Universidad de las Ciencias Informáticas
Facultad 4

Título: Sistema para el Control de los Recursos del Proyecto (Rol de Base de Datos)

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autores: Drisis Silvia Díaz Rodríguez

Osbel Veliz Díaz

Tutor: Ing. Alain Fernández Deronceré

Co-tutor: Ing. Yoandro Hechevarría Toranzo

Ciudad de la Habana, Julio, 2008

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos al Ministerio de la Fuerzas Armadas Revolucionarias (MINFAR) y a la Universidad de las Ciencias Informáticas (UCI) a hacer uso del mismo en su beneficio.

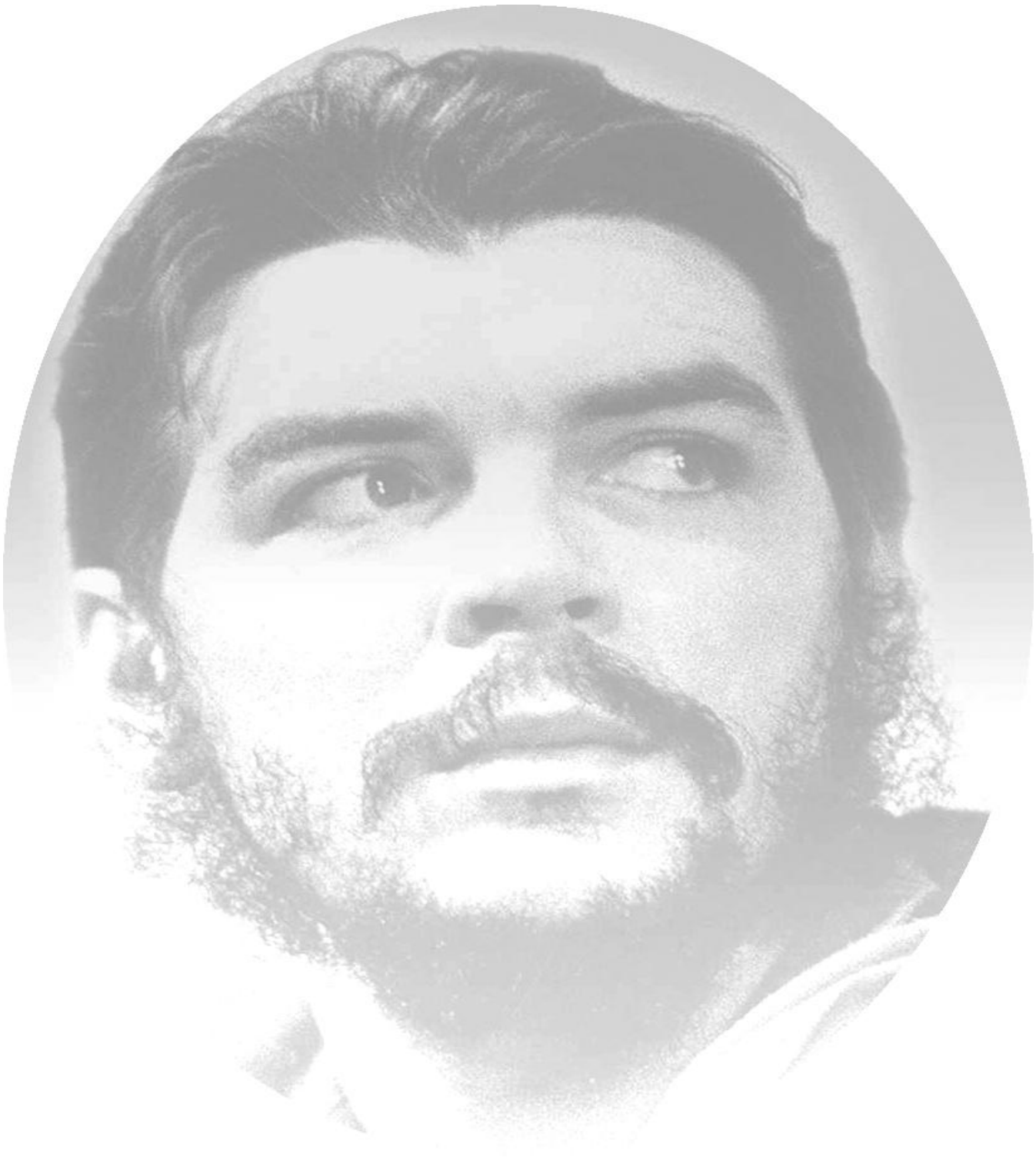
Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

_____ _____

Drisis Silvia Díaz Rodríguez Osbel Veliz Díaz

_____

Alain Fernández Deronceré



“Por que el socialismo…no se ha hecho simplemente para tener hermosas fábricas, sino se ha hecho para el hombre integral.”

Ernesto Guevara de la Serna

AGRADECIMIENTOS

Drisis:

*A mis **padres** por confiar siempre en mí y haberme dado todo el apoyo que pudieron.*

*A mi tía **Alcy** por ser otra madre para mí, por apoyarme y cuidarme como lo ha hecho siempre.*

*A mis **hermanos** en especial a **Tony**, por haber estado a mi lado cuando más lo necesité.*

*A mi novio **Osbel** por quererme tanto y ser excelente pareja de tesis.*

*A **Merianela** y **Arcel** por haber estado siempre que necesite su ayuda.*

*A **Yoandro** por estar siempre ahí para ayudarme.*

*A mi tutor **Alain** por su necesaria ayuda.*

*A toda mi **familia**, mis **amigos** y **compañeros** gracias por estar ahí.*

*A la **Revolución** y a **Fidel**.*

Osbel:

*A mis padres **Elio y Miriam** por darme todo el apoyo que necesité en todo momento.*

*A mi hermano **Abel** que me apoyó y me ayudó.*

*A mi novia **Drisis** por estar a mi lado todo este tiempo como compañera de tesis y como la persona que compartió conmigo su amor y cariño.*

*A mis primos **Amaury** y **Carmen** que me ayudaron y me brindaron su apoyo durante estos cinco años de estudio.*

*A toda mi **familia**, mis **amigos** y **compañeros** que de una forma u otra me ayudaron en esta formación.*

*A **Yoandro** y **Alain** por su necesaria ayuda.*

*A la **Revolución** y a **Fidel**.*

DEDICATORIA

Drisis:

*A mi **mamá** por haber querido siempre lo mejor para mí.*

*A mi **papá** por ser mi motivación, ejemplo y guía.*

*A mi tía **Alcy** por haber sido más que una madre para mí.*

*A mi **novio** y pareja de tesis.*

*A mis **hermanos**, toda mi **familia** y a las personas que alguna vez influyeron en mi formación profesional.*

*A mis **amigas** Jackeline, Marlies e Ismaray por ser como hermanas para mí.*

Osbel:

*A mis padres **Elio y Miriam**, que son y serán mi guía, les dedico este trabajo.*

*A mi hermano **Abel** y a mi primo **Leudys** que ha sido como mi otro hermano, por enseñarme casi todo lo que sé de la vida.*

*A mi novia **Drisis** que me apoyó mucho.*

*A toda mi **familia**, preferiblemente así de general, porque todos se lo merecen, y porque de alguna forma aportaron su granito de arena para que me graduara.*

*A mis **amigos** y **compañeros** por todo lo que me han brindado en cada momento.*

*A la **Revolución** y a **Fidel**.*

RESUMEN

En la Unidad Compatibilización Integración y Desarrollo (UCID) existe la aplicación “Sistema para el Control de los Recursos del Proyecto” capaz de gestionar todos los recursos tanto de tiempo, materiales y humanos, asociados a procesos que se llevan a cabo en dicha entidad. La base de datos del mismo presenta una serie de problemas en su diseño que imposibilitan a los usuarios tener un buen control y manipulación de la información, por lo que se hace necesario realizar un nuevo diseño donde se erradiquen los errores de la base de datos.

Por tanto es necesario desarrollar este tema de tesis, el cual fue concebido con el objetivo de diseñar y elaborar la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto”, presentando una descripción de los pasos necesarios y las herramientas empleadas para la confección de la nueva base de datos de dicho sistema, así como los métodos utilizados para la validación teórica y funcional del diseño propuesto. Se pretende realizar un estudio de las nuevas tendencias vinculadas a las bases de datos, posibilitando la creación de nuevos aportes que optimicen el desempeño de la misma.

TABLA DE CONTENIDOS

INTRODUCCIÓN

Durante los últimos años ha sido relevante el creciente desarrollo de las tecnologías de la computación. Esto ha provocado que el mundo de los ordenadores sea más amigable y tenga mayor facilidad de uso para las personas, convirtiéndose, de esta forma, en más que un entretenimiento para éstas; trayendo consigo que dichas tecnologías tengan un lugar significativo en la cotidianidad humana, ya que se necesitan de las mismas para la realización de gran cantidad de tareas y darle solución a la mayoría de los problemas que la humanidad enfrenta a diario.

Para ello existe un intermediario, o nexo, que hace posible la comunicación entre las personas y los ordenadores, éste es el software, programa, equipamiento o soporte lógico a todos los componentes intangibles de una computadora, es decir, al conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica, en contraposición a los componentes físicos del sistema (hardware). Esto incluye aplicaciones informáticas tales como procesadores de textos, que permiten al usuario realizar una tarea, sistemas operativos, que permiten al resto de los programas funcionar adecuadamente, facilitando la interacción con los componentes físicos y el resto de aplicaciones de diversas índoles.

De forma unísona con el desarrollo y optimización de las tecnologías de la información fue surgiendo la necesidad de almacenar y gestionar los datos que se capturaban o generaban mediante los softwares o aplicaciones que se hacían más potentes debido a dichos avances. Esta necesidad dio paso al surgimiento de las bases de datos. Se podría considerar como una base de datos a cualquier recopilación organizada de información sobre la que haya habido análisis documental y que disponga de un sistema de búsqueda específica. El uso de sistemas de bases de datos además brinda la posibilidad de realizar recuperaciones de la información previamente almacenada.

Las bases de datos tuvieron sus orígenes en 1960 - 1962, cuando se empezaron a usar las máquinas que codificaban la información en tarjetas perforadas por medio de agujeros. Se crearon con el objetivo de almacenar inmensas cantidades de datos que antes se almacenaba en libros, lo que era lento, costoso y complejo (cualquier actualización a realizar, había que hacerla en cada uno de los libros en los que apareciera dicha información a modificar).

Antiguamente, los predecesores de los sistemas de bases de datos fueron los sistemas de ficheros. No hay un momento concreto en que los sistemas de ficheros hayan cesado y hayan dado comienzo los sistemas de bases de datos. De hecho, todavía existen sistemas de ficheros en uso. Cuando los ordenadores evolucionan, aparecen las cintas y los discos, a la vez que las máquinas son dotadas de mucha más potencia y facilidad de manipulación, es por tanto en ese momento cuando las bases de datos comienzan a ser realmente útiles.

Las bases de datos jerárquicas surgieron a mediados de 1960 y son bases de datos que, como su nombre indica, almacenan su información en una estructura jerárquica. Luego, a finales de 1960 surgen las bases de datos de red, es un modelo ligeramente distinto del jerárquico. Su diferencia fundamental es la modificación del concepto de un nodo, permitiendo que un mismo nodo tenga varios padres (algo no permitido en el modelo jerárquico).

En los años 70 nacieron las bases de datos relacionales las cuales inicialmente no se usaron debido a que tenían inconvenientes por el rendimiento, ya que no podían ser competitivas con las bases de datos jerárquicas y de red. En la década de los 80 esto cambia, pues su nivel de programación era bajo y su uso muy sencillo por lo que ya podían competir con las bases de datos antes mencionadas. En esta década el modelo relacional consigue posicionarse dentro del mercado de las bases de datos. En este tiempo también se iniciaron grandes investigaciones paralelas y distribuidas, como las bases de datos orientadas a objetos.

A principios de la década de los 90, para la toma de decisiones, se crea el lenguaje SQL (Structured Query Language), que es un lenguaje programado para consultas. El programa de alto nivel SQL es un lenguaje de consulta estructurado que analiza grandes cantidades de información, el mismo permite especificar diversos tipos de operaciones frente a la misma información, a diferencia de las bases de datos de los 80 que eran diseñadas para las aplicaciones de procesamiento de transacciones. Los grandes distribuidores de bases de datos incursionaron con la venta de bases de datos orientada a objetos. A finales de estos años aparece la WWW (Word Wide Web) la cual facilita la consulta de las bases de datos. Actualmente tienen una amplia capacidad de almacenamiento de información, también una de las ventajas es el servicio de siete días a la semana las veinticuatro horas del día, sin interrupciones a menos que haya planificaciones de mantenimiento de las plataformas o el software e incluso, en el caso de sistemas de gestión de bases de datos de alta concurrencia se han implementado mecanismos de mantenimiento “en caliente”, es decir, se brinda este servicio sin afectar a sus consumidores, con el sistema en ejecución.

Un ejemplo de las aplicaciones o sistemas informáticas que utilizan los sistemas de bases de datos para almacenar su información de forma segura, organizada y de fácil acceso son los ERPs (Enterprise Resource Planning, Planificación de los Recursos Empresariales), estos son muy utilizados y de gran importancia en la actual sociedad. ERP es un término generalizado en el mundo del software bajo el que se engloban una gran variedad de paquetes software, generalmente multi-modulares, que ofrecen soluciones integradas diseñadas para dar soporte a múltiples procesos de una empresa o entidad. Un ERP puede contener software para gestión de producción, clientes, compras, cuentas a pagar o cobrar, contabilidad general, facturación, inventario, recursos humanos, nominas o cualquier otra función que se tenga que desarrollar dentro de las mismas.

Internacionalmente los ERP son aplicaciones que están muy asentadas en el mundo empresarial y que tienen un gran presente y un prometedor futuro. Se considera que en los próximos años, prácticamente todas las empresas dedicadas a la fabricación estarán usando algún ERP y esto también es aplicable para grandes corporaciones dedicadas a otros campos.

La informática en los últimos tiempos se ha convertido en parte del sustrato tecnológico del proceso de globalización en el cual está inmerso todo el mundo, lo que implica la necesidad de preparar a las nuevas generaciones para la asimilación y utilización de dicha tecnología. Cuba no está exenta de todo el desarrollo de la informática y de las comunicaciones que se ha realizado desde la segunda mitad del siglo pasado, se acometieron tareas del progreso científico-técnico de la manera más integral posible, no sólo creando instituciones de investigación, sino desarrollando también actividades como la información científica, la normalización y control de calidad, la organización científica del trabajo, las patentes y licencias, la proyección industrial, incluidas las tareas vinculadas con la transferencia de tecnología y su asimilación, obteniéndose resultados como la creación de una infraestructura científica y técnica que permite contar con un conjunto de centros de investigación que desarrollan resultados de reconocido prestigio internacional, el fortalecimiento de la red de los Centros de Enseñanza Superior y el impulso dado a la investigación en ellos, la creación del Ministerio de Informática y las Comunicaciones en el año 2000, el sistema de salud, de amplia base técnica, considerable integralidad y acceso irrestricto para toda la población, el cual constituye una de las expresiones más nítidas de los vínculos entre el progreso científico y el progreso social en Cuba, la creación de diversos centros de investigación en el campo de las ciencias sociales y de difusión de la cultura nacional y esclarecimiento de la problemática histórica, económica, política y filosófica de la sociedad, la utilización de las nuevas tecnologías y conocimientos que devienen rasgos fundamentales de la revolución científico-técnica, además de la creación de numerables softwares para la automatización de procesos en distintas ramas de la sociedad con el objetivo de alcanzar una mayor efectividad y eficiencia mediante su realización.

El Ministerio de las Fuerzas Armadas Revolucionarias (MINFAR) durante todos sus años de existencia ha realizado una gran cantidad de procesos propios de la organización, los cuales ha tratado de llevar a cabo con la mayor eficiencia posible a pesar de la gran magnitud de los mismos, debido a la envergadura de tal entidad. Dicho ministerio se ha dado a la tarea de informatizar una serie de procesos que se realizaban de forma manual creando para ello un ERP con el objetivo de lograr una mayor eficiencia mediante la realización de los mismos y un mejor control de éstos.

Debido al manejo de grandes cantidades de recursos necesarios para la informatización de los procesos de la entidad (recursos materiales, humanos y tiempo) comenzaron a presentarse problemas tales como, la entrega tardía de los proyectos, mala calidad en los mismos, desorganización del

personal y los medios básicos, así como problemas en la producción. La no existencia de un sistema automatizado para el control de los recursos dio lugar a que se propusiera la creación de una aplicación para la gestión de los mismos, con el objetivo de darle solución a los problemas existentes, surgiendo de esta forma la aplicación web “Sistema para el Control de los Recursos del Proyecto”. Una vez que se puso en práctica dicho sistema, la base de datos del mismo comenzó a presentar varias dificultades como la existencia de redundancia en la información, demora en el tiempo de respuesta de las consultas, no se garantizaba la integridad de los datos, entre otras, todas ellas son resultado de una mala práctica de diseño las cuales se mencionan a continuación:

1- No existió un diseño previo de la base de datos.
2- No cumple los estándares implantados en el proyecto.
3- No esta normalizada.

Por tanto, **el problema** a resolver queda formulado de la siguiente forma: ¿Cómo erradicar las deficiencias existentes en la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto”?

Como **objeto de estudio** se presenta el proceso de diseño de bases de datos para aplicaciones web.

El **campo de acción** de este trabajo es el proceso de diseño de la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto”.

El **objetivo general** queda formulado de la siguiente forma: Diseñar una base de datos donde se erradiquen las deficiencias existentes en la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto”.

Dando paso a los siguientes **objetivos específicos**:

- Realizar el diseño teórico de la investigación.
- Diseñar la base de datos.
- Desarrollar la base de datos.

Para darle cumplimiento a estos objetivos se proponen las siguientes acciones:

- Explorar cómo se comporta la tendencia del diseño de bases de datos en el mundo.
- Realizar el diagrama de clases persistentes utilizando el diseño arquitectónico seleccionado por los arquitectos.
- Realizar el diagrama de entidad relación.
- Normalizar la base de datos.
- Implementar la base de datos.
- Realizar pruebas a la base de datos.

Dada la necesidad de solución de los problemas tratados anteriormente se plantea la siguiente **hipótesis**: Con el re-diseño de la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto”, se logrará erradicar las deficiencias existentes en la misma.

El presente trabajo está estructurado en 3 capítulos los cuales contienen la siguiente información:

Capítulo 1. Fundamentación Teórica: Contiene información acerca del estado del arte de las bases de datos en la actualidad teniendo en cuenta su evolución y los modelos de datos mas conocidos hoy en día y permite conocer acerca de la metodología y herramientas para diseñar e implementar la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto”.

Capitulo 2. Descripción y análisis: Aborda acerca de los requisitos no funcionales del sistema, la descripción de la arquitectura y fundamentación de la misma. También se analiza el diseño de la base de datos mediante el diagrama entidad relación y cada una de sus tablas, se describe una propuesta de diseño de la base de datos y se hace referencia al tema de optimización de consultas.

Capitulo 3. Validación del diseño: Durante este capitulo se realizará la validación teórica del diseño realizado, teniendo en cuenta la integridad, normalización y seguridad de la base de datos. Así como la validación funcional utilizando herramientas o programas que permiten un llenado voluminoso e inteligente de datos.

Al final se podrán observar las conclusiones, recomendaciones, así como las referencias bibliográficas, bibliografía, un glosario de términos y anexos necesarios que harán más comprensible el trabajo.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1 Introducción.

El siguiente capítulo presenta una breve reseña del tema de las base de datos en la actualidad, así como de las tendencias, técnicas, tecnologías y metodologías utilizadas en el mundo para su tratamiento. Se reseñan las metodologías empleadas para el análisis, diseño e implementación de base de datos. Se abordan aspectos relacionados con el uso de las nuevas tecnologías de la informática y las telecomunicaciones.

1.2 Introducción a las Bases de Datos.

Los datos constituyen la parte esencial de un sistema de información, y son los que justifican su existencia. Para organizar y gestionar estos datos en el computador, se han desarrollado técnicas cuya evolución ha estado determinada, principalmente, por el desarrollo de la tecnología de los computadores, así como por los requisitos y necesidades planteadas por los usuarios.

En la actualidad, las técnicas de bases de datos representan la tecnología informática disponible para la organización y gestión de grandes volúmenes de datos en un computador. Se puede afirmar que el núcleo de todo sistema de información actual es una base de datos, y que el diseño y creación de ésta constituyen una etapa importante en la construcción del sistema.

1.2.1 Breve historia de los sistemas de almacenamiento de datos.

Los sistemas de almacenamiento de datos han evolucionado desde los archivos secuenciales hasta los Sistemas de Gestión de Bases de Datos que hoy se conocen. Los primeros brindaban un acceso muy rápido a la información pero tenían un inconveniente, para acceder a una posición se debía hacer de forma secuencial, es decir, se tenía que recorrer el archivo entero. Posteriormente aparecieron los archivos indexados, con los que se podía ir directamente al lugar deseado (de forma aleatoria).

