Universidad de las Ciencias Informáticas

Facultad 3

Título: ¨Diseño y configuración de la base de datos para el procedimiento Ordinario de la materia Civil en los Tribunales Municipales Cubanos.¨

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

Autor: Juan Carlos Larrinaga Ulacia.

Tutor: Ing. Alain Osorio Rodríguez.

Asesor: Ing. Elsydania López Guerra.

.

Ciudad de La Habana, Junio 2011.

DECLARACIÓN DE AUTORÍA

Declaro ser autor de la presente tesis y se le reconoce a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Juan Carlos Larrinaga Ulacia

_____

Firma del Autor

Ing. Alain Osorio Rodríguez

_____

Firma del Tutor

Ing. Elsydania López Guerra

_____

Firma del Autor

DATOS DE CONTACTO

Tutor: Ing. Alain Osorio Rodríguez.

Formación académica: Ingeniero en Ciencias Informáticas.

Centro laboral: Universidad de las Ciencias Informáticas (UCI)

Profesión:

Correo electrónico: aorodriguez@uci.cu

Dirección: Carretera San Antonio de los Baños, Torrens, Municipio Boyeros, Ciudad de La Habana, Cuba, Código postal 19370.

AGRADECIMIENTOS

Antes que nada agradecer a mis padres, mi mamá por estar a mi lado toda una vida, apoyarme y confiar en mí en todo momento, a mi papá que aunque no esté aquí, fue el motor que me motivo a estar hoy donde estoy.

A mis amigos, los del barrio, Lázaro, Yoandry, Ricardo, Jorge Luis que aunque la vida nos puso en caminos diferentes seguimos como hermanos. Los de aquí que durante estos cinco años se han ganado más que mi amistad, los vivieron conmigo y los que conocí con el paso del tiempo.

A mi familia que se preocupó día a día cuando tenía algún problema o simplemente por saber de mí, no solo la de sangre sino a las personas que por verme salir adelante me dieron todo.

A Alain Eduardo Rodríguez Arias que siempre me apoyo y más que un profe es un amigo.

A mi asesora Elsydania que siempre estuvo pendiente de mi desarrollo en todos los sentidos.

A todos los profesores de mi proyecto que me ayudaron n la elaboración de este trabajo.

Dedicatoria

A mi papá por definir el hombre que soy.

A mi mamá por hacerme saber con sus acciones diarias que lo soy todo para ella.

RESUMEN

La información es uno de los recursos fundamentales para cualquier organización, por ello se necesita de un manejo eficiente de la misma. El almacenamiento de los datos ha evolucionado desde el papel hasta los Sistemas de Bases de Datos actuales, facilitando el acceso a la información. El Tribunal Supremo Popular de Cuba encaminado a mejorar sus servicios, en colaboración con la Universidad de las Ciencias Informáticas (UCI), se encuentran inmersos en la tarea de automatizar estos procesos, con el desarrollo de un sistema que permita incrementar la eficiencia y flexibilidad de las tareas que allí se ejecutan. Una parte fundamental para el buen desempeño de este sistema es la base de dato, que permite el manejo de la información que se gestiona en cada uno de sus secciones, en específico la referente a lo Civil.

El desarrollo del diseño estuvo guiado por las actividades establecidas en la metodología utilizada, así como las herramientas, modelos y patrones definidos por la dirección del proyecto.
Se realizó el correspondiente modelado de los datos así como las configuraciones del entorno de desarrollo de manera general. Este diseño fue validado mediante pruebas tanto teóricas (normalización, integridad), como funcionales (volumen y rendimiento) al modelo de datos de la solución propuesta, además de tomar en cuenta una serie de medidas que garantizaron la seguridad de la información almacenada según el diseño realizado.

Índice

TABLA DE CONTENIDO

TABLA DE ILUSTRACIONES

ÍNDICE DE TABLAS

INTRODUCCIÓN

El uso generalizado de las Tecnologías de la Información y las Comunicaciones (TIC) ha conllevado a cambios que alcanzan todos los ámbitos de la actividad humana, al punto de provocar enormes transformaciones sociales debidas en gran medida al desarrollo científico-técnico.

Cuba se enfrenta hoy al reto de informatizar la sociedad, proyecto que se ha realizado de manera acelerada auspiciado por la máxima dirección del estado, al igual que ha puesto interés en lograr la accesibilidad a las tecnologías de la información y las comunicaciones, para convertir al país en paradigma de la sociedad de la información y el conocimiento para todos. Lograrlo implica entre otros factores la utilización de sistemas informáticos que realicen las actividades de manera flexible, para optimizar e integrar el flujo interno de información y tomar las mejores decisiones sobre los presentes y futuros desafíos de cada institución, de manera que se gane en productividad, calidad y servicio al cliente.

La justicia como uno de los principales sectores de la sociedad cubana no está exento a todo el proceso de informatización que se lleva a cabo, muestra de esto es la continua búsqueda de soluciones en varias de sus áreas, entre las que se puede mencionar el Tribunal Popular Cubano, órgano que tiene entre sus propósitos contribuir notablemente a la seguridad jurídica del país y a la realización plena de los derechos.

Los Tribunales están estructurados por tres instancias: municipal, provincial y supremo. En cada uno de ellos se tratan diferentes materias: Administrativa, Económica, Laboral, Penal y Civil. La materia Civil al igual que las demás, comprende varios procesos para lograr un trabajo más organizado: Ordinario, Sumarios, Amparo, Divorcio, Alimentos, Incidentes en Sumario de Alimentos, Incidentes en Divorcio, Procesos Sucesorios y de Jurisdicción Voluntaria.

El Tribunal Supremo Popular de Cuba consiente de la necesidad de continuar desarrollando la infraestructura tecnológica del sistema de tribunales populares y encaminado a mejorar los servicios del mismo, en colaboración con la Universidad de las Ciencias Informáticas (UCI) afronta la tarea de automatizar estos procesos, con el desarrollo de un sistema que permita incrementar la eficiencia y flexibilidad de los procesos que se ejecutan en los Tribunales.

En la actualidad todas las resoluciones, citaciones, documentos de comunicación y registro de datos en libros por parte de los jueces y secretarios se realizan de forma manual, al igual que el control de los

términos procesales de todos los expedientes. Debido a esta situación los jueces y secretarios se encuentran sobrecargados de trabajo lo cual provoca demoras en la tramitación de los procesos e incumplimiento de los términos procesales.

Además los libros y expedientes tienden a deteriorarse con el tiempo, y las necesarias búsquedas de información en estos es lenta y engorrosa debido a su gran volumen. La información estadística también se obtiene de forma manual, lo cual imposibilita que sea totalmente fiel a la realidad y que se obtenga de forma rápida. Un aspecto importante es que no se reutiliza la información existente, fundamentalmente en la creación de los documentos de los expedientes, dígase resoluciones, diligencias, citaciones y notificaciones, pues cada vez que se genera un documento hay que escribir nuevamente informaciones que con anterioridad fueron registradas.

Dada esta **situación problémica** surge el siguiente **problema de la investigación**: ¿Cómo minimizar la descentralización, duplicidad y deterioro de la información en la sección de lo Civil de los Tribunales Populares Cubanos, de modo que contribuya a la gestión rápida y eficiente de sus procesos?

Consecuentemente el **objeto de estudio:** sistemas informáticos para la gestión judicial.

Constituye el **campo de acción**: bases de datos para el sistema de tribunales.

El **objetivo general** del trabajo de diploma es: Diseñar y configurar la base de datos para el procedimiento Ordinario de la materia Civil de los Tribunales Populares Cubanos de modo que contribuya a la gestión eficiente de sus procesos.

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

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

La investigación está basada en la **idea a defender**: con el diseño y configuración de la base de datos para la materia civil, habrá una fuente de datos que permitirá una gestión eficiente de sus procesos.

Para la realización de la investigación se hizo necesario aplicar algunos métodos científicos teóricos, que tienen su sustento en la concepción materialista dialéctica y facilitan la recopilación de la información necesaria para el estudio de las base de datos en los sistemas jurídicos.

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

- El método Analítico – Sintético permitió realizar un análisis de la documentación de cada una de las herramientas que se utilizaron en el sistema a diseñar.

- El método Modelación se utilizó en la modelación de diagramas teniendo en cuenta las herramientas seleccionadas para el desarrollo del sistema.

El presente documento se estructura en: resumen, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos.

Capítulo 1: se realiza un estudio de las características de bases de datos así como de algunos sistemas de base de datos existentes. También se analizan los principales sistemas gestores de base de datos así como algunas herramientas Case[1].

Capítulo 2: se describe el entorno de desarrollo, modelo de datos, las configuraciones, así como algunas funcionalidades.

Capítulo 3: se describen los resultados alcanzados luego de aplicar diferentes métricas utilizadas para validar la solución del sistema.

[1] Computer Aided Software Engineering, Ingeniería de Software Asistido por Computadora.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo brinda una breve reseña del tema de las Bases de Datos en la actualidad para las instituciones jurídicas en Cuba y el mundo, las tendencias tecnológicas de las mismas y las metodologías de diseño con sus fases. Se abordan también temas como la seguridad, integridad de la información y normalización. Además se especificarán las herramientas empleadas para diseño y configuración de la base de datos para el procedimiento Ordinario de la materia Civil en los Tribunales Municipales Cubanos.

Conceptos fundamentales

E-gov. (Gobierno Electrónico): Para Gartner Group. "... es una innovación continua de los servicios, la participación de los ciudadanos y la forma de gobernar mediante la transformación de las relaciones externas e internas a través de la tecnología, el Internet y los nuevos medios de comunicación" (1), es decir que incluye todas aquellas actividades basadas en las modernas tecnologías informáticas, en particular Internet, que el Estado desarrolla para aumentar la eficiencia de la gestión pública, mejorar los servicios ofrecidos a los ciudadanos y proveer a las acciones del gobierno de un marco mucho más transparente que el actual.

Entonces el Gobierno Electrónico se refiere al uso por parte de las agencias gubernamentales de las TIC en función de mejorar la atención al público y acercar más a los ciudadanos a la gestión de sus problemas legales y engloba por lo menos los siguientes elementos:

- Está relacionado con la aplicación de las TIC.
- Implica innovación en las relaciones internas y externas del gobierno (Otras agencias gubernamentales, sus propios empleados, las empresas y/o el ciudadano. (2)

La Informática Jurídica: No es más que la interrelación entre las materias informáticas y derecho que tiene como fin el análisis, la estructura lógica y ordenada. (3)

Es la aplicación de técnicas informáticas a la documentación jurídica en los aspectos sobre el análisis, archivo y recuperación de información contenida en la legislación, jurisprudencia, doctrina o cualquier otro

documento con contenido jurídico relevante; en consecuencia la ciencia de la información abarca además de otras aéreas, la aplicación de técnicas documentales, entendida la documentación como el acto de reunir documentos sobre un tema dado y el tratamiento de estos en vista a su difusión. (4)

La informática jurídica de gestión: Conocida bajo las denominaciones de Ofimática (oficina automática o electrónica) o Burótica, expresiones que se traducen de la expresión inglesa “Office Automation” (Automatización de oficinas), es aquella en la cual se busca la automatización de tareas rutinarias propias de las oficinas, en estos casos judiciales, notariales, entre otros. (5)

Trata de la realización a través de soportes informáticos o telemáticos de operaciones destinadas a recibir y transmitir comunicaciones de cualquier tipo, de leer y escribir textos; de formar, organizar y actualizar archivos y registros; exigir y recibir pagos; estipular condiciones y controlar su cumplimiento.

Como manifestáramos anteriormente, la informática de gestión se ha aplicado a las siguientes áreas, Gestión: registral, notarial y en estudios jurídicos, judiciales y parlamentarios. (5)

Bases de datos jurídicas

Las bases de datos jurídicas intentan compendiar la normativa que emana de las distintas administraciones. Son un instrumento de búsqueda que facilita el acceso a la información.
El estudio de los distintos textos jurídicos precisa que exista una relación entre los mismos que actualizan, complementan, reforman y amplían cada norma. Las bases de datos permiten “navegar” desde una norma hasta otras a las que afecta o que le afectan, consultar la jurisprudencia asociada, entre otras funcionalidades. (6)
La idiosincrasia de la documentación jurídica determina ciertas particularidades de estas bases de datos, entre las que conviene destacar las siguientes:

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

b) Emplean distintas unidades documentales: texto completo del documento, extractos, zonas consideradas especialmente significativas como artículos de leyes, fundamentos jurídicos de sentencias.
c) Contienen, generalmente, un elevado volumen de documentos y, en consecuencia han ido creciendo inevitablemente en el mercado productos temáticos específicos.
d) Recogen documentos de tipología diversa, atendiendo a su vigencia. Aunque autores como López Muñiz Goñi remarcan la imposibilidad de prescindir de documentos jurídicos bajo el pretexto de su antigüedad, se observa, de un lado, la inclusión en las bases de datos de herramientas que permiten al usuario definir la consulta sobre la totalidad de los registros o sólo sobre los vigentes y, de otro, una tendencia a lanzar productos que recogen exclusivamente legislación vigente o consolidada.
e) Necesitan una actualización "puntual" de la información. La última ley promulgada es la aplicable a partir de su entrada en vigor o, en la documentación judicial, sentencias más recientes pueden manifestar un cambio jurisprudencial significativo.
f) Requieren exhaustividad en la documentación almacenada en orden a garantizar seguridad jurídica a sus usuarios.
g) Se hallan delimitadas por aspectos territoriales y jurisdiccionales. La aplicabilidad tanto de la legislación como de la jurisprudencia queda circunscrita al ordenamiento jurídico específico en el que se encuentran enmarcadas.
h) Precisan establecer referencias cruzadas para resolver las complejas interconexiones normativas -documentos anteriores y posteriores que complementan, modifican, amplían, derogan o interpretan una disposición-, las relaciones entre resoluciones judiciales que establecen o rompen una línea jurisprudencial, así como interrelaciones entre legislación, jurisprudencia y doctrina.

1.1 Sistemas existentes vinculados al campo de la informática jurídica

1.1.1 Sistemas de Gestión Jurídica nivel a internacional

En los últimos años, los increíbles progresos realizados en el campo de la informática han llevado a que su utilización se considere la solución ideal para resolver los problemas de que adolece el sistema de justicia. Estos sistemas actúan sobre la base de la tecnología de los certificados digitales permitiendo el uso de la firma digital y garantizando la autenticidad de los documentos que se envían.

A continuación se ofrece una breve descripción de algunos de estos sistemas.

