software de código abierto, licenciado bajo la Licencia Pública General (GPL) por sus siglas en inglés de la GNU[5], aunque MySQL AB distribuye una versión comercial, en lo único que se diferencia de la versión libre, es en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de otra manera, se vulneraría la licencia GPL.

MySQL Server posee las siguientes características:

- ✓ Facilidad de instalación, distribución y utilización.
- ✓ SQL Server posee una gran variedad de herramientas administrativas y de desarrollo que permite mejorar la capacidad de instalar, distribuir, administrar y utilizar SQL Server.
    - Administrativas (Administrador Corporativo).
    - Desarrollo (Analizador de Consultas).
- ✓ Almacenamiento de datos.
- ✓ SQL Server incluye herramientas para extraer y analizar datos resumidos para el proceso analítico en línea Proceso Analítico en Línea también llamado Online Analitycal Processing (OLAP), por sus siglas en inglés .También incluye herramientas para diseñar gráficamente la base de datos y analizar los datos mediante preguntas en lenguaje normal.
- ✓ SQL Server se integra con el correo electrónico, internet y Windows, permitiendo una comunicación local.

1.6.3 PostgreSQL

PostgreSQL es uno de los Sistemas Gestores de Bases de Datos más utilizados por la comunidad de software libre por las razones siguientes: Cumple con las propiedades ACID (Atomicidad, Consistencia,

[4] Lenguaje de consulta estructurado.
[5] Acrónimo recursivo que significa ´´GNU no es unix´´

Aislamiento y Durabilidad) y soporta el lenguaje común de acceso a los datos: SQL. Es multiplataforma y posee buenas interfaces de instalación y administración. Aproxima los datos a un modelo Objeto-Relacional, y es capaz de manejar completas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multiversión, soporte multiusuario, transacciones y optimización de consultas.

1.6.3.1 PostgreSQL 8.4

El Sistema Gestor de Bases de Datos PostgreSQL está establecido para su uso en el Sistema de los Tribunales Populares Cubanos por las normativas de la dirección del proyecto, la versión que se usa del mismo es la 8.4.

La versión 8.4 hace el análisis de datos mucho más sencillo a través de funcionalidades avanzadas de ANSI SQL: 2003, como las funciones Windows, expresiones comunes de tabla y joins (unión entre tablas) recursivos. Estas estructuras de consulta aumentan sustancialmente la expresividad del dialecto SQL de PostgreSQL, permitiendo a los usuarios hacer preguntas interesantes en una sola consulta, que habría sido imposible de construir antes. Las mejoras en los procedimientos almacenados, como los valores por omisión para los argumentos y los argumentos de largo variable hacen más simple y compacta la programación en la base de datos.

Entre las mejoras más populares PostgreSQL 8.4 están:

- ✓ Restauración de bases de datos en procesos paralelos, que acelera recuperación de un respaldo hasta 8 veces.
- ✓ Privilegios por columna, que permiten un control más granular de datos confidenciales.
- ✓ Configuración de ordenamiento configurable por base de datos, lo cual hace a PostgreSQL más útil en entornos con múltiples idiomas.
- ✓ Actualizaciones “en el lugar” desde 8.3 a 8.4 con muy bajo downtime (tiempo de inactividad), gracias al uso de pg_migrator beta.
- ✓ Nuevas herramientas de monitoreo de consultas.

- ✓ Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.
- ✓ Soporta el uso de índices, reglas y vistas.

1.7 Herramientas CASE

"Las herramientas de la ingeniería del software proporcionan un soporte automático o semi-automático para el proceso y los métodos, a estas herramientas se les llama herramientas CASE (C*omputer Arded Software Engineering)*." (Rumbaugh, y otros, 2000)

La realización de un nuevo software requiere que las tareas sean organizadas y completadas en forma correcta y eficiente. Las herramientas CASE fueron desarrolladas para automatizar esos procesos. Entre sus objetivos más importantes está conseguir la generación automática de programas desde una especificación a nivel de diseño.

Proporcionan asistencia a los analistas, ingenieros de software y desarrolladores durante todo el ciclo de vida de un software. Además los diagramas puedan ser fácilmente creados y modificados, por lo que hace el trabajo de diseño más fácil y agradable. Las herramientas CASE aumentan la productividad y la eficiencia del analista al disminuir la cantidad de tiempo necesaria para documentar, analizar y desarrollar sistemas de información. El analista tiene el potencial de ser más productivo ya que se pueden completar las mismas actividades de desarrollo en un tiempo menor que el que se utiliza cuando no se utiliza las herramientas.

En resumen, proporcionan ayuda para el desarrollo de software: desde la planificación, análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. A continuación se analizan algunas de las herramientas CASE más utilizadas en la actualidad.

1.7.1 Enterprise Arquitecht

Enterprise Architect es una herramienta comprensible de diseño y análisis UML, que cubre el desarrollo de software desde la captura de requerimientos a través de las etapas del análisis, modelos de diseño, pruebas y mantenimiento. Además, ofrece salida de documentación flexible y de alta calidad. Provee

generación poderosa de documentos y herramientas de reporte con un editor de plantilla completo WYSIWYG[6] .Genera reportes detallados y complejos con la información que necesita en el formato que su compañía o cliente demanda.

Soporta la generación e ingeniería inversa de código fuente para muchos lenguajes populares, incluyendo C++, C#, Java, Delphi, VB.Net, Visual Basic y PHP. Con un editor de código fuente con "resaltador de sintaxis" incorporado, EA le permite navegar y explorar su modelo de código fuente en el mismo ambiente. “Enterprise Architect provee trazabilidad completa desde el análisis de requerimientos y los artefactos de diseño, a través de la implementación y el despliegue.” (Sparx Systems, 2009)

Posibilita la agregación, composición y anidamiento entre requisitos. Ofrece una vista jerárquica para mostrar interconexiones entre elementos. Soporta la trazabilidad completa usando las vistas de jerarquía y matriz. Posee capacidad para adjuntar requisitos a otros elementos del modelo, y para ver la trazabilidad desde los requerimientos hasta los componentes desplegados. Además posee soporte de línea base para capturar los requerimientos en un punto en el tiempo y ayuda a administrar el desarrollo y cambio de requerimientos. Es una herramienta multiusuario[7], con seguridad y administración de permisos incorporada. Como una solución de modelado verdaderamente ágil, Enterprise Architect provee una sobrecarga de instalación baja, un rendimiento brillante y una interfaz intuitiva.

1.7.2 Visual Paradigm

Visual Paradigm es una herramienta UML (Unified Modeling Language) o en el español Lenguaje de Modelado Unificado, profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientado a objetos, construcción, pruebas y despliegue. La herramienta de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Visual Paradigm proporciona abundantes tutoriales, demostraciones interactivas y proyectos UML. Permite la integración con diversos IDE´s [8] como:

---

[6] what you see is what you get (lo que ves es lo que obtienes).

[7]Propiedad que permite proveer servicio y procesamiento a múltiples usuarios simultáneamente.

[8] Es un acrónimo de Internet Demo And Evaluation System, en español Modelo de demostración y sistema de evaluación.

- ✓ Net Beans (de Sun).
- ✓ JDeveloper (de Oracle).
- ✓ Eclipse (de IBM).
- ✓ JBuilder (de Borland).

Paradigm para UML es una herramienta CASE potente y de fácil uso para modelaje UML. Permite construir los siguientes diagramas UML:

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

Visual Paradigm tiene licencia tanto libre como comercial, es fácil de instalar y utilizar, puede encontrarse en varios idiomas y existe compatibilidad entre ediciones, sin embargo no permite su uso en proyectos comerciales e incluye marca de agua recordando este hecho. Muestra muchas otras funcionalidades no disponibles como gancho para las versiones de pago y las imágenes y reportes generados, no disponen de buena calidad. Además no es una herramienta destinada específicamente al diseño de bases de datos, no permite el indexado de tablas ni el direccionamiento de datos.

1.7.3 Er Studio Repositorio

Er Studio Repositorio (ER/Studio) es una herramienta de modelado de datos fácil de usar y multinivel, para el diseño y construcción de bases de datos a nivel físico y lógico. Direcciona las necesidades diarias

de los administradores de bases de datos, desarrolladores y arquitectos de datos que construyen y mantienen aplicaciones de bases de datos grandes y complejos. ER/Studio está equipado para crear y manejar diseños de bases de datos funcionales y confiables. Ofrece fuertes capacidades de diseño lógico, sincronización bidireccional de los diseños físicos y lógicos, construcción automática de bases de datos, documentación y fácil creación de reportes.

Er Studio Repositorio presenta una serie de funcionalidades como:

- ✓ Capacidad fuerte en el diseño lógico.
- ✓ Sincronización bidireccional de los diseños lógico y físico.
- ✓ Construcción automática de Base de Datos.
- ✓ Reingeniería inversa de Base de Datos.
- ✓ Documentación basada en HTML.
- ✓ Un Repositorio para el modelado.

Todas estas fueron razones que motivaron al equipo de desarrollo del proyecto a seleccionar ER/Studio en su versión 7.1 para el modelado de la base de datos, teniendo en cuenta además que brinda opción de repositorio, factor de gran importancia para el trabajo en equipo.

1.7.4 Herramienta para el mapeo de datos

1.7.4.1 Doctrine

Doctrine es un potente y completo sistema Mapeador de Objeto Relacional (ORM) para PHP 5.2.3+ con un DBAL (Capa de Abstracción de Base de Datos) incorporado. Se está empezando a ver su potencial, pero de la documentación se puede decir que tiene todas las características necesarias para ser funcional en casi cualquier proyecto. Entre otros elementos se tiene la posibilidad de exportar una base de datos existente a sus clases correspondientes y también a la inversa, es decir convertir clases (convenientemente creadas siguiendo las pautas del ORM) a tablas de una base de datos. Por otro lado, como la librería es bastante grande ésta tiene un método para ser compilada al pasar a producción.

Ventajas que facilitan enormemente tareas comunes y de mantenimiento:

Reutilización: La principal ventaja que aporta un ORM es la reutilización permitiendo llamar a los métodos de un objeto de datos desde distintas partes de la aplicación e incluso desde diferentes aplicaciones.

Encapsulación: La capa ORM encapsula la lógica de los datos pudiendo hacer cambios que afectan a toda la aplicación únicamente modificando una función.

Seguridad: Los ORM suelen implementar mecanismos de seguridad que protegen nuestra aplicación de los ataques más comunes como una Inyección SQL.

Mantenimiento del código: Gracias al correcto ordenamiento de la capa de datos, modificar y mantener el código es una tarea sencilla.

1.8 ***Conclusiones***

El presente capítulo ofreció una panorámica de la situación actual de los sistemas informáticos para instituciones jurídicas en Cuba y el mundo evidenciándose que en el país cuenta con un escaso desarrollo en esta materia. También se realizó un estudio de las bases de datos, enunciándose algunos conceptos importantes al respecto, analizándose temas de importancia como el uso del SGBD PostgreSQL 8.4 dadas las ventajas que ofrece incluyendo que es una tecnología de código abierto. Además se enuncia las diferentes herramientas case que son utilizadas para un mejor desarrollo de la base de datos, escogiendo como herramienta el Er Studio 7.1 debido a sus diversas ventajas que la diferencian de las mismas, así como los pasos a seguir para una ágil y correcta modelación de la base de datos.

Capítulo 2 Descripción de la Solución

2.1 Introducción

En el presente capítulo se definen las actividades más importantes que posibilitaron la realización del diseño de la base de datos del módulo Penal del proyecto “Sistema para los Tribunales Cubanos” a partir del Modelo de Desarrollo de Bases de Datos, exponiéndose los artefactos generados en la fase de inicio de dicho modelo. Se hará una descripción y fundamentación de la arquitectura de datos y se ofrecerá una descripción detallada de las entidades más importantes en cuestión de funcionalidad para el esquema de la materia Penal.