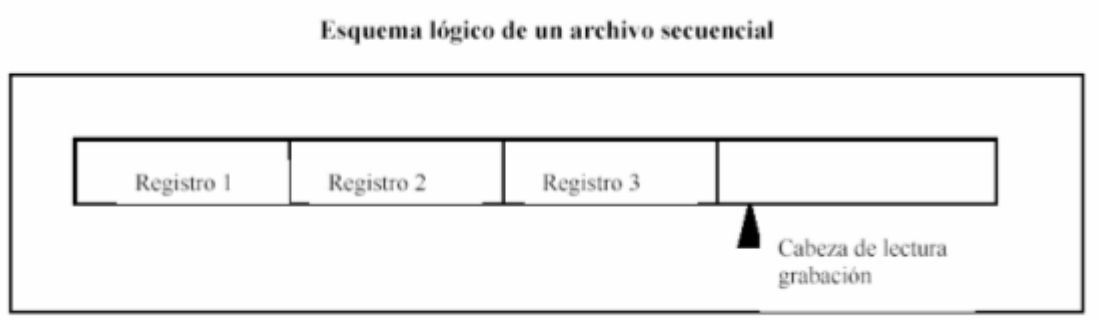


Figura 1. Esquema lógico de un archivo secuencial.

Figura 2. Esquema lógico de un archivo directo.

Por la necesidad de compartir datos entre varias máquinas surgió el NFS (Sistema de Archivos de Red), posibilitando que distintos sistemas conectados a una misma red accedan a ficheros remotos como si fueran locales, y más tarde para evitar fallos en los sistemas de archivos aparecieron los dispositivos RAID (Redundant Arrays of Inexpensive Disks), utilizándose para aumentar la integridad de los datos en los discos.

Debido a que los programas eran cada vez más complejos y grandes, se requería almacenar los datos en sistemas que garantizaran un cierto número de condiciones, permitiendo operaciones complejas sin que se violaran estas restricciones y que garantizaran la seguridad de la información. Respondiendo a estas necesidades surgieron las Bases de Datos Jerárquicas, donde los datos se situaban siguiendo una jerarquía, pero tenían el problema de que los accesos a éstos eran unidireccionales. (1)

Para dar absoluta libertad a las relaciones entre tablas surgieron las Bases de Datos Relacionales en los años 80, y con ellas los Sistemas de Bases de Datos Relacionales (SBDR). Esta nueva forma de almacenamiento de la información aportó dos características muy importantes: las propiedades ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad) y un lenguaje común de acceso a los datos: SQL, facilitando la programación de aplicaciones con bases de datos, y consiguiendo que los programas sean independientes de los aspectos físicos de la base de datos.

Con el transcurso del tiempo aparecieron, en los años 90, las Bases de Datos Distribuidas. Éstas no se almacenan completamente en una localidad central, sino que se distribuyen en una red de localidades que pueden estar geográficamente separadas y conectadas por enlaces de comunicaciones. Cada localidad tendrá su propia base de datos y está capacitada para acceder a los datos de otras localidades.

1.2.2 ¿Qué es una Base de Datos?

Es un conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, que puede considerarse una colección de datos variables en el tiempo. (2)

Una base de datos está compuesta por los siguientes componentes: (2)

1. **Hardware**

Se refiere a los dispositivos de almacenamiento en donde reside la base de datos, así como a los dispositivos periféricos (unidad de control, canales de comunicación, etc.) necesarios para su uso.

2. **Software**

Está constituido por un conjunto de programas que se conoce como Sistema Gestor de Base de Datos, el cual maneja todas las solicitudes formuladas por los usuarios de la misma.

3. **Usuarios**

Existen tres clases de usuarios relacionados con una base de datos:

- El programador de aplicaciones, quien crea programas de aplicación que utilizan la base de datos.
- El usuario final, quien accede a la base de datos por medio de un lenguaje de consulta o de programas de aplicación.
- El Administrador de la base de datos, quien se encarga del control general del Sistema Gestor de Base de Datos.

1.2.3 Sistema Gestor de Base de Datos (SGBD).

El software que permite la utilización y/o actualización de los datos almacenados en una o varias bases de datos, desde diferentes puntos de vista a la vez, por uno o varios usuarios, se denomina Sistema Gestor de Bases de Datos. (2)

El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos y de una forma práctica y eficiente, los datos, de manera que no le sea necesario conocer el modo de almacenamiento de la información en la computadora, ni el método de acceso empleado. Se compone de un lenguaje de definición de datos (DDL), de un lenguaje de manipulación de datos (DML) y de un lenguaje de consulta (SQL).

1.3 Modelos de Bases de Datos.

Un modelo de datos es básicamente una “descripción” de algo conocido como contenedor de datos (algo en donde se guarda la información), así como de los métodos para almacenar y recuperar información de esos contenedores. (2)

Los modelos de bases de datos más extendidos son el modelo relacional, el modelo de red, el modelo jerárquico. El modelo orientado a objetos es también muy popular, pero no existe un modelo estándar orientado a objetos. Con el desarrollo de la tecnología ha aparecido también el modelo de datos objeto relacional y el modelo multidimensional. (3)

1.3.1 Bases de Datos Jerárquicas.

Las Bases de Datos Jerárquicas surgieron a mediados de 1960 y son base de datos que, como su nombre indica, almacenan su información en una estructura jerárquica. En este modelo los datos se organizan en una forma similar a un árbol, en donde un nodo padre de información puede tener varios hijos. El nodo que no tiene padres se le conoce como raíz, y a los nodos que no tienen hijos se les conoce como hojas. Una de las principales limitaciones de este modelo, es su incapacidad de brindar eficientemente una solución a la redundancia de datos. (1)

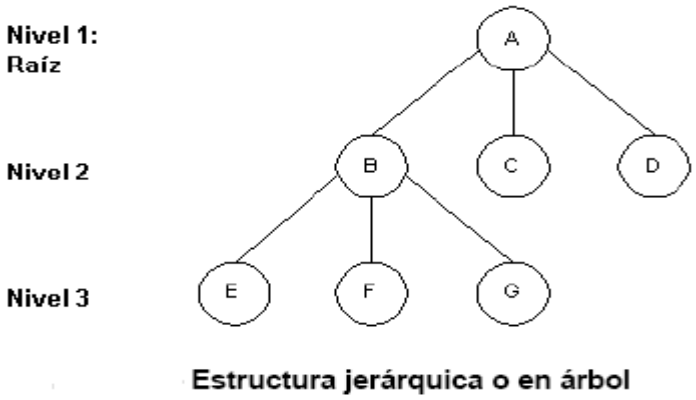


Figura 3. Estructura de una Base de Datos Jerárquica.

Los productos jerárquicos, exactamente el IMS (Information Management System) y el DL/I (Data Language/1) de IBM (International Business Machines) como máximos exponentes de estos sistemas, consiguieron altas cuotas de mercado, aunque la actual difusión de la tecnología relacional los ha llevado a convertirse en sistemas superados. Lo cual no quiere decir que no persistan todavía importantes aplicaciones soportadas en estos productos que están trabajando, por su eficiente

respuesta, a satisfacción de sus usuarios. Las aplicaciones desarrolladas sobre ellos se mantienen sin apenas cambios.

1.3.2 Bases de Datos de Red.

Es un modelo ligeramente distinto del jerárquico, y surge a finales de 1960. Su diferencia fundamental es la modificación del concepto de un nodo, permitiendo que un mismo nodo tenga varios padres (algo no permitido en el modelo jerárquico). (1)

Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos. Pero aún así, la dificultad que implica administrar la información en una Base de Datos de Red, ha significado que sea un modelo utilizado en su mayoría por programadores más que por usuarios finales. Un ejemplo de Base de Datos de Red es el sistema denominado IDMS/R.

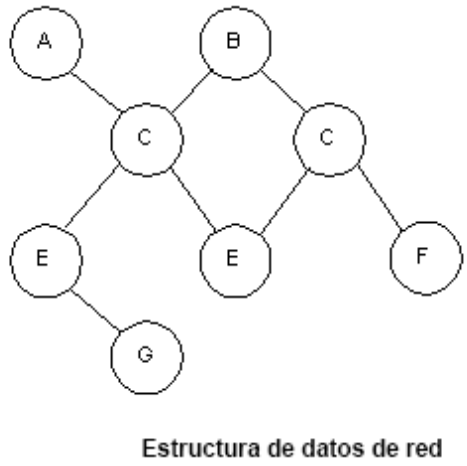


Figura 4. Estructura de una Base de Datos de Red.

1.3.3 Bases de Datos Relacionales.

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). (1)

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 casual de la base de datos. La información puede ser

recuperada o almacenada por medio de “consultas” que ofrecen una amplia flexibilidad y poder para administrarla. Ejemplos de Bases de Datos Relacionales son: Oracle, Informix, MySql, etc.

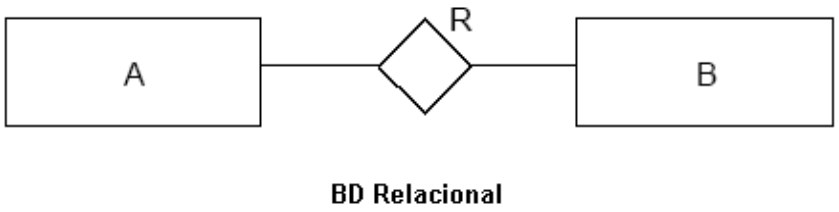


Figura 5. Estructura de una Base de Datos Relacional.

1.3.4 Bases de Datos Orientadas a Objetos.

A finales de los 80’s aparecieron las primeras BDOO (Bases de Datos Orientadas a Objetos). Están diseñadas para ser eficaces, desde el punto de vista físico, para almacenar objetos complejos y métodos. (2)

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). Combinan las mejores cualidades de los archivos planos, las Bases de Datos Jerárquicas y Relacionales. Las BDOO representan el siguiente paso en la evolución de las bases de datos para soportar análisis, diseño y programación orientada a objetos. Ofrecen un mejor rendimiento de la máquina que las Bases de Datos Relacionales, para aplicaciones o clases con estructuras complejas de datos. (2)

Entre los Gestores de Bases de Datos Orientados a Objetos más conocidos se encuentran los siguientes: (1)

Gemstone: Proporciona control de concurrencias y recuperación de la información almacenada, así como gestión de almacenamiento secundario. Soporta acceso concurrente y métodos para mantener la seguridad y la integridad de las base de datos.

Vbase: Surge en 1987 y enfatiza algunas características de la orientación a objetos.

También existen Orion, PDM, Iris y O2, entre otros.

1.3.5 Bases de Datos Objeto – Relacionales.

Las Bases de Datos Objeto Relacionales son base de datos que desde el modelo relacional evolucionan hacia bases de datos más extensas y complejas incorporando, para obtener este fin,

conceptos del modelo orientado a objetos. Se puede afirmar que un Sistema de Gestión Objeto-Relacional (SGBDOR) contiene dos tecnologías; la tecnología relacional y la tecnología de objetos. (2)

Con las Bases de Datos Objeto-Relacional, se pueden crear nuevos tipos de datos, que permiten gestionar aplicaciones más complejas con una gran riqueza de dominios. Tienen la posibilidad de incluir el chequeo de las reglas de integridad referencial a través de los disparadores, entre otras características. Uno de los gestores de Bases de Datos Objeto Relacional que más se utiliza en la actualidad es PostgreSQL.

1.3.6 Bases de Datos Multidimensionales.

Un enfoque que ha cobrado actualmente fuerza es el procesamiento analítico en línea (en inglés, denominado On-Line Analytical Processing u OLAP). Su nombre se deriva del contraste con el procesamiento de transacciones en línea (On-Line Transaction Processing, OLTP). Mientras que el OLTP depende de Bases de Batos Relacionales, el OLAP ha desarrollado una tecnología de Bases de Datos Multidimensionales. Estas bases de datos fundan los cimientos para el desarrollo de los cálculos y análisis multidimensionales que requiere la inteligencia empresarial.

Las Bases de Datos Multidimensionales son base de datos estructuradas como un hipercubo, con un eje por dimensión de estructura basada en dimensiones orientadas a consultas complejas y alto rendimiento. Estos modelos tienen gran aplicación en la bioinformática, debido al almacenamiento de grandes cantidades de información biológica. (4)

A continuación se muestra una representación espacial de una variable multidimensional con una, dos y tres dimensiones. En esta figura los cubitos representan valores de dimensión, y las esferas son datos.

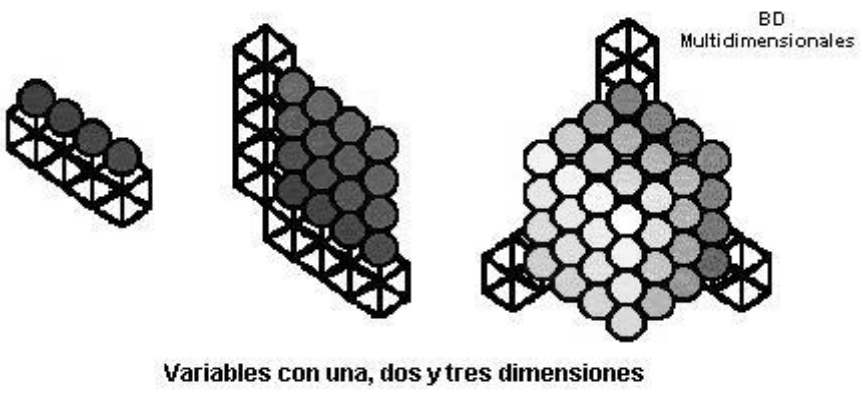


Figura 6. Representación espacial de una variable multidimensional.

Una variable unidimensional podría ser el cambio del euro con el dólar, que sólo varía en la dimensión <tiempo>. Los cubitos serían, por ejemplo, los días del año y las esferas serían los valores numéricos correspondientes al cambio monetario en cada momento. Un ejemplo de variable de dos dimensiones es el número de habitantes, que se mueve por las dimensiones <Geografía> y <tiempo>. Finalmente, los ingresos de una organización podrían almacenarse mediante una variable de tres dimensiones: <producto>, <Geografía> y <tiempo>.

Después de un estudio de los diferentes modelos de base de datos existentes se decide utilizar el modelo relacional por las características antes mencionadas y porque se ajusta al problema a resolver.

1.4 Tendencias.

La utilización de bases de datos como plataforma para el desarrollo de Sistemas de Aplicación en las organizaciones se ha incrementado notablemente en nuestros días. Actualmente las base de datos ofrecen numerosas funcionalidades de gran importancia para el almacenamiento y la manipulación de la información en las instituciones, pero siguen creciendo las demandas de nuevas operaciones que faciliten una mayor eficiencia en el manejo de los datos. Esto trae consigo el desarrollo de técnicas y herramientas que brinden una solución eficiente a las solicitudes planteadas por los usuarios.

Algunas de las tendencias actuales y futuras de bases de datos son citadas a continuación.

- La explotación efectiva de la información brinda ventajas competitivas a las organizaciones.
- El uso de las Bases de Datos Distribuidas y Multidimensionales se incrementa de manera considerable en la medida en que la tecnología de comunicación de datos brinde más facilidades para ello.
- El uso de las bases de datos facilita y soporta en gran medida a los sistemas de información para la toma de decisiones.
- Los lenguajes de consulta (SQL) permitirán el uso del lenguaje natural para solicitar información de la base de datos, haciendo más rápido y fácil su manejo.

En los últimos años se ha visto un gran crecimiento en la capacidad de generación y almacenamiento de información, debido a la creciente automatización de procesos. Desgraciadamente no ha existido un desarrollo equivalente en las técnicas de análisis de información, por lo que existe la necesidad de una nueva generación de técnicas y herramientas computacionales con la capacidad de asistir a usuarios en el análisis automático e inteligente de datos. El procesar automáticamente

grandes cantidades de datos para encontrar conocimiento útil para un usuario y satisfacerle sus metas, es el objetivo principal del área de Descubrimiento de Conocimiento en Bases de Datos o KDD (Knowledge Discovery from Data Base). Este es el campo que está evolucionando para proporcionar soluciones al análisis automatizado de datos.

A nivel internacional existen sistemas informáticos muy parecidos a la aplicación informática "Sistema para el Control de los Recursos del Proyecto", los cuales poseen bases de datos similares a la que se pretende diseñar, estos son el Egroupware, desarrollado en Postgree, con una excelente velocidad de búsqueda a pesar de contar con 148 tablas y el Dotproyect, desarrollado en MySql, con una buena velocidad de búsqueda y 68 tablas. A pesar de esto no se puede utilizar ninguna de las dos bases de datos de estas aplicaciones porque la primera es más orientada a la venta, negocios de productos y el marketing, en cambio la segunda se acerca mas a los objetivos de lo que se quiere lograr pero en realidad no cumple con las expectativas de la aplicación informática "Sistema para el Control de los Recursos del Proyecto".

1.5 Arquitectura de los Sistemas Gestores de Bases de Datos.

Las arquitecturas de bases de datos han evolucionado mucho desde sus comienzos, aunque la considerada estándar hoy en día es la descrita, a finales de los años 70, por el comité ANSI/X3/SPARC (Standard Planning and Requirements Committee of the American National Standards Institute on Computers and Information Processing) La misma se emplea como ayuda para conseguir la separación entre los programas de aplicación y los datos, el manejo de múltiples vistas por parte de los usuarios y el uso de un catálogo para almacenar el esquema de la base de datos.

Este comité propuso una arquitectura general basada en tres niveles o esquemas: el nivel físico, o de máquina, el nivel externo, o de usuario, y el nivel conceptual. Así mismo describió las interacciones entre estos tres niveles y todos los elementos que conforman cada uno de ellos.

- Nivel interno: Describe la estructura física de almacenamiento de base de datos. Emplea un modelo físico de datos y los únicos datos que existen están realmente en este nivel.
- Nivel conceptual: Describe la estructura de toda la base de datos para una comunidad de usuarios. Oculta los detalles físicos de almacenamiento y trabaja con elementos lógicos como entidades, atributos y relaciones.
- Nivel externo o de vistas: tiene varios esquemas externos o vistas de usuario. Cada esquema describe la visión que tiene de la base de datos un grupo de usuarios, ocultando el resto.

Las posibles proyecciones de datos al utilizar la arquitectura de tres niveles quedan resumidas en la gráfica:

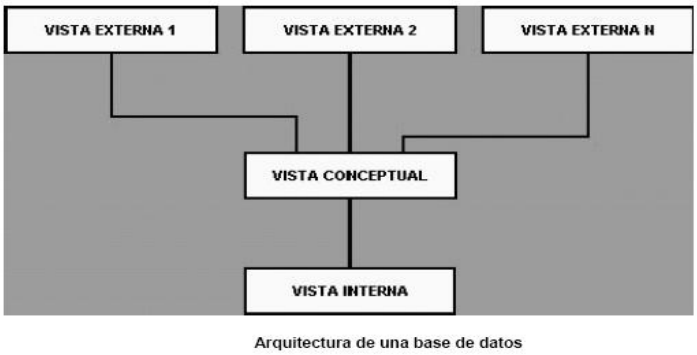


Figura 7. Posibles proyecciones de datos al utilizar la arquitectura de tres niveles.

1.6 Técnicas para diseñar Base de Datos Relacionales.

Existen dos escuelas que implican dos grupos de algoritmos a la hora de aplicar la Teoría de Normalización. Una son los métodos de Análisis o Descomposición y la otra los procedimientos de Síntesis que a continuación detallamos.

1. Diseño Descendente, análisis o descomposición

Esos métodos parten de una relación llamada Universal que contiene a todos los atributos del universo del discurso y las dependencias funcionales existentes entre ellas. Por medio de descomposiciones sucesivas y cumpliendo los principios antes analizados de conservación de la información y de las dependencias resultan esquemas de menor grado en formas normales cada vez más avanzadas, por lo que se puede garantizar que se reducen las anomalías.

Se pueden aplicar estos métodos también a cada una de las relaciones obtenidas a partir de un esquema conceptual en un modelo de datos de alto nivel (Modelo ER) y luego transformar el esquema conceptual a un conjunto de relaciones empleando un procedimiento de transformación. También se puede aplicar el método a cada una de las relaciones que se obtienen de la transformación del DER al modelo relacional.

El objetivo inicial que se pretende con estos métodos es separar la información referente a un objeto o entidad diferente.

2. Síntesis Relacional o procedimientos de Síntesis.

Este es un método de diseño alternativo a la descomposición, resulta ser un enfoque es más purista, implica contemplar el diseño de esquema de base de datos relacionales estrictamente en términos de dependencias funcionales y otros tipos especificadas para los atributos de la base de datos. Aquí los esquemas de relación en 3FN ó FNBC se sintetizan gradualmente agrupando los atributos apropiados.

Es descrito por Bernstein (1976) a partir de conjuntos de atributos y dependencias funcionales obtiene relaciones. Recorre un camino inverso a la descomposición, es decir, busca agrupar atributos a fin de tener en una relación toda la información correspondiente a un objeto o entidad.