Infolex

Software de Gestión Jurídica (España). Este software es realizado por la empresa Jurisoft la cual es líder en el sector de la Informática Jurídica. Nace en 1995 con una clara vocación de Servicio Integral, para cubrir todas las necesidades surgidas en la actividad diaria del Profesional del Derecho: Bases de Datos Documentales, Bibliografía, Infraestructura Informática, Servicios de Internet, Software de Gestión, Seguridad informática. Es compatible con todos los sistemas operativos Microsoft Windows perteneciente a las versiones XP –recomendado-, 2000, 2003, VISTA, en cualquiera de sus ediciones. Además con Microsoft SQL Server 2000, 2005 o superior (recomendado en clientes con gran volumen de datos –más de 10.000 expedientes-), Oracle 9.i o superior. Se puede optar por una versión de uso libre de SQL Server (Microsoft SQL Express 2005 suministrada con la instalación) o por otras versiones de bases de datos SQL de pago. Las versiones de pago Microsoft SQL Server 2000, 2005 (y Oracle 9i) y superior requieren las correspondientes licencias de Servidor y cliente. Microsoft SQL Express 2005 es un motor de datos gratuito Cliente-Servidor basado en SQL Server 2005 que se incluye con la licencia de Office. Las limitaciones fundamentales de SQL Express frente a SQL Server son:

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

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

Soft Class para Abogados. (Barcelona, España)

Es un sistema integral de gestión para bufetes de abogados y despachos o asesorías jurídicas que permite llevar un completo control de todas las tareas relacionadas con los mismos, además es un software adaptable a otros ámbitos. La principal función de este software es la administración de expedientes, con varios apartados en el que se gestionan independiente o conjuntamente los datos que lo conforman. El programa funciona con una base de datos en formato MS Access (programa, utilizado en

los sistemas operativos Microsoft Windows, para la gestión de bases de datos creado y modificado por Microsoft y orientado a ser usado en entorno personal o en pequeñas organizaciones). Eso da garantía de poder recuperar los datos en cualquier momento, y de poder efectuar exportaciones. Del mismo modo, pueden leer los datos de la base de datos para efectuar una combinación o realizar una selección de registros.

1.1.2 Sistemas de Gestión Jurídica a nivel nacional

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

Varios intentos de informatización de los Tribunales Populares Cubanos se han llevado a cabo en el país, entre ellos:

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

De forma general estos programas presentan las siguientes deficiencias:

- No ofrecen reportes estadísticos.
- Nula comunicación entre instancias.
- No superan las barreras del papel.
- Arraigados a los registros tradicionales (libros).
- Utilización de software propietario.
- Se puede hacer cualquier acción sobre el expediente sin tener en cuenta el estado del mismo.

Sistema de Gestión Fiscal

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

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

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

Como resultado de lo estudiado anteriormente se decide apostar por un sistema que se ajuste a la realidad jurídica de los tribunales, desarrollado por especialistas con herramientas de código abierto que permitan su mantenimiento, adaptaciones y ulteriores versiones aunque se tuvo en cuenta algunos aspectos de los sistemas antes mencionados. Este sistema al generar todas las resoluciones de manera automática garantizará que todos los expedienten estén disponibles de forma digital evitando el uso del papel.

1.2 Introducción a las Bases de Datos

Comúnmente se usa el término bases de datos sin saber del todo su significado, que no es más que la persistencia de un grupo de información en un lugar y un soporte determinado y a la cual se pueda acceder. Las antiguas bibliotecas, los archivos y todo este tipo de documentación que siempre se consolidó en papel, y se almacenó en grandes estantes y locales donde la accesibilidad se hacía un poco engorrosa, sin quitar méritos a los niveles de organización que eran también un tipo de bases de datos. En la actualidad, y debido al desarrollo tecnológico de campos como la informática y la electrónica, la mayoría

de las bases de datos están en formato digital (electrónico), que ofrece un amplio rango de soluciones al problema de almacenar datos.

1.2.1 Componentes de una Base de Datos

Hardware: constituido por dispositivos de almacenamiento como discos, tambores, cintas, entre otros.

Software: que es el Sistema Gestor de Bases de Datos o SGBD.

Datos: los cuales están almacenados de acuerdo a la estructura externa y van a ser procesados para convertirse en información.

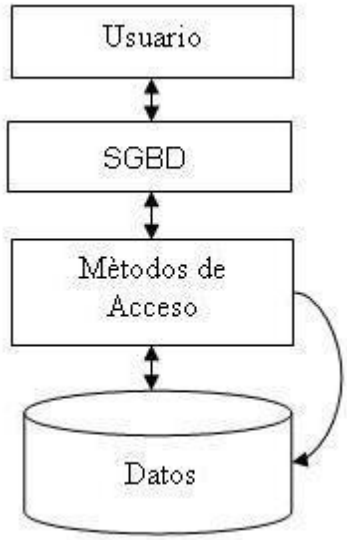


Figura 1: Flujo de acceso a datos de una BD

1.2.2 Clasificaciones de las Bases de Datos.

Las bases de datos pueden clasificarse de varias maneras, de acuerdo al criterio elegido para su clasificación:

Según la variabilidad de los datos almacenados.

Bases de datos estáticas: Éstas son bases de datos de sólo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones.

Bases de datos dinámicas: Éstas son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta.

Según el contenido.

Bases de datos bibliográficas: Solo contienen un representante de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, entre otros.

Bases de datos de texto completo: Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas.

1.3 Metodologías de diseño de Bases de Datos

El diseño de una base de datos es un proceso complejo que abarca decisiones a muy distintos niveles. La complejidad se controla mejor si se descompone el problema en subproblemas y se resuelve cada uno de estos subproblemas independientemente, utilizando técnicas específicas. Así, el diseño de una base de datos se descompone en diseño conceptual, diseño lógico y diseño físico.

1.3.1 El diseño conceptual

Parte de las especificaciones de requisitos de usuario y su resultado es el esquema conceptual de la base de datos. Un modelo conceptual se expresa mediante un modelo de datos de alto nivel. Uno de los más empleados es el modelo Entidad–Relación (7).Mediante este modelo se pretende visualizar los elementos que pertenecen a una BD, que reciben el nombre de entidades, las cuales se corresponden con el concepto de clase de la Programación Orientada a Objeto (POO) y donde cada tupla de una futura relación representaría un objeto de la POO.

1.3.2 El diseño lógico

Parte del esquema conceptual y da como resultado un esquema lógico. Se centra en las operaciones y se implementa en el algún manejador de BD. El modelo lógico es más cercano al ordenador y depende del SGBD que se vaya a utilizar. El resultado del mismo puede ser el modelo relacional, el modelo de red o el modelo jerárquico. Es el momento en que se realiza el proceso de normalización.

1.3.3 El diseño físico

Parte del esquema lógico y da como resultado un esquema físico. Tiene como salida la implementación de la base de datos con todas sus restricciones. Considera las estructuras de almacenamiento y los métodos de acceso necesarios para proporcionar un acceso eficiente a la base de datos en memoria secundaria. Depende del SGBD a utilizar.

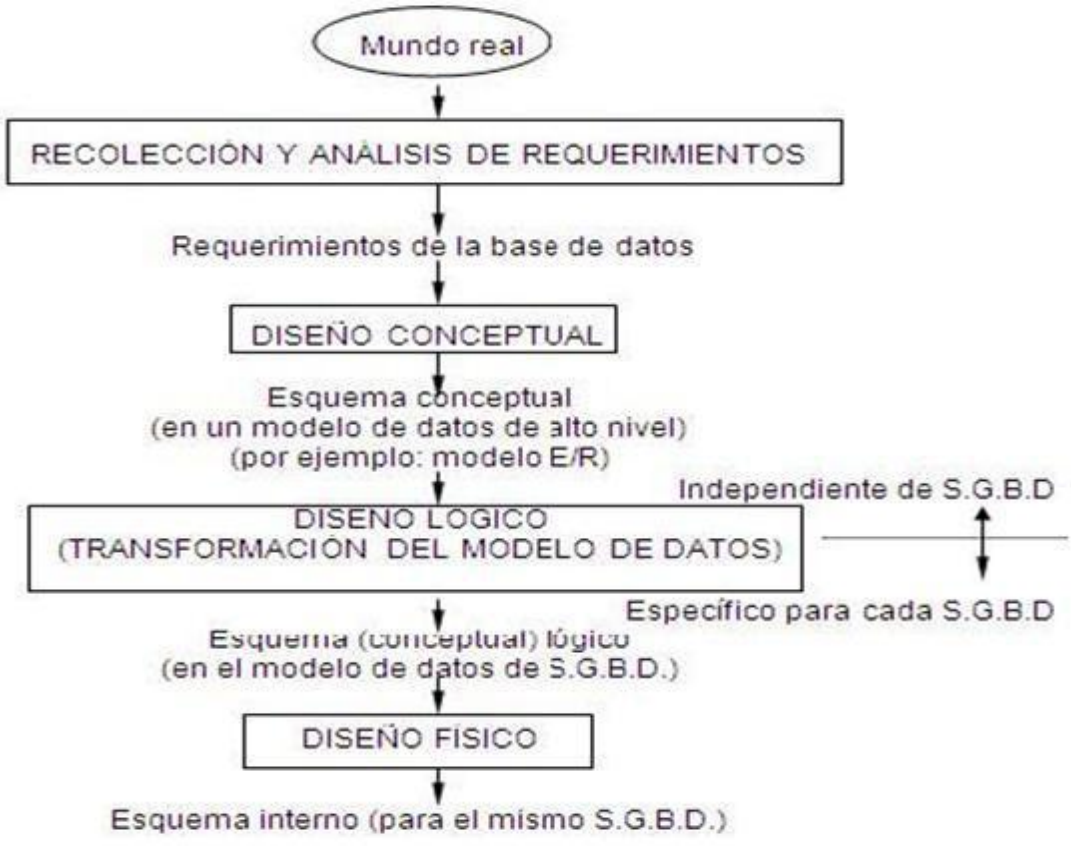


Figura 2: Pasos para el diseño e una BD

1.4 Modelos para el desarrollo de Bases de Datos

El modelado de datos es una herramienta de comunicación utilizada por los diseñadores de BD debido a su sencillez. Una de sus ventajas es la independencia del gestor de BD que se vaya a utilizar pues no está orientado a ningún gestor específico.

En el modelado de datos sólo se debe reflejar la existencia de los datos más relevantes para el sistema, no importa su dominio, ni que se hará con los datos, tampoco se tiene en cuenta las restricciones de espacio, almacenamiento o tiempo de ejecución.

1.4.1 Modelo jerárquico

Una Base de datos jerárquica es un tipo de Sistema Gestor de Bases de Datos que, como su nombre indica, almacenan la información en una estructura jerárquica que enlaza los registros en forma de estructura de árbol (similar a un árbol visto al revés), en donde un nodo padre de información puede tener varios nodos hijo.

Esta relación jerárquica no es estrictamente obligatoria, de manera que pueden establecerse relaciones entre nodos hermanos, solo que las relaciones son unidireccionales. Esto implica que solamente se puede consultar la base de datos desde los nodos hoja hacia el nodo raíz. La consulta en el sentido contrario requiere una búsqueda secuencial por todos los registros de la base de datos. El modelo jerárquico no diferencia una vista lógica de una vista física de la base de datos. De manera que las relaciones entre datos se establecen mediante referencia a direcciones físicas del medio de almacenamiento (sectores y pistas).

Entre las limitaciones del modelo jerárquico se encuentran:

- **Rigidez:** deriva de la falta de capacidad de las organizaciones jerárquicas para representar sin redundancias ciertas estructuras muy difundidas en la realidad, como son las interrelaciones reflexivas y N: M.

- **Duplicidad de registros:** no se garantiza la inexistencia de registros duplicados. Es decir, no se garantiza que dos registros cualesquiera tengan diferentes valores en un subconjunto concreto de campos.

- **Integridad referencial:** no existe garantía de que un registro *hijo* esté relacionado con un registro *padre* válido. Por ejemplo, es posible borrar un nodo *padre* sin eliminar antes los nodos *hijo*, de manera que éstos últimos están relacionados con un registro inválido o inexistente.

- **Poca flexibilidad:** puede obligar a la introducción de redundancias cuando es preciso instrumentar, mediante el modelo jerárquico, situaciones del mundo real que no responden a una jerarquía

- **Desnormalización:** las bases de datos jerárquicas no tienen controles que impidan la desnormalización de una base de datos.

1.4.2 Modelo de red

Una base de datos de red llamada algunas veces estructura de plex es una base de datos conformada por una colección o set de registros, los cuales están relacionados mediante punteros o ligas en grafos arbitrarios. Abarca más que la estructura de árbol, porque un nodo hijo en la estructura red puede tener más de un nodo padre. El modelo de red evita esta redundancia en la información, a través de la incorporación de un tipo de registro denominado el conector. La dificultad surge al manejar las conexiones o ligas entre los registros y sus correspondientes registros conectores.

El modelo en red general es muy flexible debido a la inexistencia de restricciones inherentes, pero también por esta misma razón su instrumentación física resulta difícil y poco eficiente. Esta es la causa de que se le suela introducir restricciones al llevarlo a la práctica. El modelo jerárquico y el modelo CODASYL[2] son modelos que responden a estructuras del tipo de red pero con restricciones bastante fuertes.

Este modelo presenta los siguientes problemas:

[2]CODASYL: Es un modelo de datos de tipo red que introduce restricciones inherentes. este modelo constituye una simplificación del modelo en red general

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

1.4.3Modelo de datos orientado a objetos

Los sistemas basados en modelos de datos orientados a objeto fueron inspirados a partir del paradigma de programación orientada a objeto.

Este modelo, bastante reciente, y propio de los modelos informáticos orientados a objetos, trata de almacenar en la base de datos los objetos completos (estado y comportamiento). Una base de datos orientada a objetos es una base de datos que incorpora todos los conceptos importantes del paradigma de objetos:

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

La orientación a objetos ofrece flexibilidad no está limitada por los tipos de datos y los lenguajes de consulta de los sistemas de bases de datos tradicionales.

1.4.4 Modelo de datos relacional

Uno de los puntos fuertes en el modelo relacional es la sencillez de su estructura lógica donde todos sus datos están estructurados en forma de tablas formadas por columnas y filas. Debido a su rápido entendimiento por parte de los usuarios que no tienen conocimientos profundos sobre diseño de bases de datos, es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos

dinámicamente. Su idea fundamental es el uso de entidades (tablas), compuestas por registros (las filas de una tabla), que representarían las tuplas, y campos (las columnas de una tabla). (8)

Durante su diseño, una BD relacional pasa por un proceso al que se le conoce como normalización de una base de datos.

En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos. La información puede ser recuperada o almacenada mediante consultas que ofrecen una amplia flexibilidad y poder para administrar la información.

Otras ventajas

- **Esquema específico** para cada aplicación: las bases de datos relacionales son creadas para cada aplicación específica, siendo complicado adaptar los esquemas a nuevas aplicaciones.
- **Modelo de datos complejo**: permiten manejar complejos modelos de datos que requieren muchas tablas.
- **Integridad de datos**: todos sus componentes están desarrollados para mantener la consistencia de la información en todo momento. Esto incluye operaciones de roll-back[3], integridad referencial y operaciones orientadas a transacciones[4].

➢ Por estas características es el modelo que se utilizará en el presente trabajo.

Roll-back es una operación que devuelve a la base de datos a algún estado previo. Los Roll-backs son importantes para la integridad de la base de datos, a causa de que significan que la base de datos puede ser restaurada a una copia limpia incluso después de que se han realizado operaciones erróneas.

[4] implican la modificación o borrado constante de los datos almacenados.

1.5 Metodología a utilizar para el desarrollo de la Base de Datos

La organización del trabajo es fundamental por lo que se seguirá como guía el Modelo de Desarrollo de Bases de Datos para Procesos de Desarrollo de Software propuesto en el trabajo de diploma del Ingeniero Alain Osorio Rodríguez. (9)

En el mismo se contemplan 3 fases fundamentales:

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

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

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

1. Modelado de Datos.
2. Configuraciones.
3. Implementación.
4. ADTP (Acceso a Datos, Tuning, Prueba).

Figura 3: Modelo de Desarrollo de BD

1.5.1 Modelado de Datos

La actividad tiene como tarea general la estructuración del Modelo de Datos y su respectiva documentación.