2.2 Entorno de Desarrollo

Como artefacto general del proceso de desarrollo de software se define el documento “Entorno de Desarrollo de Base de Datos”, el cual forma parte del documento de arquitectura del sistema, con el objetivo de organizar el equipo de desarrollo definiendo los responsables por cada módulo. Además contiene la descripción de las herramientas para desarrollar, la propia configuración del entorno de desarrollo, nomenclatura y estándares definidos. Para mayor información consultar el artefacto “Entorno de Desarrollo de Base de Datos”.

2.2.1 Arquitectura

Como propuesta inicial sobre la estructura de servidores para los tribunales populares cubanos estará conformada por un servidor en cada tribunal independientemente de la instancia en la que se encuentre el mismo. Además existirá un clúster de datos garantizando la comunicación entre los servidores, que facilite la replicación de los datos. Esto permitirá una alta disponibilidad y confiabilidad de los servicios que ofrece de tal manera que estos se brinden ininterrumpidamente, así como un balanceo de carga el cual se encargará de colocar en paralelo varios servidores capaces de brindar el mismo servicio y de alguna forma repartir el trabajo entre ellos, tal que los clientes vean servidas sus solicitudes en tiempos menores y aceptables.

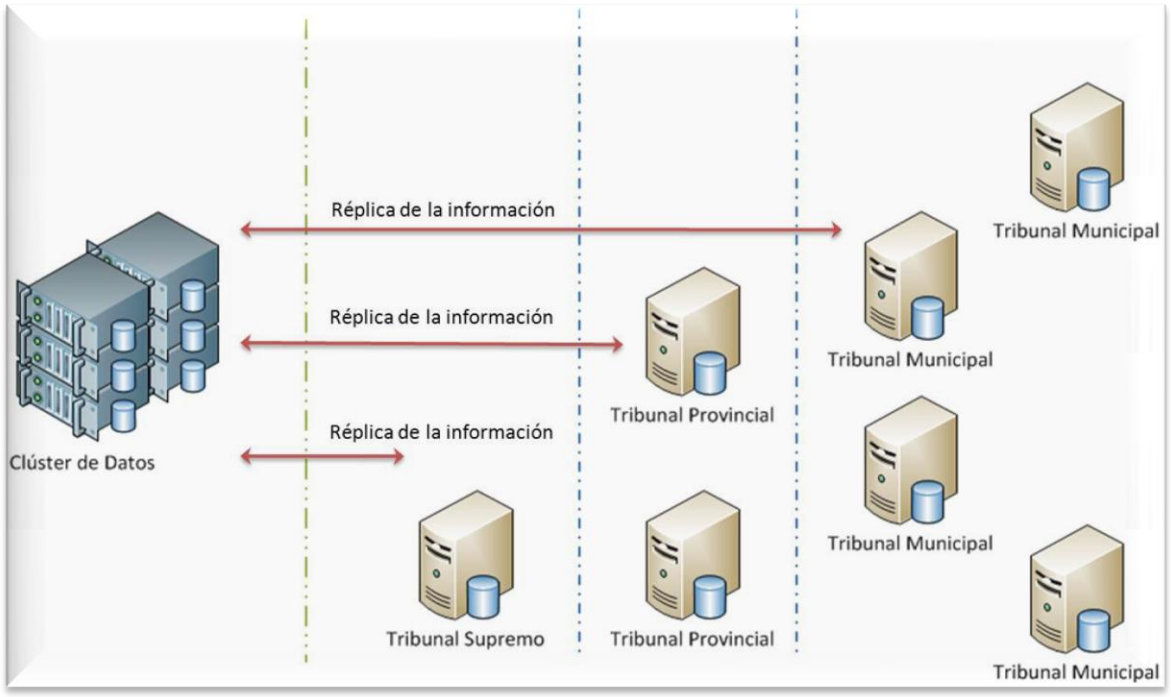


Figura 2 Estructura de Servidores

El sistema para los tribunales cubanos cuenta con la siguiente estructura de servidores: en un servidor Windows estarán instalados el Embarcadero Repositorio 4.0.1 y el SQL Server 2005 para garantizar el repositorio y el trabajo en equipo con la herramienta de modelado ER/Studio 7.1. Se configurará además, una salva automática de la base de datos del repositorio, programada diariamente a las 4 PM. En las estaciones de trabajo estará instalada la herramienta de modelado ER/Studio 7.1 (Fig2)

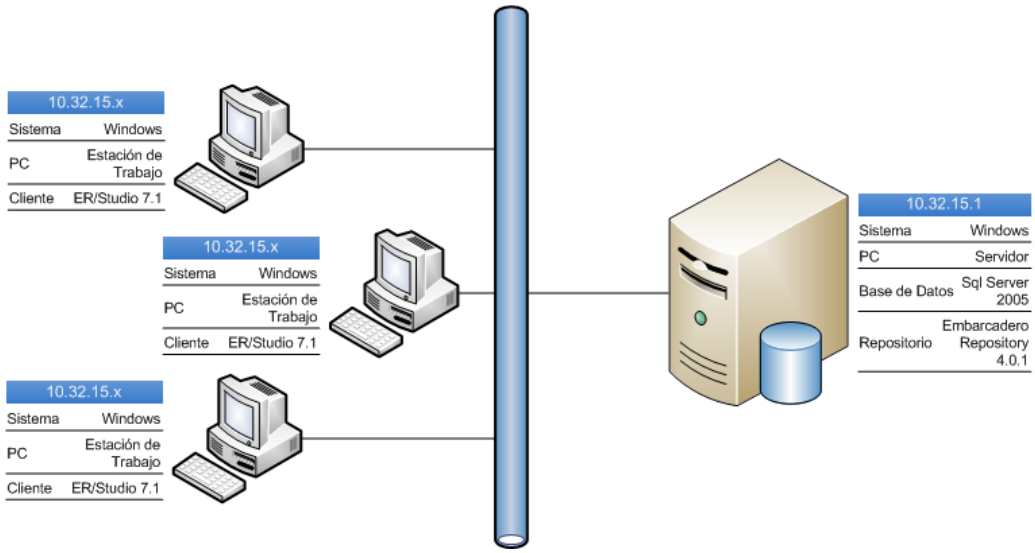


Figura 3 Entorno de desarrollo para el modelado de datos

2.3 Configuraciones

Como parte de las actividades que se realizan en la presente fase se encuentra el establecimiento de las configuraciones iniciales del servidor de base de datos, así como la estructuración de una estrategia de instalación y configuración de la arquitectura de la base de datos para diferentes entornos de desarrollo. El artefacto que se genera es el Documento de configuraciones del sistema operativo – sistema gestor de bases de datos – base de datos. Dicho artefacto contiene los principales elementos a tener en cuenta en el momento de la instalación de los programas necesarios y una vez concluida la misma.

En el servidor se instalará el sistema operativo Debian en su versión 5.0.4, el mismo es un sistema operativo libre, de ahí que su utilización esté en correspondencia con las políticas de la universidad y el país con respecto al desarrollo de software a través de tecnologías de código abierto. Una vez finalizada la instalación deberán configurarse debidamente la hora y la dirección IP del servidor, lo que garantizará un mejor funcionamiento. De igual forma debe conectarse al repositorio de actualizaciones del sistema operativo de donde podrán obtenerse, en su mayoría, los programas y servicios indispensables para el servidor. Un aspecto que no debe quedar al margen es la instalación del servicio Openssh-sever, el mismo facilita la administración remota por consola del servidor.

Otro software que debe quedar instalado para el trabajo con la base de datos es el SGBD PostgreSQL 8.4. En el mismo como medida de seguridad se eliminará el esquema public (público) que por defecto trae

asociado, creándose en su lugar un esquema para cada materia, lo que contribuirá a lograr una mayor organización de la base de datos. Cada uno de los esquemas contará con una cuenta de usuario propia. Para mayor información remitirse al artefacto “Documento de configuraciones del sistema operativo – sistema gestor de bases de datos – base de datos”.

2.4 Optimización de la Base de datos

2.4.1 Normalización

Cuando se va a realizar el diseño de una base de datos relacional, se debe considerar que las tablas que la componen cumplan con las reglas de normalización, que es el proceso que permite eliminar la redundancia de los datos y evitar los problemas al insertar, eliminar y actualizar los datos, es decir, optimizar el trabajo con la información almacenada. El grado de normalización no es el único criterio para calificar lo eficiente que es un esquema relacional. En este proceso se debe ir comprobando que cada relación (tabla) cumple una serie de reglas que se basan en la clave primaria y las dependencias funcionales. Cada regla que se cumple aumenta el grado de normalización. Si una regla no se cumple, la relación se debe descomponer en varias relaciones que sí la cumplan.

En el proceso de normalización existen varios niveles, pero los tres primeros son los más usados. Las reglas de normalización se relacionan entre ellas de forma dependiente, lo que indica que para que una base de datos se encuentre en una determinada forma normal, primeramente tiene que cumplir con las reglas de los niveles inferiores de normalización. Esto quiere decir que una base de datos está en 2da Forma Normal si previamente cumple con las reglas de normalización del 1er nivel y además cumple con las restricciones correspondientes a dicha forma normal.

El esquema de relación del módulo está en 1ra Forma Normal puesto que se puede asegurar que cada tupla contiene exactamente un valor para cada atributo de las tablas de la base de datos, o sea, no existen campos multievaluados. También se cumple que el esquema de relación está en 2da Forma Normal, pues primeramente se encuentran en 1ra Forma Normal, y además todos los atributos que no son claves en las tablas, dependen totalmente de la clave primaria.

Finalmente puede plantearse que las relaciones de las tablas en la base de datos están en 3ra Forma Normal, pues estas se encuentran en 2da Forma Normal, y además no existen dependencias transitivas

atributos no primos, pero hay algunas de las relaciones de estas tablas que se desnormalizaron para lograr flexibilidad en el diseño, la optimización de consultas, mejorar el acceso a los datos.

Ejemplo de ello se puede apreciar en la tabla **dTramite** (tabla 14), la cual contiene dos atributos sala y tribunal los cuales pertenecen a otro módulo Administración y Gobierno pero es necesario tenerlos en los trámites para obtener sus respectivos datos, se pudo crear dos tablas nuevas con esos valores pero era poco recomendable para un mejor desarrollo, ya que se repetirían valores de entidades creadas ya en otros módulos, se considera entonces que esta tabla es un ejemplo de desnormalización.

2.4.2 Estrategia Inicial de Indexado

Las demoras del sistema ante operaciones que involucren un gran volumen de datos pueden ser reducidas. Es posible lograr optimizaciones ya sea por el tipo específico de gestor con que se manejen los datos y sus configuraciones puntuales, por el modelo de datos seleccionado, por configuraciones que se realicen sobre la base de datos, como por las optimizaciones en las consultas. Las técnicas de optimización pueden enfocarse entonces tanto en el nivel físico, por ejemplo, distribuyendo la información en distintos ficheros, discos, o incluso servidores, realizando un mayor número de operaciones en paralelo; como a la hora de proponer un diseño conceptual, seleccionando un modelo con el que se prevean realizar menos operaciones costosas. Muchas veces en la práctica se deciden aplicar varias de estas alternativas juntas. Aunque la intensión no es mencionar cada una de las opciones disponibles para optimizar, sí se quisiera poner a consideración algunas de ellas.

Sobre una base de datos se realizarán, muchas veces, consultas de gran complejidad que solicitarán información que cumpla determinados criterios, es decir, los usuarios frecuentemente querrán especificar los valores con los cuales se filtrarán los datos que deberán ser retornados. La mayoría de estas consultas incluirán, probablemente, operaciones de join entre tablas muy grandes, lo cual puede resultar extremadamente costoso. Para ganar en eficiencia a la hora de realizar estas operaciones se han investigado y creado técnicas especializadas que hoy ofrecen varios gestores, como los ***índices***.

Un índice es una estructura física que permite un tipo de acceso alternativo al secuencial[9]. Es creado a partir de una o varias columnas de una tabla, y por lo general, es construido en forma de árbol balanceado

[9] Este proceso es denominado método de acceso secuencial (*sequential access method*).

