Universidad de las Ciencias Informáticas

Facultad 3

Centro de Gobierno Electrónico

Diseño e implementación de base de datos de los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de la Gestión de las Fiscalías II

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

Autor: José Miguel Fabra Gallo

Tutores: Ing. Manuel Alvarez Alonso

Ing. José Carlos Pupo Acosta

Ing. Yenier Figueroa Machado

La Habana, junio de 2014

"A las estrellas no se sube por caminos llanos"

José Martí

Declaración de autoría

Por este medio declaro ser el único autor del trabajo *"Diseño e implementación de la base de datos de los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de la Gestión de las Fiscalías II".* Y en pleno uso de mis facultades físicas y mentales autorizo a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo.

Para que así conste firmo la presente a los _____ días del mes de junio de 2014.

José Miguel Fabra Gallo

_____
Firma del Autor

Ing. José Carlos Pupo Acosta

_____
Firma del Tutor

Ing. Manuel Alvarez Alonso

_____
Firma del Tutor

Datos de contacto

Datos de tutores

Nombre: Ing. Manuel Alvarez Alonso.

Ingeniero en Ciencias Informáticas.

Correo electrónico: malvareza@uci.cu

Nombre: Ing. José Carlos Pupo Acosta.

Ingeniero en Ciencias Informáticas.

Correo electrónico: jcpupo@uci.cu

Nombre: Ing. Yenier Figueroa Machado.

Ingeniero en Ciencias Informáticas.

Correo electrónico: yfigueroa@uci.cu

Datos del Autor:

Nombre: José Miguel Fabra Gallo.

Correo electrónico: jmfabra@estudiantes.uci.cu

Agradecimientos

A mis padres por hacer de esta su causa y razón de ser, por la entrega y los no pocos sacrificios.

A mis familiares en especial a mis tíos "los gordos" y a mis hermanos. Gracias por esta ahí, gracias por hacerme sentir especial y necesario.

A Jenny y su familia por el cariño desinteresado y la preocupación constante.

A Yalice y Yoan por acogerme como un hijo, por tantos consejos y atenciones inmerecidas.

A mis tutores y en especial a Manuel, hombre en palabras mayúsculas y amigo de todas, todas.

A mis amigos, los de allá, los de la tierra más caliente y caribeña de Cuba: David, Maura, Lianne, Eliza, Orlando, Yanet, Pedro, por tanto apoyo, consejos y buenos momentos. Y a los de aquí, con los cuales he compartido 5 de los años más productivos de mi vida y en los cuales ustedes fueron mis dedos gordos del pie…mi sostén.

A mi equipo de asalto y combate -como suelo llamarlos-, Irene, Lizi y Willian por tanta paciencia con este pedazo de ser humano o con este ser humano en pedazos.

A esa gran familia de gente creativa y loca: el t3am de dragon3s.

A mi comité primario y la nombro mío porque gracias a su gente, así lo sentí. Gracias a ustedes disfrute cada noche de reunión, cada preparación y corre corre para los eventos, cada opinión acerca de mis diseños.

A los demás marineros de esta travesía -los de mis tres brigadas anteriores- que me acompañaron por 5 largos años y de los cuales aprendí que soy un tipo paciente y tolerante y que sin ustedes la carrera hubiese sido demasiado aburrida.

A todos los que de forma directa o indirecta han contribuido a mi formación como profesional y como persona.

Dedicatorias

A mis padre, mi principal motivación y ejemplo.

A los que la lejanía, el tiempo y la muerte no le impiden guiar mis pasos día a día: abuela Esperanza, Raciel, tío Eriberto y tía María.

Resumen

Hoy en día la información se ha convertido en uno de los principales recursos en el entorno empresarial, impactando directamente en la planificación, la vigilancia estratégica y la toma de decisiones. El desarrollo alcanzado por las Tecnologías de la Información y las Comunicaciones, así como la presencia de volúmenes de información cada vez más altos y complejos ha demandado una manipulación más efectiva y segura de este recurso, siendo el surgimiento de las bases de datos una respuesta viable a tal demanda.

El presente trabajo tiene el propósito de realizar el diseño e implementación de una base de datos para los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de la Gestión de las Fiscalías. Esta proveerá al sistema de una estructura que permita almacenar los datos asociados a dichos procesos, a la vez que contribuye a garantizar la seguridad e integridad de la información.

Con este propósito se realizó un estudio de los fundamentos teóricos que sustentan la investigación. A partir de este análisis se definió el modelo de datos a utilizar, las herramientas para las labores de diseño, implementación y pruebas, así como el marco de trabajo de desarrollo de base de datos. Se tuvieron en cuenta además los patrones de diseño para lograr un modelo de datos robusto, las estrategias de indexado para aumentar el rendimiento y las restricciones de integridad para garantizar la consistencia de los datos almacenados.

Palabras Claves

Base de datos, modelo de datos, seguridad, integridad.

Índice de contenido

Introducción 1

Capítulo 1. Fundamentación teórica 5

1.1. Introducción 5

1.2. Base de datos y Sistemas Gestores de Bases de Datos 5

1.2.1. Base de datos (BD) 5

1.2.2. Sistemas Gestores de Base de Datos (SGBD) 6

1.3. Modelo de datos 7

1.4. Diseño de base de datos 11

1.4.1. Patrones de diseño de bases de datos 11

1.5. Marco de trabajo (framework) de desarrollo de base de datos 14

1.6. Estrategias de indexado 15

1.7. Normalización de base de datos 16

1.8. Integridad en las base de datos 19

1.9. Seguridad en las bases de datos. 19

1.10. Herramientas y tecnologías para el desarrollo y pruebas de la base de datos 21

1.11. Conclusiones parciales 23

Capítulo 2. Análisis de la solución propuesta 24

2.1. Introducción 24

2.2. Descripción de la arquitectura de datos y políticas de respaldo 24

2.3. Descripción del entorno de desarrollo 25

2.4. Nomenclatura y normas para el modelo de base de datos 26

2.5. Modelo de datos 28

2.5.1. Modelo lógico 28
2.5.2. Modelo físico 30
2.5.2.1. Patrones utilizados en el diseño físico de la base de datos 39
2.6. Acceso a datos 40
2.6.1. Código fuente 41
2.7. Reportes 42
2.8. Normalización del modelo 43
2.9. Estrategia de indexado utilizada 43
2.10. Conclusiones parciales 44
Capítulo 3. Validación y pruebas. 45
3.1. Introducción 45
3.2. Validación teórica 45
3.2.1. Métricas 45
3.3. Validación funcional mediante pruebas de rendimiento 46
3.3.1. Pruebas de volumen 46
3.3.2. Pruebas de carga y estrés 48
3.4. Integridad y consistencia de los datos 52
3.5. Seguridad de la base de datos 53
3.6. Conclusiones parciales 54
Conclusiones Generales 55
Recomendaciones 56
Bibliografía 57
Anexos 61

Índice de figuras

Figura 1. Representación del Modelo Jerárquico de base de datos. 8

Figura 2. Representación del Modelo de Red de base de datos. 9

Figura 3. Conceptos del modelo entidad-relación. 10

Figura 4. Patrón árbol simple. 12

Figura 5. Patrón árbol fuertemente codificado. 12

Figura 6. Patrón árbol estructurado. 13

Figura 7. Patrón grafo dirigido simple. 13

Figura 8. Patrón grafo dirigido estructurado. 14

Figura 9. Fases genéricas dentro del proceso de desarrollo de software. 15

Figura 10. Formas Normales Anidadas. 17

Figura 11. Distribución de datos e información 24

Figura 12. Descripción del entorno de desarrollo. 26

Figura 13. Ejemplos de nomenclatura para entidades. 27

Figura 14. Ejemplos de nomenclatura para entidades. 27

Figura 15. Modelo lógico perteneciente al módulo Administrativo. 29

Figura 16. Modelo lógico perteneciente al módulo Revisión de Causas Penales. 29

Figura 17. Modelo físico de escenario Expediente Prejudicial. 30

Figura 18. Modelo físico de escenario Rollo Fiscal. 31

Figura 19. Modelo físico de escenario Rollo Casación. 33

Figura 20. Modelo físico de escenario Rollo Revisión Administrativa. 34

Figura 21. Modelo físico de escenario Solicitud. 35

Figura 22. Modelo físico de escenario Rollo Penal. 37

Figura 23. Modelo físico de escenario Cédula de Emplazamiento. 38

Figura 24. Ejemplos de patrón llaves subrogadas. 39

Figura 26. Ejemplo de mapeo objeto relacional. 40

Figura 27. Ejemplo de funciones en DQL. 41

Figura 28. Vista parcial de función implementada para la generación de reportes. 43

Figura 29. Vista de algunos de los campos indexados. 44

Figura 30.Pruebas de volumen sobre entidad drolloexpedienteprejudicial. 47

Figura 31. Pruebas de volumen sobre entidad drollopenal. 48

Figura 32. Consulta de mediana complejidad utilizada para pruebas en JMeter. 49

Figura 33. Configuración del plan de pruebas. 49

Figura 34. Reporte resumen correspondiente al grupo de hilos 1. 50

Figura 35. Reporte resumen correspondiente al grupo de hilos 2. 50

Figura 36. Gráfico de resultados correspondiente al grupo de hilos 1. 51

Figura 37. Gráfico de resultados correspondiente al grupo de hilos 2. 51

Índice de tablas

Tabla 1. Entidad representativa del escenario Expediente Prejudicial. 31

Tabla 2. Entidad representativa del escenario Rollo Fiscal. 32

Tabla 3. Entidad representativa del escenario Rollo de Casación. 33

Tabla 4. Entidad representativa del escenario Rollo de Casación. 34

Tabla 5. Entidad representativa del escenario Rollo Revisión Administrativa. 35

Tabla 6. Entidad representativa del escenario Solicitud. 36

Tabla 7. Entidad representativa del escenario Rollo Penal. 38

Tabla 8. Entidad representativa del escenario Cédula de Emplazamiento. 39

Tabla 9. Comparación entre ambiente de prueba y condiciones reales de las fiscalías. 47

Tabla 10. Tiempo de respuesta a peticiones (Acosta, 2010). 52

Tabla 11. Descripción de las restricciones de dominio. 53

Introducción

Desde el surgimiento de la humanidad, la necesidad de almacenar información ha sido una constante, con motivos y funciones distintas pero con el objetivo de preservar el conocimiento para las futuras generaciones. Hoy en día, debido al gran caudal de información que se genera desde y hacia la sociedad, la necesidad se va más allá de las fronteras del mero almacenamiento, demandando además un manejo eficiente y acceso oportuno, siendo el desarrollo alcanzado por las Tecnologías de la Información y las Comunicaciones (TIC) el elemento catalizador de dichos procesos.

Cuba no ha estado ajena de este fenómeno, sino que por el contrario encara desde hace ya varios años un proceso de Informatización de la Sociedad Cubana, atendiendo al impacto positivo que las TIC podrían tener en áreas estratégicas de la sociedad, y donde el almacenamiento, recuperación y procesamiento de la información son elementos de vital importancia. En este marco y siguiendo las ideas de Fidel Castro de *"…convertir la informática en una de las ramas más productivas y aportadoras de recursos para la nación",* y emplear *"…a fondo la inteligencia y el capital humano que tenemos",* (UCI, 2012) surge en el 2002 la Universidad de las Ciencias Informáticas (UCI), universidad de nuevo tipo, con la responsabilidad de formar profesionales altamente calificados y de producir aplicaciones y servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación.

La infraestructura productiva de la UCI se soporta en una red de centros de desarrollo entre los cuales se encuentra el Centro de Gobierno Electrónico (CEGEL) perteneciente a la Facultad 3. Este está dirigido al desarrollo de productos, servicios y soluciones informáticas integrales para el gobierno. Una de estas soluciones es el resultado de un convenio de colaboración entre la UCI y la Fiscalía General de la República (FGR) denominado Sistema de Informatización de la Gestión de las Fiscalías II (SIGEF II), con el objetivo de mejorar los servicios que brinda la Fiscalía en sus diferentes instancias, así como incrementar los niveles de gestión de la información a partir de su adecuada organización y preservación.

Estructuralmente SIGEF II está formado por subsistemas o departamentos que incluyen módulos que se corresponden con los procesos llevados a cabo en las fiscalías cubanas, siendo Protección a los Derechos Ciudadanos (PDC) uno de estos departamentos. Dos de los procesos que se tramitan en el subsistema son Administrativo y Revisión de Causas Penales. El primero tiene lugar cuando se ven comprometidos los intereses del estado o de personas que carezcan de representación legal, fundamentalmente en temas de vivienda. El segundo es el encargado de

atender las revisiones de causas penales que en el orden legal formulen los ciudadanos sobre presuntas violaciones a sus derechos, teniendo como objetivo fundamental el control y la preservación de la legalidad sobre la base de la vigilancia del estricto cumplimiento de la Constitución, las leyes y demás disposiciones legales.

En la actualidad ambos procesos se tornan largos, engorrosos y difíciles de ejecutar, en gran medida por el cúmulo de información que se debe manejar y la carencia de recursos humanos para ello, provocando que se presenten afectaciones como:

- Recepción de forma manuscrita de la información, generando gran cantidad de errores de escritura, tachaduras y repeticiones innecesarias.
- Tendencia a la acumulación de archivos, por la poca capacidad de procesamiento de los documentos que se generan.
- Los reportes emitidos sobre el estado de los mismos no ofrecen la información necesaria sobre los datos que se están manejando.
- El deterioro que sufren estos documentos al cabo del tiempo, provoca que se vuelvan borrosos y de difícil manejo.

Los factores antes mencionados hacen real y constatable la necesidad de informatizar los procesos Administrativo y Revisión de Causas Penales por parte de la Fiscalía General de la República. Para ello fueron incorporados al sistema módulos de igual nombre, los cuales pretenden gestionar de forma organizada la información, facilitando su acceso y manipulación. Lo cual se traduce en agilidad y efectividad en el trabajo que realizan los fiscales a lo largo del territorio nacional. Atentando contra esta realidad, la carencia de una estructura que permita almacenar los datos asociados a dichos procesos y a la vez contribuya a la seguridad e integridad de la información.

Por estas razones se identifica como **problema a resolver:** ¿Cómo garantizar el almacenamiento y manipulación de la información gestionada por los módulos Administrativo y Revisión de Causas Penales de forma que se contribuya a la seguridad de la información y la integridad de los datos?

Se define como **objeto de estudio**: Sistemas de bases de datos.

Para darle solución al problema antes identificado se define el siguiente **objetivo general:**

Desarrollar una base de datos para los módulos Administrativo y Revisión de Causas Penales que garantice el almacenamiento y manipulación de la información gestionada, contribuyendo a la seguridad e integridad de los datos.

El cual se enmarca en el **campo de acción**: Sistemas de bases de datos relacionales.

Se define como **idea a defender**: Con el desarrollo de una base de datos para los módulos Administrativo y Revisión de Causas Penales, se garantizará al almacenamiento y manipulación de los datos que generan los procesos Administrativo y Revisión de Causas Penales contribuyendo a la seguridad e integridad de la información.

Se plantean como **objetivos específicos**:

- Elaborar el marco teórico de la investigación.
    - Tarea 1: Caracterización de las herramientas de modelado.
    - Tarea 2: Caracterización del Sistema Gestor de Base de Datos.
    - Tarea 3: Caracterización del marco de trabajo.
    - Tarea 4: Definición de las estrategias de validación del diseño.
- Diseñar la propuesta de base de datos para los módulos Administrativo y Revisión de Causas Penales.
    - Tarea 1: Creación del modelo lógico.
    - Tarea 2: Creación del modelo físico.
- Implementar la propuesta de base de datos para los módulos Administrativo y Revisión de Causas Penales.
    - Tarea 1: Definición de índices.
    - Tarea 2: Implementación de dominios.
    - Tarea 3: Implementación de funciones y tipos de datos personalizados para la inclusión en el Generador Dinámico de Reportes.
    - Tarea 4: Implementación de la capa de acceso a datos.
- Validar la solución propuesta.
    - Tarea 4: Aplicación de la estrategia de pruebas seleccionada.

Los **métodos de investigación** utilizados fueron los siguientes:

Métodos teóricos:

- *Histórico-lógico:* Utilizado para analizar la trayectoria y evolución de los sistemas de bases de datos.
- *Analítico-sintético:* Utilizado para el procesamiento de toda la información relacionada con el tema de investigación, analizando los documentos que permitieron extraer los elementos más significativos relacionados con el objeto de estudio.

- *Modelación:* Se utilizó para la creación de abstracciones acerca de la realidad, dígase diagramas y artefactos en el proceso de modelado y diseño de la base de datos.

Métodos empíricos:

- *Experimental:* Posibilitó la creación de las condiciones para realizar las pruebas que validan la propuesta de solución.
- *Entrevista:* Mediante este método se realizaron consultas a especialistas en las diferentes áreas del negocio, en aras de un mayor entendimiento del mismo.

El siguiente trabajo de diploma está estructurado en tres capítulos, donde se abordan los siguientes aspectos:

Capítulo 1: Contiene las bases teóricas que fundamentan la presente investigación. Refleja las tendencias del uso de bases de datos, abordando las características y el funcionamiento de las mismas. Además se analizan y describen las herramientas utilizadas para el diseño e implementación de la propuesta de base de datos.

Capítulo 2: Describe la propuesta de solución, las características de la base de datos de los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de la Gestión de las Fiscalías II. Se describe el estándar de nomenclatura utilizado, se muestran las características generales del diseño, se ejemplifican los patrones utilizados y se presentan los modelos lógico y físico.

Capítulo 3: Aborda la validación del diseño e implementación de la base de datos desarrollada, a través de métricas y pruebas de rendimiento. Además se describen las acciones encaminadas a contribuir con la seguridad e integridad de los datos almacenados.

Capítulo 1. Fundamentación teórica

1.1. Introducción

En el presente capítulo se aborda el basamento teórico que sirve de soporte al desarrollo de la base de datos, partiendo de su conceptualización y destacando sus características fundamentales. Se hace referencia a los Sistemas Gestores de Base de Datos, modelos de datos, las características del diseño de base de datos, sus fases y patrones, y se incluyen las herramientas que se utilizan para realizar estas tareas y sus especificaciones. Por último se tratan los pasos generales para la normalización, las características de la integridad y seguridad de los datos.

1.2. Base de datos y Sistemas Gestores de Bases de Datos

1.2.1. Base de datos (BD)

La génesis de las bases de datos se remonta a la antigüedad donde ya existían bibliotecas y toda clase de registros acerca de las cosechas y medios de trabajo. Sin embargo la búsqueda de información se tornaba lenta y poco eficaz, y no se contaba con el auxilio de máquinas que pudiesen remplazar el trabajo manual. Posteriormente, y aparejado el desarrollo experimentado por la sociedad, se hizo evidente la necesidad de almacenar grandes volúmenes de datos, siendo la aparición de las computadoras el elemento que vincularía por completo el concepto de base de datos a la informática.

En la actual sociedad de la información, las bases de datos se han convertido en una de las herramientas más ampliamente difundidas y utilizadas como fuentes secundarias, en cuanto a recuperación y almacenamiento de información en todos los campos a nivel científico, social, económico, político y cultural. Por lo que el análisis de los conceptos asociados a las mismas alcanza hoy en día una importancia sin precedentes.

Son muchos los especialistas que han realizado aproximaciones conceptuales al término. De ellas se asume la realizada por la MsC. Rosa María Matos en su libro “Sistemas de Base de Datos” del 2005, al definir una base de datos como: *“Conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, que una base de datos puede considerarse una colección de datos variables en el tiempo”* (Mato, 2005). Tomando como criterio de selección el nivel de actualidad y completitud de dicha definición.

Las bases de datos pueden ser creadas desde cero, pero normalmente se emplean sistemas especializados para sus labores de creación, administración y mantenimiento, conocidos como Sistemas Gestores de Base de Datos.

