Trabajo de Diploma

Universidad de las Ciencias Informáticas

Facultad 9

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

Título: Diseño de Bases de Datos para la Empresa de Gas Manufacturado.

Autor: Raúl Mató Rodríguez.

Tutor: Ing. Alfonso Chaveco Laurencio.

Ciudad de la Habana, Mayo 2009.

"Año del 50 Aniversario del Triunfo de la Revolución".

"El único autógrafo digno de un hombre es el que deja escrito con sus obras."

Dedicatoria

Dedico el resultado de la presente investigación, a mis padres por darme su apoyo en todo momento, a mi abuelo quien aunque no está hoy aquí nunca lo olvidare, a mi abuela a quien debo todo lo que soy, a Karem que siempre confió en mí, a mis familiares, amigos y demás personas que me apoyaron durante el transcurso de la carrera.

Agradecimientos

Doy antes que nada gracias a Dios por permitirme llegar hasta aquí.
Agradezco a las personas que han contribuido a la elaboración del presente trabajo, a mi tutor Alfonso Chaveco, a los compañeros del tribunal y a la oponente por su crítica constructiva.
A mis amigos que durante estos cinco años han estado compartiendo conmigo a Arianna, Adalberto, Bárbaro, Reinel, Heikel, a todos los que de una forma o de otra hemos tenido algún momento junto durante los estudios.
Quiero agradecer especialmente a mis padres, a mi abuela quien ha sido mi segunda madre, a mis tías, en fin a toda mi familia, a Blancanieves y familia que durante estos cinco años han sido de gran apoyo, a más que mi amigo mi hermano Karem quien siempre ha depositado confianza en mí, y a todos aquellos que a través de los años han compartido algún momento de su vida junto a mí.

Datos de Contacto

Tutor: Alfonso Chaveco Laurencio (achaveco@uci.cu): Graduado de Ing. en Ciencias Informáticas en el año 2007, posee categoría docente de Profesor Instructor Recién Graduado. Cuenta con dos años de experiencia, ha impartido asignaturas como, Ing. de Software I y II, Gestión de Software, Programación III y Base de Datos, actualmente es uno de los líderes del proyecto ManuGas perteneciente al Polo PetroSoft.

.

Opinión del Tutor

Resumen

Con la tendencia actual a la informatización, las empresas se han dado a la tarea de automatizar todos los procesos posibles dentro de su negocio, esto facilita el trabajo y brinda sin duda una mayor y mejor organización además que le imprime calidad y le aporta prestigio a la institución.

La empresa de Gas Manufacturado es la encargada del control de este servicio en Ciudad de la Habana, dos de sus principales actividades, son la Facturación y el Cobro mensual que se le tasa a los clientes, y aunque anteriormente han intentado informatizar estos procesos, no cuentan con un software apropiado para agilizarlos.

El presente trabajo de diploma se centra en el diseño de una base de datos que almacene toda la información generada durante los procesos antes mencionados, sin redundancia y que permita que la información esté íntegra, disponible y accesible al personal autorizado a interactuar con la misma.

El desarrollo de este trabajo fue guiado por las actividades establecidas, y con la utilización de las herramientas propuestas, como gestor de base de datos, herramientas para el diseño y para las pruebas de volumen sobre la base de datos diseñada a partir de estudios previos sobre la situación y estado actual de estas herramientas.

Índice

Índice de Tablas.

Introducción

Con el desarrollo de las tecnologías la vida del hombre ha tomado una forma diferente, estas le han ayudado en sus quehaceres diarios imprimiéndoles mayor calidad a los mismos, la computadora se ha convertido en una poderosa herramienta y casi indispensable para numerosos actividades que se han logrado automatizar, y gracias a estas es posible ganar en calidad y tiempo en diferentes servicios en las que se utilizan.

Cuba hace un gran esfuerzo por dar pasos de avance en aras de la informatización de la sociedad y la automatización de las actividades industriales, para lograr esto se preparan profesionales a lo largo y ancho del país en esta rama, un ejemplo de estos pasos de avance es la Universidad de las Ciencias Informáticas (UCI), surgida al calor de la batalla de ideas. En dicha instalación se preparan profesionales, los cuales tienen el compromiso de llevar a cabo las tareas que se necesiten encaminadas a las actividades antes mencionadas en las cuales se encuentran envueltas el país.

La Empresa de Gas Manufacturado que presta el servicio de gas manufacturado más conocido como gas de la calle (actualmente con este servicio cuentan algunos municipios en Ciudad de La Habana), se ha dado a la tarea de informatizar sus procesos principales, fundamentalmente lo referente al cobro de dicho servicio. La estructura de dicha empresa se constituye de la siguiente manera, una Casa Comercial en cada municipio y una Casa Matriz Central a la cual tributan las Casas Comerciales Municipales; actualmente la empresa se apoya en una aplicación llamada Sistema Metrado (Sistmet) para la realización del cobro a los clientes, dicha aplicación fue realizada en el leguaje de programación Visual Basic, y es muy difícil de mantener, debido a que no fue realizada por un personal capacitado para esto y el código está muy desordenado en cuanto a su estructura, la aplicación presenta incongruencias en un sin número de actividades, el almacenamiento en la base de datos no es óptimo, además que la misma no presenta ninguna organización, no se encuentra normalizada, teniendo una redundancia de datos muy notable, haciendo los procesos de búsquedas muy tediosos y costosos en cuanto a recursos de la computadora, como resultado de lo antes planteado aparejado a que todavía quedan muchas actividades que se realizan manualmente, los procesos de facturación y cobro se demoran varios días.

Actualmente la institución desea aplicar una tarifa de Cobro Escalonado por la cual se le cobraría al cliente según su consumo, no como se hace hoy en día con una tarifa fija, esto revierte pérdidas a la empresa, con la aplicación actual resulta imposible migrar al nuevo tipo de cobro.

Debido a lo antes planteado la Empresa de Gas Manufacturado hizo un pedido a la UCI sobre un software para la gestión de estos procesos fundamentales en la empresa, con el fin de darle respuesta se creó un proyecto productivo en la Facultad 9 en el polo de PETROSOFT llamado ManuGas, el mismo se dio a la tarea de implementar un sistema para dicha empresa.

Se identificó como **problema resolver** el almacenamiento de forma única y segura los datos de la Empresa de Gas Manufacturado que intervienen directamente en las actividades relacionadas con la Facturación y el Cobro del servicio de gas a los clientes.

El **objetivo general** de este trabajo es Diseñar la Base de Datos para el nuevo sistema de Facturación y Cobro a implantar en la Empresa de Gas Manufacturado.

El **objeto de estudio** de la investigación es el Proceso de Gestión de Información para los procesos de Facturación y Cobro en la Empresa de Gas Manufacturado.

El **campo de acción** es el diseño de una Base de Datos para preservar la información referente al Sistema de Facturación y Cobro de la Empresa de Gas Manufacturado

La **idea a defender** es el diseño de una base de datos que almacene de manera eficiente y segura cada uno de los datos que intervienen en los procesos de Facturación y Cobro de la Empresa de Gas Manufacturado la cual servirá de soporte en la automatización de los procesos que se desea lograr por el Sistema de Facturación y Cobro de la Empresa de Gas Manufacturado.

Para lograr el objetivo general de la investigación es necesario realizar las siguientes **tareas de investigación**:

1. Investigar las tendencias actuales en el desarrollo de Base de Datos.
    a. Valorar las técnicas de réplica, rendimiento, optimización.
    b. Valorar las técnicas de Base de Datos

2. Revisar la Base de Datos del Actual Sistema Informático (SISTMET).
3. Revisar los diagramas de clases del modelo de objetos obtenidos durante la modelación del Negocio.
4. Analizar la manera en que el Framework Symfony para PHP lleva a cabo la manipulación de los datos de las aplicaciones.
5. Identificar las clases persistentes necesarias para el desarrollo de la solución propuesta en el Sistema de Facturación y Cobro de la Empresa de Gas Manufacturado.
6. Realizar el Modelo de Entidad Relación de la Base de Datos.
7. Realizar el Modelo Físico de la Base de Datos.
8. Obtener de la base de datos del Sistema de Facturación y Cobro de la Empresa de Gas Manufacturado.
    a. Realizar del traspaso de información de la base de datos actual para la nueva base de datos.
9. Revisar y selección de las herramientas para las pruebas de volumen.

Con la realización del sistema se espera una mejor organización en el proceso de cobro de la Empresa de Gas Manufacturado, así como una mayor rapidez en el proceso de facturación, ahora con la realización de este trabajo se obtendrá una Base de Datos concisa y normalizada, la cual aporte una eficacia significativa en los procesos en que se encuentre involucrada.

El siguiente trabajo está estructurado en tres capítulos, en **el capítulo primero** se expone toda la fundamentación teórica de la investigación, en el se explica lo referente a la tecnología de Bases de Datos, su evolución y estado actual así como sus usos en el mundo moderno, en **el capítulo segundo** se refleja la esencia del trabajo, la descripción de la solución propuesta en el mismo aparece una explicación detallada de aspectos como: la arquitectura del software, la integración con el framework en cuestión , la descripción de las entidades y los modelos lógico y físico, **el capítulo tercero** muestra la validación de todo el diseño realizado anteriormente, tocando aspectos como la normalización de base de datos, la

integridad de la misma, su seguridad así como también lo referente a las pruebas realizada a la base de datos.

CAPÍTULO 1

FUNDAMENTACIÓN TEÓRICA

1.1. Introducción.

A lo largo de este capítulo se exponen las principales tendencias actuales de Base de Datos y los programas para la administración de las mismas, los Sistemas Gestores de Base de Datos y una pequeña historia de la evolución de estos sistemas, otro aspecto tratado es la descripción de las herramientas utilizadas para llevar a cabo el presente trabajo, y la justificación de por qué se escogieron las mismas, con el objetivo de brindar información contundente y convincente del uso de las herramientas seleccionadas .

1.2. Definición de Base de Datos (BD) y Sistema Gestor de Base de Datos (SGBD).

Una Base de Datos (BD) es un conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, que una BD puede considerarse una colección de datos variables en el tiempo, un sistema de archivos electrónico. (1)

Un Sistema Gestor de Base de Datos es el software que permite la creación, la actualización, el procesamiento y la consulta de la información almacenada en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez, de forma segura y eficiente.(1)

1.3. Breve historia de las Bases de Datos.

La necesidad de guardar la información para su próxima consulta o utilización ha hecho que el hombre desde tiempo atrás se halla dado a la tarea de buscar maneras para esto, utilizando diferentes medios para esto , desde un trozo de roca, una tablilla de arcilla, un hoja papel y otros. Con el desarrollo de las tecnologías esto por supuesto no quedó atrás y se buscó la forma de llevarlo a la par del avance tecnológico, 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.

Se dice que los sistemas de bases de datos tienen sus raíces en el proyecto estadounidense Apolo de mandar al hombre a la luna, en los años sesenta. La primera empresa encargada del proyecto, NAA (North American Aviation), desarrolló un *software* denominado GUAM (General Update Access Method) que estaba basado en el concepto de que varias piezas pequeñas se unen para formar una pieza más grande, y así sucesivamente hasta que el producto final está ensamblado. Esta estructura, que tiene la forma de un árbol, es lo que se denomina una estructura jerárquica.

A mitad de los sesenta, se desarrolló IDS (Integrated Data Store), de General Electric. Este trabajo fue dirigido por uno de los pioneros en los sistemas de bases de datos, Charles Bachmann. IDS era un nuevo tipo de sistema de bases de datos conocido como sistema de red, que produjo un gran efecto sobre los sistemas de información de aquella generación. El sistema de red se desarrolló, en parte, para satisfacer la necesidad de representar relaciones más complejas entre datos que las que se podían modelar con los sistemas jerárquicos, y, en parte, para imponer un estándar de bases de datos.

Los sistemas jerárquico y de red constituyen la primera generación de los SGBD.

En 1970 Codd, de los laboratorios de investigación de IBM, escribió un artículo presentando el modelo relacional. En este artículo, presentaba también los inconvenientes de los sistemas previos, el jerárquico y el de red. Entonces, se comenzaron a desarrollar muchos sistemas relacionales, apareciendo los primeros a finales de los setenta y principios de los ochenta.

Los SGBD relacionales constituyen la segunda generación de los SGBD. Sin embargo, el modelo relacional también tiene sus fallos, siendo uno de ellos su limitada capacidad al modelar los datos. Se ha hecho mucha investigación desde entonces tratando de resolver este problema. En 1976, Chen presentó el modelo entidad-relación, que es la técnica más utilizada en el diseño de bases de datos. En 1979, Codd intentó subsanar algunas de las deficiencias de su modelo relacional con una versión extendida denominada RM/T (1979) y más recientemente RM/V2 (1990).