De las técnicas mencionadas anteriormente se escoge la de de Síntesis Relacional o Procedimientos de Síntesis ya que es la mas apropiada para el diseño de la base de datos de la aplicación informática "Sistema para el Control de los Recursos del Proyecto" además de que es la técnica de diseño mas usada en el entorno en que se diseña esta base de datos actualmente.

1.7 RUP (Proceso Unificado de Desarrollo).

De acuerdo con los lineamientos, principios y normativas para el desarrollo del software en las FAR se establece que todas sus aplicaciones informáticas utilicen RUP como Proceso de desarrollo de software en la creación de sus proyectos. La base de datos del "Sistema para el Control de los Recursos del Proyecto" se diseñó cumpliendo con lo reglamentado en el documento que plantea estas normas.

RUP (Rational Unified Process) más que un simple proceso para el desarrollo de aplicaciones, es un marco de trabajo genérico, que puede especificarse para una gran variedad de aplicaciones informáticas, para diferentes áreas de aplicación, diferentes niveles de aptitud y diferentes tamaños de proyectos. Garantiza la elaboración de todas las fases de un producto de software orientado a objetos, desde la etapa de ingeniería de requerimientos hasta la de prueba, lo que resulta muy importante en la elaboración del diseño de la base de datos, ya que depende del trabajo desarrollado en otras etapas del proyecto para la realización del mismo.

Este proceso es dirigido por casos de uso para describir lo que se espera del software, está muy orientado a la arquitectura del sistema, mostrando la visión común del sistema completo y por tanto describiendo los elementos del modelo que son más importantes para su construcción. Se apoya en UML (Unified Modeling Language) como lenguaje expresivo, claro y uniforme, que no garantiza el éxito

de los proyectos pero sí mejora sustancialmente el desarrollo de los mismos, al permitir una nueva y fuerte integración entre las herramientas, los procesos y los dominios.

Según RUP el Diseñador Principal de Bases de Datos es el responsable de definir los detalles del diseño de la base de datos y para ello crea el Modelo de Datos, garantizando su integridad y consistencia. Este artefacto describe las representaciones lógicas y físicas de los datos persistentes usados por la aplicación.

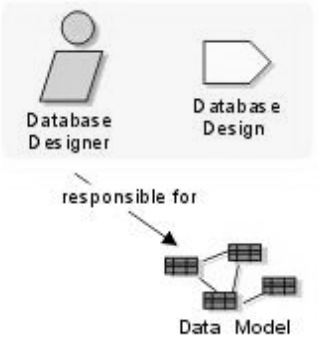


Figura 8. Artefacto construido por el diseñador de la base de datos.

1.8 Herramientas.

Para confeccionar una base de datos se necesitan herramientas que posibiliten garantizar la seguridad, consistencia e integridad de la información, así como lograr que todo el proceso de su creación sea lo más sencillo y rápido posible.

Existe una gran variedad de herramientas que se encargan del diseño de las bases de datos, entre ellas se encuentran:

1.8.1 ERwin

Brinda productividad en diseño, generación, y mantenimiento de aplicaciones. Desde un modelo lógico de los requerimientos de información, hasta el modelo físico perfeccionado para las características específicas de la base de datos diseñada, ERwin permite visualizar la estructura, los elementos importantes, y optimizar el diseño de la base de datos. Genera automáticamente las tablas y miles de líneas de procedimientos almacenados y disparadores para los principales tipos base de datos.

Soporta principalmente bases de datos relacionales SQL y bases de datos que incluyen Oracle, Microsoft SQL Server, Sybase, DB2, e Informix.

1.8.2 Visual Paradigm.

Es una herramienta CASE que ofrece un entorno de creación de diagramas para UML. Esta herramienta usa un lenguaje estándar común para todo el equipo de desarrollo que facilita la comunicación. El diseño es centrado en casos de uso y enfocado al negocio, lo que permite generar software de gran calidad. Tiene capacidad para la ingeniería directa e inversa en Java, C++, PHP, entre otros lenguajes y disponibilidad de múltiples versiones para cada necesidad. Es multiplataforma y muy útil para la generación de código fuente en PHP. Tiene la capacidad de crear el esquema de clases a partir de una base de datos y crear la de base de datos a partir del esquema de clases. Incorpora el soporte para trabajo en equipo, proporcionando que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros de equipo. (Equipo de Softonic, 1997-2007).

Esta herramienta soporta bases de datos que incluyen PostgreSQL, SB2, MySQL, DB2, Informix, Microsoft SQL Server, Oracle.

De las herramientas mencionadas anteriormente, se utiliza para el modelado de la base de datos la herramienta Visual Paradigm, que a diferencia del ERwin soporta bases de datos en PostgreSQL además de todas las otras que soporta este último. Otra ventaja de esta herramienta es que la Universidad de las Ciencias Informáticas paga su licencia para su uso.

1.8.3 MySQL.

MySQL es un sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU (General Public License). Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. MySQL fue creada por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca.

Aunque MySQL es software libre, MySQL AB distribuye una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de no ser así, se vulneraría la licencia GPL.

Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen

infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración. (5)

Ventajas:

1. La velocidad a la hora de realizar las operaciones, lo que le hace uno de los gestores que ofrecen mayor rendimiento.
2. Su bajo consumo lo hacen apto para ser ejecutado en una máquina con escasos recursos sin ningún problema.
3. Las utilidades de administración de este gestor son envidiables para muchos de los gestores comerciales existentes, debido a su gran facilidad de configuración e instalación.
4. Tiene una probabilidad muy reducida de corromper los datos, incluso en los casos en los que los errores no se produzcan en el propio gestor, sino en el sistema en el que está.
5. El conjunto de aplicaciones Apache-PHP-MySQL es uno de los más utilizados en Internet en servicios de foro y de buscadores de aplicaciones.

Desventajas:

1. Carece de soporte para transacciones, rollback y subconsultas.
2. El hecho de que no maneje la integridad referencial, hace de este gestor una solución pobre para muchos campos de aplicación, sobre todo para aquellos programadores que provienen de otros gestores que sí que poseen esta característica.
3. No es viable para su uso con grandes bases de datos, a las que se acceda continuamente, ya que no implementa una buena escalabilidad. (5)

1.8.4 Microsoft SQL Server.

Microsoft SQL Server es un sistema de gestión de bases de datos relacionales (SGBD) basada en el lenguaje SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea Así de tener unas ventajas que a continuación se pueden describir.

Entre sus ventajas están:

- ✓ Soporte de transacciones.
- ✓ Gran estabilidad.

- ✓ Gran seguridad.
- ✓ Escalabilidad.
- ✓ Soporta procedimientos almacenados.
- ✓ Incluye también un potente entorno gráfico de administración, que permite el uso de comandos Lenguaje de Definición de Datos (DDL) y Lenguaje de Manipulación de Datos (DML) gráficamente.
- ✓ Permite trabajar en modo cliente-servidor donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información.
- ✓ Además permite administrar información de otros servidores de datos.

Desventajas:

- ✓ Tiene soporte solamente en el sistema operativo Windows.
- ✓ Es un software de licencia propietaria.

1.8.5 Oracle.

Es un manejador de base de datos relacional que hace uso de los recursos del sistema informático en todas las arquitecturas de hardware, para garantizar su aprovechamiento al máximo en ambientes cargados de información.

Ventajas:

- ✓ Posee igual interacción en todas las plataformas (Windows, Unix, Macintosh y Mainframes). Ya que más del 80% de los códigos internos de Oracle son iguales a los establecidos en todas las plataformas de Sistemas Operativos.
- ✓ Oracle soporta bases de datos de todos los tamaños, desde severas cantidades de bytes y gigabytes en tamaño.
- ✓ Soporte de transacciones.
- ✓ Es mucho más fácil la realización de backups (copias de seguridad).
- ✓ Es mucho más eficaz y eficiente.
- ✓ Tiene una amplia gama de herramientas para operar con la base de datos tanto como usuario y como Administrador.

Desventajas:

- ✓ Elevado precio comercial.
- ✓ La seguridad de la plataforma, y las políticas de suministro de parches de seguridad, modificadas a comienzos de 2005 y que incrementan el nivel de exposición de los usuarios.
- ✓ Exige una gran cantidad de recursos de la máquina donde se encuentre instalado.

1.8.6 PostgreSQL.

Al igual que RUP el uso de PostgreSQL está establecido por las normativas de las FAR. En el proyecto se utilizó la versión 8.0.

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 (OpenSource) 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:

1. 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, etc. Además permite la creación de tipos de datos propios.
2. Incorpora una estructura de datos array.

3. Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc.
4. Permite la declaración de funciones propias, así como la definición de disparadores.
5. Soporta el uso de índices, reglas y vistas.
6. Incluye herencia entre tablas.
7. Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.

PostgreSQL es el servidor de bases de datos de código abierto más potente que existe y es por tanto la alternativa a MySQL cuando se necesitan características avanzadas como transacciones, procedimientos almacenados, triggers, vistas, etc.

Es el servidor de bases de datos más utilizado por los programadores de servlets de Java y, en general, por todos aquellos que realizan aplicaciones cliente-servidor complejas o críticas en el mundo Linux/Unix.

Ventajas:

- ✓ Posee una gran escalabilidad. Es capaz de ajustarse al número de Unidades Centrales de Procesamiento (CPUs) y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta.
- ✓ Implementa el uso de rollback, subconsultas y transacciones, haciendo su funcionamiento mucho más eficaz.
- ✓ Tiene la capacidad de comprobar la integridad referencial, así como también la de almacenar procedimientos en la propia base de datos.
- ✓ PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.

Desventajas:

- ✓ Tiene un límite de 8K por fila, aunque se puede aumentar a 32K, con una disminución considerable del rendimiento.
- ✓ Es lento. (6)

Comparación de los SGBD:

Criterios	**Sistemas Gestores de Bases de Datos**			
	Oracle	**SQL Server**	**MySQL**	**PostgreSQL**
Plataforma	Windows/Linux	Windows	Windows/Linux	Windows/Linux
Velocidad	+	+	+	-
Volumen Datos	+	+	+	+
Integridad	+	+	-	+
Potencia	+	+	+	+
Coste	-	-	+	+
Requerimientos de hardware	-	+	+	+

+ **Positivo** - **Negativo**.

Dadas las características anteriores y la tabla de comparación, el SGBD que se escogió para desarrollar el sistema fue PostgreSQL, ya que SQL Server no es multiplataforma y además hay que pagar un importe por su licencia, así mismo Oracle, incluso su coste es más elevado que el de SQL Server a parte de que el Oracle necesita una máquina con elevados recursos (memoria RAM, velocidad de microprocesador, capacidad de disco duro), en cuanto al MySQL se puede decir que cuando se almacena gran cantidad de datos estos pueden perderse o dañarse, además de que se necesitaba un SGBD que almacenara gran cantidad de información, por lo que se decidió escoger PostgreSQL que a pesar de ser el más lento, aventaja en los demás aspectos a los otros SGBD comparados.

1.8.7 EMS SQL Manager para PostgreSQL.

PGManager es una herramienta gráfica fácil de utilizar para la administración de PostgreSQL. Trabaja con cualquiera de sus versiones hasta la 8.1 y soporta todas las últimas características de PostgreSQL, incluyendo espacios de tablas, nombres de argumentos en funciones y más. El programa ofrece muchas herramientas poderosas para usuarios experimentados, como un Diseñador Visual de Bases de Datos y un Constructor Visual de Consultas. Es la más completa hasta ahora. Permite obtener la documentación sobre el diseño de la base de datos y tiene un grupo de herramientas importantes para la importación y exportación de datos. Posibilita asignar los derechos de usuarios de una forma sencilla y rápida. (1999-2007) La versión del PGManager que se utilizó fue la 3.1.0.1.

Características:

- ✓ Soporte completo para PostgreSQL hasta la versión 8.1.
- ✓ Administración y navegación rápida de bases de datos.
- ✓ Administración fácil de todos los objetos PostgreSQL.
- ✓ Herramientas de manipulación avanzada de datos.
- ✓ Administración efectiva de seguridad.
- ✓ Excelente herramientas visuales y de texto para la construcción de consultas.
- ✓ Capacidades de exportación e importación de datos.
- ✓ Poderoso diseñador visual de bases de datos.
- ✓ Modo guiado para labores de mantenimiento.
- ✓ Interfaz atractiva.

1.8.8 PGAdmin 3

Es una interfaz comprensible para el diseño y administración de una base de datos PostgreSQL, diseñada para ejecutarse en la mayoría de los Sistemas Operativos. La aplicación corre bajo GNU/Linux, FreeBSD y Windows 2000/XP. (7)

Características:

- ✓ Un agente de SQL/shell para tareas programadas.
- ✓ Soporte para PostgreSQL
- ✓ Autovacuum administration.
- ✓ Roles.
- ✓ Procedimientos almacenados.
- ✓ Transacciones preparadas.
- ✓ Permite registrar un servidor sin establecer conexión.
- ✓ Ayuda para usuarios nuevos.
- ✓ Las contraseñas pueden ser almacenadas en los ficheros .pgass/pgpass.conf.

Se utilizó para la administración de la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” el PGAdmin, que a pesar de no tener una interfaz amigable como la del PGManager, posee la ventaja de ser software libre a diferencia de este último que es propietario.

1.8.9 GenTipFAR

GenTipFAR es una herramienta para generar la capa de acceso a datos. Genera las clases típicas y consultas a partir de una base de datos existente, permite generarlas para todas las tablas de la base de datos o para algunas que se seleccionen, esto consiste en generar el código fuente que es constante y gestionar la parte variable, además incorpora la gestión de esquemas y la generación a partir de los mismos. Se utilizó esta herramienta por su fácil interfaz. Fue creada en la Universidad de las Ciencias Informáticas basado en los patrones y la arquitectura implantada para la creación de sistemas informáticos en el Centro-UCIFAR.

1.8.10 EMS Data Generator 2005 for PostgreSQL.

El Generador de Datos de EMS para PostgreSQL es una utilidad poderosa para generar datos de prueba a varias tablas de base de datos PostgreSQL inmediatamente. La aplicación permite que usted defina tablas y campos para generar datos, ponga variedades de valor, genere campos de trabajo por horas por la máscara, valores de carga para campos de archivos, consiga listas de valores de preguntas de SQL y muchos otros rasgos para generar datos de prueba de un modo simple y directo. Esto también le proporciona que usted genere datos usando plantillas de generación.

1.9 Conclusiones.

Mediante un estudio detallado de cada herramienta a utilizar y conceptos importantes para la comprensión y confección de una base de datos que cumpla con la política de software libre seguida para el desarrollo de aplicaciones informáticas en las Fuerzas Armadas Revolucionarias, se llega a la conclusión que la utilización de dichas herramientas es la optima solución cuando nuestro país, como muchos otros, ya se ha trazado la meta de implantarlo en numerosas esferas a nivel nacional, siendo un ejemplo de ello las FAR. Se detalla un estudio de algunas de las últimas tendencias que a nivel mundial se vienen utilizando, por ello nuestro énfasis en PostgreSQL, que es el SGBD donde reside la base de datos en estos momentos, por su gran capacidad para garantizar la seguridad e integridad de los datos. Por lo que es uno de los aspectos que se ha considerado en el momento de escoger las herramientas a utilizar para construir la solución que se propone.

CAPÍTULO 2: DESCRIPCIÓN Y ANÁLISIS

2.1 Introducción.

En este capítulo se definen las actividades más importantes que posibilitaron la construcción de la base de datos, teniendo como resultado final el diseño de la misma.

Primeramente se seleccionaron y argumentaron los requisitos no funcionales del sistema existente. Luego se describió y fundamentó la arquitectura a utilizar en el proyecto para una buena comprensión del tema del acceso a datos desde la misma. Un paso clave que se tuvo en cuenta en este capítulo fue el estudio del diagrama entidad relación de la base datos actual, por su importancia para la confección del modelo de datos. Por último se confeccionó el diagrama entidad relación de la base de datos propuesta y se describieron sus tablas.

2.2 Selección y argumentación de los requisitos no funcionales del sistema existente.

Para que un sistema de software funcione correctamente se debe lograr una comunicación efectiva entre los usuarios y el equipo de proyecto con el objetivo de llegar a un entendimiento de lo que hay que hacer, lo que constituye la clave del éxito en la producción de un software.

Aquí radica la importancia que en los últimos años se le ha dado a la identificación de los requerimientos como parte del proceso de desarrollo del software. Los requisitos se pueden clasificar en: funcionales y no funcionales.

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

2.2.1 Requisitos No Funcionales.

Los requisitos no funcionales mencionados a continuación son los que tienen correspondencia con el desarrollo de la base de datos.

Rendimiento.

- La base de datos debe estar diseñada para el consumo mínimo de recursos.
- La base de datos debe ser capaz de formular la respuesta lo más rápido posible.

Soporte.

Para el servidor de base de datos:

- Se requiere que esté instalado un gestor de base de datos que soporte grandes volúmenes de datos y velocidad de procesamiento.

Portabilidad.

- La base de datos deberá ser compatible con el sistema operativo UNIX (Linux) y con Windows (Versiones como 2000 y XP).

Hardware.

Para las computadoras del cliente:

- Se requiere tengan tarjeta de red.
- Se requiere tengan al menos 64 MB de memoria RAM.
- Se requiere al menos 100MB de disco duro.
- Procesador 512 MHz como mínimo.

Para los servidores:

- Se requiere tarjeta de red.
- Se requiere tenga al menos 256MB de RAM.
- Se requiere al menos 1GB de disco duro.
- Procesador 1.2 GHz como mínimo.

Software.

- La base de datos se desarrollará y generará con tecnología Visual Paradigm 3.0.
- Se utilizará un servidor con el sistema operativo instalado Windows 2000 o superior, o con sistema operativo UNIX (Linux) preferencialmente.
- El sistema utilizará una base datos implementada en PostgreSQL versión 8.0.

Seguridad.

- Mantener la integridad de la información, es decir que no se pierda durante su almacenamiento o transporte.

Confiabilidad.

- La información manejada por la base de datos estará protegida de acceso no autorizado y divulgación.

Integridad.

- La información manejada por la base de datos será objeto de cuidadosa protección contra la corrupción.

Fiabilidad.

- Debe garantizarse el resguardo de la información, de modo que haya varias copias, de forma tal que se posibilite la restauración de la base de datos, en caso de algún problema presentado en la explotación de la misma.

Legales.

- La base de datos debe basarse en el documento que expone las normativas para el desarrollo de softwares en las FAR.
- La mayoría de las herramientas de desarrollo son libres y del resto, las licencias están avaladas.

2.3 Descripción de la arquitectura y fundamentación.

El diseño de la aplicación informática "Sistema para el Control de los Recursos del Proyecto" utiliza la arquitectura en capas, específicamente el de tres capas, facilitando el desarrollo y posibilitando que si se produce algún cambio sólo se ataca al nivel requerido sin tener que modificar los niveles adyacentes a él.

Figura 9. Arquitectura en capas. Tipos de componentes utilizados.

Los tipos de componentes identificados en el escenario de diseño son los siguientes:

Componentes de interfaz de usuario (IU). Las interfaces de usuario se implementaron utilizando formularios, controles y funciones que permiten procesar y dar formato a los datos de los usuarios, así como adquirir y validar los datos entrantes procedentes de éstos.