1.2.2. Sistemas Gestores de Base de Datos (SGBD)

Un Sistema Gestor de Base de Datos es un sistema de software que permite la definición de bases de datos; así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos. El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular en términos abstractos los datos, o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado (Mato, 2005).

Para las labores de definir, crear, mantener y proporcionar un acceso controlado a las bases de datos, es necesario que dichos sistemas presenten los siguientes servicios (Zambrano, 2008):

- Definición y creación de bases de datos.
- Manipulación de los datos utilizando consultas, inserciones y actualizaciones.
- Acceso controlado a los datos mediante mecanismos de seguridad de acceso a los usuarios.
- Mantener la integridad de los datos.
- Poseer mecanismos de copias de respaldo y recuperación para restablecer la información en caso de fallos en el sistema.

A nivel global existe un gran número de SGBD entre los cuales se pueden mencionar DB2 de IBM, Oracle, Microsoft SQL Server, Informix, MySQL y PostgreSQL, siendo este último el definido en el proyecto para el trabajo con las bases de datos, atendiendo a las siguientes características (Comunidad de PostgreSQL-UCI, 2014):

- Distribuido bajo licencia BSD *(Berkeley Software Distribution*).
- Utiliza un modelo cliente-servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.
- Se ejecuta en casi todos los principales sistemas operativos: Linux, Unix, Mac OS, Windows, etc.
- Presenta documentación muy bien organizada y pública.
- Soporta las características de una base de datos profesional (disparadores, funciones, secuencias, relaciones, tipos de datos definidos por usuarios, vistas, etc.).

1.3. Modelo de datos

Una de las características fundamentales de los sistemas de bases de datos es que proporcionan cierto nivel de abstracción de datos. Esta abstracción se logra al ocultar las características del almacenamiento físico que la mayoría de los usuarios no necesita conocer. En este sentido los modelos de datos son el instrumento principal para ofrecer la referida abstracción mediante su jerarquía de niveles.

Para (Batini, 2004) la definición de un modelo de datos consiste en una abstracción del universo de discurso, y se considera como un enfoque utilizado para la representación de las entidades y sus características presentes en una base de datos. Posición que se limita a relacionar las entidades y sus características.

Desde otra perspectiva (Marqués, 2011) afirma que un modelo de datos es un conjunto de conceptos que sirven para describir la estructura de una BD: los datos, las relaciones entre los datos y las restricciones que deben cumplirse sobre los mismos. Es precisamente esta perspectiva con la que el autor se identifica.

Los modelos de datos según (Marqués, 2011), posición que el autor comparte, pueden ser clasificados en dependencia de los tipos de conceptos que ofrecen para describir la estructura de la base de datos, formando para ello una jerarquía de niveles:

- **Modelo conceptual:** En este nivel el modelo de datos es genérico e independiente de cualquier restricción de software o hardware.
- **Modelo lógico:** En el nivel lógico el modelo está estrechamente relacionado con un tipo particular de base de datos.
- **Modelo físico:** El este el modelo de datos está relacionado de forma particular con el Sistema Gestor de Bases de Datos a utilizar. Esta manera de tratarlo también tiene como seguidor a (Ponniah, 2007).

En este contexto a los efectos de la investigación se procede a seleccionar los modelos de datos a emplear a partir del análisis de su evolución en el tiempo.

Modelo de datos jerárquico

Es un modelo en el cual los datos se organizan de forma similar a un árbol donde un padre tiene muchos hijos. Este árbol está compuesto de elementos llamados nodos. El nivel más alto del árbol se denomina raíz. Cada nodo representa un registro con sus correspondientes campos. 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 (AIU, 2014).

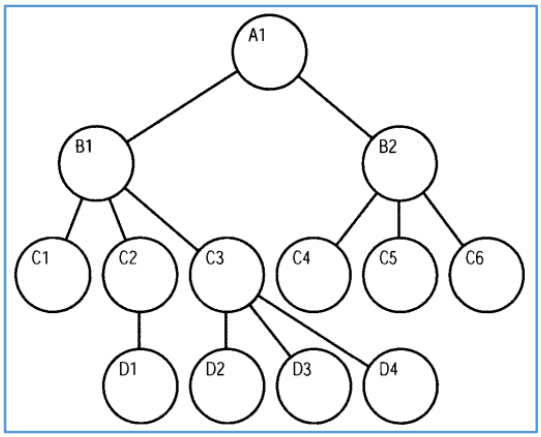


Figura 1. Representación del Modelo Jerárquico de base de datos.

Aunque en la década de los 60 supuso un gran avance para el trabajo con archivos no relacionados, presenta serios inconvenientes, que provienen en gran medida de su rigidez, entre los cuales se destacan (Hansen, y otros, 2006):

- Las características lógicas y físicas del modelo no están claramente separadas y se requieren manipulaciones para representar interrelaciones de datos no jerárquicos.
- Las relaciones muchos a muchos resultan difíciles de implementar y la agregación de nuevas relaciones puede dar lugar a grandes cambios en la estructura existente.

Modelo de datos en red o reticular

Fue desarrollado a finales de 1960, a partir del modelo jerárquico. Uno de sus principales objetivos era evitar la falta de flexibilidad de este. Las redes constituyen una manera natural de representar las interrelaciones entre los objetos, las cuales se pueden simbolizar a través de una estructura matemática llamada grafo dirigido. El modelo de datos en red muestra los datos en estructuras de retículos de los tipos de registros conectados por interrelaciones uno a uno o uno a muchos, garantizando que la recuperación y manipulación de los datos asociados se realice de forma eficiente (Cuello, 2013).

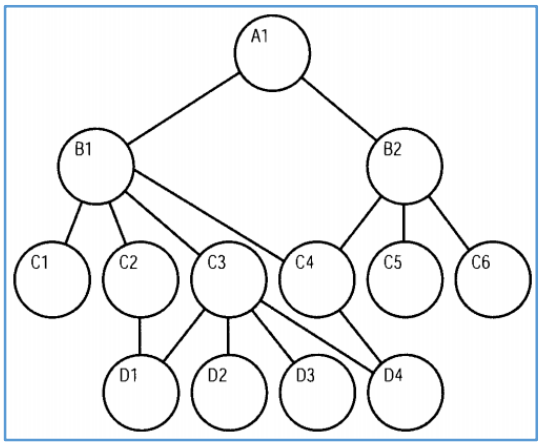


Figura 2. Representación del Modelo de Red de base de datos.

Tiende a ser efectivo cuando se realizan transacciones predeterminadas sobre la base de datos, es decir, cuando se realizan consultas repetitivas y transacciones bien definidas. Pero por el contrario presenta pocas facilidades para adaptarse a requisitos variables, resultando en ocasiones difícil de mantener.

Modelo relacional

El modelo relacional es introducido en 1970 cuando el científico informático inglés Edgar Frank Codd cambió por completo el paradigma de las bases de datos, en su trabajo "*A Relational Model of Data for Large Shared Data Banks*". En aquellos momentos, el enfoque existente para la estructura de las bases de datos utilizaba punteros físicos (direcciones de disco) para relacionar registros de distintos ficheros. Codd demostró que estas bases de datos limitaban en gran medida los tipos de operaciones que los usuarios podían realizar sobre los datos. Además, estas bases de datos eran muy vulnerables a cambios en el entorno físico. Si se añadían los controladores de un nuevo disco al sistema y los datos se movían de una localización física a otra, se requería una conversión de los ficheros de datos. Estos sistemas se basaban en el modelo de red y el modelo jerárquico, los dos modelos lógicos que constituyeron la primera generación de los SGBD (Marqués, 2011).

El modelo relacional representa la segunda generación de los SGBD. En él, 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. Pero detrás de esa simple estructura hay un fundamento teórico importante del que carecen los SGBD de la primera generación, lo que constituye otro punto a su favor.

Los primeros sistemas basados en este modelo fueron apareciendo a finales de los setenta y principios de los ochenta. Entre los que se encuentra el System R de IBM, que se desarrolló para probar la funcionalidad del modelo relacional, proporcionando una implementación de sus estructuras de datos y operaciones.

Modelo entidad-relación (MER)

Fue introducido por Peter Chen en 1976. Está formado por un conjunto de conceptos que permiten describir la realidad mediante representaciones gráficas y lingüísticas (Marqués, 2011). Originalmente, el modelo entidad-relación sólo incluía los conceptos de entidad, relación y atributo. Más tarde, se añadieron otros conceptos, como los atributos compuestos y las jerarquías de generalización, en lo que se ha denominado modelo entidad-relación extendido.

El MER tiene asociada una representación gráfica denominada Diagrama Entidad-Relación (DER). Existen diferentes representaciones y simbologías que pueden utilizarse para construir un DER, siendo la mostrada en la Figura 3 una de las más empleadas y la cual es asumida por el autor para el modelado conceptual de la base de datos.

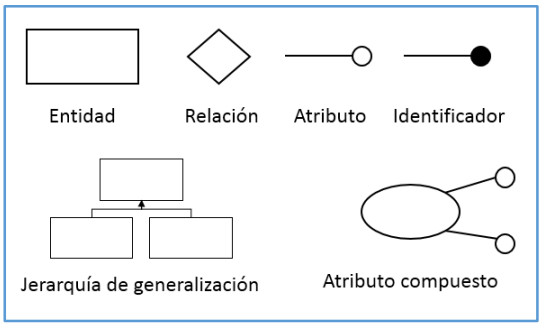


Figura 3. Conceptos del modelo entidad-relación.

Modelo de datos orientado a objetos

El modelo orientado a objetos representa las entidades del mundo real como objetos en lugar de registros y extiende la representación de estas añadiendo nociones de encapsulación, métodos e identidad de objetos. Dicho modelo tiende a funcionar bien en sistemas con comportamiento complejo, o en los que se accede a los datos por navegación en una jerarquía natural (IBM Corp, 2007). El modelo orientado a objetos no está exento de deficiencias, a continuación se listan las más significativas:

- 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.

A modo de resumen y después de haber analizado cada modelo de datos, se puede afirmar que los modelos de datos que mejor se adecuan a la propuesta de solución son el modelo entidad-relación (nivel conceptual) y el modelo relacional (nivel lógico). En ambos casos prevaleció la sencillez y flexibilidad de su estructura así como la fortaleza de sus fundamentos teóricos.

1.4. Diseño de base de datos

El autor de la presente investigación, en concordancia con (AIU, 2014), reconoce que el diseño de bases de datos no es un proceso sencillo, habitualmente la complejidad de la información y la cantidad de requisitos de los sistemas hacen que sea complicado. Por estos motivos, cuando se diseña es recomendable aplicar la estrategia de divide y vencerás; fragmentando el proceso en varias etapas:

- **Diseño conceptual:** Incluye la creación de un esquema o modelo conceptual de la base de datos. Este modelo es independiente de las consideraciones físicas, incluyendo los Sistemas de Gestión de Base de Datos, los lenguajes de programación y las plataformas de hardware. Los usuarios que por su formación no están familiarizados con los elementos técnicos deben entender el esquema, así que no debería contener detalles sobre cómo se implementará la base de datos. Sin embargo, este debe estar detallado en términos de la naturaleza, la estructura y el significado de los datos.
- **Diseño lógico:** Es el proceso de construir un esquema de la información que utiliza la empresa, basándose en un modelo de base de datos específico e independiente del SGBD concreto que se vaya a utilizar, así como de cualquier otra consideración física (Marqués, 2011). En esta etapa, se transforma el esquema conceptual obtenido en la etapa anterior del diseño, en un esquema lógico que utilizará las estructuras de datos del modelo de base de datos que se vaya a utilizar.
- **Diseño físico:** El propósito de esta etapa es describir cómo se va a implementar físicamente el esquema lógico obtenido en la fase anterior. Esta etapa parte del hecho de que ya se ha resuelto la problemática de la estructuración de la información, y permite concentrarnos en las cuestiones tecnológicas.

1.4.1. Patrones de diseño de bases de datos

Según (Blaha, 2010) un patrón es una solución probada a un problema común en un contexto dado. Los patrones de diseño de base de datos proveen soluciones abstractas y reutilizables que

enriquecen el lenguaje de modelado. Estos permiten generar modelos robustos de forma rápida, reduciendo posibles errores al modelar la base de datos. A continuación se describen diferentes patrones centrados en la estructura de datos (entidades y relaciones):

- **Patrón árbol simple:** Utilizado cuando el árbol es la representación de una estructura de datos. Los elementos a almacenar son del mismo tipo, es decir, pueden ser almacenados en la misma entidad. Como restricción, el árbol no puede tener ciclos y un hijo no puede ser su propio padre (Ver Figura 4). Es útil en diversos escenarios y un ejemplo práctico donde se evidencia su uso es en la estructura de mando de una empresa.

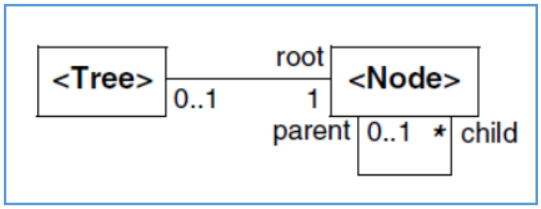


Figura 4. Patrón árbol simple.

- **Patrón árbol fuertemente codificado:** Un árbol puede ser definido como una jerarquía de nodos que tiene un nodo raíz. Un nodo puede considerarse como una entidad cuyos registros están organizados como un árbol. Un nodo particular puede, o no, ser la raíz (Blaha, 2010). Como se puede apreciar en la Figura 5, este patrón representa un árbol mediante una jerarquía de entidades. Es utilizado para asegurar la secuencia de tipos en cada nivel de la jerarquía, y cuando la estructura está bien definida y es poco probable que cambie.

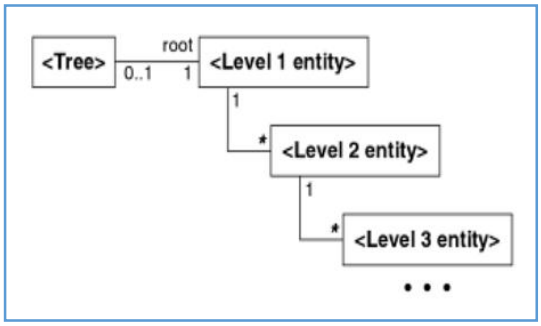


Figura 5. Patrón árbol fuertemente codificado.

- **Patrón árbol estructurado:** Este modelo es usado cuando se necesita diferenciar los nodos hojas *(leaf)*, de aquellos que generan una nueva rama *(branch)*, porque ambos tipos de nodos tienen diferentes atributos, relaciones y/o semántica. En el mismo no pueden

existir ciclos, es decir, un hijo no puede ser su propio padre. La generalización tiene cubrimiento total y exclusivo, cada elemento de la entidad *Node*, debe tener su correspondiente elemento en la entidad *Leaf* o en la entidad *Branch.*

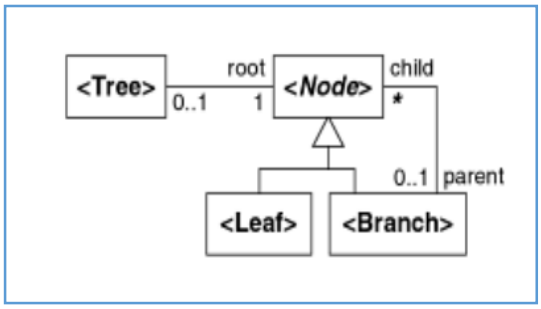


Figura 6. Patrón árbol estructurado.

- **Patrón grafo dirigido simple:** El patrón grafo dirigido simple se utiliza cuando todos los nodos contienen el mismo tipo de datos. Es similar al árbol simple, la diferencia es que en este caso la relación recursiva sobre *Node* tiene cardinalidad de muchos a muchos.

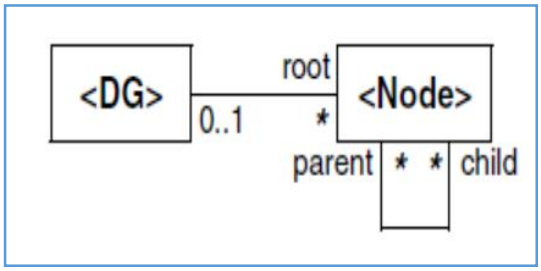


Figura 7. Patrón grafo dirigido simple.

- **Patrón grafo dirigido estructurado:** Este modelo es usado cuando se necesita diferenciar los nodos hojas (*leaf*), de aquellos que generan una nueva rama (*branch*), porque ambos tipos de nodos tienen diferentes atributos, relaciones y/o semántica. Es similar al árbol estructurado, la diferencia es que en este caso la relación recursiva sobre *Node* tiene cardinalidad de muchos a muchos.

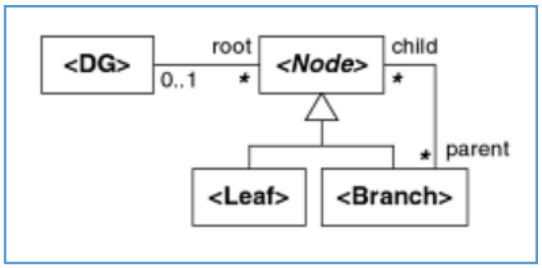


Figura 8. Patrón grafo dirigido estructurado.

- **Patrón de llaves subrogadas:** Este patrón es muy utilizado pues facilita la interacción con la BD en un futuro. El mismo plantea que se genere una llave primara única para cada entidad, en vez de usar un atributo identificador en el contexto dado. Esto permite que las tablas sean más fáciles de consultar a partir del identificador, pues todos tienen el mismo tipo en cada una de las tablas.

1.5. Marco de trabajo (*framework*) de desarrollo de base de datos

Hoy en día existe abundante bibliografía de base de datos, de diseño y programación en SQL, de lenguajes de manipulación de datos, administración y configuración según el sistema gestor seleccionado. Pero no abunda teoría de cómo coordinar intencionada y justificadamente todos los elementos requeridos para el desarrollo de una base de datos en un proceso de desarrollo de software (Osorio, y otros, 2013). Metodologías tradicionales como RUP (*Rational Unified Process*) abordan el desarrollo de base de datos como una tarea de diseño, careciendo de especificaciones necesarias en todo el esfuerzo de desarrollo y mantenimiento de una base de datos. Situación de la cual no están exentas las metodologías ágiles, al no distinguir claramente entre desarrollo de aplicaciones y desarrollo de base de datos dentro del proceso de construcción de un software.

Dbplanning es un framework de desarrollo de base de datos que posibilita al equipo establecer una línea base de desarrollo de base de datos (Osorio, y otros, 2013). O sea, permite una coordinación justificada e intencionada de los principales elementos de base de datos dentro del desarrollo de software. Para ello transita por tres fases o etapas fundamentales como se muestra en la Figura 9. Además define las siguientes actividades: Modelado de Datos, Configuraciones, Implementación y ADTP (Acceso a Datos, *Tuning*[1] y Prueba).

[1] Referente el proceso de optimización de las bases de datos (por su traducción del inglés).

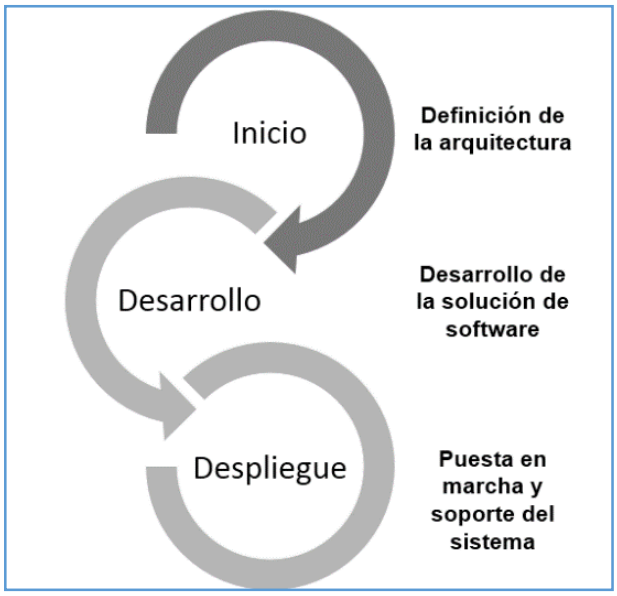


Figura 9. Fases genéricas dentro del proceso de desarrollo de software.

