Universidad de las Ciencias Informáticas

Facultad 3

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

Título: "Propuesta de optimización de la base de datos para el proyecto Sistema de Informatización de la Gestión de las Fiscalías"

Autor(es): Irael Pérez González
Madelaine Sosa Díaz
Tutor(es): Carlos Felipe Pérez León
Co-tutor: Vlamir Rodríguez Fernández

Junio, 2011

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo: "Propuesta de optimización de la base de datos para el proyecto Sistema de Informatización de la Gestión de las Fiscalías" y autorizamos a la facultad 3 y en especial al proyecto SIGEF de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

_____ _____

Irael Pérez González — Autor

Madelaine Sosa Díaz — Autor

_____

MsC Carlos Felipe Pérez León

Tutor

RESUMEN

Las innovaciones tecnológicas y particularmente la Informática, cumplen un papel primordial en los tiempos de la llamada "Sociedad de la Información". Hoy en día, las bases de datos han ido desplazado a otras formas de almacenamiento de información. Es predecible suponer que una vez completado el proceso de digitalización de las informaciones, las mismas constituirán la principal fuente de provisión de datos en uso práctico por la humanidad. De ahí proviene la importancia del correcto manejo y funcionamiento que deben tener dentro de las aplicaciones.

Actualmente el proyecto "Sistema de Informatización de la Gestión de las Fiscalías" (SIGEF) cuenta con una base de datos para almacenar la información referente a los procesos que se representan en las fiscalías, la cual no contribuye con un rendimiento eficiente de la aplicación debido a que presenta incoherencias y errores en el diseño del modelo de datos que provoca la existencia de redundancias e inconsistencias en la información recopilada; además, en su desarrollo no se aprovechan las potencialidades del gestor.

Por lo antes expuesto, el propósito del presente trabajo se centra en brindar una propuesta de optimización para la base de datos del proyecto SIGEF y en el cumplimiento del mismo se realizó un profundo estudio de las temáticas referente a las técnicas y patrones de diseño y la optimización de las bases de datos. Se brinda una propuesta de solución factible para la segunda fase del proyecto. Las validaciones realizadas fueron satisfactorias. Finalmente se logró realizar una propuesta que cumple con los objetivos definidos.

PALABRAS CLAVE

Optimización, Base de Datos relacionales, Diseño

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

INTRODUCCIÓN

En los últimos años las tecnologías informáticas se han desarrollado de un modo vertiginoso, influyendo aún más en el nivel de vida del hombre. Las Tecnologías de la Información y las Comunicaciones (TICs) se han vuelto parte imprescindible en la vida cotidiana de las diferentes sociedades. Cuba, reconociendo la trascendencia de estas transformaciones y haciendo un extraordinario esfuerzo, se ha propuesto aumentar su desarrollo en pos de mejorar el nivel de vida de sus ciudadanos en el ámbito económico, social y cultural.

Al calor de este trascendental esfuerzo nacieron un significativo número de aplicaciones o sistemas informáticos encargados de automatizar las actividades de los individuos en las diferentes esferas de la vida. Este conjunto de programas tiene múltiples objetivos, sin embargo, muchos de ellos requieren de la necesidad de hacer persistir determinados datos a lo largo del tiempo para posteriormente ser consultados; permitiendo así el avance en procesos como: la obtención de reportes, el apoyo a otros sistemas, la toma de decisiones o, sencillamente, el resguardo de datos históricos. Los software con estas funcionalidades pueden ser posibles gracias a las tecnologías de almacenamiento existentes en las ciencias de la computación, que van desde los simples ficheros o los conocidos XML (cuando no es necesario almacenar grandes volúmenes de datos) hasta complejas y potentes bases de datos (BD).

Por las razones antes mencionadas se puede advertir y comprender la necesidad, el impacto y la trascendencia del programa de Informatización de la Sociedad Cubana que se ha venido consolidando en estos últimos años y en los cuales ha participado activamente la Universidad de las Ciencias Informáticas (UCI).

Desde su surgimiento, la UCI ha realizado convenios con diferentes empresas e instituciones del país, uno de ellos es el proyecto SIGEF, surgido a raíz del convenio establecido con la Fiscalía General de la República de Cuba (FGR), institución que presenta algunas dificultades en la correcta ejecución de sus procesos. SIGEF consiste en una aplicación web, que tiene como objetivo principal informatizar los procesos que se desarrollan en la Fiscalía General de la República, Fiscalías Provinciales y/o Fiscalías Municipales, en pos de elevar la calidad en la planificación, organización, ejecución y control de los mismos, lograr mayor eficacia y eficiencia en el acceso a la información y contribuir al complejo proceso de toma de decisiones.

La FGR, según establece la Constitución de la República de Cuba en su artículo 127 capítulo XIII, es el órgano del Estado al que corresponde (como objetivos fundamentales) 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 por los organismos del Estado, entidades económicas y sociales, así como por los ciudadanos; al igual que la promoción y el ejercicio de la acción penal pública en representación del Estado. Las fiscalías de Cuba participan activamente en un importante conjunto de procesos judiciales, estos son ventilados ante los tribunales y pueden ser: penales, civiles, administrativos, laborales, económicos, entre otros.

SIGEF I está compuesto por 5 subsistemas que incluyen a su vez 13 módulos correspondientes a los procesos que representan en la FGR, entre los que se encuentran:

- ✓ **Procesos Penales (PP)**: Sumario, Ordinario e Índice de Peligrosidad Pre delictiva.
- ✓ **Gestión de Cuadros y Personal de Apoyo (GCPA):** Gestión de Cuadros, Gestión del Personal de Apoyo, Capacitación, Reportes y Búsquedas, Generalidades y Capacitación de Plantilla.
- ✓ **Verificación Fiscal (VF):** Procesos confiscatorios por LD 149.[1]
- ✓ **Herramientas Comunes a Todas las Áreas (HCTA):** Diccionario Jurídico y Gestión de Entrada y Salida de Documentación.
- ✓ **Control de la Legalidad en Establecimientos Penitenciarios (CLEP):** Inspección a locales de Detección y Centro Penitenciarios.

La información necesaria recogida en cada uno de estos módulos es almacenada en una base de datos relacional que no cuenta con toda su respectiva documentación pues la existente es insuficiente con respecto a la que se debería tener. El diseño conceptual, lógico y físico de la base de datos no han sido desarrollados por un especialista con experiencia en esta área de conocimiento. Los mismos carecen de uniformidad y no se han modificado eficientemente ante los cambios de los requisitos y reglas del negocio. El diseño de la base de datos está más enfocado y condicionado a los conocimientos y habilidades de los desarrolladores que a la necesidad de la elaboración de un diseño optimizado que tributase eficientemente a la aplicación, en consecuencia, se evidencia que el manejo de temas tan sensibles como: minimización del espacio de almacenamiento, disminución de los tiempos de respuesta y optimización del consumo de recursos, no han constituido prioridades durante el proceso de diseño de la base de datos. Además, se debe mencionar que el empleo de patrones de diseño de BD, es mínimo, se ha empleado de manera casual y no siempre de forma correcta**.**

[1] Decreto ley 149

Las razones previamente expuestas avalan la existencia de errores de diseño en el modelo de datos, lo que provoca redundancias e inconsistencias en la información y la necesidad de optimizar la base de datos del proyecto SIGEF Fase I para maximizar su rendimiento, potencialidades y beneficios. Teniendo presente la situación descrita, el **problema de la investigación** radica en**:** El Sistema de Informatización de la Gestión de las Fiscalías dispone de una base de datos que no contribuye con un rendimiento eficiente de la aplicación.

Por consiguiente el **objeto de estudio** de la investigación es: Diseño de Bases de Datos Relacionales. Como **objetivo general** de la misma se tiene: Realizar una propuesta de optimización de la base de datos del proyecto Sistema de Informatización de la Gestión de las Fiscalías mediante la implementación de técnicas y mecanismos que contribuyan a lograr un rendimiento eficiente de la aplicación. Centrando la atención en la Optimización de Bases de Datos Relacionales identificado como **Campo de acción,** lo que conlleva a la siguiente **Idea a defender:** Con la implementación de técnicas y mecanismos para la optimización de la base de datos del Sistema de Informatización de la Gestión de las Fiscalías se contribuirá a lograr un rendimiento eficiente de la aplicación. Para el correcto cumplimiento del objetivo trazado anteriormente, se desglosan los siguientes **objetivos específicos:**

1. Realizar un estudio del estado del arte referente a la optimización de bases de datos relacionales mediante el uso de los métodos definidos para la investigación, que permita elaborar la fundamentación teórica del presente trabajo.

2. Elaborar una propuesta de optimización de la base de datos del proyecto Sistema de Informatización de la Gestión de las Fiscalías a partir del estudio del fundamento teórico de la investigación realizada, que contribuya a lograr un mayor rendimiento de la aplicación.

3. Validar la propuesta de optimización mediante la realización de pruebas que evalúen el rendimiento de la base de datos del proyecto Sistema de Informatización de la Gestión de las Fiscalías.

Para alcanzar los objetivos propuestos se utilizarán un conjunto de métodos científicos, tales como:

Métodos Teóricos:

Analítico – sintético: Se utilizó este método pues permite dividir el fenómeno a estudiar y unir las partes previamente analizadas. Se analizaron los aspectos más importantes para optimizar la base de datos del

proyecto SIGEF I. Se tuvo en cuenta para esto, los requisitos del sistema actualizados y de ahí se sintetizó como realizar la propuesta planteada.

Análisis Histórico – lógico: Se empleó debido a que se hizo una profunda investigación para analizar la trayectoria completa del fenómeno, elaborando el estudio del estado del arte de este tema. Obteniendo una tendencia de cómo se comporta en la actualidad.

Métodos Empíricos:
Entrevista: Para darle cumplimiento a este método se realizaron entrevistas a especialistas con experiencias en el área de conocimiento de base de datos, con el objetivo de profundizar en las diferentes temáticas a tratar y captando ideas que sirvieron para el correcto encuadre de la investigación.

Estructura de la Tesis: El presente trabajo de diploma consta de 3 capítulos.

Capítulo 1. Fundamentación Teórica: Este capítulo recoge toda la fundamentación teórica que sustenta el desarrollo de la investigación para el posterior desarrollo de la propuesta de optimización. Se profundiza en temas relacionados con el trabajo de diploma tales como: conceptos, características y funciones asociados a los sistemas de gestión de bases de datos y a la optimización de su diseño.

Capítulo 2. Propuesta de Solución: En este capítulo se realiza un estudio de las características de la base de datos de SIGEF I, se propone la realización de un conjunto significativo de transformaciones necesarias para la optimización de la misma, identificadas durante el estudio de artefactos (actualizados) obtenidos en los flujos de trabajos: Requisitos, Análisis y Diseño del Sistema. Finalmente se presentan los módulos de la base de datos con las modificaciones aplicadas y el rendimiento esperado con la implementación de los índices propuestos.

Capítulo 3. Validación de la Propuesta: En este capítulo se muestran los resultados que permiten validar la propuesta desarrollada.

CAPÍTULO I: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

El fundamento teórico es una de las partes más importantes en el desarrollo de un trabajo de diploma pues provee de un marco de referencia para la adecuada interpretación los resultados que se recogen. En el presente capítulo se analizan y exponen un conjunto de enfoques teóricos e investigativos que han sido objeto de publicación en el área del conocimiento donde se ubica la investigación de este trabajo. Además se realiza un estudio de los diferentes conceptos, objetivos y características fundamentales asociados a los sistemas de gestión de bases de datos, a temas de patrones de diseño y técnicas optimización de los mismos y se ofrece una panorámica del gestor de bases de datos empleado.

1.2 Conceptos asociados al dominio del problema.

Para alcanzar una mejor comprensión y ampliación de las diferentes temáticas que se estarán desarrollando en el presente trabajo de diploma, se mostrarán un grupo de términos identificados durante la investigación realizada que corresponden a los Sistemas de Bases de Datos (SBD) y a la optimización de las mismas.

1.2.1 Sistema de Gestión de Base de Datos

En la actualidad el desarrollo informático ha propiciado que las instituciones basen su desempeño en el uso de aplicaciones informáticas para la gestión de la información, toda la información que es generada por estas instituciones se guardan en bases de datos. El enfoque de estas es extensamente utilizado por ser la mejor y, al mismo tiempo, la más empleada solución para el correcto manejo de grandes volúmenes de información, que tiene en cuenta la concurrencia y la complejidad de la extracción de los mismos. Las BD se han extendido por la disminución de los costos de los servidores y las necesidades de exploración de datos. Para introducir al tema de los Sistemas de Gestión de Base de Datos lo primero es comenzar por algunos conceptos o definiciones esgrimidos sobre ellos. Muchos autores, han planteado diferentes definiciones acerca de estas, sin embargo, todos siguen la misma línea o idea, por ejemplo, se dice que es: "*un sistema computarizado cuya finalidad general es almacenar información y permitir a los usuarios actualizar y recuperar esa información con base en peticiones*". (1)

Otros autores como Yunta L.R se refiere a una base de datos como "*un conjunto de información estructurada en registros y almacenada en un soporte electrónico legible desde un ordenador, donde cada registro constituye una unidad autónoma de información que puede estar a su vez estructurada en diferentes campos o tipos de datos que se recogen en dicha base de datos*". (2)

Como otra definición, se dice que es "*un conjunto de datos almacenados entre los que existen relaciones lógicas y que han sido diseñadas para satisfacer los requerimientos de información de una empresa u organización*". (3)

En el caso de Santiago de Cali propone que una base de datos es "*una entidad en la cual se pueden almacenar datos de manera estructurada, con la menor redundancia posible. Es la recopilación de datos en forma ordenada y concreta*". (4)

También se expone que es *"un conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquinas accesibles en tiempo real*". (5)

Para definir, crear y mantener las bases de datos, así como proporcionar y controlar el acceso a estas, son usados los **Sistemas de Gestión de Bases de Datos (SGBD)**. Se puede afirmar que el SGBD es un software que facilita el proceso de definición, construcción y manipulación de la base de datos de diversas aplicaciones. Pueden ser de propósito general o específico, dedicado a servir de interfaz entre la BD, el usuario y las aplicaciones que la utilizan. Se componen de tres lenguajes, uno para la definición de datos, otro para la manipulación de los mismos y un tercero de consultas. (6). Por otra parte, Edgar Frank Codd, científico informático creador del modelo relacional, ha establecido una lista con los ocho servicios que debe ofrecer todo SGBD. (3)

De todos los conceptos que se analizaron respecto a los sistemas de base de datos y las base de datos se ha llegado a la conclusión, que estas son un conjunto de datos almacenados, relacionados entre sí y que tienen un significado implícito. De igual manera se está de acuerdo con la definición propuesta por Edgar Codd . "*un conjunto de datos almacenados entre los que existen relaciones lógicas y que han sido diseñadas para satisfacer los requerimientos de información de una empresa u organización",* también es válido mencionar que para definir, crear y mantener las bases de datos, así como proporcionar y controlar el acceso a estas se emplean los SGBD.

Existen distintos objetivos que deben cumplir los SGBD:

Abstracción de la información. No es imprescindible para el usuario que una base de datos ocupe uno o cientos de archivos, debido a que los SGDB les ahorran detalles acerca del almacenamiento físico de los datos. Así se definen varios niveles de abstracción.

Independencia. Este permite que se modifique el esquema físico o lógico de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.

Consistencia. En aquellos casos en los que no se ha logrado eliminar la redundancia, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea. Por otra parte, la base de datos representa una realidad definida por determinadas condiciones que pueden ser verificadas por herramientas existentes en los SGBD.

Seguridad. Los SGBD deben garantizar que la información almacenada en una base de datos se encuentre segura y disponible según los permisos de usuarios o grupos de usuarios definidos.

Manejo de transacciones. Una transacción es un programa que se ejecuta como una sola operación. Esto significa que su ejecución se completa solo cuando no se producen fallas en la misma. Los SGBD proveen mecanismos para programar las modificaciones de los datos de una forma mucho más simple que si no se dispusiera de ellos.

Tiempo de respuesta. Permite establecer el tiempo que el SGBD demora en proporcionar la información solicitada y en almacenar los cambios realizados.

1.2.2 Sistemas de Gestión de Bases de Datos Relacionales (SGBDR)

En una computadora existen diferentes formas de almacenar información. Esto da lugar a distintos modelos de organización de la base de datos: jerárquico, red, relacional y orientada a objeto. Los sistemas relacionales son uno del más importante debido a su amplio y diversificado empleo así como las potencialidades que ofrece como por ejemplo: simplicidad y generalidad, facilidad de uso para el usuario final, períodos cortos de aprendizaje y las consultas de información se especifican de forma sencilla.

El SGBD más extendido y de mayor uso en la actualidad es el Sistema de Gestión de Base de Datos Relacional (SGBDR), como su nombre lo indica este sistema se encarga de administrar las BD relacionales, los motivos de este éxito son fundamentalmente dos:

1. Ofrecen sistemas simples y eficaces para representar y manipular los datos.
2. Se basan en un modelo (el relacional) con sólidas bases teóricas.

Los SGBDR permiten realizar búsquedas utilizando los datos de columnas especificadas de una tabla para encontrar datos adicionales en otra tabla, también proveen herramientas para evitar la duplicidad de registros, garantizan que al eliminar un registro sean eliminados todos los registros relacionados dependientes y además representan un modelo más comprensible y aplicable. Los SGBDR presentan deficiencias en cuanto al almacenamiento de datos gráficos, y puede dificultar el desarrollo de ciertas

aplicaciones que manejan datos de tipos multimedia como audio y video. Por todos estos planteamientos se requiere de su uso en el presente trabajo de diploma.

Varios autores como Hernández y G.W. Hansen han expresado lo ventajoso que resulta ser este modelo explicando, entre otras ventajas, que provee herramientas que evitan la duplicidad de registros, garantizan la integridad referencial, y así al eliminar un registro elimina todos los registros relacionados dependientes y favorece la normalización por ser más comprensible y aplicable. (8), (9)

1.2.3 Optimización de base de datos

La optimización es un aspecto de suma importancia en el desarrollo de un sistema de base de datos que permite mejorar su rendimiento, eficiencia y funcionalidades en las condiciones más adecuadas de acuerdo a sus características y propósitos. La optimización de una base de datos depende esencialmente de un correcto diseño inicial, tanto lógico como físico. El objetivo de la optimización consiste en minimizar el tiempo de respuesta para cada petición y maximizar el rendimiento de todo el sistema disminuyendo el tráfico de red, el acceso a disco y el tiempo de CPU[2].

Los planteamientos sobre la optimización deben ser propuestos durante todo el ciclo de desarrollo, y no sólo una vez que el sistema ya está implementado. Esta es la mejor forma de lograr el afinamiento de las bases de datos.

Los afinamientos se pueden clasificar en: (10)

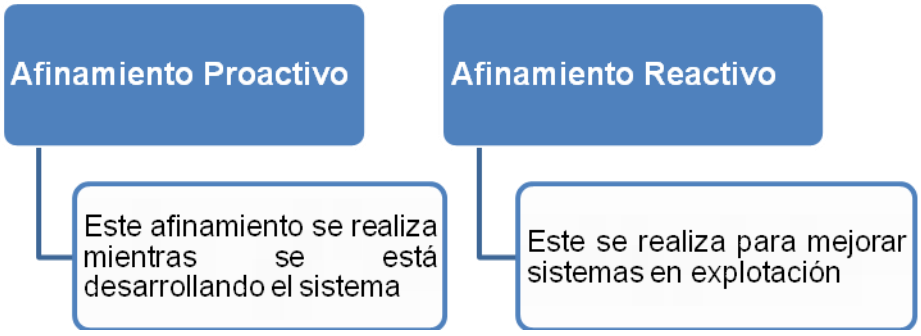

Figura 1: Tipos de Afinamientos

[2] CPU: Unidad Central de Procesamiento (Central Process Unit)

Es importante destacar que de estas dos clasificaciones de afinamientos de las bases de datos, el más efectivo es el proactivo pues (abordado con mayor profundidad y desde etapas muy tempranas) repercute extraordinariamente en el trabajo de arquitectos y diseñadores del sistema, responsables (entre otras actividades) de aplicar técnicas para mejorar el rendimiento, definir la configuración de hardware más apropiada para responder y cumplir con los requisitos y expectativas durante el análisis y diseño de la aplicación.

Aunque el afinamiento proactivo sea más eficiente y menos costoso que el reactivo, esto no implica que se pueda prescindir de este último debido a que un sistema bien diseñado para alcanzar un rendimiento óptimo cuando está en explotación está expuesto a otros riesgos que pueden identificarse y evitarse mediante el afinamiento reactivo. Es posible mejorar un sistema en explotación reactivamente comenzando por el paso final e ir subiendo poco a poco a los otros, encontrando y arreglando la mayor cantidad posible de cuellos de botella.[3]

Por otra parte es importante conocer que el problema que pueda provocar un mal rendimiento puede ocurrir después de iniciar el sistema, por ello, para comenzar, el rendimiento debe ser monitoreado y evaluado regularmente después de comenzar la operación.

El proceso de optimización no debe comenzar una vez que los usuarios de un sistema plantean quejas e inconformidades sobre el rendimiento y los tiempos de respuesta del mismo, cuando esto sucede usualmente es demasiado tarde para implementar algunas de las estrategias de optimización más efectivas. En este caso, si no es posible rediseñar completamente la base de datos, entonces lo único que se puede hacer para mejorar el rendimiento del sistema es reasignar más memoria y mejorar los parámetros de entrada y salida de disco.