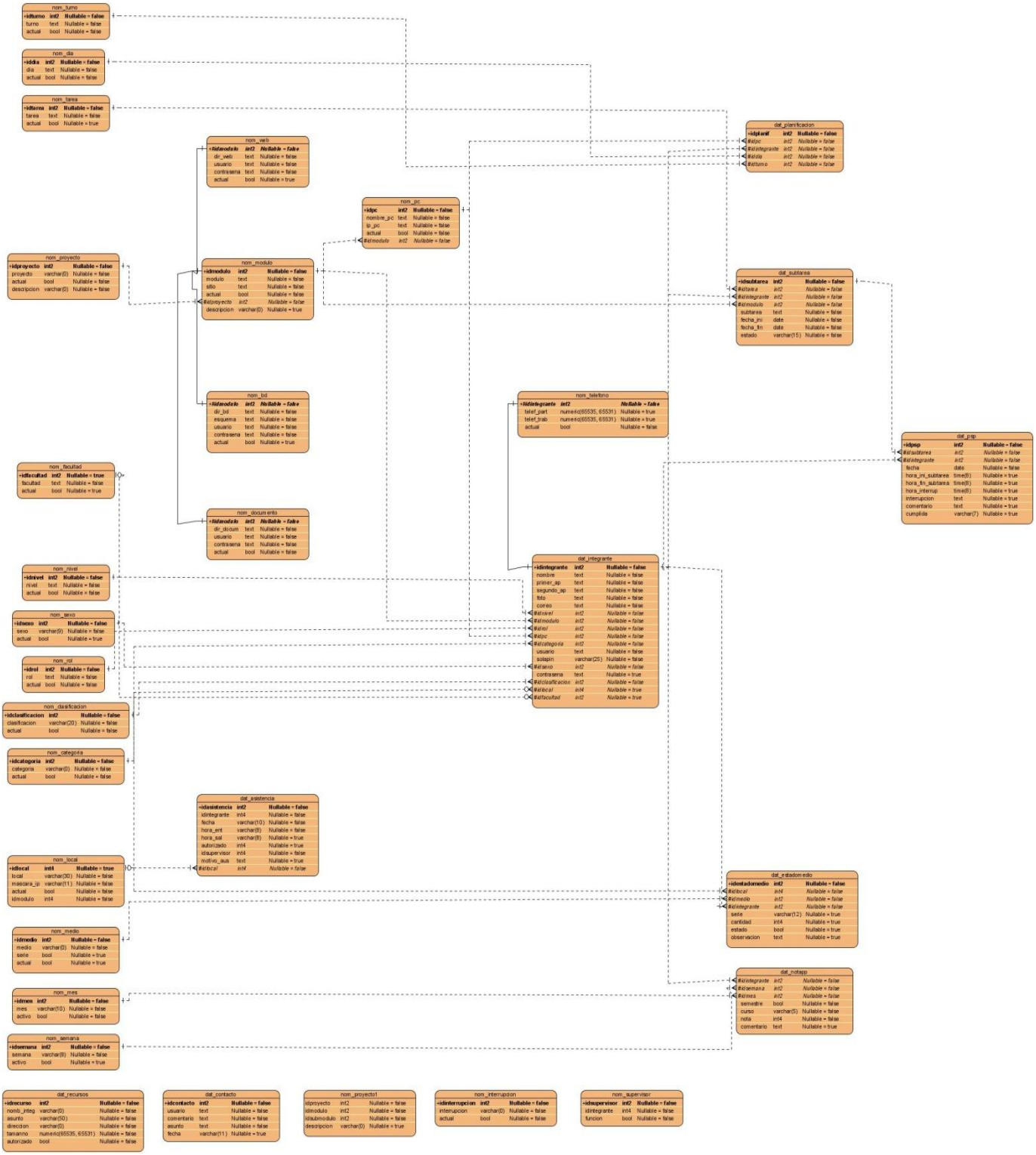
Componentes de lógica de negocio. Independientemente de si el proceso consta de un único paso o de un flujo de trabajo organizado, la aplicación requiere el uso de componentes que implementen reglas y realicen tareas.

Componentes de acceso a datos. La aplicación y los servicios necesitan obtener acceso a un almacén de datos en un momento determinado del proceso de gestión de los recursos de un proyecto. Por ejemplo, la aplicación necesita recuperar los datos de un recurso cualquiera de la base de datos para mostrar al usuario los detalles del mismo, así como insertar nueva información cuando un usuario necesite realizar algún cambio. Por tanto, es razonable abstraer la lógica necesaria para obtener acceso a los datos en una capa independiente de componentes lógicos de acceso a datos, ya que de este modo se centraliza la funcionalidad de acceso a datos y se facilita la configuración y el mantenimiento de la misma.

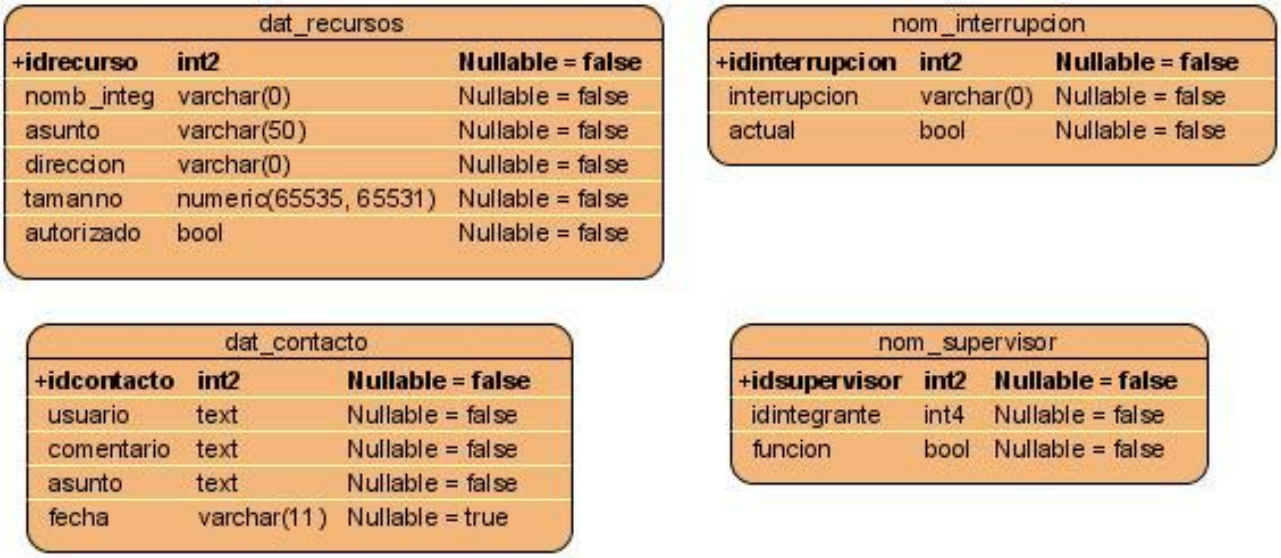
2.4 Diagrama Entidad Relación de la base de datos actual.

A continuación se muestra el diagrama entidad relación de la base de datos actual, el mismo ha sido fraccionado en varios submodelos para una mejor comprensión.

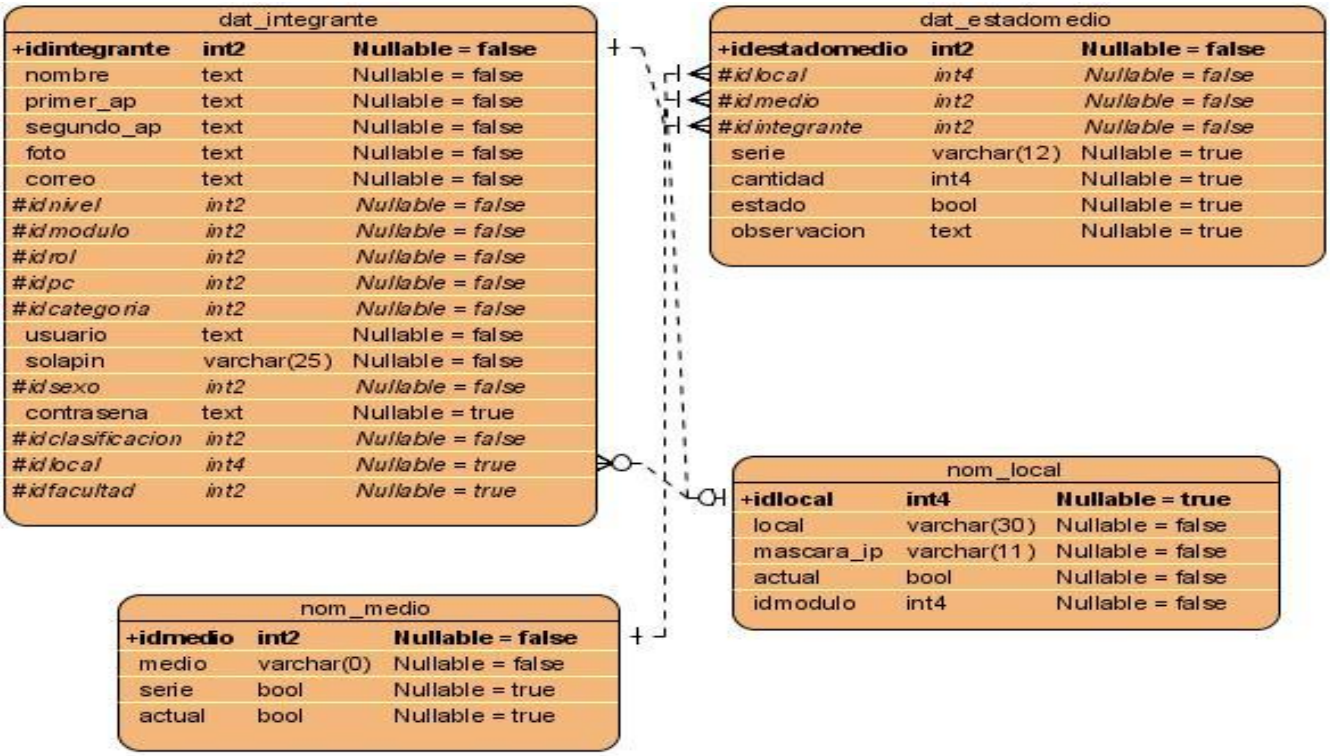
Diagrama general



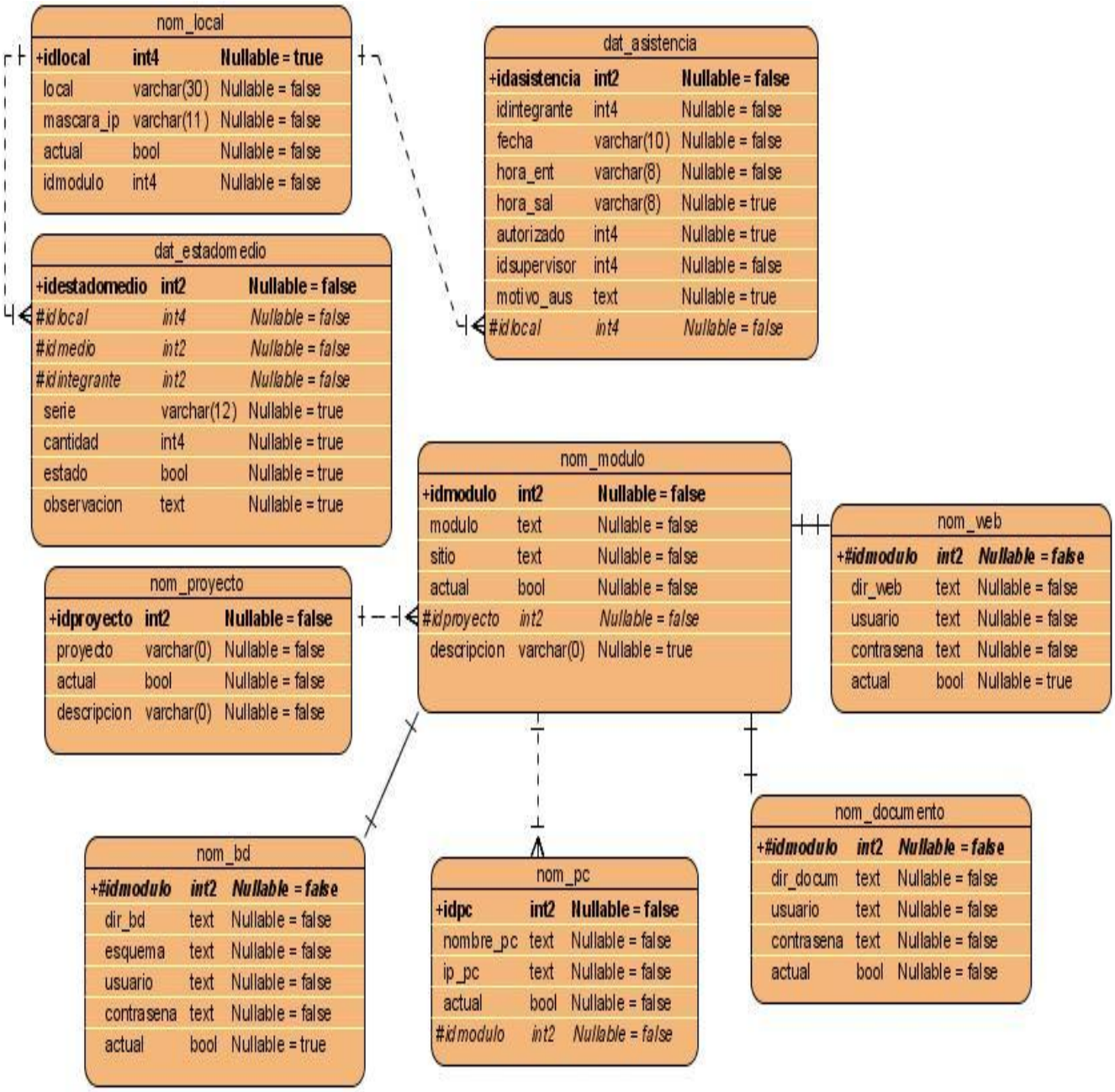
Submodelo 1



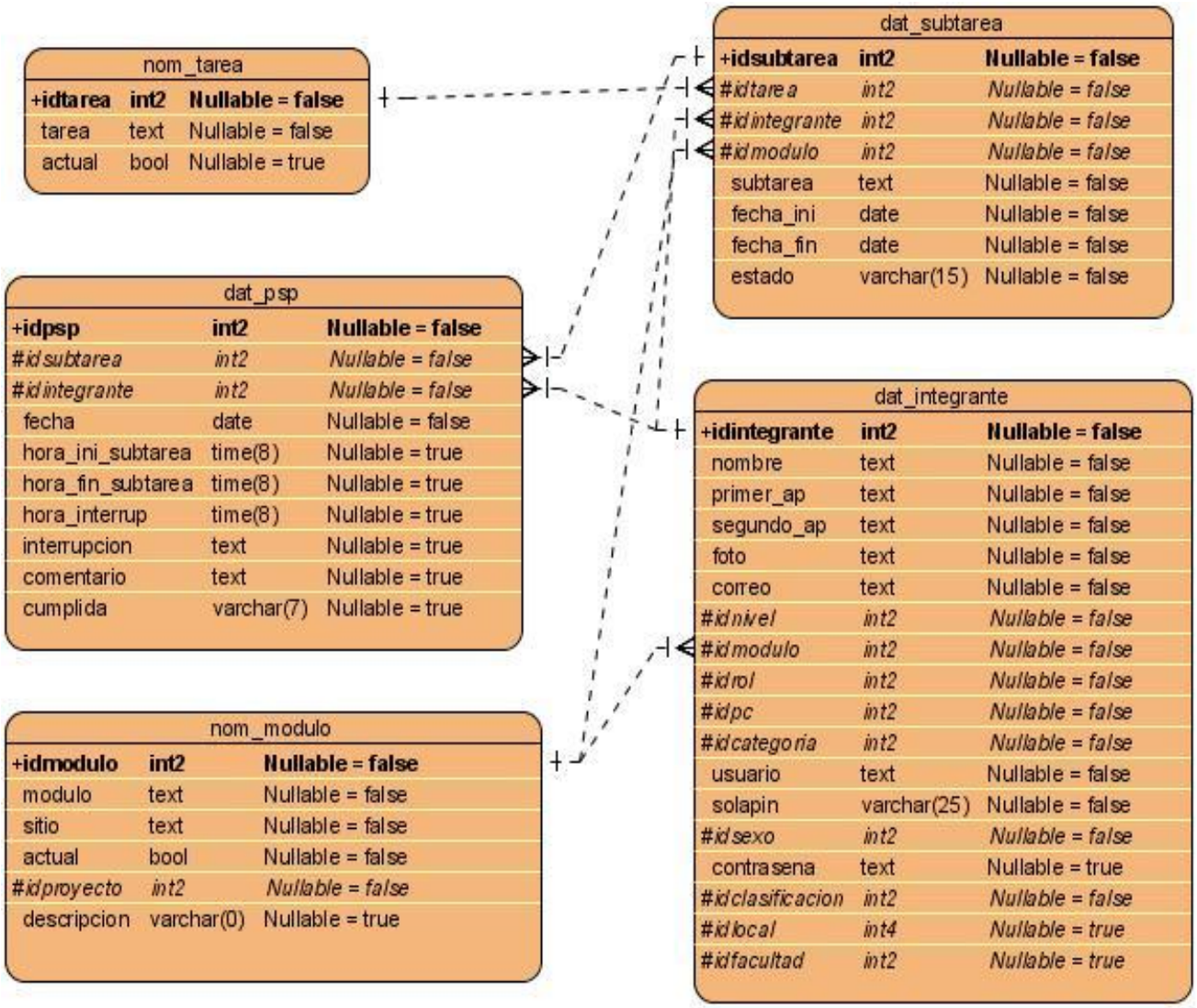
Submodelo 2



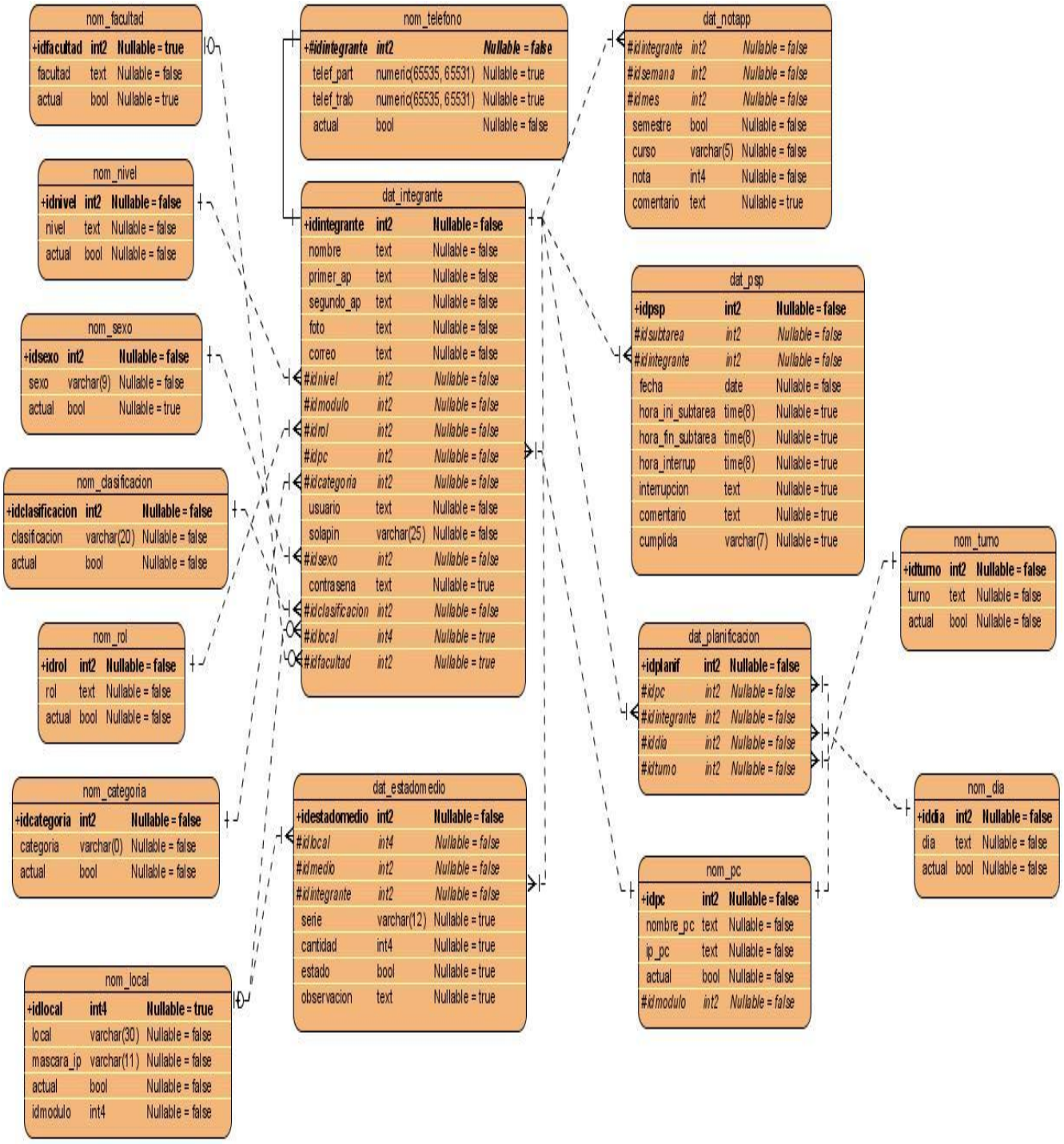
Submodelo 3


nom_local
+idlocal int4 Nullable = true
local varchar(30) Nullable = false
mascara_ip varchar(11) Nullable = false
actual bool Nullable = false
idmodulo int4 Nullable = false
dat_asistencia
+idasistencia int2 Nullable = false
idintegrante int4 Nullable = false
fecha varchar(10) Nullable = false
hora_ent varchar(8) Nullable = false
hora_sal varchar(8) Nullable = true
autorizado int4 Nullable = true
idsupervisor int4 Nullable = false
motivo_aus text Nullable = true
#idlocal int4 Nullable = false
dat_estadomedio
+idestadomedio int2 Nullable = false
#idlocal int4 Nullable = false
#idmedio int2 Nullable = false
#idintegrante int2 Nullable = false
serie varchar(12) Nullable = true
cantidad int4 Nullable = true
estado bool Nullable = true
observacion text Nullable = true
nom_modulo
+idmodulo int2 Nullable = false
modulo text Nullable = false
sitio text Nullable = false
actual bool Nullable = false
#idproyecto int2 Nullable = false
descripcion varchar(0) Nullable = true
nom_web
+#idmodulo int2 Nullable = false
dir_web text Nullable = false
usuario text Nullable = false
contrasena text Nullable = false
actual bool Nullable = true
nom_proyecto
+idproyecto int2 Nullable = false
proyecto varchar(0) Nullable = false
actual bool Nullable = false
descripcion varchar(0) Nullable = false
nom_documento
+#idmodulo int2 Nullable = false
dir_docum text Nullable = false
usuario text Nullable = false
contrasena text Nullable = false
actual bool Nullable = false
nom_bd
+#idmodulo int2 Nullable = false
dir_bd text Nullable = false
esquema text Nullable = false
usuario text Nullable = false
contrasena text Nullable = false
actual bool Nullable = true
nom_pc
+idpc int2 Nullable = false
nombre_pc text Nullable = false
ip_pc text Nullable = false
actual bool Nullable = false
#idmodulo int2 Nullable = false