El framework asume los roles de Arquitecto de Datos, Administrador de Base de Datos y Desarrollador de Base de Datos para la ejecución de las actividades y la elaboración de los artefactos que define. El Arquitecto de Datos es el encargado de la realización de la actividad de Modelado de Datos y la elaboración de los artefactos Arquitectura de Datos y Modelo de Datos. El Desarrollador de Base de Datos es el encargado de la realización de las actividades de Implementación y ADTP, y la elaboración de los artefactos Código Fuente, Scripts de Escenarios y Acceso a Datos. Y el Administrador de Base de Datos es el encargado de la realización de la actividad de Configuraciones y la elaboración del artefacto Configuraciones.

Por todo el antes mencionado el autor se inclina el empleo de Dbplanning para guiar el ciclo de desarrollo de las bases de datos para los módulos Administrativo y Revisión de Causas Penales.

1.6. Estrategias de indexado

Cuando el volumen de datos almacenados es grande el manejo y consulta de los mismo puede tornarse lento y costoso en recursos computacionales. De ahí que muchos gestores tengan creada una estrategia para eliminar estas deficiencias. Una muy utilizada son los índices, punteros que al igual que los índices de los libros sirven para encontrar más rápido aquello que se desea buscar. Por lo tanto y extrapolándolo a las bases de datos se puede decir que los índices sirven para agilizar las consultas a las tablas.

Los índices son una estructura física que permite un tipo de acceso alternativo al secuencial, es creado a partir de una o varias columnas de una tabla. Al insertar índices en las tablas se deben tener en cuenta los siguientes factores:

- Es usual indexar aquellos campos que son más utilizados en las búsquedas (los que aparecen en las cláusulas WHERE o JOIN) para mejorar una consulta con el operador (SELECT).
- Son recomendables sobre campos con valores únicos, pues funcionan de una mejor manera si el campo no tiene valores duplicados.
- Es mejor indexar campos con valores de la menor longitud posible, preferiblemente enteros y si se indexa un campo de texto, se debe evitar hacerlo sobre campos de longitud variable.
- Por último se recomienda evitar crear índices innecesariamente pues estos se actualizan con cada cambio en la tabla asociada y pueden ralentizar las modificaciones de la misma.

PostgreSQL utiliza varios tipos de índices. Están los Hash; que son una tabla hash o mapa hash, la operación principal que soporta de manera eficiente es la búsqueda. También están los GIST (*Generalized Search Tree*), que no son un solo tipo de índice, sino más bien una infraestructura dentro de la cual muchas estrategias diferentes de indexación se pueden aplicar. Otro de los tipos son los GIN (*Generalized Inverted Index*), que no son más que índices invertidos que pueden controlar los valores que contienen más de una clave, por ejemplo las matrices; al igual que con GIST, GIN puede soportar muchas estrategias de indexación diferentes definidas por el usuario.

Otra de las estrategias de indexado es “B-tree” o Árbol-B, la cual constituye la estrategia estándar del SGBD PostgreSQL. La misma permite encontrar un único valor o explorar en un área de distribución la búsqueda de los valores claves mediante el empleo de los operadores: <, <=, =, >, =>; característica que la convierte en la estrategia más recomendable para la optimización de consultas a campos recurrentes en la base de datos.

1.7. Normalización de base de datos

La normalización es un método para asegurar que el modelo de datos sea preciso, consistente, simple, no redundante y estable. Constituyendo un elemento clave en el diseño de bases de datos OLTP (*Online Transaction Processing*), no tanto así en bases de datos OLAP (*Online Analytical Processing)* ya que en estas últimas es habitual la redundancia de información y las operación de inserción y actualización no son fundamentales para su operación (Osorio, et al., 2013).

La teoría de la normalización se ha desarrollado para obtener estructuras de datos eficientes que reduzcan los problemas de anomalías de inserción, actualización y eliminación. El concepto de

normalización fue introducido por Codd y pensado para aplicarse en sistemas relacionales. Sin embargo, tiene aplicaciones más amplias (Mato, 2005).

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

Es importante destacar que en este proceso para alcanzar una forma normal superior es necesario cumplir con las reglas que esta define, así como con las de la forma normal anterior. O sea, las relaciones en la 1FN son un subconjunto del universo de todas las relaciones posibles. Las relaciones en la 2FN son un subconjunto de las que están en 1FN y así sucesivamente, como se muestra en la Figura 10.

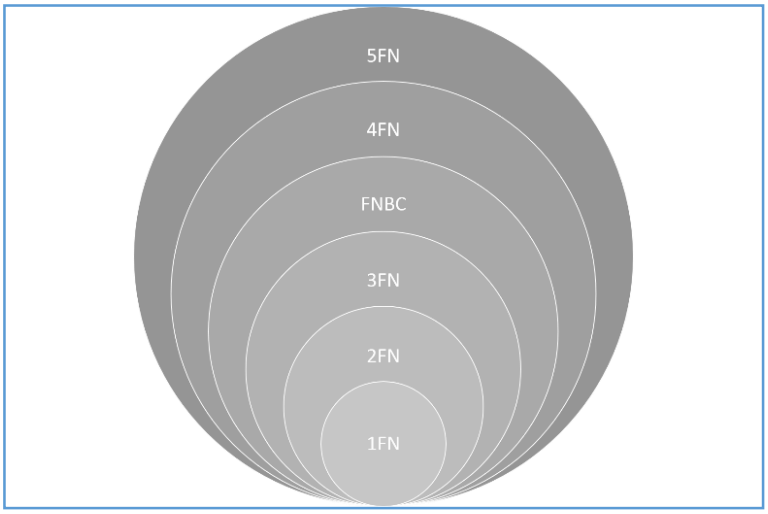


Figura 10. Formas Normales Anidadas.

Primera Forma Normal: Describe una relación en la cual no existen grupos repetitivos (atributo que contiene un conjunto de valores y no un único valor) y los dominios de los atributos contienen únicamente valores atómicos. Formalmente hablando una relación está en 1FN si cumple la propiedad de que sus dominios no tienen elementos que, a su vez, sean conjuntos. Los valores que puede tomar un atributo de una relación son los elementos contenidos en su correspondiente dominio (Mato, 2005).

Segunda Forma Normal: Una relación está en 2FN si además de estar en 1FN, todos los atributos que no forman parte de ninguna clave candidata tienen dependencia funcional completa respecto de cada una de las claves. Toda relación cuya clave está formada por un solo atributo está en 2FN.

Tercera Forma Normal: Una relación está en 3FN si además de estar en 2FN, los atributos que no forman parte de ninguna clave candidata facilitan información solo acerca de las claves y no acerca de otros atributos. Cada atributo no clave es dependiente no transitivamente de la clave primaria. Se eliminan las dependencias funcionales transitivas.

Forma Normal de Boyce-Codd: Una relación está en FNBC si lo está en 3FN y si además el conocimiento de las claves permite averiguar todas las relaciones existentes entre los datos de la relación. Las claves candidatas deben ser los únicos descriptores sobre los que se facilita información por cualquier otro atributo. Cada determinante (atributo con el cual otro atributo tiene dependencia funcional total) debe ser una clave candidata. Se eliminan claves candidatas compuestas que se solapan. No siempre es posible transformar un esquema de relación en FNBC sin que se produzca pérdida de dependencias funcionales. Si se puede hacer sin pérdida de información.

Cuarta Forma Normal: La 4FN se asegura de que las dependencias multievaluadas independientes estén correcta y eficientemente representadas en un diseño de BD (Pérez, 2011).

Quinta Forma Normal: La quinta forma normal, también conocida como forma normal de proyección-unión, es un nivel de normalización de bases de datos designado para reducir redundancia en las bases de datos relacionales que guardan hechos multivalores aislando semánticamente relaciones múltiples. Una tabla se dice que está en 5FN si y sólo si está en 4FN y cada dependencia de unión (JOIN) en ella es implicada por las claves candidatas (Pérez, 2011).

Denormalización: Una de las tareas que se realizan en el diseño lógico, después de obtener un esquema lógico normalizado, es la de considerar la introducción de redundancias controladas y otros cambios en el esquema. En ocasiones puede ser conveniente relajar las reglas de normalización introduciendo redundancias de forma controlada con el objetivo de mejorar las prestaciones del sistema (Marqués, 2011). Por regla general, la denormalización puede ser una opción viable cuando las prestaciones que se obtienen no son las deseadas y las tablas involucradas se actualizan con poca frecuencia pero se consultan muy a menudo. Aunque es importante no pasar por alto que la denormalización puede hacer que los accesos a datos sean rápidos, pero ralentiza las actualizaciones.

Autores como (Marqués, 2011) aseguran que para obtener un esquema con una estructura consistente y sin redundancias es recomendable llegar hasta la 3FN, posición que es compartida por el autor.

1.8. Integridad en las base de datos

Las restricciones de integridad proporcionan un medio de asegurar que las modificaciones hechas a la base de datos por los usuarios autorizados no provoquen la pérdida de la consistencia de los datos. Por tanto, protegen a la base de datos contra los daños accidentales.

De las restricciones de integridad existentes el autor se identifica con las planteadas por (Silberschatz, y otros, 2002):

- **Restricciones de dominio:** Los límites de dominios son la forma más elemental de restricciones de integridad. Son fáciles de probar por el sistema siempre que se introduce un nuevo dato en la base de datos. La declaración de que un atributo pertenezca a un determinado dominio actúa como una restricción sobre los valores que puede tomar. Las restricciones de los dominios no sólo permite verificar los valores introducidos en la base de datos, sino también examinar las consultas para asegurarse de que tengan sentido las comparaciones que hagan.
- **Restricción de valores nulos:** Para determinados atributos, los valores nulos pueden ser inapropiados. El SQL estándar permite que la declaración del dominio de un atributo incluya la especificación *not null.* Esto prohíbe la inserción de un valor nulo para este atributo. Cualquier modificación de la base de datos que causara que se insertase un valor nulo en un dominio *not null* genera un diagnóstico de error. Hay muchas situaciones en las que la prohibición de valores nulos es deseable. Un caso particular en el que es esencial prohibir los valores nulos es en la clave primaria de un esquema de relación.
- **Integridad referencial:** A menudo se desea asegurar que un valor que aparece en una relación para un conjunto de atributos determinado aparezca también en otra relación para un cierto conjunto de atributos. Esta condición se denomina integridad referencial.

Es importante no dejar de mencionar las restricciones de clave, las cuales parten de del precepto de que todos los elementos de un conjunto son distintos; por tanto, todas las tuplas de una relación deben ser distintas. Esto significa que no puede haber dos tuplas que tengan la misma combinación de valores para todos sus atributos (Elsmari, 1993).

1.9. Seguridad en las bases de datos.

La seguridad de las bases de datos se refiere a la protección frente a accesos malintencionados, la cual si bien no es absoluta puede elevar lo suficiente el coste para quien lo comete como para disuadir la mayor parte, si no la totalidad, de los intentos de tener acceso a la base de datos sin la

autorización adecuada (Silberschatz, y otros, 2002). Entre las violaciones de seguridad más comunes se encuentran:

- La lectura no autorizada de los datos (robo de información)
- La modificación no autorizada de los datos
- La destrucción no autorizada de los datos

Para proteger la base de datos hay que adoptar medidas de seguridad en varios niveles:

- **Sistema de bases de datos:** Puede que algunos usuarios del sistema de bases de datos sólo estén autorizados a tener acceso a una parte limitada de la base de datos. Puede que otros usuarios estén autorizados a formular consultas pero tengan prohibido modificar los datos. Es responsabilidad del sistema de bases de datos asegurarse de que no se violen estas restricciones de autorización.
- **Sistema operativo:** Independientemente de lo seguro que pueda ser el sistema de bases de datos, la debilidad de la seguridad del sistema operativo puede servir como medio para el acceso no autorizado a la base de datos.
- **Red:** Dado que casi todos los sistemas de bases de datos permiten el acceso remoto mediante terminales o redes, la seguridad en el nivel del software de la red es tan importante como la seguridad física, tanto en Internet como en las redes privadas de las empresas.
- **Físico:** Los sitios que contienen los sistemas informáticos deben estar protegidos físicamente contra la entrada de intrusos.
- **Humano:** Los usuarios deben ser autorizados cuidadosamente para reducir la posibilidad de que alguno de ellos dé acceso a intrusos a cambio de sobornos u otros favores.

En tal sentido y atendiendo a la necesidad de asegurar los datos e información que se manejan, los SGBD establecen claves para identificar al personal autorizado a utilizar la base de datos. PostgreSQL 9.1 presenta una serie de características que facilita el trabajo para tener un alto nivel de seguridad:

- Protección de los ficheros de la base de datos, porque todos los ficheros almacenados en la base de datos están protegidos contra escritura por cualquier cuenta que no sea la del súper usuario de postgres.
- Las conexiones de los clientes se pueden restringir por dirección IP y/o por nombre de usuario mediante el fichero pg_hba.conf.

- A cada usuario de postgres 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 estos grupos.

Características que serán explotadas en la propuesta de base de datos para los módulos Administrativo y Revisión de Causas Penales, en aras de contribuir con la seguridad de la información almacenada.

1.10. Herramientas y tecnologías para el desarrollo y pruebas de la base de datos

El éxito de cualquier sistema informático depende en gran medida de las tecnologías y herramientas que se utilizan para su desarrollo. Es por ello que en este punto es necesario analizar las facilidades que ofrecen las herramientas definidas en la Arquitectura Base para el desarrollo del sistema en cuestión.

Visual Paradigm 8.0

Herramienta CASE (*Computer Aided Software Engineering*) que permite generación de código y la base de datos a partir de los diagramas UML (*Unified Modeling Language*) realizados. Principalmente es utilizada en la modelación de negocio y en el diseño es de gran ayuda a los analistas porque pueden visualizar el flujo central y detallado de cada proceso mediante diagramas, además posibilita crear un conjunto amplio de artefactos utilizados con mucha frecuencia durante las disciplinas análisis, diseño, implementación o el despliegue. Esta herramienta permite a múltiples usuarios trabajar sobre el mismo proyecto (Visual Paradigm, 2013).

Subversion 1.5

Herramienta para el control de versiones bajo licencia Apache/BSD. Subversion puede acceder al repositorio a través de redes, lo que le permite ser usado por personas que se encuentran en distintos ordenadores. A cierto nivel, la capacidad para que varias personas puedan modificar y administrar el mismo conjunto de datos desde sus respectivas ubicaciones fomenta la colaboración. Se puede progresar más rápidamente sin un único conducto por el cual deban pasar todas las modificaciones. Y puesto que el trabajo se encuentra bajo el control de versiones, no hay razón para temer porque la calidad del mismo vaya a verse afectada por la pérdida de ese conducto único, si se ha hecho un cambio incorrecto a los datos, simplemente se deshace ese cambio (Collins-Sussman, et al., 2004).

PGAdmin III 1.14.0

Es una aplicación gráfica para el Gestor de Base de Datos PostgreSQL, siendo la más completa y popular con licencia *open source*. Está escrita en C++ usando la biblioteca gráfica multiplataforma wxWidgets, lo que permite que se pueda usar en sistemas operativos Linux, FreeBSD, Solaris, Mac OS y Windows. Es capaz de gestionar versiones a partir de la PostgreSQL 7.3 (Ubuntu, 2008). Está diseñado para responder a las necesidades de múltiples usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados y mucho más. La conexión al servidor puede hacerse mediante conexión TCP/IP o Unix Domain Sockets (en plataformas Unix), y puede encriptarse mediante SSL (*Secure Socket Layer*) para mayor seguridad (Ubuntu, 2008).

ORM Doctrine 2.3.2

Es un marco de trabajo de mapeo de objeto relacional (ORM) para *PHP* 5.3.0 o versiones superiores, el cual proporciona persistencia transparente de objetos PHP, situándose en la parte superior de una poderosa capa de abstracción de base de datos (*DBAL por Data Base Abstraction Layer*). La principal tarea de los ORM para PHP es la traducción transparente entre objetos (PHP) y las filas relacionales de la base de datos (Blanco, 2012).

IDE Netbeans 7.3

NetBeans es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Existe además un número importante de módulos para extender este entorno de desarrollo integrado. Es un producto libre y gratuito sin restricciones de uso (Netbeans, 2013). Entre las novedades principales de esta versión destaca mejoras en el soporte para la nueva sintaxis de Java 7, soporte de HTML (*HyperText Markup Language*) versión 5, mejoras al editor Java y soporte para PHP 5.3.

Datanamic Data Generator for PostgreSQL

Generador de datos que incluye características útiles para la ejecución de pruebas a bases de datos PostgreSQL entre las que podemos mencionar (Datanamic Solutions BV, 2014):

- Acceso directo a bases de datos PostgreSQL (con controlador nativo)
- Rellenar la base de datos directamente o generar scripts de inserción
- Soporte para PostgreSQL versión 7,8 y 9

- Carga datos aleatorios de fuentes tanto externas como internas
- Previsualización en tiempo real de los datos que se generan
- Configuración para definir el porcentaje de valores nulos

Apache JMeter 2.9

Es un software de código abierto realizado en Java, puede ser utilizado para probar el rendimiento tanto en recursos estáticos y dinámicos. Se puede utilizar para simular una carga pesada en el servidor, de red o un objeto para probar su resistencia o para analizar el rendimiento general bajo diferentes tipos de carga. Se puede utilizar para hacer un análisis gráfico de rendimiento o para probar su servidor / script / comportamiento del objeto bajo carga pesada concurrentes (Jmeter, 2013).

1.11. Conclusiones parciales

Una vez realizado un estudio detallado de la teoría existente acerca de los sistemas de base de datos, se puede concluir que:

- PostgreSQL como Sistema Gestor de Bases Datos dota a los desarrolladores de una herramienta robusta y acorde a las necesidades de desarrollo.
- Los modelos de datos deberán determinar la organización y estructura de las bases de datos.
- Dbplanning Framework constituye una guía certera para el desarrollo de la base de datos para los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de la Gestión de las Fiscalías, al establecer una línea base de desarrollo y darle el seguimiento pertinente dentro del ciclo de desarrollo del software.
- Para obtener un esquema con una estructura consistente es recomendable llegar hasta la 3FN, sin descartar la introducción de redundancia en el modelo en aras de lograr un mejor rendimiento de la base de datos.
- Es necesario implementar acciones que contribuyan a garantizar el aseguramiento de la integridad y seguridad de los datos.
- Las herramientas definidas responden a las necesidades de desarrollo del sistema, cumpliendo estas con los requisitos indispensables para el entorno nacional como el de ser multiplataforma y software libre.

Capítulo 2. Análisis de la solución propuesta

2.1. Introducción

En el presente capítulo se abordan especificaciones del desarrollo de la base de datos, como son las configuraciones del entorno trabajo, nomenclaturas que se deben aplicar para hacer más entendible el resultado final, así como otros aspectos de interés. Sobre la solución obtenida se abordan las descripciones de los datos, los patrones utilizados, las formas de acceso y las estrategias utilizadas para la optimización de la base de datos.

2.2. Descripción de la arquitectura de datos y políticas de respaldo

Como parte del desarrollo de la aplicación SIGEF II, se tiene concebida una distribución de datos e información a lo largo de las 15 provincias y 168 municipios, incluyendo el municipio especial Isla de la Juventud. Para ello los servidores de base de datos estarán situados en cada una de las fiscalías del país y los datos deben fluir bidireccionalmente entre instancias superiores e inferiores como se muestra en la Figura 11.