(B-Tree). Al ser estructuras físicas, los índices van a tener un fichero asociado, en cuyas páginas se pueden almacenar uno o varios nodos del árbol. Cada uno de ellos apunta hacia otros nodos del árbol o hace referencia a las filas de la tabla. En cada nodo, los valores están ordenados, y los que se encuentran en un nodo hijo son menores o iguales que el valor en el nodo padre que le hace referencia. Los nodos que apuntan hacia las filas reciben el nombre de "páginas hojas", y están enlazados entre sí: una página hoja apunta a otra hoja que contiene el próximo conjunto de valores.

Existe un tipo de índice con el cual se impone que los datos de la tabla estén ordenados en el nivel físico, y reciben el nombre de índices clusterizados (clusteredindex). Para cada tabla sólo se puede especificar un índice clusterizado, pues este afecta la forma en que son almacenadas las filas. Aquellos que no influyen en la organización física se denominan índices no clusterizados y varios pueden ser creados para una misma tabla[10]. (England and Powell 2007).

Las ventajas que tiene el uso de los índices están dadas, precisamente, por su estructura. Por ejemplo, las búsquedas de filas en las que un valor en particular aparezca no implican recorrer toda la tabla, sino que se utiliza la estructura arbórea del índice que se haya definido. Bajando desde la raíz del árbol, sólo es necesario desprenderse por una de las ramas hasta encontrar, en las páginas hojas, las referencias a las filas en el fichero. Con esto se consume menos tiempo en hallar el resultado y es menor la cantidad de veces que se accede al disco para leer.

Se podría pensar entonces que la mejor opción es crear un índice por cada combinación de columnas. Sin embargo, sobre todas ellas en la práctica no se definen buenos criterios de búsqueda, por lo que no deberían crearse estas estructuras innecesariamente. Además, la creación de demasiados índices puede traer consecuencias no deseadas:

- ✓ Si se modifican valores en la tabla asociados a columnas sobre las que se hayan creado índices, o se insertan o eliminan filas, la estructura del índice se actualiza, pues el árbol asociado debe ser consistente con respecto a la información de la tabla. Esto va a influir en el comportamiento del gestor, pudiendo reducir la velocidad de procesamiento a la hora de realizar dichas operaciones. Aunque las operaciones que mayormente serán realizadas en una base de datos son de lectura,

[10] Para profundizar en el tema de los índices clusterizados y no clusterizados, se recomienda consultar (England and Powell 2007).

esto se debe tener en cuenta a la hora de realizar las cargas hacia el sistema, donde las operaciones de inserción y modificación son abundantes. Una alternativa que se podría analizar es eliminar los índices antes de comenzar la carga y volverlos a crear después.

- ✓ Como los índices se almacenan en ficheros al igual que los datos de una tabla, van a ocupar espacio de almacenamiento físico. Mientras más grande sea una tabla, mayores serán los índices asociados a ella. Por lo tanto, se debe analizar la capacidad de almacenamiento de que se dispone.

La solución más apropiada es decidir cuáles índices implicarán una mejora significativa en el rendimiento del sistema ante consultas. Algunas instrucciones que se pueden seguir son:

- ✓ Crear índices para las llaves primarias y foráneas: debido a que las operaciones de join consumen mucho tiempo, y para la mayoría de ellos las columnas por las que se realiza la unión son llaves foráneas, crear índices en las llaves implicadas en la unión puede ser ventajoso.

- ✓ Definir índices para las columnas incluidas en criterios de selección: si frecuentemente se deben seleccionar las filas de una tabla, filtrando por valores de una columna, es conveniente que dicha columna tenga definido un índice. Pero un criterio más fuerte que la frecuencia de consulta, lo brindan el número de filas en la tabla (cardinalidad de la tabla) y el número de valores diferentes en la columna (cardinalidad de la columna): el impacto de un índice es generalmente mayor mientras mayor sea la cardinalidad de la tabla y/o de la columna.

La mayoría de los Sistemas Gestores de Bases de Datos proporcionan herramientas de prueba y valuación para determinar la efectividad de un índice, con las cuales, luego de creado, se puede determinar si traerá mejoras significativas en el sistema. Debido a la complejidad de muchas consultas que involucran realizar operaciones de *joins (unión)* entre tablas grandes, algunas formas especiales de índices han sido desarrolladas para agilizar este tipo de consultas. Algunos gestores los han incorporado, permitiendo lograr mayor eficiencia en los tiempos de respuesta ante solicitudes con propósitos analíticos.

Los índices multitabla o índices join, por ejemplo, permiten definir índices sobre columnas de dos o más tablas. Desde el punto de vista físico, la modificación con respecto a los índices antes explicados es que las referencias de las páginas hojas apuntan a varias filas en tablas diferentes. Esto mejora notoriamente

las operaciones de join donde participen dichas columnas. Otros índices son los de columnas virtuales, también denominados índices basados en funciones, que dan la posibilidad de definir índices sobre una expresión más allá que sobre columnas. En lugar de almacenar en el árbol los valores que aparecen en la columna, primero la expresión especificada es calculada y luego guardada en dicho árbol. Las hojas apuntan a las filas en las cuales el resultado de la expresión es igual al almacenado. La principal ventaja que ofrecen es la mejoría de la velocidad de procesamiento en consultas donde se utilice la expresión para filtrar. Existen también otras formas especiales de índices que se basan en estructuras diferentes del **B-Tree**, como el *índice Hash* y *el Bitmap*, este último utilizado generalmente cuando la cardinalidad de la columna es baja.

La solución de la base datos para el subsistema Penal posee implementado un indexado, el que trae por defecto el gestor PostgreSQL para la búsqueda de datos utilizando las llaves primarias y foráneas. Todas las llaves primarias, que son llaves subrogadas, poseen índices de tipo “B-Tree” (Árboles-B) lo que implica que cualquier búsqueda que se realice utilizando las llaves se optimizará mediante este método. El ejemplo más claro que posee el sistema para entender este funcionamiento, es la relación existente entre las tablas de hechos con las dimensiones ya que utiliza el método “B-Tree” para optimizar la recuperación de los datos cuando se le realicen cortes a la información almacenada. La utilización de más índices sobre la tabla de hechos u otras estructuras no es una tarea sencilla debido a que un consumo excesivo de índices lejos de mejorar el rendimiento del sistema atenta de forma directa a su ralentización y crecen.

2.5 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.

Un patrón de diseño es una solución a un problema de diseño. 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 **reusable**, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

2.5.1 Patrón máquina de Estado para escenarios

Este modelo representa la ocurrencia del cambio de estado en un escenario de una entidad dada, por lo tanto considera el tiempo y la persistencia del mismo en las tablas resultantes. También representa la

ocurrencia de un estímulo en una fecha y los estados por los que ha pasado, caracterizados por la fecha de inicio y la fecha fin.

En el caso del modelo planteado es controlado el estado en que se encuentran diferentes entidades ya sean escritos, expedientes, resoluciones y se tiene en cuenta el origen y el destino de esos estados.

2.5.2 Llaves subrogadas

Este patrón es muy utilizado pues se decide generar una llave primara única para cada entidad en vez de usar un atributo identificador en el contexto dado. Normalmente se usa enteros en columnas identity (identidad) que está demostrado que no se repiten. Permite que las tablas sean más fáciles de consultar por el identificador dado que se conoce el mismo tipo de todos en cada tabla.

2.6 Modelo de Datos

2.6.1 Diseño de la base de datos.

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

2.6.2 Diagrama Entidad relación de la base de datos del subsistema Penal

La base de datos propuesta para el módulo Penal contiene ochenta y nueve tablas en modelo físico, las principales tablas son: Escrito, Persona Natural, Abogado, Acusado, Acusado-Abogado, Diligencia, Causa Documento, Resolución-Judicial, Testigo, Expediente-preparatorio las cuales recogen la información de mayor peso en el módulo. Dentro de las ochenta y nueve tablas existen 41 nomencladores. (Ver anexo1)

Es necesario mencionar que el diseño de la base de datos para el subsistema Penal está sujeto a cambios debido a que el proyecto se encuentra inmerso en diferentes etapas de desarrollo por lo que este modelo propuesto no será el último ya que sufrirá cambios a medida del avance del proyecto.

2.6.3 Descripción de las principales tablas del diagrama entidad relación de la base de datos del subsistema Penal

2.6.3.1 Relación de las principales tablas que conforman la Persona

En la siguiente figura se muestra las tablas que influyen en la creación de una persona. Parte desde la tabla dPersona, la cual contiene dos tipos de personas, dPersonaNatural guardando esta los datos principales de una persona común, como primer nombre, segundo nombre entre otros datos de vital importancia para identificar a una persona y dPersonaJurídica que son las entidades en general(UCI, Tribunal Supremo Popular). La tabla dPersonaNatural contiene una relación con la tabla dAcusado, que además de tener todos los datos de la persona natural a través de la llave primaria IdPersona, también tendrá atributos específicos de un acusado como sobrenombre, raza, causa, el nombre de la madre, el nombre del padre, estos dos últimos se obtendrán a través de la persona natural por el IdMadre y el IdPadre. Entre otras relaciones que contiene la persona natural se encuentra, centro de trabajo a la que esta pertenece, dirección de la misma y el abogado que será el encargado de defender al acusado.

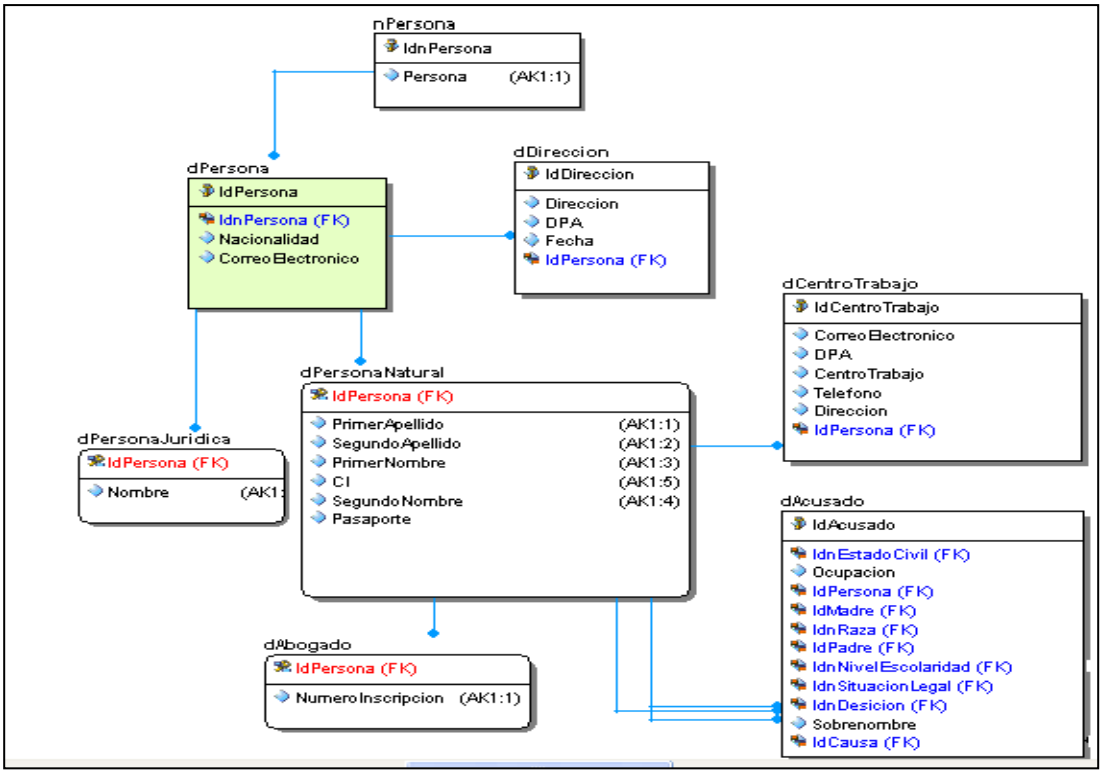


Figura 4 Relación Persona

A continuación se muestran las descripciones de las principales tablas de esta relación

dPersona				
Descripción: Personas				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdPersona		SERIAL/INTEGER	No	Identificador de la persona. Llave primaria
Nacionalidad	Texto	TEXT	Si	Nacionalidad de la persona
CorreoElectronico	Email	TEXT	No	Correo Electrónico de la persona
IdnPersona	EnteroGrande	BIGINT	No	Identificador del tipo de persona. Llave foránea