Submodelo 4


nom_tarea
+idtarea int2 Nullable = false
tarea text Nullable = false
actual bool Nullable = true
dat_subtarea
+idsubtarea int2 Nullable = false
#idtarea int2 Nullable = false
#idintegrante int2 Nullable = false
#idmodulo int2 Nullable = false
subtarea text Nullable = false
fecha_ini date Nullable = false
fecha_fin date Nullable = false
estado varchar(15) Nullable = false
dat_psp
+idpsp int2 Nullable = false
#idsubtarea int2 Nullable = false
#idintegrante int2 Nullable = false
fecha date Nullable = false
hora_ini_subtarea time(8) Nullable = true
hora_fin_subtarea time(8) Nullable = true
hora_interrup time(8) Nullable = true
interrupcion text Nullable = true
comentario text Nullable = true
cumplida varchar(7) Nullable = true
dat_integrante
+idintegrante int2 Nullable = false
nombre text Nullable = false
primer_ap text Nullable = false
segundo_ap text Nullable = false
foto text Nullable = false
correo text Nullable = false
#idnivel int2 Nullable = false
#idmodulo int2 Nullable = false
#idrol int2 Nullable = false
#idpc int2 Nullable = false
#idcategoria int2 Nullable = false
usuario text Nullable = false
solapin varchar(25) Nullable = false
#idsexo int2 Nullable = false
contrasena text Nullable = true
#idclasificacion int2 Nullable = false
#idlocal int4 Nullable = true
#idfacultad int2 Nullable = true
nom_modulo
+idmodulo int2 Nullable = false
modulo text Nullable = false
sitio text Nullable = false
actual bool Nullable = false
#idproyecto int2 Nullable = false
descripcion varchar(0) Nullable = true

Submodelo 5


nom_facultad
+idfacultad int2 Nullable = true
facultad text Nullable = false
actual bool Nullable = true
nom_telefono
+#idintegrante int2 Nullable = false
telef_part numeric(65535, 65531) Nullable = true
telef_trab numeric(65535, 65531) Nullable = true
actual bool Nullable = false
dat_notapp
#idintegrante int2 Nullable = false
#idsemana int2 Nullable = false
#idmes int2 Nullable = false
semestre bool Nullable = false
curso varchar(5) Nullable = false
nota int4 Nullable = false
comentario text Nullable = true
nom_nivel
+idnivel int2 Nullable = false
nivel text Nullable = false
actual bool Nullable = false
dat_integrante
+idintegrante int2 Nullable = false
nombre text Nullable = false
primer_ap text Nullable = false
segundo_ap text Nullable = false
foto text Nullable = false
correo text Nullable = false
#idnivel int2 Nullable = false
#idmodulo int2 Nullable = false
#idrol int2 Nullable = false
#idpc int2 Nullable = false
#idcategoria int2 Nullable = false
usuario text Nullable = false
solapin varchar(25) Nullable = false
#idsexo int2 Nullable = false
contrasena text Nullable = true
#idclasificacion int2 Nullable = false
#idlocal int4 Nullable = true
#idfacultad int2 Nullable = true
dat_psp
+idpsp int2 Nullable = false
#idsubtarea int2 Nullable = false
#idintegrante int2 Nullable = false
fecha date Nullable = false
hora_ini_subtarea time(6) Nullable = true
hora_fin_subtarea time(6) Nullable = true
hora_interrup time(6) Nullable = true
interrupcion text Nullable = true
comentario text Nullable = true
cumplida varchar(7) Nullable = true
nom_sexo
+idsexo int2 Nullable = false
sexo varchar(9) Nullable = false
actual bool Nullable = true
nom_clasificacion
+idclasificacion int2 Nullable = false
clasificacion varchar(20) Nullable = false
actual bool Nullable = false
nom_turno
+idturno int2 Nullable = false
turno text Nullable = false
actual bool Nullable = false
nom_rol
+idrol int2 Nullable = false
rol text Nullable = false
actual bool Nullable = false
dat_planificacion
+idplanif int2 Nullable = false
#idpc int2 Nullable = false
#idintegrante int2 Nullable = false
#iddia int2 Nullable = false
#idturno int2 Nullable = false
nom_categoria
+idcategoria int2 Nullable = false
categoria varchar(0) Nullable = false
actual bool Nullable = false
dat_estadomedio
+idestadomedio int2 Nullable = false
#idlocal int4 Nullable = false
#idmedio int2 Nullable = false
#idintegrante int2 Nullable = false
serie varchar(12) Nullable = true
cantidad int4 Nullable = true
estado bool Nullable = true
observacion text Nullable = true
nom_dia
+iddia int2 Nullable = false
dia text Nullable = false
actual bool Nullable = false
nom_pc
+idpc int2 Nullable = false
nombre_pc text Nullable = false
ip_pc text Nullable = false
actual bool Nullable = false
#idmodulo int2 Nullable = false
nom_local
+idlocal int4 Nullable = true
local varchar(30) Nullable = false
mascara_ip varchar(11) Nullable = false
actual bool Nullable = false
idmodulo int4 Nullable = false

2.5 Representación de las clases persistentes.

A continuación se representan las clases persistentes con sus atributos y tipos de datos correspondientes.

Nombre: dat_asistencia	
Tipo de clase: entidad	
Atributo	**Tipo**
idasistencia	int
idintegrante	int
fecha	varchar
hora_ent	varchar
hora_sal	varchar
idsupervisor	int
autorizado	int
motivo_aus	text
idlocal	int

Nombre: dat_contacto	
Tipo de clase: entidad	
Atributo	**Tipo**
idcontacto	int
usuario	text
comentario	text
asunto	text
fecha	varchar

Nombre: dat_estadomedio	
Tipo de clase: entidad	
Atributo	**Tipo**
idestadomedio	int
idlocal	int
idmedio	int
idintegrante	int
serie	varchar
cantidad	int
estado	bool
observación	text

Nombre: dat_integrante	
Tipo de clase: entidad	
Atributo	**Tipo**
idintegrante	int
nombre	text
primer_ap	text
segundo_ap	text
foto	text
correo	text
idnivel	int
idmodulo	int
idrol	int
idpc	int
idcategoria	int
usuario	text
solapin	varchar
idsexo	int
contraseña	text
idcalsificacion	int
idlocal	int
idfacultad	int

Nombre: dat_planificacion	
Tipo de clase: entidad	
Atributo	**Tipo**
idplanific	int
idpc	int
idintegrante	int
idia	int
idturno	int

Nombre: dat_psp	
Tipo de clase: entidad	
Atributo	**Tipo**
idpsp	int
idsubtarea	int
idintegrante	int
fecha	date
hora_ini_subtarea	time
hora_fin_subtarea	time
hora_interrup	time
interrupcion	text

comentario	text
cumplida	varchar

Nombre: dat_recurso	
Tipo de clase: entidad	
Atributo	**Tipo**
idrecurso	int2
nom_integ	varchar
asunto	varchar
direccion	varchar
tamanno	numeric
autorizado	bool

Nombre: dat_subtarea	
Tipo de clase: entidad	
Atributo	**Tipo**
idsubtarea	int
idtarea	int
idintegrante	int
idmodulo	int
subtarea	text
fecha_ini	date
fecha_fin	date
estado	varchar

2.6 Diseño de la Base de Datos.

Las bases de datos necesitan de una definición de su estructura que le permita almacenar datos, reconocer el contenido, y recuperar la información. La estructura tiene que ser desarrollada para satisfacer las necesidades de las aplicaciones que la usarán.

La puesta en práctica de la base de datos es el paso final en el desarrollo de aplicaciones de soporte del negocio. Se conforman con los requisitos del proceso del negocio, que son la primera abstracción de la vista de la base de datos.

Para el diseño de la base de datos se realizó el diagrama entidad relación en el Visual Paradigm.

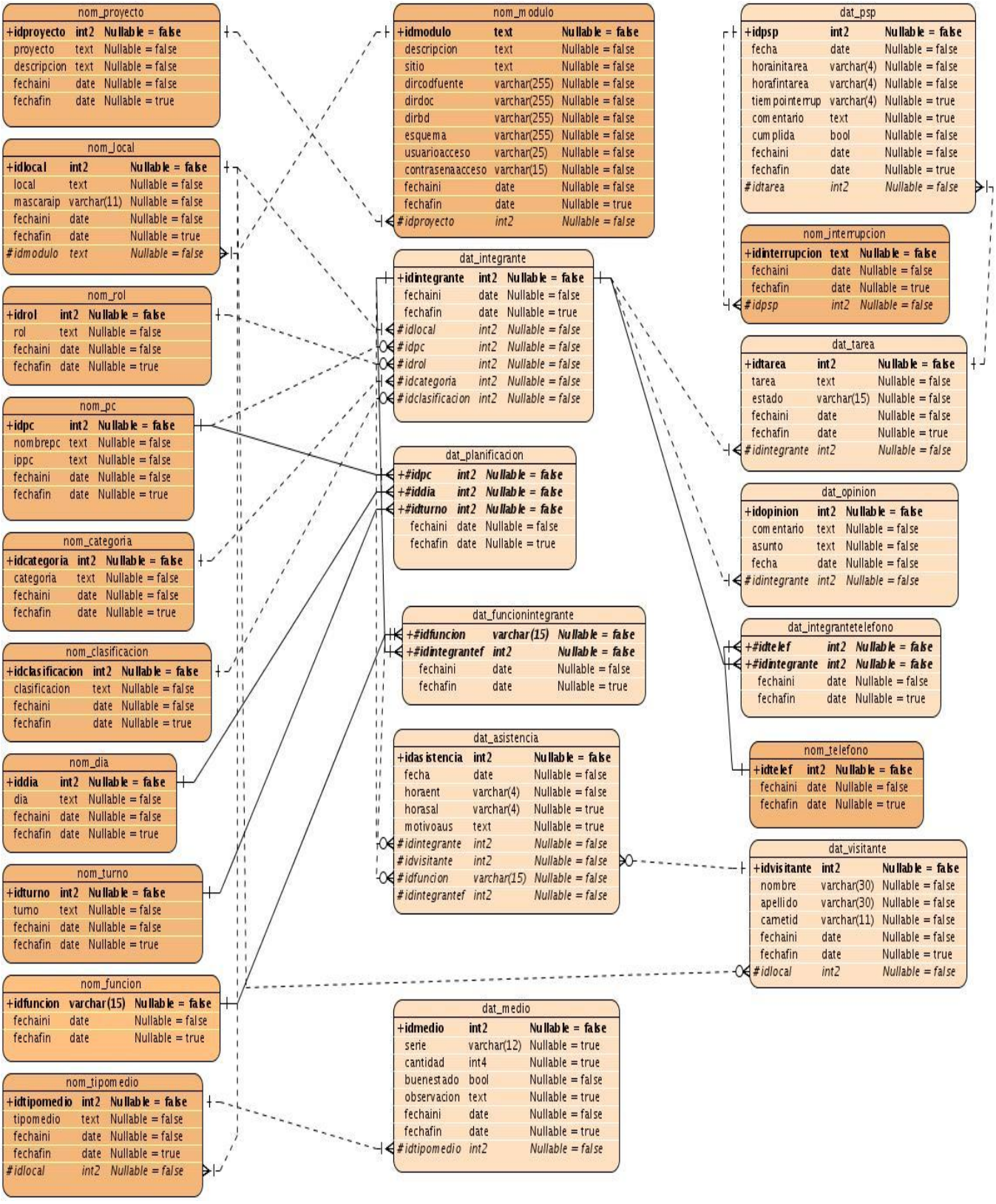
Una clase persistente (persistent) es una clase entidad que tiene la capacidad de mantener su valor en el espacio y en el tiempo. Lo contrario son las clases temporales (transient) que son manejadas y almacenadas por el sistema en tiempo de ejecución por lo que dejan de existir cuando termina el programa. (8)

En el diagrama entidad relación se especificaron los detalles físicos de la base de datos.

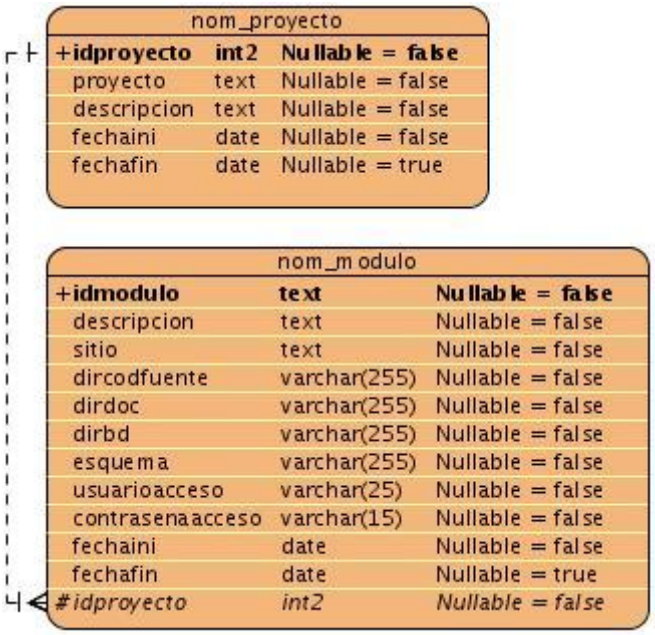
2.6.1 Diagrama Entidad Relación de la base de datos propuesta.

La siguiente figura representa el diagrama entidad relación correspondiente a la base de datos propuesta para la aplicación informática “Sistema para el Control de los Recursos del Proyecto”. A continuación se representan y describen los diferentes submodelos en los que se ha fragmentado la misma para una mejor comprensión.