Existen algunos **tipos de optimización de consultas** como son (11):

Optimización basada en reglas

- ✓ Heurísticas generales basadas en la experiencia del administrador o de los diseñadores, en la que el programador puede especificar reglas que guíen al optimizador en su trabajo.

Optimización basada en costos

Estimación de los costos de realizar una operación física. Depende:

[3] Cuellos de botella en un proceso proactivo, es una fase de la cadena de producción más lenta que otras, que ralentiza el proceso de producción global.

- ✓ Tamaño de la estructura: estadísticas
- ✓ Tamaño de la memoria: depende de los procesos que se ejecuten simultáneamente

Los optimizadores basados en costos asignan un costo (que intenta estimar el costo de la consulta en términos de operaciones de entrada-salida requeridas, requerimientos de CPU y otros factores) a cada uno de esos planes y selecciona el de menor valor. El conjunto de planes de ejecución se forma examinando los posibles caminos de acceso (mediante índices o secuenciales), algoritmos de "join" (sort-merge join, hash join, bucles anidados). Este optimizador no puede ser accedido directamente por los usuarios, sino que, una vez enviadas las consultas al servidor, pasan primero por el analizador y entonces llegan al optimizador.

1.3 Gestor de Base de datos PostgreSQL V 8.4

En la actualidad existen diversos gestores de bases de datos, dentro de los más relevantes se pueden encontrar: Oracle, Microsoft SQL Server, MySQL, y PostgreSQL. Este último es el seleccionado para el desarrollo del presenta trabajo en pos de estar en total correspondencia con el gestor de bases de datos definido por el proyecto SIGEF I y formando parte de la estratégica tendencia nacional de migración del software propietario hacia el software libre.

PostgreSQL es un potente sistema gestor de bases de datos relacionales orientadas a objetos. Es considerado como el gestor más avanzado de código abierto de los que se disponen en la actualidad, soporta gran conjunto de funcionalidades, lo que sitúa en una posición de vanguardia por delante de algunos SGBD comerciales. Su código fuente está disponible y liberado bajo licencia BSD (la más liberal de las licencias del open source). Tiene más de 15 años de activo desarrollo y arquitectura probada que se ha ganado una muy buena reputación por su confiabilidad e integridad de datos. Funciona en todos los sistemas operativos importantes. Cuenta también con un amplio conjunto de enlaces con lenguajes de programación (incluyendo C, C++, Java, perl, tcl y python). Entre sus principales características se encuentran:

- ✓ Soporta casi toda la sintaxis SQL (incluyendo subconsultas, transacciones, tipos y funciones definidas por el usuario) pues tiene soporte total para llaves extranjeras, *joins*, vistas, reglas, índices, disparadores y procedimientos almacenados (en múltiples lenguajes).

- ✓ Acceso concurrente multiversión, MVCC Control de Concurrencia Multi-Versión (Multi-Version Concurrency Control), es una técnica avanzada para mejorar las prestaciones de una base de datos en un entorno multiusuario, no se bloquean las tablas, ni siquiera las filas cuando un proceso escribe. Es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. Mediante el uso

de MVCC, este gestor se evita el problema de que procesos lectores estén esperando a que se termine de escribir para poder leer. En su lugar, el gestor mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos, siendo capaz entonces de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los registros estén disponibles.

- ✓ Cliente/Servidor: Usa una arquitectura proceso-por-usuario cliente/servidor, la cual es similar al método del Apache 1.3.x para manejar procesos. Existe un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL.

- ✓ Write Ahead Logging (WAL): característica de PostgreSQL que incrementa la dependencia de la base de datos al registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos deje de funcionar, existirá un registro de las transacciones a partir del cual se podrá restaurar la base de datos, lo que representa un enorme beneficio pues cualquier cambio que no haya sido escrito en la base de datos cuando dejó de funcionar el servidor, puede ser recuperado usando el dato que previamente registrado.

- ✓ Lenguajes Procedurales: Tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido.

Herencia de tablas:

- ✓ Incluye la mayoría de los tipos de datos SQL92 y SQL99 (INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, y TIMESTAMP), soporta almacenamiento de objetos grandes binarios, además de tipos de datos y operaciones geométricas.

- ✓ Puntos de recuperación a un momento dado, tablespaces, replicación asincrónica, transacciones jerarquizadas (savepoints), Backups en línea.

- ✓ Un sofisticado analizador/optimizador de consultas.

- ✓ Soporta juegos de caracteres internacionales, codificación de caracteres multibyte.

Según lo expresado en bibliografías oficiales del gestor, con PostgreSQL se han aumentado y mejorado sus características y capacidades, aunque el trabajo continúa en todas las áreas incluyendo en sus principales mejoras, entre las que se encuentran:

- ✓ Los bloqueos de tabla han sido sustituidos por el control de concurrencia multi-versión, el cual permite a los accesos de sólo lectura continuar leyendo datos consistentes durante la actualización de registros, y permite copias de seguridad en caliente desde pg_dump mientras la base de datos permanece disponible para consultas.

- ✓ Se han implementado importantes características del motor de datos, incluyendo subconsultas, valores por defecto, restricciones a valores en los campos (constraints) y disparadores (triggers).

- ✓ Se han añadido funcionalidades en línea con el estándar SQL92, incluyendo claves primarias, identificadores entrecomillados, forzado de tipos de cadenas literales, conversión de tipos y entrada de enteros binarios y hexadecimales. (12)

A continuación se mencionan algunos de los programas usados para administrar PostgreSQL:

PgAdmin III: Herramienta gráfica, permite ver la estructura de las bases de datos, realizar operaciones SQL, ver datos, operaciones de administración. Diseñada para ejecutarse en muchos sistemas operativos (Windows, Linux, MacOS).

PhppgAdmin III: Es una poderosa herramienta de administración basada en una interfaz Web para bases de datos PostgreSQL. Equivalente a la anterior herramienta, pero está realizada en una interfaz web con PHP. Tiene la ventaja de que no requiere la instalación en los clientes, así como se puede centralizar la conexión a las bases de datos, se impide el acceso desde estaciones de trabajo y, a la vez, se facilita el trabajo a los potenciales usuarios porque no tienen que dedicar tiempo a configurar conexiones. Dispone de soporte para procedimientos almacenados, triggers y vistas.

PgAccess: Es una interfaz gráfica para el gestor de bases de datos PostgreSQL escrito por Constantin Teodorescu en el lenguaje Tcl/Tk. Permite al usuario interactuar con PostgreSQL de una manera similar a muchas aplicaciones de bases de datos para PC, con menús de opciones y diversas herramientas gráficas. Esto significa que el usuario puede evitar la línea de comandos para la mayoría de las tareas. Tiene opciones para creación de formularios e informes.

1.4 Técnicas de optimización

En cuanto a lo planteado en el epígrafe 1.2.3 se puede apreciar que entre las técnicas existentes para optimizar las bases de datos, una de las más utilizadas es la realización apropiada de su diseño inicial, además de la utilización correcta de índices en las tablas.

1.4.1 Diseño de base de datos

Es relevante destacar que de la elaboración de un adecuado diseño depende la eficacia del cumplimiento de los objetivos definidos para la base de datos, razón por la cual es lógico e imperioso el empleo del tiempo que sea requerido para aprender los principios y aspectos fundamentales de un correcto diseño.

1.4.1.1 Fases en el diseño de una Base de Datos Relacional

El proceso de diseño de bases de datos se rigen por una metodología que está conformada por seis fases, estas son: (13)

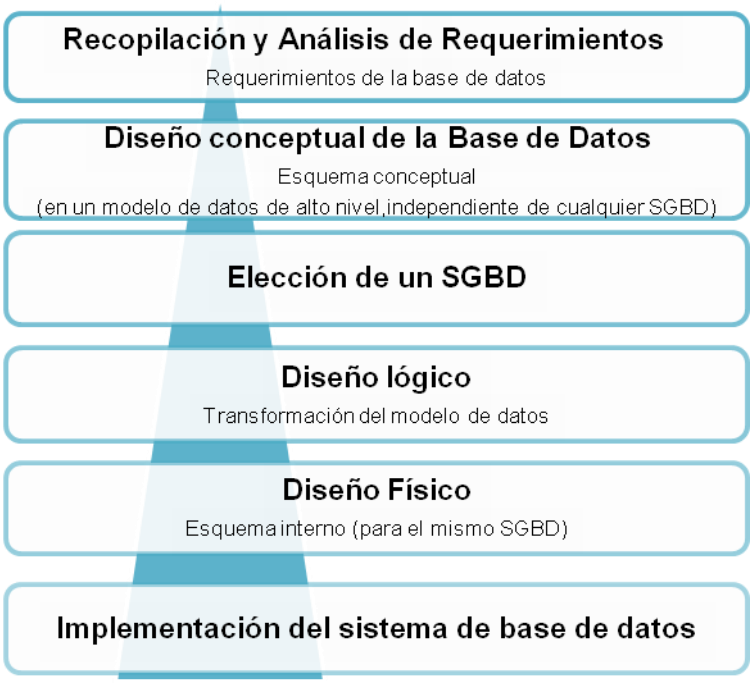

Figura 2: Fases del Diseño de una BDR

Fase 1: Recopilación y Análisis de Requerimientos.

En esta fase los diseñadores entrevistan a los futuros usuarios de la aplicación para conocer sus expectativas y, recoger y documentar sus necesidades de información. Paralelamente, conviene definir los requerimientos funcionales que consisten en operaciones (transacciones) que se aplicarán a la base de datos e incluyen la obtención de datos y la actualización. También se identifican los grupos de usuarios reales y posibles, las áreas de aplicación y se revisa la documentación existente.

Fase 2: Diseño Conceptual de la Base de Datos

Esta fase se subdivide en otras dos. Una que corresponde al Diseño del esquema conceptual y la otra al diseño de transacciones. Una vez recogidos todos los requerimientos, el siguiente paso es crear un esquema conceptual para la base de datos mediante un modelo de datos conceptual de alto nivel, el cual no puede utilizarse para implementar directamente la estructura de la base de datos. El esquema conceptual contiene una descripción detallada de los requerimientos de información de los usuarios e incluyen descripciones de los tipos de datos, relaciones entre ellos y restricciones.

La fase que corresponde al diseño de transacciones (aquellas aplicaciones en las que se manipulan los datos almacenados) se suelen identificar mediante el estudio de las entradas y salidas de datos y su comportamiento funcional. De esta forma se identifican transacciones de recuperación, de actualización y mixtas.

Fase 3: Elección de un SGBD.

Para el cumplimiento de esta fase se consideran diferentes factores técnicos, económicos y de beneficio, de servicio técnico y formación de usuarios, organizativos de rendimiento, etc. Sin embargo, resulta difícil la medida y la forma de examinar y considerar los diferentes factores.

Fase 4: Transformación del Modelo de Datos o Fase de Diseño Lógico.

El siguiente paso en el proceso de diseño, es el más cercano a la implementación de un sistema manejador de bases de datos, con el SGBD seleccionado, transformando el modelo conceptual al modelo de datos empleados por el SGBD (jerárquico, red o relacional).

Fase 5: Diseño Físico.

En este paso se especifican las Estructuras de Almacenamiento Internas y las Estrategias de Acceso:

- ✓ **Estructuras de almacenamiento**: como almacenar y organizar los archivos de la base de datos, para alcanzar un rendimiento óptimo de las aplicaciones de la misma.
- ✓ **Estrategias de Acceso:** acceso secuencial, acceso binario.

Los objetivos principales en esta etapa de diseño son:

- ✓ Disminuir los tiempos de respuesta.

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

Estos objetivos son fundamentales para optimizar el diseño físico y el ratio costo/beneficio. Los criterios adoptados suelen ser la utilización de espacio, el tiempo de respuesta y el volumen de transacciones por minuto, generalmente estos últimos se reducen a la selección de índices para acelerar el acceso. También selecciona los tipos de datos.

Fase 6: Implementación del sistema de base de datos.

En esta fase final se hace realidad la base de datos, mediante la creación y la compilación de uno o varios esquemas y ficheros de bases de datos, así como de las transacciones a través de las aplicaciones.

La metodología expuesta, que puede servir como marco de referencia general, puede modificarse y adaptarse según las características del contexto en el que se diseña y despliega el sistema de bases de datos. (14)

1.4.1.2 Modelo Relacional

Hoy en día existen diversos modelos para el diseño de bases de datos. Cada uno de ellos utiliza alguna herramienta matemática para la descripción de la base de datos, por ejemplo: en los Modelos de Red se usan herramientas Semánticas, en los Orientados a Objetos se utilizan los grafos; en el Modelo Jerárquico, los árboles; y en el Modelo Relacional, las relaciones.

El Modelo Relacional fue presentado por Edgar Codd en una publicación en 1970, refiriéndose a un modelo de datos en específico, que se caracterizaba por contar con Relaciones como único objeto de tratamiento en el modelo, así también definió un álgebra como lenguaje de consulta a la que llamó Álgebra Relacional (AR), pero no contaba con ninguna manera de expresar actualizaciones, restricciones y/o cálculos sobre el modelo. (15)

En el Modelo Relacional se utiliza un grupo de tablas para representar los datos y las relaciones entre ellos. Cada tabla está compuesta por varias columnas mientras cada columna posee un nombre único. Las relaciones entre datos deben ser representadas explícitamente. Otras de las características del modelo relacional son:

- ✓ Cada tabla es a su vez un conjunto de registros almacenados en filas o tuplas.

- ✓ Cada registro representa un objeto del mundo real.
- ✓ Los valores almacenados en una columna deben ser del mismo tipo de dato.
- ✓ Todas las filas de una misma tabla poseen el mismo número de columnas.
- ✓ No se considera el orden en que se almacenan los registros en las tablas.
- ✓ No se considera el orden en que se almacenan las tablas en la base de datos.

El modelo relacional es el modelo de datos más ampliamente usado, una amplia mayoría de sistemas de bases de datos actuales están basados en él. Este modelo se encuentra a un nivel de abstracción inferior al modelo de datos Entidad –Relación. (16)

El modelo relacional se propone, como principal objetivo, aislar al usuario de las estructuras físicas de los datos, consiguiendo así la independencia de las aplicaciones respecto de los datos, finalidad perseguida desde los inicios de las bases de datos.

En este modelo, basado en la teoría matemática de las relaciones, los datos se estructuran lógicamente en forma de relaciones. Esta formalización matemática convirtió rápidamente al modelo en una fuente fundamental de investigación. (17)

1.4.1.3 Normalización de una Relación

La normalización es un proceso que pretende conseguir tablas con una estructura óptima y eficaz. El proceso de normalización está basado en lograr la independencia de los datos respecto a las aplicaciones que los usan. Es una técnica para diseñar la estructura lógica de los datos de un sistema de información en el modelo relacional, desarrollada por Edgar Frank Codd en 1972. Es una estrategia de diseño que permite la eliminación de dependencia entre atributos no deseados. Las ventajas de la normalización son las siguientes (18):

- ✓ Evita anomalías en inserciones, modificaciones y borrados.
- ✓ Mejora la independencia de datos.
- ✓ No establece restricciones artificiales en la estructura de los datos.

Otra ventaja de la normalización es con respecto al consumo de espacio, una base de datos normalizada ocupa menos espacio en disco que una que no lo está. Hay menos repetición de datos, lo que tiene como consecuencia un menor uso de espacio en disco.

La normalización se lleva a cabo en una serie de pasos. Cada paso corresponde a una forma normal que cuenta con determinadas condiciones o propiedades. Conforme se va avanzando en la normalización, las

relaciones tienen un formato más estricto y, por lo tanto, son menos vulnerables a las anomalías de actualización. Codd propuso tres formas normales, a las cuales llamó primera, segunda y tercera formas normales (1FN, 2FN, 3FN respectivamente). Luego, Raymond Boyce y Codd, conocidos por sus investigaciones en el campo de las bases de datos relacionales, propusieron una definición más estricta de 3FN, a la que se conoce como forma normal de Boyce-Codd (FNBC). Todas estas formas normales se basan en las dependencias funcionales entre los atributos de una relación. Más adelante se propusieron una cuarta forma normal (4FN) y una quinta (5FN), con fundamento en los conceptos de dependencias multivaluadas y dependencias de reunión, respectivamente.

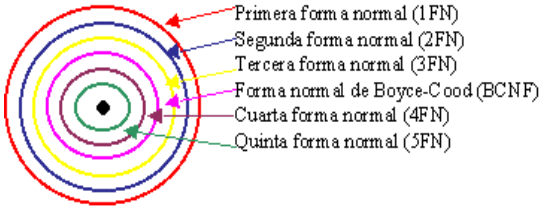

Figura 3: Relación entre las formas normales

El modelo relacional sólo requiere un conjunto de relaciones en primera forma normal. Las restantes formas normales son opcionales. Sin embargo, para evitar las anomalías de actualización, es recomendable llegar al menos a la tercera forma normal.

Las formas normales, consideradas aparte de otros factores, no garantizan un buen diseño de BD. En general no basta con comprobar por separado que cada esquema de relación de la BD esté en 3FN o FNBC, además es importante en este proceso confirmar la existencia de propiedades adicionales que los esquemas relacionales, en conjunto, deben poseer. Dos de estas propiedades son:

1. La propiedad de reunión sin pérdida, que garantiza que no se presentará el problema de las tuplas erróneas.
2. La propiedad de conservación de las dependencias, que asegura que todas las dependencias funcionales estén representadas en alguna de las relaciones individuales resultantes.

A continuación se describen los elementos más importantes a tener en cuenta de las formas normales para lograr un mayor y mejor entendimiento de las mismas.

Primera Forma Normal (1FN)

Se dice que una relación está en primera forma normal (1FN) cuando los valores para cada atributo de la relación son atómicos, esto quiere decir que cada atributo sólo puede pertenecer a un dominio y que tiene un valor único para cada fila. La 1FN se definió para prohibir los atributos multivaluados, compuestos y sus combinaciones.

Poner una base de datos en la 1FN resuelve el problema de los encabezados de columna múltiples. Muy a menudo, los diseñadores inexpertos de bases de datos harán algo similar a la tabla no normalizada. Una y otra vez, crearán columnas que representen los mismos datos. La normalización ayuda a clarificar la base de datos y a organizarla en partes más pequeñas y más fáciles de entender. En lugar de tener que entender una tabla gigantesca y monolítica que tiene diferentes aspectos, sólo hay que entender los objetos pequeños y más tangibles, así como las relaciones que guardan con otros objetos también pequeños. (19)

Segunda Forma Normal (2FN)

Una relación está en 2FN si está en 1FN y si los atributos que no forman parte de ninguna clave dependen de forma completa de la clave principal, o sea, si no existen dependencias parciales. Una dependencia parcial es un término que describe a aquellos datos que no dependen de la llave primaria de la tabla para identificarlos. Se puede afirmar que la segunda forma normal está basada en el concepto de dependencia completamente funcional. (19)

Una vez alcanzado el nivel de la Segunda Forma Normal, se controlan la mayoría de los problemas de lógica y se puede insertar un registro sin un exceso de datos en las tablas del modelo.

Tercera Forma Normal (3FN)

La tabla se encuentra en 3FN si está en 2FN y cada atributo que no forma parte de ninguna clave, depende directamente y no transitivamente, de la clave primaria. (19)

Cuando hay dependencias funcionales transitivas, se crea una nueva tabla con los atributos que tienen dependencia funcional transitiva, eliminándose el atributo dependiente de la tabla original.

Forma normal de Boyce-Codd (BCFN)

Una relación está en la forma normal de Boyce-Codd si, y sólo si, todo determinante es una clave candidata.

La 2FN y la 3FN eliminan las dependencias parciales y las dependencias transitivas de la clave primaria. Pero este tipo de dependencias todavía pueden existir sobre otras claves candidatas, si éstas existen, la BCFN es más fuerte que la 3FN, por lo tanto, toda relación en BCFN está en 3FN.

La violación de la BCFN es poco frecuente pues ocurre bajo ciertas condiciones que raramente se presentan. Se debe comprobar si una relación viola la BCFN si tiene dos o más claves candidatas compuestas que tienen al menos un atributo en común. (18)

Es importante destacar en este acápite que la normalización de las tablas de la base de datos constituye una técnica más de optimización.

1.4.2 Índices

Hacer que una consulta funcione correctamente no es lo mismo a que lo haga de la forma más rápida. Se puede acelerar las consultas de dos maneras básicamente. Una de ellas es afinando el servidor para que responda lo mejor posible mientras que la otra es haciendo uso de los índices de una manera inteligente.

Un índice en informática es como el índice de un libro, donde se tienen los capítulos del libro y la página donde empieza. En una base de datos, los índices son estructuras de datos que mejoran la velocidad de las operaciones en las tablas asociadas, permiten búsquedas de información más rápidas al proporcionar un orden e indicador de búsquedas para las consultas y evitan que el gestor revise todos los datos disponibles para devolver el resultado. Son un grupo de datos que el gestor de bases de datos asocia con una o varias columnas de la tabla en los que aparece la relación entre el contenido y el número de fila donde está ubicado.

La indexación, tanto de claves primarias como extranjeras, se puede obtener del modelo relacional.

- ✓ Las claves primarias identifican unívocamente a cada elemento de una tabla.
- ✓ Las claves extranjeras marcan las relaciones entre tablas.

Disponer de índices en los campos adecuados optimizará sus resultados. A continuación se ofrecen algunas de las mejores prácticas encontradas durante la investigación.

- ✓ Crear un índice sobre los campos que son utilizados en las búsquedas (los que aparecen en las cláusulas **WHERE o JOIN**) para mejorar una consulta (SELECT).