Los objetivos de esta actividad son:

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

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

El artefacto que se genera es el Modelo de Datos.

1.5.2 Configuraciones

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

El objetivo de esta actividad es:

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

Los artefactos que se generan en esta actividad son:

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

2. Estructura automatizada de aplicación de las configuraciones.

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

1.5.3 Implementación

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

Los objetivos de esta actividad son:

1. Implementar las funcionalidades de la base de datos.

2. Documentar las funcionalidades.

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

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

1.5.4 ADTP

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

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

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

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

Los objetivos de esta actividad son:

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

El artefacto que se genera es el Acceso a Datos.

1.6 Normalización de las Bases de Datos

El proceso de normalización de las BD consiste en aplicar una serie de reglas a las relaciones obtenidas tras el paso del modelo entidad relación al modelo relacional. Las bases de datos relacionales se normalizan para:

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

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

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

Las formas normales someten el esquema de relación a una serie de pruebas para certificar si pertenecen a una cierta forma normal, cuando el esquema relacional esté en FNBC, es que ya el esquema se llevó a 3FN, y para estar en 3FN el esquema se tuvo que haber llevado a 2FN y así sucesivamente, por lo que se puede decir que la FNBC es una de las FN más deseadas para un buen diseño de BD. No obstante, no siempre que el diseño de BD esté en la FNBC podemos garantizar que sea un diseño idóneo de los datos. Ya que esta forma normal es una versión ligeramente más fuerte de la Tercera forma normal (3FN). La forma normal de Boyce-Codd requiere que no existan dependencias funcionales no triviales de los atributos que no sean un conjunto de la clave candidata, es decir que no permite ninguna dependencia funcional en la cual el conjunto determinante[5] de atributos no sea una clave candidato.

[5] Determinante: un atributo del cual depende funcionalmente (por completo) algún otro atributo

1.7 Seguridad de las Bases de Datos

Las bases de datos hoy en día son de gran importancia en cada una de las aplicaciones que la requieran, es por esto que la seguridad para su acceso y manejo de la información se restringe en una jerarquía de usuarios. Los SGBD permiten mantener la seguridad mediante el establecimiento de claves para identificar al personal autorizado a utilizar la base de datos. Las autorizaciones se pueden realizar a nivel de operaciones, de modo que un usuario puede estar autorizado a consultar ciertos datos pero no a actualizarlos. En el caso específico el gestor de bases de datos PostgreSQL 8.4 presenta una serie de características que facilita el trabajo para tener un alto nivel de seguridad.

La seguridad de la base de datos está implementada en varios niveles:

1. Protección de los ficheros de la base de datos, ya que todos los ficheros almacenados en la base de datos están protegidos contra escritura por cualquier cuenta que no sea la del súper usuario de Postgres.

2. Las conexiones de los clientes se pueden restringir por dirección IP y/o por nombre de usuario mediante el fichero pg_hba.conf situado en PG_DATA.

3. Las conexiones de los clientes pueden ser autentificadas mediante otros paquetes externos.

4. A cada usuario de Postgres se le asigna un nombre de usuario y (opcionalmente) una contraseña. Por defecto, los usuarios no tienen permiso de escritura a bases de datos que no hayan creado.

Los usuarios pueden ser incluidos en grupos, y el acceso a las tablas puede restringirse en base a esos grupos.

1.8 Definición de integridad

Son restricciones que definen los estados de consistencia de la Base de Datos, esto significa que la BD o los programas que generaron su contenido, incorporen métodos que aseguren que el contenido de los datos del sistema no se rompa, así como las reglas del negocio. (10) La integridad de los datos puede

verse afectada añadiendo datos no válidos, modificando datos existentes, que se les asigne un valor incorrecto o eliminando datos que violen alguna regla.

La integridad de datos presenta varias restricciones que posibilitan la declaración y comprobación de condiciones para expresar la consistencia, corrección y exactitud de datos almacenados, estas son: Integridad de dominio, de clave, de entidad, referencial, entre otras.

1.8.1 Integridad de los datos

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

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

Datos Requeridos: Establece que una columna tenga un valor no NULL. Se define efectuando que la declaración de una columna es NOT NULL cuando la tabla que contiene las columnas se crea por primera vez, como parte de la sentencia CREATE TABLE.

Chequeo de Validez: Cuando se crea una tabla donde en cada columna tiene un tipo de datos y el sistema gestor de base de datos asegura que solamente los datos del tipo especificado sean ingresados en la tabla.

Integridad de entidad: Establece que la clave primaria de una tabla debe tener un valor único para cada fila de la tabla; sino, la base de datos perderá su integridad. Se especifica en la sentencia CREATE TABLE. El sistema gestor de base de datos comprueba automáticamente la unicidad del valor de la clave primaria con cada sentencia INSERT Y UPDATE. Un intento de insertar o actualizar una fila con un valor de la clave primaria ya existente fallará.

Integridad Referencial: La integridad referencial es un sistema de reglas que utilizan la mayoría de las bases de datos relacionales para asegurarse que los registros de tablas relacionadas son válidos y que no se borren o cambien datos relacionados de forma accidental produciendo errores de integridad.

1.9 Sistema Gestor de Bases de Datos (SGBD)

Los SGBD son los responsables de tratar todas las peticiones de información de los usuarios. Es un software que sirve de interfaz entre la BD, el usuario y las aplicaciones que la utilizan, pretende manejar de manera clara, sencilla y ordenada un conjunto de datos, permite la utilización y/o actualización de los datos en una o varias BD, desde diferentes puntos de vista a la vez, por uno o varios usuarios.

Este sistema es un conjunto de programas de propósito general, que permite a los usuarios controlar el acceso y la utilización de la base de datos para incluir, modificar o recuperar información, incluyendo prestaciones con el fin de conseguir la independencia, integridad y seguridad de los datos y la concurrencia de los usuarios.

Los sistemas gestores de bases de datos se dividen en 2 grupos en cuanto a forma de adquirirlas:

SGBD Propietarios:

- Oracle (Oracle Corp.).
- MySQL (SUN Microsystems).
- DB2 (IBM).
- InterBase (CodeGear).
- MaxDB (SAP).
- JavaDB (SUN Microsystems).
- SQL Server (Microsoft Corp.).
- Infomix (IBM).

SGBD Libres:

- PostgreSQL.
- FireBird.
- SQLite.

1.9.1Oracle

Oracle es básicamente una herramienta cliente/servidor para la gestión de Bases de Datos fabricado por Oracle Corporation. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales.

Principales características:

- ✓ Entorno cliente/servidor.
- ✓ Gestión de grandes bases de datos.
- ✓ Usuarios concurrentes.
- ✓ Alto rendimiento en transacciones.
- ✓ Sistemas de alta disponibilidad.
- ✓ Disponibilidad controlada de los datos de las aplicaciones.
- ✓ Adaptación a estándares de la industria, como SQL-92.
- ✓ Gestión de la seguridad.
- ✓ Autogestión de la integridad de los datos.
- ✓ Opción distribuida.
- ✓ Portabilidad.
- ✓ Compatibilidad.
- ✓ Replicación de entornos.

Ventajas:

- Puede ejecutarse en todas las plataformas, desde una computadora personal hasta un supercomputador.

- Oracle es la BD más utilizada por las corporaciones más grandes del mundo, por su robustez y por la seguridad.
- Oracle ofrece soporte mundial a través de sus centros de soporte y sus sitios Web.
- Proporcionan actualización de versiones gratis al cliente; siempre y cuando la licencia de soporte lo cubra.
- El 80% o más de los sitios Web en Internet tienen bases de datos Oracle.
- Oracle es más que un manejador de BD. Oracle corporation ofrece otras soluciones a la plataforma de Negocio, E-Business, E-commerce.
- Tiene productos para interactuar con otras BD, tales como los Transparent Gateways para: DB2, SQL Server, Informix, Mysql y muchas otras.

Desventajas:

- El mayor inconveniente de Oracle es quizás su precio. Incluso las licencias de Personal Oracle son excesivamente caras.
- Necesidad de ajustes. Un error frecuente consiste en pensar que basta instalar el Oracle en un servidor y conectar directamente las aplicaciones clientes. Un Oracle mal configurado puede ser desesperantemente lento.
- También es elevado el coste de la formación, y sólo últimamente han comenzado a aparecer buenos libros sobre asuntos técnicos distintos de la simple instalación y administración.

1.9.2 MySQL

MySQL surgió alrededor de la década del 90, Michael Windenis comenzó a usar mSQL para conectar tablas usando sus propias rutinas de bajo nivel (ISAM). Tras unas primeras pruebas, llegó a la conclusión de que mSQL no era lo bastante flexible ni rápido para lo que necesitaba, por lo que tuvo que desarrollar nuevas funciones. Esto resulto en una interfaz SQL a su base de datos, totalmente compatible a mSQL.

El origen del nombre MySQL no se sabe con certeza de donde proviene, por un lado se dice que en sus librerías han llevado el prefijo “my” durante los diez últimos años, por otra parte, la hija de uno de los desarrolladores se llama My. Así que no está claramente definido cuál de estas dos causas han dado lugar al nombre de este conocido gestor de bases de datos.

Principales características:

Inicialmente, MySQL carecía de algunos elementos esenciales en las bases de datos relacionales, tales como integridad referencial y transacciones. A pesar de esto, atrajo a los desarrolladores de páginas web con contenido dinámico, debido a su simplicidad, de tal manera que los elementos faltantes fueron complementados por la vía de las aplicaciones que la utilizan. Poco a poco estos elementos faltantes, están siendo incorporados tanto por desarrolladores internos, como por desarrolladores de software libre.

En las últimas versiones se pueden destacar las siguientes características principales:

- El principal objetivo de MySQL es velocidad y robustez
- Soporta gran cantidad de tipos de datos para las columnas.
- Gran portabilidad entre sistemas, puede trabajar en distintas plataformas y sistemas operativos.
- Cada base de datos cuenta con 3 archivos: Uno de estructura, uno de datos y uno de índice y soporta hasta 32 índices por tabla.
- Aprovecha la potencia de sistemas multiproceso, gracias a su implementación multihilo.
- Flexible sistema de contraseñas (passwords) y gestión de usuarios, con un muy buen nivel de seguridad en los datos.
- El servidor soporta mensajes de error en distintas lenguas

Ventajas:

- Velocidad al realizar las operaciones, lo que le hace uno de los gestores con mejor rendimiento.
- Bajo costo en requerimientos para la elaboración de bases de datos, ya que debido a su bajo consumo puede ser ejecutado en una máquina con escasos recursos sin ningún problema.
- Facilidad de configuración e instalación.
- Soporta gran variedad de Sistemas Operativos
- Baja probabilidad de corromper datos, incluso si los errores no se producen en el propio gestor, sino en el sistema en el que está.
- Conectividad y seguridad

Desventajas:

- Un gran porcentaje de las utilidades de MySQL no están documentadas.

- No es intuitivo, como otros programas.
- No soporta subconsultas.
- No soporta transacciones.
- No soporta vistas.
- Se hace inestable cuando contiene gran cantidad de datos.
- No soporta transacciones, "roll-backs" ni subselects.
- No considera las claves ajenas. Ignora la integridad referencial, dejándola en manos del programador de la aplicación.

1.9.3PostgreSQL

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

PostgreSQL es uno de los Sistemas Gestores de Bases de Datos más utilizados por la comunidad de software libre por las razones siguientes: Cumple con las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) y soporta el lenguaje común de acceso a los datos: SQL. Es multiplataforma y posee buenas interfaces de instalación y administración. Aproxima los datos a un modelo Objeto-Relacional, y es capaz de manejar completas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQL declarativas, control de concurrencia multiversión, soporte multiusuario, transacciones y optimización de consultas.

Está basado en el proyecto POSTGRES, de la universidad de Berkeley. Es una derivación libre (Open Source) de este proyecto, y utiliza el lenguaje SQL92/SQL99. Fue el pionero en muchos de los conceptos existentes en el Sistema Objeto-Relacional actual, incluido más tarde en otros sistemas de gestión comerciales. Incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores e integridad transaccional. Lleva más de una década de desarrollo, siendo hoy en día un sistema bastante avanzado, que tiene soporte nativo para los lenguajes de programación: C, C++, Java, Python, PHP y muchos más. Se encuentra bajo la licencia BSD (Berkeley Software Distribution).

Entre otras características que presenta se encuentran las siguientes:

- Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP...), cadenas de bits, entre otros.
- Además permite la creación de tipos de datos propios.
- Incorpora una estructura de datos array.
- Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, entre otras.
- Permite la declaración de funciones propias, así como la definición de disparadores.
- Soporta el uso de índices, reglas y vistas.
- Incluye herencia entre tablas.
- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

1.10 Herramientas CASE

La realización de un nuevo software requiere que las tareas sean organizadas y completadas en forma correcta y eficiente. Las Herramientas CASE fueron desarrolladas para automatizar esos procesos y facilitar las tareas de coordinación de los eventos que necesitan ser mejorados en el ciclo de desarrollo de software.

Se puede definir a las Herramientas CASE (Computer Aided Software Engineering) como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclode Vida de desarrollo de un Software.

1.10.1 Visual Paradigm for UML

Es una herramienta profesional, es decir, un software de modelado que utiliza UML como lenguaje de modelado. Soporta el ciclo completo de vida del software (análisis y diseño orientados a objetos, implementación, pruebas y despliegue); permitiendo en cada una de sus etapas generar los diagramas necesarios sin ningún tipo de problema. Está orientado para posibilitar tanto ingeniería directa como inversa, pues posee varios lenguajes de programación que aprueban la generación de código. Esta herramienta CASE soporta la importación y exportación de varias versiones de XML. Facilita la

modelación de diversos tipos de diagramas, transformando códigos de estos modelos, concibiendo de esta manera, los códigos fuentes de los diagramas.

Algunas características de Visual Paradigm:

Posee generación de código para Java y la exportación de todos los diagramas a formato HTML y jpg.

- Ostenta de un medio de creación de diagramas para UML 2.0.
- Posibilita la integración a los principales IDE.
- Cuenta con un diseño enmarcado en casos de uso y dirigido al negocio.
- Contiene facilidades para representar especificaciones de casos de uso del sistema.

Visual Paradigm es multiplataforma; posibilita la transformación de diagramas de entidad-relación en tablas de base de datos. Posee además una distribución automática de diagramas, ya que cuenta con una reorganización de las figuras y conectores de los diagramas UML. Permite exportar los diagramas a imágenes y páginas HTML. Además facilita la conversión de diagramas de colaboración a secuencia y viceversa.

1.10.3 Embarcadero Erwin Studio

ER / Studio es la arquitectura de datos y software de diseño de base de datos desarrollada por Embarcadero Technologies. Funciona a través de múltiples plataformas de bases de datos y es utilizado por los arquitectos de datos, los modeladores de datos, administradores de base de datos y analistas de negocio para crear y gestionar diseños de base de datos, documentar y reutilizar los activos de datos.