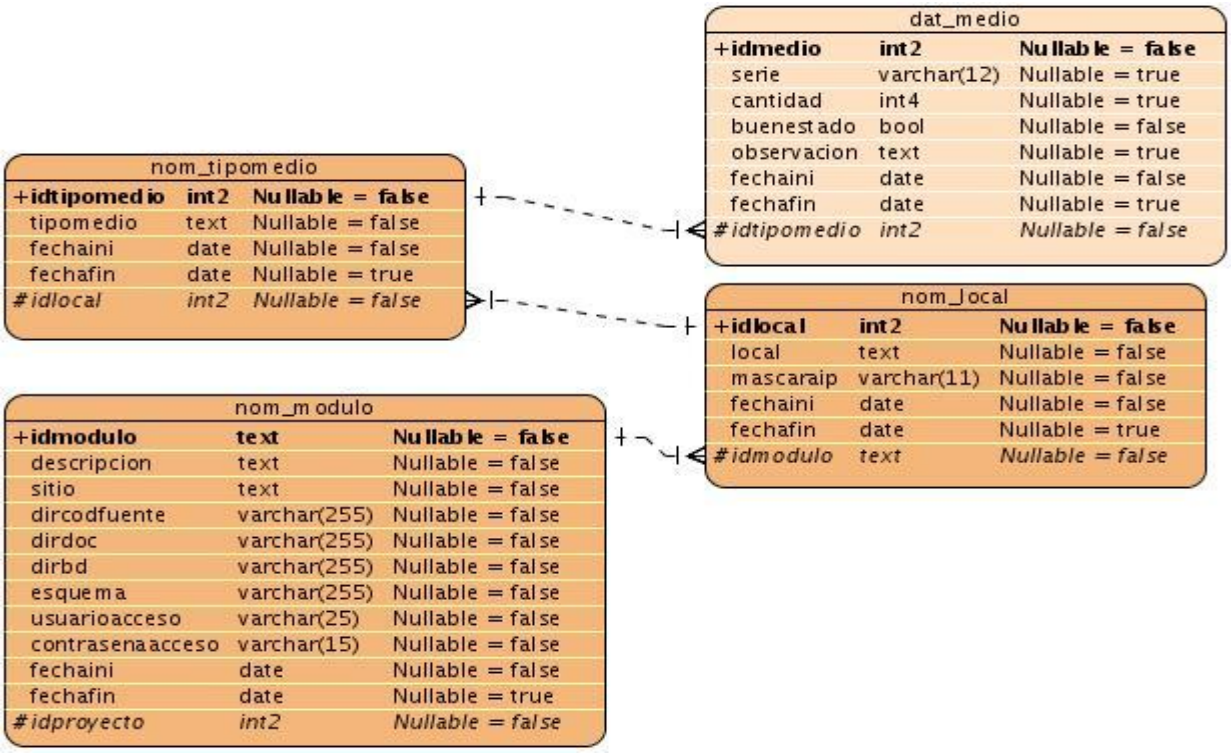
Diagrama general



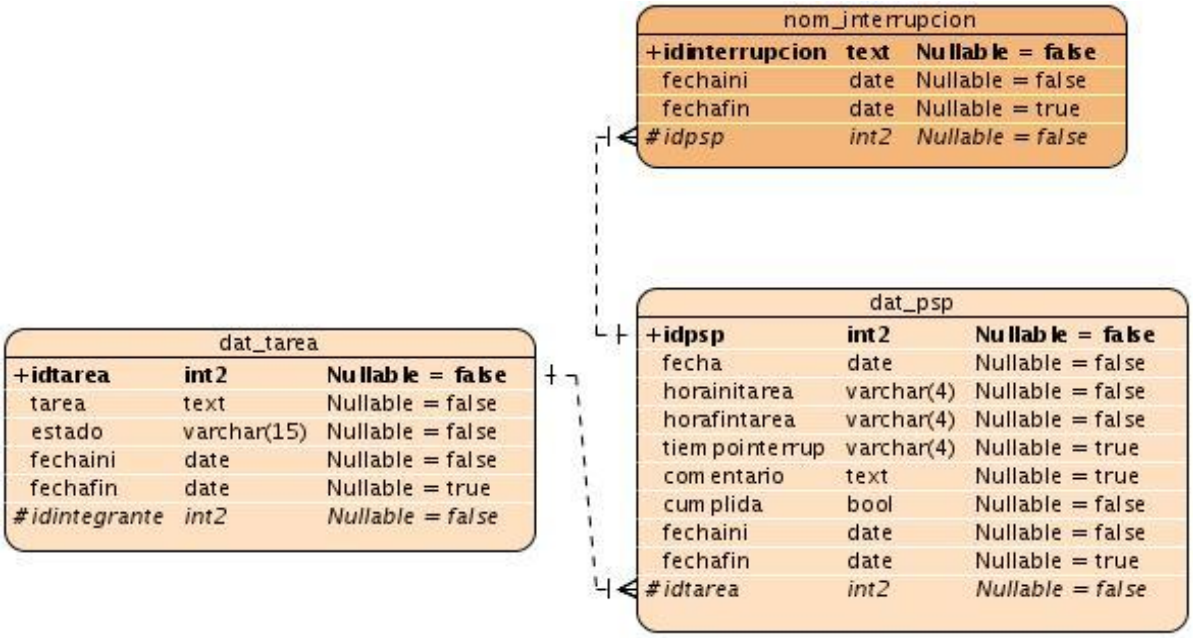
Submodelo1



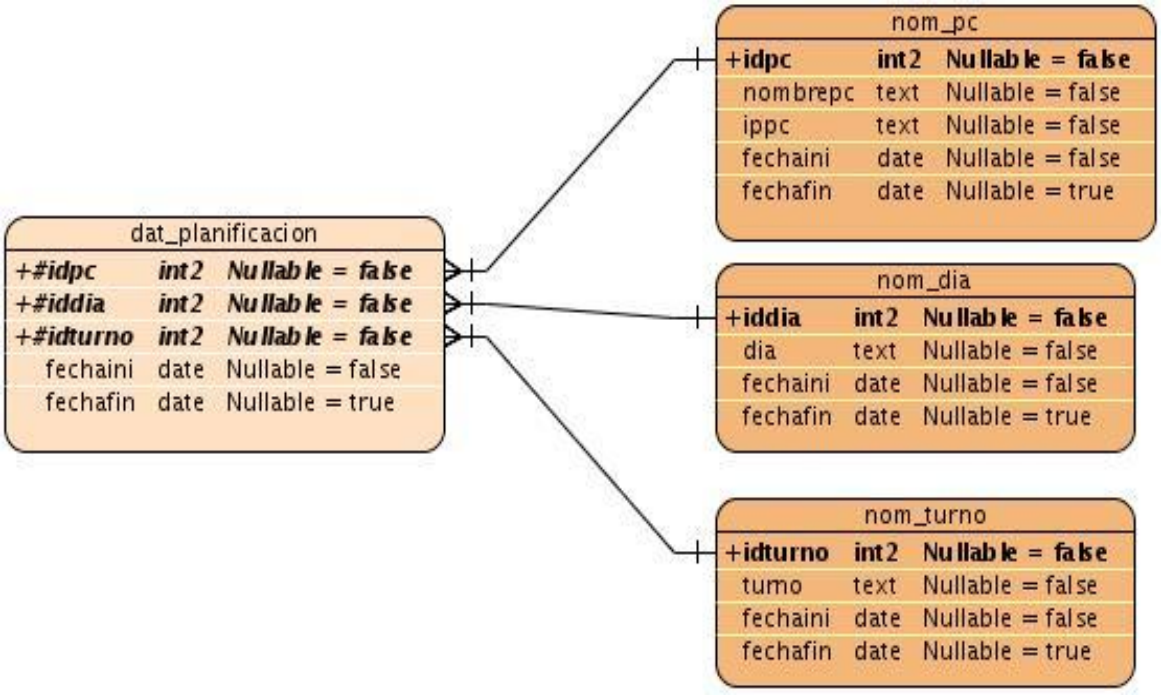
Submodelo 2



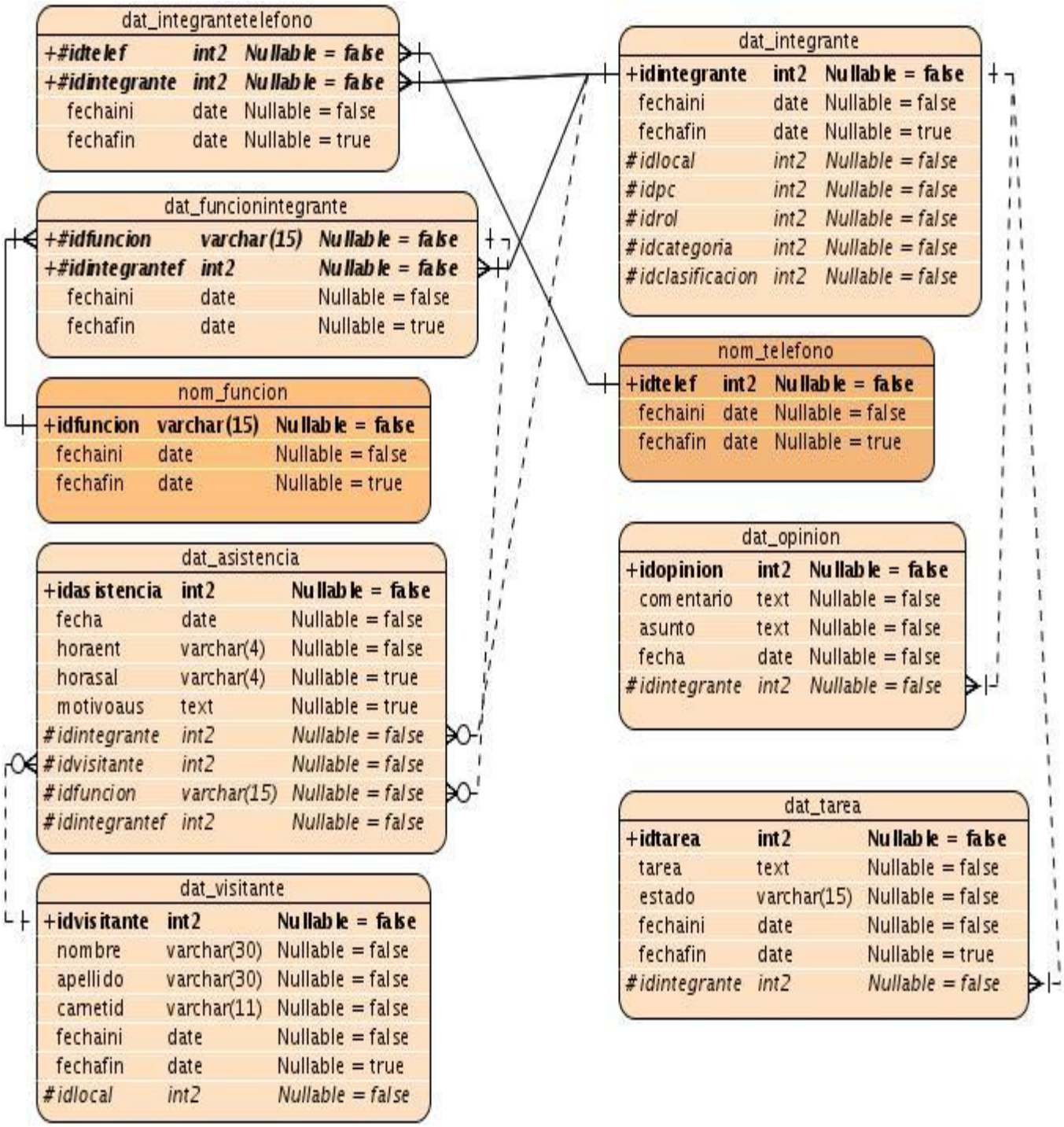
Submodelo 3



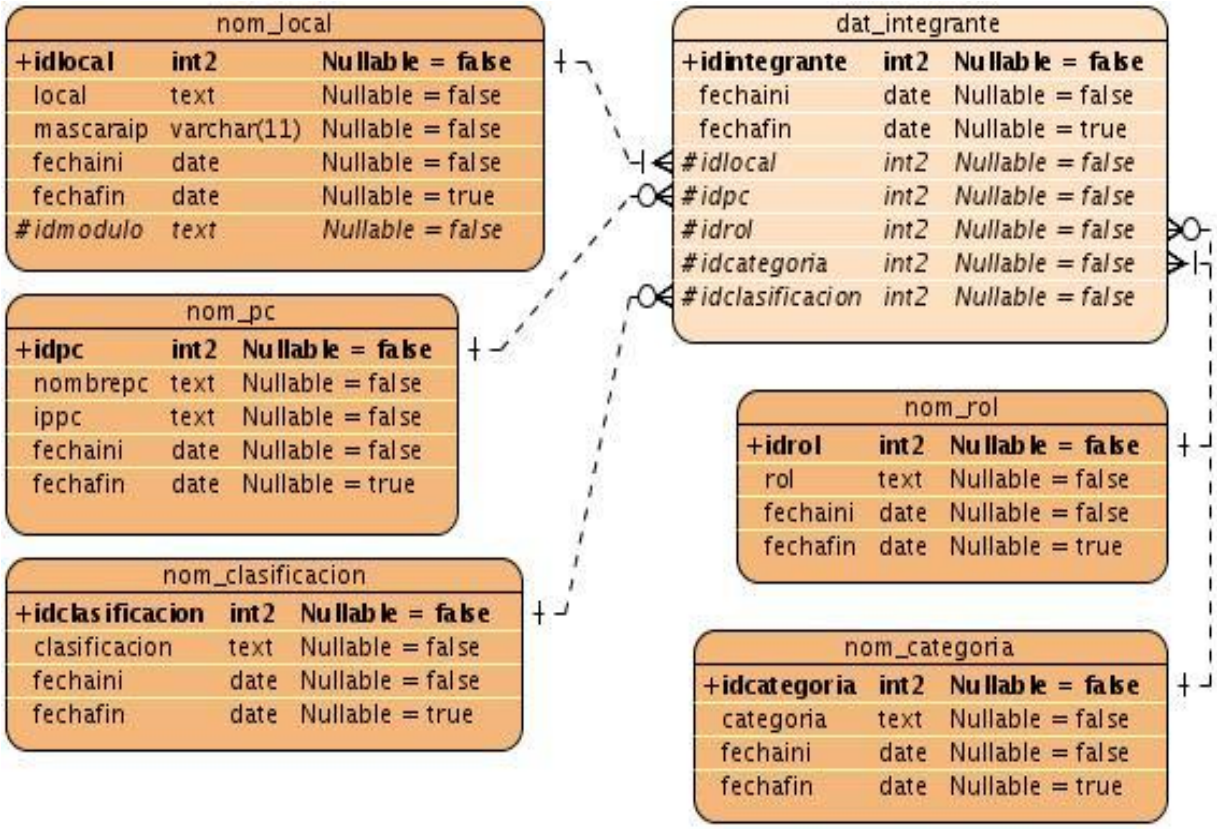
Submodelo 4



Submodelo 5


dat_integrantetelefono
+#idtelef int2 Nullable = false
+#idintegrante int2 Nullable = false
fechaini date Nullable = false
fechafin date Nullable = true
dat_integrante
+idintegrante int2 Nullable = false
fechaini date Nullable = false
fechafin date Nullable = true
#idlocal int2 Nullable = false
#idpc int2 Nullable = false
#idrol int2 Nullable = false
#idcategoria int2 Nullable = false
#idclasificacion int2 Nullable = false
dat_funcionintegrante
+#idfuncion varchar(15) Nullable = false
+#idintegrantef int2 Nullable = false
fechaini date Nullable = false
fechafin date Nullable = true
nom_telefono
+idtelef int2 Nullable = false
fechaini date Nullable = false
fechafin date Nullable = true
nom_funcion
+idfuncion varchar(15) Nullable = false
fechaini date Nullable = false
fechafin date Nullable = true
dat_opinion
+idopinion int2 Nullable = false
comentario text Nullable = false
asunto text Nullable = false
fecha date Nullable = false
#idintegrante int2 Nullable = false
dat_asistencia
+idasistencia int2 Nullable = false
fecha date Nullable = false
horaent varchar(4) Nullable = false
horasal varchar(4) Nullable = true
motivoaus text Nullable = true
#idintegrante int2 Nullable = false
#idvisitante int2 Nullable = false
#idfuncion varchar(15) Nullable = false
#idintegrantef int2 Nullable = false
dat_tarea
+idtarea int2 Nullable = false
tarea text Nullable = false
estado varchar(15) Nullable = false
fechaini date Nullable = false
fechafin date Nullable = true
#idintegrante int2 Nullable = false
dat_visitante
+idvisitante int2 Nullable = false
nombre varchar(30) Nullable = false
apellido varchar(30) Nullable = false
cametid varchar(11) Nullable = false
fechaini date Nullable = false
fechafin date Nullable = true
#idlocal int2 Nullable = false

Submodelo 6



2.6.2 Descripción de las tablas del diseño de base de datos propuesto.

En esta sección se describen de forma general los datos que se almacenan en cada tabla del diagrama entidad relación de la base de datos propuesta para la aplicación informática “Sistema para el Control de los Recursos del Proyecto”, al igual que se explica brevemente lo que representan todos sus atributos.

Nombre: integrante		
Descripción: Almacena los datos correspondientes a los miembros del proyecto.		
Atributo	**Tipo**	**Descripción**
idintegrante	int2	Número identificador del integrante.
fechaini	date	Fecha de alta del integrante.
fechafin	date	Fecha de baja del integrante.
idlocal	int2	Es el identificador del local al que pertenece el integrante.

idpc	int2	Identificador de la PC a la cual esta asignado el miembro.
idrol	int2	Identificador del rol o papel del miembro dentro del proyecto o modulo.
idcategoria	int2	Identificador de la categoría del integrante, nivel de acceso a la aplicación.
idclasificacion	int2	Identificador de la clasificación del integrante.(Ej.: profesor, militar, estudiante)

Nombre: asistencia		
Descripción: Almacena los datos correspondientes a la asistencia de los integrantes del proyecto.		
Atributo	**Tipo**	**Descripción**
idasistencia	int2	Número identificador de la asistencia.
fecha	date	Fecha en que se le está tomando la asistencia al integrante.
horaentrada	varchar(4)	Hora de entrada del integrante.
horasalida	varchar(4)	Hora de salida del integrante.
motivoaus	text	Motivo de la ausencia del integrante.
idintegrante	int2	Número identificador del integrante.
idvisitante	int2	Número identificador del visitante.
idfuncion	varchar(15)	Identificador y descripción de la función.
idintegrantef	int2	Número identificador del integrante función.

Nombre: psp		
Descripción: Almacena los datos correspondientes a los registros PSP de los integrantes del proyecto.		
Atributo	**Tipo**	**Descripción**
idpsp	int2	Número identificador del registro psp.
fecha	date	Fecha de creación del registro.
horainitarea	time(4)	Hora en que se comienza a realizar la tarea.
horafintarea	time(4)	Hora en que se termina de realizar la tarea.
tiempointerrup	time(4)	Es la suma de todas las interrupciones que ha tenido el usuario durante la realización de la tarea.
comentario	text	Comentario adicional de la tarea.
cumplida	bool	Indica si se cumplió o no la tarea.
fechaini	date	Fecha de alta del registro psp.
fechafin	date	Fecha de baja del registro psp.
idtarea	int2	Identificador de la tarea seleccionada

Nombre: tarea		
Descripción: Almacena las tareas asignadas a los miembros de los proyectos o módulos.		
Atributo	**Tipo**	**Descripción**
idtarea	int2	Número identificador de una tarea.
tarea	text	Tarea asignada.
fechaini	date	Hora en que se comienza a realizar la tarea.
fechafin	date	Hora en que se termina de realizar la tarea.
estado	varchar(15)	Estado en que se encuentra la tarea.(Ej.:

		cumplida, en ejecución)
idintegrante	int2	Número identificador del integrante.

Nombre: planificación		
Descripción: Contiene los datos de la planificación realizada para cada miembro de los proyectos o módulos.		
Atributo	**Tipo**	**Descripción**
idpc	int2	Número identificador de una PC.
iddia	int2	Número identificador del día.
idturno	int2	Número identificador del turno.
fechaini	date	Fecha de alta de la planificación.
fechafin	date	Fecha de baja de la planificación.

Nombre: integranteteléfono		
Descripción: Almacena los datos correspondientes de los teléfonos de los integrantes.		
Atributo	**Tipo**	**Descripción**
idintegrante	int2	Número identificador del integrante.
idtelef	int2	Número de teléfono.
fechaini	date	Fecha de alta del teléfono del integrante.
fechafin	date	Fecha de baja del teléfono del integrante.

Nombre: opinión		
Descripción: Almacena las opiniones o comentarios de los integrantes del proyecto acerca de la aplicación.		
Atributo	**Tipo**	**Descripción**
idopinion	int2	Número identificador de una opinión.
comentario	text	Texto que les escriben los usuarios al moderador.
asunto	text	Asunto del comentario.
fecha	date	Fecha emisión del Mensaje.
idintegrante	int2	Número identificador del integrante que introdujo la opinión.

Nombre: medio		
Descripción: Almacena los datos de un medio determinado del proyecto.		
Atributo	**Tipo**	**Descripción**
idmedio	int2	Número identificador de un medio.
buenestado	bool	Estado en que se encuentra un medio.
observación	text	Comentario adicional acerca del medio.
serie	varchar(12)	Número de serie del medio.
cantidad	int4	En caso de que el medio no tenga número de serie se pone la cantidad que existe.
fechaini	date	Fecha de alta del medio.
fechafin	date	Fecha de baja del medio.

idtipomedio	int2	Número identificador del tipo de medio.

Nombre: funcionintegrante		
Descripción: Almacena los datos correspondientes a las funciones que realiza cada integrante del proyecto.		
Atributo	**Tipo**	**Descripción**
idfuncion	varchar(15)	Identificador y descripción de la función.
idintegrantef	int2	Número identificador del integrante función.
fechaini	date	Fecha de alta del integrante.
fechafin	date	Fecha de baja del integrante.

Nombre: visitante		
Descripción: Almacena los datos de un visitante.		
Atributo	**Tipo**	**Descripción**
idvisitante	int2	Número identificador del visitante.
nombre	varchar(30)	Nombre del visitante.
apellido	varchar(30)	Apellido del visitante.
carnetid	varchar(11)	Número del carnet de identidad del visitante.
fechaini	date	Fecha de alta del visitante.
fechafin	date	Fecha de baja del visitante.

2.7 Análisis de optimización de consultas.

Las consultas son pedidos que se le hacen a la base de datos, para obtener información que hay guardada en ella. Durante la ejecución de las consultas se pasa por 4 fases:

- Parser: Chequea la sintaxis de la consulta y genera el árbol de consulta (Query tree).
- Rewriter: Procesa el árbol de consulta y comprueba la existencia de reglas reescribiendo de nuevo la consulta si es necesario.
- Planer: Genera el plan de consulta a partir del árbol de consulta analizando todos los posibles caminos y escogiendo el de menor coste.
- Executor: Recorre recursivamente el árbol del plan de consulta y accede al disco para obtener los datos necesarios. Realiza las uniones entre tablas, cálculos y ordenamientos necesarios para devolver las tuplas resultado de las consultas. (9)

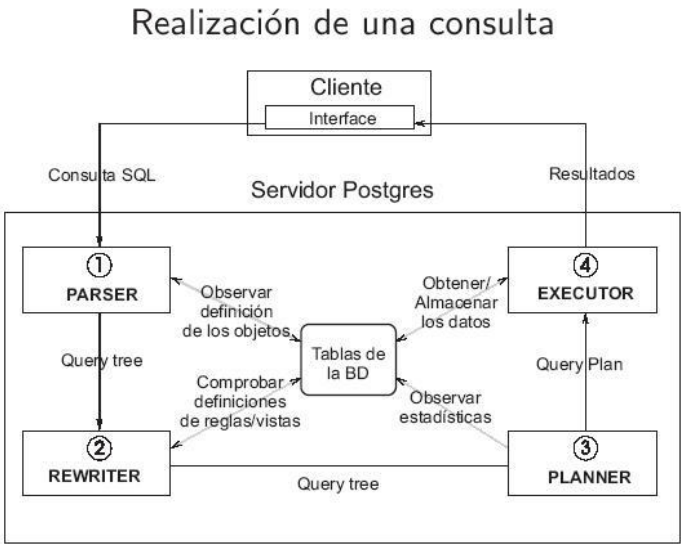


Figura 10. Diagrama de ejecución de una consulta en PostgreSQL.

La optimización de consultas es algo necesario para un mejor funcionamiento y rapidez de la base de datos, esto consiste en buscar las mejores opciones para realizar consultas.

Para analizar el tema de optimización de consultas en la base de datos propuesta se definieron determinados parámetros que son de alta importancia para que el diseño sea el más adecuado posible.

1. Diseño de las tablas.

- Todas las tablas de la base de datos están normalizadas hasta la tercera forma normal, lo que evita la duplicidad en los datos y permite que se aproveche al máximo el almacenamiento en las mismas.
- El tamaño de los campos de la tabla está ajustado al máximo para evitar desperdiciar espacio.

2. Gestión y elección de los índices.

Los índices son campos elegidos arbitrariamente por el constructor de la base de datos para permitir la búsqueda a partir de dicho campo a una velocidad notablemente superior. Sin embargo, esta ventaja se ve contrarrestada por el hecho de ocupar mucha más memoria (el doble más o menos) y de requerir para su inserción y actualización un tiempo de proceso superior.

En las tablas de la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” están indexados los campos que son llaves y los que contienen valores únicos, que son

los índices que PostgreSQL admite por defecto. No es necesario crear más índices porque precisamente por medio de estos campos se van a realizar la mayoría de las consultas.

3. Realización de las consultas.

Las consultas de la aplicación no se van a realizar en el mismo gestor de datos, con motivo de que si ocurre un cambio de sistema no se tenga que implementar toda esta parte de nuevo para otro gestor. Se recomienda que a la hora de confeccionar las mismas se tengan en cuenta los siguientes pasos.

Campos a Seleccionar:

A la hora de realizar una consulta utilizando varias tablas se debe especificar a qué tabla pertenece cada campo, así se le ahorrará tiempo al gestor de localizar donde se encuentra ubicado cada atributo.

Ejemplo:

En lugar de

```
SELECT    idasistencia, idintegrante
FROM      dat_integrante,      dat_asistencia
WHERE     (idintegrante=idintegrante)          AND  autorizado <> 0;
```

es conveniente usar:

```
SELECT    dat_asistencia.idasistencia, dat_integrante.idintegrante,
FROM      dat_integrante, dat_asistencia
WHERE     (dat_integrante.idintegrante = dat_asistencia.idintegrante) AND
          dat_asistencia.autorizado <> 0;
```

Campos de Filtro:

Se debe procurar elegir en la cláusula WHERE aquellos campos que formen parte de la clave del fichero por el cual se interroga. Además se deben especificar en el mismo orden en el que estén definidos en la clave.

Se recomienda interrogar siempre por campos que sean clave.

PostgreSQL cuenta con un comando denominado VACUUM, que limpia y analiza una base de datos. El VACUUM da la posibilidad de imprimir un reporte detallado de la actividad de análisis para cada tabla y de actualizar las estadísticas de columnas usadas por el optimizador para determinar la manera más eficiente de ejecutar una consulta. Las estadísticas representan la dispersión de los datos en cada columna.

La ejecución de VACUUM periódicamente aumenta la velocidad de la base de datos al procesar las consultas del usuario. La consulta VACUUM puede ser ejecutada en cualquier momento, particularmente, después de copiar una clase grande en PostgreSQL o después de borrar un gran número de registros. Esto actualizará los catálogos del sistema con todos los cambios recientes, y permitirá al organizador de consultas de PostgreSQL tomar las mejores decisiones al planear las consultas de los usuarios.