Tabla 1 Persona

dPersonaNatural				
Descripción: Datos de las personas naturales				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdPersona		INTEGER	No	Identificador de la persona. Llave primaria
Pasaporte	Texto	TEXT	Si	Pasaporte de la persona en caso que no sea cubana
CI	NumeroCI	Varchar(11)	Si	Carnet de Identidad de la persona
PrimerApellido	Texto	TEXT	No	Primer apellido de la persona
SegundoApellido	Texto	TEXT	No	Segundo apellido de la persona
PrimerNombre	Texto	TEXT	No	Primer nombre de la persona
SegundoNombre	Texto	TEXT	Si	Segundo nombre de la persona

Tabla 2 Persona Natural

dPersonaJuridica				
Descripción: Personas jurídicas (Entidades, Empresas)				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdPersona		INTEGER	No	Identificador de la persona. Llave primaria
Nombre	Texto	TEXT	No	Nombre de la entidad

Tabla 3 Persona Jurídica

dDireccion				
Descripción: Dirección de las personas				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdDireccion		SERIAL/INTEGER	No	Identificador de la dirección. Llave primaria
Pais	Texto	TEXT	No	País
Direccion	Texto	TEXT	No	Última dirección conocida
DPA	Texto	TEXT	Si	División Política Administrativa
Fecha	FechaHoraActual	TIMESTAMP/DATE	No	Fecha de la última dirección conocida
IdPersona		INTEGER	No	Identificador de la persona. Llave foránea

Tabla 4 Dirección

2.6.3.2 Relación de las tablas que conforman el código Penal

En la siguiente figura se refleja la relación de tablas que conforman el código penal. Las sanciones propuestas por el juez guardadas en la tabla nSancion, relacionada a su vez con la tabla nTipoSancion (Accesoria o Principal) y con la tabla nPersonaSancion la cual muestra cuales son las sanciones aplicables a las personas, jurídica o natural. Estas sanciones son dadas por determinado delito los cuales son guardados en la tabla nDelito, relacionada con la entidad que especifica los tipos de delitos nombrada nTipoDelito. La tabla nArticulo mostrará entonces los artículos de la ley referente a los delitos y sanciones mencionados anteriormente, con el número, el apartado y el inciso del mismo así como el rango de la sanción (mínimo y máximo). En la tabla nUnidad se guardará la unidad de medida (meses o años) de la sanción que se le está imputando.

La ley del código penal está compuesta además por eximentes y adecuantes estas últimas pueden ser adecuativas, agravantes y atenuantes. Son pronunciamientos de la ley q permiten ajustar los marcos legales de los delitos, en el caso de las eximentes provocan que no se le exija la responsabilidad penal a pesar de cometer el delito, las adecuativas son para adaptar los límites del delito, la atenuante para rebajar la sanción y la agravante para agravar la sanción impuesta.

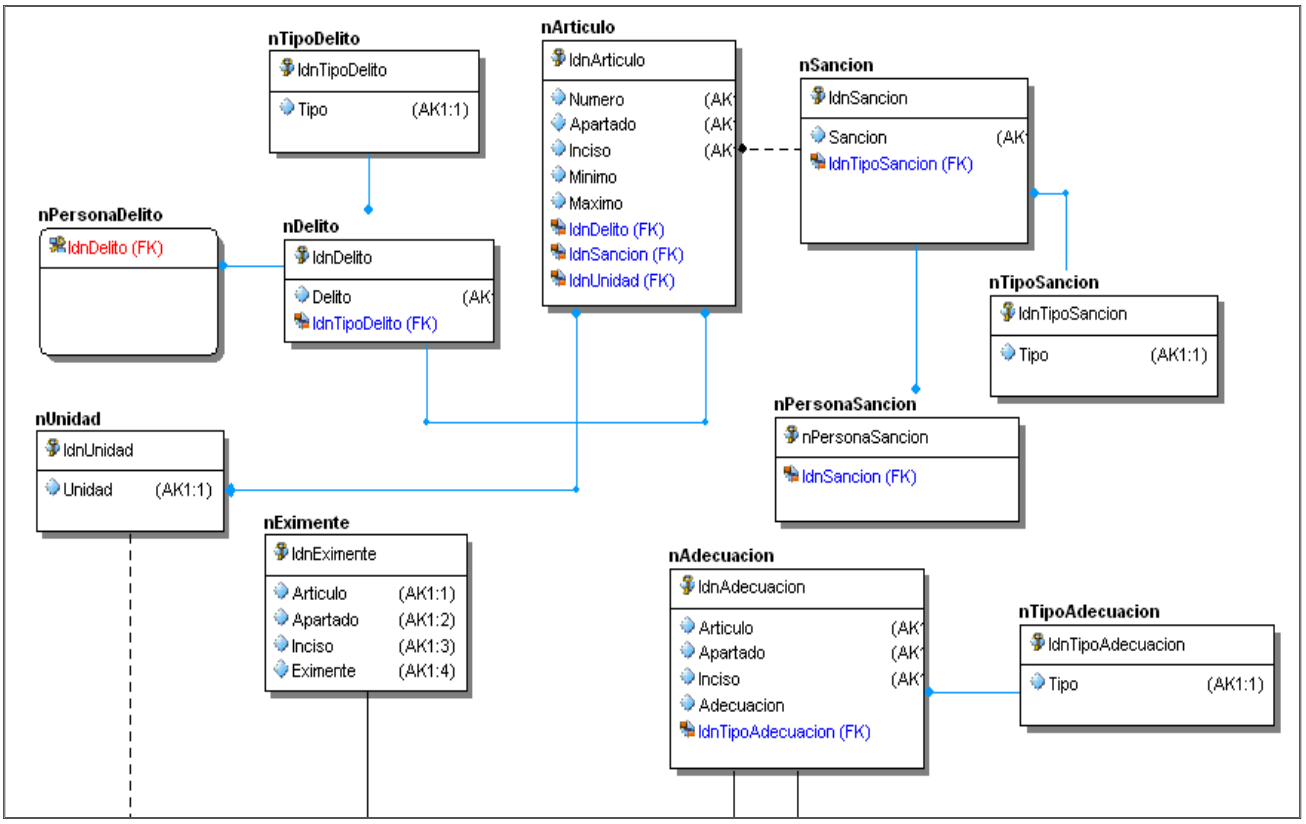


Figura 5 Código Penal

NArticulo				
Descripción: Artículo de la ley referente a los delitos y las sanciones				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdnArticulo	EnteroGrande	BIGINT	No	Identificador del artículo. Llave primaria
Numero	EnteroGrande	BIGINT	No	Número del artículo
Apartado	EnteroGrande	BIGINT	Si	
Inciso	Texto	TEXT	Si	

Minimo	EnteroGrande	BIGINT	Si	
Maximo	EnteroGrande	BIGINT	Si	
IdnDelito	EnteroGrande	BIGINT	No	Identificador del tipo de delito. Llave foránea
IdnSancion	EnteroGrande	BIGINT	No	Identificador del tipo de sanción. Llave foránea
IdnUnidad	EnteroGrande	BIGINT	Si	Identificador de la unidad. Llave foránea

Tabla 5 Nomencladora Artículo

nDelito				
Descripción: Delitos cometidos				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdnDelito	EnteroGrande	BIGINT	NO	Identificador del delito. Llave primaria
Delito	Texto	TEXT	NO	Nombre del delito
IdnTipoDelito	EnteroGrande	BIGINT	NO	Identificador del tipo de delito. Llave foránea

Tabla 6 Nomencladora Delito

NTipoDelito				
Descripción: Tipos de delitos cometidos				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdnTipoDelito	EnteroGrande	BIGINT	NO	Identificador del tipo de delito. Llave primaria
Tipo	Texto	TEXT	NO	Tipo de delito (Homicidio)

Tabla 7 Nomencladora tipo de Delito

NSancion				
Descripción: Sanciones aplicadas dependiendo del delito cometido.				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdnSancion	EnteroGrande	BIGINT	NO	Identificador de la sanción. Llave primaria
Sancion	Texto	TEXT	NO	Nombre de la sanción
IdnTipoSancion	EnteroGrande	BIGINT	NO	Identificador del tipo de sanción. Llave foránea

Tabla 8 Nomencladora Sanción

NTipoSancion				
Descripción: Tipos de sanciones				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdnTipoSancion	EnteroGrande	BIGINT	NO	Identificador del tipo de sanción. Llave primaria
Tipo	Texto	TEXT	NO	Tipo de sanción(Accesoria o Principal)

Tabla 9 Nomencladora tipo de sanción

Además existen otras tablas que son consideradas importantes para el diseño de la base de datos del subsistema penal como son:

dTramite				
Descripción: Tabla donde se registran los actos procesales que se van realizando.				
Atributo	**Dominio**	**Tipo de texto**	**Nulo**	**Descripción**
IdTramite		SERIAL/INTEGER	NO	Identificador del trámite. Llave primaria.

MAC	MAC	INTEGER	NO	Número MAC
IP	IP	INTEGER	NO	Dirección IP
Sala	Texto	TEXT	NO	
Tribunal	Texto	TEXT	NO	
Fecha	FechaHoraActual	TIMESTAMP/DATE	NO	
IdnTramite	EnteroGrande	BIGINT	NO	Identificador del nomenclador trámite. Llave foránea
IdPersona		INTEGER	NO	Identificador de la persona. Llave foránea

Tabla 10 Trámite

2.7 Seguridad de la base de datos

Las bases de datos están bajo constante amenaza de sufrir ataques de personas o sistemas no autorizados que ponen en riesgo su integridad y confidencialidad; es por ello que 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 pues ellos constituyen un recurso valioso que debe ser estrictamente controlado y gestionado al igual que cualquier otro recurso corporativo.

Los sistemas gestores de bases de datos 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 autenticados. Lo primero que debe garantizarse es que sólo puedan acceder a los datos los usuarios autorizados.

La configuración de la seguridad de la base de datos está implementada en dos niveles. 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 base de datos, 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. Para mayor información consultar artefacto “Documento de configuraciones del sistema operativo – sistema gestor de bases de datos – base de datos”.

En el segundo nivel la seguridad se define por usuarios y grupos de usuarios. Los permisos para ejecutar ciertas operaciones son asignados a roles específicos. Se les asigna roles particulares a los miembros del equipo, y a través de esos roles asignados obtienen permiso para ejecutar funciones determinadas en la base de datos. Como a los usuarios no se les asigna permisos directamente, sino que los adquieren a través de su rol (o roles), el manejo de los permisos de cada usuario se convierte en una cuestión de simplemente de asignar los roles apropiados al usuario, esto simplifica las operaciones comunes, como adicionar un usuario. En la solución propuesta los roles y usuarios pertenecientes a estos roles son los que se muestran en la figura.