La idea de una base de datos orientada a objetos se articuló por primera vez por Copeland y Maier en 1984, con el sistema prototipo GemStone. Uno de los sistemas más famosos de los finales de los ochenta y principios de los noventa fue el sistema ObjectStore, por Lamb en 1991. Al principio de los noventa, los

primeros Sistemas de Gestión de Bases de Datos Orientados a Objetos (SGBDOO, o simplemente, SGBDO) empezaron a aparecer en el mercado.

El modelo objeto-relacional es un desarrollo más reciente y parece haber tenido bastante efecto. No es una tecnología en sí, sino una aglutinación de los modelos relacional y orientado a objetos. De hecho, algunas extensiones objetuales a los sistemas relacionales se pueden datar en los principios de los ochenta [Zaniolo 1983].

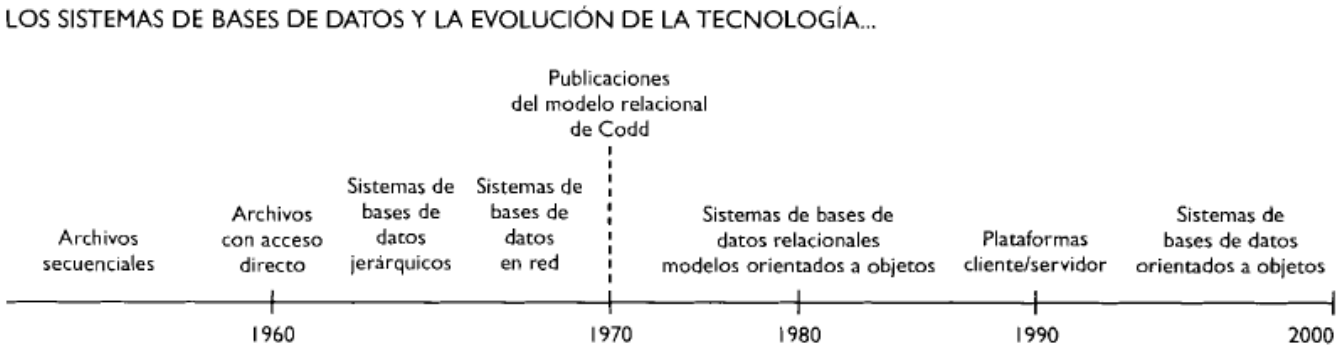


Fig. 1.1 Evolución de los Sistemas de Bases de Datos (SBD)[1]

1.4. Modelos de Bases de Datos.

Las Bases de Datos a través de su evolución han atravesado diferentes estructuras, desde los modelos de red hasta los modelos actuales orientados a objetos, cada de estos modelos tiene características peculiares que los identifican además de la fecha en que surgen, estos modelos se pueden agrupar en dos grandes grupos: los modelos tradicionales (Modelo jerárquico, modelo de red, modelo relacional) y los modelos avanzados (de los modelos orientados a objetos en adelante).

Un modelo de datos es por tanto una colección de conceptos bien definidos matemáticamente que ayudan a expresar las propiedades estáticas y dinámicas de una aplicación con un uso de datos intensivo.

[1]Tomado de Diseño y Administración de Bases Datos. Hansen, Gary W. y Hansen, James V.

1.4.1. Modelo Jerárquico.

El sistema jerárquico más comúnmente conocido es el sistema IMS de IBM. Esta base de datos tiene como objetivo establecer una jerarquía de fichas, de manera que cada ficha puede contener a su vez listas de otras fichas, y así sucesivamente. Una base de datos jerárquica está compuesta por una secuencia de bases de datos físicas, de manera que cada base de datos física se compone de todas las ocurrencias de un tipo de registro o ficha determinada. Una ocurrencia de registro es una jerarquía de ocurrencias de segmento. Cada ocurrencia de segmento está formada por un conjunto de ocurrencias o instancias de los campos que componen el segmento.

Las bases de datos jerárquicas son especialmente útiles en el caso de aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento.
Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos.

1.4.2. Modelo en Red.

Se considera el modelo de bases de datos en red como de una potencia intermedia entre el jerárquico y el relacional. Su estructura es parecida a la jerárquica aunque bastante más compleja, con lo que se consiguen evitar, al menos en parte, los problemas de aquél.

Los conceptos fundamentales que debe conocer el administrador para definir el esquema de una base de datos en red, son los siguientes:

- *Registro:* Viene a ser como cada una de las fichas almacenadas en un fichero convencional.
- *Campos o elementos de datos:* Son cada uno de los apartados de que se compone una ficha.
- *Conjunto:* Es el concepto que permite relacionar entre sí tipos de registro distintos.

La dificultad que significa 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.

1.4.3. Modelo Relacional.

Este modelo intenta representar la base de datos como un conjunto de tablas. Aunque las tablas son un concepto simple e intuitivo, existe una correspondencia directa entre el concepto informático de una tabla,

y el concepto matemático de relación, lo cual es una gran ventaja, pues permite efectuar formalizaciones de una forma estricta mediante las herramientas matemáticas asociadas, como pueda ser el álgebra relacional en el ámbito de las consultas.

En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos.
Éste es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente.

Los conceptos básicos del modelo relacional son:

- *Registro:* Es algo así como cada ficha de un fichero convencional.
-*Tabla:* Es un conjunto de fichas de un mismo tipo.

1.4.4. Modelo Orientado a Objetos.

Trata los problemas desde un punto de vista realista, y modelando cada uno de ellos como si se tratase de un conjunto de elementos u objetos que interrelacionan entre sí para solucionar el problema. Cada uno de estos objetos es un elemento. De esta manera, una modificación del estado de un objeto por parte de un usuario, desencadena una serie de acciones cuyo objetivo final es solucionar un problema al usuario.

Los principales conceptos que se definen en este modelo son:

-*Clase:* Cuando hay varios objetos semejantes, pueden agruparse en una clase. De hecho, todo objeto debe pertenecer a una clase, que define sus características generales

-*Estado:* Son las características propias de cada objeto. Siguiendo con el caso de los engranajes, su estado puede ser el número de dientes, el tamaño, etc. El estado se utiliza especialmente para guardar la situación del objeto que varía con el tiempo.

-*Encapsulación:* Cada objeto es consciente de sus propias características.

-*Mensaje:* Es cada uno de los estímulos que se envían a un objeto.

-*Herencia:* Para facilitar la programación, se puede establecer toda una jerarquía de tipos o clases. Con clases que tengan objetos similares, de esta forma se evita la declaración de datos de forma redundante.

-*Polimorfismo:* Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

1.4.5. Modelo Declarativo.

Este modelo usado para bases de conocimiento, que no son más que bases de datos con mecanismos de consulta en los que el trabajo de extracción de información a partir de los datos recae en realidad sobre el ordenador, en lugar de sobre el usuario. Estos mecanismos de consulta exigen que la información se halle distribuida de manera que haga eficiente las búsquedas de los datos, ya que normalmente las consultas de este tipo requieren acceder una y otra vez a los datos en busca de patrones que se adecúen a las características de los datos que ha solicitado el usuario.

Entre las bases de datos declarativas se puede citar fundamentalmente dos: las deductivas, y las funcionales. Ambas extienden paradigmas o métodos de programación (al igual que ocurre con la programación orientada a objetos) a las bases de datos, de manera que ambos, programa y base de datos puedan cooperar más eficientemente en la resolución del problema.

1.4.6. Modelo Bases de Datos Documentales.

Permiten la indexación a texto completo, y en líneas generales realizar búsquedas más potentes.

1.4.7. Modelo de Bases de Datos Distribuida.

La base de datos está almacenada en varias computadoras conectadas en red. Surgen debido a la existencia física de organismos descentralizados. Esto les da la capacidad de unir las bases de datos de cada localidad y acceder así a distintas universidades, sucursales de tiendas.

1.5. Tipos de Bases.

Las Bases de Datos se pueden clasificar de diversas maneras, pero para ello se debe escoger un criterio de clasificación. Ampliando más estos conceptos se tiene: que según la variabilidad de los datos almacenados podemos encontrar Bases de Datos *estáticas* o *dinámicas*, las primeras son bases de datos de solo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden

utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones por su parte en las Bases de Datos *dinámicas* la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta.

Otra tipo de clasificación de las Bases de Datos es según su contenido, se pueden encontrar dentro de esta clasificación: Bases de Datos *bibliográficas*, las cuales solo contienen un representante de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, etc. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque si no estaríamos en presencia de una base de datos a texto completo. Como su nombre lo indica, el contenido son cifras o números; Bases de Datos de *texto completo*, almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas y Bases Datos (bibliotecas) de *información Biológica*, como su nombre lo indica estas Bases de Datos gestionan información sobre la investigación de las ciencias de la vida o médicas.

1.6. Sistemas de Gestión de Bases de Datos.

Los gestores de Base de Datos, son programas para la creación y manipulación de las bases de datos los mismos deben permitir, definir una base de datos(especificar tipos, estructuras y restricciones de datos), construir la base de datos(guardar los datos en algún medio controlado por el mismo SGBD) y manipular la base de datos(realizar consultas, actualizarla, generar informes) hoy en día estos programa se desarrollan constantemente, y especializándose en los diferentes modelos de Bases de Datos.

1.6.1. Características y Objetivos de un Sistema Gestor de Base de Datos SGBD

- *Control de la redundancia:* La redundancia de datos tiene varios efectos negativos (duplicar el trabajo al actualizar, desperdicia espacio en disco, puede provocar inconsistencia de datos) aunque a veces es deseable por cuestiones de rendimiento.

- *Independencia:* La independencia de los datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.

- *Consistencia:* En aquellos casos en los que no se ha logrado eliminar la redundancia, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea. Por otra parte, la base de datos representa una realidad determinada que tiene determinadas condiciones, por ejemplo que los menores de edad no pueden tener licencia de conducir. El sistema no debería aceptar datos de un conductor menor de edad. En los SGBD existen herramientas que facilitan la programación de este tipo de condiciones.

- *Restricción de los accesos no autorizados:* cada usuario ha de tener permisos de acceso y autorización.

- *Cumplimiento de las restricciones de integridad:* el SGBD ha de ofrecer recursos para definir y garantizar el cumplimiento de las restricciones de integridad.

- *Facilidad de manipulación de la información:* El SGBD debe contar con la capacidad de una búsqueda rápida por diferentes criterios, permitir que los usuarios planteen sus demandas de una forma simple, aislándolo de las complejidades del tratamiento de los ficheros y del direccionado de los datos.

- *Respaldo:* Los SGBD deben proporcionar una forma eficiente de realizar copias de respaldo de la información almacenada en ellos, y de restaurar a partir de estas copias los datos que se hayan podido perder.

- *Tiempo de respuesta*: Lógicamente, es deseable minimizar el tiempo que el SGBD tarda en darnos la información solicitada y en almacenar los cambios realizados (2).

1.6.2. Arquitectura de los SGBD.

Los SGBD suelen tener una arquitectura en tres niveles, *Nivel Externo* (Vistas de usuarios individuales) *Nivel Conceptual o Lógico* Global (Vistas general) y *Nivel Interno* (Vistas lógicas y de almacenamiento).

1. *Nivel Interno:* tiene un esquema interno que describe la estructura física de almacenamiento de la BD. El esquema interno emplea un modelo físico de los datos y describe todos los detalles para su almacenamiento, así como los caminos de acceso a la BD.

2. *Nivel Conceptual:* tiene un esquema conceptual que describe la estructura de toda la BD para una comunidad de usuarios. Oculta detalles de la estructura física de almacenamiento y se concentra en describir entidades, tipos de datos, vínculos, operaciones de los usuarios y restricciones.

3. *Nivel Externo o Vistas:* incluye varios esquemas externos o vistas de usuarios. Cada uno describe la parte de la BD que interesa a un grupo de usuarios.

1.6.3. Lenguajes de los SGBD.

Otra característica de los SBGD son los lenguajes que soportan, estos lenguajes pueden ser:

• *Lenguaje de Definición de Datos* (DDL) permite definir ambos esquemas. El SGBD cuenta con un compilador de DDL cuya función es procesar enunciados escritos en el DDL para implementar las descripciones de los elementos de los esquemas y almacenar la descripción del esquema en el catálogo del SGBD.

• *Lenguaje de Definición de Almacenamiento* (SDL) permite especificar el esquema interno. Las correspondencias entre los dos esquemas se pueden especificar en cualquiera de los lenguajes.

• *Lenguaje de definición de Vistas* (VDL) permite especificar las vistas del usuario y su correspondencia con el esquema conceptual.

• *Lenguaje de Manipulación de Datos* (DML) una vez compilados los esquemas de la BD y que en esta se han introducido datos, los usuarios requerirán algún mecanismo para manipularla. Las operaciones más comunes son: obtención, inserción, eliminación y modificación de los datos.