- ✓ Utilizar índices sobre campos con valores únicos. Los índices funcionan peor si el campo tiene valores duplicados.

- ✓ Tratar de que los índices sean cortos. Si indexa un campo de texto, evite hacerlo sobre campos de longitud variable, y acorte siempre el tamaño del índice a lo que considere más adecuado.

- ✓ Evitar la creación de índices innecesarios. Estos se actualizan con cada cambio en la tabla asociada y pueden ralentizar las modificaciones de la misma.

Cuando en una tabla no existen índices en los cuales auxiliarse, el gestor tendrá que leer todos los registros de la tabla de manera secuencial para resolver una consulta. Esto es comúnmente llamado un "escaneo completo de una tabla", y es muchas veces algo que se debe evitar pues trae consigo:

- ✓ **Sobrecarga de CPU.** El proceso de chequear cada uno de los registros en una tabla es insignificante cuando se tienen pocos datos, pero puede convertirse en un problema a medida que va aumentando la cantidad de registros en la tabla. Existe una relación proporcional entre el número de registros que tiene una tabla y la cantidad de tiempo que le toma a los gestores revisarla completamente.

- ✓ **Concurrencia.** Mientras un gestor de BD está leyendo los datos de una tabla, éste la bloquea, de tal manera que nadie más puede escribir en ella, aunque si pueden leerla. Ocurre de forma similar cuando se está actualizando o eliminando filas de una tabla, con la diferencia que aquí no se puede leer.

- ✓ **Sobrecarga de disco.** En una tabla muy grande, un escaneo completo consume una gran cantidad de entrada/salida en el disco. Esto puede calentar significativamente el servidor de bases de datos, especialmente si se tiene un disco IDE antiguo.

La operación de indexación, creada por el SGDB, ordena los registros de un archivo de datos de acuerdo con los campos utilizados como llave primaria e incrementa sensiblemente la velocidad de ejecución de algunas operaciones sobre el archivo de datos. Normalmente para cada archivo de datos debe existir un índice cuya llave de indexación sea idéntica a su llave primaria, el mismo es denominado **índice primario.** También existen los **índices secundarios** asociados a uno o varios atributos, diferentes a la clave primaria. Estos son utilizados para reducir el tiempo de localización de una determinada información dentro de un archivo o para clasificar los registros del archivo de acuerdo con el orden necesario para la obtención de la información deseada.

Existen otros **tipos de índices** que son soportados por PostgreSQL. Como son:

Btree (árbol-B): El árbol-B es un tipo de índice estándar, donde B es sinónimo de equilibrio. Un árbol equilibrado es aquel en el cual la cantidad de datos en el lado izquierdo y derecho de cada división es la misma. El árbol-B se puede utilizar para encontrar un único valor o para explorar en un área de distribución la búsqueda de los valores claves mediante el empleo de los operadores: <, <=, =, >, =>. También funciona tanto en datos numéricos como de texto. (20).

Los índices Btree se utilizan para evitar las grandes operaciones de ordenación. Se obtiene un mejor resultado cuando se aplican sobre columnas con una alta cardinalidad, es decir, sobre columnas que tengan muchos valores diferentes. Actualmente, sólo el acceso de este tipo brinda soporte para índices multi-columna.

Hash: Una tabla hash o mapa hash es una estructura de datos que asocia llaves o claves con valores. La operación principal que soporta de manera eficiente es la búsqueda. Las tablas hash son más útiles cuando se almacenan grandes cantidades de información.

Estas almacenan la información en posiciones pseudo-aleatorias, por lo que el acceso ordenado a su contenido es bastante lento. El hash sólo puede ser usado para consultas de equivalencia, o sea, para consultas con el operador: =. No soporta el comando IS NULL, a diferencia del B-tree que si lo permite. (21)

GiST: Es el acrónimo de Generalized Search Tree, (árbol de búsqueda generalizada). Los índices GiST no son un solo tipo de índice, sino más bien una infraestructura dentro de la cual muchas estrategias diferentes de indexación se pueden aplicar. En consecuencia, los operadores concretos con los que un índice GiST pueden ser utilizados, varían dependiendo de la estrategia de indexación (la clase del operador). Por ejemplo, la distribución estándar de PostgreSQL incluye clases de operador de GiST para varios tipos de datos geométricos bidimensionales, que admiten consultas indizadas utilizando estos operadores: <<, &<, &>, >>, <<|, &<|, |&>, |>>, @>, <@, ~=, &&, estos son operadores para datos geométricos. Se utilizan para búsqueda de texto completo. Los GiSt han sido mejorados de manera que soporta concurrencia de alta velocidad, recuperabilidad y rendimiento de actualizaciones, que antes estaba disponible sólo para los índices B-Tree. Es utilizado para aumentar la velocidad de búsqueda en todos los tipos de estructura de datos irregulares (arreglos enteros, datos espaciales. etc) que no son posibles de ordenar con un índice de árbol B.

Los índices GiST pueden ser utilizados como "null safe", significa que pueden indexar columnas que incluyen valores nulos, soportan el concepto de "lossiness" el cual es muy importante cuando se manejan grandes objetos GIS que el tamaño de página de PostgreSQL. Lossiness permite a PostgreSQL

almacenar solo la parte "importante" de un objeto en el índice – en el caso de los objetos GIS, solo la caja envolvente. (20)

GIN: Es el acrónimo de Generalized Inverted Index (Índice Invertido Generalizado). Los GIN son 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, los operadores particulares con los que se puede utilizar un índice GIN varían dependiendo de la estrategia de indización. Por ejemplo, la distribución estándar de PostgreSQL incluye clases de operador GIN para matrices unidimensionales, que admiten consultas indizadas utilizando estos operadores: <@, @>, =, &&, estos son operadores para datos tipo array y para "Full Text Searching" dentro de documentos a través de lexemas. (20)

Cuando persisten las dificultades en el sistema debido a problemas de hardware, software o con la correcta definición de los índices, es importante considerar y efectuar una **reindexación**, importante tarea de administración que elimina las páginas que no son usadas y vuelve a reasignar los índices de manera que estos optimicen la búsqueda de los datos y mejoren el rendimiento de los servidores.

En la propuesta se utilizarán índices de tipo B-Tree pues además de ser muy fácil su uso y poseer probados beneficios en sus métodos de inserción y eliminación al mantener balanceado el árbol, todos ellos están acordes con los requisitos funcionales del sistema.

1.5 Optimización para PostgreSQL

En la actualidad, una de las tareas más importantes de la optimización se realiza en PostgreSQL cambiando algunos de los parámetros de configuración a conveniencia de los administradores y condicionado por los requisitos del sistema. En otros casos se emplean una serie de pasos, no tan conocidos, elaborados por Josh Berkus (integrante de la comunidad Internacional de PostgreSQL). Estos pasos son más bien la primera forma que se describía, pero más organizado. Sin embargo, no poseen alguna técnica o procedimiento que ofrezca una visión práctica del estado del rendimiento actual del servidor. Además puede resultar un poco esquemático, teniendo en cuenta que el rendimiento varía según múltiples factores en cada uno de los casos y que la optimización se realiza siempre del mismo modo.

Una meta común es hacer que PostgreSQL corra más rápido en la plataforma en la que se encuentra instalado, ya sea afinando los parámetros de rendimiento o añadiendo más o mejor hardware. En PostgreSQL se utilizan sentencias o comandos para garantizar la optimización de las consultas, a continuación se mencionan algunos de estos aspectos:

- ✓ **Hardware:** se manejan elementos como el CPU, la RAM, la red y otros.
- ✓ **Postgresql.conf:** este es el archivo donde se guarda la configuración del PostgreSQL y se optimizan parámetros como la cache, la memoria compartida y la paginación. En PostgreSQL estos aspectos son vistos de la siguiente forma: shared_buffers, cache_miss, wal_buffers, full_page_writes, effective_cache_size.
- ✓ **Diseño de la aplicación:** se trata sobre optimizar el diseño de las tablas de la base de datos, la forma de indexar, el diseño de las consultas, entre otros.
- ✓ **Afinamiento de consultas:** en este punto se optimizan las consultas SQL realizadas en la base de datos. Este optimizador debe garantizar un plan de ejecución que logre la efectividad de la consulta y en su comprobación toma un valor significativo las pruebas mediante el "*explain analyze*"[4] para conocer el camino de la ejecución de las consultas, los tiempos asociados a la misma y mejorarlas en caso de que los resultados no respondan a los requisitos. (22)

1.6 Patrones de diseño de Base de Datos

En el diseño y el modelado de bases de datos es frecuente la presencia de elementos repetitivos en disímiles modelos, durante el tratamiento y resolución de estos últimos se pueden manifestar excelentes prácticas que pueden llegar a constituir patrones de diseño. Por lo tanto, un patrón de diseño es una abstracción de una solución en un nivel alto. Los patrones solucionan problemas que existen en muchos niveles de abstracción y abarcan las distintas etapas del desarrollo. Concluyendo, un patrón de diseño es un conjunto de reglas que describen como afrontar tareas y solucionar problemas que surgen durante el desarrollo de software (23).

Los Patrones de Diseños ayudan a conseguir diseños optimizados de manera ágil y con pocos esfuerzos al proponer buenas prácticas y soluciones pensadas y testeadas. Estos aportan ideas reutilizables y adaptables a un amplio campo de problemas. En la actualidad se cuenta con algunos patrones de diseño de base de datos que le sirve de mucha ayuda a los diseñadores de BD para realizar el diseño de una forma óptima. (24)

[4] EXPLAIN ANALYZE es un comando que se utiliza para chequear la exactitud del plan de ejecución, lo muestra de una forma más detallada, lo que el planificador Postgres genera para la consulta dada. Muestra los costos promedios para cada iteración y el costo total

1.6.1 Patrones de Árboles

Antes de abordar el tema de patrones de árboles es válido mencionar que en ciencias de la informática, un árbol es una estructura de datos ampliamente usada que imita la forma de un árbol (un conjunto de nodos conectados). En teoría de grafo, un árbol es un grafo conexo y acíclico. Actualmente, los árboles tienen una amplia utilización en el diseño de BD y existen patrones que se derivan de ellos, como por ejemplo:

- ✓ **Árboles Simples:** Este patrón es más utilizado cuando el árbol es la representación de una estructura de datos. Restringe los nodos de un único árbol. Ejemplo un nombre único global para cada nodo, un diccionario.

- ✓ **Árboles Fuertemente Codificados (Hardcodedtree):** Este patrón se utiliza mayormente en las relaciones de uno a muchos para representar jerarquías donde la estructura está bien definida y es importante mostrar la correspondencia entre las entidades. Se representa una entidad para cada nivel de la jerarquía y cada una coge la llave primaria de su descendiente como su llave foránea.

1.6.2 Patrones para flujos de trabajo

Se conoce por flujo de trabajo al estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan, cómo fluye la información que soporta las tareas y cómo se le hace seguimiento al cumplimiento de las tareas. Es muy usual para la vida práctica representar y persistir flujos de trabajo, si bien UML tiene su notación para la representación de un flujo de trabajo en el modelado de un base de datos es recurrente el tema de cómo representar esto en un modelo. En los últimos años existe un fuerte consenso en llevar estos patrones al mundo de las bases de datos orientada a objetos o a persistir la información no de manera relacional sino binaria usando extensiones del propio modelo relacional. En la actualidad existen dos patrones definidos para diseñar un flujo de trabajo, ellos son:

- ✓ **Máquina de estado para un tipo de entidad:** se emplea para representar los posibles cambios de estado por los que puede atravesar un tipo de entidad, no refleja el almacenamiento de las ocurrencias sino las propiedades del flujo de trabajo.

- ✓ **Máquina de estado para escenario (Control de flujo):** se utiliza para representar la ocurrencia del cambio de estado en un escenario de una entidad dada. Por lo tanto, considera el tiempo y su persistencia en las tablas resultantes. También representa la ocurrencia de un estímulo en una fecha y los estados por los que ha pasado en un intervalo de tiempo determinado.

1.6.3 Patrón Entidad-Atributo-Valor

El Modelo de entidad-atributo-valor (EAV) es un patrón flexible de modelo de datos para describir las entidades donde el número de atributos (propiedades o parámetros) que se pueden utilizar para describirlos es potencialmente enorme, pero que aplicados individualmente a una entidad concreta es pequeño. En matemáticas, este modelo se concibe como una matriz dispersa. EAV también es conocido como modelo objeto-atributo-valor, vertical de base de datos del modelo y el esquema abierto. (25)

El patrón Modelo EAV correctamente empleado es un medio para simplificar el esquema físico de una base de datos. El mismo presenta beneficios entre los que se encuentran:

- ✓ Flexibilidad. No hay límites arbitrarios sobre el número de atributos por entidad. El número de parámetros puede crecer como la base de datos se desarrolla, sin rediseño del esquema.
- ✓ Espacio de almacenamiento eficiente de datos muy escasos: No es necesario reservar espacio para los atributos cuyos valores son nulos.

1.6.4 Patrón de llaves subrogadas

Una llave o clave subrogada es un identificador único que se asigna a cada registro de una tabla. Esta clave, generalmente, no tiene ningún sentido específico de negocio. Normalmente es de tipo numérico, preferiblemente entero autoincremental, o GUID (Global Unique Identifier). El número total de claves únicas de este último asciende a la cifra de 2^{128}, es tan grande que la posibilidad de que se genere un mismo número dos veces puede considerarse nula en la práctica.

Es relevante referirse a que el empleo del patrón de llave subrogada, en la actualidad, goza de una amplia aplicación y aceptación por parte de los diseñadores de bases de datos. A continuación se enuncian algunas de las ventajas y desventajas del mismo.

Ventajas

- ✓ La lógica de negocio no está en las claves. Protege ante cambios.
- ✓ Son pequeñas. Enteros que ocupan muy poco. Joins más rápidos.
- ✓ No hay contención pues los mecanismos de generación secuencial son rapidísimos y los provee el sistema.
- ✓ Uniformidad. Se pueden programar tareas de mantenimiento sobre tablas que asumen un esquema de llave primaria común.

Desventajas

- ✓ Siempre se requiere de joins para buscar en las tablas hijas/relacionadas.
- ✓ Normalización. Se requiere un índice único adicional sobre la clave candidata (natural) para evitar la existencia de llaves candidatas duplicadas en el dominio.
- ✓ Si el mecanismo de generación de claves lo controla la BD, hay que tener en cuenta el soporte para distintas bases de datos, que no funcionan igual.

De acuerdo con el estudio realizado respecto a los patrones de diseño de base de datos que existen hoy en día y enfocándolos a la realización de este trabajo de diploma, se ha decidido utilizar, por las potencialidades y beneficios aportados, el patrón de llaves o claves subrogadas y el de modelo entidad-atributo-valor.

Conclusiones Parciales

En este capítulo se hizo un análisis de los diferentes conceptos asociados al problema que tributan al desarrollo de la investigación, como son las bases de datos. También se abordó el tema de la optimización, la forma en que es llevada a cabo y algunas técnicas utilizadas como el buen diseño de las bases de datos y el empleo de índices. Se hizo un estudio de los diferentes patrones que existen actualmente para realizar un diseño de base de datos óptimo, en cuanto a este tema no se cuenta con abundante bibliografía. Como resultado del estudio realizado se obtuvieron los conocimientos teóricos necesarios para continuar con la investigación y se le dio cumplimiento al primer objetivo trazado en este trabajo de diploma.

CAPÍTULO II: PROPUESTA DE SOLUCIÓN

2.1 Introducción

En el presente capítulo se contribuirá significativamente al cumplimiento objetivo general del trabajo de diploma. Se brindarán los argumentos pertinentes que avalan cada una de las propuestas ofrecidas. También se describen los procesos principales que tiene lugar para la optimización de la BD, entre los que se encuentran: la creación de varios esquemas, la propuesta de integración de nomencladores, las propuestas de transformaciones del actual modelo de datos para la construcción de modelos optimizados en cada módulo de los 5 subsistema de SIGEF I, así como la agregación de nuevos índices. Además se hará un análisis del impacto y la repercusión que pueden traer los cambios planteados en el desarrollo del sistema.

2.2 Breve caracterización de la BD de SIGEF I

La base de datos de SIGEF I es una base de datos relacional (según su diseño conceptual) y distribuida pues, una vez desplegado el sistema, estará distribuido geográficamente en diversas computadoras por todas las fiscalías del país. La misma está diseñada para intervenir en la informatización y gestión de los procesos fundamentales en los que interviene la Fiscalía General de la República, a fin de garantizar con mayor economía y celeridad, el control y la preservación de la legalidad, las leyes y demás disposiciones legales, así como la promoción y el ejercicio de la acción penal pública en representación del Estado. La BD de SIGEF I se encuentra estructurada por un solo esquema que contiene 325 tablas (de las cuales 72 son nomencladores), 296 secuencias y 422 índices. En el esquema único están contenidos los 5 subsistemas de SIGEF I (**PP**, **GCPA**, **CLEP**, **HCTA** y **VF**) los cuales a su vez están integrados por 13 módulos correspondientes a los procesos que se representan en la FGR.

2.3 Transformación de un esquema único en varios

El número de tablas de la base de datos de SIGEF I (325) es una cifra significativa pero no es definitiva. El aumento de este valor en el negocio no es una posibilidad futura sino una realidad debido a que los requisitos del sistema siguen siendo analizados, revisados y en algunos casos, modificados. A lo antes descrito, se debe sumar que el desarrollo de la segunda fase del proyecto traerá un importante incremento del número de tablas en la base de datos, se podría prever un difícil y engorroso trabajo con respecto a la organización y gestión de los permisos sobre cada una de las tablas en el esquema único. Ante el evidente riesgo, se propone transformar el esquema único de la base de datos en varios esquemas, los mismos estarían definidos esencialmente por la lógica del negocio.

El empleo de varios esquemas, mediante la correcta gestión de los permisos, aportará mayor organización, seguridad, integridad y confidencialidad a la base de datos y permitirá:

- ✓ Viabilizar la definición de propietarios de esquemas siempre y cuando sea requerido.
- ✓ Posibilitar un mayor uso de las base de datos por múltiples usuarios sin interferencias.
- ✓ Permitir que se puedan instalar aplicaciones realizadas por terceros sin que existan colisiones en los nombres de los objetos.
- ✓ Tener tablas con nombre idénticos en esquemas diferentes evitando así el empleo de prefijos o sufijos para diferenciar las tablas.
- ✓ Prescindir de sufijos o prefijos en la denominación de las tablas para lograr su identificación con el módulo o subsistema al que pertenecen, o evitar que estén ubicadas desordenadamente por todo el esquema único.

El esquema único de la base de datos **(public)** se disgregará en 8 esquemas. La mitad de los mismos se ha realizado atendiendo a la organización lógica de 4 subsistemas **(clep, gcpa, hcta y vf)**. El otro subsistema **(PP),** debido a su complejidad, se fragmentará en tres esquemas adicionales **(ipp, ord y sum).** Finalmente se define un esquema "com" en el cual se agruparán las tablas de la base de datos que son requeridas en varios de los esquemas antes mencionados.

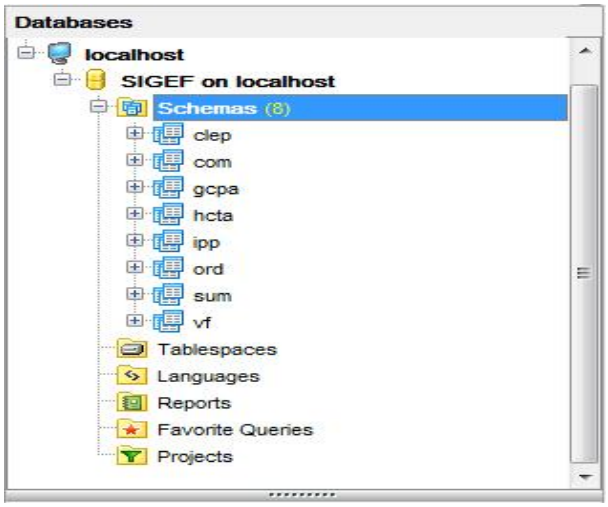

Figura 4: Esquemas Definidos

La descomposición de un esquema en varios es una actividad de complejidad media pues aunque existen esquemas como: COM, HCTA, IPP y SUM, que tienen pocas referencias hacia otros esquemas; el resto no es así, por ejemplo: GCPA necesita de los nomencladores contenidos en el esquema COM, de igual

manera CLEP, ORD y VF, referencian en múltiples ocasiones a fiscales que están almacenados en GCPA.

2.4 Propuesta de nomenclatura de la BD de SIGEF

Consideraciones generales de nomenclatura

- ✓ Los nombres se escribirán con minúscula, están limitados a 30 caracteres y siempre comenzarán con una letra, nunca con un número o carácter especial.
- ✓ Para nombrar se emplearán solo sustantivos en su forma singular.
- ✓ El uso de números, guiones bajos, acrónimos y abreviaciones serán el mínimo e indispensable, siempre teniendo en cuenta la nomenclatura de las tablas y los nombres reales del negocio.
- ✓ No se permite el uso de tildes, diéresis ni "espacios" entre palabras.
- ✓ Se sustituirá la letra "**ñ**" por "**nn**".

Tablas

Una vez dividido el esquema único en varios esquemas, es realmente importante definir la nomenclatura que se empleará para nombrar las tablas, columnas y relaciones de los mismos. Esta definición obtiene mayor relevancia cuando en la actualidad no existe una nomenclatura general definida en SIGEF I. A continuación se presenta una propuesta que tiene en cuenta todas de las transformaciones expuestas previamente.