El administrador de la base de datos debe realizar un VACUUM periódicamente para actualizar los cambios y aumentar la velocidad de las consultas.

Otro comando que tiene PostgreSQL es el EXPLAIN. Este comando muestra el plan de ejecución que el planificador de PostgreSQL genera para la consulta dada. El plan de ejecución muestra la manera en que serán escaneadas las tablas referenciadas. La parte más crítica de la presentación es el costo estimado de ejecución de la consulta, que es la suposición del planificador sobre el tiempo que tomará correr la consulta (medido en unidades de captura de páginas de disco).

Esto es de mucha ayuda a la hora de construir una consulta, ya que analiza la manera más óptima de ejecutarla, brindando al usuario la posibilidad de escoger entre las de menos costo. Es de gran importancia para determinar cuáles serán las mejores formas de crear las consultas que se utilizarán para acceder a la información de la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto”, que por supuesto, serán las de menos costo.

A continuación se presentan algunos aspectos específicos donde se puede mejorar el rendimiento de una consulta:

1. La cláusula DISTINCT es costosa de ejecutar porque generalmente involucra un ordenamiento de las tuplas resultantes para eliminar duplicados. Es necesario analizar si realmente hace falta usar esa cláusula.

2. Las subconsultas en muchos manejadores se ejecutan ineficientemente. Una razón para ello es que al estar anidada no se utiliza algún índice relevante, al eliminar la subconsulta es posible lograr que el manejador utilice el índice apropiado.

3. En algunas consultas el usuario almacena resultados intermedios explícitamente en tablas temporales. Estas tablas pueden bajar el rendimiento de la consulta por dos razones, una porque fuerza un orden de ejecución que quizás no sea ideal y segundo, porque obliga a ejecutar actualizaciones al Diccionario de Datos (DD) lo cual tiene el peligro de convertirlo en un cuello de botella.

4. Por otro lado, las tablas temporales pueden tener un efecto positivo al reescribir consultas que contienen subconsultas correlacionadas complejas. Las subconsultas correlacionadas pueden ejecutarse ineficientemente, pero si se logra calcular primero las tuplas de la subconsulta y almacenarlas en una tabla temporal, se puede reescribir la consulta original, sin la subconsulta, accediendo a la tabla temporal.

5. Otro uso beneficioso de las tablas temporales es para evitar el uso de la cláusula ORDER BY que puede ser costosa.

6. Las condiciones de join se pueden evaluar más eficientemente contra un índice primario. Y en general, en términos de rendimiento es preferible evaluar una condición de igualdad numérica que una condición de igualdad sobre cadenas (”strings”) de caracteres.

7. Es preferible no usar la cláusula HAVING si la condición deseada se puede expresar en la cláusula WHERE.

8. Es importante conocer las particularidades del manejador, esto se refiere a conocer como han sido implementadas las rutinas de procesamiento de las consultas. Por ejemplo, es importante saber, al existir una condición disyuntiva (con OR) en la consulta si se usan o no los índices existentes.

9. Otra particularidad importante del manejador que es bueno conocer es si el orden de las tablas en la cláusula FROM puede afectar la implementación del join utilizada, posiblemente esto es relevante para joins que involucran 5 tablas o más.

10. El uso de vistas pueden causar una ejecución ineficiente de consultas. Muchas veces la ejecución de consultas sobre las tablas base es más eficiente. (5)

A continuación se analiza una consulta simple donde la tabla nom_modulo está indexada por su llave id_modulo y se relaciona con la tabla nom_proyecto la cual esta indexada por su llave id_proyecto.

```
SELECT    nom_modulo.idmodulo, nom_modulo.descripcion, nom_modulo.sitio,
          nom_modulo.dirbd, nom_modulo.dirdoc, nom_modulo.dircodfuente,
          nom_proyecto.proyecto

FROM      nom_modulo, nom_proyecto

WHERE     (nom_modulo.idproyecto = nom_proyecto.idproyecto) AND
           (nom_modulo.idproyecto >=1)      AND   (nom_modulo.idproyecto <= 150)
```

Se obtiene un resultado de 636 tuplas en un tiempo de 0.08 segundos.

Para esta otra:

```
SELECT    nom_modulo.idmodulo, nom_modulo.descripcion, nom_modulo.sitio, nom_modulo.dirbd,
          nom_modulo.dirdoc, nom_modulo.dircodfuente, nom_proyecto.proyecto

FROM      nom_modulo, nom_proyecto

WHERE     (nom_modulo.idproyecto = nom_proyecto.idproyecto) AND
          (nom_modulo.idproyecto BETWEEN 1 AND 150)
```

Se obtienen 636 tuplas en un tiempo 0.03 segundos.

De los resultados obtenidos se puede concluir que si se tiene un dominio bastante amplio de las maneras de hacer las consultas y con las herramientas que permitan calcular tiempos de ejecución de dichas consultas, se pueden hacer peticiones al servidor de base de datos con mejores resultados en cuanto al tiempo de respuesta.

2.8 Conclusiones.

En el diseño propuesto de la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto”, las tablas han sido construidas de un modo óptimo para mejorar el tiempo de respuestas de las consultas. Se ha reducido la redundancia en la información debido a que ya no existen datos repetidos innecesariamente, con lo cual también se aumenta la integridad de los mismos. Las tres fases empleadas en la normalización han sido utilizadas correctamente, evitando que se creen anomalías en las consultas realizadas para extraer e insertar datos en el sistema.

CAPÍTULO 3: VALIDACIÓN DEL DISEÑO

3.1 Introducción.

El presente capítulo trata de la validación teórica y funcional del diseño de la base de datos realizado. En la primera se recogen aspectos como la integridad, seguridad y normalización de la base de datos, así como del análisis de la redundancia de la información. La segunda se basa en las pruebas realizadas a la base de datos para comprobar su funcionamiento.

3.2 Validación teórica del diseño.

No basta solamente con realizar el diseño de una base de datos, se tienen que tener en cuenta también los aspectos que garanticen la consistencia, integridad y seguridad de la misma. Los Sistemas Gestores de Bases de Datos deben cumplir estos objetivos para facilitar la manipulación y confidencialidad de los datos.

3.2.1 Integridad.

La integridad de los datos se contempla en diferentes niveles. Las restricciones de dominio, transacciones y entidades definen las reglas para el mantenimiento de la integridad de las relaciones individuales. Las relaciones de integridad referencial aseguran que se mantienen las asociaciones necesarias entre las relaciones. Las restricciones de integridad de la base de datos gobiernan la misma como un todo y las restricciones de integridad de transacciones controlan la forma en que se manipulan los datos, dentro de una o entre múltiples bases de datos. (10)

Integridad de Dominio.

Una restricción de integridad de dominio es una regla que define valores válidos para los atributos de las diferentes tablas de una base de datos. Puede ser necesario definir más de una restricción de dominio para describir por completo un dominio.

El primer paso es la elección de un tipo de datos lógico (un tipo de datos lógico puede ser una fecha, cadena, número). Si un valor de una cadena es de no más de 30 caracteres se debe usar un char (30). Por ejemplo en el nomenclador de locales la mascaraip admite caracteres y números por lo que es de tipo varchar y su tamaño no sobrepasa la cantidad máxima de 11 por lo tanto su dominio es varchar (11).

El siguiente aspecto a considerar sobre Integridad de dominio es si al dominio se le permite contemplar valores desconocidos o inexistentes, sabiendo que no es lo mismo desconocido, que inexistente. Esto se puede explicar de la siguiente forma; es posible que en la tabla dat_psp el campo fecha sea desconocido, pero si efectivamente se ha introducido un psp, tiene que haber una fecha en que se creó el mismo y por tanto no puede ser inexistente, es decir, no puede ser nulo. Otro ejemplo es el integrante en la tabla dat_asistencia. Si el integrante tiene una ausencia se debe introducir el motivo de la misma, pero si no está en esta condición el motivo sería inexistente y el campo admitiría valores nulos.

El último aspecto de integridad de dominio es que se deberá definir el conjunto de los valores representados por un dominio lo más específicamente que se pueda. En la tabla dat_medio existe un campo que define el estado del medio, este es el campo estado. El dominio para este campo admite solamente el conjunto de valores siguientes: true, para especificar que el estado es bueno y false para cuando el estado sea malo, evitando que el usuario introduzca un dato no adecuado que afecte la integridad del sistema.

Integridad de Transacciones.

Define los estados por los que una tupla puede pasar válidamente, como son: introducido, pendiente, seleccionado, enviado, cancelado y terminado. Está encargada de asegurar que el estado de una determinada tupla, no pase de un estado inicial a uno final, sin haber pasado por los estados intermedios. En el caso de la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” no se encuentran restricciones de este tipo.

Integridad de Entidades.

Las restricciones de entidades aseguran la integridad de las entidades que son modeladas por el sistema.

Un caso de este tipo de restricción es el de las llaves primarias para cada tabla. En el caso de la base de datos que se está analizando cada entidad tiene definida su llave primaria, que no puede ser nula ni se puede repetir.

Integridad Referencial.

Otra de las reglas, es la llamada restricción de integridad referencial que se especifica entre dos relaciones y sirve para mantener la consistencia entre tuplas de las dos relaciones. Establece que una tupla en una relación que haga referencia a otra, deberá referirse a un valor existente en esa

relación. Un ejemplo de aplicación de este concepto en el modelo de datos es el siguiente: se asegura que los atributos idlocal e idpc de la tabla dat_integrante sean llaves foráneas, debido a que provienen y son llaves primarias de las entidades nom_local y nom_pc respectivamente. Además cumplen con las restricciones siguientes:

1. Los números identificadores del local y de la pc en la tabla dat_integrante tienen los mismos dominios que en sus entidades de origen.

2. Al introducir los números identificadores del local y de la pc en la tabla dat_integrante, éstos tienen que encontrarse en su entidad de origen, es decir, dat_integrante no puede tener un número identificador de local y de pc que no estén presentes en la tabla nom_local y nom_pc respectivamente.

Por lo planteado anteriormente se comprueba que en el modelo de datos se cumple con la regla de integridad referencial, que es una de las más importantes dentro de la integridad de la base de datos.

Integridad entre varias entidades de una Base de Datos.

Otra forma de integridad, que fue chequeada en la base de datos diseñada, es la que relaciona a varias entidades de la base de datos. Una forma de mantener la integridad es el control de concurrencia en la base de datos. A diferencia de la mayoría de otros sistemas de bases de datos que usan bloqueos para el control de concurrencia, PostgreSQL mantiene la consistencia de los datos en un modelo multiversión. Esto significa que mientras se consulta una base de datos, cada transacción ve una imagen de los datos (una versión de la base de datos) como si fuera tiempo atrás, sin tener en cuenta el estado actual de los datos que hay por debajo. Esto evita que la transacción vea datos inconsistentes que pueden ser causados por la actualización de otra transacción concurrente en la misma fila de datos, proporcionando aislamiento transaccional para cada sesión de la base de datos.

Existen cuatro niveles de aislamiento transaccional en función de tres hechos que deben ser tenidos en cuenta entre transacciones concurrentes. Estos hechos no deseados son:

- Lecturas sucias.

Una transacción lee datos escritos por una transacción no esperada, no cursada.

- Lecturas no repetibles.

Una transacción vuelve a leer datos que previamente había leído y encuentra que han sido modificados por una transacción cursada.

- Lecturas fantasmas.

Una transacción vuelve a ejecutar una consulta, devolviendo un conjunto de filas que satisfacen una condición de búsqueda y encuentra que otras filas que satisfacen la condición han sido insertadas por otra transacción cursada.

Los cuatro niveles de aislamiento transaccional se describen en la tabla en la siguiente.

	Lectura "sucia"	**Lectura no repetible**	**Lectura "fantasma"**
Lectura no cursada	Posible	Posible	Posible
Lectura cursada	No posible	Posible	Posible
Lectura repetible	No posible	No posible	Posible
Serializable	No posible	No posible	No posible

PostgreSQL ofrece lectura cursada y niveles de aislamiento serializables.

La Lectura cursada es el nivel de aislamiento por defecto en PostgreSQL. Cuando una transacción se ejecuta en este nivel, la consulta sólo ve datos cursados antes de que se ejecutara y nunca ve ni datos "sucios" ni los cambios en transacciones concurrentes cursados durante la ejecución de la consulta.

La serialización proporciona el nivel más alto de aislamiento transaccional. Cuando una transacción está en el nivel serializable, la consulta sólo ve los datos cursados antes de que la transacción comience y nunca ve ni datos sucios ni los cambios de transacciones concurrentes cursados durante la ejecución de la transacción. Por lo tanto, este nivel emula la ejecución de transacciones en serie, como si las transacciones fueran ejecutadas una detrás de otra, en serie, en lugar de concurrentemente.

Para controlar la concurrencia en la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” se empleará el modo serializable, para evitar que cuando haya más de un usuario trabajando concurrentemente en la base de datos se produzcan hechos no deseados entre las transacciones, que puedan afectar la integridad de los datos en el sistema.

3.2.2 Normalización de la Base de Datos.

La teoría de la normalización se ha desarrollado para obtener estructuras de datos eficientes que eviten las anomalías de actualización. Es la expresión formal del modo de realizar un buen diseño y

provee los medios necesarios para describir la estructura lógica de los datos en un sistema de información. Evita anomalías en la actualización y mejora la independencia de los datos, permitiendo realizar extensiones de la base de datos, afectando muy poco, o nada, a los programas de aplicación existentes que accedan a la base de datos.
Para normalizar una base de datos se deben realizar varias fases en orden. (11)

- Primera Fase Normal (1FN).
- Segunda Fase Normal (2FN).
- Tercera Fase Normal (3FN).

También existen la Forma Normal de Boyce-Codd (FNBC), la cuarta y la quinta formas normales.
La base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” ha sido normalizada hasta la tercera forma normal, debido a que de esta manera se resuelven los problemas de inconsistencia y redundancia en sus tablas.

- Primera Forma Normal:

Una entidad está en 1FN si cumple la propiedad de que sus dominios no tienen elementos que, a su vez, sean conjuntos.
La base de datos propuesta tiene un atributo multivaluado en la tabla dat_tarea llamado tarea, proponemos que se le de solución a este problema a nivel de programación ya que no se puede hacer un nomenclador que predefina todos las tareas que existen pues puede que surjan nuevas o dejen de existir cualquiera de las existentes en cualquier instante de tiempo.

- Segunda Forma Normal:

Se dice que una entidad está en 2FN si está en 1FN y si sus atributos no llaves (ni primarias ni candidatas) son funcional y completamente dependientes de la llave primaria. Lo relacionado con lo de la dependencia funcional completa se aplica solo a entidades con llaves compuestas.
La base de datos que se está analizando cumple con la 2FN. Todas sus tablas se encuentran en 1FN y las que tienen llaves compuestas, sus atributos no llaves son dependientes de la llave en su totalidad.

- Tercera Forma Normal:

Una entidad está en 3FN si está en 2FN y cada atributo no llave, depende directamente y no transitivamente, de la llave primaria.

En la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” en las tablas que presentan llaves compuestas, todos los atributos no llaves dependen directamente de la llave primaria, por tanto esta en 3FN.

3.2.3 Análisis de redundancia de información.

Con el diseño de la nueva base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” se ha disminuido considerablemente la redundancia en los datos que existía en el sistema anterior. Un punto muy importante para lograr lo anterior fue la normalización de la base de datos, posibilitando que en cada tabla estén presentes los datos que se necesiten realmente. A parte de lo anterior se disminuyeron los datos repetidos innecesariamente en las mismas.

3.2.4 Análisis de la seguridad de la Base de Datos.

La seguridad garantiza el acceso autorizado a los datos, para irrumpir cualquier intento de acceso no autorizado, ya sea por error del usuario o por mala intención.

Para proteger la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” el administrador de la base de datos va a ser el único que tendrá todos los privilegios para acceder al SGBD y darle mantenimiento a la misma, y solo el dará los privilegios a demás usuarios para insertar, modificar o recuperar datos. La información debe ser recuperada en caso de que ocurra algún error en el servidor de la base de datos o alguna falla de energía. Deben realizarse copias de seguridad de las bases de datos regularmente. Dado que PostgreSQL gestiona sus propios ficheros en el sistema, no se recomienda confiar en los sistemas de copia de seguridad del sistema para las copias de respaldo de las base de datos; no hay garantía de que los ficheros estén en un estado consistente que permita su uso después de la restauración.

Para la realización de copias a la base de datos, PostgreSQL cuenta con un comando denominado pg_dump que se encarga de hacerle un backup a la misma en el momento que el administrador decida, luego la información se recupera por medio de otro comando: pg_restore. Además, este SGBD se puede ejecutar en dos modos: fsync y no fsync. Si se ejecuta en el primer modo el gestor sería más lento, pero garantizaría que si el sistema operativo se bloquea o se produce una pérdida de energía, todos sus datos estarían almacenados y sin daños en el disco. El modo de ejecución de PostgreSQL en el servidor de base de datos de la aplicación será fsync, para evitar pérdida de la información en caso de que ocurra algún error inevitable en el sistema.

En PostgreSQL existen tres niveles de acceso:

El nivel 0 es el que se encarga sobre las máquinas (host) y los usuarios. Es decir es el permite configurar que máquina/s y/o usuario/s se pueden conectar a la Base de Datos. Utilizando las opciones del fichero de configuración pg_hba.conf.

El nivel 1 es el que se encarga de los usuarios y las Bases de Datos. Es el que permite configurar a que Bases de Datos se pueden conectar. Utilizando las opciones del fichero de configuración pg_ident.conf.

El nivel 2 es el que se encarga de las tablas. Es el que permite configurar a que tablas pueden acceder. Utilizando los comandos GRANT para dar permisos y REVOKE para eliminar los permisos. (12)

3.3 Validación funcional.

Para garantizar que la base de datos cumpla con los requisitos de los usuarios y opere sin problemas se necesitan buenos procedimientos de pruebas. Las pruebas de validación normalmente se centran en las operaciones siguientes:

- La carga de la base de datos se realiza sin violar la integridad de los datos.
- La correcta interfaz de la aplicación con la base de datos.
- El rendimiento del sistema satisface las necesidades para las que se adquirió el SGBD.

Para la realización de pruebas de un llenado voluminoso e inteligente de la base de datos, sin violar la integridad de la misma, se empleó PostgreSQL Data Generator. Esta herramienta es muy poderosa en la generación de datos de pruebas para bases de datos construidas en PostgreSQL. Permite seleccionar tablas y columnas para generar datos, definir rangos de valores, generar columnas de caracteres por máscara. También brinda la posibilidad de generar los datos a través de consultas SQL o de escogerlos por medio de listas de valores. Una de las ventajas que tiene es que genera los datos que provienen de otras tablas para evitar errores de integridad referencial. El empleo de esta herramienta posibilitó chequear la integridad de la base de datos, obteniéndose muy buenos resultados.

Las tablas fueron llenadas con un rango de datos equivalente al que tendrá la base de datos en un año aproximadamente.

Otra forma de probar que la base de datos funciona correctamente es a través de consultas. De acuerdo a las operaciones que se espera se realicen una vez implantada la base de datos se ejecutaron, entre otras, las siguientes consultas:

- Obtener el idintegrante, idcategoria, idclasificacion, idlocal, idpc, idrol, y tarea donde el idintegrante este entre 1 y 190, ordenado por el idintegrante.

SELECT dat_integrante.idintegrante, dat_integrante.idcategoria, dat_integrante.idclasificacion, dat_integrante.idlocal, dat_integrante.idpc, dat_integrante.idrol, dat_tarea.tarea FROM dat_integrante, dat_tarea WHERE (dat_integrante.idintegrante = dat_tarea.idintegrante) AND (dat_integrante.idintegrante BETWEEN 1 AND 190) ORDER BY dat_integrante.idintegrante;

El resultado obtenido a partir de esta consulta fue de 193 filas en un tiempo de 0,03 segundos.

- Obtener el numero identificador de los registros psp, la fecha, la hora de inicio de la tarea, la tarea y el numero identificador del integrante

SELECT dat_psp.idpsp, dat_psp.fecha, dat_psp.horainitarea, dat_tarea.tarea, dat_integrante.idintegrante FROM dat_psp, dat_tarea, dat_integrante WHERE (dat_psp.idtarea =dat_tarea.idtarea) AND (dat_tarea.idintegrante = dat_integrante.idintegrante) AND (dat_psp.idpsp > 50) ORDER BY dat_integrante.idintegrante;

Se obtuvo un resultado de 150 filas en un tiempo de 0,05 segundos.