1.6.4. Componentes de un SGBD.

- *Administrador de almacenamiento:* Controla el acceso a la información de la base de datos almacenada en el disco ya sean datos o metadatos. Se encarga de la gestión tanto de los buffer en la memoria principal, como de los archivos donde se almacena la base de datos.

- *Procesador de consultas:* Recibe las peticiones de consulta o actualización y encuentra la mejor manera de realizarlas (plan de ejecución). Emite órdenes al *Administrador de almacenamiento* que las ejecuta.

- *Gestor de transacciones:* Se encarga de conservar la integridad de la base de datos, una transacción es una colección de operaciones que realizan una única función lógica sobre la base de datos, asegura el estado correcto de la base de datos a pesar de fallos en el sistema, controla la interacción entre transacciones concurrentes.

- *Compilador DDL:* Procesa las definiciones de esquemas y almacena las descripciones en el catalogo.

- *Compilador de consultas:* Maneja las consultas de alto nivel que se introducen interactivamente, analiza la sintaxis y el contenido de las consultas y llama al *Procesador de consultas*.

- *Pre compilador:* Extrae órdenes de un programa de aplicación que contiene sentencias DML y las traduce en órdenes para el *Compilador de DML*.

- *Compilador de DML:* Cuando el programa de aplicación se traduce a código ejecutable este compilador incluye llamadas apropiadas al *Procesador de Consultas*.

1.5.5. Algunos Gestores de Bases de Datos.

En la actualidad existen un gran número de SGBD, con el auge de la programación orientada a objetos, los gestores orientados hacia este paradigma han tomado un camino revolucionario y han aparecido muchos nuevos, al igual los SGBD que funcionan sobre el modelo relacional tampoco se han quedado atrás y más aun cuando este es el modelo más usado actualmente, por esto es difícil decidir cuál es el mejor gestor de bases de datos, dentro de los gestores más usados tenemos:

Microsoft Access es un programa Sistema de gestión de base de datos relacional creado y modificado por Microsoft para uso personal de pequeñas organizaciones. Es un componente de la suite Microsoft Office aunque no se incluye en el paquete "básico". Una posibilidad adicional es la de crear ficheros con bases de datos que pueden ser consultados por otros programas.

Desventajas

- Este SGBD no es recomendable para bases de datos de gran calibre (en cuanto a volumen de datos o de usuarios).

- Entre sus mayores inconvenientes figuran que no es multiplataforma, pues está disponible para sistemas operativos de Microsoft nada más, Su uso es inadecuado para grandes proyectos de software que requieren tiempos de respuesta críticos o muchos accesos simultáneos a la base de datos. (5)

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional (ORDBMS) basado en el proyecto POSTGRES, de la universidad de Berkeley. El director de este proyecto es el profesor Michael Stonebraker, y fue patrocinado por Defense Advanced Research Projects Agency (DARPA), el Army Research Office (ARO), el National Science Foundation (NSF), y ESL, Inc. PostgreSQL es una derivación libre (OpenSource) de este proyecto, y utiliza el lenguaje SQL92/SQL9. Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluido, más tarde en otros sistemas de gestión comerciales. PostgreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. A pesar de esto, PostgreSQL no es un sistema de gestión de bases de datos puramente orientado a objetos. (3)

Historia

PostgreSQL (llamado también Postgres95) fue derivado del proyecto PostgreSQL. La implementación de PostgreSQL DBMS comenzó en 1986, y no hubo una versión operativa hasta 1987. La versión 1.0 fue liberada en Junio de 1989 a unos pocos usuarios, tras la cual se liberó la versión 2.0 en Junio de 1990 debido a unas críticas sobre el sistema de reglas, que obligó a su re implementación. La versión 3.0 apareció en el año 1991, e incluyó una serie de mejoras como una mayor eficiencia en el ejecutor de peticiones. El resto de versiones liberadas a partir de entonces, se centraron en la portabilidad del sistema. El proyecto se dio por finalizado en con la versión 4.2, debido al gran auge que estaba teniendo, lo cual causó la imposibilidad de mantenimiento por parte de los desarrolladores.

En 1994, Andrew Yu y Jolly Chen añadieron un intérprete de SQL a este gestor. Postgres95, como así se llamó fue liberado a Internet como un proyecto libre (OpenSource). Estaba escrito totalmente en C, y la primera versión fue un 25% más pequeña que PostgreSQL, y entre un 30 y un 50% más rápida. A parte de la corrección de algunos bugs, se mejoró el motor interno, se añadió un nuevo programa monitor, y se compiló usando la utilidad GNU Make y el compilador gcc sin necesidad de parchearlo (como había hecho falta en versiones anteriores).

En 1996, los desarrolladores decidieron cambiar el nombre a al DBMS, y lo llamaron PostgreSQL (versión 6.0) para reflejar la relación entre PostgreSQL y las versiones recientes de SQL. Se crearon nuevas mejoras y modificaciones, que repercutieron en un 20-40% más de eficiencia, así como la incorporación del estándar SQL92. (7)

Ventajas

- Por su arquitectura de diseño, escala muy bien al aumentar el número de CPUs y la cantidad de RAM.
- Soporta transacciones y desde la versión 7.0, claves ajenas (con comprobaciones de integridad referencial).
- Tiene buen soporte para triggers y procedimientos en el servidor.
- Soporta un subconjunto de SQL92. Además, tiene ciertas características orientadas a objetos.

Desventajas:

- Consume bastantes recursos y carga el sistema.
- Límite del tamaño de cada fila de las tablas a 8k (se puede ampliar a 32k recompilando, pero con un coste añadido en el rendimiento).
- Menos funciones en PHP.

Oracle es un sistema de gestión de base de datos relacional fabricado por Oracle Corporation Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacando su: Soporte de transacciones, estabilidad, escalabilidad, soporte multiplataforma.

Historia

El manejador de Base de datos ORACLE, surgió a final de los años 70 y principio de los años 80 por Oracle Corporation, en el año 1977 el precursor a Oracle, Larry Ellison, fundo los Laboratorios de Desarrollo de software, junto a Bob Miner, y Ed Oates. Ya para el año 1978 tenían la versión 1 de ORACLE, escrita en el lenguaje ensamblador, esta se ejecutaba en PDP-11 bajo RSX, con 128K de memoria. La aplicación separaba el código de ORACLE del usuario. Oracle V1 nunca se lanzo oficialmente. En 1979 aparece ORACLER Versión 2, el primer Gestor de Base de Datos relacional SQL

este si fue lanzado. La tercera versión de ORACLE aparece en 1983 hecha en lenguaje C, de esta manera se mantienen hasta que en el 1988 aparece ORACLE con mayores ventajas y funcionalidades que permitían a varios usuario trabajar sobre una misma tabla y con funciones para el guardado de la base de datos en ejecución, En los años 90 ORACLE revelo un avance significativo en las tecnologías, aumentan su potencia de cálculo con la introducción de PL/SQL y una mayor efectividad en la administración con el Universal Server en el año 1996. Para los primeros años del nuevo milenio Oracle abrazó la tecnología del internet y continuó innovando y lanzando productos como el ORACLE Fusion y dando soporte a la compañía de Linux. (6)

Ventajas

- Puede ejecutarse en todas las plataformas.
- Oracle soporta todas las funciones que se esperan de un servidor "serio": un lenguaje de diseño de bases de datos muy completo (PL/SQL) que permite implementar diseños "activos", con triggers y procedimientos almacenados, con una integridad referencial declarativa bastante potente.
- Permite el uso de particiones para la mejora de la eficiencia, de replicación e incluso ciertas versiones admiten la administración de bases de datos distribuidas.
- El software del servidor puede ejecutarse en multitud de sistemas operativos.
- Existe incluso una versión personal para Windows 9x, lo cual es un punto a favor para los desarrolladores que se llevan trabajo a casa.
- ORACLE es la base de datos con más orientación hacía INTERNET.
- Un aceptable soporte.

Desventajas

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

MySQL es un sistema de gestión de base de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB desde enero de 2008 una subsidiaria de Sun Microsystems desarrolla MySQL como software libre en un esquema de licenciamiento dual.

Historia

MySQL surgió alrededor de la década del 90, Michael Windenis comenzó a usar mSQL para conectar tablas usando sus propias rutinas de bajo nivel (ISAM). Tras unas primeras pruebas, llegó a la conclusión de que mSQL no era lo bastante flexible ni rápido para lo que necesitaba, por lo que tuvo que desarrollar nuevas funciones. Esto resulto en una interfaz SQL a su base de datos, totalmente compatible a mSQL. (4)

El origen del nombre MySQL no se sabe con certeza de donde proviene, por un lado se dice que en sus librerías han llevado el prefijo “my” durante los diez últimos años, por otra parte, la hija de uno de los desarrolladores se llama My. Así que no está claramente definido cuál de estas dos causas han dado lugar al nombre de este conocido gestor de bases de datos. (5)

Ventajas

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

Desventajas

- Un gran porcentaje de las utilidades de MySQL no están documentadas.
- No es intuitivo, como otros programas.

Microsoft SQL Server es un sistema de gestión de bases de datos relacionales (SGBD) basado en el lenguaje Transact-SQL, y específicamente en Sybase IQ, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea.(7)

Ventajas

- Un sistema barato y la tendencia de los directivos a aceptar preferentemente productos de Microsoft.
- La interfaz de acceso OLE DB y ADO. Aunque se trata de una interfaz universal, SQL Server es una de las primeras bases de datos en soportarla.
- Limpieza automática de las memorias intermedias sucias del caché al disco.
  La tarea de limpieza tiene las siguientes ventajas:
  - Mejor utilización de la CPU
  - Menor necesidad de limpieza de las memorias intermedias durante el procesamiento de las transacciones
  - Puntos de verificación más rápidos
  - Menor tiempo de recuperación
- Registro de las transacciones de tal modo que las actualizaciones en una de ellas siempre se puedan recuperar o reducir al último estado consistente si el equipo cliente o servidor falla.

Desventajas

- Bloqueo a nivel de página, dispositivos con crecimiento manual, un tamaño de página fijo y demasiado pequeño (2048KB).
- Software de Licencia Propietaria.
- Solo para sistema operativo Microsoft.

1.7. Justificación de la elección de SQL SERVER.

Para la realización del trabajo se escogerá como SGBD, SQL SERVER, en específico SQL SERVER 2000, viendo las grandes ventajas de este gestor que aunque es privativo, es un sistema barato y muy potente. SQL SERVER 2000 otorga a los administradores una herramienta potencialmente robusta, provista de las herramientas suficientes que le permiten mantener un óptimo nivel de seguridad en la

utilización de los recursos del sistema y de la base de datos. SQL SERVER cuenta con características, que lo hacen un SGBD fácil de de uso y de gran aceptación, entre ellas se tienen, su gran escalabilidad y disponibilidad, así como características de Bases de Datos Corporativas, ya que protege la integridad de los datos y minimiza la carga de trabajo de consultas a las bases de Datos por miles de usuarios, por otra parte la facilidad de instalación, distribución y utilización así como la gran cantidad de documentación disponible para el aprendizaje de esta potente herramienta , lo convierten en un SGBD a tener en cuenta dentro del campo de procesamiento de datos.

1.8. Lenguaje a Utilizar.

Lenguaje SQL El SQL es un lenguaje de acceso a bases de datos que explota la flexibilidad y potencia de los sistemas relacionales permitiendo gran variedad de operaciones sobre los mismos. Es un lenguaje declarativo de "alto nivel" o "de no procedimiento", que gracias a su fuerte base teórica y su orientación al manejo de conjuntos de registros, y no a registros individuales, permite una alta productividad en codificación y la orientación a objetos.