La denominación de las tablas de la base de datos contará con la siguiente nomenclatura:

"<nombre del esquema + '.' + nombre de la tabla>"

Ejemplos:

- ✓ ***gcpa.cuadro***
- ✓ ***ipp.expediente***
- ✓ ***sum.denuncia***

Las tablas nomencladoras que no son recogidas (por sus especificidades) en la propuesta de integración de nomencladores definidos en el próximo acápite, se podrán encontrar en el esquema común ***"com"*** y se denotarán de la siguiente manera: *"<**n** + nombre de la tabla>"*

Ejemplos:

- ✓ ***ndelito***
- ✓ ***ntipofichero***

Columnas

Al nombrar los atributos de la tabla no se requerirá mencionar el nombre de la misma a menos que se trate de la clave primaria, en este caso sería:

"<id + nombre de la tabla>".

Para nombrar atributos que contienen claves foráneas se empleará la siguiente nomenclatura:

"<id + nombre de la tabla referenciada>".

(Coincidiría con el nombre del id de la tabla referenciada).

Dándose el caso de que una tabla contenga dos o más claves foráneas procedentes de la misma tabla, entonces la nomenclatura se definiría de la siguiente forma:

"<id + nombre de la tabla referenciada + '(guión bajo)' + rol desempeñado>".

Si no se pueden definir estos roles entonces sería:

"<id + nombre de la tabla referenciada + #>".

En el caso particular de que la llave foránea sea de un nomenclador recogido dentro de la propuesta de integración, la nomenclatura sería:

"<idtipo + *nombre del nomenclador*>".

Si en una misma tabla existen dos o más llaves foráneas correspondientes a un mismo nomenclador recogido dentro de la propuesta de integración entonces la nomenclatura se definiría de la siguiente forma:

*"<**idtipo** + nombre del nomenclador + #>".*

La restricción de llave primaria se denominará de la siguiente manera:

"<nombre de la tabla + '(guión bajo)' + pkey>".

En caso de que la tabla tenga más de una clave primaria se agregará a la denominación un número de la siguiente manera:

"<nombre de la tabla + '(guión bajo)' + pkey + # >".

Las claves foráneas deben tener el mismo nombre que aparece en la tabla a la que hace referencia donde son llaves primarias. La nomenclatura de su restricción estaría determinada de la siguiente manera:

"<nombre de la tabla referenciada + '(guión bajo)' + fk>".

En caso de que existan varias llaves foráneas procedentes de la misma tabla entonces se agregará un número a la nomenclatura:

"<nombre de la tabla referenciada + '(guión bajo)' + fk + # >".

2.5 Propuestas de integración de nomencladores

Durante el estudio del negocio, los requisitos y la documentación existente relacionada con la base de datos de SIGEF I, uno de los aspectos más interesantes encontrados lo constituyó el empleo de tablas nomencladoras que contenían en sus tuplas todos los posibles valores en los que podría expresarse un determinado atributo. Por ejemplo, para gestionar el nivel alcanzado por un fiscal en cuanto a su superación idiomática, se tiene una tabla (***nnivelalcanzado***) con dos atributos: el identificador y el tipo, el primero un identificador autoincrementable (patrón de llave subrogada) y el segundo los valores específicos del nivel alcanzado (Alto, Medio o Bajo).

Tabla 1: Estudio Estadístico

Aspectos analizados	**Valor**
Total de tablas	325
Total de tablas nomencladoras	72
Porciento de tablas nomencladoras del total de tablas	22, 15%.
Media de la cantidad de tuplas existentes en las tablas nomencladoras	7,93
Total de tablas nomencladoras con menos de 10 tuplas	60
Porciento de tablas nomencladoras (- 10 tuplas) del total de tablas nomencladoras	83,3%
Total de tablas nomencladoras con solo los atributos id y tipo	59
Porciento de tablas nomencladoras (con id y tipo) del total de tablas nomencladoras	81,95%
Cantidad de tuplas existentes en las tablas nomencladoras con solo los atributos id y tipo.	328
Media de la cantidad de tuplas existentes en las tablas nomencladoras con solo los atributos id y tipo	5,56

Una vez realizado el análisis se concluye que, actualmente, en la base de datos de SIGEF I más del 20% de las tablas presentes son tablas nomencladoras cuyo fin es contener en sus tuplas los diferentes valores en los que se puede manifestar un determinado atributo. De estas tablas (72 en total) casi 82% se

caracterizan por tener solo dos atributos: identificador y tipo; mientras que el 83,3% son tablas nomencladoras con menos de 10 tuplas. Un elemento muy interesante resulta el hecho de que la media de la cantidad de tuplas existentes en las tablas nomencladoras es **7,93**; sin embargo, la media de la cantidad de tuplas existentes en las tablas nomencladoras con solo los atributos identificador y tipo es de solo **5,56.**

Partiendo de la información expuesta y el análisis de las potencialidades de dos patrones de diseño de bases de datos: modelo Entidad-Atributo-Valor y llave subrogada, se propone el uso de dos tablas que suplirán la utilización de otras 59 (valor que pudiera aumentar con el desarrollo de la base de datos). Esta idea en su esencia constituye una variante del modelo EAV que empleando identificadores autoincrementables propone un manejo óptimo de los nomencladores con solo los atributos: identificador y tipo. Estos constituyen un poco más de las cuatro quintas partes del total de tablas nomencladoras del sistema.

La propuesta se basa en la creación de una tabla (***nomenclador***) con los atributos: ***idnomenclador*** (patrón de llave subrogada) y ***nombrenomenclador*** (***varchar***), en la que se almacenarán los nombres de todos los nomencladores que cumplen con las condiciones de la propuesta. La misma estará asociada la tabla ***tiponomenclador*** por la llave o clave foránea (***idnomenclador***), la cual contará también con los atributos: ***idtiponomenclador*** y ***tiponomenclador***. Además se recopilarán todos los posibles valores en lo que se puede manifestar cada nomenclador contenido en esta propuesta.

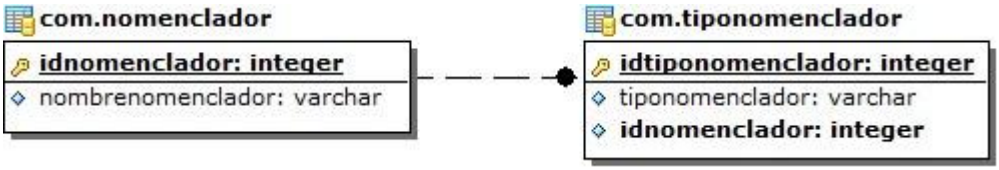

Figura 5: Propuesta de nomencladores

La propuesta de integración presentada es ante todo un medio para la simplificación del esquema físico de la base de datos, un modo de describir el sistema más activo (teniendo en cuenta su funcionamiento real) y una manera de utilizar y explotar aún más los metadatos definidos. Aporta significativos beneficios como:

- ✓ Espacio de almacenamiento eficiente de datos muy escasos: No es necesario reservar espacio para los atributos cuyos valores son nulos.
- ✓ Flexibilidad. No hay límites arbitrarios sobre el número de atributos por nomenclador. El número de parámetros puede crecer como la base de datos se desarrolla, sin rediseño del esquema.

Con respecto a este último aspecto se hace imprescindible señalar que aunque la propuesta está diseñada para nomencladores con solo: id y tipo, el diseño y la concepción de la misma no deja de ser flexible ni es vulnerable a los impactos de los cambios futuros. Ante la ocurrencia de cambios en los requisitos del sistema, específicamente en el uso de los nomencladores, de los cuales además del id y el tipo, se necesitaría conocer la trayectoria histórica de los mismos. Dada las potencialidades de la propuesta, la solución estaría simplemente en agregar una tabla (***codigonomenclador***) con los atributos: ***idcodigoanterior*** (***integer***) e ***idodigocactual*** (***integer***); la cual registraría los cambios históricos de un determinado tipo de nomenclador.

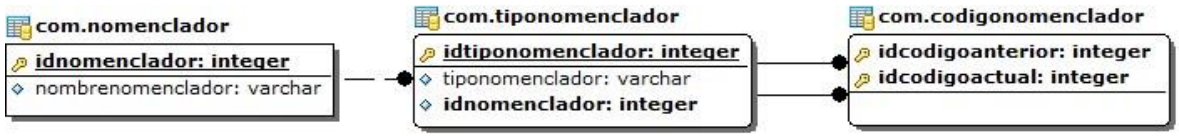

Figura 6: Otra propuesta de nomencladores

Es importante destacar que la idea inicial no es esquematizar todos los nomencladores en la propuesta sino aprovechar las características comunes de la generalidad de los mismos. Aquellos que tengan atributos muy específicos se tratarían como tablas nomencladoras independientes. En caso de que un determinado atributo se generalice, pudiera agregarse perfectamente como una nueva tabla, siguiendo la misma concepción del ejemplo expuesto, relacionada con ***nomenclador*** o ***tiponomencladores*** según sus respectivas características.

La complejidad de la implementación de esta propuesta se debe valorar por esquemas. En la mayoría de los casos resulta una tarea de baja complejidad pues la cantidad de cambios referenciales (relaciones con las nuevas tablas contenedoras de los nomencladores y sus diferentes manifestaciones) es pequeña. En cambio, por la cantidad de nomencladores que se prevé incorporar a la propuesta del esquema GCPA, esta actividad tendrá una complejidad superior.

2.6 Optimización del diseño del modelo de datos de SIGEF I.

El eficiente diseño de una base de datos es la clave para lograr su optimización y, en consecuencia, el rendimiento óptimo de la aplicación. Sin embargo, si este es deficiente puede comprometer el rendimiento de todo el sistema.

La optimización de la base de datos de SIGEF I depende en gran medida del mejoramiento y modificación de su diseño. Para alcanzar el objetivo mencionado se han revisado, junto a los principales analistas, la Especificación de requisitos de software y el Modelo del sistema de cada uno de los módulos de los 5 subsistemas. También se analizó el actual modelo de datos y se detectó la imperiosa necesidad de

realizar importantes transformaciones que tributen a eliminar redundancias y ambigüedades, normalizarlo hasta la forma que el negocio lo permita, adaptar el diseño a los nuevos requisitos definidos recientemente, conseguir la máxima seguridad e integridad de los datos y mejorar el consumo de recursos.

En los acápites que se desarrollarán a continuación se mostrarán los actuales modelos de datos de los subsistemas o módulos, con una breve descripción de sus objetivos y las principales transformaciones propuestas. Finalmente se mostrará el módulo de datos obtenido como resultado de la aplicación de los cambios previamente planteados y de la nomenclatura propuesta en este capítulo. La secuencia en el que se mostrarán los modelos responde al orden de los esquemas a los que pertenecen.

2.6.1 Esquema CLEP

El módulo Inspección a Locales y Centros Penitenciarios del subsistema CLEP tiene como objetivo principal la informatización del proceso de planificación, preparación, ejecución y conclusión de las visitas de inspección a los establecimientos penitenciarios, centros de reclusión de asegurados, centros correccionales, unidades en que se cumpla la prisión provisional de acusados y cualquier otro centro de reclusión o internamiento.

Para una mejor comprensión de las transformaciones realizadas que se expondrán a continuación del modelo CLEP ver el Anexo 1.

Principales Transformaciones:

Eliminación de tablas innecesarias:

Las tablas ***tbcomunicacion, tbviolacioninspeccion, tbotramodalidad, tbnsector, tbrespotramodalidad, tbrespuestafichero, tbninfractor, tbficheroclep, tbregresoluciones*** son innecesarias según los requisitos y reglas del negocio (actualizadas). La existencia de estas no responde a ninguna funcionalidad y para evitar redundancias en la base de datos se propone su eliminación.

Con la definición de varios esquemas dentro de una misma base de datos, las tablas utilizadas por varios módulos se agruparon en el esquema ***"com"***. En el módulo CLEP las tablas con esta condición son: ***nprovincia***, ***municipio***, ***ntipofichero***, ***fichero***, ***datosfichero*** y ***fiscalias***, por este motivo y para evitar redundancias en la base de datos (dado que ya se encuentran en otro esquema) se propone su eliminación.

Eliminación de nomencladores:

El nomenclador ***ntrimestre*** es innecesario pues el objetivo del mismo es recoger un trimestre determinado. Es favorable con respecto a memoria y complejidad tener un atributo (***integer***) que guarde el número del trimestre, que tener una tabla completa para guardar 4 tuplas con las cadenas de caracteres (Primer trimestre, Segundo trimestre, Tercer trimestre y Cuarto trimestre). Por las razones anteriores se propone su eliminación.

El nomenclador ***nvisitasconjuntas*** cumple con las condiciones de la propuesta de solución de nomencladores, motivo por el cual su contenido se incorporará a las tablas ***tbnomenclador*** y ***tbtiponomenclador***, y se propone su eliminación de la base de datos.

Eliminación de atributos innecesarios:

En las tablas: **tb*actainspeccion***, **tb*estudiolegalidad***, **tb*notificacion***, **tb*planinspeccion***, **tb*resolucion***, **tb*planmedidas***, **tb*planobjetivos***, **tb*registroinspeccion*** y **tb*resoluciontipo,*** se encuentra el atributo ***fecha*** referido a la fecha de creación del documento, que estas a su vez, son tablas que responden a especializaciones de la tabla ***documento,*** por tal motivo se propone eliminar el atributo mencionado de las tablas especializadas y agregarlo en la tabla genérica.

En las tablas: ***tbresolucionclep***, ***tbnotificacion***, ***tbresoluciontipoclep***, ***tbactainpeccionclep***, ***tbregistroinspeccionclep***, ***tbplanobjetivosclep***, ***tbplaninspecionclep*** y ***tbestudiolegalidadclep,*** se encuentra el atributo referido al fiscal que interviene en la confección del documento, estas tablas , a su vez, responden a especializaciones de la tabla ***tbdocumentoclep,*** por tal motivo se propone eliminar el atributo mencionado de las tablas especializadas y agregarlo en la genérica.

En la tabla **tb*actainspeccion*** se propone la eliminación de los atributos: ***fechainicio***, ***fechafin***, ***horainicio*** y ***horafin***, pues estos pueden declararse como un solo atributo ***fecha (date)***.

En la tabla **tb*resoluciontipo*** se propone la eliminación de los atributos: ***idcentro*** y ***fechaimpugnacion,*** pues estos carecen de funcionalidad según los requisitos y reglas del negocio (actualizadas). En esta misma tabla se propone la eliminación de los atributos: ***cargoimpugna*** y ***gradoimpugna*** puesto que el atributo ***personaimpugna*** puede declararse como una clave foránea procedente de la tabla **tb*persona***, en la misma se encuentran los dos atributos antes mencionados y es un modo eficiente de reutilizar la información presente en la BD.

En la tabla **tb*resolucion*** se propone la eliminación de los atributos: ***fechainicio***, ***fechafin***, ***horainicio*** y ***horafin,*** pues estos pueden declararse como un solo atributo ***fecha (date)***. En la misma tabla los atributos: ***clasificada***, ***impugnada*** y ***fechaimpugnacion*** son redundantes pues según los requisitos y reglas del negocio (actualizadas) carecen de funcionalidad por lo que se propone su eliminación.

Finalmente se propone la eliminación de los atributos: ***cargodestinatario*** y ***gradodestinatario*** debido a que el atributo ***nombredestinatario*** puede declararse como clave foránea procedente de la tabla ***persona***, en esta se encuentran los tres atributos antes mencionados y es un modo eficiente de reutilizar la información presente en la BD.

En la tabla ***registroinspeccion*** se propone la eliminación del atributo: ***funcionarios*** debido a que según los requisitos y reglas del negocio (actualizadas) su existencia carece de funcionalidad.

En la tabla **tb*resolucion*** se propone la eliminación de los atributos: ***cargodestinatario*** y ***gradodestinatario*** debido a que el atributo ***destinatario*** puede declararse como clave foránea procedente de la tabla ***persona***, en esta se encuentran los tres atributos antes mencionados y es un modo eficiente de reutilizar la información presente en la BD.

En la tabla **tb*planinspeccion*** se propone la eliminación de los atributos: ***origen, conjuntafiscaliamilitar, idplanprovincial, departamento y conjuntaotros,*** pues estos carecen de funcionalidad según los requisitos y reglas del negocio (actualizadas).

En la tabla **tb*notificacion*** se propone la eliminación del atributo: ***personaanotificar,*** puesto que carece de funcionalidad según los requisitos y reglas del negocio (actualizadas).

En la tabla **tb*planobjetivo*** se propone la eliminación de atributo ***objetivos*** debido a la existencia de la tabla ***planobjetivo*** que responde de una manera más eficiente y abarcadora a la misma funcionalidad.

En la tabla ***tbdocumentoclep*** se propone la eliminación de los atributos: ***idfiscalía*** e ***idprovincia*** ,pues estos carecen de funcionalidad según los requisitos y reglas del negocio (actualizadas), por lo que se propone su eliminación.

Cambio del tipo de atributo:

En la tabla ***tbviolacion*** se propone el cambio del atributo ***tipodoc*** de ***integer*** a ***boolean*** avalado por la naturaleza booleana del mismo***.***

Creación de tablas:

Se propone la creación de la tabla ***tbimpugnacion***, pues esta responde a nuevas funcionalidades agregadas a los requisitos del sistema.

Se propone la creación de la tabla ***tbimpugnacionfichero*** con los identificadores de las tablas ***impugnacion(serial)*** y ***datosfichero(idfichero),*** resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente.

Se propone la creación de la tabla ***tbhistorialcentro,*** con esta se pretende resguardar la información histórica de cambios que pueden producirse en todos los centros CLEP.

Agregación de atributos a una tabla:
Se propone la adición del atributo ***fechacreacion*** (***date***) a la tabla ***centro*** con el objetivo de conservar la información de los cambios que puedan ocurrir en los centros CLEP. Esta auxiliará el control histórico que se lleva en la tabla ***tbhistoriacentro****.*

Se propone la adición del atributo ***fecha*** (***date***) a la tabla ***tbdocumentoclep*** debido a que todas las tablas especializadas de la misma lo requieren, se considera que es óptimo contener este atributo en la tabla genérica y no en cada una de las tablas especializadas.

Se propone la adición del atributo ***idfiscal*** (***char 11***) a la tabla ***tbdocumentoclep*** debido a que todas las tablas especializadas de la misma lo requieren, se considera que es óptimo contener este atributo en la tabla genérica y no en cada una de las tablas especializadas.

Con todas estas transformaciones en el modelo CLEP, la propuesta quedaría como se muestra en el Anexo 2.

2.6.2 Esquema COM

El esquema ***"com"*** tiene la peculiaridad de no pertenecer a un módulo específico en un determinado subsistema, pues en el mismo se agrupan aquellas tablas de la base de datos que son requeridas por otras tablas de varios esquemas. En la misma también se encuentran las soluciones dadas a los problemas generales de la base de datos de SIGEF I.

Tabla 2: Descripción de las tablas

Tabla	Descripción
persona	Constituye la tabla genérica de todas las tablas que almacena información sobre personas. Su llave primaria es una clave semántica conformada por el número de identidad de la persona (CHAR 11).
usuario	Es una especialización de la tabla tbpersona. Su objetivo es almacenar el usuario, la contraseña y la dirección email de todas las personas poseedoras de estas y autorizadas al uso de las mismas.
historial	Se encarga de guardar el historial de acciones realizadas por los usuarios autenticados en el sistema.
fichero	Almacena los ficheros, independientemente de su naturaleza, guardados

	en el sistema. Para esta funcionalidad se emplea un atributo de tipo bytea. La misma es parte de la solución común implementada para el tratamiento y almacenamiento de todos los documentos que circulan en las fiscalías.
datosfichero	Es una especialización de la tabla fichero. Su objetivo es almacenar de manera independiente atributos de los documentos almacenados que son significativos y requeridos para funcionalidades del sistema. Es parte de la solución común implementada para el tratamiento y almacenamiento de todos los documentos que circulan en las fiscalías.
ntipofichero	Nomenclador cuyo objetivo es guardar cada uno de los tipos de documentos que son archivados en el sistema. Es la tercera y última tabla que integra la solución común implementada para el tratamiento y almacenamiento de todos los documentos que circulan en las fiscalías.
nomenclador	Contiene los nomencladores identificados en el negocio y en el sistema que cumplen con todas las condiciones de la propuesta de solución de nomencladores.
tiponomenclador	Almacena cada uno de los tipos en los que pueden manifestarse los nomencladores contenidos en la tabla tbnomenclador. También es parte de la propuesta de solución de nomencladores.
fiscalia	Guarda atributos tan significativos como: denominación, dirección y municipio al que pertenece de todas las fiscalías del país.
unidadpnr	Salva atributos necesitados por el sistema tales como: el código, el teléfono, si está activa o no, el jefe de unidad y el municipio, de cada una de las unidades de PNR registradas desde el esquema "ipp" o "sum".
nmunicipio	Nomenclador que almacena cada uno de los municipios del país, vigentes o no, posterior a la división político administrativa de 1975.
nprovincia	Nomenclador que archiva cada una de las provincias del país, vigentes o no, posterior a la división político administrativa de 1975.
nnombreanno	Nomenclador definido para unificar en una tabla común la manera de almacenar en el sistema el nombre de los años, ante la existencia de varias tablas nomencladoras que cumplen con el mismo objetivo pero de diferentes modos.
ndelito	Nomenclador definido para el almacenamiento de cada uno de los delitos y sus descripciones, presentes en los procesos fiscales.

La propuesta del modelo COM se puede ver en el Anexo 3.