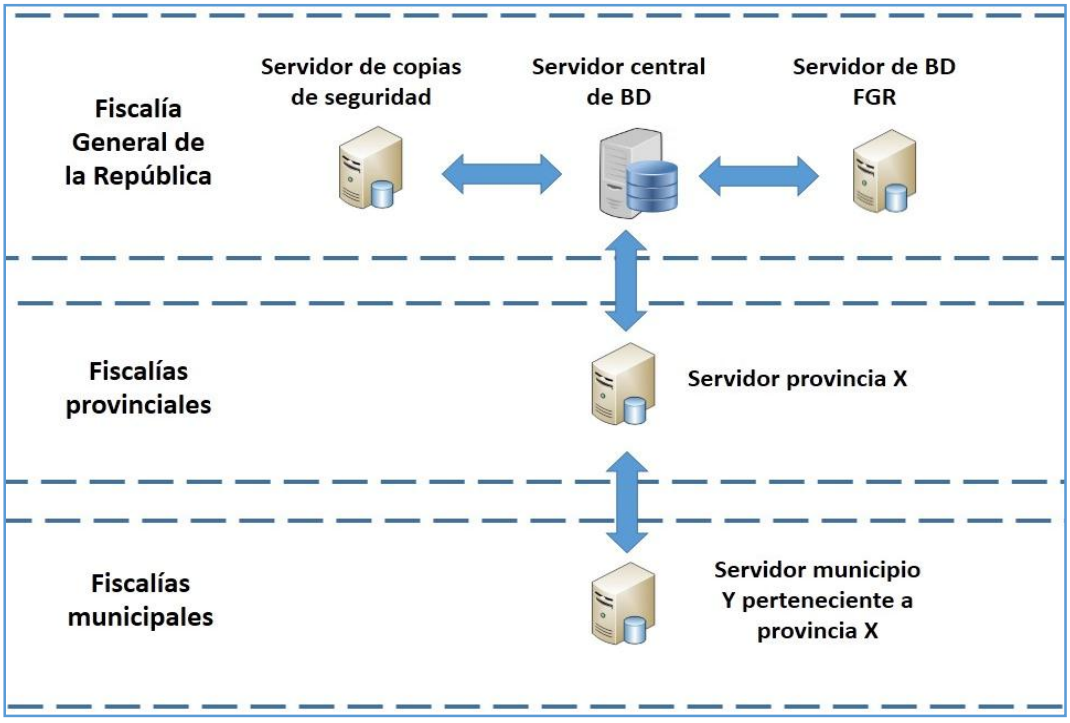


Figura 11. Distribución de datos e información

En concordancia con las políticas para el respaldo de información existente en la Fiscalía General de la República, se han definido un conjunto de acciones con la finalidad de minimizar o evitar la pérdida de los datos almacenados en la base de datos de SIGEF II, en caso de presentarse alguna falla o contingencia (Pupo, 2010):

- Realizar una copia de seguridad (*backup*) de la base de datos mensualmente y guardar copias anteriores en dependencia con los recursos tecnológicos. La combinación de copias de seguridad normal e incremental será la empleada para tal fin, al requerir del mínimo espacio de almacenamiento posible y ser uno de los métodos de copia de seguridad más rápidos.
- Realizar pruebas a las copias de seguridad mensualmente utilizando para ello servidores pasivos.

2.3. Descripción del entorno de desarrollo

Resulta importante destacar que el entorno de desarrollo utilizado para implementar la base de datos de los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de la Gestión de las Fiscalías ha sido constituido de forma similar al entorno real donde se desplegará la aplicación. En el cual el equipo de desarrollo cuenta con un repositorio Subversion para el control de versiones y un servidor compartido en el cual se alojan el servidor de base de datos y el servidor de aplicaciones. Cada uno con sistema operativo Ubuntu LTS 12.04, 2 GB de memoria RAM y 160 GB de espacio en disco duro (Ver Figura 12). De forma consecuente, los servidores y estaciones de trabajo disponen de las mismas aplicaciones definidas en la Arquitectura Base del sistema. Para el modelado de datos fue utilizada la herramienta Visual Paradigm en su versión 8.0, como herramienta de gestión y diseño de base de datos el PGAdmin III en su versión 1.14.0. Para la implementación de la capa de acceso a datos el IDE NetBeans en su versión 7.3 y como Sistema Gestor de Base de Datos el PostgreSQL en su versión 9.1.

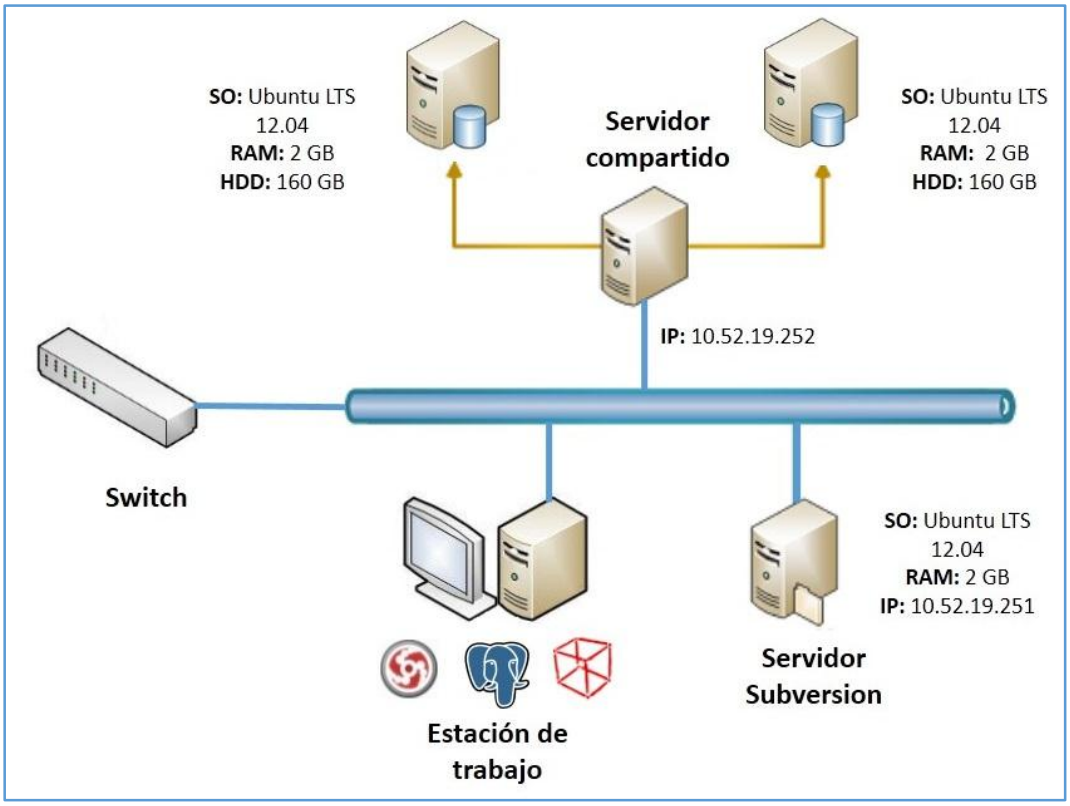


Figura 12. Descripción del entorno de desarrollo.

2.4. Nomenclatura y normas para el modelo de base de datos

Con el objetivo de contribuir a la calidad del producto y propiciar una buena comunicación entre los integrantes del equipo de desarrollo, se utilizan los estándares y buenas prácticas de codificación propuestas en la Arquitectura Base del sistema (Acosta, 2010):

Generalidades

- Todos los caracteres para nombrar atributos y entidades estarán escritos con minúscula.
- Caracteres como la **ñ**, **punto** y **tildes** no estarán permitidos.

Nomenclatura para entidades

- Los nombres deberán ser claros y estar en consonancia con el negocio, sin exceder para ello los 35 caracteres. Además deberán adaptarse a los nombres definidos en el diagrama de clases.

- Los nomencladores estarán descritos inicialmente por una **n** y a continuación el nombre identificativo de la entidad especificada en el diagrama de clases. Ejemplo: **n**tipobien.
- Las tablas de datos variados tendrán de prefijo una **d** y a continuación el nombre identificativo de la entidad, especificada en el diagrama de clases. Ejemplo: **d**bien.

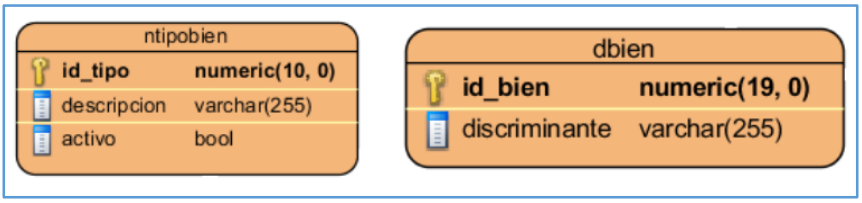


Figura 13. Ejemplos de nomenclatura para entidades.

- En las relaciones de mucho a mucho donde siempre se genera una nueva tabla, el nombre estará compuesto por el de las dos tablas separado por un guión bajo (_). Ejemplo: dsolicitudprueba_nsolicitadopor.
- Los nombres compuestos se escribirán juntos. Ejemplo: **dsolicitudprueba.**

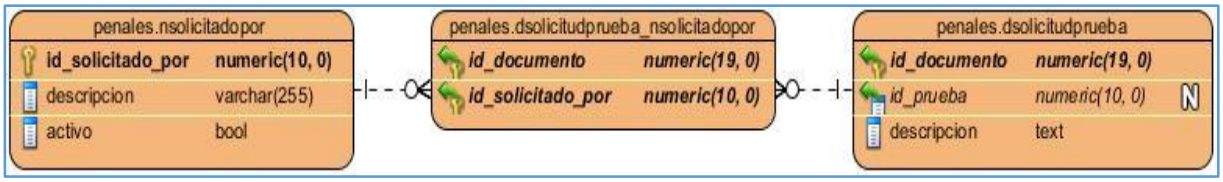


Figura 14. Ejemplos de nomenclatura para entidades.

Nomenclatura para atributos

- Los atributos tendrán nombres identificativos y claros de lo que representan y no excederán los 35 caracteres.
- Las llaves primarias tendrán como prefijo **id** seguido de un guión bajo (_) y el nombre de la tabla. Ejemplo: **id**_decision de la entidad ndecision.
- Para atributos con nombres compuestos se deberá separar las palabras con guión bajo (_), tratando siempre de no superar los 35 caracteres.

Nomenclatura para funciones

- Las funciones se nombrarán según el tipo de funcionalidad a la que responden, utilizando un verbo que describa el tipo de operación que realiza.

Nomenclatura para índices

- Los índices se nombrarán utilizando el prefijo **idx** seguido de guión bajo (_) y nombre del campo.

Nomenclatura para disparadores (triggers)

- Los disparadores se nombrarán utilizando el prefijo **tr** seguido del nombre de la tabla. De forma similar las funciones asociadas a los triggers se nombran utilizando el prefijo **ftr** seguido del nombre de la tabla.

Nomenclatura para secuencias

- Las secuencias se nombrarán siguiendo la siguiente estructura: ***nombre entidad_nombre campo_seq***. Ejemplo: **dbien_id_bien_seq**.

2.5. Modelo de datos

El modelo de datos que a continuación se presenta describe la estructura fundamental de datos asociados a los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de la Gestión de las Fiscalías. Dicho modelo está conformado por el diseño lógico y el diseño físico.

2.5.1. Modelo lógico

A partir del esquema o modelo conceptual se realizó el diseño lógico de la base de datos. El mismo brinda la flexibilidad de analizar y diseñar la aplicación lógicamente, sin depender de una tecnología o Sistema Gestor de Base de Datos específico. A partir de la generación del diseño lógico se determina la organización de los datos que se necesitan almacenar en la base de datos antes de ser creada.

Es importante destacar que el modelo lógico que a continuación se muestra es un diseño macro del modelo lógico general (Ver Anexos 1 y 2), utilizado para representar los principales procesos que se gestionan en los módulos Administrativo y Revisión de Causas Penales.

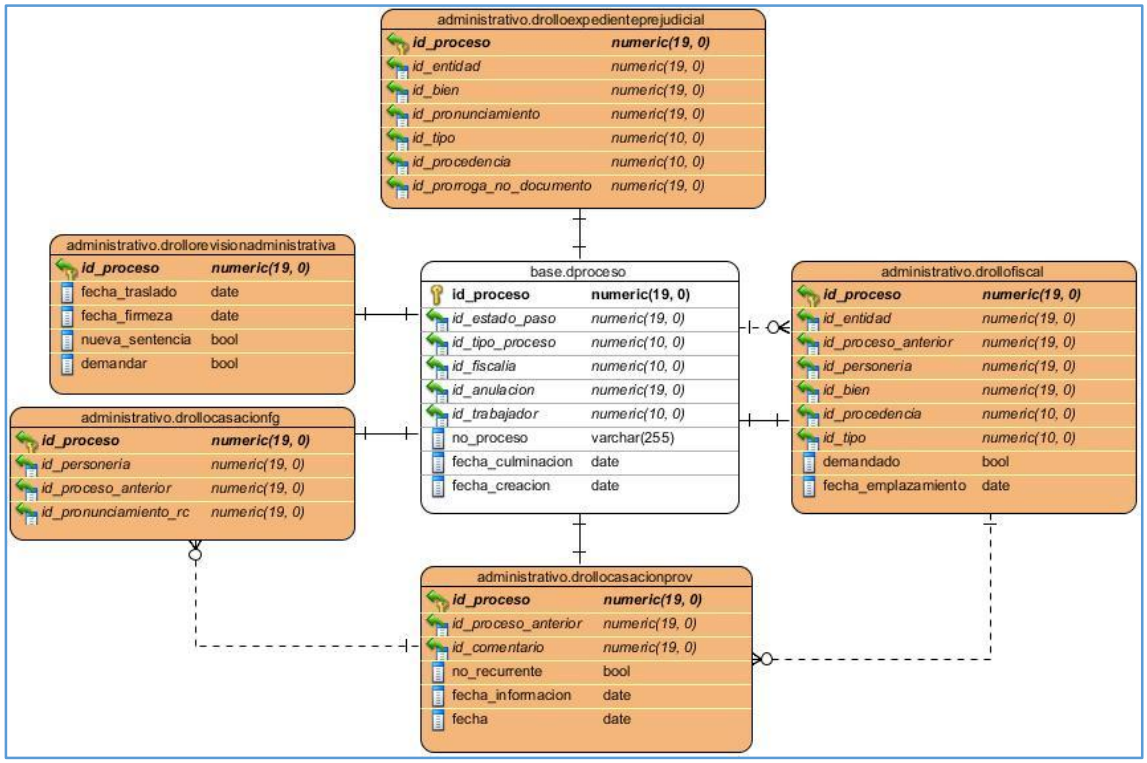


Figura 15. Modelo lógico perteneciente al módulo Administrativo.

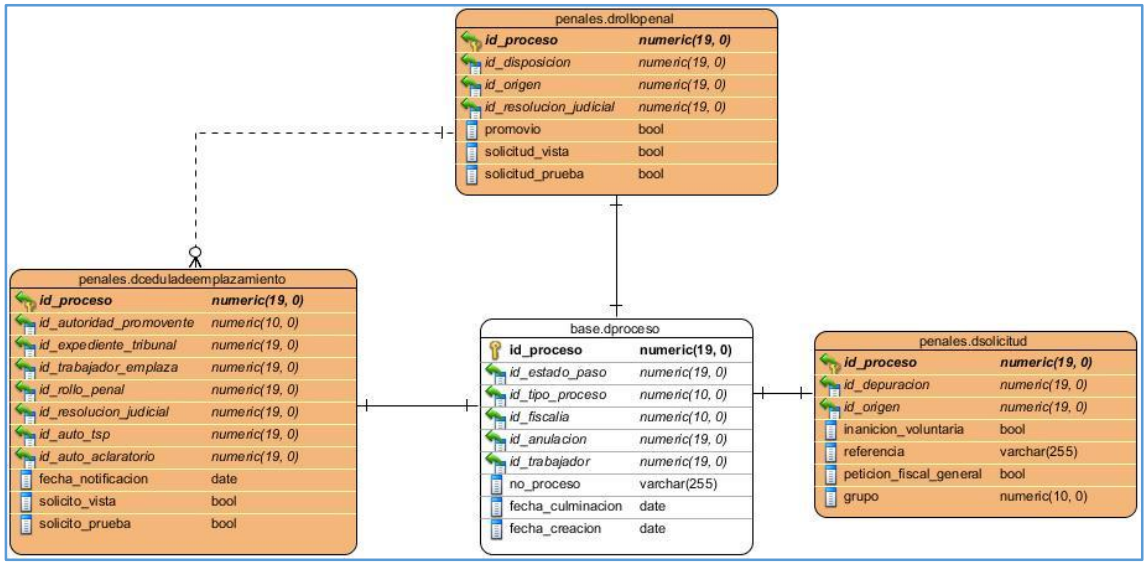


Figura 16. Modelo lógico perteneciente al módulo Revisión de Causas Penales.

2.5.2. Modelo físico

Con el objetivo de lograr un mejor entendimiento del modelo de datos del sistema, el mismo fue dividido en escenarios o sub modelos, los cuales se corresponden con cada uno de los subprocesos que se gestionan en los módulos Administrativo y Revisión de Causas Penales. A continuación se describe y representa el extracto del modelo físico de datos concerniente a las funcionalidades determinadas para cada uno de los escenarios. Se describen además las entidades representativas de cada uno de ellos, tomando como criterio de selección el grado de asociación de estas con gran parte de la información que se desea almacenar.

Escenario Expediente Prejudicial

Abarca la creación y consulta de expedientes prejudiciales, los cuales contienen las pruebas del derecho vulnerado y los documentos generados durante la investigación. A continuación se muestra el diseño del sub modelo físico, así como la descripción de la entidad representativa del escenario en cuestión.

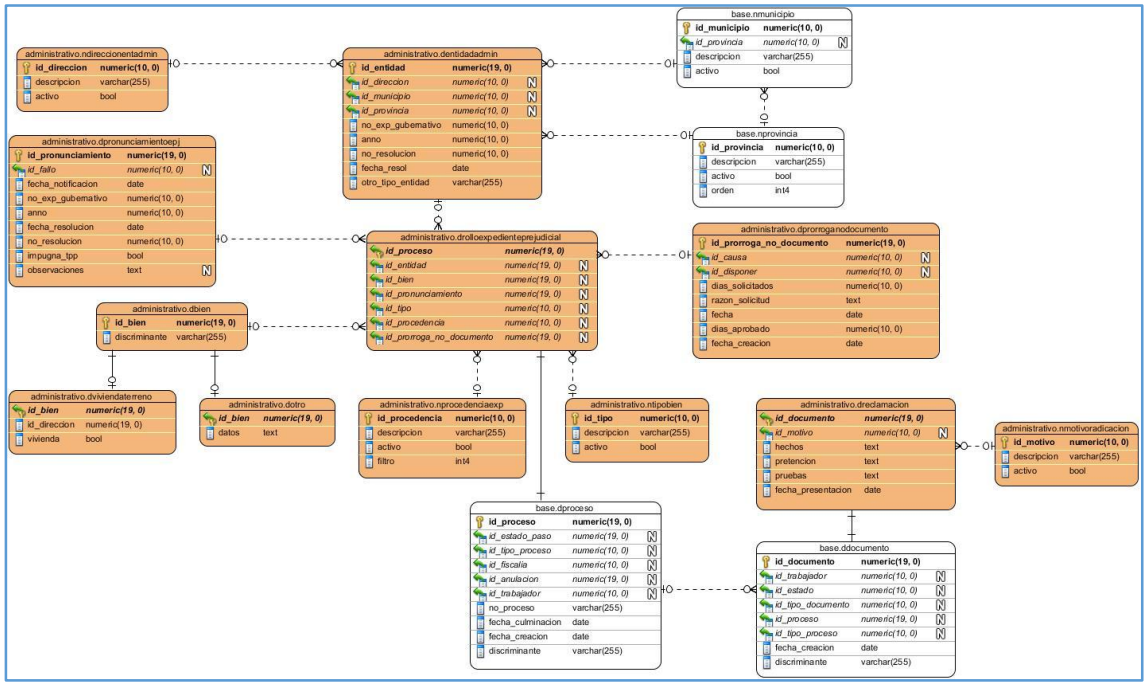


Figura 17. Modelo físico de escenario Expediente Prejudicial.