Historia

Los orígenes del SQL están ligados a los de las bases de datos relacionales. La historia de Structured Query Language (SQL) empieza en 1974 con la definición, por parte de Donald Chamberlin y de otras personas que trabajaban en los laboratorios de investigación de IBM, de un lenguaje para la especificación de las características de las bases de datos que adoptaban el modelo relacional. Este lenguaje se llamaba SEQUEL (Structured English Query Language) y se implementó en un prototipo llamado SEQUEL-XRM entre 1974 y 1975. Las experimentaciones con ese prototipo condujeron, entre 1976 y 1977, a una revisión del lenguaje (SEQUEL/2), que a partir de ese momento cambió de nombre por motivos legales, convirtiéndose en SQL. El prototipo (System R), basado en este lenguaje, se adoptó y utilizó internamente en IBM y lo adoptaron algunos de sus clientes elegidos. Gracias al éxito de este sistema, que no estaba todavía comercializado, también otras compañías empezaron a desarrollar sus productos relacionales basados en SQL. A partir de 1981, IBM comenzó a entregar sus productos relacionales y en 1983 empezó a vender DB2. En el curso de los años ochenta, numerosas compañías (por ejemplo Oracle y Sybase, por citar algunos) comercializaron productos basados en SQL, que se convierte en el estándar industrial de hecho por lo que respecta a las bases de datos relacionales. En 1986, el ANSI adoptó SQL como estándar para los lenguajes relacionales y en 1987 se transformó en

estándar ISO. Esta versión del estándar va con el nombre de SQL/86. En los años siguientes, éste ha sufrido diversas revisiones que han conducido primero a la versión SQL/89 y, posteriormente en 1999 SQL/2000, donde se agregaron expresiones regulares, consultas recursivas (para relaciones jerárquicas), triggers y algunas características orientadas a objetos. En el 2003 SQL/2003 introduce algunas características de XML, cambios en las funciones, estandarización del objeto sequence y de las columnas auto numéricas. Más tarde en el 2006 SQL/2006 ISO/IEC 9075-14:2006 define las maneras en las cuales el SQL se puede utilizar conjuntamente con XML. Define maneras importar y guardar datos XML en una base de datos SQL, manipulándolos dentro de la base de datos y publicando el XML y los datos SQL convencionales en forma XML. Además, proporciona facilidades que permiten a las aplicaciones integrar dentro de su código SQL el uso de XQuery, lenguaje de consulta XML publicado por el W3C (World Wide Web Consortium) para acceso concurrente a datos ordinarios SQL y documentos XML. Y ya hoy en día se cuenta con una distribución de SQL/2008 el cual ofrece mejoras en disponibilidad, la capacidad de administración, la programación, de escalabilidad, rendimiento y de seguridad.

En la actualidad el SQL es el estándar de facto de la inmensa mayoría de los SGBD comerciales. Y, aunque la diversidad de añadidos particulares que incluyen las distintas implementaciones comerciales del lenguaje es amplia, el soporte al estándar SQL-92 es general y muy amplio. (8)

1.9. Herramientas a Utilizar.

Para la realización del siguiente trabajo se utilizaran varias herramientas las cuales se describen a continuación, así como los lenguajes de programación utilizados, a continuación una breve explicación de cada uno de ellos y sus principales características.

Framework Symfony.

Los frameworks simplifican el desarrollo de las aplicaciones mediante la automatización de muchas de las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al programador a crear código más legible y más fácil de mantener. Symfony es un framework para construir aplicaciones web con PHP. En otras palabras, Symfony es un enorme conjunto de herramientas y utilidades que simplifican el desarrollo de las aplicaciones web. Symfony emplea el tradicional patrón de diseño MVC (modelo-vista-controlador) para separar las distintas partes que forman una aplicación web. El modelo representa la información con la que trabaja la aplicación y se encarga de acceder a los datos.

La primera versión se publicó en octubre de 2005; su licencia es de tipo software libre; está desarrollado completamente con PHP 5 ha sido desarrollado por una empresa francesa llamada Sensio Labs; ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. Yahoo ha elegido Symfony como su framework PHP de desarrollo, con el que ha construido Yahoo Bookmarks (20 millones de usuarios y 12 idiomas) y partes de Yahoo Answers. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas Unix, Linux, como en plataformas Windows.

Se utiliza Symfony 1.0.17 siendo esta la última versión estable del producto, Symfony ofrece un sin número de facilidades en cuanto la conexión con Base de Datos encargándose el mismo de gestionar dicha conexión con la única especificación del SGBD que se utilice, otra característica es en cuánto al tratamiento de las vistas y procedimientos almacenados, al utilizar MVC (modelo vista controlador) permite una mayor reusabilidad de código, además que al darle tratamiento al trabajo por capas, cualquier modificación en una capa no afecta a la otra.

Embarcadero ER/Studio

Es una herramienta de modelado de datos fácil de usar y multinivel, para el diseño y construcción de bases de datos a nivel lógico y físico. Direcciona las necesidades diarias de los administradores de bases de datos, desarrolladores y arquitectos de datos que construyen y mantienen aplicaciones de Bases de Datos grandes y complejas. El mismo ofrece funcionalidades tales como capacidad fuerte en el diseño lógico. Sincronización bidireccional de los diseños lógico y físico, construcción automática de Base de Datos, ingeniería inversa de Base de Datos, documentación basada en HTML, un Repositorio para el modelado. (11)

ER/Studio es una excelente herramienta para el modelado de Bases de Datos, una de sus grandes potencialidades es la cantidad de SGDB, con los cuales es compatible entre ellos se pueden citar:

- Oracle 7.3® , 8.x & 9i
- Sybase® System 11.9.2, 12.x & 12.5
- Microsoft® SQL Server 6.5, 7 & 2000
- IBM® DB/2® Universal Database® 4.x, 5.x, 6.x & 7.x
- Open Systems, OS/390® & AS/400 4.5

- Informix® OnLine and SE
- SQL Anywhere™ and Watcom™ SQL
- InterBase ® 4
- Microsoft Access 2.0, 95, 97 & 2000
- Microsoft Visual FoxPro®

1.10. Conclusiones

En este capítulo se realizó un estudio sobre las tecnologías actuales y todo lo concerniente a los Sistemas Gestores de Base de Datos y las Bases de Datos, así como la descripción de las herramientas utilizadas para el desarrollo del trabajo.

A partir de lo antes expuestos podemos definir que en la actualidad los sistemas de base de datos son una tecnología imprescindible en la sociedad moderna, debido a la cantidad de información circulante es imposible manejar esta de una forma rápida, eficaz y segura sin contar con la ayuda de un SGBD.

CAPÍTULO

2

DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA

2.1. Introducción

El capítulo siguiente describe, la arquitectura, integración con otros programas, en este caso con el framework Symfony, una breve descripción de cada tabla y los valores que almacenan las mismas, así como los modelos correspondientes a la base de datos en cuestión, teniendo en cuenta los requisitos tanto funcionales, como no funcionales trazados desde el negocio y la fase de requerimientos.

2.2. Integración con el Framework Symfony

La información mínima que necesita Symfony para realizar peticiones a la base de datos es su nombre, los datos de acceso y el tipo de base de datos. Esta información se indica en el archivo databases.yml que se encuentra en el directorio config.

Las opciones de la conexión se establecen para cada entorno. Se pueden definir diferentes opciones para los entornos prod, dev y test, o para cualquier otro entorno definido en la aplicación.

Para el caso de la conexión con SQL Server el prefijo que se debe utilizar es mssql, para modificar el fichero .yml y se debe verificar que las librerías (pdo_mssql.dll, msqll.dll) necesarias están activas en el servidor web que se esté utilizando.

También es posible redefinir esta configuración en cada aplicación, estableciendo diferentes valores para las opciones en un archivo específico de la aplicación, de esta forma es posible disponer de políticas de seguridad diferentes para las aplicaciones públicas y las aplicaciones de administración del proyecto, y definir distintos usuarios de bases de datos con privilegios diferentes.

2.3. Descripción de la Arquitectura.

La arquitectura Cliente-Servidor (SQL Server)

SQL Server utiliza la arquitectura *Cliente / Servidor* para coordinar el trabajo entre el equipo cliente y el equipo servidor. Dependiendo del tipo de aplicación que se quiera programar dependerá la repartición de la carga de trabajo entre el cliente y el servidor.
El **equipo cliente** se suele encargar de la parte lógica y de mostrar la información al usuario que realiza una petición, en el cliente consta de los siguientes componentes:

- ***Aplicación Cliente:*** Construye sentencias Transact-SQL que envía al servidor y recibe sus resultados.
- ***Base de Datos API (Application Programming Interfaces):*** Usa proveedores, o DLL (Data Definition Language) para enviar sentencias T-SQL y retornar resultados, ocultando la complejidad de los protocolos de red requeridos para comunicarse con SQL Server.
- ***Librería de Red del Cliente:*** Controla las conexiones de red y enrutamiento para la transmisión de peticiones y respuestas de SQL usando el protocolo de red adecuado.

El **equipo servidor SQL Server**, se encarga de administrar la base de datos, de gestionar los recursos del servidor. (CPU, memoria, etc.), y por supuesto de resolver y devolver en forma de solución la petición realizada por el cliente. En el servidor por su parte cuenta con los componentes expuestos a continuación:

- ***Librerías de Red del Servidor:*** SQL Server puede monitorear múltiples librerías de red del cliente al mismo tiempo. Los protocolos de red usados por SQL Server son TCP/IP, Named Pipes, NWLink, IPX/SPK, VIA ServerNet II SAN, VIA GigaNet SAN, Banyan VINES, y AppleTalk.
- ***Servicio Open-Data:*** Hace los servicios de datos disponibles a un cliente. Este componente maneja las conexiones de red, pasando las peticiones de los clientes a SQL Server para su procesamiento y retornando los resultados hacia los clientes SQL. Automáticamente escucha todas las librerías instaladas en el servidor.
- ***Motor Relacional:*** Traduce sentencias T-SQL, optimiza y ejecuta planes de ejecución, procesa sentencias de definición (DDL), y provee seguridad a los datos.
- ***Motor de Almacenamiento***: Maneja los archivos físicos de la base de datos así como el espacio de disco usado por estos, maneja buffers y operaciones físicas de entrada y salida (I/O), controla

concurrencia, procesa operaciones de log y recuperación, e implementa comandos utilitarios de la base de datos, como backup(salvado) y restore(recuperación) .

Fig. 2.1 Arquitectura Cliente –Servidor en SQL server

Arquitectura del sistema

La Arquitectura de Software es la organización y estructura fundamental de un sistema. Se representa a través de sus componentes, las relaciones entre ellos, el ambiente y los principios que orientan su diseño y evolución. Se concentra en requerimientos no funcionales, que se satisfacen mediante los modelos y diseños de la aplicación. Un diseño correcto de la Arquitectura del sistema es esencial para el éxito o fracaso del proyecto.

La base arquitectónica del Sistema de Facturación y Cobro para la Empresa de Gas Manufacturado (Producto: SISTMET) se ha modelado haciendo uso del Patrón Arquitectónico Modelo Vista Controlador (Model View Controller MVC). La razón fundamental por la que se escoge dicha arquitectura es porque el framework de desarrollo que se utilizará, está basado en este patrón MVC, con el objetivo de utilizar la separación de las responsabilidades de cada una de las capas lógicas que lo conforman y así lograr facilidades en desarrollo, la reutilización.

2.4. Selección de los requisitos.

Los requisitos son propiedades que deben ser exhibidas por un software para resolver un problema particular. Condiciones o capacidades que necesita el usuario para resolver un problema o conseguir un objetivo determinado.

Los requisitos de un software suelen ser una combinación compleja de los requisitos de diferentes personas en diferentes niveles de una organización y del entorno en el cual operará el software, Es fundamental que un requisito sea verificable, los requisitos deben ser lo más claros y no ambiguos que se pueda, y cuantificables (si es posible).

Una agrupación de los requisitos puede ser del tipo Funcionales-No funcionales, que es la que se utiliza a continuación, los requisitos Funcionales describen las funciones que lleva a cabo el software, cómo debe reaccionar ante ciertas entradas y cómo debe comportarse en situaciones particulares. Por su parte los requisitos No Funcionales son restricciones sobre las funciones o servicios ofrecidos por el sistema (13).

2.4.1. Requisitos Funcionales

R1. Introducir Nombre Fichero TPL
R2. Seleccionar Ruta a Facturar
R3. Buscar Datos de Clientes a Facturar
R4. Cargar Datos de Clientes a Facturar
R5. Cargar fichero al TPL
R6. Preparar TPL.
R7. Crear Modelo de Tramitadora.
R8. Mostrar datos a cargar al TPL
R9. Imprimir datos a cargar en el TPL
R10. Cargar datos a Facturar
R11. Actualizar Bases de Datos UEB Comercial
R12. Selección de Casa Comercial para Facturar
R13. Selección criterios a facturar
R14. Cargar la Tarifa Escalonada
R15. Cargar consumo de cliente
R16. Facturar el cliente
R17. Mostrar clientes facturados
R18. Corregir errores facturación
R19. Guardar datos de Facturación
R20. Imprimir chequeras
R21. Imprimir cobros

R22.Descargar datos del TPL

R23. Comprobación de datos del TPL

R24. Comprobación de % de lecturas

R25. Buscar errores en datos procesados por el TPL

R26. Corrección de errores en datos del TPL

R27. Cálculo de consumo del cliente en el mes

R28. Guardar datos de lecturas y consumo de cada cliente

R29. Mostrar informes de lecturas corregidas

R30. Exportar datos de lectura a fichero

R31. Emitir listado a entregar al lector-cobrador

R32. Registrar cobro de clientes en las casas comerciales

R33. Generar recibo de cobro al cobrar servicio de gas

R34. Generar IDC (Informe Diario de Caja)

R35. Generar Modelo liquidación en efectivo por cobrador

R36. Generar Informe Diario del Trabajo realizado por cobradores

R37. Comprobar las cuentas cobradas y por cobrar con las presentes en el modelo Cargo al Cobrador

R38. Generar Resumen Diario de cobros en efectivo

R39. Generar Movimiento Diario de Efectivo en Caja

R40. Generar Devolución de Efectivo

R41. Generar Reporte de Cuadre H67

R42. Generar Reporte de Cuadre de Talo

R43. Generar Reporte Resumen de Cobro

R44. Generar Reporte Resumen de Deuda

R45. Generar Reporte Resumen de H67 por caja

R46. Generar Reporte Resumen de Caja

R47. Generar Reporte del lector cobrador

R48. Generar Reporte Certificación de Deuda

2.4.2. Requisitos no Funcionales

Usabilidad

- Preparar a los Usuarios en la gestión de Roles y Permisos.
- Consistencia en la interfaz de usuario.