2.6.3 Esquema GCPA

El subsistema de Gestión de Cuadros y Personal de Apoyo tiene como objetivo principal garantizar la gestión de los cuadros (fiscales) y el personal de apoyo (estudiantes que cursan la carrera de derecho y que están haciendo las prácticas en la fiscalía). Implica además, la gestión de datos de cuadros y del personal de apoyo que están vinculados a la fiscalía, la gestión de la capacitación de cuadros así como, el movimiento de cuadros dentro de la fiscalía. Otros de los objetivos es garantizar los niveles de seguridad y acceso tanto de los cuadros como del personal de apoyo.

Para una mejor comprensión de las transformaciones realizadas que se expondrán a continuación del modelo GCPA, ver el Anexo 4.

Principales Transformaciones

Eliminación de tablas innecesarias:

Las tablas ***tbnotificaciones, tbsecuencias, tbnormativas, tbterminosrelacionados, tbdiccionario, tbterminosinonimos, tbtrimestreeval, tbvalorescuadro, tbfechaidioma, tbcategfiscalcuadro, tbaspectosdireccion, tstiposuptipocap, tscuadroscargo, tbacuerdocuadrocargo, tbcuadrocentrodoc, tscuadrounjc, tbcuadrocapatulosprov, tbrelacionadacon, tbpersonasesferaw, tbcargospermisos, tbpersonapermiso, tbclaves, tblugartrabajo, tbotraspersonas*** y ***tbpersonatipolincencia***, son tablas que cumplían diferentes funciones u objetivos en las primeras versiones del módulo, sin embargo, partiendo de la actualización de las reglas del negocio y las necesidades de los requisitos de sistema (que han cambiado con respecto a las primeras versiones) se considera que estas tablas carecen de objetividad y pueden ser sustituidas por otras que desempeñarían la misma función pero de una manera óptima. Por tales motivos se propone eliminarlas de la base de datos.

Eliminación de nomencladores:

Los nomencladores ***nestadoactual, nmedicion, ncalificacion, nestadosalud, nnivelalcanzado, nidioma, npais, ncondecoracion, ntiposuperacion, ntipocapacitacion, naspectodireccion, ntiporesolucion, netnia, nestadocivil, nplazo, nmilitancia, ntiposancion, ntipoacuerdo, nesferasw, nmotivobaja, nsexo, , asignaturas, npermanecia, ncarrera, ntipolicencia*** y ***ntiponivelreserva,*** cumplen con todas las condiciones de la propuesta de solución de nomencladores, motivo por el cual su contenido se incorporará a las tablas ***tbnomenclador*** y ***tbtiponomenclador*** y se propone su eliminación de la base de datos.

Se propone la eliminación de los nomencladores ***nemisor, ntiponorma, ntipocuadros, ncamposhabilitados, nmac, nsinonimo, nterminosasociados, nevaltrimestre, nevaltrimestre ntipoevaluacion, nhabilidad, nevaluacion, ntrimestre, ncategoriadocente, ncategoriafiscal, nestado, ncentrodocente, ninstitucion*** y ***ncapitulosprov***. Estos nomencladores, que en un momento determinado respondían a diferentes requisitos, con la revisión y actualización de los mismos ya no cumple ningún objetivo en la base de datos.

Eliminación de claves innecesarias:

Debido a cambios que se han producido en la modelación del negocio y a la actualización de los requisitos del sistema se propone la eliminación de las claves o llaves primarias ***idevaluacion*** e ***idhabilidad***. En el caso de la clave ***idnivelalcansado*** pasaría de primaria a foránea. Todas estas transformaciones ocurrirían en la tabla ***tbcuadroidioma***.

Creación de tablas:

Se propone la creación de la tabla ***tbcapacitacionaspectosdireccion*** con los identificadores de las tablas ***tbcapacitacioncuadro*** (***idcapacitacioncuadro***) y ***tiponomenclador*** (***idtiponomenclador***), en esta última se encuentran almacenados todos los aspectos de dirección. La tabla propuesta se crea como resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente. Esta propuesta presenta una solución óptima al manejo de los aspectos de dirección en los que puede capacitarse uno o varios cuadros.

Se propone la creación de la tabla ***tbcuadrotipolicencia*** con los identificadores de las tablas ***tbcuadro*** (***idpersona***) y ***tiponomenclador*** (***idtiponomenclador***), en esta última se encuentran almacenados todos los tipos de licencia de conducción que puede tener un cuadro. La tabla propuesta se crea como resultado de la relación de muchos a muchosentre las tablas mencionadas anteriormente. Esta propuesta presenta una solución óptima al manejo de todos los tipos de licencias de conducción que pueden tener uno o varios cuadros.

Se propone la creación de la tabla ***tbcuadroesferaw*** con los identificadores de las tablas ***tbcuadro*** (***idpersona***) y ***tiponomenclador*** (***idtiponomenclador***), en esta última se encuentran almacenadas todas las esferas de trabajo presentes en las fiscalías del país. La tabla propuesta se crea como resultado de la relación de muchos a muchosentre las tablas mencionadas anteriormente. Esta propuesta presenta una solución óptima al manejo de todas las esferas de trabajo por la cual pueden transitar uno o varios cuadros (fiscales).

Se propone la creación de la tabla ***tbcuadropermisos*** con los identificadores de las tablas ***tbcuadro*** (***idpersona***) y ***tiponomenclador*** (***idtiponomenclador***), en esta última se encuentran almacenados todos los permisos que puede tener un cuadro. La tabla propuesta se crea como resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente. Esta propuesta presenta una solución óptima al manejo de todos los permisos que se le pueden proporcionar a uno o varios cuadros (fiscales).

Se propone la creación de la tabla ***tbtipolicenciapersonaapoyo*** con los identificadores de las tablas ***tbpersonaapoyo*** (***idpersona***) y ***tiponomenclador*** (***idtiponomenclador***) en esta última se encuentran almacenados todos los tipos de licencia de conducción que puede tener el personal de apoyo. La tabla propuesta se crea como resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente. Esta propuesta presenta una solución óptima al manejo de todos los tipos de licencia de conducción que pueden poseer una o varias personas del personal de apoyo.

Se propone la creación de la tabla ***tbpermisospersonaapoyo*** con los identificadores de las tablas ***tbpersonaapoyo*** (***idpersona***) y ***tiponomenclador*** (***idtiponomenclador***), en esta última se encuentran almacenados todos los permisos que puede tener el personal de apoyo. La tabla propuesta se crea como resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente. Esta propuesta presenta una solución óptima al manejo de todos los permisos que se le pueden proporcionar al personal de apoyo.

Se propone la creación de la tabla ***tbacuerdoanppoce***, una tabla producto de la especialización de la tabla ***tbacuerdo***, que responde de una manera óptima a los requisitos del sistema.

Se propone la creación de la tabla ***tbpersonalapoyo***, una tabla producto de la especialización de la tabla ***tbpersona***, que responde de una manera óptima a los requisitos del sistema, específicamente en el registro de todas las personas que apoyan el trabajo en todas las fiscalías del país.

Agregación de atributos a una tabla:

Se propone la adición de los atributos: ***fechagraduacion*** (***date***), ***expresionoral*** (***integer***), ***lectura*** (***integer***) y ***escritura*** (***integer***) a la tabla ***tbcuadroidioma*** los cuales son requeridos por los requisitos y reglas del negocio.

Partiendo de las necesidades de los requisitos del sistema y de los cambios en el modelado del sistema para optimizar el diseño de la base de datos se propone adicionarle a la tabla ***tbcuadro*** los siguientes atributos: ***fiscalia*** (***integer***), ***direccionparticular*** (***varchar***), ***municipiorecide*** (***integer***), ***telefono***

(***integer***), ***noregistro*** (***integer***), ***nopasaporte*** (***integer***), ***nolicenciaconduccion*** (***integer***), ***experienciafiscal*** (***integer***) y ***activo*** (***boolean***).

Con todas estas propuestas, el modelo GCPA quedaría como se muestra en el Anexo 5.

2.6.4 Esquema HCTA

El subsistema de Herramientas Comunes a Todas las Áreas tiene como objetivo principal la organización de los subsistemas de la aplicación. Es el punto de partida de la aplicación, el módulo que va a engranar todos los demás subsistemas y a partir de él los usuarios pueden acceder a los demás subsistemas. Engloba toda una serie de herramientas comunes que estarán inmersas en el resto de los subsistemas como son el Diccionario Jurídico (DJ) y el Gestor de Normativas Jurídicas.

- ✓ **Módulo Diccionario Jurídico**

El módulo de Diccionario Jurídico perteneciente al subsistema HCTA tiene como objetivo principal la gestión de los términos categorizados por materias propias del derecho (penal, civil, laboral u otros) o no, como puede ser el caso de términos médicos, pero que son requeridos por los fiscales en el desempeño de sus funciones.
Para una mejor comprensión de las transformaciones realizadas que se expondrán a continuación del modelo DJ, ver el Anexo 6.

Principales Transformaciones

Eliminación de claves innecesarias:
La relación de muchos a muchos entre una misma entidad se traduce en una nueva tabla que tiene como identificador dos id de la entidad. Por este motivo se propone la eliminación de los indicadores en las siguientes tablas:

Tabla 3: Indicadores eliminados

Tabla (nombre anterior)	Tabla(nombre actual)	Indicador
tbrelaciontema_dj	tbtemarelacionado	idrelaciontema
tbsinonimo_dj	tbsinonimo	idsinonimo_dj
tbterminorelacionado_dj	tbterminorelacionado	Idterminorelacionado
tbdiccionarioimagen_dj	tbterminofichero	iddiccionarioimagen

En estos casos, el patrón de llave subrogada no está eficientemente empleado.

Eliminación de atributos innecesarios:

En la tabla ***tbdiccionario_dj*** el atributo ***imagen*** es redundante debido a la existencia de la tabla ***tbdiccionarioimagen_dj*** actualmente nombrada ***tbterminofichero,*** esta es creada para responder a la misma necesidad pero de una manera completa según los requisitos del sistema.

Con todas estas transformaciones en el modelo DJ, la propuesta quedaría como se muestra en el Anexo 7.

- ✓ **Módulo: Gestión de Entrada y Salida de Documentos (GESD)**

El módulo de Gestión de Entrada y Salida de Documentos perteneciente al subsistema HCTA tiene como objetivo principal la informatización de los procesos relacionados con la gestión de la correspondencia y documentación oficial que entra y sale de la fiscalía, brindando la posibilidad de conocer su estado y ubicación en todo momento.

Para una mejor comprensión de las transformaciones realizadas que se expondrán a continuación del modelo GESD ver el Anexo 8.

Principales transformaciones

Eliminación de tablas innecesarias:

Se propone la eliminación de las tablas ***tbdocumentoremitentedoc, tbentradadocumentodoc, tbmovimientodoc, tbmovimientoexternodoc, bsecretaria, tbdepartamentodoc, tbdirecciondoc, tbsalidadocumentodo y tbpersonanaturaldoc*** debido a que presentan ambigüedades en su diseño, lo que provoca redundancia de datos a partir del análisis de los requisitos del sistema y las reglas del negocio (actualizadas).

Con la definición de varios esquemas dentro de una misma base de datos, las tablas utilizadas por varios módulos se agruparon en el esquema ***"com"***. En el módulo Gestión de Entrada y Salida de Documentos las tablas con esta condición son: ***nprovincia***, ***tbmunicipio***, ***tbpersona***, ***ntipofichero***, ***tbfichero***, y ***tbdatosfichero***; también se emplea la tabla ***tbcuadro*** del esquema ***"gcpa"***, por estos motivos y para evitar redundancias en la base de datos (dado que ya se encuentran en otro esquema) se propone su eliminación.

Eliminación de nomencladores:

Los nomencladores ***nsexo, nclasificaciondoc, norigendoc y ntipodoc*** cumplen con las condiciones de la propuesta de solución de nomencladores, motivo por el cual su contenido se incorporará a las tablas ***tbnomenclador*** y ***tbtiponomenclador***, y se propone su eliminación de la base de datos.

Eliminación de claves innecesarias:

La relación de muchos a muchos entre una misma entidad se traduce en una nueva tabla que tiene como identificador dos id de la entidad. Por este motivo se propone en la tabla ***tbdocgdficher (sustituida por tbdocumentofichero),*** la eliminación del identificador ***iddocgdfichero*** (serial). En este caso, el patrón de llave subrogada no está eficientemente empleado.

Creación de tablas:

Se propone la creación de las tablas: ***tbportador*** y ***tbhistorialdoc*** que reemplazan eficientemente las funciones de varias de las tablas eliminadas y proponen una solución óptima a los requisitos del sistema y las reglas del negocio (actualizadas).

Transformaciones de tablas existentes:

La tabla ***tbdocumentodoc*** se transforma en las tablas ***tbdocumento, tbdocumentopromovente*** y ***tbdocumentooficial,*** estas dos últimas surgen a partir de una especialización de la primera. Esta solución propuesta, sale de un riguroso análisis de los requisitos del sistema y las reglas del negocio (actualizadas).

Con todas estas transformaciones en el modelo GESD, la propuesta quedaría como se muestra en el Anexo 9.

2.6.5 Esquema IPP

El subsistema de PP tiene como objetivo principal la Informatización del proceso de la promoción y el ejercicio de la acción penal pública en representación del Estado. Garantiza la función de velar por el cumplimiento de la legalidad en los Establecimientos Penitenciarios, al controlar la situación jurídica de los acusados que procesa la Fiscalía. El subsistema de PP está integrado por los módulos de Índice de Peligrosidad Predelictiva (IPP), Ordinario (ORD) y Sumario (SUM).

- ✓ **Módulo: IPP**

El módulo de Índice de Peligrosidad Predelictiva tiene como objetivo principal la gestión de los expedientes de peligrosidad predelictiva que se le abren a pretensos asegurados, personas consideradas en estado peligroso, propensas a cometer delitos y que tienen una demostrada conducta contradictoria con las normas sociales y la moral socialista.

Para una mejor comprensión de las transformaciones realizadas que se expondrán a continuación del modelo IPP, ver el Anexo 10.

Principales Transformaciones

Eliminación de tablas innecesarias:

Las tablas ***tbdeclaracionpretensoasegurado, tbinformedti, tbinvestigacionpersona, tbinformejefesector, tbinformeconclusivojatc, tbdeclaracionvictimaotestigo, tbadvertenciasoficiales y tbcharlasprofilacticas*** según los requisitos del sistema y las reglas del negocio (actualizadas) no cumplen con ningún objetivo, motivo por el cual se propone su eliminación de la base de datos del sistema.

Con la definición de varios esquemas dentro de una misma base de datos, las tablas utilizadas por varios módulos se agruparon en el esquema ***"com"***. En el módulo Índice de Peligrosidad Predelictiva las tablas con esta condición son ***nnombreanno*** y ***tbunidadpnr***; también se emplea la tabla ***tbcuadro*** del esquema ***"gcpa"***, por estos motivos y para evitar redundancias en la base de datos (dado que ya se encuentran en otro esquema) se propone su eliminación.

Eliminación de nomencladores:

Los nomencladores ***nsexo, nclasificacionpretensoasegurado, nexpedientedecision, nsituacionprocesal, nestadofactura, norigendestino*** cumplen con las condiciones de la propuesta de solución de nomencladores, motivo por el cual su contenido se incorporará a las tablas ***tbnomenclador*** y ***tbtiponomenclador***, y se propone su eliminación de la base de datos.

Eliminación de claves innecesarias:

En las tablas ***tbm9***, ***tbm10, tbm11***, ***tbm12, tbactadetencion, tblibrocontrol, tbhistorialexp*** y ***tbexpediente_factura*** la clave foránea **tbexpedienteippnumexpedienteipp**, responde a la misma necesidad que la clave foránea **tbexpedienteippidexpedienteipp *(idexpedienteipp)***, por lo que en pos de minimizar el espacio de almacenamiento, se propone eliminar la primera (es de tipo **varchar**) y dejar la segunda (de tipo **integer**).

Eliminación de atributos innecesarios:

En la tabla ***tbexpedienteipp*** los atributos ***secexpediente***, ***secdocumentacion*** y ***secdocactual*** no cumplen objetivos según los requisitos y reglas del negocio, por lo cual se propone su eliminación.

En la tabla ***tbm9***, los atributos ***tbdenunciaiddenuncia y tbdenunciaidnodenuncia***, no cumplen ningún objetivo según los requisitos y reglas del negocio de Índice de Peligrosidad Predelictiva motivos por el cual se propone su eliminación. Estos atributos son empleados en el módulo de Sumario por lo que cada vez que se haga en IPP un ***m9***, siempre estos dos atributos serán nulos innecesariamente.

En la tabla ***tbm11***, el atributo ***nopresentescrito*** no cumple ningún objetivo según los requisitos y reglas del negocio, por el cual se propone su eliminación.

La tabla ***tbpretensoasegurado*** es una especialización de la tabla ***tbpersona*** por lo que los atributos: ***idpa***, ***ci***, ***nombre***, ***primerapellido***, ***segundoapellido*** y ***nsexoidsexo*** son redundantes en la misma, se propone su eliminación.

En la tabla ***tbindicacion*** el atributo ***idatestado*** no cumple objetivo según los requisitos y reglas del negocio, por lo cual se propone su eliminación.

En la tabla ***tblibrocontrol*** el atributo ***idacusado*** no cumple objetivo según los requisitos y reglas del negocio, por lo cual se propone su eliminación.

Partiendo de los requisitos del sistema y el análisis del negocio se propone la eliminación en la tabla ***tblibrocontrol*** del atributo: ***iddenuncia,*** el cual es redundante, pues ese valor está contenido en una tabla relacionada mediante la clave foránea ***idexpedienteipp*** procedente de la tabla ***tbexpedienteipp***.

La tabla ***tbactadetencion*** no requiere de los atributos: ***motivos, estado, iddenuncia y idnodenuncia*** según los requisitos del negocio (actualizado) y el diseño del sistema. De igual manera el atributo **hora** es redundante pues está contenido en el atributo ***fecha.*** Razón por la cual se propone su eliminación.

Cambio del tipo de atributo:

En la tabla ***tbm9*** se proponen los cambios en los atributos ***tribunaprovincial*** y ***tribunalmunicipal*** de **varchar** a **integer** y emplearlos como claves foráneas con motivo de minimizar espacio de almacenamiento y reutilizar información contenida en la base de datos, específicamente en las tablas ***tbnprovincia*** y ***tbnmunicipio***.

En la tabla ***tbexpedienteipp*** se propone el cambio en el atributo ***denuncia*** de **varchar** a **integer** y emplearlo como clave foránea con motivo de minimizar espacio de almacenamiento y reutilizar información contenida en la base de datos, específicamente en la tabla "***sum"."tbdenuncia".***

En la tabla ***tbindicacion*** se propone el cambio en el atributo ***usuario*** de **varchar** a **integer** y emplearlo como clave foránea con motivo de minimizar espacio de almacenamiento y reutilizar información contenida en la base de datos, específicamente en la tabla ***"com"."tbusuario".***

En la tabla ***tbhistorialexp*** se proponen los cambios en los atributos ***origen, destino***, ***estado*** y ***situacionprocesalacusado*** de **varchar** a **integer** y emplearlos como claves foráneas con motivo de

minimizar espacio de almacenamiento y reutilizar información contenida en la base de datos, específicamente en las tablas ***tbnomenclador*** y ***tbtiponomenclador***.

Creación de tablas:
Se propone la creación de la tabla ***tbficheroexpedienteipp*** con los identificadores de las tablas ***tbexpedienteipp***(***idexpedienteipp***)y ***"com"."tbdatosfichero"*** (***idfichero***), resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente.

Se propone la creación de la tabla ***tbunidadpnrcuadros*** con los identificadores de las tablas ***"com".tbunidadpnr*** (***idunidadpnr***) y ***"gcpa"."tbcuadro"*** (***idpersonas***), resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente.

Con todas estas transformaciones en el modelo IPP, la propuesta quedaría como se muestra en el Anexo 11.

2.6.6 Esquema ORD

El módulo de Ordinario tiene como objetivo principal la gestión de los expedientes que se le abren a las personas que le imputan delitos sancionables por más de un año de privación de libertad, multas mayores de 300 cuotas o ambos.

Para una mejor comprensión de las transformaciones realizadas que se expondrán a continuación del modelo ORD, ver el Anexo 12:

Principales Transformaciones

Eliminación de tablas innecesarias:
Se propone la eliminación de la tabla ***narticulo*** de la base de datos del módulo, pues según los requisitos del sistema y las reglas del negocio (actualizadas) no cumple objetivo alguno.

Con la definición de varios esquemas dentro de una misma base de datos, las tablas utilizadas por varios módulos se agruparon en el esquema ***"com"***. En el módulo Ordinario las tablas con esta condición son: ***nprovincia***, ***tbmunicipio***, ***tbunidadpnr***, ***tbpersona***, ***nnombreanno***, ***ntipofichero***, ***tbfichero***, ***tbdatosfichero*** y ***tbndelito***, también se emplean las tablas ***tbmedidacautelar*** y ***tbdocumentosvf*** del esquema ***"vf"*** y la tabla ***tbcuadro*** del esquema ***"gcpa"***, por estos motivos y para evitar redundancias en la base de datos (dado que ya se encuentran en otro esquema) se propone su eliminación.