Nombre entidad		drolloexpedienteprejudicial
Atributos	**Tipo**	**Descripción**
id_proceso	*numeric(19)*	Identificador del proceso.
id_entidad	*numeric(19)*	Identificador de la entidad administrativa.
id_bien	*numeric(19)*	Identificador del bien.
id_pronunciamiento	*numeric(19)*	Identificador del pronunciamiento.
id_tipo	*numeric(10)*	Identificador del tipo de bien.
id_procedencia	*numeric(10)*	Identificador de la procedencia del Expediente Prejudicial.
id_prorroga_no_documento	*numeric(19)*	Identificador de la prorroga en tiempo.

Tabla 1. Entidad representativa del escenario Expediente Prejudicial.

Escenario Rollo Fiscal

Almacena la información asociada a los rollos o procesos fiscales, los cuales pueden iniciar como continuidad del proceso Expediente Prejudicial o puede tener lugar en las fiscalías provinciales y en la Fiscalía General de la República sin antes haber transitado por el proceso anteriormente mencionado.

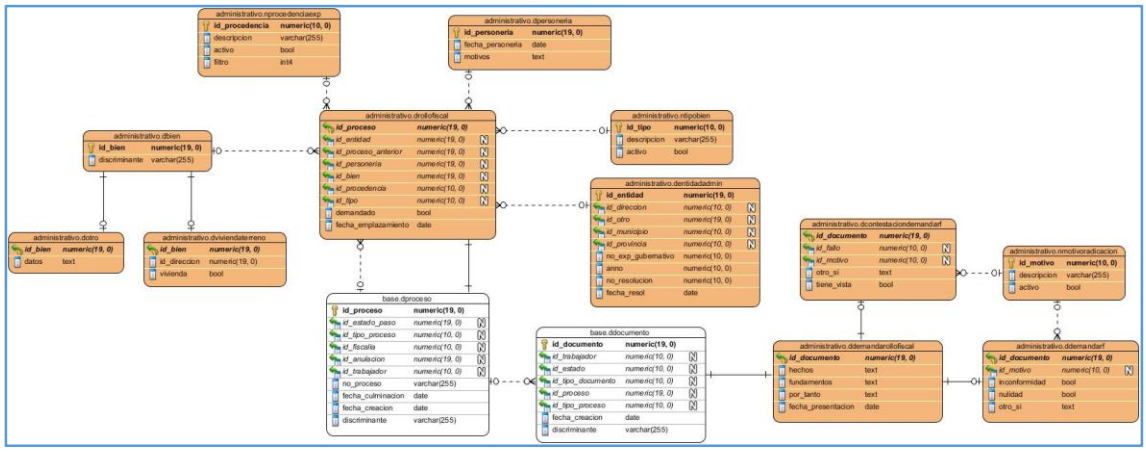


Figura 18. Modelo físico de escenario Rollo Fiscal.

Nombre entidad		drollofiscal
Atributos	**Tipo**	**Descripción**
id_proceso	*numeric(19)*	Identificador del proceso.
id_entidad	*numeric(19)*	Identificador de la entidad administrativa.
id_personeria	*numeric(19)*	Identificador del documento de personería.
id_bien	*numeric(19)*	Identificador del bien.
id_tipo	*numeric(10)*	Identificador del tipo de bien.
id_procedencia	*numeric(10)*	Identificador de la procedencia del Rollo Fiscal.
demandado	*bool*	Define la actuación del Fiscal.
fecha_emplazamiento	*date*	Fecha de información al Fiscal del proceso.

Tabla 2. Entidad representativa del escenario Rollo Fiscal.

Escenario Rollo de Casación

Almacena la información concerniente a las inconformidades presentes en un proceso fiscal. Puede ser de tipo Casación Provincial o Casación Fiscalía General en dependencia de la instancia donde se realice. A continuación se muestra el sub modelo del escenario en cuestión, así como las descripciones de las entidades representativas.

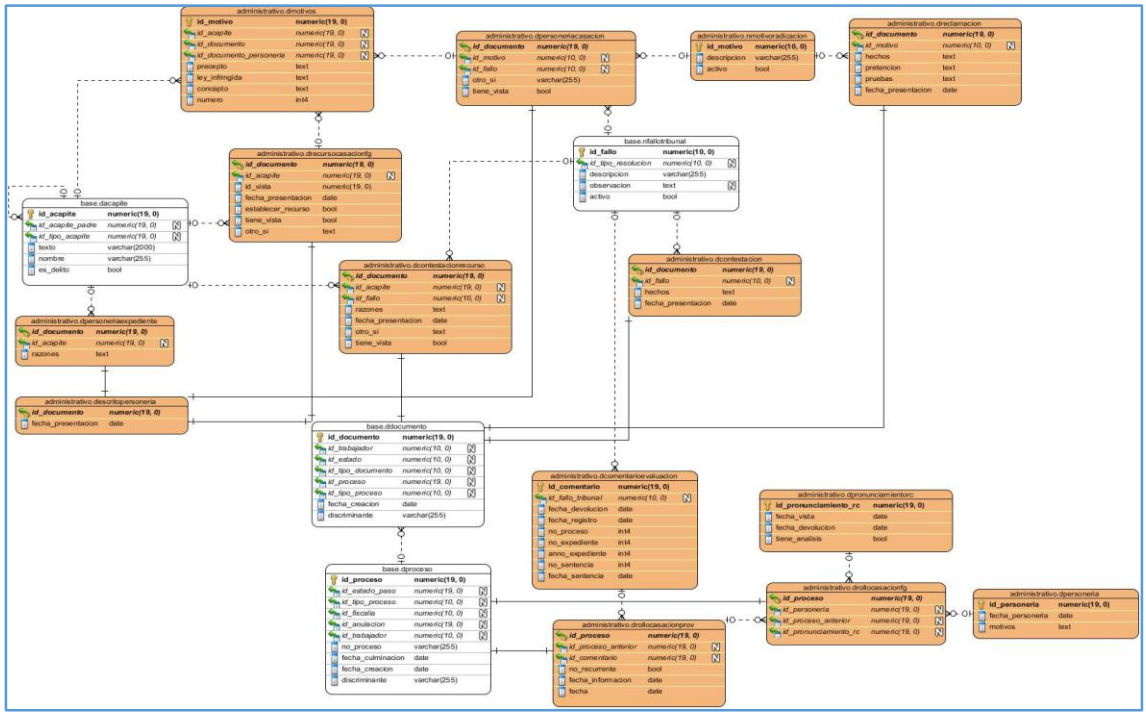


Figura 19. Modelo físico de escenario Rollo Casación.

Nombre entidad		drollocasacionprov
Atributos	**Tipo**	**Descripción**
id_proceso	*numeric(19)*	Identificador del proceso.
id_proceso_anterior	*numeric(19)*	Identificador del proceso que le antecede (Rollo Fiscal).
id_comentario	*numeric(19)*	Identificador de las conclusiones del proceso recibidas desde la Fiscalía General de la República.
no_recurrente	*bool*	Define la actuación del Fiscal.
fecha_informacion	*date*	Fecha de información del proceso a la Fiscalía General.
fecha	*date*	Fecha de información al Fiscal del proceso.

Tabla 3. Entidad representativa del escenario Rollo de Casación.

Nombre entidad		drollocasacionfg
Atributos	**Tipo**	**Descripción**
id_proceso	*numeric(19)*	Identificador del proceso.
id_personeria	*numeric(19)*	Identificador del acto de personería.
id_proceso_anterior	*numeric(19)*	Identificador del proceso anterior.
id_pronunciamiento_rc	*numeric(19)*	Identificador de la conclusión asociada al proceso.

Tabla 4. Entidad representativa del escenario Rollo de Casación.

Escenario Rollo Revisión Administrativa

Recoge los datos asociados a los proceso de Revisión Administrativa. El diseño del sub modelo físico, así como la descripción de la entidad representativa del escenario se puede observar a continuación.

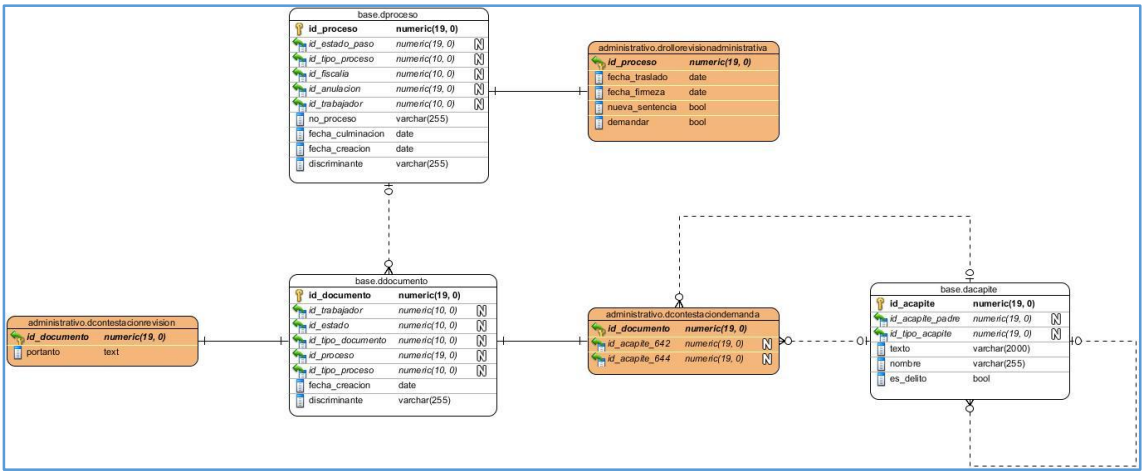


Figura 20. Modelo físico de escenario Rollo Revisión Administrativa.

Nombre entidad		drollorevisionadministrativa
Atributos	**Tipo**	**Descripción**
id_proceso	*numeric(19)*	Identificador del proceso.
fecha_traslado	*date*	Fecha de notificación al Fiscal.
fecha_firmeza	*date*	Fecha en que la sentencia se hace firme para revisar.
nueva_sentencia	*bool*	Verifica si la sentencia a revisar es nueva.
demandar	*bool*	Define la actuación del Fiscal.

Tabla 5. Entidad representativa del escenario Rollo Revisión Administrativa.

Escenario Solicitud

Abarca el proceso de solicitud de revisiones penales. El mismo recoge los datos del sancionado con su(s) causa(s) y los del solicitante. Luego el Fiscal depura la solicitud decidiendo si se traslada, tramita o archiva. A continuación se muestra el diseño del sub modelo físico, así como la descripción de la entidad más importante.

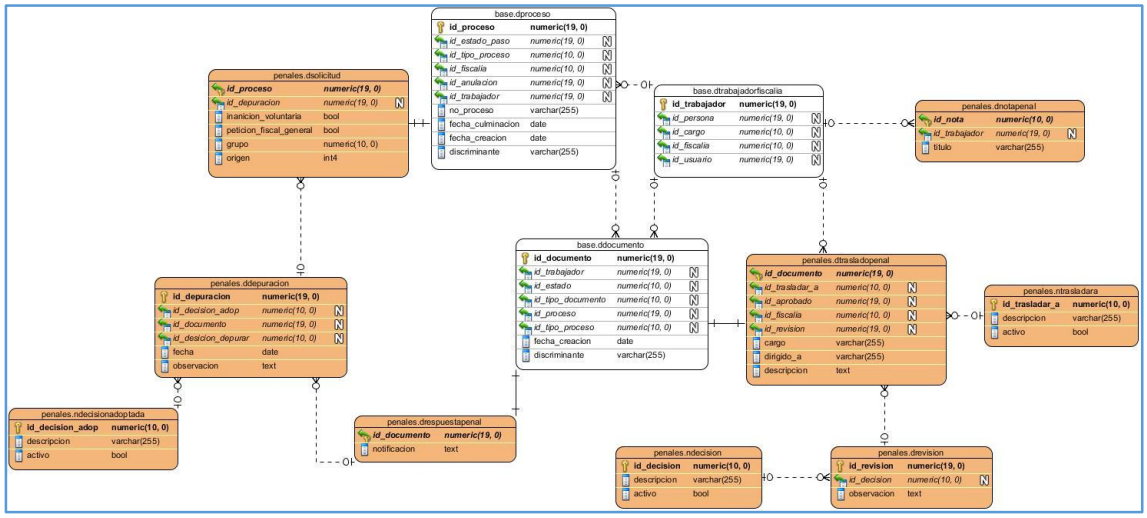


Figura 21. Modelo físico de escenario Solicitud.

Nombre entidad		dsolicitud
Atributos	**Tipo**	**Descripción**
id_proceso	*numeric(19)*	Identificador del proceso.
id_depuracion	*numeric(19)*	Identificador de la depuración.
inanicion_voluntaria	*bool*	Verificar si la solicitud es inanición voluntaria.
peticion_fiscal_general	*bool*	Verifica si la solicitud es a petición del Fiscal General.
grupo	*numeric(10)*	Número que permite agrupar las solicitudes.
origen	*integer*	Número que especifica el origen de la solicitud.

Tabla 6. Entidad representativa del escenario Solicitud.

Escenario Rollo Penal

En el mismo se gestiona la información asociada a los rollos que fueron creados cuando se decide tramitar una solicitud. Agregándole notas y decidiendo que hacer con estos (emitir un dictamen denegatorio o un escrito de promoción). A continuación se muestra el diseño del sub modelo físico, así como la descripción de la entidad representativa del escenario en cuestión.

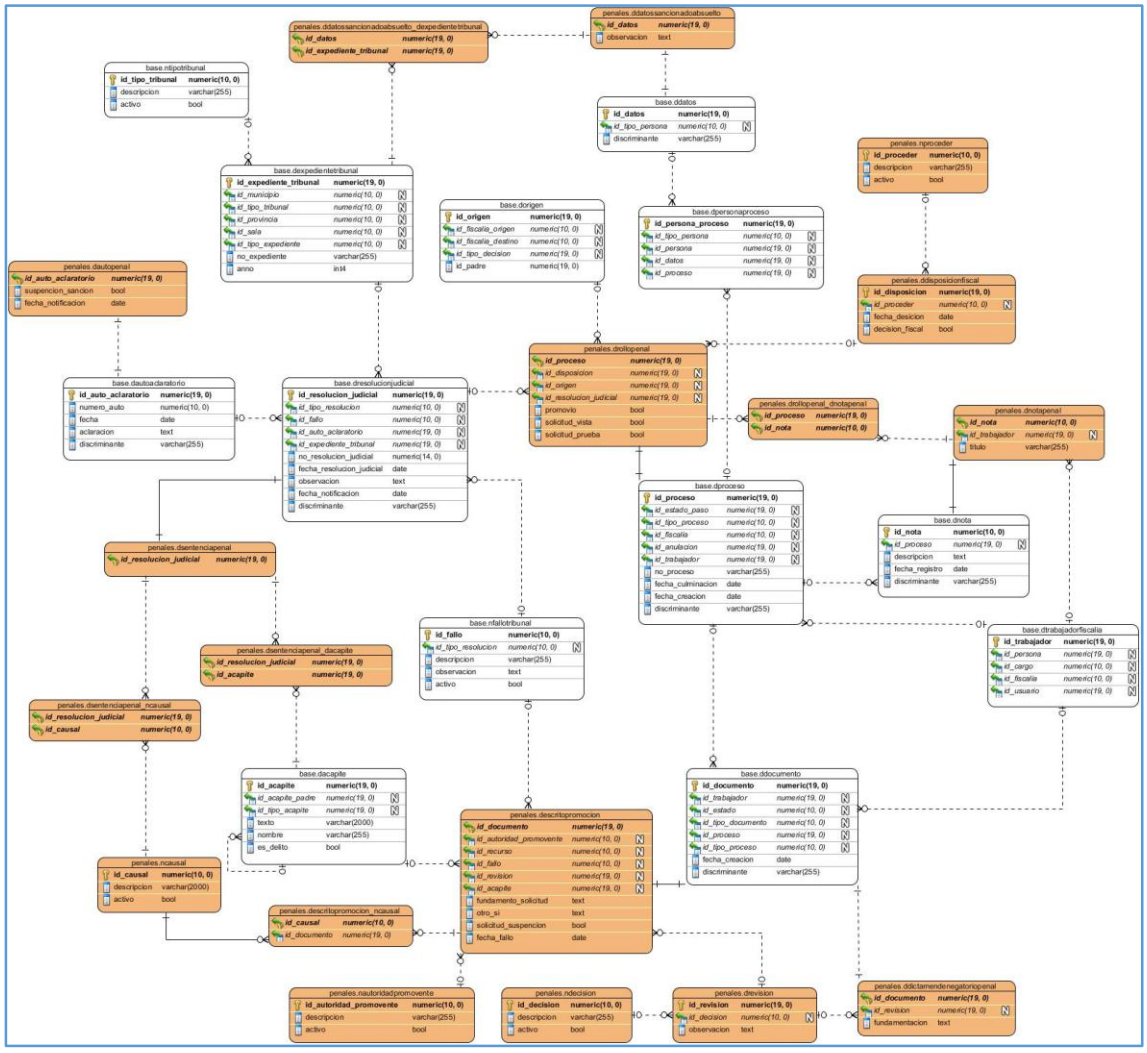


Figura 22. Modelo físico de escenario Rollo Penal.

Nombre entidad		drollopenal
Atributos	**Tipo**	**Descripción**
id_proceso	*numeric(19)*	Identificador del proceso.
id_origen	*numeric(19)*	Identificador del origen de la solicitud.
id_resolucion_judicial	*numeric(19)*	Identificador que hace referencia a la decisión del Tribunal.
id_disposicion	*numeric(19)*	Identificador de la decisión del Fiscal.

promovio	*bool*	Define si el rollo promueve o no.
solicitud_vista	*bool*	Verificar si tendrá una solicitud de vista.
solicitud_prueba	*bool*	Verificar si tendrá una solicitud de prueba.

Tabla 7. Entidad representativa del escenario Rollo Penal.

Escenario Cédula de Emplazamiento

Abarca el proceso de emplazamiento a un Fiscal por el Tribunal y la información asociada a este. El diseño del sub modelo físico, así como la descripción de la entidad representativa del escenario se puede observar a continuación.