Confiabilidad

- Ante fallas del sistema, corregirlas en un período menor a un mes.
- Corregir errores de facturación.
- Búsqueda y corrección de errores en datos del TPL.

Rendimiento

- El sistema debe responder en un tiempo relativamente rápido a las peticiones del usuario (menos de 5 segundos).

Requerimientos de Hardware

- Hardware de la estación de trabajo del servidor Web, donde se ejecutará el sistema.

  - Servidor Web (Procesador Pentium D 2x2 cache 3.00 GHz, Memoria RAM no menos 1 GB)

- Hardware de la estación de trabajo servidor de Base de Datos.

  - Servidor de Base de Datos (Procesador Pentium D 2x2 cache. 3.00 GHz, Memoria RAM 2 GB, Almacenamiento en discos de 80 GB)

- Hardware de la estación de trabajo del cliente

  - Cliente web (Procesador Pentium 3 (o superior), Memoria RAM mínima de 256 MB)

  - Cliente web (Impresora de impactos, Se necesita una impresora matricial: EPSON LX-300 o superior a ella para imprimir las chequeras, cobros y datos de los clientes que se van a cargar en el TPL.)

Requisitos de Software

- Software instalado en la estación de trabajo del cliente.

  - Sistema Operativo Windows 95 o Superior.

  - El Navegador Web compatible con HTML 2.0 y CSS, podrá ser Netscape 3 (o superior), Internet Explorer 4.2 (o superior) y compatibles.

- Software instalado en el Servidor Web.

  - Servidor Web Apache.

- Software instalado en el servidor de Base de Datos

  - Servidor de Bases de Datos SQL-Server 2000

Requisitos de Seguridad

- Seguridad del sistema.

  - *Confidencialidad*: la información manejada por el sistema está protegida de acceso no autorizado y divulgación.

  - *Integridad:* la información manejada por el sistema es objeto de cuidadosa protección contra la corrupción y estados inconsistentes. Se incluyen también mecanismos de chequeo de integridad y realización de auditorías por personal calificado de la entidad.

  - *Disponibilidad:* los usuarios autorizados (autenticados por dominio y según su roll) se les garantizará el acceso a la información, los dispositivos o mecanismos utilizados para lograr la seguridad no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado.

2.5. Descripción de las Tablas.

Tabla 2.1 Persona

Nombre de la Tabla	persona
Descripción	La tabla persona almacenara la información correspondientes a todas las personas involucradas de alguna u otra forma en el negocio, ya

	sean clientes o trabajadores, a partir de ella se hacen dos tablas más, que son subtipos de ella.	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
municipio_id	int	Identificador de la relación.
primer_nombre	varchar (20)	Nombre 1 de la persona.
segundo_nombre	varchar(20)	Nombre 2 de la persona.
primer_apellido	varchar(20)	Apellido paterno.
segundo_apellido	varchar(20)	Apellido materno.
ci	varchar(11)	Carnet de Identidad.
create_at	timestamp	Fecha en que ingresa la persona ya sea cliente o trabajador a la base de datos.
update_at	timestamp	En caso de que se realice alguna modificación en los datos de la persona.

Tabla 2.2 Cliente

Nombre de la Tabla	cliente	
Descripción	Contiene la información referente a todos los clientes que cuentan con el servicio de gas manufacturado, los clientes no interactúan con el sistema, aunque forman parte del.	
Nombre del Campo	**Tipo**	**Descripción**
id_persona	int	Identificador de la persona.
codigo	varchar(12)	Código dentro del sistema asignado al cliente.
numero_metro	varchar(4)	Número del metro del cliente
numero_nucleo	varchar(4)	Número de oficoda.
cant_personas_nucleo	int	Cantidad de personas de un cliente.

observacion	text	Algún apunte sobre el cliente.

Tabla 2.3 Trabajador

Nombre de la Tabla	trabajador	
Descripción	Esta tabla contiene información que lo que refiere a los trabajadores de la empresa o casa comercial en cuestión, aquí se registran todos los trabajadores tanto aquellos que interactúan con el sistema como aquellos que no, los trabajadores también pueden ser clientes.	
Nombre del Campo	**Tipo**	**Descripción**
id_persona	int	Identificador de la persona.
rol_id	int	Identificador de la relación.
usuario	varchar(15)	Nombre de usuario del sistema.
password	varchar(30)	Contraseña para acceder al sistema.
trabajador_sistema	bool	Conocer si interactúa con el sistema o no.

Tabla 2.4 Rol

Nombre de la Tabla	rol	
Descripción	Los roles son las actividades que desempeñan los trabajadores dentro de la empresa o casa comercial, como pueden ser, lector-cobrador, trabajadora de la caja entre otros.	
Nombre del Campo	**Tipo**	**Descripción**
Id	int	Identificador de la tabla.
nombre_rol	varchar(30)	Nombre que describe el rol de una persona en el sistema, es único.
descripción	text	Breve descripción del rol.

Tabla 2.5 Provincia

.**Nombre de la Tabla**	provincia	
Descripción	Contiene las provincias en las que esta divido políticamente nuestro país.	
Nombre del Campo	**Tipo**	**Descripción**
Id	int	Identificador de la tabla.
nombre_corto	varchar(10)	Siglas abreviadas de la provincia.
nombre_largo	varchar(25)	Nombre completo de la provincia.

Tabla 2.6 Municipio

Nombre de la Tabla	municipio	
Descripción	Contiene los municipios del país por cada provincia.	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
provincia_id	int	Identificador de la relación.
nombre	varchar(25)	Nombre del municipio.

Tabla 2.7 Ruta

Nombre de la Tabla	ruta	
Descripción	Las rutas son un cumulo de direcciones de un mismo municipio agrupadas, para organizar el cobro del servicio y las lecturas de los Metrocontadores.	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
municipio_id	int	Identificador de la relación.
folio1	varchar(4)	Identifica el folio de la ruta.

Tabla 2.8 Clave Lectura

Nombre de la Tabla	clave_lectura

Descripción	Estas claves son situaciones que se pueden encontrar los lectores-cobradores cuando realizan las lecturas de los metrocontadores, como pueden ser metro roto, lectura alta, metro desmantelado entre otras.	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
clave	int	Número que identifica el tipo de clave.
descripcion	text	Breve descripción de la clave.
create_at	timestamp	Fecha en que ingresa la clave a la base de datos.
update_at	timestamp	En caso de que se realice algún cambio en las claves, fecha en la cual se modifica la misma.

Tabla 2.9 Lectura

Nombre de la Tabla	lectura	
Descripción	Las lecturas son, aquellas que recogen los lectores-cobradores, para el nuevo sistema se guardaran las lecturas hasta tres años atrás por cada cliente, a partir de estas lecturas se sabe el consumo de los cliente y el saldo que deberán abonar a la empresa	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
persona_id	int	Identificador de la relación.
ruta_id	int	Identificador de la relación.
clave_lectura_id	int	Identificador de la relación.
fecha_lectura	date	Fecha en que se realizó la lectura.
create_at	timestamp	Fecha en que se introduce la lectura al sistema.
update_at	timestamp	Fecha en que se modifica la

		lectura luego de ser entrada al sistema.
lectura	int	Lectura tomada del metro.

Tabla 2.10 Factura

Nombre de la Tabla	factura	
Descripción	En esta tabla se guardan las facturas obtenidas luego del proceso de facturación.	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
persona_id	Int	Identificador de la relación.
trabajador_id	int	Identificador de la relación.
create_at	timestamp	Fecha en que se realiza la factura, puede ser introducida automáticamente.
importe	float	Saldo a pagar por el cliente mensualmente.

Tabla 2.11 Dirección

Nombre de la Tabla	direccion	
Descripción	Direcciones en las que viven las personas de la empresa.	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
ruta_id	int	Identificador de la relación.
persona_id	int	Identificador de la relación.
calle_transversal	varchar(20)	Nombre de la calle transversal.
calle	varchar(20)	Nombre de la calle.
sector	int	Número del sector en que vive.
consejo	varchar(20)	Consejo en que vive el cliente
numero	int	Número de la casa.

Tabla 12.12 Clave Lectura Siguiente

Nombre de la Tabla	clave_lectura_siguiente	
Descripción	La siguiente tabla contiene la relación de sucesión de las claves de facturación.	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
clave_lectura_id	int	Identificador de la relación.
clave_lectura_siguiente_id	int	Identificador de la relación y además define la siguiente clave.

Tabla 2.13 Libro

Nombre de la Tabla	libro	
Descripción	La tabla siguiente contiene los libros por los cuales están organizadas las rutas en los distintos municipios.	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
num_libro	char	Número del libro en cuestión.
municipio_id	int	Identificador de la relación, y además representa el municipio al que pertenece el libro.

Tabla 2.14 Crendencial

Nombre de la Tabla	credencial	
Descripción	La tabla credencial es la encargada de almacenar los permisos que se le atribuyen a un rol determinado	
Nombre del Campo	**Tipo**	**Descripción**
id	int	Identificador de la tabla.
nombre	varchar (25)	Indica un permiso determinado a un rol.
credencial_padre_id	int	Identificador de la relación

		recursiva de la tabla, esto debido a que la credencial puede englobar otros permisos dentro de ella.
rol_id	int	Identificador de la relación, y rol al que se le asigna el permiso en cuestión.

Tabla 2.15 Trabajador Lectura

Nombre de la Tabla	manugas_trabajador_manugas_lectura	
Descripción	La siguiente tabla de la relación de muchos a muchos entre las tablas trabajador y lectura.	
Nombre del Campo	**Tipo**	**Descripción**
persona_trabajador_id	int	Identificador de la relación (identificador del trabajador).
id	int	Identificador de la relación (identificador de la lectura).

2.6. Modelos de la Base de Datos.

Los modelos de base de datos que se presentan son el modelo lógico y el modelo físico, para realizar el diseño de la base de datos a partir del análisis, se convierte la información en un primer diseño de datos (modelo lógico) en este modelo se muestran los elementos obtenidos de la información y su interrelación, pero todavía ese modelo no es una base de datos, el sirve como entrada para el modelo físico, a partir del cual se obtiene la base de datos.

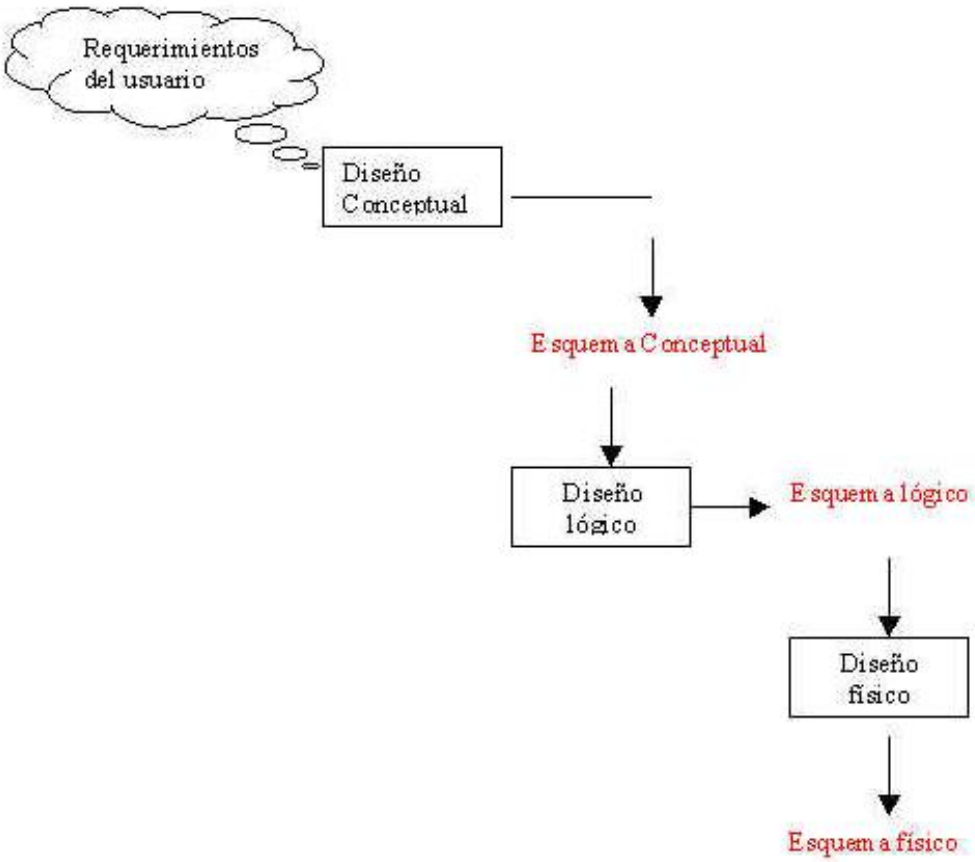


Fig. 2.2 Diseño de Base de datos.