Eliminación de nomencladores:

Se propone la eliminación de la tablas nomencladoras: ***nsexo, npais, netnia, nestadocivil, nvincualcionlaboral, nestadocivil, nivelescolaridad*** *y* ***nestadosalud,*** porque cumplen con las condiciones de la propuesta de solución de nomencladores, motivo por el cual su contenido se incorporará a las tablas ***tbnomenclador*** y ***tbtiponomenclador*** y se eliminarán de la base de datos.

Se propone la eliminación de la tabla nomencladora ***ncategoriadocente*** que en un momento determinado era utilizada, sin embargo, con la revisión y actualización de los requisitos del sistema, ya no cumple objetivo su permanencia en la base de datos.

Eliminación de claves innecesarias:

La tabla ***tbpersonanatural*** es una especialización de la tabla ***tbpersona*** por lo cual la clave primaria de la tabla sería ***idpersona*** y no ***idpersonanatural*** **(clave subrogada).** Debido a lo expuesto anteriormente se propone la eliminación de esta última clave o llave primaria.

La tabla ***tbacusadoord*** es una especialización de la tabla ***tbpersonanatural*** por lo cual la clave primaria de la tabla sería ***idpersona*** y no ***idacusadoord*** **(clave subrogada*).*** Debido a lo expuesto anteriormente se propone la eliminación de esta última clave o llave primaria.

Eliminación de atributos innecesarios:

Partiendo de las necesidades de los requisitos del sistema y de los cambios en el modelo del sistema se propone eliminar en la tabla ***tbexpedienteord*** los atributos ***municipio, provincia*** (son redundantes pues se pueden obtener mediante la relación con la tabla ***tbunidadpnr***), ***annodenuncia***, ***sancion***, ***indice*** e ***iddelito***.

El atributo ***idprovincia*** encontrado en las tablas ***tbexpedienteord***, ***tbp23***, ***tbp30a***, ***tbp30b*** y ***tbp30c***, se considera redundante, pues al ser cada una de estas tablas en el negocio una actuación y siendo las actuaciones documentos oficiales que se le hacen a un determinado expediente ordinario, no hay razón para volver a guardar el atributo, el mismo se puede obtener mediante la clave foránea ***idunidadpnr*** procedente de la tabla ***tbunidadpnr.*** Por tales razones se propone su eliminación de la base de datos.

El atributo ***idacusado*** encontrado en las tablas: ***tbpa1***, ***tbpa2***, ***tbpa3***, ***tbp5***, ***tbp6***, ***tbp7***, **tbp8**, ***tbp9***, ***tbp10***, ***tbp12***, ***tbp13***, ***tbp19***, ***tbp20***, ***tbp22***, ***tbp24***(ambas claves: ***idacusado1*** e ***idcusado2***), ***tbp25***, ***tbp28*** y ***tbp29***, no se corresponde con los cambios en el modelo del sistema y al ser redundante, se propone su eliminación.

El atributo ***acusados*** encontrado en las tablas: ***tbpa1***, ***tbpa2***, ***tbpa4***, ***tbpa5***, ***tbpa6***, ***tbp4***, ***tbp5***, ***tbp6***, ***tbp7***, ***tbp8***, ***tbp9***, ***tbp10***, ***tbp11***, ***tbp12***, ***tbp13***, ***tbp16***, ***tbp17***, ***tbp19***, ***tbp18***, ***tbp20***, ***tbp21***, ***tbp22***, ***tbp23***,

tbp24, ***tbp25***, ***tbp26***, ***tbp27***, ***tbp29***, ***tbp30b*** y ***tbp30c*** y el atributo ***idmedidacuatelar*** encontrado en las tablas: ***tbexpedienteord***, ***tbpa1***, ***tbp8*** (las dos llaves), ***tbp5***, ***tbp6*** y ***tbp9***, no se corresponden con los cambios en el modelo del sistema y al ser redundante, se propone su eliminación.

El atributo ***idunidadpnr*** presente en las tablas: ***tbei1*** y ***tbp4***, se considera redundante debido a que cada una de estas tablas en el negocio constituye una actuación. Siendo las actuaciones documentos oficiales que se le hacen a un determinado expediente ordinario, no hay razón para volver a guardar el atributo mencionado pues ya está contenido en la tabla ***tbexpedienteord***. Por tales razones se propone su eliminación de la base de datos.

En la tabla ***tbp7*** el atributo ***fechamedidacautelar*** no cumple objetivo según los requisitos y reglas del negocio, por lo cual se propone su eliminación.

En la tabla ***tbp5*** los atributos ***fechadisposicionfiscal***, ***horadisposicionfiscal***, ***horadetencion*** *y* ***fechadetencion*** carecen de funcionalidad en el sistema según los requisitos y reglas del negocio, por lo que se propone su eliminación.
En la tabla ***tbp16*** los atributos ***idarticulo***, ***tbdenunciaiddenuncia***, *y* ***tbdenunciaidnodenuncia*** carecen de funcionalidad en el módulo Ordinario según los requisitos y reglas del negocio motivos por el cual se propone su eliminación.

En la tabla ***tbpersonanatural*** el atributo ***edad*** es un atributo derivado, que se puede obtener del número del carnet de identidad, valor que se almacena en la llave primaria de la tabla (***idpersona***), por lo cual se propone su eliminación.

En la tabla ***tbpersonanatural*** el atributo ***dirección*** no cumple objetivo según los requisitos y reglas del negocio, por lo cual se propone su eliminación***.***

La tabla ***tbpersonanatural*** es una especialización de la tabla ***tbpersona*** por lo que los atributos: ***cidentidad***, ***nombre***, ***apellido1***, ***apellido2*** y ***idsexo*** son redundantes, se propone su eliminación.

En la tabla ***tbpa3*** los atributos ***cantidaddehojas***, ***idacusadooc*** *y* ***idacusadood*** carecen de funcionalidad en el módulo Ordinario según los requisitos y reglas del negocio, por lo cual se propone su eliminación.

En la tabla ***tbp18*** los atributos ***tbdenunciaiddenuncia***, *y* ***tbdenunciaidnodenuncia*** carecen de funcionalidad en el módulo Ordinario según los requisitos y reglas del negocio, por lo cual se propone su eliminación.

En la tabla ***tbp24*** el atributo ***nohojas*** no cumple objetivo según los requisitos y reglas del negocio, por lo cual se propone su eliminación.

Partiendo de las necesidades de los requisitos del sistema y de los cambios en el modelo del sistema, se propone eliminar en la tabla ***tbactuaciones*** el atributo: ***municipio,*** el cual es redundante, pues ese valor está contenido en una tabla relacionada mediante la clave foránea ***idfiscalia*** procedente de la tabla ***tbfiscalia***.

Cambio del tipo de atributo:

En la tabla ***tbpa4*** se proponen los cambios de tipo del atributo ***situacionprocesal*** de **varchar** a **integer** y emplearlo como clave foránea con motivo de minimizar espacio de almacenamiento y reutilizar información contenida en la base de datos, específicamente en la tabla ***tbtiponomenclador.***

En la tabla ***tbpa2*** se propone el cambio en el atributo ***seleccionado*** de **integer** a **boolean** con motivo de minimizar espacio de almacenamiento.

Creación de tablas:

Se propone la creación de la tabla ***tbdelitoexpord*** con los identificadores de las tablas ***tbexpedienteord*** (***idexpedienteord***) y ***tbndelito*** (***iddelito***). La tabla propuesta se crea como resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente. Esta propuesta presenta una solución óptima al manejo de todos los delitos que puedan ocurrir simultáneamente en un expediente ordinario.

Teniendo en cuenta las nuevas necesidades de los requisitos y de los cambios en el modelo del sistema, se propone la creación de la tabla ***tbp30d***. Esta almacenará toda la información del documento Recurso de Casación.

Agregación de atributos a una tabla:

Se propone la adición de los atributos ***fecharchivoprovisional*** (***date***) a la tabla ***tbei1, motivos*** (***varchar***) a la tabla ***tbp30b, preceptoautorizante2*** (***varchar***), ***preceptoinfringido*** (***varchar***) y ***preceptoinfringido*** (***varchar***) a la tabla ***tbp30c, resultadodeoficio*** (***varchar***) y ***fechaoficio*** (***date***) a la tabla ***tbp8*** ,***diligencia*** (***varchar***) a la tabla ***tbpa1, fechavencimiento*** (***date***) a la tabla ***tbp27*** , ***participacionacusado*** (***varchar***) y ***hechonarrado*** (***varchar***) a la tabla ***tbp29, idtipotribunal*** (***integer***), ***articulos*** (***varchar***) y ***documentalhojas*** (***varchar***) a la tabla ***tbpa3***,
iddelito (***integer***) a la tabla ***tbpa2*** y ***articulos*** (***varchar***) y ***documentalhojas*** (***varchar***) a la tabla ***tbpa3*** debido a que todos ellos son requeridos según los requisitos y reglas del negocio.

Con todas estas transformaciones en el modelo ORD, la propuesta quedaría como se muestra en el Anexo 13.

2.6.7 Esquema SUM

El módulo de Sumario tiene como objetivo principal la gestión de las denuncias imputadas a personas por delitos sancionables inferiores al año de privación de libertad, multas menores de 300 cuotas o ambos.

Para una mejor comprensión de las transformaciones realizadas que se expondrán a continuación del modelo SUM ver el Anexo 14.

Principales Transformaciones

Eliminación de tablas innecesarias:
Partiendo de los requisitos y el modelo del sistema se encontró un caso singular con respecto a la tabla ***tbactuaciones***. Esta tabla fue modelada como una generalización de las tablas: ***tbp3***, ***tbp14***, ***tbp16***, ***tbp18***, ***tbp18.1*** y ***tbp20***. Sin embargo, en el negocio, los atributos de la tabla mencionada no se corresponden con los atributos requeridos, para evitar que ante la creación de una actuación solo se guarde el identificador autoincrementable, mientras que el resto de los 7 atributos se queden nulos, se propone su eliminación de la base de datos.

Con la definición de varios esquemas dentro de una misma base de datos, las tablas utilizadas por varios módulos se agruparon en el esquema ***"com"***. En el módulo Sumario las tablas con esta condición son: ***nprovincia***, ***tbmunicipio, tbunidadpnr***, ***ntipofichero***, ***tbfichero***, ***tbdatosfichero***, ***ndelito*** y ***tbpersona***. Por este motivo y para evitar redundancias en la base de datos (dado que ya se encuentran en otro esquema) se propone su eliminación.

Eliminación de nomencladores:
Los nomencladores: ***nsexo, nsituacionprocesal, nestadofactura, norigendestino*** cumplen con las condiciones de la propuesta de solución de nomencladores, motivo por el cual su contenido se incorporará a las tablas ***tbnomenclador*** y ***tbtiponomenclador***, y se propone su eliminación de la base de datos.

El nomenclador ***narticulo*** no es necesario, pues el sistema esta modelado de tal forma que cuando se desee guardar el artículo referido, este es introducido por el usuario. Esto garantiza la propuesta de usar un atributo (***varchar***) y no una tabla aparte para referenciar a un artículo.

Eliminación de claves innecesarias:

En las tablas ***tbm1, tbm3, tbm4, tbm5***, ***tbm6***, ***tbm7***, ***tbm9***, ***tbp3***, ***tbp16***, ***tbp18***, ***tbp18.1***, ***tbp20***, ***tblibrocontroldenuncia, tbdenunciafactura, tbdenunciafichero, tbdenunciaacusado y tbdenunciadelito*** la clave foránea **tbdenunciaidnodenuncia**, responde a la misma necesidad que la clave foránea **tbdenunciaiddenuncia *(iddenuncia)***, por lo que en pos de minimizar el espacio de almacenamiento, se propone eliminar la primera (de tipo **varchar**) y dejar la segunda (de tipo **integer**).

La tabla ***tbacusado*** es una especialización de la tabla ***tbpersona,*** por lo cual la clave primaria de la tabla sería ***idpersona*** y no ***idacusado*** (clave subrogada) debido a lo expuesto se propone la eliminación de esta última clave o llave primaria.

Eliminación de atributos innecesarios:

En la tabla ***tbdenuncia*** los atributos **tbmunicipioidmunicipio, radicación, nsituacionprocesalidsituacionprocesal, horadetencion, secdenuncia, fechadetencion, nmedidacautelaridmedidacautelar, secdocactualdenuncia *y* secdocumentaciondenuncia**, con la actualización de las reglas del negocio y de los requisitos del sistema ya no son requeridos, razón por la cual se propone su eliminación de la base de datos.

En la tabla ***tbm9*** los atributos **tbexpedienteippidexpedienteipp *y* tbexpedienteippnumexpedienteipp**, no cumplen objetivo según los requisitos y reglas del negocio de Sumario, por lo cual se propone su eliminación. Estos atributos son empleados en el módulo de Índice de Peligrosidad Predelictiva pues cada vez que se haga en Sumario un ***"m9"***, estos serán nulos innecesariamente.

En la tabla ***tbindicacion*** el atributo ***idatestado*** no cumple objetivo según los requisitos y reglas del negocio por lo cual se propone su eliminación.

La tabla ***tbacusado*** es una especialización de la tabla ***tbpersona*** por lo cual los atributos ***nombre***, ***primerapellido*** y ***segundoapellido*** son redundantes en la misma, por tanto se propone su eliminación.

En la tabla ***tblibrocontroldenuncia*** el atributo ***acusado*** no cumple objetivo según los requisitos y reglas del negocio, por lo cual se propone su eliminación.

En la tabla ***tbp3*** los atributos ***instructor, no_denuncia_instruccion, causa_instruccion, iddelito*** y ***acusado*** carecen de funcionalidad en el sistema según sus requisitos y reglas del negocio, por lo cual se propone su eliminación.

En la tabla ***tbp14*** los atributos ***efp_denuncia*** y ***organoisntriccion_unidadpnr*** no cumplen objetivo según los requisitos y reglas del negocio por lo cual se propone su eliminación.

En la tabla ***tbp16*** los atributos ***tblimiteidlimite, idarticulo, iddelito, acusados*** y ***jefefiscal*** carecen de funcionalidad en el sistema según sus requisitos y reglas del negocio motivos por el cual se propone su eliminación.

En la tabla ***tbp20*** los atributos ***fecha_rebeldía, idacusado*** y ***acusado*** carecen de funcionalidad en el sistema según sus requisitos y reglas del negocio motivos por el cual se propone su eliminación.

Los atributos ***iddelito*** e ***idmunicipio*** de la tabla ***tbm6***, ***fecha1*** y ***fecha2*** de la tabla ***tbm7***, ***tbfechaidfecha, iddelito*** y ***tbmunicipioidmunicipio*** de la tabla ***tbm5 ,documentoiddocumento*** y ***numeroactuaciones*** de la tabla ***tbm4, iddelito*** de la tabla ***tbm3, caso, fechasobreseguimiento*** y ***acusado*** de la tabla ***tbp18 y idarticulo y acusados*** de la tabla ***tbp18.1*** son redundantes, pues no son requeridos según los requisitos y reglas del negocio por lo cual se propone su eliminación.

Cambio del tipo de atributo:

En la tabla ***tbindicacion*** se propone cambiar el tipo de dato en el atributo ***usuario*** de **varchar** a **integer** y emplearlo como clave foránea con motivo de minimizar el espacio de almacenamiento y reutilizar información contenida en la base de datos, específicamente en la tabla ***"com"."tbusuario".***

En la tabla ***tbm9*** se propone cambiar el tipo de dato en los atributos ***tribunaprovincial*** y ***tribunalmunicipal*** de **varchar** a **integer** y emplearlos como claves foráneas con motivo de minimizar el espacio de almacenamiento y reutilizar información contenida en la base de datos, específicamente en las tablas ***tbnprovincia*** y ***tbnmunicipio***.

Creación de tablas:

Se proponen la creación de la tabla ***tbm2***, la cual hace referencia a uno de los requisitos incorporados recientemente en la base de datos.

Se propone la creación de la tabla ***tbm1opciones,*** pues en los nuevos requisitos incorporados, el ***tbm1*** requería de seis atributos (seis diferentes opciones) adicionales, pero en la mayoría de los casos no se empleaban todos, por este motivo, para evitar la concurrencia de atributos nulos, se define una nueva tabla en la que se registre solo aquellas opciones empleadas en cada documento ***m1***.

Se propone la creación de la tabla ***tbunidadpnrcuadros*** con los identificadores de las tablas ***"com". "tbunidadpnr"*** (***idunidadpnr***) y ***"gcpa". "tbcuadro"*** (***idpersonas***), resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente.

Agregación de atributos a una tabla:

Se propone la adición de los atributos ***remision*** (***varchar***) a la tabla ***tbp14, caso*** (***varchar***) y ***fecha*** (***date***) a la tabla ***tbp18.1, razones*** (***varchar***) a la tabla ***tbp3*** y ***fecha*** (***date***) a la tabla ***tbp16,*** debido a que los requisitos y reglas del negocio requieren de su existencia.

Con todas estas transformaciones en el modelo SUM, la propuesta quedaría como se muestra en el Anexo 15.

2.6.8 Esquema VF

El subsistema de Verificación Fiscal tiene como objetivo principal la informatización del proceso llevado a cabo en las Verificaciones Fiscales, incluyendo la gestión y actualización de los archivos con informaciones preliminares.

Para una mejor comprensión de las transformaciones realizadas que se expondrán a continuación del modelo VF, ver el Anexo 16:

Principales transformaciones

Eliminación de tablas innecesarias:

La tabla ***tb3beneficiario*** no se requiere al agregar el atributo ***tercerbeneficiario*** a la tabla ***tbenriquecido,*** que aprovechando el resto de los atributos resuelve la misma funcionalidad según los requisitos y reglas del negocio (actualizadas). Por los motivos anteriores se propone su eliminación de la base de datos.

Con la definición de varios esquemas dentro de una misma base de datos, las tablas utilizadas por varios módulos se agruparon en el esquema ***"com"***. En el módulo Verificación Fiscal las tablas con esta condición son ***nprovincia***, ***tbmunicipio***, ***tbpersona***, ***nnombreannosvf***, ***ntipofichero***, ***tbfichero***, ***tbdatosfichero*** y ***tbfiscalias***; también se emplea la tabla ***tbcuadro*** del esquema ***"gcpa"***, por estos motivos y para evitar redundancias en la base de datos (dado que ya se encuentran en otro esquema) se propone su eliminación.

Eliminación de nomencladores innecesarios:

Se propone la eliminación de las tablas nomencladores ***nsexo, npais, netnia, nestadocivil, nvincualcionlaboral, nestadocivil, nivelescolaridad, nestadosalud*** *y* ***ncategoriadocente,*** estas cumplen con las condiciones de la propuesta de solución de nomencladores, por lo cual su contenido se incorporará a las tablas ***tbnomenclador*** y ***tbtiponomenclador***, y se propone su eliminación de la base de datos.

Se propone la eliminación de la tabla nomencladora ***ncategoriadocente*** que fue utilizada en algún momento, después haber realizado una revisión y actualización a los requisitos del sistema.

Eliminación de claves innecesarias:

La tabla ***tbpersonanatural*** es una especialización de la tabla ***tbpersona*** por lo que su llave primaria debe ser ***idpersona*** (la misma que su tabla genérica ***tbpersona***)***,*** debido a esto la clave autoincremental definida (***idpersonanatural***) no es requerida y por tanto se propone su eliminación. En este caso no está bien empleado el patrón de clave o llave subrogada.

Eliminación de atributos innecesarios:

En las tablas ***tbpretenconfisca, tbresolucioncontrapersonasjuridicas, tbresolregistrodomiciliario, tbresolarchivo, tbresolucionsinefectomedidacautelar, tbresolmodmedidacautelar, tbactacomparecencia, tbescritohechodelictivo, tbaplicacioprocespecif, tbresultregristro*** y ***tbmedidacautelar,*** la existencia de un atributo para contener el identificador del fiscal provincial es redundante, debido a que estas tablas son una especialización de la tabla ***tbdocumentosvf*** que, entre otros atributos, tiene el identificador de la provincia. Conociendo que solo un fiscal desempeña la función de fiscal provincial por cada provincia, entonces, sabiendo la provincia se puede obtener todos los datos del fiscal responsable con solo una consulta. Por los motivos antes expuestos se propone que sean eliminados.

En las tablas ***tbactacomparecencia***, ***tbactadevoluciondoc***, ***tbactaentrega***, ***tbactaocupacion***, ***tbactarecepciondoc***, ***tbactaremisionocupacion***, ***tbactarequerimiento***, ***tbaplicacioprocespecif***, ***tbavaluo***, ***tbcierre***, ***tbcitacionoficial***, ***tbdenegacionrevision***, ***tbdevolucionactuaciones***, ***tbdevolucionexpconfisprov***, ***tbdictamenpericial***, ***tbescritohechodelictivo***, ***tblevantbienes***, ***tbmedidacautelar***, ***tbpretenconfisca***, ***tbpromoviendosolrevision***, ***tbprorroga***, ***tbresolarchivo***, ***tbresolcontrapersjuridicas***, ***tbresolmodmedidacautelar***, ***tbresolregistrodomiciliario***, ***tbresolucionsinefectomedidacautelar***, ***tbresolucionviolatoria*** y ***tbresultregristro***, la existencia de un atributo que define el identificador del fiscal que interviene en la redacción de los documentos, es redundante, debido a que estas tablas son una especialización de la tabla ***tbdocumentosvf*** que, entre otros atributos, contiene el identificador como llave foránea de la tabla "***gcpa***"."***cuadro***" Por los motivos antes expuestos se propone su eliminación de todas estas tablas.