Ofrece a los administradores y desarrolladores de bases de datos la posibilidad de modelado de datos de forma visual, permitiendo el diseño y mantenimiento de bases de datos transaccionales, da soporte a la toma de decisiones y para Web. ER/Studio también soporta diseño multinivel y ofrece la capacidad de controlar, documentar y desplegar rápidamente cambios en el diseño en las principales plataformas.

Soporta los motores de bases de datos Oracle, DB2, SQL Server y Sybase; además de conexiones vía ODBC (Open Database Connectivity). (11)

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

1.11 Otras Herramientas

1.11.1Mapeador de Doctrine (Doctrine Mapping)

Doctrine es un potente y completo sistema Mapeador de Objeto Relacional (ORM) para PHP 5.2.3+ con un DBAL (Capa de Abstracción de Base de Datos) incorporado. Se está empezando a ver su potencial, pero de la documentación se puede decir que tiene todas las características necesarias para ser funcional en casi cualquier proyecto. Entre otros elementos se tiene la posibilidad de exportar una base de datos existente a sus clases correspondientes y también a la inversa, es decir convertir clases (convenientemente creadas siguiendo las pautas del ORM) a tablas de una base de datos. Por otro lado, como la librería es bastante grande ésta tiene un método para ser compilada al pasar a producción. (12)

Ventajas que facilitan enormemente tareas comunes y de mantenimiento:

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

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

Portabilidad: Utilizar una capa de abstracción que permite cambiar en mitad de un proyecto de una base de datos MySQL (Sistema de gestión de base de datos relacional) a una Oracle sin ningún tipo de complicación. Esto es debido a que no se utiliza una sintaxis (MySQL, Oracle o SQLite) para acceder al modelo, sino una sintaxis propia del ORM utilizado que es capaz de traducir a diferentes tipos de bases de datos.

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

Mantenimiento del código: Gracias al correcto ordenamiento de la capa de datos, modificar y mantener el código es una tarea sencilla. (12)

Conclusiones Parciales

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

CAPÍTULO 2: PROPUESTA DE SOLUCIÓN

Introducción

La solución propuesta está sustentada sobre la base de este capítulo, en el cual se definen las actividades más importantes que posibilitaron la realización del diseño de la Base de Datos del módulo Civil del proyecto “Sistema para los Tribunales Cubanos” a partir del Modelo de Desarrollo de Bases de Datos, que dio paso a la implementación de la misma, exponiéndose los artefactos generados en la fase de inicio de dicho modelo.

También se brinda una descripción detallada de las entidades más importantes en cuestión de funcionalidad para el módulo Civil.

2.1 Entorno de desarrollo

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

2.1.1 Configuraciones

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

En el servidor se instalará el sistema operativo Debian en su versión 5.0.4, el mismo es un sistema operativo libre, de ahí que su utilización esté en correspondencia con las políticas de la universidad y el

país con respecto al desarrollo de software a través de tecnologías de código abierto. Una vez finalizada la instalación deberán configurarse debidamente la hora y la dirección IP del servidor, lo que garantizará un mejor funcionamiento. De igual forma debe conectarse al repositorio de actualizaciones del sistema operativo de donde podrán obtenerse, en su mayoría, los programas y servicios indispensables para el servidor.

Otro software que debe instalarse para el trabajo con la base de datos es el SGBD PostgreSQL 8.4. En el mismo como medida de seguridad se eliminará el esquema público que por defecto trae asociado, creándose en su lugar un esquema para cada materia, lo que contribuirá a lograr una mayor organización de la base de datos. Cada uno de los esquemas contará con una cuenta de usuario propia. Para mayor información remitirse al artefacto “Documento de configuraciones del sistema operativo – sistema gestor de bases de datos – base de datos”.

2.1.1.1 Arquitectura

Como propuesta inicial la solución informática para los tribunales estará conformada por un servidor en cada tribunal independientemente de la instancia en la que se encuentre el mismo. Al mismo tiempo existirá un centro de datos, con toda la información de todos los tribunales, el cual estará sincronizado con cada uno de los tribunales de forma tal que en cada momento se pueda consultar información, con el acceso autorizado, de cualquier tribunal. Los tribunales no se comunicarán directamente entre sí, se comunicarán a través del centro de datos. De esta forma se garantiza la independencia de la tramitación judicial en cada uno de los tribunales y un control centralizado de toda la información que se maneja en estos. (Fig. 4).

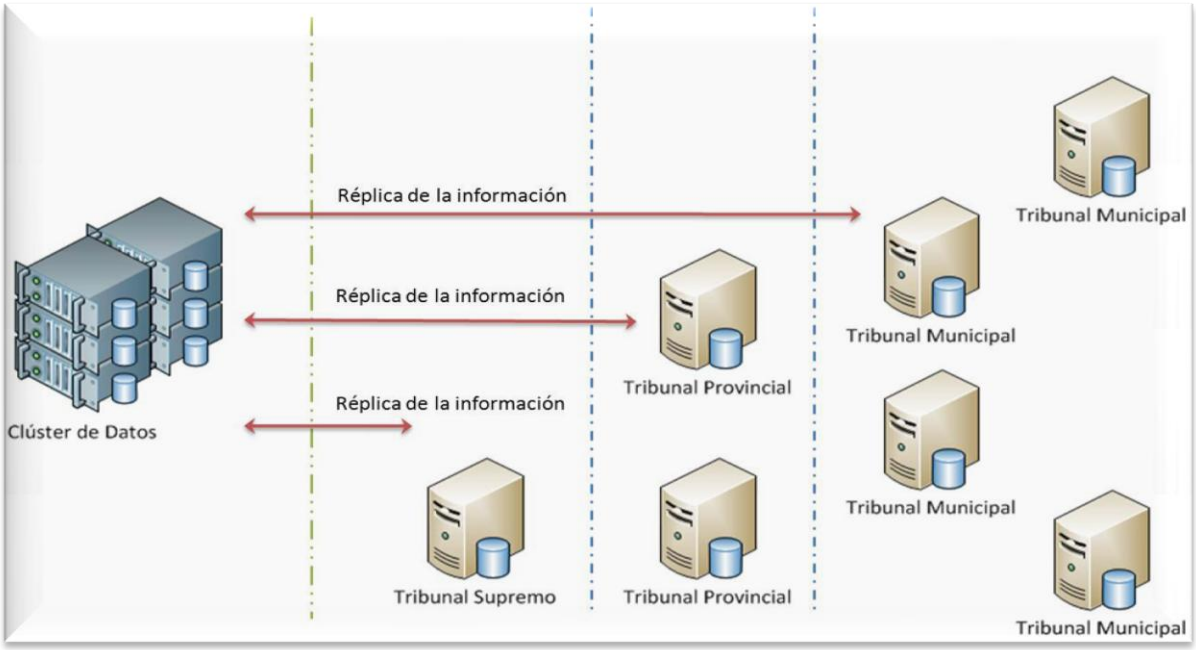


Figura 4: Estructura de servidores

2.1.1.2 Seguridad

Las bases de datos están bajo constante amenaza de sufrir ataques de personas o sistemas no autorizados que ponen en riesgo su integridad y confidencialidad; es por ello que deben tomarse una serie de medidas que impidan estos sucesos y permitan conservar salvas de la BD para restaurar los datos si se pierden o corrompen por alguna razón pues ellos constituyen un recurso valioso que debe ser estrictamente controlado y gestionado al igual que cualquier otro recurso corporativo.

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

La configuración de la seguridad de la base de datos está implementada en dos niveles. En el primer nivel se configuran los permisos de conexión para los host y los usuarios a la o las BD en el archivo pg_hba.conf, se define qué dirección o direcciones IP tendrán acceso a cuál o cuáles BD, y en qué modo podrán conectarse: conexión sin contraseña, validando el usuario y la contraseña o que rechace cualquier conexión desde el IP o rangos IP y usuarios seleccionados. Para mayor información consultar artefacto

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

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

Figura 5: Roles y usuarios

2.1.2 Arquitectura del entorno de desarrollo

El sistema para los tribunales cubanos cuenta con la siguiente estructura de servidores: en un servidor Windows estarán instalados el Embarcadero Repository 4.0.1 y el SQL Server 2005 para garantizar el repositorio y el trabajo en equipo con la herramienta de modelado ER/Studio 7.1. Se configurará además, una salva automática de la base de datos del repositorio, programada diariamente a las 4 PM. En las estaciones de trabajo estará instalada la herramienta de modelado ER/Studio 7.1 (Fig. 5).

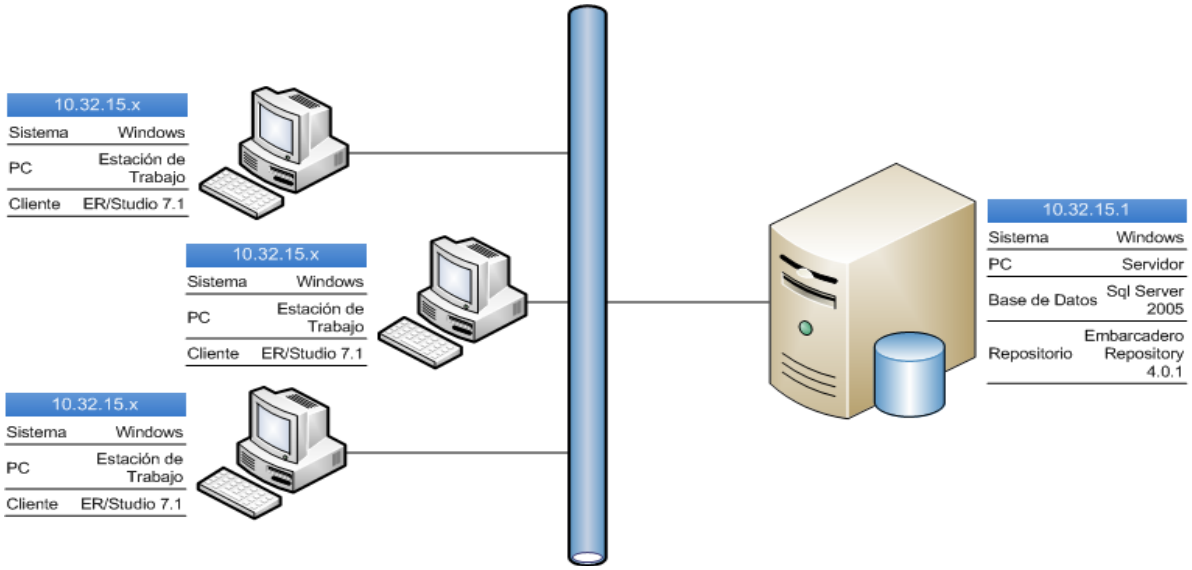


Figura 6: Entorno de desarrollo para el modelado de datos

2.2 Modelado de Datos

2.2.1 Nomenclatura y normas para el Modelo de Datos

Las reglas de nomenclatura para los objetos de la BD se realizó siguiendo el estándar definido por el proyecto Tribunales Populares Cubanos plasmado en el documento: “Entorno de Desarrollo de Bases de Datos”. Destacar el uso de macros[6] para nombrar las llaves primarias y foráneas y para las definiciones de las tablas y sus atributos. También se hizo uso de un estándar de codificación para el trabajo con las funciones en la BD.

[6] Script que automatiza acciones sobre el Modelo de Datos.

2.2.2 Descripción general del modelo entidad relación del Subsistema Civil

La base de datos propuesta para el módulo Civil cuenta con 67 tablas en modelo lógico y físico. Dentro de las 67 tablas existen 23 nomencladores y 44 tablas de datos. Las tablas más importantes son aquellas que gestionarán la información de: personas, expediente, escritos, trámites, demandantes, demandados, actos procesales, las cuales recogen la información de mayor peso en el módulo. Modelo Lógico (Ver anexo 1), Modelo físico (Ver anexo 2).

Descripción de las principales tablas del modelo Entidad-Relación

Dpersona				
Descripción: Personas				
Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**
Idpersona		Serial/Integer	No	
Nacionalidad	Texto	Text	Si	Nacionalidad de la persona
CorreoElectronico	Email	Text	No	Correo Electrónico de la persona
IdnPersona	EnteroGrande	Bigint	No	Identificador del tipo de persona(Natural, Jurídica)

Tabla 1: dPersona

dPersonaJuridica				
Descripción: Personas jurídicas.				
Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**
Idpersona	EnteroGrande	Bigint	No	Identificador de la persona
Nombre	Texto	Text	No	Nombre de la persona jurídica

Tabla 2: dPersonaJuridica

dPersonaNatural				
Descripción: Datos de las personas naturales				
Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**

IdPersona	EnteroGrande	Bigint	No	Identificador de la persona
PrimerApellido	Texto	Text	No	Primer apellido de la persona
SegundoApellido	Texto	Text	No	Segundo apellido de la persona
PrimerNombre	Texto	Text	No	Nombre de la persona
SegundoNombre	Texto	Text	Si	Segundo nombre de la persona
CI	NumeroCI	VARCHAR(11)	Si	Carnet de Identidad de la persona
Pasaporte	Texto	Text	Si	Pasaporte de la persona

Tabla 3: dPersonaNatural

dPersonaParte				
Descripción: Contiene los atributos de las personas demandantes y demandadas				
Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**
IdPersona	EnteroGrande	Bigint	No	Identificador de la persona
Naturaleza	Texto	Text	No	Lugar de nacimiento de la persona
Oficio	Texto	Text	No	Empleo de la persona
IdnEstadocivil	Texto	Text	No	Identificador del estado civil de la persona

Tabla 4: dPersonaParte

dExpediente				
Descripción: Legajo que contiene toda la tramitación judicial en un caso concreto.				
Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**
IdExpediente		Seria/Integer	No	NO identificador del expediente
Numero	Texto	Text	No	Número del expediente
FechaRadicacion	FechaHoraActual	TimesTamp/Date	No	Fecha de radicación del expediente

Tabla 5: dExpediente

dTramite
Descripción: Tabla donde se registran los actos procesales que se van realizando.

Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**
IdTramite		Serial/Integer	No	Identificador del trámite
Mac	Mac	Integer	No	Número MAC
Ip	Texto	Text	No	Ip de la pc donde se realiza el trámite
Sala	Texto	Text	No	Sala del tribunal donde se realiza el trámite
Tribunal	Texto	Text	No	Tribunal donde se realiza el trámite
Fecha	FechaHoraActual	TimeStamp/Dat	No	Fecha en que se realiza el trámite
IdnTramite	Texto	Text	Si	Identificador del nomenclador NTramite donde se relacionan los actos procesales y sus estados.
IdPersona	EnteroGrande	Bigint	No	Identificador de la persona que realiza el trámite.
IdExpediente	EnteroGrande	Bigint	No	Identificador del expediente donde se registra el trámite.

Tabla 6: dTramite

dEscritoDemanda				
Descripción: Escritos de demanda.				
Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**
IdEscrito		Entero	No	Identificador del escrito.
Otrosíes	Texto	Text	No	Otros elementos solicitados en la demanda.
Pretensiones	Texto	Text	No	Lo que se pretende con la demanda.
Fundamentos	Texto	Text	No	Artículos de la ley que fundamentan la demanda.
Hechos	Texto	Text	No	Situaciones que provoca la interposición de la demanda.