2.6.1. Modelo lógico de la Base de Datos.

Ver Anexo 1

2.6.2. Modelo físico de la Base de Datos.

Ver Anexo 2

2.7. Programación del sistema.

La programación del sistema estará fundamentalmente, del lado del servidor web, una de las razones fundamentales por la cual se ha decidido hacer esto es para que el sistema sea totalmente independiente del gestor de base de datos lo que permite cambiar del mismo sin ningún tipo de problemas, esto le da una fuerte flexibilidad al sistema y al diseño realizado, teniendo en cuenta la migración hacia el software libre que se lleva a lo largo del país una posible migración hacia PostgreSQL por citar un ejemplo sería fácil y sin ningún tipo de contratiempos, y solamente se cambiarían algunos parámetros de conexión, sin afectar los procesos llevados a cabo normalmente por el sistema.

Otra ventaja por la que se toma en cuenta la programación de lado del servidor web es por la facilidad que brinda el framework Symfony al manejar las tablas como clases y los datos de las mismas como objetos, lo que nos da la oportunidad de aprovechar al máximo las capacidades de la programación orientada a objetos.

2.8. Conclusiones

En el capítulo anterior se describe el diseño realizado, basado en el análisis y la ingeniería llevada a cabo por el analista del proyecto, así también consultando la antigua base de datos de Sistmet, se ha descrito cada entidad presente en la base de datos, y han sido plasmados los modelos lógicos y físicos de la misma, en la descripción de cada entidad se tuvo minucioso cuidado a la hora de describir los atributos presentes en las mismas, así como los datos que almacenará la misma, de forma que sea comprensible para el cliente u otro interesado en el trabajo.

CAPÍTULO

3

VALIDACIÓN DEL DISEÑO REALIZADO

3.1. Introducción.

En el siguiente capítulo se definen importantes parámetros dentro del diseño de la base de datos, aspectos tan importantes como son la seguridad, normalización integridad entre otros, también se hace referencia a la validación funcional del trabajo como es el caso del llenado de la base y a través del mismo realizar las pruebas de volumen, en fin este capítulo es el complemento del trabajo realizado en el diseño de la base de datos, es decir las conclusiones del trabajo.

3.2. Integridad de datos.

La exigencia de integridad de los datos garantiza la calidad de los datos de la base de datos, la misma se encarga de asegurar que las operaciones ejecutadas por los usuarios sean correctas y rnantengan la consistencia de la base de datos. Para la lograr la integridad de datos se definen una serie de restricciones, una restricción es una regla que limita los valores que pueden estar presentes en la base de datos. (14)

El modelo de datos relacional de Cood incluye varias restricciones que se usan para verificar la validación de los datos en una Base de datos.

- *Integridad de entidad:* Define una fila como entidad única para una tabla determinada. La integridad de entidad exige la integridad de las columnas de los identificadores o la clave principal de una tabla, mediante índices y restricciones UNIQUE, o restricciones PRIMARY KEY.

- *Integridad de dominio*: Viene dada por la validez de las entradas para una columna determinada. La integridad de dominio puede exigir para restringir, el tipo mediante tipos de datos, el formato mediante reglas y restricciones CHECK, o el intervalo de valores posibles mediante restricciones FOREIGN KEY, restricciones CHECK, definiciones DEFAULT, definiciones NOT NULL y reglas.

- *Integridad referencial:* Protege las relaciones definidas entre las tablas cuando se crean o se eliminan filas. En SQL Server la integridad referencial se basa en las relaciones entre claves externas y claves principales o entre claves externas y claves exclusivas, mediante restricciones FOREIGN KEY y CHECK. La integridad referencial garantiza que los valores de clave sean coherentes en las distintas tablas. Para conseguir esa coherencia, es preciso que no haya referencias a valores inexistentes y que, si cambia el valor de una clave, todas las referencias a ella se cambien en consecuencia en toda la base de datos.
  Cuando se exige la integridad referencial, SQL Server impide a los usuarios:

  - Agregar o cambiar filas en una tabla relacionada si no hay ninguna fila asociada en la tabla principal.
  - Cambiar valores en una tabla principal que crea filas huérfanas en una tabla relacionada.
  - Eliminar filas de una tabla principal cuando hay filas relacionadas coincidentes.

- *Integridad definida por el usuario*: Permite definir reglas de empresa específicas que no pertenecen a ninguna otra categoría de integridad. Todas las categorías de integridad admiten la integridad definida por el usuario. Esto incluye todas las restricciones de nivel de columna y nivel de tabla en CREATE TABLE, procedimientos almacenados y desencadenadores.

3.3. Normalización de Base de Datos.

La descomposición o normalización es el proceso de dividir relaciones en múltiples relaciones para eliminar las anomalías y mantener la integridad de los datos. Para hacer esto, se usan las formas normales o reglas para relaciones estructuradas. (14)

Las formas nórmales buscan optimizar estas estructuras eliminando básicamente la redundancia utilizando como medio principal las dependencias funcionales. Las primeras de estas reglas son suficientes para cubrir las necesidades de la mayoría de las bases de datos. El creador de estas tres primeras formas normales (o reglas) fue Edgar F. Codd. A partir de ellas aparecen la forma normal de Boyce – Codd, la cuarta y quinta formas normales, los números altos de formas normales son más restrictivos, es decir la quinta forma normal cumple todas las anteriores.

Con la tercera forma normal los procesos de refinamiento sucesivo a realizar en la normalización de las relaciones, nos llevan a descomponer la tabla primaria hasta obtener tablas sencillas donde la redundancia y repeticiones sean mínimas. (15)

Primera Forma Normal

Una tabla se encuentra en primera forma normal si impide que un atributo de una tupla pueda tomar más de un valor, esto es lo mismo que decir que existan atributos divisibles en la tabla.

A continuación un ejemplo en cual el nombre es un atributo compuesto y puede dividirse, la primera tabla no se encuentra en primera forma normal (FN).

codigo	**nombre**	**calle**	**numero**
211044012321	NORMA PATERSON ROJAS	44	5109
211044012128	ISABEL REY GONZALEZ	44	5109
211045118401	ROGELIO TUYA BARRIOS	45 A	5409

Lo correcto sería:

codigo	**nombre**	**primer apellido**	**segundo apellido**	**calle**	**numero**
211044012321	NORMA	PATERSON	ROJAS	44	5109
211044012128	ISABEL	REY	GONZALEZ	44	5109
211045118401	ROGELIO	TUYA	BARRIOS	45 A	5409

Segunda Forma Normal

Ocurre si una tabla está en primera forma normal y además cada atributo que no sea clave, depende de forma funcional completa respecto de cualquiera de las claves. Toda la clave principal debe hacer dependientes al resto de los atributos, si hay atributos que dependen solo de parte de la clave, entonces esa parte de la clave y esos atributos formarán otra tabla.

codigo	**nombre**	**primer apellido**	**segundo apellido**	**calle**	**numero**	**municipio**

211044012321	NORMA	PATERSON	ROJAS	44	5109	1
211044012128	LUIS	NARANJO	BENITEZ	44	5335	1
211045118401	JOSE	CRUZ	AGUERO	100	5409	11

Aquí codigo y municipio forman la clave principal para esta tabla, solo el nombre, los apellidos y el número tienen dependencia funcional completa. La calle depende de forma completa del municipio. La tabla no está en segunda FN. Normalizando a segunda FN el ejemplo anterior quedaría:

codigo	**nombre**	**primer apellido**	**segundo apellido**
211044012321	NORMA	PATERSON	ROJAS
211044012128	LUIS	NARANJO	BENITEZ
211045118401	JOSE	CRUZ	AGUERO

codigo	**calle**	**numero**	**municipio**
211044012321	44	5109	1
211044012128	44	5335	1
211045118401	100	5409	11

Tercera Forma Normal

Ocurre cuando una tabla está en 2FN y además ningún atributo que no sea clave depende transitivamente de las claves de la tabla. Es decir no ocurre cuando algún atributo depende funcionalmente de atributos que no son clave.

codigo	**nombre**	**primer apellido**	**segundo apellido**	**municipio**	**IdMunicipio**
211044012321	NORMA	PATERSON	ROJAS	Playa	1
211044012128	ISABEL	REY	GONZALEZ	Playa	1
211045118401	JOSE	CRUZ	AGUERO	Marianao	11

El municipio depende funcionalmente del IdMunicipio, lo que hace que no esté en 3FN. Lo correcto sería:

codigo	**nombre**	**primer apellido**	**segundo apellido**	**IdMunicipio**

211044012321	NORMA	PATERSON	ROJAS	1
211044012128	ISABEL	REY	GONZALEZ	1
211045118401	JOSE	CRUZ	AGUERO	11

municipio	**IdMunicipio**
Playa	1
Marianao	11

Una vez llevado a cabo el proceso de normalización, se concluye el mismo dejando la base de datos en tercera forma normal, sin violar ninguna regla, o restricción, tratando de mantener y respetar la integridad de la base de datos, y eliminando las anomalías y datos redundantes.

3.4. Análisis de redundancia de información.

Con el diseño de la nueva base de datos para Sistmet, se elimina gran parte de la redundancia de datos presente en la antigua BD, un buen diseño de BD tributa a la casi anulación de dicha redundancia. En esta tarea juega un papel muy importante el trabajo de normalización realizado, además se han eliminado gran cantidad de entidades de la antigua BD, las cuales contenían información que puede ser obtenida a través de cálculos.

Algunos ejemplos de lo antes expuesto son:

- Eliminación de la tabla nuevos, todos los clientes aparecerán en la tabla cliente la cual reemplaza a la tabla maestro.

- Eliminación de la tabla histórico, se mantendrán las lecturas el tiempo que se estime necesario en la base de datos y a partir de cálculos se obtendrá la información que se requiera del cliente.

En el caso de la tabla clave_lectura y clave_lectura_id hay información repetida, esto se hace necesario debido a que se requiere de una sucesión lógica en las claves y se hace de esta forma para lograr el almacenamiento correcto de la información en la base de datos de lo contrario podría traer inconformidades del cliente con el producto.

3.5. Análisis de la seguridad de la base de datos.

La seguridad es un punto crucial dentro de una base de datos para mantener la integridad y disponibilidad de la información, ya que a través de la misma se puede controlar el acceso no autorizado a los datos que se tengan almacenados, también es parte de la seguridad el tener un respaldo (backup) de la base datos sobre todo si es posible en algún lugar que no sea donde se encuéntrala instalación de la misma.

El gestor de base datos permite, la asignación de permisos, a los diferentes usuarios que hayan sido definidos para el acceso a la base de datos, fundamentalmente en lo referente a actualizar (UPDATE), insertar (INSERT), y borrar (DELETE) la información presente en la base de datos, el administrador de bases de datos se encarga de la asignación de dichos permisos, este administrador será el encargado de mantener la base de datos, para el acceso a la base de datos desde la aplicación se crea un pequeño modulo de seguridad que comprende la siguientes tablas:

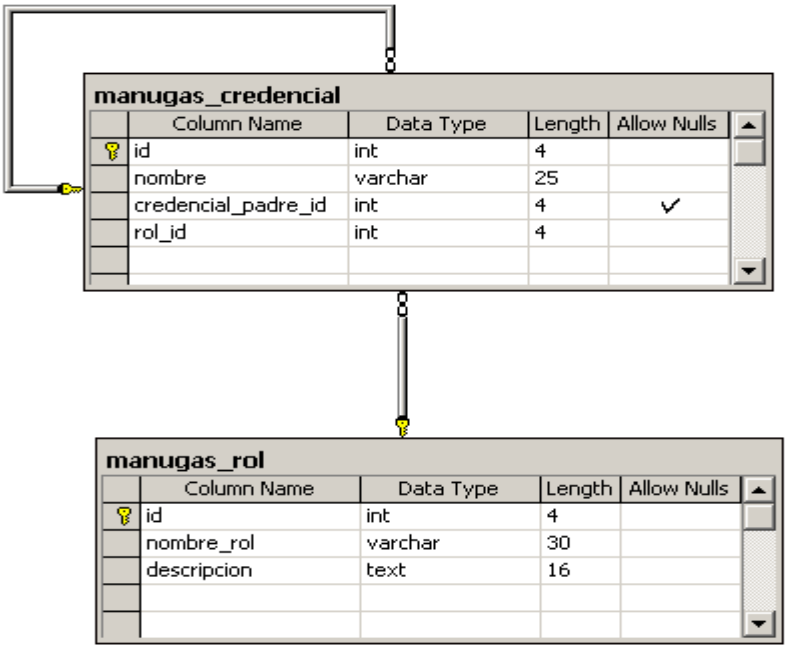


Fig. 3.1 Módulo de seguridad

Esto permite a los usuarios del sistema, tener un determinado nivel de acceso a la información.