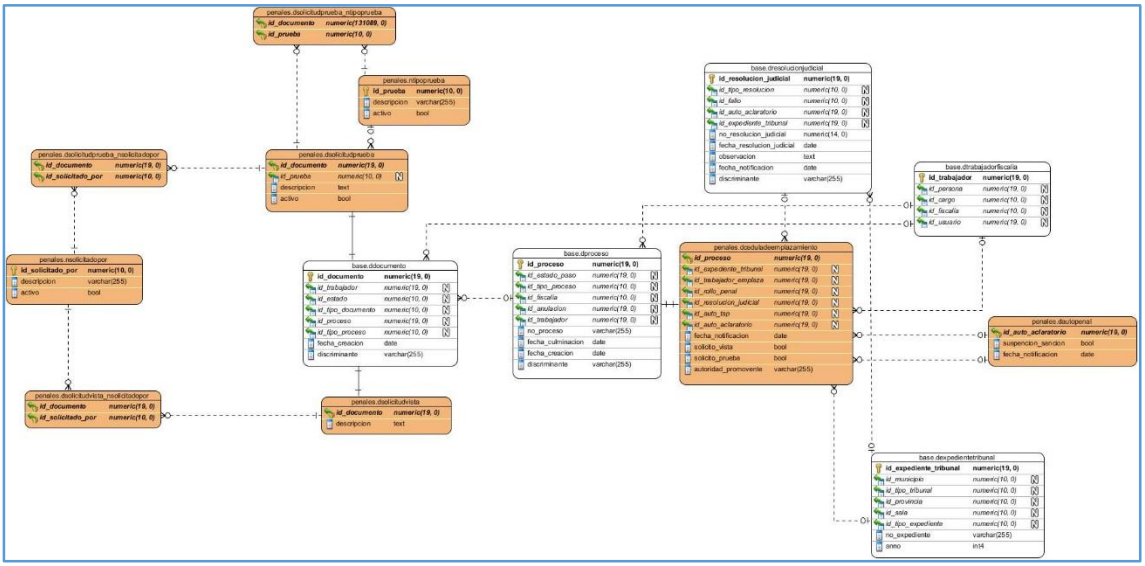


Figura 23. Modelo físico de escenario Cédula de Emplazamiento.

Nombre entidad		**dceduladeemplazamiento**
Atributos	**Tipo**	**Descripción**
id_proceso	*numeric(19)*	Identificador del proceso.
id_expediente_tribunal	*numeric(19)*	Identificador del Expediente Tribunal asociado al proceso.
id_rollo_penal	*numeric(19)*	Identificador del proceso en el cual el Fiscal es emplazado.
id_trabajador_emplaza	*numeric(19)*	Identificador del Fiscal emplazado en el proceso.

id_resolucion_judicial	*numeric(19)*	Identificador que hace referencia a la decisión del Tribunal.
id_auto_aclaratorio	*numeric(19)*	Identificador del auto aclaratorio emitido por el Tribunal.
id_auto_tsp	*numeric(19)*	Identificador del auto emitido por el Tribunal Supremo Popular.
fecha_notificacion	*date*	Fecha en que se le notifica al Fiscal que ha sido emplazado.
solicito_vista	*bool*	Verificar si tendrá una solicitud de vista.
solicito_prueba	*bool*	Verificar si tendrá una solicitud de prueba.
autoridad_promovente	*varchar(255)*	Especificación de la autoridad que promueve el emplazamiento.

Tabla 8. Entidad representativa del escenario Cédula de Emplazamiento.

2.5.2.1. Patrones utilizados en el diseño físico de la base de datos

En la praxis es usual la búsqueda de características y/o fenómenos que se repiten con cierta regularidad, y que de su análisis y comprensión se puedan desprender soluciones a los problemas cotidianos. De igual manera en el diseño de las bases de datos y en el modelado de datos en general se presentan elementos repetitivos en disímiles modelos, los que correctamente identificados pasan a ser patrones de diseño. Según (Blaha, 2010). el empleo de patrones asegura mejores resultados y disminuye considerablemente los esfuerzos en la labores de diseño. Atendiendo a esto en la propuesta de base de datos se emplea uno de los patrones de diseño más utilizados en la actualidad, el mismo se describe a continuación:

- **Llaves subrogadas:** Al asignarle una llave única -independiente del contexto- a cada tabla. Empleando para ello como tipo de dato el *numeric* (numérico) de tamaño 19. Un ejemplo lo constituyen las entidades **ntipobien** del módulo Administrativo y **ntipoprueba** del módulo Revisión de Causas Penales.

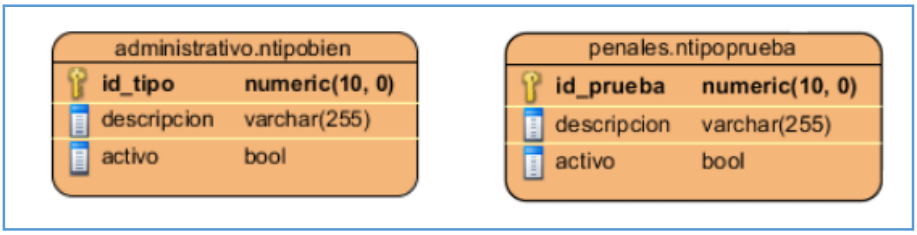


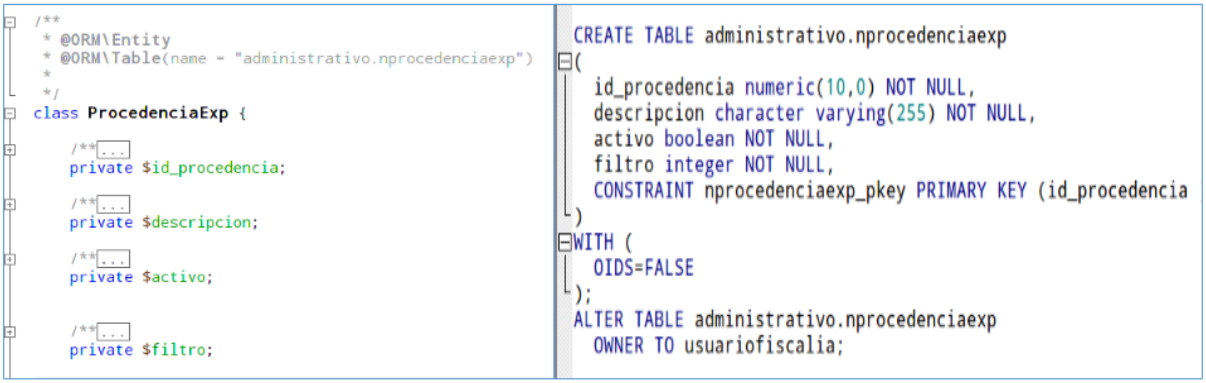
Figura 24. Ejemplos de patrón llaves subrogadas.

2.6. Acceso a datos

En el desarrollo de una aplicación suelen estar involucradas dos entidades diferentes, por una parte el código que mueve la aplicación y por otra los datos que se manejan. Con el tiempo estas dos entidades han evolucionado de manera diferente, y el acceso a los datos desde los programas se ha vuelto una tarea en ocasiones, complicada. Siendo la aparición de los sistemas de Mapeo Objeto-Relacional (ORM) el elemento clave para combatir esta complicación, al proporcionar un acceso a los datos de una manera sencilla y rápida.

Antes de la aparición de estos sistemas las consultas se tenían que realizar a mano dentro de las propias aplicaciones, con lo cual la ventaja de los lenguajes orientados a objetos se perdía, ya que había que crear una petición a la base de datos de manera manual y específica para cada sistema, ya que no todos los Gestores de Base de Datos tienen la misma implementación del lenguaje SQL.

Entre los ORM más conocidos se encuentra Doctrine, el cual permite convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional, es decir, las tablas de la base de datos pasan a ser clases y los registros objetos que podemos manejar con facilidad, como se puede apreciar en la Figura 26.



```
/**
 * @ORM\Entity
 * @ORM\Table(name = "administrativo.nprocedenciaexp")
 *
 */
class ProcedenciaExp {

    /**...
    private $id_procedencia;

    /**...
    private $descripcion;

    /**...
    private $activo;


    /**...
    private $filtro;
```

```
CREATE TABLE administrativo.nprocedenciaexp
(
  id_procedencia numeric(10,0) NOT NULL,
  descripcion character varying(255) NOT NULL,
  activo boolean NOT NULL,
  filtro integer NOT NULL,
  CONSTRAINT nprocedenciaexp_pkey PRIMARY KEY (id_procedencia
)
WITH (
  OIDS=FALSE
);
ALTER TABLE administrativo.nprocedenciaexp
  OWNER TO usuariofiscalia;
```

Figura 25. Ejemplo de mapeo objeto relacional.

Una de las características claves de Doctrine es la posibilidad de escribir las consultas de base de datos en un dialecto SQL propio orientado a objetos llamado Lenguaje de Consulta Doctrine (DQL por *Doctrine Query Language*), inspirado en Hibernates. Además DQL difiere ligeramente de SQL en que abstrae considerablemente la asignación entre las filas de la base de datos y objetos, permitiendo a los

desarrolladores escribir poderosas consultas de una manera sencilla y flexible. (Doctrine Team, 2006-2014).

```
class MotivosRepository extends EntityRepository {
    public function listarMotivoRecursos($id_recurso) {
        $qb = $this->getEntityManager()->createQueryBuilder();
        $qb->select('rc', 'mt')
                ->from('BaseBundle:Administrativo\Motivos', 'rc')
                ->leftJoin('rc.recurso_casacion_fg', 'mt')
                ->orderBy('rc.numero')
                ->where($qb->expr()->eq('mt.id_documento', ':prmRecurso'))
                ->setParameter('prmRecurso', $id_recurso);
            $result = $qb->getQuery()->getResult();
            return $result;
    }

    public function listarMotivoPersoneria($id_recurso) {
        $qb = $this->getEntityManager()->createQueryBuilder();
        $qb->select('rc', 'mt')
                ->from('BaseBundle:Administrativo\Motivos', 'rc')
                ->leftJoin('rc.personeria_casacion', 'mt')
                ->orderBy('rc.numero')
                ->where($qb->expr()->eq('mt.id_documento', ':prmRecurso'))
                ->setParameter('prmRecurso', $id_recurso);
            $result = $qb->getQuery()->getResult();
            return $result;
    }
}
```

Figura 26. Ejemplo de funciones en DQL.

2.6.1. Código fuente

El Código Fuente es uno de los artefacto propuesto por el marco de trabajo Dbplanning El mismo abarca la implementación de las principales funcionalidades de almacenamiento y recuperación de datos, así como sus respectivas descripciones. A continuación una muestra de las principales funcionalidades:

listarProrrogas(): Lista todas las prórrogas en tiempo asociadas a un Expediente Prejudicial almacenadas en el sistema.

listarPronunciamientos($no_expediente): Lista los pronunciamientos que su *no_exp_gubernativo* sea igual al recibido por parámetro.

listarExpedienteParaRolloPaso($id_paso_actual, $lista_paso_anteriores): Dado un identificador del paso actual y una lista de pasos anteriores, devuelve una lista de los Expedientes Prejudiciales adicionados al sistema.

listarRolloAdministrativoPaso($id_paso_actual, $lista_paso_anteriores, $actuacion): Dado el paso actual, una lista de pasos anteriores y una actuación, lista todos los rollos de tipo Revisión Administrativa en los cuales la actuación del Fiscal (atributo *demandar*) sea igual al valor recibido por parámetro.

listarSolicitudes($id_paso, $id_usuario): Lista las solicitudes hechas por el usuario con identificador *$id_usuario* y estén en un paso dado.

listarSolicitudesPorOrigen($_origen): Lista todas las solicitudes que pertenecen a un origen determinado.

listarNotasRolloPenal($id_rollo): Dado el identificador de un Rollo Penal, lista todas las notas asociadas a este.

listarSolicitudesPorGrupo($grupo): Dado un número de grupo, lista las solicitudes que pertenecen al mismo.

2.7. Reportes

Una vez desplegado, SIGEF pretende convertirse en una fuente de información estadística para las más altas instancias del país, facilitando la toma de decisiones a los diferentes niveles del Estado, para ello debe implementar mecanismos que permitan proveer a los usuarios del sistema de una interfaz adecuada para la obtención de dicha información. Siendo los reportes una opción viable al organizar y exhibir la información contenida en la base de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios, de esta forma se le confiere una mayor utilidad a los datos.

En el caso específico de los módulos Administrativo y Revisión de Causas Penales la generación de reportes está encaminada a analizar el comportamiento de indicadores como los recursos de casación y procesos contenciosos en los que interviene el Fiscal en sus diferentes instancias, así como el total de solicitudes de revisión a procesos penales recibidas por la Fiscalía General de la República.

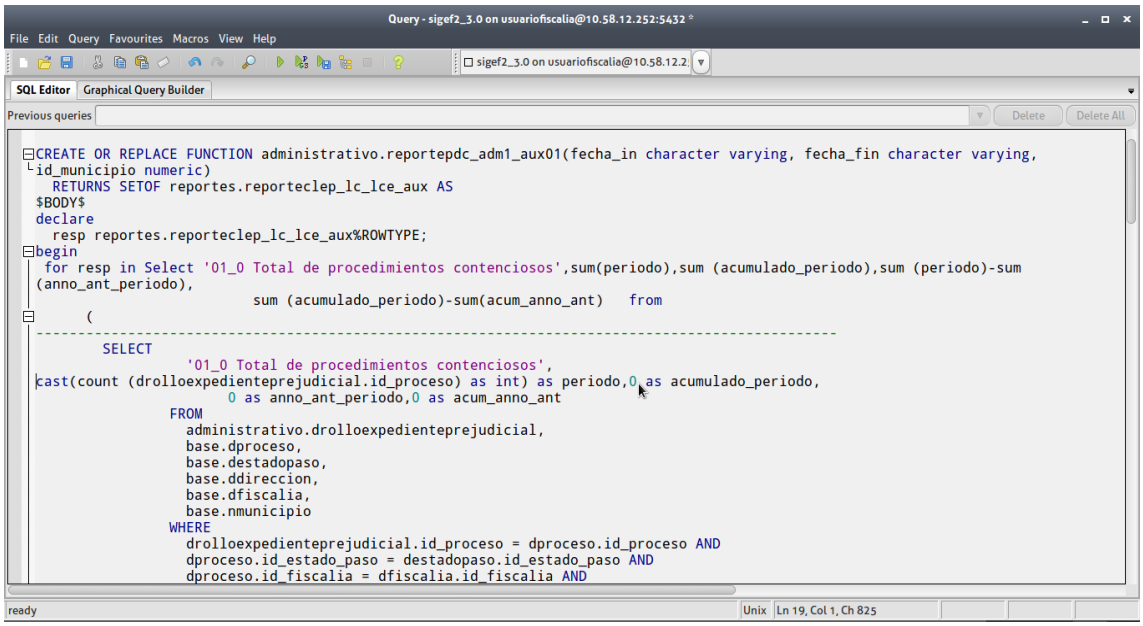


Figura 27. Vista parcial de función implementada para la generación de reportes.

2.8. Normalización del modelo

Los modelos obtenidos de los correspondientes módulos Administrativo y Revisión de Causas Penales se encuentran en 1FN ya que puede asegurar que todos los atributos de las entidades son atómicos, o sea cada tulpa contiene exactamente un valor para cada atributo de las tablas de la base de datos. Además se puede decir que están en 2FN, pues se encuentran en 1FN y todos los atributos que no son claves en las tablas, dependen totalmente de la clave primaria. Al igual están en 3FN, al encontrarse en 2FN y no tener dependencias transitivas los atributos no primos en una misma relación.

Es importante destacar que en aras de mejorar el rendimiento del sistema, o sea agilizar las consultas y transacciones más frecuentes, es que se decide sacrificar las ventajas proporcionadas por la normalización e introducir redundancia en tres de las tablas del modelo: **dsolicitud** y **dceduladeemplazamiento** del esquema penales y **dentidadadmin** del esquema administrativo.

2.9. Estrategia de indexado utilizada

Para la optimización de las operaciones de búsqueda sobre la base de datos se definió utilizar la estrategia de indexado estándar del gestor, “B-tree”. Esta constituye una estructura eficiente y flexible que no requiere de reorganizaciones masivas ya que se va reorganizando poco a poco. Además ofrece gran variedad de operadores, y puede ser utilizada sobre los campos más recurridos en las

consultas, 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 definido, con esto se consume menos tiempo y recursos computacionales.

En la propuesta de base de datos todas las llaves primarias y foráneas poseen índices de tipo “B-tree” lo que implica que cualquier búsqueda que se realice utilizando las llaves se optimizará mediante este método. Además se identificaron un conjunto de atributos recurrentes en las búsquedas los cuales fueron indexados (Ver Figura 29).

```
CREATE INDEX idx_grupo
  ON penales.dsolicitud
  USING btree
  (grupo );

CREATE INDEX idx_expediente_gubernativo
  ON administrativo.dpronunciamientoepj
  USING btree
  (no_exp_gubernativo );
```

Figura 28. Vista de algunos de los campos indexados.

2.10. Conclusiones parciales

En el presente capítulo se realizó una detallada descripción de la solución obtenida en el diseño e implementación de los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de la Gestión de las Fiscalías II. Obteniendo los modelos lógico y físico, como artefactos fundamentales del proceso de diseño. Se explica además las técnicas de optimización utilizadas para dar mayor calidad al resultado obtenido y cuestiones relacionadas con la implementación de funciones para el acceso a datos y la generación de reportes.

Capítulo 3. Validación y pruebas.

3.1. Introducción.

En este capítulo se aborda lo referente a la validación y las pruebas realizadas a la propuesta de base de datos, auxiliándose para ello de técnicas de validación teórica así como pruebas de rendimiento.

3.2. Validación teórica

La validación teórica tiene el propósito de revisar y validar los modelos de datos lógico y físico generados durante la fase de diseño del sistema, comprobando que se aplican las normas básicas de buen diseño de modelos de datos, las nomenclaturas establecidas en la Arquitectura Base y que la estructura de datos permite cubrir los requerimientos de información solicitados.

3.2.1. Métricas

Las métricas son mecanismos que permiten mejorar considerablemente la calidad de productos de software. En el entorno específico de las bases de datos existen diferentes métricas para controlar la calidad de las mismas. Una de las más empleadas es el Ratio de Normalidad o NR (del inglés *Normality Ratio*), del científico norteamericano Jim Gray, la cual mide el nivel de normalización del sistema de base de datos, utilizando para ello la siguiente expresión:

$$NR = \frac{tTFNS}{tTotal}$$

Donde ***tTFNS****:* representa la cantidad de tablas del esquema en tercera forma normal o superior y ***tTotal:*** el número total de tablas.

En el caso de la base de datos desarrollada para los módulos Administrativo y Revisión de Causas Penales los cálculos del Ratio de Normalidad arrojaron los resultados que a continuación se muestran, los cuales indican que el grado de normalización de la base de datos es elevado.

$$NRadministrativo = \frac{tTFNS}{tTotal} \qquad NRadministrativo = \frac{33}{34} = 0.97$$

$$NRpenales = \frac{tTFNS}{tTotal} \qquad NRpenales = \frac{36}{38} = 0.95$$

3.3. Validación funcional mediante pruebas de rendimiento

Las pruebas de rendimiento se realizan desde la perspectiva de encontrar errores en los sistemas que incidan negativamente en el rendimiento del mismo. Estas pruebas pueden servir para demostrar que el sistema cumple los criterios de rendimiento, para comparar dos sistemas y encontrar cuál de ellos funciona mejor o para medir qué partes del sistema provocan que el rendimiento del mismo sea el no deseado. Dentro de las pruebas que existen para evaluar el rendimiento de sistemas se encuentran las pruebas de volumen, las pruebas de carga y estrés, entre otras. A continuación se describen las utilizadas y los resultados que arrojaron.

3.3.1. Pruebas de volumen

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 (Cuesta, 2011).

En la propuesta de base de datos para los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de las Fiscalías, las pruebas de volumen se realizan con el objetivo de identificar los posibles problemas de diseño y comprobar la robustez del modelo, utilizando para ello la herramienta Datanamic Data Generator for PostgreSQL, la cual colma la base de datos de una determinad cantidad de datos previamente establecida, datos que son arbitrarios pero que coinciden en cuanto a tipos y volúmenes con los datos reales que maneja la entidad.