Tabla 7: dEscritoDemanda

dDemandante
Descripción: Relaciona todos los demandantes de los procesos

Atributo	Dominio	Tipo de Dato	Nulo	Descripción
IdDemandnte		Serial/Integer	No	Identificador de la persona demandante
IdPersona	Mac	Bigint	No	Identificador de la persona
IdLitigio	Texto	Integer	No	Identificador de dLitigio, relaciona a que litigio está asociado.

Tabla 8: dDemandante

dDemandado				
Descripción: Relaciona todos los demandantes de los procesos				
Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**
IdDemandnte		Serial/Integer	No	Identificador de la persona demandado
IdPersona	Mac	Bigint	No	Identificador de la persona
IdLitigio	Texto	Integer	No	Identificador de dLitigio, relaciona a que litigio está asociado.

Tabla 9: dDemandado

nTramite				
Descripción: Relaciona la secuencia de los actos procesales y sus estados.				
Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**
Idntramite	EnteroGrande	Bigint	No	Identificador del Tramite
IdnActoProcesalOrigen	EnteroGrande	Bigint	No	Identificador del acto procesal
IdnActoProcesalDestino	EnteroGrande	Bigint	No	Identificador del acto procesal
IdnEstadoOrigen	EnteroGrande	Bigint	Si	Identificador del estado de origen del expediente
IdnEstadoDestino	EnteroGrande	Bigint	Si	Identificador del estado de destino del expediente
Termino	EnteroGrande	Bigint	Si	Cantidad de días hábiles para realizar el acto procesal.

Tabla 10: nTramite

nActoProcesal				
Descripción: Actos procesales de los procedimientos.				
Atributo	**Dominio**	**Tipo de Dato**	**Nulo**	**Descripción**
IdnActoProcesal	EnteroGrande	Bigint	No	Identificador del acto procesal
ActoProcesal	Texto	Text	No	Acto Procesal.

Tabla 11: nActoProcesal

2.3 Acceso a Datos

Doctrine es un ORM que posee una poderosa capa de abstracción de Bases de Datos. Una de sus características es la opción de escribir consultas de Bases de Datos en un objeto apropiado orientado al dialecto SQL (Structured Query Language) y que se le denomina Lenguaje de Consulta de Doctrine o DQL del inglés Doctrine Query Language, inspirado por el Lenguaje de Consulta de Hibernate (HQL por sus siglas en inglés). Este proporciona a los desarrolladores una poderosa alternativa al SQL que mantiene la flexibilidad sin requerir duplicación de código innecesario. (13)

2.3.1 Principales características de Doctrine

Doctrine es un framework para el mapeo objeto – relacional para PHP que está dividido en dos capas principales, la DBAL del inglés Database Abstraction Layer y el ORM. La imagen que se muestra a continuación refleja cómo las capas de Doctrine trabajan juntos.

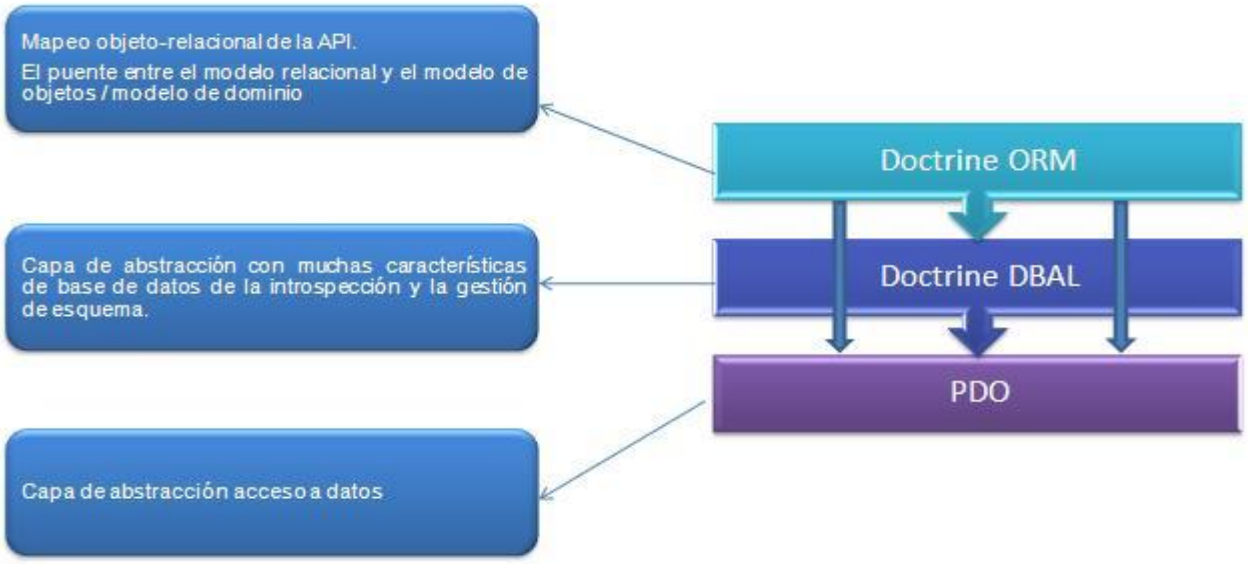


Figura 7: Estructura del Framework Doctrine

Doctrine Generator es una aplicación para la persistencia de objetos relacionales, basada en el ORM Doctrine, que ofrece funcionalidades de mapeo. Doctrine Generator fue diseñado con el principio de brindar a los desarrolladores que necesiten persistir su información mediante Doctrine, un ambiente de desarrollo amigable y fácil de usar que cubra sus necesidades respecto a la persistencia de esta información. La herramienta proporciona la posibilidad de generar la misma lógica de negocio que genera el framework Doctrine con un buen rendimiento en cuanto a funcionamiento corrigiendo algunos de sus problemas en las salidas como por ejemplo la ausencia de las relaciones entre las tablas del esquema con el que se está interactuando y la imposibilidad de personalizar estas salidas.

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

de datos, la herramienta posibilita conectividad con este gestor, permitiendo el mapeo de los esquemas existentes y generando los correspondientes ficheros. (14)

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

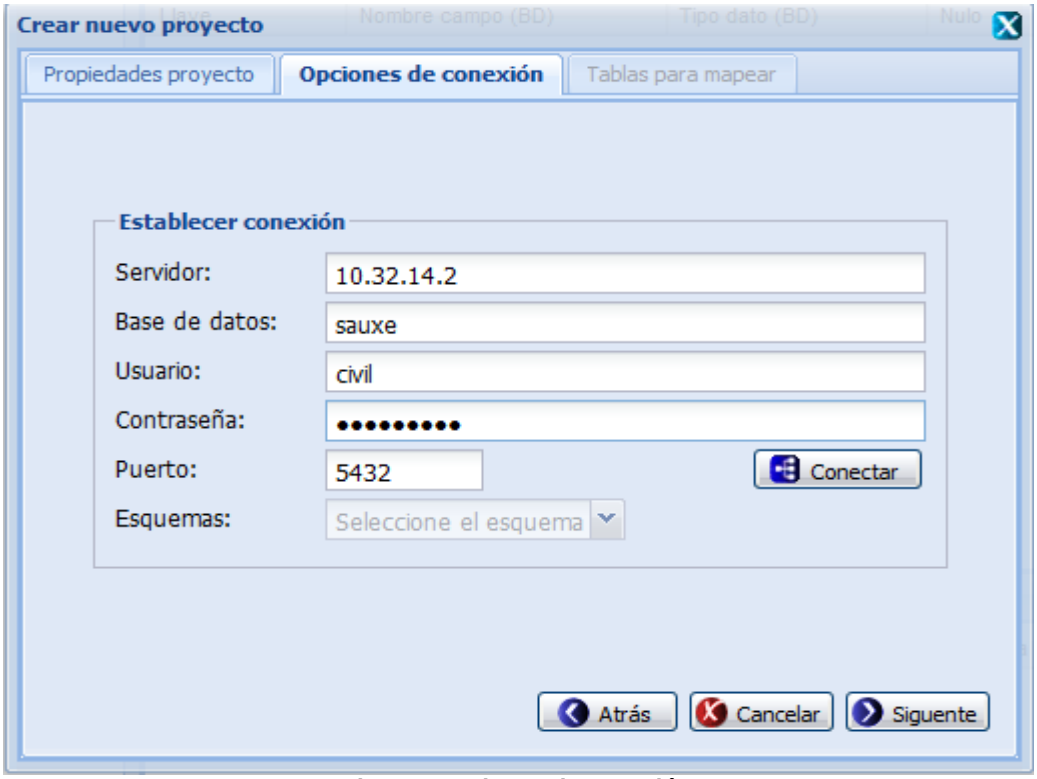


Figura8 Opciones de conexión

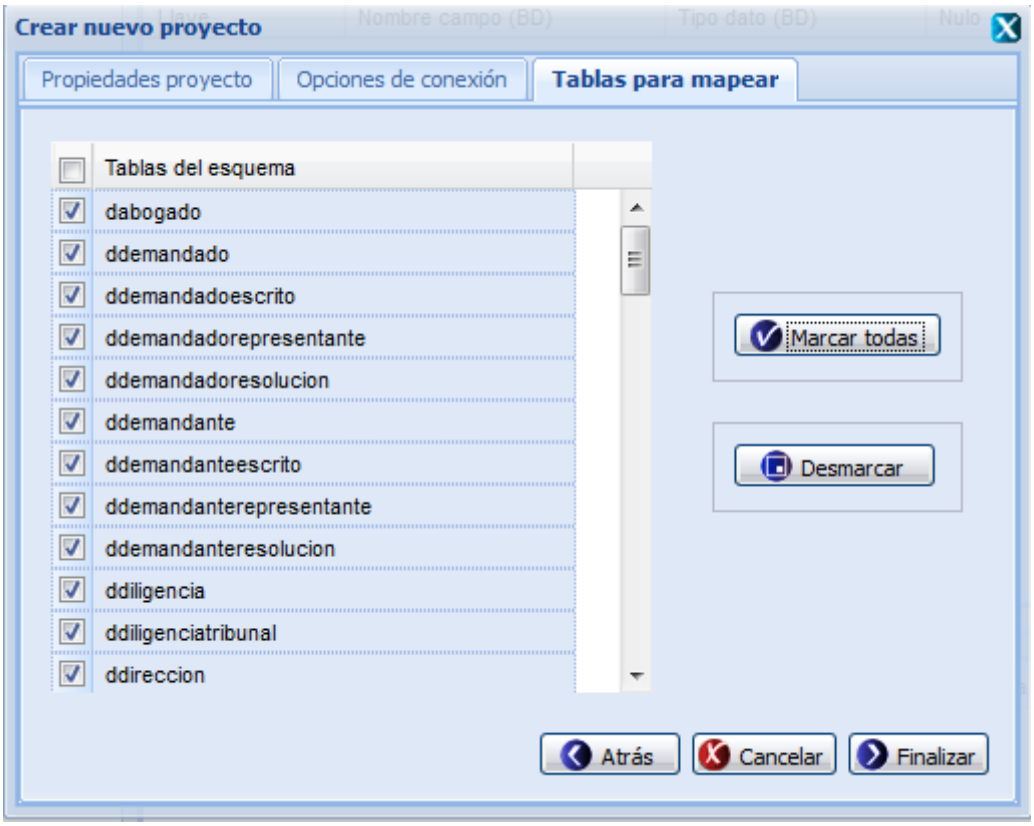


Figura9 Tablas para mapear

Este proyecto después es guardado (*.dg) y puede ser reeditado posteriormente. El sistema genera la siguiente estructura de carpeta y brinda la facilidad de seleccionar el lugar donde se generará:

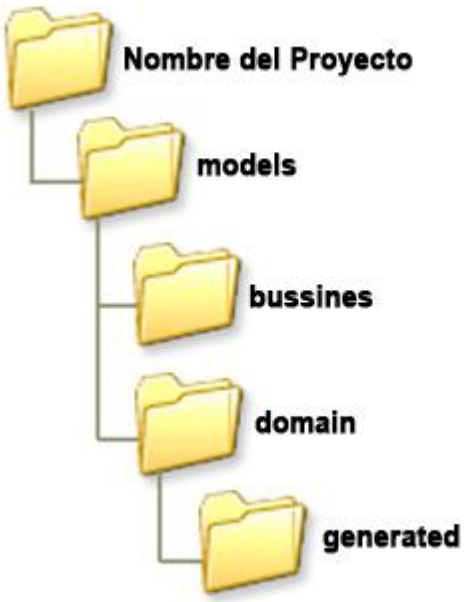


Figura 10 Estructura de carpetas generadas por Doctrine Mapeador

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

2.4 Optimización

La optimización es un aspecto de suma importancia, ya que esta actividad permite mejorar el rendimiento y que las bases de datos funcionen en las condiciones óptimas de acuerdo a sus características y propósitos.

Para optimizar el diseño físico de una base de datos relacional es necesario:

- Disminuir los tiempos de respuesta.
- Minimizar el espacio de almacenamiento.

- Conseguir la máxima seguridad de los datos.
- Optimizar el consumo de recursos.

Existen algunos **tipos de optimización** como son:

-Optimización basada en reglas

- Heurísticas generales basadas en la experiencia del administrador o de los diseñadores

-Optimización basada en costes

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

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

2.4.1Estrategia inicial de indexado

Las demoras del sistema ante operaciones que involucren un gran volumen de datos pueden ser reducidas. Es posible lograr optimizaciones ya sea por el tipo específico de gestor con que se manejen los datos y sus configuraciones puntuales, por el modelo de datos seleccionado, por configuraciones que se realicen sobre la base de datos, como por las optimizaciones en las consultas. Las técnicas de optimización pueden enfocarse entonces tanto en el nivel físico, por ejemplo, distribuyendo la información en distintos ficheros, discos, o incluso servidores, realizando un mayor número de operaciones en paralelo; como a la hora de proponer un diseño conceptual, seleccionando un modelo con el que se prevean realizar menos operaciones costosas.

Muchas veces en la práctica se deciden aplicar varias de estas alternativas juntas. Aunque la intensión no es mencionar cada una de las opciones disponibles para optimizar, sí se quisiera poner a consideración algunas de ellas.

Sobre una BD se realizarán, muchas veces, consultas de gran complejidad que solicitarán información que cumpla determinados criterios, es decir, los usuarios frecuentemente querrán especificar los valores

con los cuales se filtrarán los datos que deberán ser retornados. La mayoría de estas consultas incluirán, probablemente, operaciones de join entre tablas muy grandes, lo cual puede resultar extremadamente costoso. Para ganar en eficiencia a la hora de realizar estas operaciones se han investigado y creado técnicas especializadas que hoy ofrecen varios gestores, como los ***índices***.