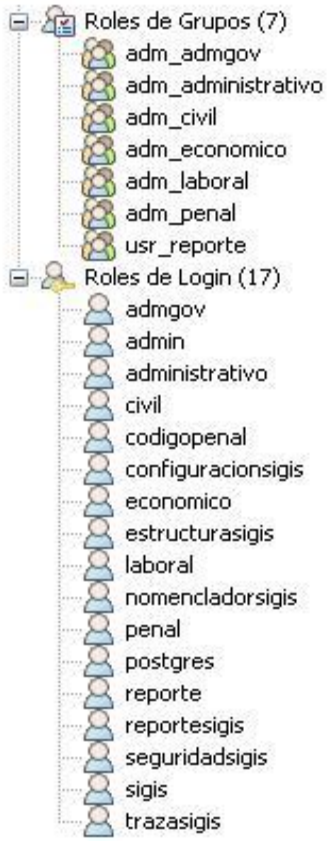


Figura 6 Roles y Usuarios

2.8 Acceso a Datos

Doctrine es un ORM que posee una poderosa capa de abstracción de Bases de Datos. Una de sus características es la opción de escribir consultas de Bases de Datos en un objeto apropiado orientado al dialecto SQL (Structured Query Language), en español lenguaje estructurado de consultas y que se le denomina Lenguaje de Consulta de Doctrine o DQL del inglés Doctrine Query Language, inspirado por el Lenguaje de Consulta de Hibernate (HQL por sus siglas en inglés). Este proporciona a los desarrolladores una poderosa alternativa al SQL que mantiene la flexibilidad sin requerir duplicación de código innecesario.

2.8.1 Doctrine Generator

Doctrine Generator (Generador de Doctrine) es una aplicación para la persistencia de objetos relacionales, basada en el ORM Doctrine, que ofrece funcionalidades de mapeo. Doctrine Generator fue diseñado con

el principio de brindar a los desarrolladores que necesiten persistir su información mediante Doctrine, un ambiente de desarrollo amigable y fácil de usar que cubra sus necesidades respecto a la persistencia de esta información. La herramienta proporciona la posibilidad de generar la misma lógica de negocio que genera el framework[11] Doctrine con un buen rendimiento en cuanto a funcionamiento corrigiendo algunos de sus problemas en las salidas como por ejemplo la ausencia de las relaciones entre las tablas del esquema con el que se está interactuando y la imposibilidad de personalizar estas salidas.

El desarrollo de esta herramienta pretende cubrir parcialmente o en su totalidad los problemas que tienen los equipos de desarrollo del proyecto Tribunales Populares Cubanos, que utilizan el framework Doctrine para la persistencia de datos, a la hora de generar los ficheros de mapeo de los esquemas relacionales, brindándole la posibilidad al desarrollador de configurar a su gusto la forma en que el sistema generará las salidas para los ficheros de mapeo. Para complementar lo antes dicho el sistema permite al desarrollador personalizar su propio mapeo, en cuanto a relaciones que existen entre los objetos presentes en el mismo, relaciones como las de uno a uno, uno a mucho y mucho a mucho. Teniendo en cuenta que en el proyecto se desarrollan las soluciones empleando el lenguaje PHP con PostgreSQL como gestor de bases de datos, la herramienta permite conectividad con este gestor, permitiendo el mapeo de los esquemas existentes y generando los correspondientes ficheros.

Si es la primera vez que se ejecuta, el sistema brinda una interfaz que da la posibilidad de crear un nuevo proyecto. Una vez definido el nombre se puede configurar la conexión con el gestor antes mencionado y obtener una lista de los esquemas a los que se tiene acceso de acuerdo a los datos introducidos cuando se configuró la conexión, brindando la opción de seleccionar por esquema las tablas que se desean mapear haciendo así más personalizado el trabajo de los desarrolladores. Una vez definido estos aspectos se cargan los objetos seleccionados para el mapeo.

A continuación se muestran figuras explicando la realización del mapeo:

[11] Macro de Trabajo

Figura 7 Opciones de conexión**.**

Figura 8 Tablas para Mapear.

El proyecto es guardado en la carpeta que se creó cuando se comenzó con el mapeo (*.dg) y puede ser reeditado posteriormente. El sistema genera la siguiente estructura de carpeta y brinda la facilidad de seleccionar el lugar donde se generará:

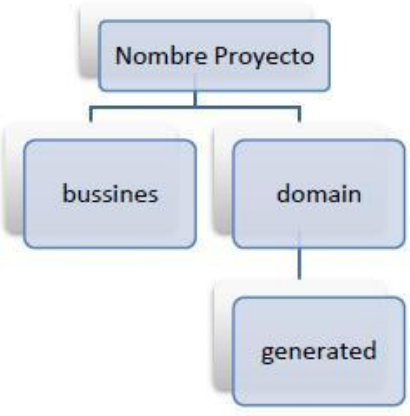


Figura 9 Estructura de carpetas generadas por Doctrine

Para cada objeto el sistema generará las clases (*.php) con el prefijo “Base” seguido del nombre de la clase que es el nombre del objeto; con el sufijo “Model” antecedido por el nombre de la clase y otra con el nombre del objeto dentro de las carpetas “generated”, “bussines” y “domain” respectivamente. En las clases del dominio es donde se implementan las funcionalidades con DQL.

2.9 Conclusiones

En el presente capítulo se ha descrito detalladamente la descripción de la solución desarrollada para la base de datos del módulo Penal y la arquitectura propuesta para su desarrollo, junto con las configuraciones de la base de datos, la seguridad para interactuar y trabajar con las herramientas y gestores, así como la explicación del artefacto entorno de desarrollo el cual define las pautas a seguir para un buen diseño del modelo de datos. Se plasma las descripciones de las principales tablas del modelo lógico, además de los patrones utilizados para el diseño de la misma. Se muestra los pasos a seguir para una buena optimización de la base de datos a través de la normalización y el uso de índices y se explican los pasos a tener en cuenta para realizar el acceso a datos

Capítulo 3 Validaciones

3.1 Introducción

Durante el proceso de desarrollo de un software, los errores pueden comenzar a aparecer incluso desde el mismo momento en que fueron definidos los objetivos y estos a su vez especificados de forma errónea e imperfecta, de la misma forma en los posteriores pasos del diseño y desarrollo.

En el capítulo anterior se planteó una propuesta de solución para el diseño y configuración de la Base de Datos del módulo Penal del proyecto “Sistema para los Tribunales Cubanos”, en el presente capítulo se realiza la validación de la misma, abordando aspectos como la integridad, seguridad y normalización de la Base de Datos.

3.2 Validación teórica del diseño

Se hace imprescindible lograr un correcto diseño de la base de datos. Debe tenerse en cuenta que un buen diseño, debe garantizar sobre todo la consistencia y seguridad de los datos, la no redundancia de los mismos, y que no ocurran, en general, las llamadas “anomalías de actualización”. Para ello deben considerarse elementos como las restricciones de integridad, la normalización de la base de datos y la seguridad de la misma.

3.2.1 Restricciones de integridad de la BD

Establecer las reglas correspondientes a la consistencia de los datos requeridos en las tablas, chequeo de la unicidad de determinados atributos, así como las restricciones correspondientes a las llaves primarias y foráneas son aspectos a los que hay que prestar especial atención para garantizar la integridad de los datos.

La integridad de datos pertenece a una de las siguientes categorías:

- ✓ Integridad de entidad: la clave primaria de una entidad no puede tener valores nulos y siempre deberá ser única, por ejemplo el CI en la tabla Persona Natural.
- ✓ Integridad de dominio: restringe los valores que puede tomar un atributo respecto a su dominio, por ejemplo, la tabla nMateria sólo puede tomar los valores de las 5 materias que pueden existir en un tribunal (Económico, Administrativo, Laboral, Civil, Penal).

- ✓ Integridad referencial: La base de datos no debe contener valores de llaves ajenas sin concordancia.
- ✓ Datos Requeridos: establece que una columna tenga un valor no nulo.

Integridad de Entidad

Esta regla se aplica a las claves primarias de las relaciones base: ningún atributo que forme parte de una llave primaria puede aceptar valores nulos. La unicidad de la clave primaria se refiere a que dos tuplas no pueden ofrecer el mismo valor para la clave primaria. Para expresar estas restricciones pueden utilizarse determinados elementos del lenguaje SQL como PRIMARY KEY (llave primaria), y UNIQUE (única). Este último para expresar la unicidad los atributos que puedan constituir claves alternativas.

Es por esto que en muchas de las tablas de cada uno de los esquemas de la base de datos del sistema para los Tribunales Populares Cubanos, generan las llaves de forma automática usando secuencias, garantizando así que no se cometan errores, uniformidad en la creación de la llaves, además el principal objetivo de la existencia de las mismas está dado por el tema de las réplicas entre los servidores.

Integridad de Dominio

La integridad de dominio viene dada por el conjunto de valores posibles para un atributo determinado. Puede exigir la integridad de dominio restringiendo los valores de un atributo mediante tipos de datos, reglas y restricciones de verificación, definiciones por defecto y definiciones no nulas. Un dominio de valores puede estar asociado a cada atributo. Los límites de dominio son la forma más elemental de restricciones de integridad. Son fáciles de probar en el sistema siempre que se introduce un nuevo dato.

Un ejemplo ilustrativo de algunas de las restricciones de dominio implementadas se expone a continuación, indicando el dominio, tipo de dato correspondiente al mismo, campos a los que será aplicado así como la restricción perteneciente a cada uno de ellos. Para mayor información consultar el artefacto “Entorno de Desarrollo de Bases de Datos”.

Dominio	Tipo de Dato	Campo	Restricción
Binario	BINARY	Documentos, imágenes, etc.	

Bool	BIT	Verdadero o Falso	@var = TRUE or @var = FALSE
Contraseña	TEXT	Contraseña de usuario	@var like '%_____%'
EMail	TEXT	Correo electrónico	@var like '%_@___%'
Entero	INTEGER	Números enteros positivos	@var> 0
FechaHora	DATETIME	Fecha y hora	A nivel de tabla comparando las fechas: FechaInicio<= FechaFin
Nombre	VARCHAR(30)	Nombre hasta 30 caracteres	
Numero	NUMERIC	Valores numéricos y decimales	@var> 0
NumeroCI	VARCHAR	Número de carnet de identidad	@var like '_____'
Texto	TEXT	Texto en general	

Figura 10 Diccionario de datos

Integridad Referencial

La integridad referencial significa que la clave externa de una tabla de referencia siempre debe aludir a una fila válida de la tabla a la que se haga referencia, garantizando que los valores de clave sean coherentes en las distintas tablas. Para conseguir esa coherencia, es preciso que no haya vínculos a valores inexistentes y que, si cambia el valor de una clave, todas las referencias a ella se cambien en consecuencia en toda la base de datos.

La integridad referencial permite que la relación entre dos tablas permanezca sincronizada durante las operaciones de actualización y eliminación. Debe proporcionar Durabilidad, Aislamiento, Consistencia e Indivisibilidad a la base de datos.

- ✓ **Indivisibilidad**: es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.

- ✓ **Consistencia**: es la propiedad que asegura que sólo se empieza aquello que se puede acabar. Por lo tanto se ejecutan aquellas operaciones que no van a romper la reglas y directrices de integridad de la base de datos.
- ✓ **Aislamiento**: es la propiedad que asegura que una operación no puede afectar a otras. Esto asegura que dos transacciones sobre la misma información nunca generarán ningún tipo de error.
- ✓ **Durabilidad**: es la propiedad que asegura que una vez realizada la operación, ésta persistirá y no se podrá deshacer aunque falle el sistema.

Un ejemplo de lo antes expuesto es que se tiene una tabla dPersonaNatural que tiene como llave primaria IdPersona, la cual se obtuvo a su vez de la tabla dPersona, además se encuentra la tabla dAcusado quien tiene como llave primaria IdAcusado. La relación que se establece entre estas tablas es de uno a muchos (ya que una persona natural puede tener asociados varios acusados pero estos sólo podrán pertenecer a una persona natural) razón por la cual Idpersona pasa para la tabla Tribunal como llave foránea. De esta forma se conocería a que persona pertenece un determinado acusado.