Para tener una idea de la cantidad de datos que se generan en las fiscalías, se toman como muestra las tablas **drolloexpedienteprejudicial** y **drrollopenal**, pues gran parte de las actividades de inserción de datos hacen referencia a estas entidades. Estimando un total de 500 procesos anuales de estas índoles entre todas las fiscalías a nivel nacional y asumiendo que este comportamiento no varíe en los próximos 20 años, es que se define insertar un total de 10 000 tuplas (500 procesos/año x 20 años).

Es importante destacar que el ambiente de la prueba fue restringido a condiciones de hardware inferiores a las existentes en las fiscalías cubanas (Ver Tabla 9).

Indicadores	Entorno de despliegue	Entorno de pruebas
Microprocesador	Quad-Core Intel(R) a 2.6 GHz	Pentium (R) Dual-Core a 1.6 GHz
RAM	2 GB	1 GB
HDD	1TB	160 GB

Tabla 9. Comparación entre ambiente de prueba y condiciones reales de las fiscalías.

Como resultado de la prueba realizada se concluye, que la base datos puede soportar la carga de datos que se genera en las fiscalías del país, sin presentar límites de capacidad, desbordamiento de columnas, atributos o tipos de datos. Las llaves autogeneradas no se salieron del rango especificado, ni se detectaron problemas con los tipos de datos definidos en la etapa de diseño. Tampoco se detectaron peticiones excesivas de hardware durante la ejecución de las pruebas. 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.

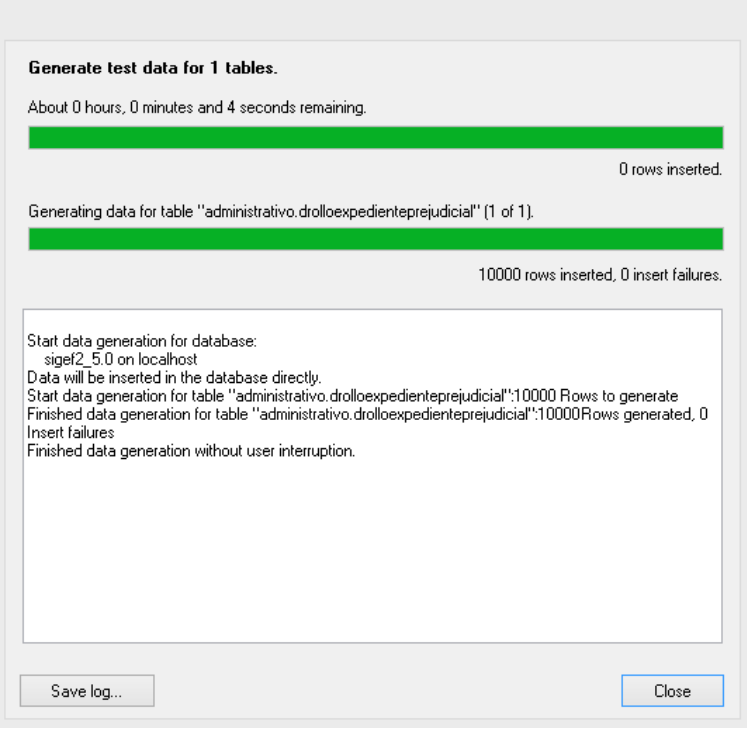


Figura 29.Pruebas de volumen sobre entidad drolloexpedienteprejudicial.

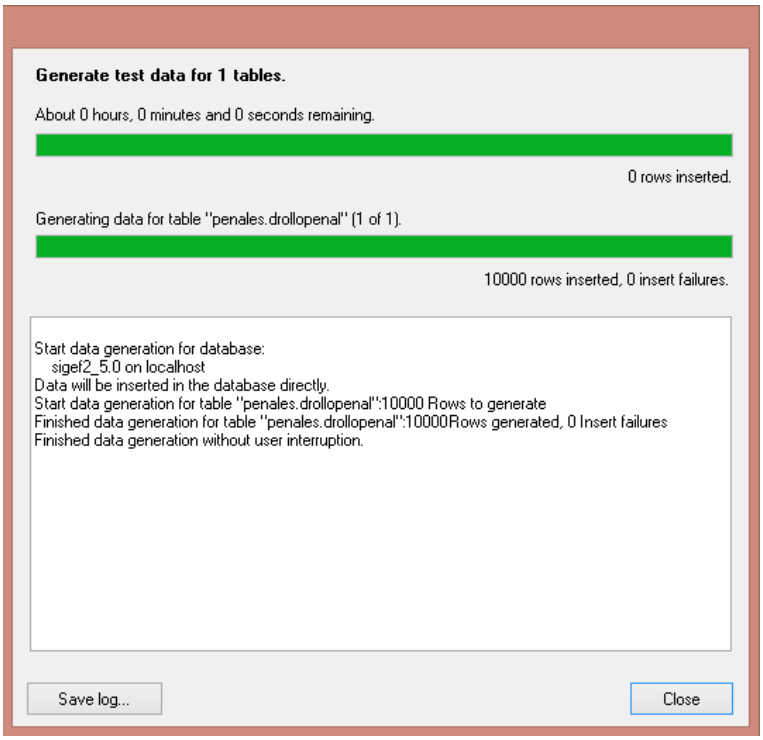


Figura 30. Pruebas de volumen sobre entidad drollopenal.

3.3.2. Pruebas de carga y estrés

La prueba de carga y estrés consiste en someter al sistema a una carga elevada como valores numéricos complejos, elevado número de entradas y de peticiones, que comprueban el límite de carga que la aplicación puede soportar. El testeo de estrés intenta romper el sistema desbordando sus recursos o reduciendo la cantidad de estos. Según (Instituto Tecnológico de Informática, 2014) el objetivo principal de este tipo de pruebas es determinar si el sistema satisface los requisitos de rendimiento y encontrar “cuellos de botella” y soluciones no óptimas. Para ello requieren de gran exactitud y precisión, por lo que se necesitan herramientas automatizadas para su ejecución. Por lo antes mencionado es que se decide emplear la aplicación de código abierto Apache JMeter.

Para realizar las pruebas se creó un plan de pruebas con dos grupos de hilos. En ambos casos se simula la solicitud de consultas SQL de mediana complejidad (Ver Figura 32), repitiendo dicho proceso en 100 ocasiones. En el primer grupo de hilos las consultas son realizadas por 100 usuarios de forma simultánea, mientras que en el segundo estas son realizadas por 150 usuarios, en correspondencia con las distintas instancias de la Fiscalía General de la Republica, para un total de 10 000 y 15 000 solicitudes JDBC (*Java. Database Connectivity*) respectivamente en cada grupo.

```
SELECT
  *
FROM
  administrativo.drolloexpedienteprejudicial,
  base.dproceso,
  administrativo.dentidadadmin,
  administrativo.ndireccionentadmin
WHERE
  drolloexpedienteprejudicial.id_entidad = dentidadadmin.id_entidad AND
  dproceso.id_proceso = drolloexpedienteprejudicial.id_proceso AND
  dentidadadmin.id_direccion = ndireccionentadmin.id_direccion AND
  dentidadadmin.id_direccion = 1;
```

Figura 31. Consulta de mediana complejidad utilizada para pruebas en JMeter.

En la siguiente figura se muestra la configuración de la herramienta JMeter para los grupos de hilos descritos anteriormente.

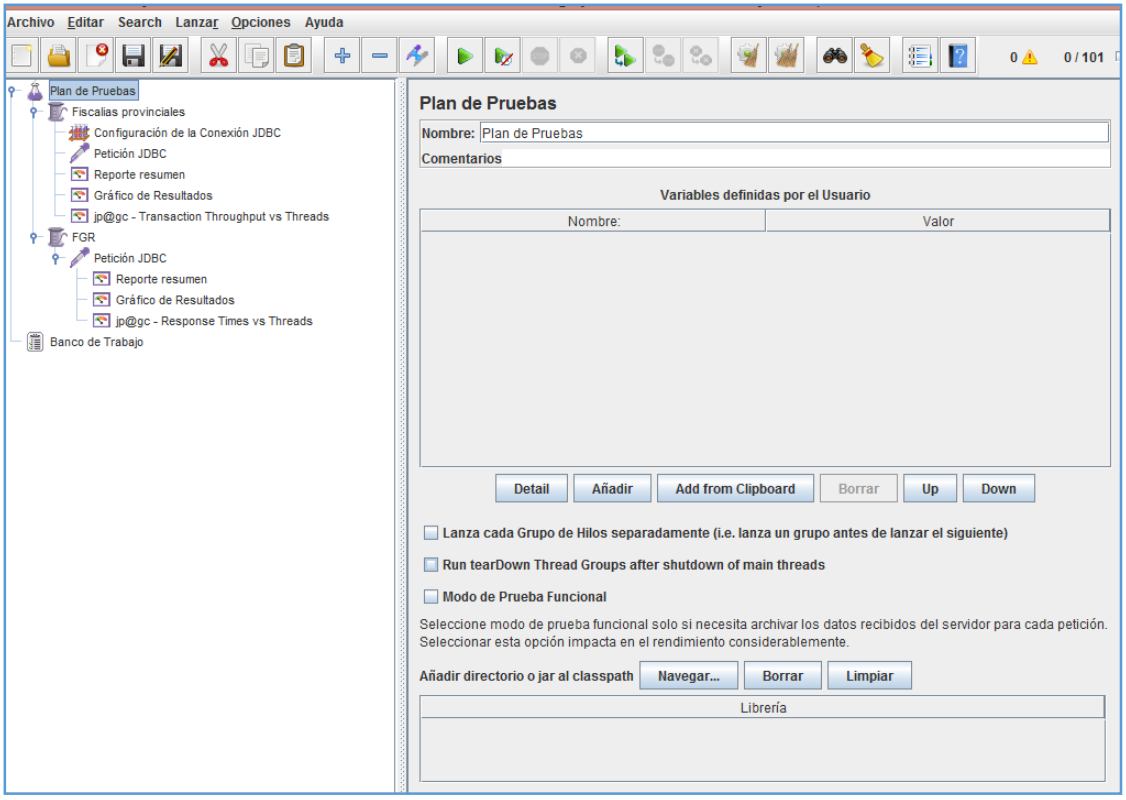


Figura 32. Configuración del plan de pruebas.

Para visualizar los resultados de las pruebas JMeter permite incorporar al plan de pruebas receptores o listeners, los cuales muestran los detalles de las solicitudes de todas las respuestas de los servidores y provee una gran cantidad de información de forma gráfica y tabular. A continuación se muestran alguno de ellos y la descripción de sus parámetros.

Reporte resumen: Muestra un informe detallado con los resultados de la prueba a través de los siguientes campos:

- **Etiqueta:** Nombre de la etiqueta de muestra.
- **# Muestras:** El número de muestras con la misma etiqueta.
- **Media:** El tiempo promedio de un conjunto de resultados.
- **Mediana:** La mediana es el tiempo en el medio de un conjunto de resultados.
- **90% Línea:** 90% de las muestras no tardó más de este tiempo.
- **Min:** El tiempo más corto para las muestras con la misma etiqueta.
- **Max:** El tiempo más largo para las muestras con la misma etiqueta.
- **Error%:** Porcentaje de solicitudes con errores.
- **Rendimiento:** El rendimiento se mide en solicitudes por segundo / minuto / hora.
- **Kb / s:** El rendimiento se mide en kilobytes por segundo.

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Media de Bytes
Petición JDBC	10000	99	1	824	85,51	0,00%	625,9/sec	242,05	396,0
Total	10000	99	1	824	85,51	0,00%	625,9/sec	242,05	396,0

Figura 33. Reporte resumen correspondiente al grupo de hilos 1.

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Media de Bytes
Petición JDBC	15000	132	1	1617	95,58	0,00%	691,2/sec	267,32	396,0
Total	15000	132	1	1617	95,58	0,00%	691,2/sec	267,32	396,0

Figura 34. Reporte resumen correspondiente al grupo de hilos 2.

Gráfico de rendimiento de transacciones contra tiempo: Muestra el comportamiento de la carga de peticiones y su impacto directo en el rendimiento del sistema.

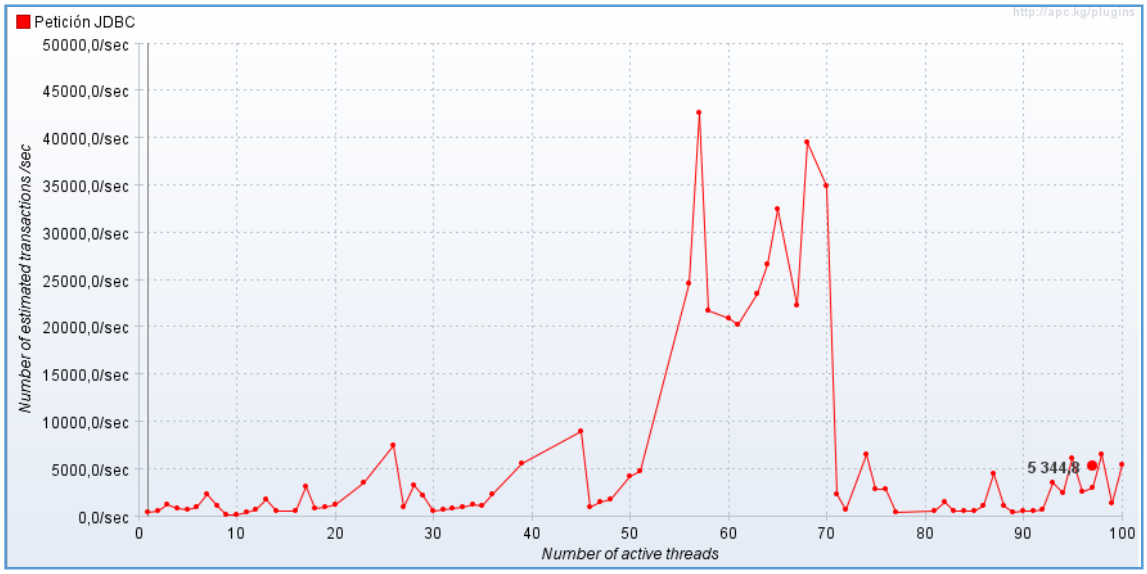


Figura 35. Gráfico de resultados correspondiente al grupo de hilos 1.

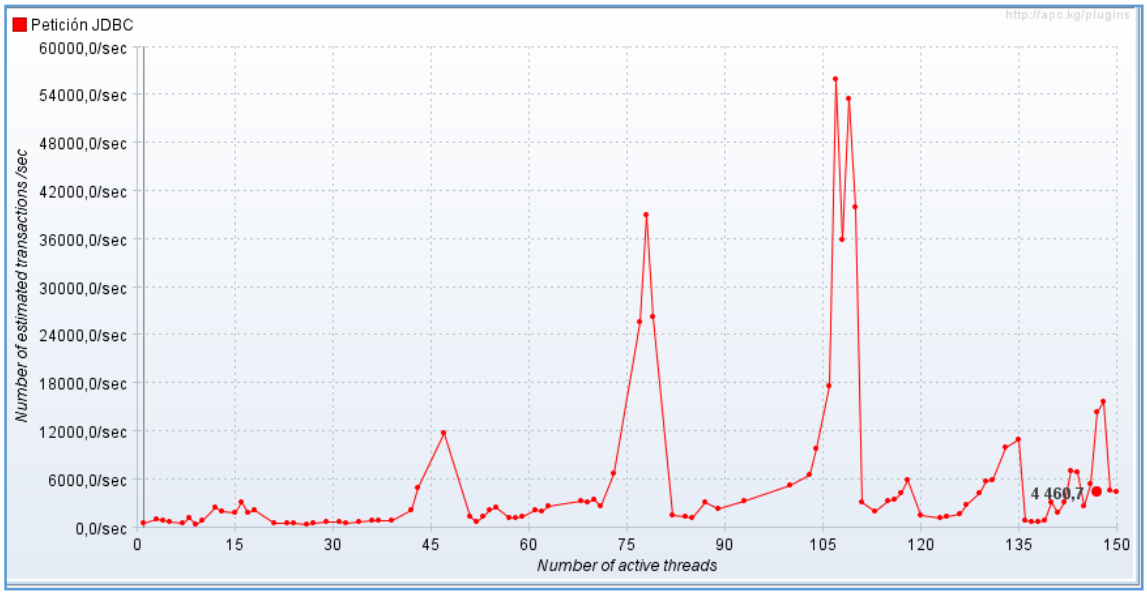


Figura 36. Gráfico de resultados correspondiente al grupo de hilos 2.

Los resultado arrojados por los receptores demuestran que la base de datos está en condiciones de atender como promedio una petición en poco menos de un segundo (0.099 y 0.132 respectivamente) para ambos grupos de hilos. Además en casos críticos la petición que más se tardaría en atender no superaría los dos segundos, tiempos que a su vez son catalogados como

aceptables, si se toma como referencia los tiempos de respuesta definidos en la Arquitectura Base del sistema (Ver Tabla 10).

Tipo de respuesta	Tiempo
Aceptable	Menor o igual a los 3 segundos
Superior a lo esperado	Mayor a los 3 segundos

***Tabla 10. Tiempo de respuesta a peticiones* (Acosta, 2010).**

3.4. Integridad y consistencia de los datos

Según (Instituto Tecnológico de Informática, 2014) obtener un modelo encaminado a las necesidades del usuario final por sí solo no asegura un correcto funcionamiento de la base de datos. Por lo que se hace necesario garantizar aspectos como la consistencia y la integridad. Es por ello que el uso de técnicas y reglas que regulen el acceso a los datos se convierte en otro de los procesos medulares en el desarrollo de la base de datos. En la propuesta de base de datos la integridad y consistencia de los datos se protegen por medio de restricciones sobre los valores que pueden tomar los elementos de los datos.

Restricciones de clave: Las llaves primarias (*Primary Key*) de cada entidad son atributos no nulos y únicos.

Integridad referencial: Las llaves foráneas (*Foreing Key*) agregadas en una tabla tomaran valores referentes y en concordancia con las entidades de donde se derivan.

Restricciones de valores nulos: Se definieron atributos no nulos, con la cláusula *not null*, en las tablas por la importancia que contienen a la hora de agregar datos.

Restricciones de dominio: Se definieron restricciones sobre los tipos de datos utilizados en las columnas, para chequear que los valores agregados cumplan con un dominio.

Dominio	Tipo de dato	Ejemplos de campos
descripción	*varchar(255)*	descripcion, referencia
comentario	*Texto*	motivos, notificacion, fundamentacion,

id_tabla	*numeric(19)*	id_personeria, id_otro, id_depuracion
id_nomenclador	*numeric(10)*	id_direccion, id_prueba
fecha	*Date*	fecha_personeria, fecha_notificacion
booleano	*Bool*	activo, inanicion_voluntaria

Tabla 11. Descripción de las restricciones de dominio.

3.5. Seguridad de la base de datos

Con el objetivo de prevenir accesos no deseados a la base de datos, el SGBD fue configurado para validar la autenticación y autorización tanto de los programas como de los usuarios que intentan acceder al mismo. Como parte de esta configuración se definió un esquema de contraseñas que soporta la robustez del mecanismo de autenticación. El mismo establece que: la longitud de la contraseña debe ser superior a los ocho caracteres, se debe hacer uso de combinaciones de letras mayúsculas y minúsculas, así como números y caracteres que no sean alfanuméricos.

Para cubrir las necesidades de control de usuarios sobre el servidor de base de datos, se utilizó el patrón RBAC (*Control de Acceso Basado en Roles*), que implementa el Gestor de Base de Datos. En este los permisos para realizar ciertas operaciones son asignados a roles específicos. Los usuarios son asignados a roles particulares, a través de los cuales adquieren permisos para realizar funciones particulares (Lopez, 2007).

Para el trabajo con la base de datos del proyecto SIGEF II en el servidor PostgreSQL se crearon los siguientes roles:

Administrador (adminbd): Tiene permisos para administrar las bases de datos en su totalidad, dígase crear nuevas tablas, eliminar otras ya existentes, asignar permisos, etc.