Para realizar una copia de respaldo, debemos tener en cuenta, el tipo de copia a realizar, esta puede ser una copia completa (FULL), una copia diferencial (DIFF), o un registro de las acciones realizadas en la base de datos (LOG). Una copia completa crea un backup completo de la base de datos, una copia diferencial, crea un backup en donde se guardan los cambios a partir de la última copia diferencial o completa, y por su parte el registro únicamente guarda las transacciones que se fueron realizadas en la base de datos.

SQL Server permite programar tareas, que se realicen de forma automática, dado un tiempo y una fecha es posible, a través de los comandos BACKUP y RESTORE, salvar o restaurar copias de una base de datos especificando el tipo de copia y la fecha en que se quiere se ejecute dicha tarea.

3.6. Trazabilidad de las acciones.

La trazabilidad es la capacidad que posee un sistema para monitorear las acciones que se realizan sobre él, los registros (logs), ficheros que se generan cuando se realizan una determinada acción sobre el sistema en este caso la base de datos, son un mecanismo efectivo para esta tarea, el SQL Server posee un sistema de administración de logs estos se guardan en la carpeta LOGS de la instalación del mismo, en esta direccion se guarda un fichero diario, dentro del fichero están contenidas un grupo de acciones realizadas sobre el gestor, incluso errores, para poder obtener información de estos ficheros se debe acceder a ellos a través del gestor, dentro de cada fichero se encontrara, con la fecha y la hora exacta la acción que se realizó y cual fue esta.

3.7. Validación Funcional.

Para un llenado voluminoso de la base de datos, con el fin de realizar pruebas de volumen, se utiliza la herramienta EMS SQL Data Generator 2008, esta herramienta permite, el llenado de tablas según determinados patrones, y aunque no genera datos consistentes o reales, sí son una muy buena práctica para las probar la consistencia de las tablas, y para permitirle a los desarrolladores probar el trabajo realizado hasta el momento.

Para el llenado de las tablas este se debe hacer en forma de árbol, debido a las llaves auto numéricas, para respetar la dependencia de llaves entre tablas, nunca se debe hacer en dirección contraria, esto también es importante para poder esclarecer algún que otro error en el diseño de la base de datos.
La prueba de volumen sujeta al elemento de prueba a grandes cantidades de datos, para determinar si se alcanzan los límites que hacen fallar al software. También identifica la carga máxima continua o volumen que el elemento de prueba puede manejar por un período dado.

Por ejemplo, si el elemento de prueba está procesando un sistema de expedientes de la base de datos para generar un informe, una prueba de volumen utilizaría una base de datos grande para la prueba, y comprobaría que el software se comportó normalmente y que produjo el informe correcto. (16)

3.8. Conclusiones

En el capítulo anterior se tomaron en cuenta, rasgos importantes para obtener una base de datos consistente, ejemplo de ello es la integridad de datos, es decir aceptar los datos correctos en la base de datos o por ejemplo la normalización de la misma con el fin de tener la información en solamente un lugar, así también se toma el tema de la seguridad, tanto del acceso a datos como el guardado automático por si falla el sistema no perder la información, se concluye el mismo con el tema de las pruebas dentro de la base de datos, este capítulo es sin duda de crucial importancia en la obtención de un producto acorde a lo esperado por el cliente.

Conclusiones Generales

Las bases de datos se utilizan desde hace muchos años atrás y con el pasar del tiempo han evolucionado y sufrido transformaciones y han surgido nuevos conceptos en estos. El trabajo anterior expone el diseño de una base de datos, en él se abordan temas sobre la actualidad de los sistemas de almacenamientos a partir de sus inicios, para caer en la especificidad de la situación problemática e intentar dar una solución a la misma.

La culminación del mismo está dado por el diseño de la base datos para la empresa de gas manufacturado, con el cual podrán aplicar un sistema de cobro escalonado según el consumo y no con el actual, lo que sin duda traerá beneficios y ganancias para esta empresa.

Se considera que se han cumplido los objetivos trazados al comienzo y ejecutado todas las actividades planteadas al inicio y todo esto ha conllevado a la demostración de la idea a defender que se expuso desde un principio y que ha guiado todo el desarrollo.
La base expuesta puede ser observada en el gestor de la forma que se muestra en el anexo 3.

Recomendaciones

Se recomienda a los interesados:

- La implementación del diseño realizado de la base de datos para la Empresa de Gas Manufacturado luego de los resultados arrojados a través de la investigación.
- Continuar con la investigación del negocio para otros procesos que no se han dispuesto a automatizar en dicha empresa.
- Investigar sobre los temas de migración hacia el software libre, instando al cliente sobre un posible traspaso hacia este modelo.
- Una vez implementado el diseño expuesto, mantener sobre la base de datos un proceso de mantenimiento y copias de seguridad periódicas, con la intención de mantener la fiabilidad, integridad y funcionamiento óptimo de la base de datos.

Referencias Bibliográficas

1. Mato Gracia, Rosa M. Diseño de Bases de Datos, Octubre 2005.

2. Ángel, Miguel. Funciones de los Sistemas Gestores de Bases de Datos.[citado noviembre de 2008]; Disponible en: http://cnx.org/content/m17543/latest/

3. Microsoft Corporation, Microsoft Access.[citado: noviembre de 2008]; Disponible en: http://office.microsoft.com

4. Pecos, Daniel. PostgreSQL vs. MySQL. [citado: noviembre de 2008]; Disponible en: http://www.netpecos.org/docs/mysql_postgres/index.html

5. Aguilar, Vicente; Suau, Pablo. MySQL vs. PostgreSQL. [noviembre de 2008]; Disponible en: http://www.bisente.com/documentos/mysql-postgres.html

6. Oracle, Oracle's History: Innovation, Leadership, Results. [noviembre de 2008]; Disponible en: http://www.oracle.com/corporate/story.html

7. Microsoft. SQL Server 2008. [citado en noviembre de 2008]; Disponible en: http://www.microsoft.com/sqlserver

8. ANSI/ISO/IEC International Standard (IS). Database Language SQL. [citado: noviembre de 2008]; Disponible en: http://www.ncb.ernet.in/education/modules/dbms/SQL99/ansi-iso-9075-1-1999.pdf

9. Lic. Guillermo González Suárez. Msc. Ing. Rafael Cruz Iglesias. Ing. José Luis Capote Fernández. Lic. Osmani Herrera González. Tec. Liset Becerra Lugones. Sistema de Información Geográfica en web para la Telefonía Pública, basado en las especificaciones de OPENGIS. Julio del 2004 [citado: noviembre de 2008]; Disponible en: http://www.mappinginteractivo.com/plantilla.asp?id_articulo=668

10. Softel. Sistema de Gestión de Donantes de Sangre y Distribución de Sangre y Componentes Sanguíneos. [citado: noviembre de 2008]; Disponible en http://www.softel.cu/doc/Proyecto%20Red%20de%20Bancos%20de%20Sangre.doc

11. Embarcadero Technologies. Embarcadero ER/Studio.[citado: noviembre de 2008]; Disponible en: http://www.embarcadero.com

12. Grupo TRESS. Portal TRESS. [citado: noviembre de 2008];Disponible en: http://www.tress.com.mx/esp

13. Prof. Ruiz, Francisco, Universidad de Cantabria Facultad de Ciencias, Requisitos[citado: marzo de 2009]; Disponible en http://personales.unican.es/ruizfr/is1/doc/teo/03/is1-t03-trans.pdf

14. Hansen, Gary W; Hansen James V. Diseño y Administración de Bases de Datos. 2da Edición.

15. Msc. Pérez Fernández, Vicenta; Libro de Base de Datos[citado: abril de 2008]: Disponible en: http://www.ispcmw.rimed.cu/sitios/digbiblio/cont/SA/sgbd/libro_bd.pdf

16. Rational Unified Process 2001.

Bibliografía

- Marqués, Mercedes. Historia de los sistemas de bases de datos. 12 de febrero del 2001.[citado noviembre de 2008]; Disponible en: http://www3.uji.es/~mmarques/f47/apun/node6.html
- Hernández, José O. La Disciplina de los Sistemas de Bases de Datos, Historia, Situación Actual y Perspectivas .mayo 2002. [citado noviembre de 2008]; Disponible en: http://www.dsic.upv.es/~jorallo/docent/BDA/DisciplinaBD.pdf
- Hansen, Gary W; Hansen James V. Diseño y Administración de Bases de Datos. 2da Edición.
- Mato Gracia, Rosa M. Diseño de Bases de Datos, Octubre 2005.
- Date C. J. Introducción a los sistemas de bases de datos, Editorial Félix Varela, La Habana, 2003.
- Hernández Diez, Carmen. El Sistema Gestor de Bases de Datos. [citado: noviembre de 2008]; Disponible en: http://www.infor.uva.es/~chernan/Bases/Teoria/TEMA2.pdf
- Dr. Conejo Muñoz, Ricardo Tipos de Bases de Datos.[citado: noviembre de 2008];Disponible en: http://www.lcc.uma.es/~galvez/ftp/bdst/Tema2.pdf
- Dr. Cruz Chávez, Marco Antonio. MySQL. [citado noviembre de 2008]; Disponible en http://www.uaem.mx/posgrado/mcruz/cursos/miic/MySQL.pdf
- Zaninotto, François; Potencier Fabien. Symfony, La guía definitiva 1.1, Editorial Apress. Disponible en: http://www.librosweb.es/symfony/index.html.
- E.F. Codd. Modelo Relacional de Datos para grandes Bancos de Información , June 1970
- Microsoft Corporation. Administering a Microsoft SQL Server 2000 Database[citado enero 2009]; Disponible en : http://www.microsoft.com/learning/en/us/syllabi/2072afinal.mspx
- Maestros del WEB. El framework Symfony, una introducción práctica (I parte). [citado: noviembre de 2008]; Disponible en: http://www.maestrosdelweb.com/editorial/el-framework-symfony-una-introduccion-practica-i-parte/

- SWEBOK - Guide to the Software Engineering Body of Knowledge, 2004 Version. Disponible en http://personales.unican.es/ruizfr/is1/doc/teo/03/IEEE830.pdf

- Microsoft; Integridad de los datos,[citado: marzo de 2009];Disponible en: http://msdn.microsoft.com/es-es/library/ms184276.aspx

- C. J. Date; Sistemas de Base de Datos; Disponible en: http://www.google.com.cu/books?id=Vhum351T-K8C&printsec=frontcover&dq=intitle:Sistemas+intitle:de+intitle:Bases+intitle:de+intitle:Datos+inauthor:C.+J.+inauthor:Date&lr=&as_brr=0&as_pt=ALLTYPES

Glosario de Términos

- **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.

- **ManuGas:** Empresa de gas manufacturado, encargada de este servicio, actualmente solamente en Ciudad Habana.

- **Servidor:** 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.

- **Sistema Metrado:** Sistema utilizado en la empresa ManuGas.

- **IBM** (acrónimo de International Business Machines)**:** es una empresa que fabrica y comercializa herramientas, programas y servicios relacionados con la informática. Tiene su sede en Armonk (Estados Unidos).

- **Escalabilidad:** La capacidad del sistema informático de cambiar su tamaño o configuración para adaptarse a las circunstancias cambiantes.

- **Entidad:** representación de un objeto individual concreto del mundo real. Cualquier tipo de objeto o concepto sobre el que se recoge información puede ser una cosa, persona, concepto abstracto o suceso.

- **Disponibilidad**: nivel de servicio proporcionado por aplicaciones, servicios o sistemas. Los sistemas altamente disponibles tienen un tiempo de inactividad mínimo, ya sea previsto o no.