Las pruebas realizadas a una base de datos nunca pueden tomarse como resultados reales, aunque dan un aproximado al mismo, dependiendo del grado de acercamiento que se utilice a los procesos de la vida real. Aún así, la implantación y utilización de una base de datos, está marcada por factores como, cantidad de usuarios a conectarse y el grado de concurrencia de las solicitudes hechas por los mismos, los cuales hay que tener en cuenta siempre. Además, es necesario velar el cumplimiento de todos los requerimientos no funcionales propuestos para la base de datos, ya que validan el funcionamiento correcto y máximo rendimiento de la misma. Aunque las pruebas realizadas brinden resultados que puedan parecer alentadores, no pueden ser motivo de confianza, todo lo contrario, la mejor prueba que se le puede realizar a cualquier resultado informático lo constituye la interacción directa del usuario, por eso existen los procesos de mantenimiento de software y la realización de versiones del mismo que buscan mejorar o resolver, posibles errores que se detecten durante su vida útil.

3.4 Valoración de resultados.

Luego de concluir el diseño y la implementación de la base de datos de la aplicación informática "Sistema para el Control de los Recursos del Proyecto" se obtuvieron los resultados siguientes:

- La base de datos cumple con los requisitos de los usuarios.
- Las relaciones entre las tablas fueron construidas adecuadamente.
- Correcta normalización de la base de datos.
- Las restricciones de integridad garantizan que los datos introducidos o modificados sean los correctos.
- La redundancia en la información se ha reducido en gran medida.

3.5 Conclusiones.

Con la validación teórica del diseño realizado se chequearon parámetros muy importantes a tener en cuenta en el correcto funcionamiento de la base de datos. Se definieron restricciones de integridad para garantizar la validez de los datos introducidos o modificados en la misma. Fueron tomados en cuenta aspectos para garantizar la seguridad de los datos en el sistema. Por otra parte, las pruebas realizadas a la base de datos permitieron chequear la integridad y seguridad de los datos y el correcto funcionamiento de la misma.

CONCLUSIONES

Las bases de datos se han venido utilizando desde hace ya varios años. Su marcada importancia hace que muchas empresas las incorporen en sus organizaciones para el procesamiento de la información. Los sistemas administradores de bases de datos, que facilitan la manipulación y actualización de los datos de una aplicación, han aumentado su potencialidad con el transcurso del tiempo, siendo hoy en día los sistemas de bases de datos relacionales los más utilizados. Otros modelos de base de datos que también se van desarrollando son los orientados a objetos, los distribuidos, los multidimensionales y los objetos relacionales, los últimos emplean la estructura relacional pero incorporan conceptos de la orientación a objetos.

Un ejemplo de sistema gestor de base de datos objeto relacional es PostgreSQL, en el cual reside la base de datos confeccionada para el sistema informático “Sistema para el Control de los Recursos del Proyecto”. Este sistema ha posibilitado garantizar la integridad de la base de datos por medio de restricciones que chequean la validez de los datos y definir aspectos de seguridad de la información.

Se llegó a la conclusión de que la nueva base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” cumple con el proceso de normalización hasta la tercera forma normal, posibilitando que en las tablas se encuentren datos que realmente pertenecen a ellas y que se disminuya la redundancia en la información. Se han tomado en cuenta, además, aspectos que permitan la reducción del tiempo de ejecución de las consultas de acuerdo a las necesidades de los usuarios.

Las pruebas realizadas a la base de datos han demostrado que cumple con los requisitos de los clientes y que la misma está acta para su funcionamiento, de acuerdo a su grado de integridad y seguridad.

RECOMENDACIONES

1. Que se ponga en práctica el diseño de esta base de datos.
2. Que se realice para futuras bases de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto” una tabla auditoria en la cual se controlen los eventos realizados por cada usuario que acceda a la base de datos.
3. Que se tenga en cuenta el diseño propuesto en la tesis para nuevas versiones de la base de datos de la aplicación informática “Sistema para el Control de los Recursos del Proyecto”.

REFERENCIAS BIBLIOGRÁFICAS

1. **Hansen, James W.** Diseño y Administracion de base de datos.
2. **Date, J.C.** Introducción a los Sistemas de Base de Datos. La Habana : s.n., 2003.
3. **Anónimo.** mailxmail.com. mailxmail.com. [En línea] 13 de 04 de 2005. [Citado el: 11 de 01 de 08.] http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm..
4. **Gascón, Manuel de la Herrán.** Administración y Optimización de Bases de Datos Oracle. Administración y Optimización de Bases de Datos Oracle. [En línea] 1999-2004. [Citado el: 18 de 01 de 2008.] http://www.redcientifica.com/oracle/c0001p0005.html..
5. **Anónimo.** MySQL 5.0 Reference Manua. MySQL 5.0 Reference Manua. [En línea] 1997-2007. [Citado el: 16 de 01 de 2008.] http://dev.mysql.com/doc/refman/5.0/es/index.html..
6. **Mota, S.A.** Entonación de Bases de Datos. abril-junio de 2005.
7. **Anónimo.** Todo-Linux.com. Todo-Linux.com. [En línea] 2007. [Citado el: 1 de 02 de 2008.] http://www.todo-linux.com/modules.php?name=News&file=article&sid=3421..
8. **Sarmiento Cutiño, Alieski.** LIMS DE CALIDAD DEL CENTRO DE INGENIERÍA GENÉTICA Y BIOTECNOLOGÍA. La Habana. : s.n., 2006.
9. **Quintela, J.A.R.** Puesta en marcha de un servidor de bases de datos utilizando PostgreSQL8. mayo de 2001.
10. **Riordan, Rebeca M.** Diseño de bases de datos relacionales con Acces y SqlServer.
11. **Garcia, Rosa M.** Sistema de Base de Datos. 2005.
12. **Quintela, J.A.R.** Puesta en marcha de un servidor de bases de datos utilizando PostgreSQL8. mayo, 2001.

BIBLIOGRAFÍA

1. **Hansen, James W.** Diseño y Administracion de base de datos.
2. **Date, J.C.** Introducción a los Sistemas de Base de Datos. La Habana : s.n., 2003.
3. **Anónimo.** mailxmail.com. mailxmail.com. [En línea] 13 de 04 de 2005. [Citado el: 11 de 01 de 08.] http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm..
4. **Gascón, Manuel de la Herrán.** Administración y Optimización de Bases de Datos Oracle. Administración y Optimización de Bases de Datos Oracle. [En línea] 1999-2004. [Citado el: 18 de 01 de 2008.] http://www.redcientifica.com/oracle/c0001p0005.html..
5. **Anónimo.** MySQL 5.0 Reference Manua. MySQL 5.0 Reference Manua. [En línea] 1997-2007. [Citado el: 16 de 01 de 2008.] http://dev.mysql.com/doc/refman/5.0/es/index.html..
6. **Mota, S.A.** Entonación de Bases de Datos. abril-junio de 2005.
7. **Anónimo.** Todo-Linux.com. Todo-Linux.com. [En línea] 2007. [Citado el: 1 de 02 de 2008.] http://www.todo-linux.com/modules.php?name=News&file=article&sid=3421..
8. **Sarmiento Cutiño, Alieski.** LIMS DE CALIDAD DEL CENTRO DE INGENIERÍA GENÉTICA Y BIOTECNOLOGÍA. La Habana. : s.n., 2006.
9. **Quintela, J.A.R.** Puesta en marcha de un servidor de bases de datos utilizando PostgreSQL8. mayo de 2001.
10. **Riordan, Rebeca M.** Diseño de bases de datos relacionales con Acces y SqlServer.
11. **Garcia, Rosa M.** Sistema de Base de Datos. 2005.
12. **Quintela, J.A.R.** Puesta en marcha de un servidor de bases de datos utilizando PostgreSQL8. mayo, 2001.

ANEXOS

Anexo 1. Descripción de las clases del diagrama de clases persistentes de la base de datos actual.

Nombre: nom_turno	
Tipo de clase: entidad	
Atributo	**Tipo**
idturno	int2
turno	text
actual	bool

Nombre: nom_dia	
Tipo de clase: entidad	
Atributo	**Tipo**
iddia	int2
dia	text
actual	bool

Nombre: nom_tarea	
Tipo de clase: entidad	
Atributo	**Tipo**
idtarea	int2
tarea	text
actual	bool

Nombre: nom_proyecto	
Tipo de clase: entidad	
Atributo	**Tipo**
idproyecto	int2
proyecto	varchar(0)
actual	bool
descripcion	varchar(0)

Nombre: nom_facultad	
Tipo de clase: entidad	
Atributo	**Tipo**
idfacultad	int2
facultad	text
actual	bool

Nombre: nom_nivel	
Tipo de clase: entidad	
Atributo	**Tipo**
idnivel	int2
nivel	text
actual	bool

Nombre: nom_sexo	
Tipo de clase: entidad	
Atributo	**Tipo**
idsexo	int2
sexo	varchar(9)
actual	bool

Nombre: nom_rol	
Tipo de clase: entidad	
Atributo	**Tipo**
idrol	int2
rol	text
actual	bool

Nombre: nom_clasificacion	
Tipo de clase: entidad	
Atributo	**Tipo**
idclasificacion	int2

clasificacion	varchar(20)
actual	bool

Nombre: nom_categoria	
Tipo de clase: entidad	
Atributo	**Tipo**
idcategoria	int2
categoria	varchar(0)
actual	bool

Nombre: nom_local	
Tipo de clase: entidad	
Atributo	**Tipo**
idlocal	int4
local	varchar(30)
mascara_ip	varchar(11)
actual	bool
idmodulo	int4

Nombre: nom_medio	
Tipo de clase: entidad	
Atributo	**Tipo**
idmedio	int2
medio	varchar(0)
serie	bool
actual	bool

Nombre: nom_mes	
Tipo de clase: entidad	
Atributo	**Tipo**
idmes	int2
mes	varchar(8)
activo	bool

Nombre: nom_semana

Tipo de clase: entidad	
Atributo	**Tipo**
idsemana	int2
semana	varchar(8)
activo	bool

Nombre: nom_interrupcion	
Tipo de clase: entidad	
Atributo	**Tipo**
idinterrupcion	int2
interrupcion	varchar(0)
actual	bool

Nombre: nom_supervisor	
Tipo de clase: entidad	
Atributo	**Tipo**
idsupervisor	int2
idintegrante	int4
funcion	bool

Nombre: nom_web	
Tipo de clase: entidad	
Atributo	**Tipo**
idmodulo	int2
dir_web	text
usuario	text
contrasena	text
actual	bool

Nombre: nom_modulo	
Tipo de clase: entidad	
Atributo	**Tipo**

idmodulo	int2
modulo	text
sitio	text
actual	bool
idproyecto	int2
descripción	varchar(0)

Nombre: nom_bd	
Tipo de clase: entidad	
Atributo	**Tipo**
idmodulo	int2
dir_bd	text
esquema	text
usuario	text
contrasena	text
actual	bool

Nombre: nom_documento	
Tipo de clase: entidad	
Atributo	**Tipo**
idmodulo	int2
dir_docum	text
usuario	text
contrasena	text
actual	bool

Nombre: nom_telefono	
Tipo de clase: entidad	
Atributo	**Tipo**
idintegrante	int2
telef_part	numeric
telef_trab	numeric
actual	bool

Nombre: nom_pc	
Tipo de clase: entidad	
Atributo	**Tipo**
idpc	int2
nombre_pc	text
ip_pc	text
actual	bool
idmodulo	int2

Anexo 2. Descripción de las tablas del diagrama entidad relación del diseño de base de datos propuesto.

Nombre: nom_proyecto		
Descripción: Almacena datos de los proyectos.		
Atributo	**Tipo**	**Descripción**
idproyecto	int2	Número identificador del proyecto.
proyecto	text	Nombre del proyecto
descripcion	text	Descripción del proyecto.
fechaini	date	Fecha de alta del proyecto.
fechafin	date	Fecha de baja del proyecto.

Nombre: nom_local		
Descripción: Almacena datos de los locales.		
Atributo	**Tipo**	**Descripción**
idlocal	int2	Número identificador del local.
local	text	Nombre del local
mascaraip	varchar(11)	Mascara ip del local.
fechaini	date	Fecha en que se crea el local.
fechafin	date	Fecha en que se elimina el local.
idmodulo	text	Número identificador del módulo.

Nombre: nom_rol		
Descripción: Almacena datos de los roles.		
Atributo	**Tipo**	**Descripción**
idrol	int2	Número identificador del rol.
rol	text	Nombre del rol (Ej:programador,analista).
fechaini	date	Fecha en que se crea el rol.
fechafin	date	Fecha en que se elimina el rol.

Nombre: nom_pc		
Descripción: Almacena datos de las pcs.		
Atributo	**Tipo**	**Descripción**
idpc	int2	Número identificador de la pc.
nombrepc	text	Nombre de la pc.
ippc	text	Numero ip de la pc.
fechaini	date	Fecha de alta de la pc.
fechafin	date	Fecha de baja de la pc.

Nombre: nom_categoria		
Descripción: Almacena datos de las categorías. Nivel de acceso a la aplicación		
Atributo	**Tipo**	**Descripción**
idcategoria	int2	Número identificador de la categoría.
categoria	text	Nombre de la categoría (Ej: administrador,usuario).
fechaini	date	Fecha de alta de la categoría.
fechafin	date	Fecha de baja de la categoría.

Nombre: nom_clasificacion		
Descripción: Almacena datos de las clasificaciones.		
Atributo	**Tipo**	**Descripción**
idclasificacion	int2	Número identificador de la clasificación.
clasificacion	text	Nombre de la clasificación. (Ej.: profesor, militar, estudiante)
fechaini	date	Fecha de alta de la clasificación.
fechafin	date	Fecha de baja de la clasificación.

Nombre: nom_dia		
Descripción: Almacena datos de los dias.		
Atributo	**Tipo**	**Descripción**
iddia	int2	Número identificador del día.
dia	text	Nombre del día.
fechaini	date	Fecha de alta del día.
fechafin	date	Fecha de baja del día.

Nombre: nom_turno		
Descripción: Almacena datos de los turnos.		
Atributo	**Tipo**	**Descripción**
idturno	int2	Número identificador del turno.
turno	text	Nombre del turno.
fechaini	date	Fecha en que se crea un turno.
fechafin	date	Fecha en que se elimina un turno.

Nombre: nom_funcion

Descripción: Almacena los nombres de las funciones.		
Atributo	**Tipo**	**Descripción**
idfuncion	varchar(15)	Nombre e identificador de las funciones.
fechaini	date	Fecha de alta de la función.
fechafin	date	Fecha de baja de la función.

Nombre: nom_tipomedio		
Descripción: Almacena datos de los tipos de medios.		
Atributo	**Tipo**	**Descripción**
idtipomedio	int2	Número identificador del tipo del medio.
tipomedio	text	Nombre del tipo del medio.
fechaini	date	Fecha de alta del tipo de medio.
fechafin	date	Fecha de baja del tipo de medio.
idlocal	int2	Numero identificador del local a donde pertenezca el medio.

Nombre: nom_telefono		
Descripción: Almacena los números de los teléfonos.		
Atributo	**Tipo**	**Descripción**
idtelefono	int2	Número identificador del teléfono.
fechaini	date	Fecha de alta del teléfono.
fechafin	date	Fecha de baja del teléfono.

Nombre: nom_modulo		
Descripción: Almacena los datos de los módulos. .		
Atributo	**Tipo**	**Descripción**
idmodulo	text	Número identificador del modulo.
descripción	text	Descripción de el modulo.
sitio	text	Sitio donde esta publicado el modulo.
dircodfuente	varchar(255)	Dirección del código fuente del modulo.
dirdoc	varchar(255)	Dirección donde se encuentra la documentación del modulo.
dirbd	varchar(255)	Dirección donde se encuentra la base de datos del modulo.
esquema	varchar(255)	Nombre del esquema de la base de datos que se esta utilizando.
usuarioacceso	varchar(25)	Usuario para acceder a la documentación, al código fuente y a base de datos.
contrasenaacceso	varchar(15)	Contraseña para acceder a la documentación, al código fuente y a base de datos.
fechaini	date	Fecha de alta del módulo.
fechafin	date	Fecha de baja del módulo.
idproyecto	int2	Numero identificador del proyecto al que pertenece el modulo.

Nombre: nom_interupcion
Descripción: Almacena los datos de las interrupciones.

Atributo	Tipo	Descripción
idinterrupcion	text	Nombre e identificador de la interrupción.
fechaini	date	Fecha de alta de la interrupción.
fechafin	date	Fecha de baja de la interrupción.
idpsp	int2	Número identificador del registro psp.

GLOSARIO

Microsoft (acrónico de Microcomputer Software)

Es una empresa de Estados Unidos, fundada por Bill Gates y Paul Allen. Dueña y productora de los sistemas operativos: Microsoft DOS y Microsoft Windows, que se utilizan en la mayoría de las computadoras del planeta.

Sistema operativo (SO)

Es un conjunto de programas destinados a permitir la comunicación del usuario con un computador y gestionar sus recursos de una forma eficaz.

Archivos secuenciales

Un archivo organizado secuencialmente es un conjunto de registros lógicamente relacionados cuya secuencia de acceso está determinada por su ordenamiento. Los registros deben ser grabados consecutivamente cuando el archivo es creado, y deben ser leídos de la misma manera cuando es usado posteriormente como entrada.

Archivos indexados

Es la aplicación de incluir índices en el almacenamiento de los archivos; de esta forma será más fácil buscar algún registro sin necesidad de ver todo el archivo.

Un índice en un archivo consiste en un listado de los valores del campo clave que ocurren en el archivo, junto con la posición de registro correspondiente en el almacenamiento masivo.

RAID (Redundant Array Of Independent/Inexpensive Disks).

Es un término que hace referencia a un conjunto de discos redundantes e independientes. Se utiliza para mejorar el rendimiento, la tolerancia, fallos y errores en los discos, así como también mejora la integridad de los datos.

ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad)

- Atomicidad (Indivisible): es la propiedad que asegura que la operación se ha realizado o no, y por lo tanto ante un fallo del sistema no puede quedar a medias.
- Consistencia: es la propiedad que asegura que sólo se empieza aquello que se puede acabar. Por lo tanto se ejecutan aquellas operaciones que no van a romper la reglas y directrices de integridad de la Base de Datos.

- Aislamiento: es la propiedad que asegura que una operación no puede afectar a otras. Esto asegura que dos transacciones sobre la misma información nunca generarán ningún tipo de error.
- Durabilidad: es la propiedad que asegura que una vez realizada la operación, ésta persistirá y no se podrá deshacer aunque falle el sistema.

SQL (Structured Query Language)

Es un lenguaje de acceso a las Bases de Datos, permite especificar todas las operaciones sobre la base de datos como por ejemplo: Inserción, Borrado, Actualización. Utiliza características de álgebra y cálculo relacional permitiendo de esta forma realizar consultas a la base de datos de forma sencilla.

DDL (Lenguaje de definición de datos)

Es el lenguaje proporcionado por el Sistema Gestor de Base de Datos para llevar a cabo tareas de definición de estructuras de la Base de Datos.

DML (Lenguaje de manipulación de datos)

Permite a los usuarios llevar a cabo consultas y manipulación de los datos.

IBM (International Business Machines Corporation)

Conocida coloquialmente como el Gigante Azul, es una empresa que fabrica y comercializa hardware, software y servicios relacionados con la informática.

Archivos planos

Un archivo plano es un archivo secuencial puro, o txt.

Conjunto de caracteres ANSI organizados de tal forma que permiten ser almacenados y recuperados. Se utilizan para la transferencia de datos.

Disparadores (trigger)

Un disparador es una regla en la que se especifica una acción que el SGBD debe ejecutar como respuesta a la ocurrencia de un evento, siempre que se cumpla la condición especificada.

Hipercubo

Un hipercubo se define como un cubo desfasado en el tiempo, es decir, cada instante de tiempo por el cual se movió pero todos ellos juntos.

Vistas

Una vista es el resultado dinámico de una o varias operaciones relacionales realizadas sobre las relaciones base. El contenido de una vista está definido como una consulta sobre una o varias relaciones base.

Catálogo (Diccionario de datos)

Es una estructura propia de la base de datos en la que se definen los elementos que forman parte de la misma.

Artefacto

Un artefacto es una información que es utilizada o producida mediante un proceso de desarrollo de software. Pueden ser artefactos un modelo, una descripción o un software. Los artefactos de UML se especifican en forma de diagramas, éstos, junto con la documentación sobre el sistema constituyen los artefactos principales que el modelador puede observar.

Herramienta CASE (Computer Aided Software Engineering, Ingeniería de Software Asistida por Ordenador)

Las Herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.

Multiplataforma

Es un término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de software, que puedan funcionar en diversas plataformas.

Una plataforma es una combinación de hardware y software usada para ejecutar aplicaciones; en su forma más simple consiste únicamente de un sistema operativo, una arquitectura, o una combinación de ambos.

Transacciones

Una transacción es un conjunto de procesos que se ejecutan uno después del otro. Si algún subproceso falla, lo realizado anteriormente debe reversarse para que los datos no se alteren, a este comportamiento se lo denomina todo o nada.

Servidores

Un servidor, en informática o computación, es el ordenador en el que se ejecuta un programa que realiza alguna tarea en beneficio de otras aplicaciones llamadas clientes.

Protocolo

Un protocolo de comunicación es la manera de comunicarse que tiene una computadora con otra, cuando se están transmitiendo datos entre sí.

1