Usuario (usuariofiscalia): Contiene los privilegios necesarios para gestionar las tablas ya existentes en la base de datos, en concordancia con los permisos asignados a este, dígase actualización, inserción y eliminación de datos.

Con el objetivo de limitar el número de host que se pueden conectar al servidor se realizan los siguientes cambios en el fichero **pg_hba.conf** del gestor:

- Se permitirá el acceso a la base de datos solamente a los usuarios adminbd, usuariofiscalia y postgres desde la subred de la fiscalía.
- El método de autenticación empleado será md5.

Además la conexión entre el cliente y el servidor se realizará utilizando el protocolo SSL cifrando de esta manera los datos que viajan por dicha conexión. Para ello es necesario cambiar el valor del parámetro ssl en el fichero **postgresql.conf**.

3.6. Conclusiones parciales

Una vez realizadas las validaciones teóricas y funcionales al modelo se puede asegurar que el mismo cumple con los objetivo trazados al inicio de la investigación, logrando que la base de datos este en consecuencia con las métricas y restricciones de integridad, las cuales garantizan la consistencia y calidad de los datos. Además se pudo comprobar que los tiempos de respuesta respecto a la carga y consulta de datos son los requeridos por el proyecto.

Conclusiones Generales

En el presente trabajo se detalló la solución propuesta para el diseño e implementación de la base de datos de los módulos Administrativo y Revisión de Causas Penales del Sistema de Informatización de la Gestión de las Fiscalías. Una vez finalizada la investigación se concluye que:

- El estudio de los fundamentos teóricos permitió sentar las bases para un correcto diseño e implementación de base de datos.
- Del diseño realizado se obtuvo como artefacto principal los modelos lógico y físico de la base de datos.
- Las estrategias de optimización aplicadas, contribuyeron al mejoramiento de la solución y la eliminación de problemas de diseño.
- Las pruebas de rendimiento realizadas, corroboraran que la base de datos está en condiciones de soportar los volúmenes de datos y carga propios del entorno real de explotación.

Recomendaciones

Una vez satisfechas las expectativas que se tenían respecto al presente trabajo se proponen las siguientes recomendaciones:

- Continuar utilizando la estrategia de trabajo utilizada para el diseño e implementación de la propuesta de base de datos en los restantes módulos del proyecto SIGEF II.
- Una vez desplegada y operativa la base de datos, realizar un trabajo regular de soporte y mantenimiento.

Bibliografía

Acosta, Jose Carlos Pupo. 2010. *Arquitectura_Base_SIGEF II.* 2010.

Aguilar, Carlos Proal. 2012. Interactive and Coorporative Technologies Lab. [En línea] Departamento de Ingeniería de Sistemas Informáticos Universidad de las Américas Puebla, 2012. [Citado el: 15 de diciembre de 2013.] http://ict.udlap.mx/people/carlos/.

AIU. 2014. Open AIU Courses. [En línea] Atlantic International University, 2014. [Citado el: 14 de mayo de 2014.] https://cursos.aiu.edu/.

Baizán, Covadonga Fernández. 1987. *El modelo racional de datos: De los fundamentos a los modelos deductivos.* s.l. : Ediciones Díaz de Santos, 1987.

Batini, Carlos. 2004. *Diseño conceptual de bases de datos.* 2004.

Blaha, Michael. 2010. *Patterns of Data Modeling.* s.l. : CRC Press, 2010.

Blanco, Héctor Fuentes. 2012. *Documento Arquitectura de Software SIGEF II.* 2012.

Collins-Sussman, Ben, W. Fitzpatrick, Brian y Pilato, C. Michael . 2004. *Control de versiones con Subversion.* s.l. : O'Reilly Media, 2004.

Comunidad de PostgreSQL-UCI. 2014. PostgreSQL:Comunidad Técnica de Desarrollo. [En línea] 2014. [Citado el: 14 de marzo de 2014.] https://postgresql.uci.cu/?page_id=30.

Cuello, Félix Rafael Borges. 2013. *Diseño e implementación de la base de datos para el Sistema de Gestión de Ferias y Eventos de la Cámara de Comercio de la República de Cuba.* La Habana : Universidad de las Ciencias Informaticas, 2013.

Cuesta, Ernesto Alejandro Figueredo. 2011. *Diseño de la base de datos del procedimiento Ordinario de la materia Penal del proyecto Tribunales Populares Cubanos.* La Habano : Universidad de las Ciencias Informaticas, 2011.

Datanamic Solutions BV. 2014. Datanamic. [En línea] 2014. [Citado el: 4 de abril de 2014.] http://www.datanamic.com/datagenerator-for-postgresql/index.html.

Doctrine Team. 2006-2014. Doctrine. [En línea] 2006-2014. [Citado el: 20 de marzo de 2014.] http://www.doctrine-project.org/projects.html.

Elsmari, Navathe R. 1993. *Sistemas de Bases de Datos, Conceptos fundamentales.* s.l. : Addison-Wesley, 1993.

Gray, R.H.M., y otros. 1991. *Design metrics for database systems.* s.l. : ROYAUME-UNI, 1991.

Grueso, Cesar David Fernandez. 2009-2010. slideshare. [En línea] 2009-2010. [Citado el: 12 de noviembre de 2013.] http://www.slideshare.net/senaticscesar/bases-de-datos-conceptos-basicos#.

Gudiño, Pedro Fletes, Hernández Barbosa , Oscar Daniel y Benavides Delgado, J. Reyes. 2007. Instituto Tecnológico de Colima, Departamento de Sistemas de Computación, Tutorial de Fundamentos de bases de datos. [En línea] 2007. [Citado el: 15 de diciembre de 2013.] http://labredes.itcolima.edu.mx/fundamentosbd/sd_u2_1.htm.

Hansen, James V. y Garry, W. 2006. *Diseño y administración de Bases de Datos. 2da edición.* s.l. : Prentice Hall , 2006.

IBM Corp. 2007. *Classic RUP for SOMA.es.* 2007.

IBM Corporation. 2013. *Rational Unified Process.* 2013.

Instituto Tecnológico de Informática. 2014. ITI:Instituto Tecnológico de Informática. [En línea] 2014. [Citado el: 2 de mayo de 2014.] http://www.iti.es/servicios/servicio/resource/7240/index.html.

Jmeter, Apache. 2013. The Apache Software Fundation. [En línea] 2013. [Citado el: 22 de enero de 2014.] http://jmeter.apache.org/.

Linux Six Blogspot. 2011. Linux Six Blog. [En línea] 2011. [Citado el: 5 de diciembre de 2013.] http://linuxsix.blogspot.com/2011/10/rapidsvn-svn-en-linux.html.

Lopez, Gracia Fernandez. 2007. *Seguridad en Sistemas Operativos.* 2007.

Marqués, Mercedes. 2011. *Bases de Datos.* s.l. : Publicacions de la Universitat Jaume I. Servei de Comunicació i Publicacions, 2011.

Mata, Manel Pérez. 2009. Programación, SEO e Internet. [En línea] 2009. [Citado el: 15 de marzo de 2014.] http://manelperez.com/programacion/que-es-doctrine-orm/.

Mato, Rosa Maria. 2005. *Sistemas de Base de Datos.* La Habana : Pueblo y Educación, 2005.

Netbeans. 2013. Netbeans. [En línea] 2013. [Citado el: 12 de enero de 2014.] https://netbeans.org/community/releases/73/.

Oracle. 2011. Guía de administración del sistema: servicios de seguridad . [En línea] 2011. [Citado el: 22 de enero de 2014.] http://docs.oracle.com/cd/E24842_01/html/E23286/rbac-1.html#scrolltoc.

Ortíz Noriega, Dresky y López Boullón, Javier . 2010. *Módulo de Reko para la resolución de conflictos.* s.l. : Universidad de las Ciencias Informáticas, 2010.

Osorio, Alain y Lopez, Mairelys. 2013. *Guía práctica para Arquitecturas de Datos Empresariales.* s.l. : Latin American and Caribbean Conference for Engineering and Technology, 2013.

Peña, Angela Gloria Gomez y Alfonso Valdé, Javier . 2010. *Desarrollo del módulo para la transmisión de datos de gran tamaño para el sistema Reko.* s.l. : Universidad de las Ciencias Informáticas, 2010.

Pérez, M., Valero, E.,Zavala, M. 2011. *Base de Datos Ingeniería de Sistemas.* s.l. : Universidad politécnica de la fuerza Armada Bolivariana, 2011.

Ponniah, Paulraj. 2007. *Data Modeling Fundamentals.* 2007.

Pressman. 2005. *Aprendiendo UML en 24 horas.* 2005.

Pupo, Jose Carlos. 2010. 0120_4 Arquitectura Vista de Datos. s.l. : Universidad de las Ciencias Informáticas, 2010.

Quiroz, Javier. 2003. El modelo relacional de bases de datos. *Boletín de Política Informática.* 2003. 6.

SENA-Servicio Nacional de Aprendizaje. *Manual Apache Jmeter.*

Silberschatz, Abraham, F. Korth, Henry y Sudarshan, S. 2002. *FUNDAMENTOS DE BASES DE DATOS.* Madrid : McGraw-Hill Inc, 2002.

The Apache Software Foundation. 1997-2014. Apache HTTP Server Project. [En línea] 1997-2014. [Citado el: 10 de abril de 2014.] http://httpd.apache.org/.

Tumbarell, Abel Andres Irsula. 2013. *Diseño e implementación de la base de datos de los módulos Queja, Peticiones y Denuncias, y Comunes.* La Hanana : Universidad de las Ciencias Informáticas, 2013.

Ubuntu. 2008. Portal-Guía Ubuntu. [En línea] 2008. [Citado el: 9 de diciembre de 2013.] http://www.guia-ubuntu.com/index.php?title=PgAdmin_III..

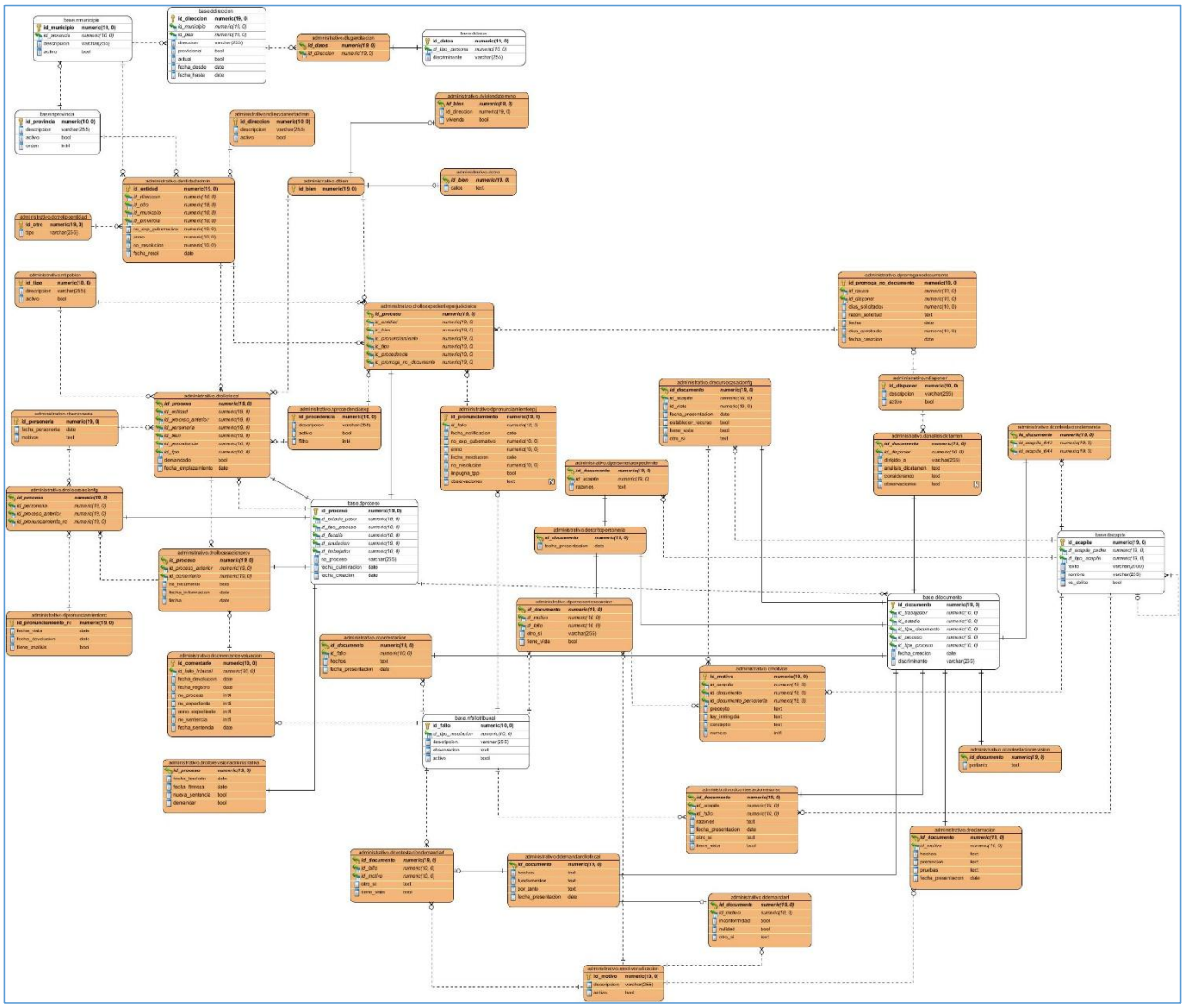
UCI. 2012. Portal de la Universidad de las Ciencias Informáticas. [En línea] 2012. [Citado el: 12 de noviembre de 2013.] www.uci.cu.

Visual Paradigm. 2013. Visual Paradigm. [En línea] 2013. http://www.visual-paradigm.com/.

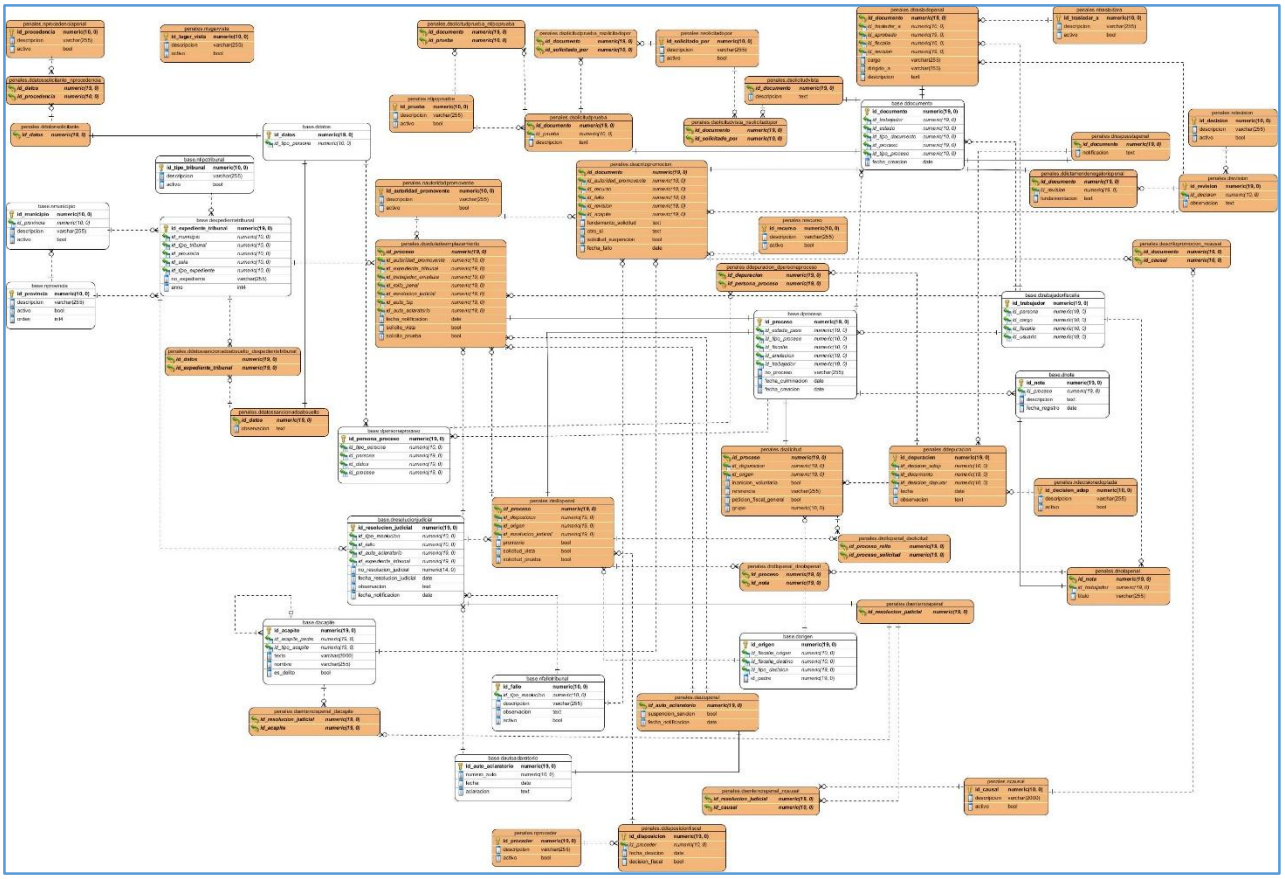
Zambrano, Raquel. 2008. *Sistemas Gestores de Bases de Datos.* Cordoba : s.n., 2008.

Anexos

Anexo 1. Modelo lógico del módulo Administrativo.



Anexo 2. Modelo lógico del módulo Revisión de Causa Penales.



Anexo 3. Requisitos de acceso a datos de los módulos Administrativo de Revisión de Causas Penales.

Requisitos	Descripción
Administrativo	
Buscar expediente	Permite buscar los Expedientes Prejudiciales adicionados al sistema.
Listar expedientes	Permite listar los Expedientes Prejudiciales que se hayan adicionado al sistema, según el paso a realizar.
Listar la procedencia	Lista la procedencia de los Expedientes Prejudiciales.
Listar las prorrogas	Lista las prórrogas en tiempo asociadas a un Expediente Prejudicial.
Listar Rollos Judiciales	Lista los Rollos Judiciales que se hayan adicionado al sistema, según el paso a realizar.
Listar Rollos de Casación FP	Lista los Rollos de Casación que se hayan adicionado al sistema en las fiscalías provinciales, según el paso a realizar.
Listar motivos	Muestra la lista de los motivos asociados al recurso de casación.
Listar Rollos de Casación FGR	Muestra la lista de los Rollos de Casación que se hayan adicionado al sistema, según el paso a realizar.
Buscar Rollos de Revisiones Administrativas	Permite realizar la búsqueda de los rollos adicionados al sistema.
Listar Rollos de Revisiones Administrativas	Permite listar los Rollos de Revisiones Administrativas que se hayan adicionado al sistema, según el paso a realizar.
Listar pronunciamientos	Lista los pronunciamientos asociados a Expedientes Prejudiciales a partir del número de expediente.
Revisión de Causas Penales	
Listar solicitudes	Permite listar las solicitudes que se hayan adicionado al sistema dado su origen.
Listar solicitudes con respuesta	Permite listar las solicitudes que han recibido respuesta por una Fiscalía determinada.
Listar rollos para revisión	Permite listar los rollos que se encuentran listos para su revisión.

Listar causas	Permite listar las causas adicionas al sistema.
Listar notas	Muestra una lista de las notas de un rollo.
Listar respuestas	Permite listar las respuestas.
Listar Rollos Penales	Lista los Rollos Penales que hayan sido registrados en el sistema.
Listar cédula	Permite listar las Cédulas de Emplazamiento.
Buscar cédula	Permite buscar las Cédulas de Emplazamiento.