Para entender qué es un índice y cuál es su utilidad, se puede hacer un símil con los índices de los libros. Si un libro no tuviera índice e interesara leer sobre un tema en específico, se tendría que recorrer cada página hasta encontrar lo buscado, llegando necesariamente hasta el final, pues no se podrían determinar cuándo se ha encontrado la última referencia al tópico de interés. Este proceso, definitivamente, haría consumir una cantidad considerable de tiempo. De manera similar, en las consultas que incluyen filtrar de acuerdo a uno o varios valores, se deben recorrer todas las filas de manera secuencial[7], buscando las que cumplen la condición. Si se tuviera una estructura que, al igual que un índice en un libro, guíe hasta encontrar las páginas de interés más rápido, serían más eficientes las búsquedas en el sistema. Una de las técnicas de las que se dispone en SQL para reducir los tiempos de respuesta es, precisamente, los índices. Sin embargo, para usarlos de forma efectiva primero se debe saber cómo funcionan. Un índice es una estructura física que permite un tipo de acceso alternativo al secuencial. Es creado a partir de una o varias columnas de una tabla, y, por lo general, es construido en forma de árbol balanceado ***(B-Tree)***. Al ser estructuras físicas, los índices van a tener un fichero asociado, en cuyas páginas se pueden almacenar uno o varios nodos del árbol. Cada uno de ellos apunta hacia otros nodos del árbol o hace referencia a las filas de la tabla. En cada nodo, los valores están ordenados, y los que se encuentran en un nodo hijo son menores o iguales que el valor en el nodo padre que le hace referencia. Los nodos que apuntan hacia las filas reciben el nombre de “páginas hojas”, y están enlazados entre sí: una página hoja apunta a otra hoja que contiene el próximo conjunto de valores.

Existe un tipo de índice con el cual se impone que los datos de la tabla estén ordenados en el nivel físico, y reciben el nombre de índices clusterizados (clustered index). Para cada tabla sólo se puede especificar un índice clusterizado, pues este afecta la forma en que son almacenadas las filas. Aquellos que no influyen en la organización física se denominan índices no clusterizados y varios pueden ser creados para una misma tabla. (15)

[7] Este proceso es denominado método de acceso secuencial (*sequentialaccessmethod*).

Las ventajas que tiene el uso de los índices están dadas, precisamente, por su estructura. Por ejemplo, las búsquedas de filas en las que un valor en particular aparezca no implican recorrer toda la tabla, sino que se utiliza la estructura arbórea del índice que se haya definido. Bajando desde la raíz del árbol, sólo es necesario desprenderse por una de las ramas hasta encontrar, en las páginas hojas, las referencias a las filas en el fichero. Con esto se consume menos tiempo en hallar el resultado y es menor la cantidad de veces que se accede al disco para leer. Se podría pensar entonces que la mejor opción es crear un índice por cada combinación de columnas. Sin embargo, sobre todas ellas en la práctica no se definen buenos criterios de búsqueda, por lo que no deberían crearse estas estructuras innecesariamente. Además, la creación de demasiados índices puede traer consecuencias no deseadas:

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

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

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

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

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

la columna (cardinalidad de la columna): el impacto de un índice es generalmente mayor mientras mayor sea la cardinalidad de la tabla y/o de la columna.

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

El estudio de los índices ha sido y continúa siendo un campo en desarrollo. A medida que surgen nuevas necesidades informativas y las consultas van ganando en complejidad, se hacen necesarias estas técnicas de optimización, con el fin de mejorar el comportamiento de los sistemas ante las solicitudes. Cada tipo de índice generalmente está enfocado a hacer eficientes las consultas, pero teniendo en cuenta los datos almacenados, su cantidad y variabilidad, factores que influyen a la hora de tomar la decisión de qué índices definir.

La solución de esta BD posee implementado un indexado, que trae por defecto el gestor PostgreSQL para la búsqueda de datos utilizando las llaves primarias y foráneas. Todas las llaves primarias, que son llaves subrogadas, poseen índices de tipo “B-Tree” (Árboles-B) lo que implica que cualquier búsqueda que se realice utilizando las llaves se optimizará mediante este método.

2.4.2 Mantenimiento de la base de datos (VACUUM)

Esta opción se basa en el comando VACUUM de Postgres. Este comando no es propio del estándar SQL92. Esta opción escaneará la BD o tabla por filas. Si una fila es modificada o eliminada, su contenido previo no será reemplazado, pero será marcado como no válido y no ser podrá usar. El nuevo dato será insertado en la BD, este si será accesible. Se necesita regularmente realizar una recolección de basura, para asegurarse que la BD no contenga demasiados datos sin uso y se afecte el rendimiento de la BD.

VACUUM sirve para dos propósitos en Postgres como medio para reclamar almacenamiento, y también para recolectar información para el optimizador. VACUUM abre cada clase en la base de datos, limpia los registros de transacciones ya pasadas y actualiza las estadísticas en los catálogos del sistema. Las

estadísticas mantenidas incluyen el número de tuplas y el número de páginas almacenadas en todas las clases. La ejecución de VACUUM periódicamente aumentará la velocidad de la base de datos al procesar las consultas del usuario.

El comando VACUUM es utilizado en la solución propuesta después de una considerable cantidad de modificaciones sobre una tabla y cuando no existe utilización sobre la base de datos, por ejemplo en horas nocturnas fuera del horario de trabajo.

2.5 Patrones de diseño

Desde el surgimiento de la humanidad los seres humanos han estado buscando en la naturaleza características que repetidamente se presentan entre los fenómenos naturales. De igual manera en el diseño de las bases de datos y en el modelado de datos en general se presentan elementos repetitivos en disimiles modelos, los que correctamente identificados pasan a ser patrones de diseño.

Por lo tanto se pude definir un patrón como el fragmento de un modelo que es recurrente. Un patrón es una solución a un problema específico que se ha mantenido a pesar del tiempo. (16)

2.5.1 Máquina de estado para escenarios

Este modelo representa la ocurrencia del cambio de estado en un escenario de una entidad dada, por lo tanto considera el tiempo y la persistencia del mismo en las tablas resultantes. También representa la ocurrencia de un estímulo en una fecha y los estados por los que ha pasado, caracterizados por la fecha de inicio y la fecha fin. (17)

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

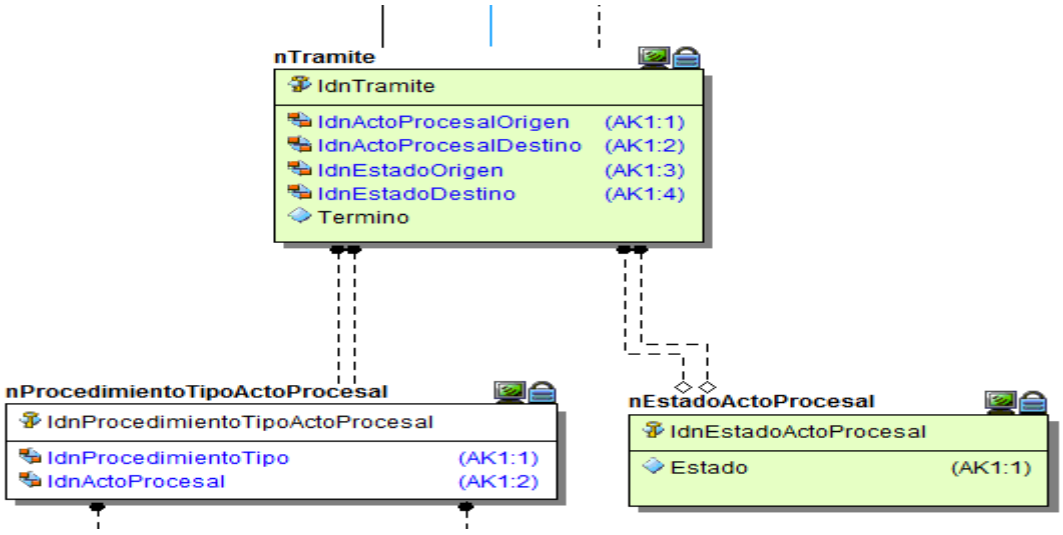


Figura 11 Tabla en la que se hace se pone de manifiesto el patrón Máquina de estado para escenarios

2.5.2 Llaves subrogadas

Este patrón es muy utilizado pues se decide generar una llave primara única para cada entidad en vez de usar un atributo identificador en el contexto dado.
Normalmente se usa enteros en columnas identity o GUID (Global UniqueIdentifier) que están demostrados que no se repiten o con una probabilidad extremadamente baja.

Permite que las tablas sean más fáciles de consultar por el identificador dado que se conoce el mismo tipo de todos en cada tabla.

Conclusiones Parciales

En el presente capítulo se ha descrito detalladamente la descripción de la solución desarrollada para la base de datos del módulo Civil y la arquitectura propuesta para su desarrollo, junto con las configuraciones de la base de datos, la seguridad para interactuar y trabajar con las herramientas y gestores, así como la explicación del artefacto entorno de desarrollo el cual define las pautas a seguir para un buen diseño del modelo de datos. Se plasman las descripciones de las principales tablas del modelo lógico, además de los patrones utilizados para el diseño de la misma. Se muestran los pasos a seguir para una buena optimización de la base de datos a través del uso de índices y mantenimiento gracias al comando VACUUM. Se explica además cómo fue generada la capa de acceso a datos.

CAPÍTULO 3: VALIDACIÓN DE LA SOLUCIÓN

Introducción

En el presente capítulo se realiza la validación teórica y funcional del diseño de la base de datos del módulo Civil del proyecto “Sistema para los Tribunales Cubanos”. Se abordaran aspectos de particular importancia entre los que destacan: la integridad, normalización y pruebas a la BD. Además se realizan pruebas de rendimiento o de carga intensiva centradas en comprobar el correcto funcionamiento de la BD.

3.1 Validación teórica del diseño

Se debe tener primeramente un diseño acorde con las funcionalidades que requiere el sistema, pero esto no es suficiente; es muy importante tener en cuenta aspectos claves como son la consistencia, integridad y seguridad de los datos almacenados así como la normalización de la base de datos, garantizado de esta manera el acceso a los datos por el personal autorizado y que la información no se altere como consecuencia de acciones no controladas.

3.1.1 Restricciones de integridad de la Base de Datos

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

La integridad de datos se distribuye en los siguientes tipos:

- Integridad de entidad: la clave primaria de una entidad no puede tener valores nulos y siempre deberá ser única, por ejemplo el CI en la tabla dPersonaNatural.
- Integridad de dominio: restringe los valores que puede tomar un atributo respecto a su dominio, por ejemplo, la tabla nTipoProcedimiento sólo puede tomar los valores de los procedimientos que existen en la materia Civil (Ordinario, Sumarios, Amparo, Divorcio, Alimentos, Incidentes en Sumario de Alimentos, Incidentes en Divorcio, Procesos Sucesorios y de Jurisdicción Voluntaria).

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

3.1.1.1 Integridad de Entidad

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

Es por esto que en muchas de las tablas de cada uno de los esquemas de la base de datos del sistema para los Tribunales Populares Cubanos generan las llaves de forma automática usando secuencias, garantizando así que no se cometan errores y uniformidad en la creación de la llaves.

3.1.1.2 Integridad de Dominio

La integridad de dominio viene dada por el conjunto de valores posibles para un atributo determinado. Puede exigir la integridad de dominio restringiendo los valores de un atributo mediante tipos de datos, reglas y restricciones CHECK, definiciones DEFAULT, definiciones NOT NULL. Un dominio de valores puede estar asociado a cada atributo. Los límites de dominio son la forma más elemental de restricciones de integridad. Son fáciles de probar en el sistema siempre que se introduce un nuevo dato.

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

Dominio	Tipo de Datos	Campo	Restricción
Binario	BINARY	Documentos, imágenes, etc.	
Bool	BIT	Verdadero o Falso	`@var = TRUE or @var = FALSE`
Contraseña	TEXT	Contraseña de usuario	`@varlike '%_____%'`
EMail	TEXT	Correo electrónico	`@varlike '%_@___%'`
Entero	INTEGER	Números enteros positivos	`@var> 0`
FechaHora	DATETIME	Fecha y hora	A nivel de tabla comparando las fechas: `FechaInicio<= FechaFin`
Nombre	VARCHAR(30)	Nombre hasta 30 caracteres	
Numero	NUMERIC	Valores numéricos y decimales	`@var> 0`
NumeroCI	VARCHAR(11)	Número de carnet de identidad	`@varlike '_____'`
Texto	TEXT	Texto en general	

Figura 12 Diccionario de datos

3.1.1.3 Integridad Referencial

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

La integridad referencial permite que la relación entre dos tablas permanezca sincronizada durante las operaciones de actualización y eliminación, para lograrlo se utilizan las sentencias SQL ON DELETE CASCADE o RESTRICT y ON UPDATE CASCADE o RESTRICT en dependencia de la relación que exista entre las tablas implicadas.

Un ejemplo de lo antes expuesto es que se tiene una tabla dPersona quien tiene como llave primaria IdPersona, además se encuentra la tabla dDireccion quien tiene como llave primaria IdDireccion. La relación que se establece entre estas tablas es de uno a muchos (ya que una persona puede tener varias

direcciones) razón por la cual IdPersona pasa para la tabla dDireccion como llave foránea. De esta forma se conocería a que persona pertenece una determinada dirección.

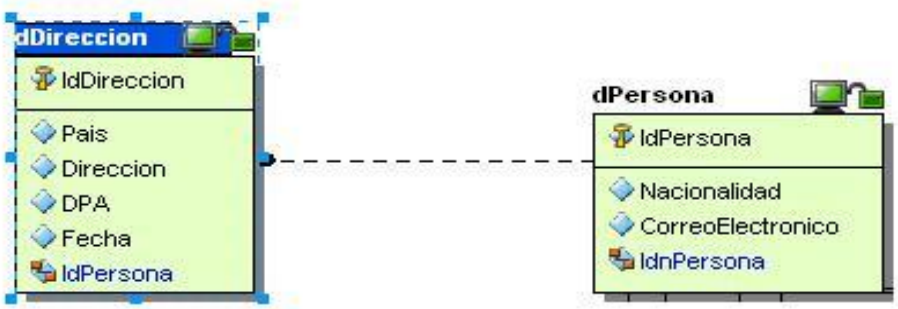


Figura 13 Integridad referencial entre dos tablas

3.1.1.4 Datos Requeridos

Se define efectuando que la declaración de una columna es NOT NULL cuando la tabla que contiene las columnas se crea por primera vez, como parte de la sentencia CREATE TABLE.

En la mayoría de los casos resultaría inútil declarar que un campo acepte valores nulos si se quiere guardar información importante en el mismo, sin embargo en otros casos resulta necesario, por ejemplo, la tabla dPersonaNatural contiene el atributo Ci y Pasaporte, los mismos aceptan valores nulos pues una persona se puede identificar con el carnet de identidad o pasaporte dependiendo de su nacionalidad.