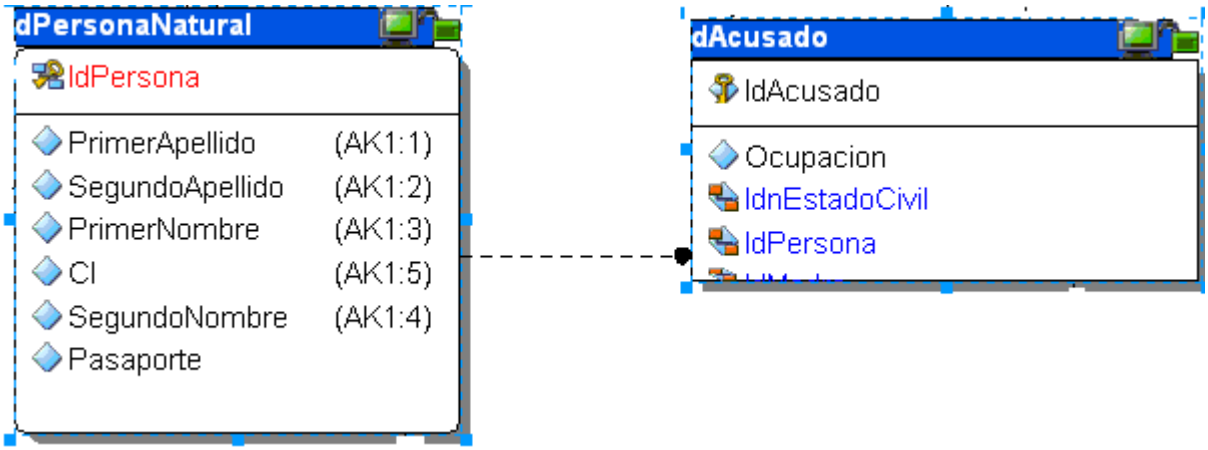
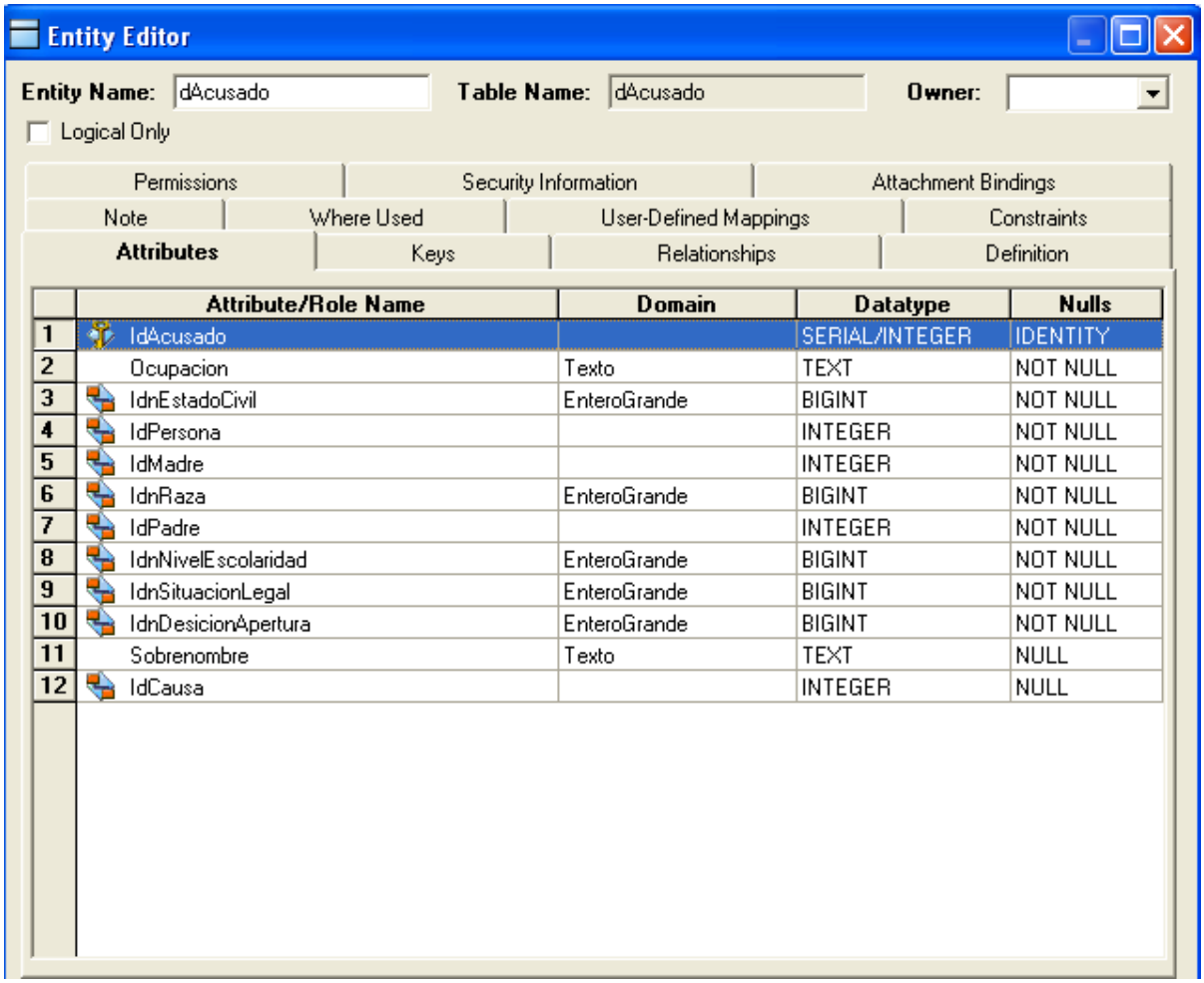


Figura 11 Ejemplo de integridad referencial entre dos tabla

Datos Requeridos

Se define efectuando que la declaración de una columna es no nula cuando la tabla que contiene las columnas se crea por primera vez, como parte de la sentencia crear tabla.

En la mayoría de los casos resultaría inútil declarar que un campo acepte valores nulos si se quiere guardar información importante en el mismo, sin embargo en otros casos resulta necesario, por ejemplo, la tabla dAcusado contiene el atributo Sobrenombre, el mismo acepta valores nulos pues en el caso del acusado, puede presentar un sobrenombre o no.



	Attribute/Role Name	Domain	Datatype	Nulls
1	IdAcusado		SERIAL/INTEGER	IDENTITY
2	Ocupacion	Texto	TEXT	NOT NULL
3	IdnEstadoCivil	EnteroGrande	BIGINT	NOT NULL
4	IdPersona		INTEGER	NOT NULL
5	IdMadre		INTEGER	NOT NULL
6	IdnRaza	EnteroGrande	BIGINT	NOT NULL
7	IdPadre		INTEGER	NOT NULL
8	IdnNivelEscolaridad	EnteroGrande	BIGINT	NOT NULL
9	IdnSituacionLegal	EnteroGrande	BIGINT	NOT NULL
10	IdnDesicionApertura	EnteroGrande	BIGINT	NOT NULL
11	Sobrenombre	Texto	TEXT	NULL
12	IdCausa		INTEGER	NULL

Figura 12 Datos requeridos de una tabla

3.3 Pruebas de volumen y rendimiento

Las pruebas de volumen son pruebas típicas de entornos que utilicen bases de datos. Las mismas se realizan para analizar el comportamiento del sistema o base de datos con volúmenes de datos almacenados lo más similar posible a los esperados en la explotación real del sistema. Para el sistema en cuestión la BD se pobló con datos aleatorios generados por una herramienta llamada EMS Data Generator (Generador de Datos EMS), esta herramienta colma a la base de datos de una cantidad determinada de datos previamente establecida por quién esté realizando el llenado de datos, en este caso, el propio desarrollador. Se configuró esta generación con datos arbitrarios, pero coincidentes en cuanto a sus tipos y volúmenes con los datos reales que maneja la entidad. El uso de este generador se pudiera considerar una prueba más, ya que si existen inconsistencia en el diseño de la Base de Datos, este no comienza el poblado de datos hasta tanto no quede un correcto diseño.

Al introducir los datos no se presentaron problemas de límite de capacidad, ni de volumen de datos. Tampoco se detectaron desbordamientos de matrices, columnas, atributos, tipos de datos, ni peticiones excesivas de memoria. Las llaves autogeneradas no se salieron del rango especificado, ni se detectaron problemas con los tipos de datos definidos en el paso de diseño. Lo anteriormente planteado garantiza que el gestor utilizado y el diseño de las estructuras de la base de datos implementadas soportan completamente el almacenamiento de los niveles de información requeridos.

Para validar la rapidez de respuesta de la base de datos frente a las solicitudes de los usuarios se llevaron a cabo pruebas de rendimiento. Estas pruebas se realizaron implementando consultas SQL a diferentes tablas. A continuación se muestra un cuadro descriptivo que hace referencia a las tablas involucradas así como a la cantidad de filas generadas para cada tabla.

Nombre de la tabla	Cantidad de tuplas
dPersona	100000
dPersonaNatural	4024
dPersonaJuridica	1336
dDireccion	5000

Figura 13 Datos de tablas

Consulta 1

explain analyze **SELECT**

dpersona.nacionalidad

FROM

penal.dpersona,

penal.dpersonanatural

WHERE

dpersona.idpersona = dpersonanatural.idpersona AND

dpersonanatural.ci = '05319002143'

Resultados Arrojados

Salida de datos | Comentar | Mensajes | Historial

	QUERY PLAN text
1	Nested Loop (cost=0.00..105.59 rows=1 width=6) (actual time=1.093..1.136 rows=1 loops=1)
2	-> Seq Scan on dpersonanatural (cost=0.00..97.30 rows=1 width=8) (actual time=1.066..1.107 rows=1 loops=1)
3	Filter: ((ci)::text = '05319002143'::text)
4	-> Index Scan using pk_dpersona on dpersona (cost=0.00..8.28 rows=1 width=14) (actual time=0.018..0.019 rows=1 loops
5	Index Cond: (dpersona.idpersona = dpersonanatural.idpersona)
6	Total runtime: 1.186 ms

Figura 14 Consulta 1 sin utilizar índice

Salida de datos | Comentar | Mensajes | Historial

	QUERY PLAN text
1	Nested Loop (cost=0.00..16.56 rows=1 width=6) (actual time=0.171..0.173 rows=1 loops=1)
2	-> Index Scan using ci on dpersonanatural (cost=0.00..8.27 rows=1 width=8) (actual time=0.132..0.133 rows=1 loops=1)
3	Index Cond: ((ci)::text = '05319002143'::text)
4	-> Index Scan using pk_dpersona on dpersona (cost=0.00..8.28 rows=1 width=14) (actual time=0.030..0.031 rows=1 loops=1)
5	Index Cond: (dpersona.idpersona = dpersonanatural.idpersona)
6	Total runtime: 0.247 ms

Figura 15 Consulta 1 con utilización de índice

Se presenta a continuación los resultados arrojados utilizando una consulta de más complejidad.