El atributo ***idprovincia*** se encuentra en las tablas ***tbactarequerimiento, tbprorroga, tbresolucionsinefectomedidacautelar, tbpretenconfisca, tbresolregistrodomiciliario, tbresolmodmedidacautelar, tbresolarchivo, tbresolucioncontrapersonasjuridicas, tbescritohechodelictivo, tbdevolucionexpconfisprov, tbresolucionviolatoria, tbavaluo,***

tbaplicacioprocespecif, tblevantbienes, tbresultregristro, tbdictamenpericial, tbmedidacautelar, tbdevolucionactuaciones y ***tbcierre,*** estas a su vez, responden a especializaciones de la tabla ***tbdocumentosvf,*** por tal motivo se propone eliminar el atributo mencionado de las tablas especializadas y agregarlo en la tabla genérica.

En la tabla ***tbresolucionsinefectomedidacautelar*** el atributo ***funcionario*** es redundante, debido a que no se requiere según los requisitos y reglas del negocio. Por este motivo se propone su eliminación.

En la tabla ***tbresolregistrodomiciliario*** los atributos: ***testigo1, direcciontestigo1, testigo2 y direcciontestigo2,*** no contribuyen al adecuado manejo de la información relacionada con los testigos por parte de la resolución de registro domiciliario, por este motivo se propone su eliminación.

Los atributos ***noresolucion, nombre, dirección y ci*** de la tabla ***tbresolmodmedidacautelar, noresolucion*** de la tabla ***tbresolucioncontrapersonasjuridicas, anno*** de la tabla ***tbescritohechodelictivo, ciudad*** de la tabla ***tbresolarchivo, annotramitacionexpediente de*** la tabla ***tbresolucionviolatoria, annotramitacionexpediente*** de la tabla ***tbresolucionviolatoria, anno*** y ***patrimonio*** de la tabla ***tbavaluo*** y ***numresolucion, identidad*** y ***anno*** de la tabla ***tbaplicacioprocespecif,*** son redundantes, debido a que no se requieren según los requisitos y reglas del negocio. Por este motivo se propone su eliminación

En la tabla ***tbcitacionoficial*** el atributo ***direccionfiscalia*** es redundante debido a que este atributo está contenido en la tabla ***tbfiscalia***, tabla con la que se relaciona mediante la clave foránea ***idfiscalia***, por tal motivo se propone su eliminación.

En la tabla ***tbmedidacautelar*** los atributos ***cuidad, idpersonacustodia, noresolucion*** y ***anno*** son redundantes debido a que no son requeridos según los requisitos y reglas del negocio. Motivo por el cual se propone su eliminación.

En la tabla ***tbperito*** los atributos ***licencia*** y ***cargo*** son redundantes debido a que no son requeridos según los requisitos y reglas del negocio. Motivo por el cual se propone su eliminación.

La tabla ***tbpersonanatural*** es una especialización de la tabla ***tbpersona*** por lo que los atributos: ***cidentidad***, ***nombre***, ***apellido1***, ***apellido2*** y ***idsexo*** son redundantes, por tanto se propone su eliminación.

Los atributos ***idrepresentante*** de la tabla ***tbentidad, numeroexp*** de la tabla ***tbexpedientepreliminar*** y ***hora*** de la tabla ***tbdevolucionactuaciones*** son redundantes debido a que no son requeridos según los requisitos y reglas del negocio, por lo cual se propone su eliminación.

Cambio del tipo de atributo:

En la tabla ***tbavaluo*** se propone cambiar el tipo de dato en el atributo ***montoascendente*** de ***integer*** a ***real*** para cumplir su objetivo y evitar la posible pérdida de valores cuando el monto asciende a un valor real en vez de entero.

Creación de tablas:

Se propone la creación de la tabla ***tbtestigovf*** con los identificadores de las tablas ***tbpersona***(***idcipersona***) y ***tbresolregistrodomiciliario*** (***idresolregistrodomiciliario***), resultado de la relación de muchos a muchos entre las tablas mencionadas anteriormente. Esta propuesta presenta una solución óptima al manejo de los testigos por parte de la resolución de registro domiciliario.

Se propone la creación de las tablas: ***tbactaentrega, tbactaocupacion, tbactarecepciondoc*** y ***tbactaremisionocupacion,*** las cuales responden a nuevos requisitos solicitados por los clientes y definidos recientemente en el negocio.

Agregación de atributos a una tabla:

Se propone la adición del atributo ***idprovincia*** (***integer***) a la tabla ***tbdocumentosvf*** debido a que todas las tablas especializadas de la misma lo requieren, se considera que es óptimo contener este atributo en la tabla genérica y no en cada una de las tablas especializadas.

Se propone la adición del atributo ***fechanotificacion*** (***date***) a la tabla ***tbresolregistrodomiciliario*** pues según los requisitos y reglas del negocio este es requerido.

Se propone la adición del atributo ***idfiscal*** (***char 11***) a la tabla ***tbdocumentosvf*** debido a que todas las tablas especializadas de la misma lo requieren, se considera que es óptimo contener este atributo en la tabla genérica y no en cada una de las tablas especializadas.

Se propone la adición de los atributos ***idinformacion*** (***integer***) y ***tercerbeneficiario*** (***boolean***) a la tabla ***tbenriquecido*** pues según los requisitos y reglas del negocio son requeridos.

Con todas estas transformaciones en el modelo VF la propuesta quedaría como se muestra en el Anexo 17.

De todas las trasformaciones, las más complejas de aplicar son las propuestas sobre el diseño, debido al número de cambios identificados y su repercusión directa en el modelo de la base de datos y en el sistema. Las propuestas realizadas al módulo DJ (esquema HCTA) y a los esquemas CLEP, IPP, SUM, ORD y VF, son consideradas como viables y su implementación oscila entre la baja y la media complejidad. Es importante mencionar que poco más del 60% de estas propuestas ya han sido ejecutadas, principalmente las relacionadas con la eliminación y agregación de tablas y atributos a los modelos.

Aunque el módulo de GESD no es complejo su diseño es incorrecto y las propuestas realizadas al mismo son pequeñas pero muy significativas por esas razones la ejecución de las transformaciones es de media complejidad. En el caso del esquema GCPA, existen muchas entidades que fueron eliminadas o transformadas al no cumplir objetivo en el sistema o al ser incorporados a la solución propuesta para el tratamiento de los nomencladores, por lo que se puede decir que su implementación oscila entre la media y la alta complejidad.

Tabla 4: Resumen

PROPUESTA DE INDICADORES	TOTAL
Tablas eliminadas	54
Tablas reutilizadas	58
Nomencladores eliminados	23
Nomencladores incorporados a la solución	59
Tablas agregadas	41
Cantidad de tablas resultantes	222
Atributos eliminados	331
Atributos agregados	44
Total	832

2.7 Propuesta de Índice

Los índices pueden ser creados utilizando una o más columnas de la base de datos, proporcionando las bases para un acceso rápido a los registros. La indexación es una técnica que optimiza el acceso a los datos, mejora el rendimiento acelerando las consultas y otras operaciones. Generalmente, su utilidad aumenta mientras mayor número de registros existan.

Con respecto al tema del uso de índices, uno de los problemas existentes lo constituye el hecho de que todos los que están definidos en la actual base de datos son solo aquellos que por defecto agrega el SGBD. Lo anterior muestra que durante el análisis, la creación y el mantenimiento de la base de datos no

se desarrolló un estudio sobre la implementación y resultados de la utilización de índices asociados a atributos (que no sean llaves foráneas) o relaciones para mejorar el rendimiento del sistema.

El tipo de índice más común en la base de datos optimizada continúa siendo el de **Primary Key**. Con respecto a las claves primarias empleadas, es significativo decir que el número de claves semánticas es muy reducido en comparación con los de tipo numérico que sean autoincrementables, de manera que se favorece el trabajo de los índices. En todos los casos, las llaves primarias indexadas, por defecto, son generados por el SGBD y sus valores son únicos y no nulos. La cantidad de índices de este tipo, asciende a un total de 222.

Una vez realizado el análisis del negocio, el modelo lógico y físico, se identificaron atributos que, sin estar definidos como llaves primarias, no permitían la existencia de valores repetidos. Estos atributos, con valores únicos, traen implícitamente asociados la creación de índices de tipo Unique. Ellos son:

Tabla 5: índices

MÓDULO	TABLA	ATRIBUTO	ÍNDICE AGREGADO
COM	usuario	usuario	usuario_unique
ORD	expediente	noefp	noefp_unique

Los índices de tipo **Index**, son índices que pueden estar asociados a una o varias columnas de la tabla, pero con la característica de que aceptan valores nulos y repetidos. Estos también juegan un papel importante en la disminución de los tiempos de consultas y transacciones, cuando se usan correctamente. Para el empleo apropiado de los índices de tipo **Index,** específicamente de los índices por valor, se realizó un estudio exhaustivo sobre un grupo de columnas que podrían estar asociadas con los mismos en las tablas de cada uno de los esquemas. Estas columnas se caracterizaban por utilizarse frecuentemente en transacciones y consultas, contener un amplio rango de valores distintos entre sí y devolver cantidades relativamente pequeñas de valores o filas en la ejecución de tales consultas. Finalmente, se realizaron pruebas para ver los tiempos de respuestas de las consultas y a partir de los resultados se definieron los índices a implementar. Ellos son:

Tabla 6: Índices Index

MÓDULO	TABLA	ATRIBUTO	ÍNDICE AGREGADO
CLEP	registroinspeccion	idcentroclep	registroinspeccion_ idcentroclep_index
COM	historial	idpersona	historial_persona_index
GCPA	capacitacioncuadro	idpersona	capacitacioncuadro_persona_index
IPP	expediente	idtipoclasificacionpa	expediente_clasificacionpa_index

ORD	actuaciones	idexpediente	actuaciones_expediente_index
SUM	denuncia	idexpedienteord	denuncia_ idexpedienteord_index
VF	informacion	numeroexpediente	informacion_ numeroexpediente_index

La implementación de todos los índices propuestos resulta una tarea sencilla pues la gestión de los mismos es totalmente independiente al modelo de la aplicación.

2.7.1 Resultados de las pruebas de los índices

El tipo de índice utilizado en todos los casos fue el **Btree**. Aunque es el que por defecto define PostgreSQL, no ha sido esa la razón que ha motivado su empleo en cada uno de los casos, sino los resultados de pruebas con los índices del tipo: **Hash**, **GIST** y **GIN**, en los que cuales, los resultados obtenidos no mostraron mejorías con respecto a los tiempos de respuestas.

Características generales*:*
Número de tuplas por tablas: 1000.
Cantidad de test realizados por índices: 10.

Tabla 7: Significado de los indicadores de pruebas

Cost	Intervalo de tiempo en el que recoge (A..B): A: Costo inicial estimado (tiempo invertido antes de que el inicio del escaneo, por ejemplo, el tiempo para realizar la ordenación en un nodo de tipo). B: Coste total estimado.
Rows	Número estimado de filas de salida.
Width	Ancho medio estimado (en bytes) de salida de las filas.
Actual time	Intervalo real de tiempo (milisegundos) empleados en la consulta.
Rows	Cantidad de filas devueltas.
Loops	Número de ciclos realizado en la consulta.
Total runtime (X)	Media del tiempo de ejecución de la consulta.

Tabla 8: índice (registroinspeccion_ idcentroclep_index).

	(Cost; Rows; Width)	***(Actual time; Rows; Loops)***	***Total runtime ($\bar{X}$)***
Resultados previos	(0.00..39.50; 10; 96)	(0.030..0.813; 10; 1)	0.933 ms
Resultados posteriores	(0.00..4.33; 10; 0)	(0.040..0.040; 10; 1)	0.227 ms

Tabla 9: Índice (historial_persona_index).

	(Cost; Rows; Width)	***(Actual time; Rows; Loops)***	***Total runtime ($\bar{X}$)***
Resultados previos	(0.00..26.50; 32; 4)	(0.047..0.822; 32; 1)	0.979 ms
Resultados posteriores	(0.00..4.49; 32; 0)	(0.092..0.092; 32; 1)	0.381 ms

Tabla 10: índice (capacitacioncuadro_persona_index).

	(Cost; Rows; Width)	***(Actual time; Rows; Loops)***	***Total runtime ($\bar{X}$)***
Resultados previos	(0.00..29.91; 24; 4)	(0.088..1.096; 24; 1)	1.229 ms
Resultados posteriores	(0.00..4.43; 24; 4)	(0.089..0.089; 24; 1)	0.365 ms

Tabla 11: Índice (expediente_clasificacionpa_index).

	(Cost; Rows; Width)	***(Actual time; Rows; Loops)***	***Total runtime ($\bar{X}$)***
Resultados previos	(0.00..24.50; 5; 55)	(0.182..0.663; 5; 1)	0.758 ms
Resultados posteriores	(0.00..4.27; 5; 0)	(0.036..0.036; 5;1)	0.181 ms

Tabla 12: índice (actuaciones_expediente_index).

	(Cost; Rows; Width)	***(Actual time; Rows; Loops)***	***Total runtime ($\bar{X}$)***
Resultados previos	(0.00..28.48; 10; 4)	(0.041..0.660; 10; 1)	0.816 ms
Resultados posteriores	(0.00..4.33; 10; 0)	(0.046..0.046; 10; 1)	0.235 ms

Tabla 13: índice (denuncia_ idexpedienteord_index).

	(Cost; Rows; Width)	***(Actual time; Rows; Loops)***	***Total runtime ($\bar{X}$)***
Resultados previos	(0.00..30.74; 1; 301)	(0.306..0.398; 1; 1)	0.498 ms
Resultados posteriores	(0.00..8.27; 1; 301)	(0.036..0.040; 1; 1)	0.124 ms

Tabla 14: índice (informacion_ numeroexpediente_index).

	(Cost; Rows; Width)	***(Actual time; Rows; Loops)***	***Total runtime ($\bar{X}$)***
Resultados previos	(0.00..45.49; 1; 201)	(0.913..0.917; 1; 1)	1.027 ms
Resultados posteriores	(0.00..8.27; 1; 201)	(0.058..0.061; 1; 1)	0.123 ms

Conclusiones Parciales

Con la propuesta de optimización de la base de datos que se obtuvo en la realización de este capítulo, se logró minimizar las redundancias presentes evitando la existencia de tablas y atributos repetidos. Se logró reducir la cantidad de tablas en la base de datos pues de 325 existentes actualmente en SIGEF, se redujeron a 222 tablas. El empleo de los índices propuestos permitió tener un mejor tiempo de respuestas de las consultas realizadas lo que demostró la necesidad de su existencia. Por las razones expresadas anteriormente se puede afirmar que ha sido cumplido el segundo objetivo específico planteado en la investigación.

.

CAPÍTULO III: VALIDACIÓN DE LA PROPUESTA

3.1 Introducción

En el presente capítulo se realizará la validación de la propuesta de solución ofrecida en el desarrollo del trabajo, teniendo en cuenta aspectos teóricos y funcionales. Se utilizaron diferentes herramientas para verificar los tiempos de respuestas de las consultas y el rendimiento de la aplicación. Además se describirán los elementos que se tuvieron en cuenta para lograr que el diseño fuera óptimo.

3.2 Validación de la teoría del diseño

El acertado diseño de una base de datos es un punto crucial en el desarrollo de un sistema, por lo cual se deben tener presentes una serie de puntos elementales que garanticen la eficiencia de la misma.

Los aspectos que se van a tratar en este epígrafe tienen que ver con la exactitud, consistencia, confiabilidad de la información, la privacidad y confidencialidad de los datos, la integridad, el análisis de redundancia, análisis de seguridad y la normalización. Todo esto garantiza el acceso a los datos por el personal autorizado, la calidad de la información procesada y almacenada, así como la protección de la misma ante la ocurrencia de acciones no controladas.

3.2.1 Integridad

Las pruebas de integridad en una base de datos verifican que los datos sean exactos, completos, coherentes y autorizados, indican si hay errores en los controles de entrada o procesamiento. La tabla muestra el resultado de algunos aspectos a comprobar.

Tabla 15*:* Integridad de los datos

No.	Aspectos a comprobar	Cumplido
1.	Se modifica y se eliminan los datos en las tablas tal como se especifica en la funcionalidad	si
2.	¿Cuando se guarda un determinado conjunto en la base de datos, se guarda cada valor completo? (En otras palabras, no se produce el truncamiento de cadenas y el redondeo de los valores numéricos.)	si
3.	Los valores predeterminados se guardan en la base de datos cuando no se ha especificado la entrada del usuario	si

La integridad se basa en un conjunto de reglas que utilizan la mayoría de las bases de datos relacionales. A continuación se detalla cada una de las reglas utilizadas en la propuesta desarrollada en este trabajo:

- ✓ *Integridad de entidad:*

Este tipo de prueba es relacionada con las claves o llaves primarias de las tablas pues cada entidad debe poseer una clave principal y es aquí donde se comprueba la existencia de estas, además dicta que ningún atributo que forme parte de la llave primaria puede aceptar valores nulos.

En la propuesta de solución que se brinda, las llaves primarias han sido definidas en cada tabla y ninguna clave identificada toma valor nulo o repetido. Esto se asegura a través del uso de restricciones PRIMARY KEY permitiendo garantizar que no se tendrá información repetida o discordante para un valor de clave y puede ser usada como control, para evitar la inclusión de información inconsistente en las tablas.

- ✓ *Integridad referencial:*

Gracias a la integridad referencial se garantiza que una entidad (fila o registro) siempre se relaciona con otras entidades válidas (existentes en la base de datos). Implica que en todo momento dichos datos sean correctos, sin repeticiones innecesarias, sin datos perdidos ni relaciones mal resueltas.
En la propuesta de solución esta regla de integridad se evidencia, pues para cada valor de clave foránea existe un valor de clave primaria concordante.

3.2.2 Normalización de la solución

En la propuesta de solución brindada uno de los aspectos más importantes tenidos en cuenta durante el estudio, el análisis y la transformación del mismo fue la normalización del diseño hasta su tercera forma normal, siempre que beneficiara el proceso de informatización del negocio. Con respecto a la normalización se encontraron varias deficiencias y se propusieron soluciones a las mismas. A continuación se muestra algunos ejemplos:

Figura 7: Ejemplo 1

En la tabla ***public.tbm1***, se puede apreciar la existencia de varios atributos identificados como opciones (del 1 al 6 consecutivamente). Estos son posibles opciones que se pueden redactar en un documento M1, sin embargo, en la generalidad de los casos no se redactarán las seis por lo que puede esperarse un

número significativo de tuplas nulas con la inserción de nuevos datos. Dado el caso, se propone eliminar el grupo de opciones (grupo repetitivo) de la tabla individual, crear una tabla separada por cada grupo de datos relacionados e identificar cada grupo de datos relacionado con una clave primaria.

La propuesta quedaría concretada como se muestra a continuación:

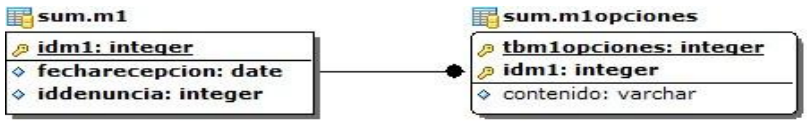

Figura 8: Tabla normalizada.

En el conjunto de tablas que se muestran a continuación se evidencia la existencia de atributos que no dependen de la clave primaria y son redundantes, pues se guardan en cada una de ellas valores de atributos almacenados en una tabla de la cual se tiene su identificador como llave foránea. Por ejemplo:

- ✓ En la tabla ***tbexpedienteord*** conociendo la unidad de la PNR relacionada con el expediente, no es necesario volver a almacenar el municipio, ni la provincia de la unidad debido a que ambos valores se pueden obtener conociéndose el identificador de la misma.
- ✓ En la tabla ***tbactuaciones*** conociendo la fiscalía donde se redacta la actuación, no es necesario volver a almacenar el municipio donde se encuentra la fiscalía, pues el valor se puede obtener conociéndose el identificador de la misma.
- ✓ En la tabla ***tbacusado*** es redundante almacenar los atributos: ***nombre, primerapellidos***, ***segundoapellido*** y ***ci***, pues todos ellos están contenidos en la tabla (persona) de la cual ***tbacusado*** es una especialización.

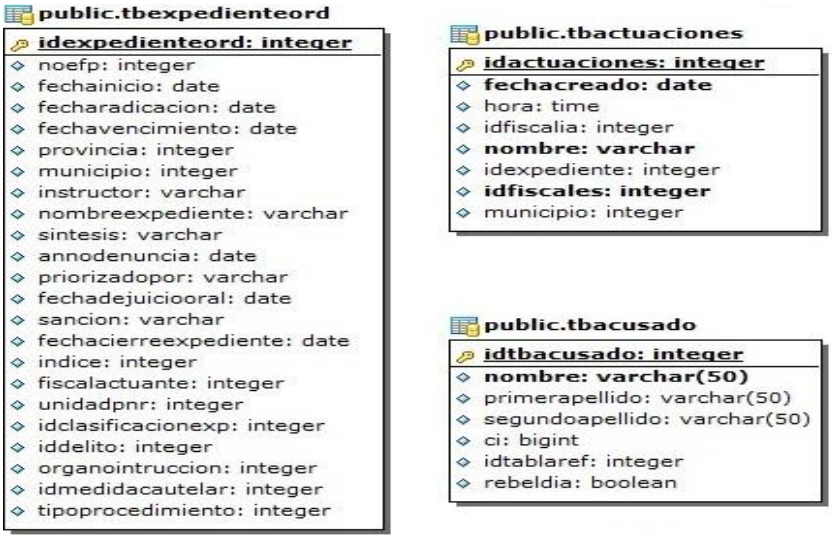

Figura 9: Ejemplo 2

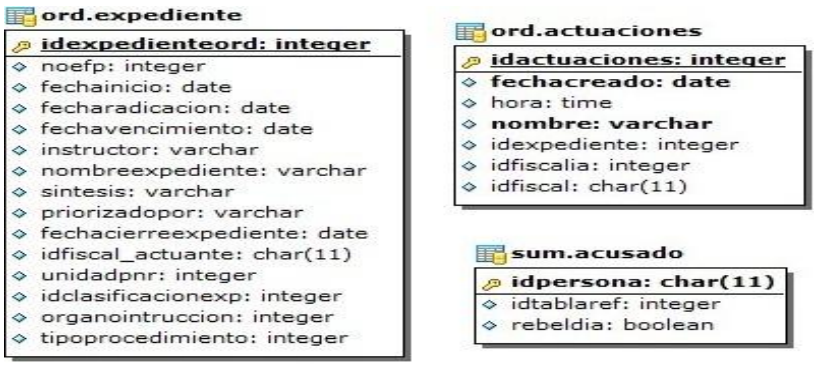

Figura 10: Tablas Normalizadas

Como resultado de la normalización realizada se obtuvo una propuesta de diseño libre de anomalías en la actualización y mejora de la independencia de los datos, mayor rapidez en la ordenación y en la creación de índices, menor número de valores NULL y menos oportunidades para generar incoherencias. De esta forma, se aumentó el rendimiento de la base de datos.