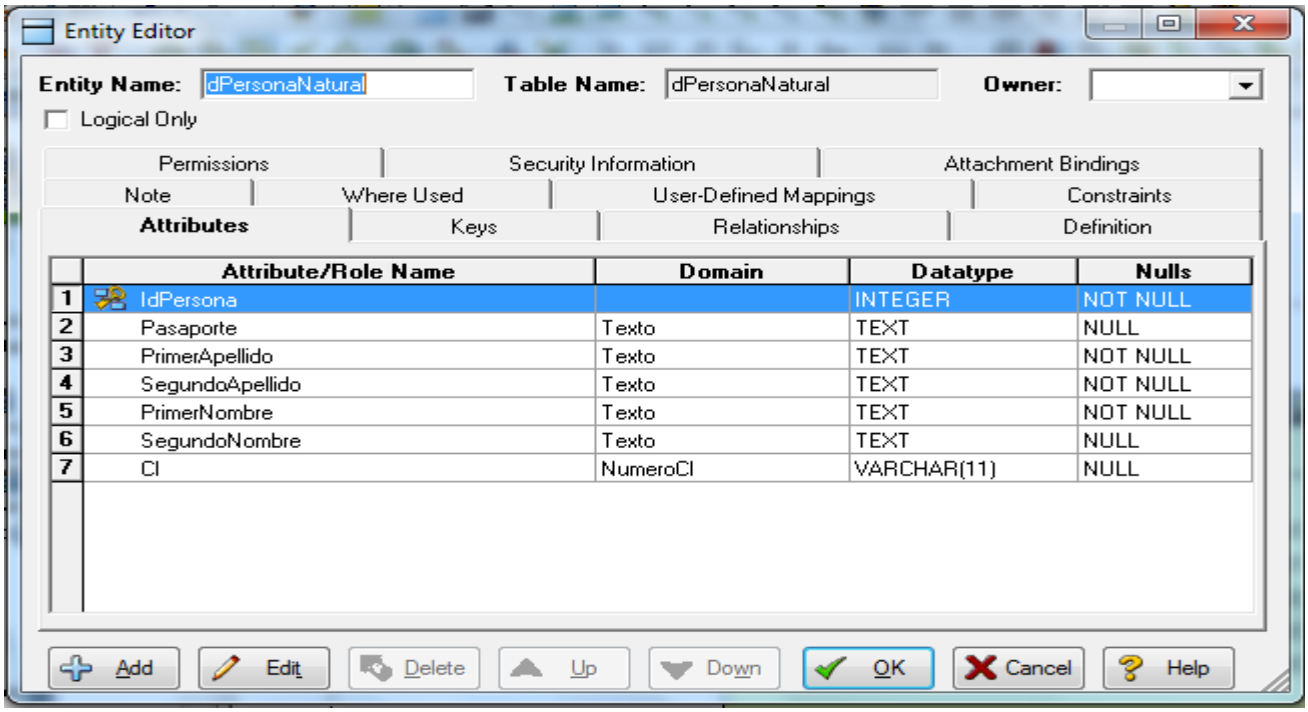


Figura 14 Datos requeridos en una tabla

3.2 Normalización

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

En el proceso de normalización existen varios niveles, pero los tres primeros son los más usados. Las reglas de normalización se relacionan entre ellas de forma dependiente, lo que indica que para que una base de datos se encuentre en una determinada forma normal, primeramente tiene que cumplir con las reglas de los niveles inferiores de normalización.

El esquema de relación del módulo está en 1ra Forma Normal puesto que se puede asegurar que cada tupla contiene exactamente un valor para cada atributo de las tablas de la base de datos, o sea, no existen campos multievaluados. También se cumple que el esquema de relación está en 2da Forma Normal, pues primeramente se encuentran en 1ra Forma Normal, y además todos los atributos que no son claves en las tablas, dependen totalmente de la llave primaria. Finalmente puede plantearse que el esquema de relación se encuentran en 3ra Forma Normal, pues se encuentra en 2da Forma Normal, y además no existen dependencias transitivas de atributos no primos. Se puede afirmar entonces que el diseño del módulo de la base de datos propuesto se encuentra normalizado hasta Tercera Forma Normal, lográndose de esta forma mayor independencia entre los datos y evitándose anomalías al trabajar con los datos.

3.3 Validación Funcional

Para comprobar el rendimiento de la BD y su correcto funcionamiento es necesario realizar determinadas pruebas, asegurando así que la misma cumpla con los requisitos definidos. Las pruebas persiguen el objetivo de simular una carga de producción real y observar cómo se comporta la BD ante esta carga, comprobando que el sistema satisface las necesidades además de que esto se realice sin violar la integridad de los datos.

3.3.1 Pruebas de volumen y rendimiento

Las pruebas de volumen son pruebas típicas de entornos que utilicen bases de datos. Las mismas se realizan para analizar el comportamiento del sistema o base de datos con volúmenes de datos almacenados lo más similar posible a los esperados en la explotación real del sistema. Para el sistema en cuestión la BD se pobló con datos aleatorios generados por una herramienta llamada EMS Data Generator, esta herramienta colma a la base de datos de una cantidad determinada de datos previamente establecida por quién esté realizando el llenado de datos en este caso, el propio desarrollador.

Se configuró esta generación con datos arbitrarios, pero coincidentes en cuanto a sus tipos y volúmenes con los datos reales que maneja la entidad. El uso de este generador se pudiera considerar una prueba más, ya que si existen inconsistencia en el diseño de la Base de Datos, este no comienza el poblado de datos hasta tanto no quede un correcto diseño.

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

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

Nombre de la tabla	**Cantidad de tuplas**
dPersona	10000
dPersonaNatural	2140
dPersonaJuridica	2111
dDireccion	10000
dPersonaParte	1580
dTelefono	4971

Tabla 12: Tablas pobladas

Consulta 1

```
explain analyze SELECT
dpersona.nacionalidad
FROM
civil.dpersona,
civil.dpersonanatural
WHERE
  dpersona.idpersona = dpersonanatural.idpersona AND
  dpersonanatural.ci = '986316438'
```

Resultados arrojados

Panel de Salida

Salida de datos | Comentar | Mensajes | Historial

	QUERY PLAN text
1	Nested Loop (cost=0.00..73.03 rows=1 width=9) (actual time=0.197..0.616 rows=1 loops=1)
2	-> Seq Scan on dpersonanatural (cost=0.00..64.75 rows=1 width=8) (actual time=0.176..0.593 rows=1 loops=1)
3	Filter: ((ci)::text = '986316438'::text)
4	-> Index Scan using pk_dpersona on dpersona (cost=0.00..8.27 rows=1 width=17) (actual time=0.017..0.018 rows=1 loops=1)
5	Index Cond: (dpersona.idpersona = dpersonanatural.idpersona)
6	Total runtime: 0.664 ms

Figura 12: Consulta donde no se utiliza índice

Panel de Salida

Salida de datos | Comentar | Mensajes | Historial

	QUERY PLAN text
1	Nested Loop (cost=0.00..16.55 rows=1 width=9) (actual time=0.037..0.039 rows=1 loops=1)
2	-> Index Scan using ci on dpersonanatural (cost=0.00..8.27 rows=1 width=8) (actual time=0.025..0.026 rows=1 loops=1)
3	Index Cond: ((ci)::text = '986316438'::text)
4	-> Index Scan using pk_dpersona on dpersona (cost=0.00..8.27 rows=1 width=17) (actual time=0.007..0.008 rows=1 loops=1)
5	Index Cond: (dpersona.idpersona = dpersonanatural.idpersona)
6	Total runtime: 0.093 ms

Figura 13: Consulta utilizando índices

Consulta 2

explain analyze SELECT

dpersonanatural.primernombre,

dpersonanatural.primerapellido,

dpersonanatural.segundoapellido,

dpersonaparte.profesion,

dtelefono.numero,

dpersona.nacionalidad,

dpersona.correoelectronico,

ddireccion.pais,

ddireccion.direccion,

ddireccion.dpa,

dpersonaparte.naturalde,

dpersonanatural.segundonombre,

dpersonanatural.ci,

dpersonanatural.pasaporte

FROM

civil.dpersona,

civil.dpersonanatural,

civil.dtelefono,

civil.dpersonaparte,

civil.ddireccion

WHERE

dpersona.idpersona = ddireccion.idpersona AND

dpersonanatural.idpersona = dpersona.idpersona AND

dtelefono.idpersona = dpersona.idpersona AND

dpersonaparte.idpersona = dpersonanatural.idpersona AND

dpersonanatural.primernombre = 'zrvp'

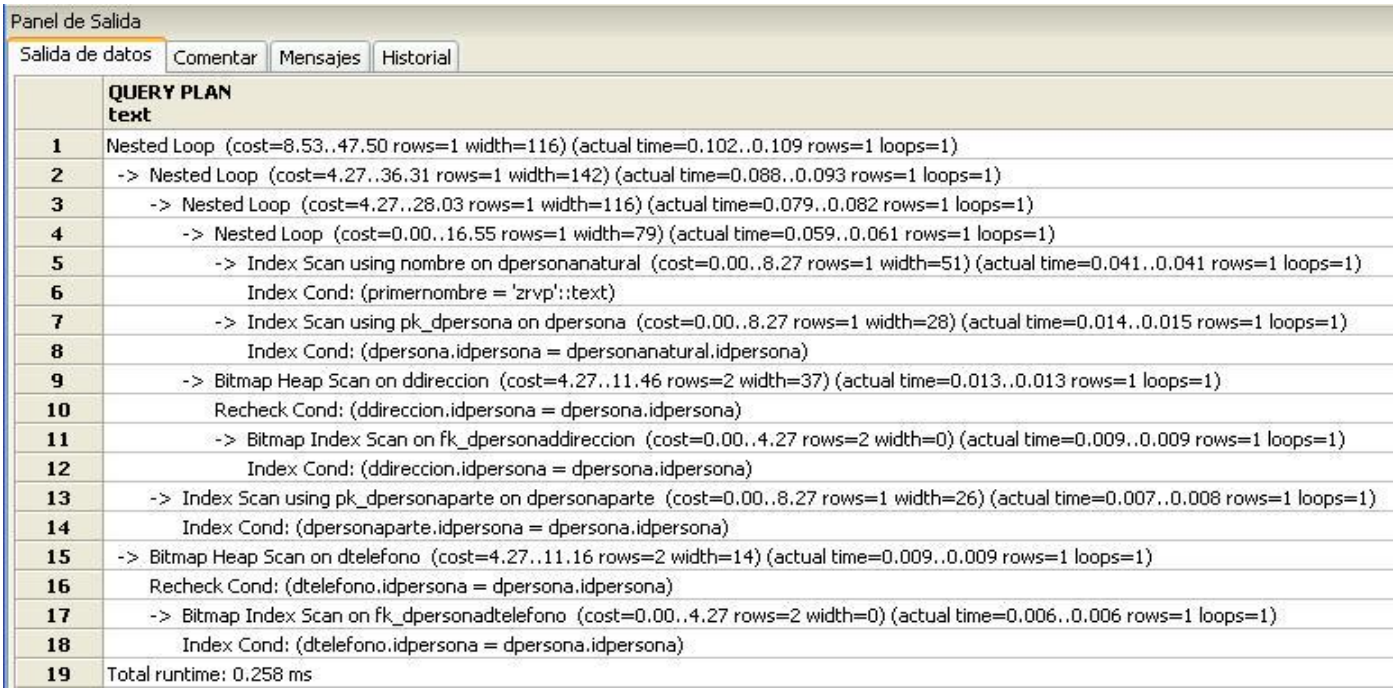
Resultados arrojados

Panel de Salida

Salida de datos | Comentar | Mensajes | Historial

	QUERY PLAN text
1	Nested Loop (cost=4.27..104.81 rows=1 width=116) (actual time=0.152..0.645 rows=1 loops=1)
2	-> Nested Loop (cost=0.00..93.62 rows=1 width=142) (actual time=0.132..0.623 rows=1 loops=1)
3	-> Nested Loop (cost=0.00..85.34 rows=1 width=116) (actual time=0.121..0.611 rows=1 loops=1)
4	-> Nested Loop (cost=0.00..73.03 rows=1 width=79) (actual time=0.110..0.599 rows=1 loops=1)
5	-> Seq Scan on dpersonanatural (cost=0.00..64.75 rows=1 width=51) (actual time=0.092..0.579 rows=1 loops=1)
6	Filter: (primernombre = 'zrvp'::text)
7	-> Index Scan using pk_dpersona on dpersona (cost=0.00..8.27 rows=1 width=28) (actual time=0.015..0.015 rows=1 loops=1)
8	Index Cond: (dpersona.idpersona = dpersonanatural.idpersona)
9	-> Index Scan using fk_dpersonaddireccion on ddireccion (cost=0.00..12.29 rows=2 width=37) (actual time=0.008..0.009 rows=1 loops=1)
10	Index Cond: (ddireccion.idpersona = dpersona.idpersona)
11	-> Index Scan using pk_dpersonaparte on dpersonaparte (cost=0.00..8.27 rows=1 width=26) (actual time=0.009..0.009 rows=1 loops=1)
12	Index Cond: (dpersonaparte.idpersona = dpersona.idpersona)
13	-> Bitmap Heap Scan on dtelefono (cost=4.27..11.16 rows=2 width=14) (actual time=0.013..0.013 rows=1 loops=1)
14	Recheck Cond: (dtelefono.idpersona = dpersona.idpersona)
15	-> Bitmap Index Scan on fk_dpersonadtelefono (cost=0.00..4.27 rows=2 width=0) (actual time=0.009..0.009 rows=1 loops=1)
16	Index Cond: (dtelefono.idpersona = dpersona.idpersona)
17	Total runtime: 0.800 ms

Figura 14: Consulta donde no se utiliza índice



Panel de Salida

Salida de datos | Comentar | Mensajes | Historial

	QUERY PLAN text
1	Nested Loop (cost=8.53..47.50 rows=1 width=116) (actual time=0.102..0.109 rows=1 loops=1)
2	-> Nested Loop (cost=4.27..36.31 rows=1 width=142) (actual time=0.088..0.093 rows=1 loops=1)
3	-> Nested Loop (cost=4.27..28.03 rows=1 width=116) (actual time=0.079..0.082 rows=1 loops=1)
4	-> Nested Loop (cost=0.00..16.55 rows=1 width=79) (actual time=0.059..0.061 rows=1 loops=1)
5	-> Index Scan using nombre on dpersonanatural (cost=0.00..8.27 rows=1 width=51) (actual time=0.041..0.041 rows=1 loops=1)
6	Index Cond: (primernombre = 'zrvp'::text)
7	-> Index Scan using pk_dpersona on dpersona (cost=0.00..8.27 rows=1 width=28) (actual time=0.014..0.015 rows=1 loops=1)
8	Index Cond: (dpersona.idpersona = dpersonanatural.idpersona)
9	-> Bitmap Heap Scan on ddireccion (cost=4.27..11.46 rows=2 width=37) (actual time=0.013..0.013 rows=1 loops=1)
10	Recheck Cond: (ddireccion.idpersona = dpersona.idpersona)
11	-> Bitmap Index Scan on fk_dpersonaddireccion (cost=0.00..4.27 rows=2 width=0) (actual time=0.009..0.009 rows=1 loops=1)
12	Index Cond: (ddireccion.idpersona = dpersona.idpersona)
13	-> Index Scan using pk_dpersonaparte on dpersonaparte (cost=0.00..8.27 rows=1 width=26) (actual time=0.007..0.008 rows=1 loops=1)
14	Index Cond: (dpersonaparte.idpersona = dpersona.idpersona)
15	-> Bitmap Heap Scan on dtelefono (cost=4.27..11.16 rows=2 width=14) (actual time=0.009..0.009 rows=1 loops=1)
16	Recheck Cond: (dtelefono.idpersona = dpersona.idpersona)
17	-> Bitmap Index Scan on fk_dpersonadtelefono (cost=0.00..4.27 rows=2 width=0) (actual time=0.006..0.006 rows=1 loops=1)
18	Index Cond: (dtelefono.idpersona = dpersona.idpersona)
19	Total runtime: 0.258 ms

Figura 15: Consulta utilizando índice

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

Para medir la eficiencia de los resultados se establecieron los siguientes intervalos para tiempo de respuesta:

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

Tabla 13: Intervalos para tiempos de respuesta

Luego de realizadas las pruebas y valorados los resultados se puede afirmar que las respuestas a las solicitudes de los usuarios, son relativamente rápidas y teniendo en cuenta la comparación entre las que utilizan índices y las que no lo hacen, se puede definir que los índices optimizan las consultas a tablas que manejan una gran cantidad de datos.