Nombre de la tabla	Cantidad de tuplas
dExpedientePreparatorio	5000
dTramite	1000
dPiezaConviccion	1000
dEscrito	1000
dEscritoAcusado	1000
dAcusado	1000

Figura 16 Datos de las tablas

Consulta 2

```
explain analyze

SELECT

SUB.idexpedientepreparatorio, SUB.numero, cant_piezas, E.fecharadicado AS fecharadicado,

   COUNT(A.idacusado) AS cant_acusados , COUNT(N.situacionlegal) as cant_acusadospp

   FROM (SELECT SUB1.idexpedientepreparatorio , SUB1.numero , COUNT(P.idexpedientepreparatorio)
AS cant_piezas

   FROM (SELECT EX.idexpedientepreparatorio, EX.numero

   FROM penal.dexpedientepreparatorio EX

   INNER JOIN penal.dexpedienteultimotramite EXU ON EX.idexpedientepreparatorio =
EXU.idexpedientepreparatorio

   INNER JOIN penal.dtramite T ON EXU.idtramite = T.idtramite

   INNER JOIN penal.ntramite NT ON T.idntramite = NT.idntramite

   INNER JOIN penal.nestadoactoprocesal NEA ON NT.idnestadodestino = NEA.idnestadoactoprocesal

   WHERE NEA.estado = 'Pendiente a dar presentado') SUB1

   LEFT JOIN penal.dpiezaconviccion P ON SUB1.idexpedientepreparatorio = P.idexpedientepreparatorio

   GROUP BY SUB1.numero, SUB1.idexpedientepreparatorio) SUB
```

INNER JOIN penal.descritoexpedientepreparatorio EEX ON SUB.idexpedientepreparatorio = EEX.idexpedientepreparatorio

INNER JOIN penal.descrito E ON EEX.idescrito = E.idescrito

INNER JOIN penal.descritoacusado EA ON E.idescrito = EA.idescrito

INNER JOIN penal.dacusado A ON EA.idacusado = A.idacusado

LEFT JOIN penal.nsituacionlegal N ON A.idnsituacionlegal = N.idnsituacionlegal AND N.situacionlegal = 'Prisión Provisional'

9	-> Nested Loop (cost=6.74..10.26 rows=1 width=64) (actual time=0.219..0.298 rows=4 loops=1)
10	Join Filter: (ea.idacusado = a.idacusado)
11	-> Nested Loop (cost=6.74..9.01 rows=1 width=56) (actual time=0.210..0.243 rows=4 loops=1)
12	Join Filter: (eex.idescrito = ea.idescrito)
13	-> Hash Join (cost=6.74..7.83 rows=1 width=40) (actual time=0.201..0.206 rows=3 loops=1)
14	Hash Cond: (eex.idexpedientepreparatorio = ex.idexpedientepreparatorio)
15	-> Seq Scan on descritoexpedientepreparatorio eex (cost=0.00..1.06 rows=6 width=16) (actual time=0.007..0.009 rows=6 loops=1)
16	-> Hash (cost=6.73..6.73 rows=1 width=24) (actual time=0.182..0.182 rows=3 loops=1)
17	-> HashAggregate (cost=6.70..6.72 rows=1 width=24) (actual time=0.177..0.179 rows=3 loops=1)
18	-> Nested Loop Left Join (cost=4.36..6.70 rows=1 width=24) (actual time=0.121..0.161 rows=6 loops=1)
19	Join Filter: (ex.idexpedientepreparatorio = p.idexpedientepreparatorio)
20	-> Hash Join (cost=4.36..5.54 rows=1 width=16) (actual time=0.114..0.124 rows=3 loops=1)
21	Hash Cond: (ex.idexpedientepreparatorio = exu.idexpedientepreparatorio)
22	-> Seq Scan on dexpedientepreparatorio ex (cost=0.00..1.12 rows=12 width=16) (actual time=0.002..0.006 rows=12 loops=1)
23	-> Hash (cost=4.35..4.35 rows=1 width=4) (actual time=0.103..0.103 rows=3 loops=1)
24	-> Nested Loop (cost=2.12..4.35 rows=1 width=4) (actual time=0.072..0.100 rows=3 loops=1)
25	Join Filter: (t.idtramite = exu.idtramite)
26	-> Hash Join (cost=2.12..3.24 rows=1 width=8) (actual time=0.034..0.045 rows=6 loops=1)
27	Hash Cond: (t.idntramite = nt.idntramite)
28	-> Seq Scan on dtramite t (cost=0.00..1.08 rows=8 width=16) (actual time=0.002..0.005 rows=8 loops=1)
29	-> Hash (cost=2.11..2.11 rows=1 width=8) (actual time=0.023..0.023 rows=1 loops=1)
30	-> Nested Loop (cost=0.00..2.11 rows=1 width=8) (actual time=0.015..0.020 rows=1 loops=1)
31	Join Filter: (nt.idnestadodestino = nea.idnestadoactoprocesal)
32	-> Seq Scan on nestadoactoprocesal nea (cost=0.00..1.04 rows=1 width=8) (actual time=0.006..0.007 rows=1 loops=1)
33	Filter: (estado = 'Pendiente a dar presentado'::text)
34	-> Seq Scan on ntramite nt (cost=0.00..1.03 rows=3 width=16) (actual time=0.002..0.003 rows=3 loops=1)
35	-> Seq Scan on dexpedienteultimotramite exu (cost=0.00..1.05 rows=5 width=8) (actual time=0.001..0.003 rows=5 loops=6)
36	-> Seq Scan on dpiezaconviccion p (cost=0.00..1.07 rows=7 width=8) (actual time=0.001..0.004 rows=7 loops=3)
37	-> Seq Scan on descritoacusado ea (cost=0.00..1.08 rows=8 width=16) (actual time=0.001..0.004 rows=8 loops=3)
38	-> Seq Scan on dacusado a (cost=0.00..1.11 rows=11 width=16) (actual time=0.001..0.005 rows=11 loops=4)
39	-> Seq Scan on descrito e (cost=0.00..1.06 rows=6 width=16) (actual time=0.001..0.003 rows=6 loops=4)
40	-> Seq Scan on nsituacionlegal n (cost=0.00..1.08 rows=1 width=40) (actual time=0.003..0.003 rows=0 loops=4)
41	Filter: (n.situacionlegal = 'Prisión Provisional'::text)
42	Total runtime: 0.909 ms

Figura 17 Resultado de consulta 2 sin índice

11	-> Seq Scan on descritoexpedientepreparatorio eex (cost=0.00..27.70 rows=1770 width=16) (actual time=0.002..0.004 rows=6 loops=1)
12	-> Hash (cost=61.38..61.38 rows=3 width=24) (actual time=0.123..0.123 rows=2 loops=1)
13	-> HashAggregate (cost=61.31..61.35 rows=3 width=24) (actual time=0.120..0.122 rows=2 loops=1)
14	-> Nested Loop Left Join (cost=22.89..61.29 rows=3 width=24) (actual time=0.082..0.106 rows=5 loops=1)
15	-> Nested Loop (cost=22.89..60.06 rows=3 width=16) (actual time=0.076..0.089 rows=2 loops=1)
16	-> Hash Join (cost=22.89..57.29 rows=9 width=4) (actual time=0.068..0.073 rows=2 loops=1)
17	Hash Cond: (exu.idtramite = t.idtramite)
18	-> Seq Scan on dexpedienteultimotramite exu (cost=0.00..27.70 rows=1770 width=8) (actual time=0.003..0.004 rows=
19	-> Hash (cost=22.86..22.86 rows=2 width=8) (actual time=0.047..0.047 rows=6 loops=1)
20	-> Nested Loop (cost=4.29..22.86 rows=2 width=8) (actual time=0.032..0.044 rows=6 loops=1)
21	-> Nested Loop (cost=4.29..21.08 rows=5 width=8) (actual time=0.025..0.027 rows=1 loops=1)
22	-> Index Scan using ak_nestadoactoprocesal on nestadoactoprocesal nea (cost=0.00..8.27 rows=1 width=8
23	Index Cond: (estado = 'Pendiente a dar presentado'::text)
24	-> Bitmap Heap Scan on ntramite nt (cost=4.29..12.75 rows=5 width=16) (actual time=0.009..0.010 rows=1
25	Recheck Cond: (nt.idnestadodestino = nea.idnestadoactoprocesal)
26	-> Bitmap Index Scan on fk_nestadoactoprocesalntram55 (cost=0.00..4.29 rows=5 width=0) (actual time
27	Index Cond: (nt.idnestadodestino = nea.idnestadoactoprocesal)
28	-> Index Scan using fk_ntramitedtramite on dtramite t (cost=0.00..0.33 rows=2 width=16) (actual time=0.005..(
29	Index Cond: (t.idntramite = nt.idntramite)
30	-> Index Scan using pk_dexpedientepreparatorio on dexpedientepreparatorio ex (cost=0.00..0.29 rows=1 width=16) (actu
31	Index Cond: (ex.idexpedientepreparatorio = exu.idexpedientepreparatorio)
32	-> Index Scan using fk_dexpedientepreparatoriodpie on dpiezaconviccion p (cost=0.00..0.37 rows=3 width=8) (actual time=0.(
33	Index Cond: (ex.idexpedientepreparatorio = p.idexpedientepreparatorio)
34	-> Index Scan using fk_descritodescritoacusado on descritoacusado ea (cost=0.00..0.43 rows=9 width=16) (actual time=0.003..0.004 rows=:
35	Index Cond: (ea.idescrito = eex.idescrito)
36	-> Hash (cost=9.44..9.44 rows=11 width=40) (actual time=0.063..0.063 rows=11 loops=1)
37	-> Hash Left Join (cost=8.28..9.44 rows=11 width=40) (actual time=0.040..0.053 rows=11 loops=1)
38	Hash Cond: (a.idnsituacionlegal = n.idnsituacionlegal)
39	-> Seq Scan on dacusado a (cost=0.00..1.11 rows=11 width=16) (actual time=0.008..0.012 rows=11 loops=1)
40	-> Hash (cost=8.27..8.27 rows=1 width=40) (actual time=0.021..0.021 rows=0 loops=1)
41	-> Index Scan using ak_nsituacionlegal on nsituacionlegal n (cost=0.00..8.27 rows=1 width=40) (actual time=0.020..0.020 rows=0 loo
42	Index Cond: (situacionlegal = 'Prisión Provisional'::text)
43	-> Index Scan using pk_descrito on descrito e (cost=0.00..0.27 rows=1 width=16) (actual time=0.003..0.003 rows=1 loops=3)
44	Index Cond: (e.idescrito = eex.idescrito)
45	Total runtime: 0.559 ms

Figura 18 Resultado de consulta 2 con índice

Para interpretar los resultados hay una serie de aspectos a tener en cuenta. En la primera parte de la línea superior se observa el costo de inicio estimado, este es el tiempo inicial que toma devolverse la primera tupla. Luego el costo total estimado que es el tiempo total para devolver todas las tuplas. A continuación se puede apreciar el número estimado de filas escaneadas (se cumple solamente si la ejecución de la consulta es completa). Por último se tiene la cantidad estimada de filas de salida. En la segunda parte se encuentra el tiempo real en que se devuelven las tuplas así como las iteraciones que realiza. Las líneas continuas brindan la misma información pero detalladas para cada ejecución.

Para medir la eficiencia de los resultados se establecieron los siguientes intervalos respondiendo al requisito no funcional definido por el proyecto para tiempo de respuesta que establece que debe ser menor que 3 segundos:

Bajo	0 milisegundos < x < 1 milisegundos
Medio	1 milisegundos < x < 3 milisegundos
Alto	X > 3 milisegundos

Luego de realizadas las pruebas y valorados los resultados se puede afirmar que las respuestas a las solicitudes de los usuarios son relativamente rápidas y teniendo en cuenta la comparación entre las que usan índices y las que no usan, se puede definir que las que usan ese tipo de estructura son más óptimas para tablas que manejan una gran cantidad de datos.

3.4 Pruebas de estrés y carga

Las pruebas de estrés y carga consisten en someter a una aplicación o base de datos a un régimen de carga de trabajo (habitualmente por simulación de concurrencia) similar al esperado en la explotación real del sistema. El objetivo de estas pruebas es buscar consultas mal diseñadas, consultas candidatas a optimización, la necesidad de índices adicionales, código mal diseñado, tiempo de demora de respuesta de magnitudes inaceptables, hardware insuficiente, problemas de control de concurrencia, etc.

Para la realización de las Pruebas de Carga existen diversos mecanismos y herramientas que automatizan dicho proceso. Se pueden utilizar desde navegadores ordinarios, trazas del servidor de base de datos, una aplicación simplificada con consultas de la aplicación real con un mínimo de código y sin complejidad algorítmica ni iteraciones, la utilización de herramientas diseñadas con este fin, entre otras. Para realizar las pruebas se utilizarán las bondades que brinda le herramienta Jmeter por las facilidad de su uso y las funcionalidades que brinda.

La herramienta posee dos tipos de generación de carga, indirecta, es decir, a través de una aplicación y directa que basa fundamentalmente su utilización en consultas grabadas en la traza o log del servidor de base de datos. La que se va a utilizar para las pruebas del sistema es la directa configurada específicamente para la realización de consultas sobre el servidor de base de datos.