3.2.3 Análisis de redundancia de la información

La redundancia de la información es un hecho que se evidencia mucho en la base de datos que actualmente existe en SIGEF y es uno de los problemas que presenta; de ahí la propuesta planteada. Esta no es más que la repetición innecesaria de información en la base de datos. Con el trabajo realizado se eliminaron esas inconsistencias, ejemplo de esto es la eliminación de atributos, tablas e identificadores innecesarios en el modelo. En las figuras 9 y 10 mostradas en el acápite anterior, se muestran ejemplos de eliminación de redundancias mediante la normalización de la BD.

3.2.4 Análisis de la seguridad de la base de datos

En nuestros días las bases de datos son componentes esenciales para almacenar todo tipo de información, debido a que estas pueden ser considerablemente sensibles o secretas, por eso se debe considerar seriamente la forma de protegerlas. La seguridad en las base de datos son acciones a tomar para garantizar que la información almacenada sea consultada solo por los usuarios autorizados, además para asegurar la recuperación de la información en caso de que ocurra alguna anomalía.

En la propuesta brindada se definió una serie de reglas que garantizan la seguridad de la información. Estas son:

- ✓ Se le realizan copias de respaldo diariamente a la BD.

- ✓ Se encuentra instalado y configurado los servidores de seguridad.
- ✓ Los passwords pueden ser alterados solo por el Administrador de la base de datos.
- ✓ La BD registra todos los intentos de acción, tanto los exitosos como los infructuosos.
- ✓ Todo intento de conexión con la BD se registra.
- ✓ Los passwords deberían al menos contener un carácter no alfabético.

3.3 Validación Funcional

Luego de realizar la validación teórica de la base de datos, se pasa a la parte más importante que es la validación funcional. Es en esta validación donde se muestran algunos resultados obtenidos con respecto al rendimiento y al correcto funcionamiento de la misma. Además se mostrará la comparación de los tiempos de respuestas de las consultas y otros aspectos entre la actual BD y la propuesta. Las pruebas que se realizarán tienen como objetivo simular una carga de producción real y observar cómo se comportan ante esta carga. De este modo se comprueba cual de las bases de datos satisface mejor los requisitos del sistema sin violar la integridad de los datos.

3.3.1 Prueba de rendimiento

Las pruebas de rendimiento son muy importantes en el desarrollo de una aplicación, pues de ellas se asume, si la aplicación funciona, es estable y sólida. Las pruebas de rendimiento pueden servir para diferentes propósitos:

- ✓ Pueden demostrar que el sistema cumple los criterios de rendimiento.
- ✓ Pueden comparar dos sistemas para encontrar cuál de ellos funciona mejor.
- ✓ Pueden asegurar el posterior funcionamiento del sistema y garantizar un correcto y eficiente servicio, carente de fallas, errores y retrasos a sus usuarios.
- ✓ Pueden medir qué partes del sistema o de carga de trabajo provocan que el conjunto rinda mal.

Es importante tener presente que los resultados de las pruebas y el rendimiento del sistema está estrictamente ligado al hardware de la computadora que realizará la función de servidor.

A continuación se presentan algunas de estas pruebas de rendimiento, para demostrar con más claridad la factibilidad de la solución brindada.

3.3.1.1 Prueba de Volumen

Este es el tipo más sencillo de pruebas de rendimiento, aquí se analiza el comportamiento de la BD verificando si la misma alcanza su límite de almacenamiento y puede causar fallas. Para realizar esta

prueba en la propuesta de solución, se utilizó la herramienta Data Generator 2005 para PostgreSQL para el llenado voluminoso de la BD.

En las siguientes tablas se muestra el comportamiento del llenado de las tablas escogidas de la base de dato actual de SIGEF y de la optimizada

Tabla 16: Base de Datos actual

Tabla	Número de tuplas (insertadas)	Tiempo	Errores
public.tbdocumentoclep	20000	13 min 01 seg	0
public.tbpersona	20000	22 min 44 seg	0
public.tbexpedienteord	20000	58 min 13 seg	0
public.tbdenuncia	20000	44 min 23 seg	0
public.documentosvf	20000	10 min 33 seg	0

Tabla 17: Base de Datos Optimizada

Tabla	Número de tuplas (insertadas)	Tiempo	Errores
clep.documento	20000	8 min 15 seg	0
com.persona	20000	4 min 26 seg	0
ord.expediente	20000	20 min 04 seg	0
sum.denuncia	20000	17 min 03 seg	0
vf.documento	20000	16 min 42 seg	0

Al introducir los datos en las tablas de la base de datos no se presentaron problemas de límite de capacidad. La utilización de esta herramienta permitió verificar la integridad de los datos, además como se puede apreciar en la base de datos actual se generaron las mismas cantidades de tuplas que en la optimizada en mayor tiempo, esto demuestra que la propuesta brindada responde con mejor eficiencia y soporta el cúmulo de información requerida para el funcionamiento de la BD en menor tiempo.

En el caso de ***vf.documento*** tiene un mayor tiempo que la anterior, pues a la misma durante la optimización del diseño se le agregaron atributos lo que representa un incremento del tiempo de carga.

3.3.1.2 Prueba de Stress

Este tipo de prueba se realiza para determinar la solidez de la aplicación en los momentos de carga extrema y ayuda a los administradores a determinar si la aplicación rendirá lo suficiente en caso de que la

carga real supere a la carga esperada. Otra razón por la cual se realiza esta prueba, es para comprobar y comparar un grupo importante de indicadores estadísticos que garanticen cuantitativamente la mejoría, o no, de la propuesta de optimización de la BD de SIGEF I. Esta prueba puede mostrar los tiempos de respuesta de todas las transacciones importantes de la aplicación. Para ello se utilizará un software de distribución gratuita: JMeter, que es una herramienta Java dentro del proyecto Jakarta y permite realizar Pruebas de Rendimiento y Pruebas Funcionales sobre Aplicaciones Web. Es una herramienta de carga para llevar a cabo simulaciones sobre cualquier recurso software, esta se destaca por su versatilidad, estabilidad y por ser de uso gratuito.

Se realizaron un conjunto de seis pruebas a las bases de datos. Las tres primeras a la base de datos de SIGEF I y las otras tres a la base de datos optimizada que se propone. A continuación se muestran las propiedades de los hilos de cada una de las pruebas, así como el significado y valor definido para cada uno de ellos.

- ✓ **Número de hilos**: Número de usuarios a simular.
- ✓ **Período de subida (en segundos)**: Tiempo que debiera llevarle a JMeter lanzar todos los hilos (si se seleccionan 10 hilos y el periodo de subida es de 1 segundo, entonces cada hilo comenzará 0,1 segundo después de que el hilo anterior haya sido lanzado).
- ✓ **Contador del bucle**: Número de veces a realizar la prueba.

Tabla 18: Pruebas

Propiedades de los hilos	Prueba 1	Prueba 2	Prueba 3
Número de hilos	10	20	40
Período de subida (en segundos)	1	1	1
Contador del bucle	5	5	5

En los informes de cada una de las pruebas se muestran los siguientes indicadores:

- ✓ **Label**: El nombre de la muestra (conjunto de muestras).
- ✓ **# Muestras**: El número de muestras de peticiones JDBC.
- ✓ **Media**: El tiempo medio transcurrido para un conjunto de resultados.
- ✓ **Mediana**: Mediana aritmética[5].
- ✓ **Desviación**: Desviación Estándar.[6]

[5] Elemento de una serie ordenada de valores crecientes de forma que la divide en dos partes iguales, superiores e inferiores a él.

[6] Diferencia entre la medida de una magnitud y el valor de referencia.

- ✓ **Mín.**: El mínimo tiempo transcurrido para las muestras de peticiones JDBC.
- ✓ **Máx.**: El máximo tiempo transcurrido para las muestras de peticiones JDBC.
- ✓ **Error %**: Porcentaje de las peticiones con errores.
- ✓ **Rendimiento**: Rendimiento medido en base a peticiones por segundo/minuto/hora.
- ✓ **Kb/seg**: Rendimiento medido en Kilobytes por segundo.

Tabla 19: Resultados

	BD de SIGEF I			**BD optimizada en la propuesta**		
Label	Petición JDBC	Petición JDBC	Petición JDBC	Petición JDBC	Petición JDBC	Petición JDBC
No. Muestras	1000	1000	1000	1000	1000	1000
Media	55	347	666	11	136	438
Mediana	28	136	152	15	35	79
Desviación	65	381	891	12	193	669
Mín	18	19	19	0	7	7
Máx	563	1999	4282	171	1042	2797
% Error	0	0	0	0	0	0
Rendimiento	25,7/seg	29,9/seg	36,0/seg	23,2/seg	43,3/seg	53,7/seg
Kb/seg	674,6	785,1	946	16	30,1	36,9

A partir del Anexo 18, se muestran los resultados de las pruebas realizadas con la herramienta Jmeter expuestos en la tabla 19.

Para cada una de las seis pruebas realizadas el número de muestras ascendió a 1000 peticiones JDBC. En el análisis de la media (uno de los indicadores más importantes por lo que representa), el tiempo promedio transcurrido para la obtención del conjunto de resultados de las pruebas realizadas en la propuesta, fueron en los tres casos inferiores a los de las pruebas homólogas en la base de datos de SIGEF I, arrojando datos interesantes, como es el caso de la prueba 1, donde la media obtenida en la base de datos optimizada constituía solo el 20% de la media de la prueba homóloga.

Con respecto a la mediana aritmética y la desviación estándar (significativos indicadores estadísticos), el primero constituye el valor que ocupa el lugar central cuando el conjunto de resultados está ordenado de menor a mayor mientras que el segundo es una medida que permite determinar el promedio aritmético de fluctuación de los datos respecto a la media, se puede decir que las cifras obtenidas en la BD optimizada son mejores en todos los casos que las obtenidas en la BD de SIGEF I. De igual manera ocurre con los valores encontrados en los tiempos mínimos y máximos transcurridos para las muestras de peticiones

JDBC. Un elemento destacable para ambas bases de datos es el hecho que en las seis pruebas desarrolladas ninguna de las dos mostró alguna incidencia de error en sus peticiones.

Finalmente cuando analizamos los indicadores de rendimiento (Rendimiento medido en base a peticiones por segundo y rendimiento medido en Kilobytes por segundo), a excepción de la primera prueba (10 hilos) de rendimiento de peticiones por segundo en la que la BD de SIGEF I que muestra un resultado ligeramente mayor (25,7/seg > 23,2/seg), en el resto de las pruebas los mejores números los exhiben los de la base de datos optimizada e incluso con diferencias más notables. Un aspecto a destacar es que en las pruebas a medida que aumenta el número de hilos o el número de usuarios a simular, las diferencias de los dos indicadores de rendimientos entre ambas bases de datos aumenta también y en este caso siempre favorece a la base de datos optimizada que se propone en el presente trabajo de diploma.

Conclusiones Parciales

En este capítulo se analizó un conjunto de criterios a tener en cuenta para validar la propuesta de solución brindada que le servirá al proyecto de SIGEF en su segunda fase. Las pruebas realizadas demostraron la disminución de los tiempos de respuestas y el mejoramiento del rendimiento. Con la culminación de este capítulo se le dio cumplimiento al tercer objetivo específico trazado en la investigación.

CONCLUSIONES GENERALES

A partir del estudio realizado durante la investigación, se estudiaron un grupo de elementos conceptuales que permitieron a los autores apropiarse de los conocimientos necesarios en el área de la optimización y diseño de base de datos relacionales, lo cual propició elaborar la Fundamentación teórica del presente trabajo.

A partir de las técnicas de diseño y los mecanismos de optimización, se obtuvo una propuesta de solución que permite garantizar el rendimiento eficiente de la aplicación.

La propuesta de optimización elaborada fue validada mediante la realización de pruebas carga- volumen y prueba de stress, demostrando que los resultados obtenidos tributan al rendimiento eficiente de la BD del proyecto SIGEF.

Con la propuesta de optimización, se garantizó la integridad de los datos, seguridad, confidencialidad, normalización, disminuir los tiempos de respuestas, minimizar el espacio de almacenamiento y el mantenimiento de la base de datos.

Como conclusión general, puede afirmarse que se le dio cumplimiento a todos los objetivos planteados al inicio del trabajo y se verificó la validez de la idea a defender materializada en la solución propuesta.

RECOMENDACIONES

Luego de la investigación realizada en el presente trabajo y teniendo en cuenta las ideas que surgieron durante el progreso de la misma, además de que las metas trazadas fueron cumplidas satisfactoriamente se recomienda:

- ✓ Aplicar las propuestas realizadas y demostradas en el desarrollo de SIGEF II y de otros proyectos con semejantes características en el Centro de Gobierno Electrónico.
- ✓ Continuar el desarrollo de las potencialidades de la solución propuesta para el eficiente manejo de nomencladores de la base de datos.

BIBLIOGRAFÍAS

Referencias bibliográficas

1. **C.J.Date.** Introducción a los Sistemas de Bases de Datos. Habana : Félix Varela, 2003.

2. **Yunta, L. R.** Bases de Datos documentales: estructura y uso. Madrid : CINDOC, 2001.

3. **Codd.** [En línea] María Mercedes Marqués André, 12 de febrero de 2001. [Citado el: 20 de marzo de 2011.] http://www3.uji.es/~mmarques/f47/apun/node1.html.

4. **CALI, SANTIAGO DE.** Base de Datos: Conceptualización y sistemas de administración - 221560. [En línea] 12 de FEBRERO de 2011. http://www.buenastareas.com/ensayos/Bases-De-Datos-Conceptos-Basico/1536159.html.

5. **J., HYDE.** Base de datos. [En línea] 2002. [Citado el: 22 de marzo de 2011.] http://www.monografias.com/trabajos11/basda/basda.shtml.

6. **Pérez Valdés, Damián.** ¿Qué son las bases de datos? [En línea] 2007. [Citado el: 22 de marzo de 2011.] http://www.maestrosdelweb.com/principiantes/%C2%BFque-son-las-bases-de-datos/.

7. **C.J.Date.** Sistemas de Base de datos. La habana : Felix Varela, 2003.

8. **G.W. Hansen, J. H.** Diseño y Administración de Bases de Datos. Prentice Hall : s.n., 1997.

9. **Hernández, M. J.** Database Design for Mere Mortals. Addison-Wesley Developers Press. : s.n., 1997.

10. **Mariluz Hernández Perdomo, Enrique José González Fernández.** Tesi"Guía para la optimización de servidores de bases de datos de PostgreSQL". ciudad Habana : UCI, 2008-2009.

11. **Elmasri and Navathe.** Optimización,Fundamentos de Sistema de Bases de datos,. [En línea] 2007. http://asignaturas.inf.udec.cl/~basedato/files/optimizacion.pdf..

12. **Medina, J. M.** Administración Postgres. Valencia : s.n., 2006.

13. **Andrés, María Mercedes Marqués.** [En línea] 12 de febrero de 2001. http://www3.uji.es/~mmarques/f47/apun/node93.html.

14. **Tardieu,Codd,.** Diseño de Base de datos relacional. [En línea] 2010. [Citado el: 30 de marzo de 2011.] http://usuarios.multimania.es/cursosgbd/UD4.htm.

15. **Diego, Flores Condori Roger.** El álgebra relacional como soporte teórico de la verificación. La Paz, Bolivia : s.n., 2006.

16. **S., Korth Henry F. – Silberschatz Abraham – Sudarshan.** Fundamentos de Bases de Datos. España : Cuarta, 2002.

17. **Velthuis, Adoración de Miguel Castaño – Mario Piattini.** Diseño de bases de datos. México, D. F : ALFAOMEGA GRUPO EDITOR, 2002.

18. **Andrés, María Mercedes Marqués.** Normalización. [En línea] 12 de 2 de 2001. [Citado el: 1 de abril de 2011.] http://www3.uji.es/~mmarques/f47/apun/node90.html.

19. **Lauro Soto, Ensenada.** Formas Normales Bases Datos. [En línea] http://www.mitecnologico.com/Main/FormasNormalesBasesDatos.

20. **Gregory Smith.** PostgreSQL 9.0. [En línea] 2010. www.wowebook.com.

21. **Castro, Rogelio S. Silverio.** Elementos de análisis de algoritmos. Cuba : s.n.

22. **Group.** The PostgreSQL Global Development. California : PostgreSQL 8.3.0 Documentation, 1996-2008.

23. **Jasuncionez.** Aplicacion Con Acceso a Bases de Datos Java MySql y Patrones de Diseño Parte 1. [En línea] 5 de agosto de 2010. http://www.microsoft.com/aplicacion_con_acceso_bases_de_datos_java_mysql_y_patrones_de_diseno_parte_1.htm.

24. Patterns of Data Modeling. **Blaha, Michael.** Atlanta, Georgia, U.S.A. : Sham Navathe, 2010.

25. **J Med Internet Res.** Genéricas de diseño web basado en bases de datos clínicos . Entidad-Atributo-Valor de Diseño. [En línea] Jacob Anhøj, MD, DIT, 4 de noviembre de 2003. http://viaclinica.com/article.php?pmc_id=1550574.

26. **Guillermo Hector Rodriguez.** MSDN. [En línea] Microsoft. Reservados todos los derechos., abril de 12 de 2011. [Citado el: 30 de mayo de 2011.] http://msdn.microsoft.com/es-es/library/ms184276.aspx

Bibliografías Consultadas

- ✓ **Berkus., Josh.** Toolbox for IT. Toolbox for IT. [En línea] 1 de mayo de 2009. http://it.toolbox.com/blogs/database-soup/writing-maintainable-queries-part-ii-29120.
- ✓ **CALI, SANTIAGO DE.** Base de Datos: Conceptualización y sistemas de administración - 221560. [En línea] 12 de FEBRERO de 2011. http://www.buenastareas.com/ensayos/Bases-De-Datos-Conceptos-Basico/1536159.html.
- ✓ **C.J.Date.** *Sistemas de Base de datos.* La habana : Felix Varela, 2003.
- ✓ **Hernández Castellanos, Y., & Pérez García**, W. Diseño de Bases de Datos para la Intranet 2. Ciudad de La Habana: Universidad de las Ciencias Informáticas, 2007.
- ✓ **Menéndez, Dr. Eugenio Santos.** Ejemplos de Optimización de Consultas. [En línea] 1 de 12 de 08. http://bd.eui.upm.es/DYOBD/Ejemplos_optimizacion_consultas_09.pdf
- ✓ **Marc Gibert Ginestà, Oscar Pérez Mora.** Bases de datos en PostgreSQL. s.l. : UOC. P06/M2109/02152.
- ✓ **M, Wenceslao Palma.** Almacenamiento y Recuperación de la Información. . 2005.
- ✓ **Normalización de bases de datos**. [En línea] mysql - hispano.org, 29 de de mayo de 2003. . http://usuarios.lycos.es/sanjudas/download/Normalizacion%20Base%20de
- ✓ **Perez, Manuel Bollain. 09.** Asignatura Diseño y Optimización de Bases de Datos. [En línea] 22 de 11 de 09. http://bd.eui.upm.es/DYOBD/bd22.html.
- ✓ **Powell, Gavin.** Beginning Database Design: . s.l. : Inc Wiley Publishing, 2006.
- ✓ **Raghu Ramakrishnan, Johannes Gehrke.** Database Management System. s.l. : Segunda Edición.
- ✓ **Ramakrishnan, Gehrke.** Fundamentos de Bases de Datos. Procesamiento y Optimización. . s.l. : CSI-INCO, 2008.
- ✓ **Reisel González Pérez.** Introducción al Sistema de Gestión de Base de Datos PostgreSQL. Ciudad de La Habana: Universidad de las Ciencias Informáticas: s.n., 2008.

- ✓ **SOFIA, Albert Anibal Osiris y HEREDIA, Raúl.** Diagrama Entidad-Relación. [En línea] http://170.210.92.6/osofia/$Bdd/Practicas/DERw.pdf.
- ✓ **S., Korth Henry F. – Silberschatz Abraham – Sudarshan.** Fundamentos de Bases de Datos. España : Cuarta, 2002.
- ✓ **Zorrilla Pantaleón**. Modelos de datos. M. E, 2009.