3.3.2 Pruebas de estrés y carga

Las pruebas de rendimiento y stress son de las pruebas necesarias que se le realizan a una base de datos para asegurar el posterior funcionamiento del sistema y garantizar un correcto y eficiente servicio, carente de fallas, errores y retrasos a sus usuarios. 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.

Se realizaron dos pruebas a la base de datos. 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.
- **Periodo 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 el test.

Propiedades de los hilos	**Prueba 1**	**Prueba 2**
Número de hilos	50	200
Período de subida(en segundos)	1	3
Contador del bucle	10	10

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

- **Label**: El nombre de la muestra (conjunto de muestras).
- **# Muestras**: El número de muestras de peticiones JDBC.

- **Media**: El tiempo medio transcurrido para un conjunto de resultados.
- **Mediana**: Mediana aritmética.

DRAE: f. Mat. Elemento de una serie ordenada de valores crecientes de forma que la divide en dos partes iguales, superiores e inferiores a él.

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

	Prueba 1	**Prueba 2**
Label	Petición JDBC	Petición JDBC
No. Muestras	500	2000
Media	690	1445
Mediana	375	265
Mínimo	15	15
Máximo	4406	10630
% Error	0	1,05
Rendimiento	87,7/seg	81,8/seg

Tabla 14: Resultados de pruebas con JMeter

Como se muestra en los resultados de las pruebas realizadas para un número de muestras igual a 500 y 2000 respectivamente el tiempo de respuesta fue de menos de un segundo para la primera y para la segunda de un segundo y medio aproximadamente por lo que se puede afirmar que la base de datos responde con la rapidez requerida a un grupo de solicitudes concurrentes cumpliendo de esta manera con los requisitos no funcionales. En los resultados también se puede apreciar que la primera prueba se ejecutó correctamente con un por ciento de errores igual a cero y con un rendimiento estimado de 87,7 consultas por segundo. Mientras que para la segunda prueba el por ciento de error varió arrojando un resultado igual a 1,05 teniendo en cuenta que el número de muestras es cuatro veces mayor que en la

anterior por lo que se puede asumir que es un buen por ciento para dicha cantidad, además que tuvo un rendimiento con un valor estimado de 81,8 consultas por segundo.

Conclusiones Parciales

Al concluir este capítulo se puede plantear que se han cumplido los objetivos propuestos para el mismo, superando al mismo tiempo las expectativas esperadas. La validación del diseño culminó con la valoración de aspectos importantes como la integridad de la información almacenada, definiéndose varias restricciones que permitieran garantizar la eficiencia de la misma. Además la normalización de la Base de Datos que posibilitó saber hasta qué punto pueden ser tratados los datos evitando anomalías al trabajar con los. La validación se efectuó poniendo en práctica las experiencias adquiridas como diseñadores de BD, con el uso de herramientas como el EMS Data Generator, JMeter y del EXPLAIN ANALIZE para optimizar consultas que consumen muchos recursos.

CONCLUSIONES

Los Sistemas Gestores de Bases de Datos son cada día más usados en el cúmulo de aplicaciones que se desarrollan, por lo que la implementación de los diseños en los mismos deben ser mejores y más eficaces ya que los usuarios cada vez necesitan más recursos en tecnología, es por eso que surgen las evoluciones de sistemas, y por ende de las bases de datos.

Concluido el presente trabajo de investigación se puede afirmar que se ha cumplido con el objetivo del mismo a través del diseño e implementación de una BD que permitió centralizar la información del subsistema planificación material y financiera. Terminada la investigación se tienen las siguientes conclusiones:

- Durante la investigación se realizó un estudio acerca de los elementos teóricos principales relacionados con el diseño de base de datos que favoreció el desarrollo del trabajo permitiendo sentar las bases para un correcto diseño de BD.
- Se realizó el diseño de la BD, el cual cumple con los requerimientos especificados, aplicando diversas soluciones que contribuyeron a mejorar el mismo.
- Se implementó el acceso a datos, que permitió transformar el modelo entidad-relación al lenguaje de definición de datos del gestor correspondiente, en este caso el modelo relacional, estableciendo las estructuras de almacenamiento adecuadas.
- Se analizó el comportamiento de la BD ante situaciones determinadas a través de diferentes pruebas, a las cuales respondió satisfactoriamente.

De manera general se obtuvo una BD que permite el almacenamiento de la información requerida, permitiendo mayor organización y control de la misma.

RECOMENDACIONES

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

- Extender el modelo utilizado al resto de los proyectos de la universidad.
- Utilizar la propuesta de diseño de la base de datos presentada en este trabajo en el proyecto Sistema para los Tribunales Cubanos.
- Realizar un profundo estudio de las técnicas de optimización, ya sean las descritas u otras, aplicadas sobre las base de datos.
- Continuar con el desarrollo de la base de datos acorde al resto de las fases propuestas por el Modelo de Desarrollo de Bases de Datos.
- Proporcionar mantenimiento y soporte regular a la base de datos enfocados a su mejor funcionamiento.
- Proponer la utilización de la herramienta Chrono utilizada por el proyecto RAP para la configuración del clúster garantizando la replicación de los datos.

BIBLIOGRAFÍA

1. **Abraham Sotelo Nava.** Fundación Ciudad Política: liderar el desarrollo de la ciencia política para mejorar la sociedad global. [En línea] 2001. [Citado el: 22 de Enero de 2011.] http://www.ciudadpolitica.com/modules/news/article.php?com_mode=thread&com_order=1&storyid=8.

2. Cinvestav. *Coordinación General de Servicios de Cómputo Académico.* [En línea] [Citado el: 25 de Enero de 2011.] http://www.cinvestav.mx/Portals/0/cgsca/E-Gobierno.pdf.

3. **Riestra, Ema.** Informatica Juridica. [En línea] [Citado el: 30 de Enero de 2011.] http://www.slideshare.net/nspascuasc/informatica-juridica-2203562.

4. **Trujillo, Nixa Tatiana.** INFORMÁTICA JURÍDICA. [En línea] [Citado el: 30 de Enero de 2011.] http://www.slideshare.net/nspascuasc/informatica-juridica-2203562.

5. DERECHOINCREIBLE . [En línea] [Citado el: 30 de Enero de 2011.] http://derechoincreible.blogspot.com/.

6. **Díez, Mª. Luisa Alvite.** *EVOLUCIÓN DE LAS BASES DE DATOS JURÍDICAS EN ESPAÑA.* España : s.n., 2004.

7. **Barceló, José Manuel Morejón.** *GUÍA REFERATIVA PARA EL DISEÑO DE BASE DE DATOS DEL SISTEMA DE GESTIÓN CEDRUX.* 2010.

8. **Hansen, Gary W.** *Diseño y Administración de base de datos. .* 1997.

9. **Ing.Rodríguez, Alain Osorio.** Modelo de Desarrollo de Bases de Datos para Procesos de Desarrollo de Software. Ciudad de la Habana : s.n., 2009.

10. **Venezuela, Universidad Central de.** *Taller 3 Integridad.*

11. Embarcadero. [En línea] 2010. [Citado el: 19 de mayo de 2010.] http://www.embarcadero.com/.

12. **Manel Pérez Mata .** TecnoRetales. [En línea] 3 de julio de 2009. [Citado el: 12 de 6 de 2011.] http://www.tecnoretales.com/programacion/que-es-doctrine-orm/.

13. Symfony. *The symfony and Doctrine book.* [En línea] http://www.symfony-project.org/doctrine/1_2/es/01-Getting-Started.

14. **René R. Bauta Camejo1, Ariel Torres Galvez2.** *GENERCIÓN DE FICHEROS DE MAPEO MEDIANTE EL USO DE UNA.* 2010.

15. **England, K. and G. Powell.** *Performance Optimization and Tuning Handbook.* Estados Unido : s.n., 2007.

16. “PATRONES DE DISEÑO:UNA HERRAMIENTA DE DESARROLLO DE SOFTWARE Y SU APLICACION. [En línea] http://repositorio.puce.edu.ec/bitstream/22000/1117/1/T-PUCE-0616.pdf.

17. EcuRed. [En línea] http://www.ecured.cu/index.php/Patrones_de_dise%C3%B1o_de_bases_de_datos.

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

19. **AmbySoft Copyright © 2002-2009 Scott W. Ambler.** Techniques for Successful Evolutionary/Agile Database Development. *Agile Data Home Page.* [En línea] http://www.agiledata.org/.

20. **Hassan, Ahmed M. y Elssamadisy, Amr.** *Extreme Programming and Database Administration: Problems, Solutions, and Issues.*

21. **Ambysoft.** Home Page. *The Agile Unified Process.* [En línea] Ambysoft Inc. Copyright ©, 2005 - 2006. http://www.ambysoft.com/unifiedprocess/agileUP.html.

22. **AmbySoft Copyright © 2002-2009 Scott W. Ambler.** Agile Data Method. [En línea] http://www.agiledata.org/essays/vision.html.

23. —. Evolutionary/Agile Database Best Practices. [En línea] http://www.agiledata.org/essays/bestPractices.html.

24. **© Copyright 2009 IEEE – All Rights Reserved.** *IEEE.* [En línea] http://www.ieee.org/portal/site.

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

26. **PROAÑO, ING. DIEGO JAVIER BURBANO.** *ANALISIS COMPARATIVO DE BASES DE DATOS DE CODIGO ABIERTO.* Quito,Ecuador : s.n., 2006.

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

28. **Rodríguez, Alexei Domínguez.** *Diseño e implementación de la Base de Datos del Sistema de planificación del Entrenamiento deportivo de Judo.* Ciudad de la Habana : s.n., 2009.

29. **Alfaro, Félix Murillo.** *Herramientas CASE.* 1999.

30. PosgreSQL. [En línea] Copyright © 1996 – 2010. [Citado el: 19 de abril de 2010.] http://www.postgresql.org/docs/8.0/interactive/plpgsql.html.

31. **Solano, Orlando.** Slideshare. [En línea] 2003. [Citado el: 14 de mayo de 2010.] http://www.slideshare.net/guest9ca8c4/informatica-juridica-de-gestion-97-2003.

32. **Alarcón, José Manuel H.** *Administración de SGBD PostgreSQL.* 2006.

33. **Alburquerque, Amado.** *Sistema Integrador de Gestión Estadística (SIGE).* Ciudad de la Habana : Universidad de las Ciencias Informáticas, 2007.

34. **Alvarez, Sara.** *Desarrolloweb.* [En línea] 14 de agosto de 2007. [Citado el: 9 de febrero de 2010.] http://www.desarrolloweb.com/articulos/modelos-base-datos.html.

35. **B, Orlando Solano.** *Manual de informática Jurídica.* Bogotá, Colombia : Ediciones Jurídicas Gustavo Ibáñez, 1997.

36. **Benítez, Jorge Parra.** *Informática Jurídica y Derecho Informático.* Medellín, Colombia : Señal Editora, 1994.

37. **Judicial, Centro de Información.** CIJ . *Centro de Información Judicial.* [En línea] 30 de noviembre de 2008. [Citado el: 10 de febrero de 2010.] http://www.cij.gov.ar/nota-257-Avanza-la-informatizacion-de-la-gestion-judicial-en-el-pais.html.

38. **Erich Mario Gómez Pérez, Ariel Torres Gálvez.** *Administración y optimización de un Sistema de Base de Datos Descentralizado, en PostgreSQL.* Ciudad de la Habana : Universidad de las Ciencias Informáticas, 2008.

39. **Fábregas, Yuniesky y Fernández, Daniel.** *Administración, configuración y optimización de un Sistema de Base de Datos Descentralizado en Oracle Database 10g release 2.* Ciudad de la Habana : Universidad de las Ciencias Informáticas, 2007.

40. **Florez, Fernando Jordan.** *La Informática Jurídica (teoría y práctica).* Bogotá, Colombia : Editora Guadalupe LTDA, 1987.

41. **Marqués, Merche.** *Base de datos Orientado a Objetos. Diseño de Sistemas de Bases de Datos.* España : s.n., 2002.

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

43. **Ortiz, Antonio Moreno.** Elies. [En línea] 2000. [Citado el: 9 de febrero de 2010.] http://elies.rediris.es/elies9/4-2-3.htm.

44. Bujarra. [En línea] 1 de Diciembre de 2007. [Citado el: 22 de marzo de 2010.] http://www.bujarra.com/black/pdfs/ProcedimientoRAID1.pdf.

45. Webmaster. [En línea] 12 de febrero de 2010. [Citado el: 9 de abril de 2010.] http://www.guia-ubuntu.org/index.php?title=PostgreSQL.

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

47. GEDEX.net. *GEDEX es un producto español. © Copyright 1996-2011.* [En línea] http://www.brindys.com/gedex/casmenu.html.

48. **DLD, FLD.** REQUISITOS DEL SISTEMA, INFOLEX. [En línea] http://www.infolex.com.mx/ie/doc/REQUISITOS%20INFORMATICOS%20INFOLEX%207.pdf.

GLOSARIO

- ✓ BD: Base de datos
- ✓ TPC: Tribunales Populares Cubanos
- ✓ Módulo**:** Es un componente autocontrolado de un sistema, el cual posee una interfaz bien definida hacia otros componentes; algo es modular si es construido de manera tal que se facilite su ensamblaje, acomodamiento flexible y reparación de sus componentes.
- ✓ Subsistema: Parte del sistema global con una interfaz razonablemente bien definida.
- ✓ Modelo entidad relación: Un diagrama o modelo entidad-relación (a veces denominado por su siglas, E-R "Entity relationship", o, "DER" Diagrama de Entidad Relación) es una herramienta para el modelado de datos de un sistema de información. Estos modelos expresan entidades relevantes para un sistema de información, sus inter-relaciones y propiedades.
- ✓ -DBAL: Acrónimo de Database Abstraction Layer (Capa de Abstracción de la Base de Datos). Una capa de abstracción de bases de datos es una interfaz de programación de aplicaciones que unifica la comunicación entre una aplicación informática y bases de datos tales como MySQL, PostgreSQL, Oracle o SQLite. La capa de abstracción de bases de datos permite reducir la cantidad de trabajo proporcionando una API consistente para el desarrollador y ocultado las especificaciones de la base de datos, detrás de la interfaz que provee, tanto como sea posible.
- ✓ Entidad: Se refiere a cualquier concepto del mundo real con una existencia independiente.
- ✓ Llave primaria o primary key: Conjunto de atributos de su esquema que son elegidos para servir de identificador unívoco de sus tuplas.
- ✓ Tupla: Es la representación de una fila en una de las tablas que se está almacenando datos. Y las cuales serán llamadas por los administradores de base de datos en el tiempo de ejecución de un sistema.
- ✓ Integridad referencial es un sistema de reglas que utilizan la mayoría de las bases de datos relacionales para asegurarse que los registros de tablas relacionadas son válidos y que no se borren o cambien datos relacionados de forma accidental produciendo errores de integridad.