Se realizaron dos pruebas a la base de datos, variando el número de **usuarios a simular** o **hilos** como es denominado en la herramienta Jmeter. Además se especifica el **período de subida (en segundos)** que no es más que el tiempo que debiera llevarle a la herramienta lanzar todos los hilos (si se seleccionan 10 hilos y el período de subida es de 1 segundo, entonces cada hilo comenzará 0,1 segundo después de que el hilo anterior haya sido lanzado) y el **contador del bucle** el cuál representará el número de veces a realizar la prueba.

Propiedades de los hilos	Prueba 1	Prueba 2
Número de hilos	300	500
Período de subida(en segundos)	1	1
Contador del bucle	5	10

Los resultados arrojados para estas dos pruebas son:

Label	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
Petición JDBC	1500	1242	1109	3078	0	7593	0,00%	160,5/sec	29,2
TOTAL	1500	1242	1109	3078	0	7593	0,00%	160,5/sec	29,2

Figura 19 Resultados de la primera prueba con el Jmeter

Label	# Muestras	Media	Mediana	Linea de 90%	Mín	Máx	% Error	Rendimiento	Kb/sec
Petición JDBC	5000	2189	234	7297	0	11031	5,22%	160,5/sec	27,6
TOTAL	5000	2189	234	7297	0	11031	5,22%	160,5/sec	27,6

Figura 20 Resultados de la segunda prueba con el Jmeter

En los informes de cada prueba se muestran los siguientes indicadores:

- ✓ **Label**: El nombre de la muestra (conjunto de muestras).

- ✓ **# Muestras**: El número de muestras de peticiones JDBC[12].
- ✓ **Media**: El tiempo medio transcurrido para un conjunto de resultados.
- ✓ **Mediana**: Mediana aritmética.
- ✓ **DRAE**: Función Matemática. Elemento de una serie ordenada de valores crecientes de forma que la divide en dos partes iguales, superiores e inferiores a él.
- ✓ **Mín**: El mínimo tiempo transcurrido para las muestras de peticiones JDBC.
- ✓ **Máx**: El máximo tiempo transcurrido para las muestras de peticiones JDBC.
- ✓ **Error %**: Porcentaje de las peticiones con errores.
- ✓ **Rendimiento**: Rendimiento medido en base a peticiones por segundo/minuto/hora.

Se puede concluir entonces para el resultado de las pruebas: para la primera prueba con 300 usuarios a simular y con 5 iteraciones por cada uno realizando una consulta a la base de datos, en total 1500 de peticiones, esta demorará 1.2 segundos de tiempo de respuesta (dado por la media) en atender la solicitud de los usuarios. Para la segunda prueba se toman 500 usuarios pero con 10 iteraciones por cada uno, en total 5000 peticiones, retornando como tiempo de respuesta de la base de datos a los usuarios 2.1 segundos en atender su solicitud. Se concluye entonces apoyándose en el requisito no funcional definido por el proyecto para tiempo de respuesta que establece que debe ser menor que 3 segundos, la base de datos está apta para soportar gran cantidad de servicios solicitados simultáneamente, logrando así la eficiencia requerida para el buen funcionamiento de la misma.

3.5 *Conclusiones*

Al concluir este capítulo se puede plantear que se han cumplido los objetivos propuestos para el mismo, superando al mismo tiempo las expectativas esperadas. La validación del diseño culminó con la valoración de aspectos importantes como la integridad de la información almacenada, definiéndose varias restricciones que permitieran garantizar la validez de la misma. Se demostró a través de las pruebas realizadas que se cuenta con una BD robusta y segura para brindar los servicios requeridos.

[12] Java Database Connectivity(Conectividad a la base de datos a través del código Java)

Conclusiones Generales

Hoy en día el mundo se mueve vertiginosamente: el paso de una tecnología a otra y la rapidez requerida a la hora de dar respuesta a determinados problemas puede que se piense, en ocasiones, que no queda tiempo suficiente para ser formales durante el desarrollo de los proyectos. Por esto, seguramente en varias ocasiones se ha tenido que regresar al punto de partida, al no seguir una metodología que ayudara a definir cada detalle a tener en cuenta durante el largo camino a recorrer.

Se realizó un estudio completo a los sistemas informáticos existentes que utilizan base de datos jurídicas tomando sus principales deficiencias para no cometerlas en el posterior desarrollo de la investigación.

Se definieron todas las entidades que van a persistir conjuntamente con las relaciones entre estas, siguiendo metodologías de diseño, cumpliendo con estándares definidos por la directiva de Arquitectura del proyecto y la experiencia de los diseñadores.

Se obtuvo el modelo entidad relación que se implementó en el gestor PostgreSQL, generando de esta forma físicamente la Base de Datos para el subsistema Penal del sistema para los Tribunales Populares Cubanos. Se pudo comprobar mediante las validaciones que el Gestor de Base de Datos utilizado, en este caso PostgreSQL 8.4 soporta los volúmenes de datos previsto desde un principio en el diseño de la base de datos.

A través de las pruebas realizadas a la misma y al diseño se garantiza la centralización de la información procesada en los Tribunales Populares Cubanos contribuyendo a la gestión eficiente de los procesos, además se evidenció que cumple con los requisitos de los clientes y que la misma está apta para su funcionamiento, de acuerdo a su grado de integridad y seguridad.

Recomendaciones

Las metas planteadas con este trabajo se han cumplido y en base a los resultados obtenidos se recomienda:

- ✓ Extender el modelo utilizado al resto de los proyectos de desarrollo jurídico.
- ✓ Utilizar la propuesta de diseño de la base de datos en las próximas fases de desarrollo del proyecto.
- ✓ Realizar un profundo estudio de las técnicas de optimización, ya sean las descritas u otras, aplicadas sobre las base de datos.
- ✓ Continuar con el desarrollo de la base de datos acorde al resto de las fases propuestas por el Modelo de Desarrollo de Bases de Datos.
- ✓ Proporcionar mantenimiento y soporte regular a la base de datos enfocados a su mejor funcionamiento.
- ✓ Realizar un análisis profundo de los temas de clúster y réplica para definir su utilización en el desarrollo del proyecto.

Bibliografía

Aportela Rodrígues, Lic. Ivett M. Intranets. 2007. Las tecnologías de información y comunicación en función de la organización. [En línea] 2007. http://bvs.sld.cu/revistas/aci/vol16_4_07/aci041007.html..

Booch, Grady, Jacobson, Ivar and Rumbaugh, James. 2000.*El Proceso Unificado de Desarrollo de Software.* 2000.

Change Manager.Embarcadero Technologies,Inc. [En línea] .http://www.embarcadero.com/products/change-manager..

Copyright © 1996 – 2011 PostgreSQL Global Development Group. PostgresSQL. [En línea] [Citado el: 2 de febrero de 2011.] http://www.postgresql.org/.

Date, CJ. 2003.*Sistemas de Bases de Datos.* La Habana : Félix Varela, 2003.

Desarrollo Web. [En línea] [Citado el: 3 de febrero de 2011.] http://www.desarrolloweb.com/

Diez, María Luisa Alvite. 2007. *Anales de Documentación.Evolución de las bases de datos jurídicas en España.* España : s.n., 2007.

Doctrine. [En línea] [Citado el: 13 de marzo de 2011.] http://www.doctrine-project.org..

England, K. and G. Powell. (2007). *Performance Optimization and Tuning Handbook. .* Estados Unidos : s.n., (2007).

Freedownloadmanager. [En línea] [Citado el: 3 de febrero de 2011.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_

García, Rosa María Mato. 2005. *Sistemas de bases de datos.* Ciudad de la Habana : Pueblo y Educación, 2005.

Guerra, Elsydania López. 2009. *Desarrollo de los requerimientos de software del módulo Compras y Servicios del subsistema Administración Financiera del Proyecto Registros y Notarías.* Ciudad de la Habana : s.n., 2009.

Jurisoft. 1995. Jurisoft. [En línea] 1995. www.jurisoft.es.

Moraga, M. Ángeles. 2001.*Modelo de Datos Jerárquico. .* La Mancha : Universidad de Castilla, 2001 : s.n., 2001.

Ochoa, Darian González. 2009.*Diseño e Implementación de un Almacén de Datos Operacionales para la Corporación CIMEX.* Ciudad de la Habana : s.n., 2009.

Oracle. [En línea] [Citado el: 12 de diciembre de 2010.] http://www.oracle.com/..

Pírez, Renén Quiroz. 1999. *Manual de derecho penal 1.* La habana : Félix Varela, 1999.

PostgresSql. [En línea] [Citado el: 2 de febrero de 2011.] http://postgresql-for-windows.softonic.com/.

Rafael Camps Paré, Luis Alberto Casillas Santillán, Dolors Costal Costa, Marc Gibert Ginesta. 2005 .*Bases de Datos.* Barcelona : s.n., 2005 .

Rodríguez, Alain Osorio. 14 de junio 2008.*2008. Proceso general de desarrollo – administración de bases de datos: propuesta de modelo de desarrollo de bases de datos para procesos de desarrollo de software.* Caracas : s.n., 14 de junio 2008.

Silva, Luis Raciel Rodríguez. 2009.*Los sistemas de gestión de servicios legales. Propuesta de modelación de un sistema para una oficina de asistencia legal en Cuba. .* Ciudad de la Habana : s.n., 2009.

Sitio del Gobierno de la República de Cuba. [En línea] [Citado el: 4 de enero de 2011.] http://www.cubagob.cu/des_soc/derecho.htm..

Solano, Orlando. 2003. Slideshare. [En línea] 2003. http://www.slideshare.net/guest9ca8c4/informatica-juridica-de-gestion-97-2003..

Sparxsystems. [En línea] [Citado el: 5 de enero de 2011.] http://sparxsystems.com.es/products/ea.html

Valdes, Julio Téllez. 1987. *Derecho Informático.* 1987.

Velazco, Roberto Hernando. rhernando. [En línea] http://www.rhernando.net/.

Visual Paradigm. *Visual Paradigm for UML.* [En línea] [Citado el: 23 de enero de 2011.] http://www.visual-paradigm.com/product/vpuml/..

Yinet Garbey, Alfredo Badillo. 2010. *Fase de Inicio del desarrollo de la base de datos del módulo Gobierno y Administración del proyecto Sistema para los Tribunales Cubanos.* Ciudad de la Habana : s.n., 2010

Glosario de términos

-BD: Base de datos

-TPC: Tribunales Populares Cubanos

-Módulo: Es un componente autocontrolado de un sistema, el cual posee una interfaz bien definida hacia otros componentes; algo es modular si es construido de manera tal que se facilite su ensamblaje, acomodamiento flexible y reparación de sus componentes.

-Subsistema: Parte del sistema global con una interfaz razonablemente bien definida.

-Modelo entidad relación: Un diagrama o modelo entidad-relación (a veces denominado por su siglas, E-R "Entity relationship", o, "DER" Diagrama de Entidad Relación) es una herramienta para el modelado de datos de un sistema de información. Estos modelos expresan entidades relevantes para un sistema de información, sus inter-relaciones y propiedades.

-Llaves foráneas o foreign key: Expresan relaciones entre objetos representados, incluyendo en el esquema de una relación atributos de otra. Utilizado para relacionar tablas.

-Llave primaria o primary key: Conjunto de atributos de su esquema que son elegidos para servir de identificador unívoco de sus tuplas.

-Tupla: Es la representación de una fila en una de las tablas que se está almacenando datos. Y las cuales serán llamadas por los administradores de base de datos en el tiempo de ejecución de un sistema.

-DBAL: Acrónimo de Database Abstraction Layer (Capa de Abstracción de la Base de Datos). Una capa de abstracción de bases de datos es una interfaz de programación de aplicaciones que unifica la comunicación entre una aplicación informática y bases de datos tales como MySQL, PostgreSQL, Oracle o SQLite. La capa de abstracción de bases de datos permite reducir la cantidad de trabajo proporcionando una API consistente para el desarrollador y ocultado las especificaciones de la base de datos, detrás de la interfaz que provee, tanto como sea posible.

-Entidad: Se refiere a cualquier concepto del mundo real con una existencia independiente.