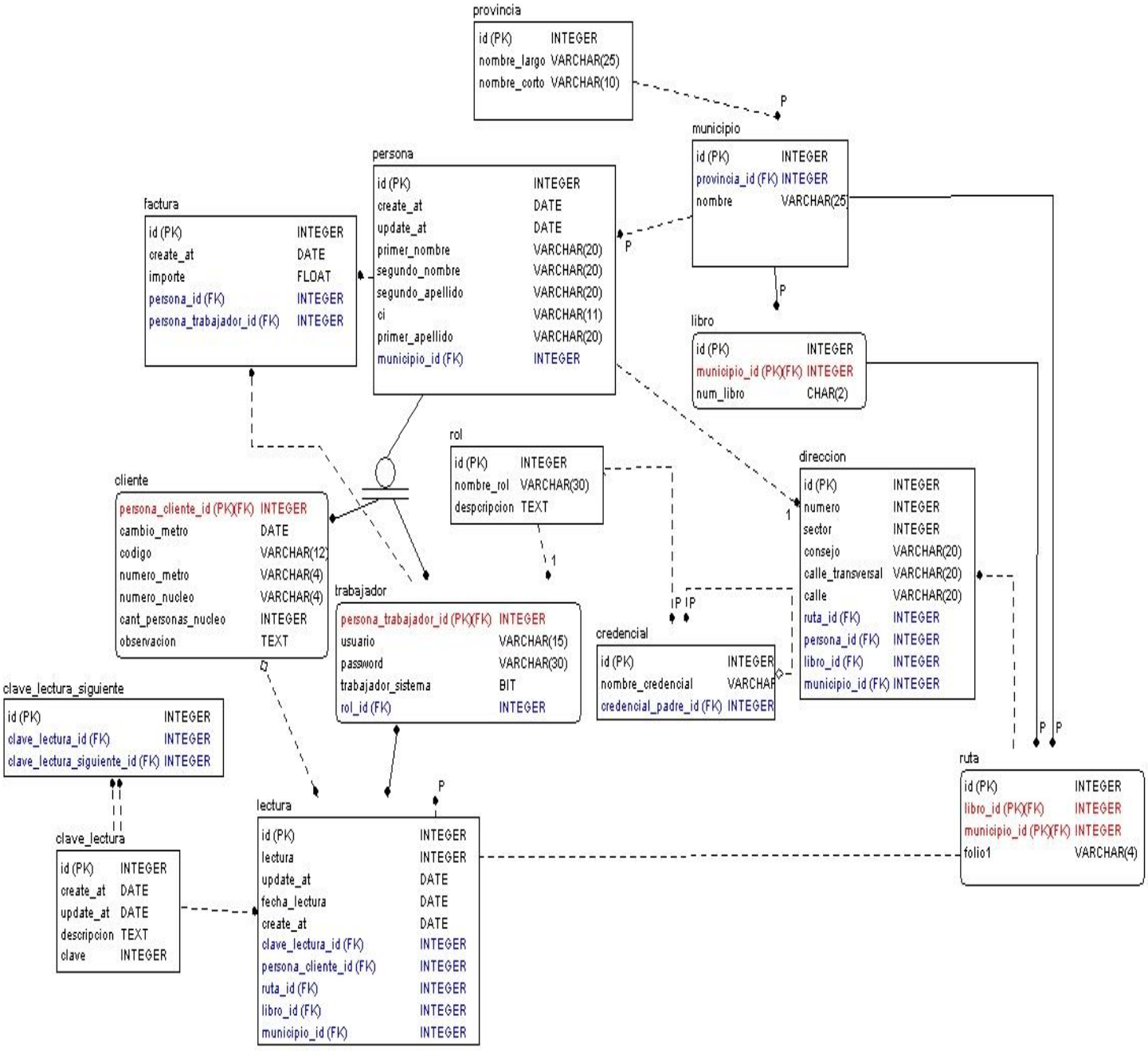
- **Framework:** en el desarrollo de software, es una estructura de soporte definida, mediante la cual otro proyecto de software puede ser organizado y desarrollado.

- **Redundancia:** Repetición de una información previamente existente.

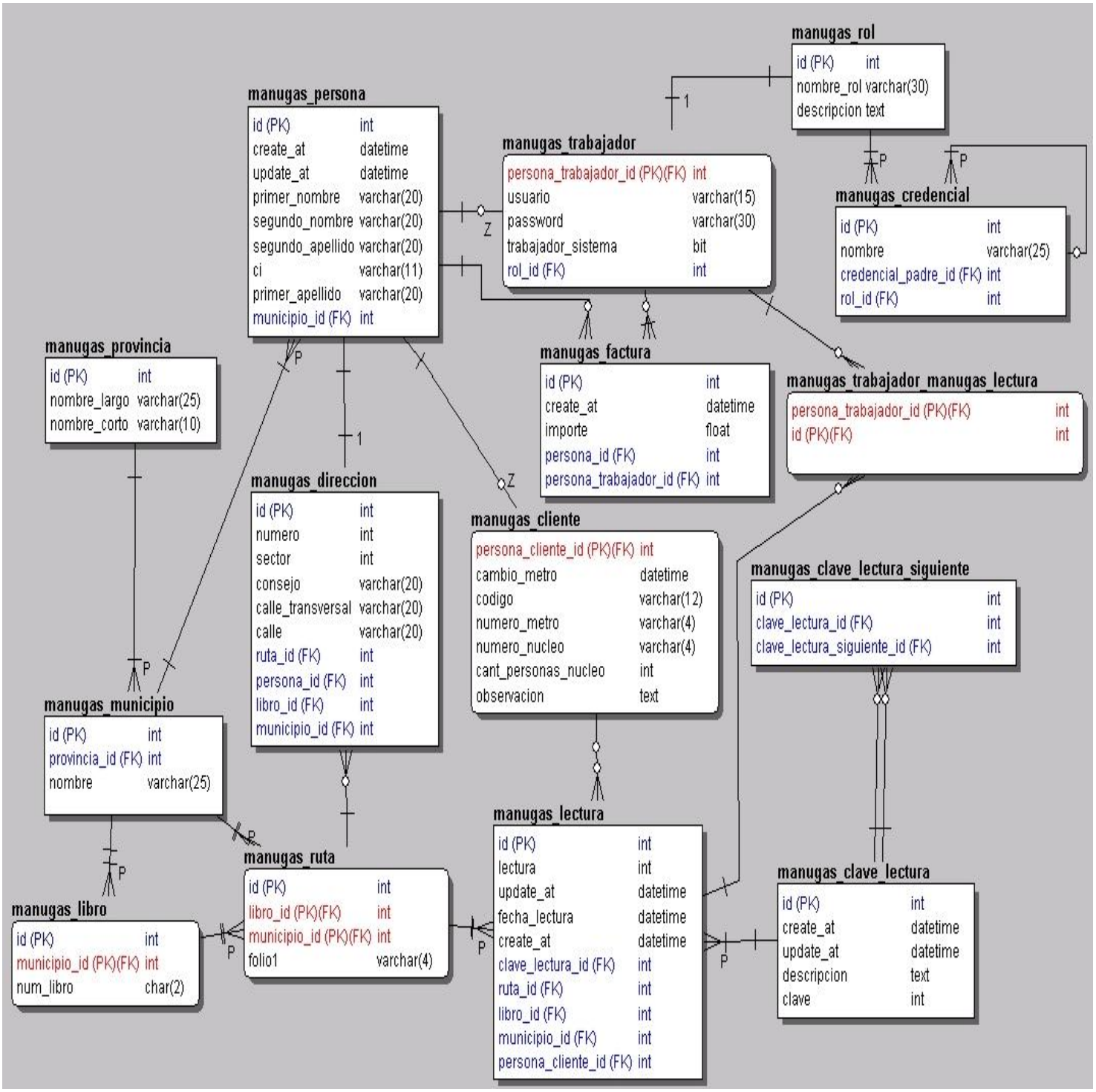
- **Trazabilidad:** Aptitud de reconstruir la historia, la utilización o la localización de un producto por medio de identificaciones registradas.
- **Transacciones:** Una transacción es un conjunto de procesos que se ejecutan uno después del otro. Si algún subproceso falla, lo realizado anteriormente debe reversarse para que los datos no se alteren, a este comportamiento se lo denomina todo o nada.
- **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.
- **Rol:** Papel desempeñado por las personas en la sociedad.
- **Normalización:** Proceso de reducción sobre una estructura de datos que procura aumentar la integridad, disminuir la redundancia y las dependencias funcionales de esa estructura.
- **Protocolo:** Un protocolo de comunicación es la manera de comunicarse que tiene una computadora con otra, cuando se están transmitiendo datos entre sí.

Anexos

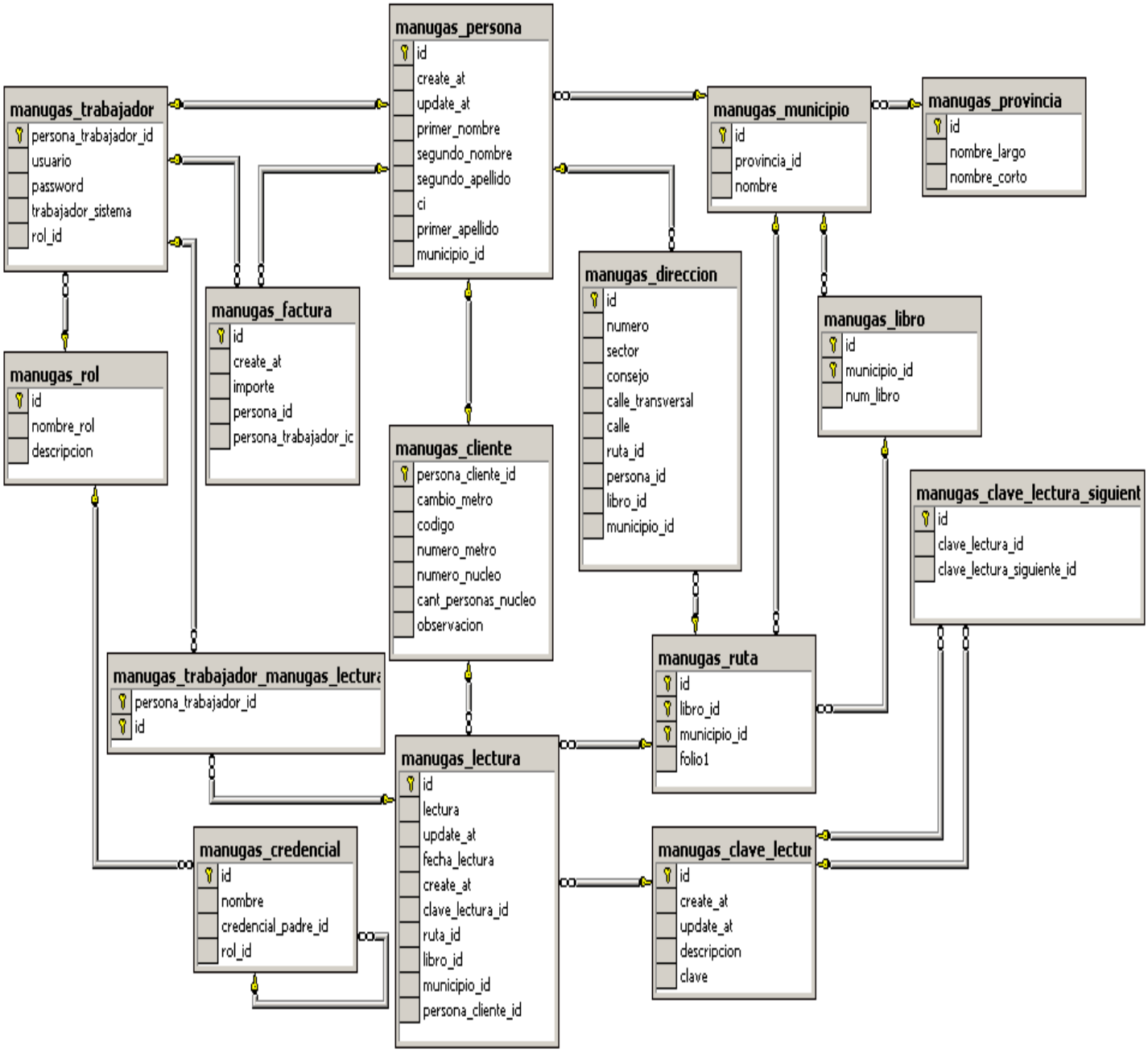
Anexo 1



Anexo 2


manugas_rol
id (PK) int
nombre_rol varchar(30)
descripcion text
manugas_persona
id (PK) int
create_at datetime
update_at datetime
primer_nombre varchar(20)
segundo_nombre varchar(20)
segundo_apellido varchar(20)
ci varchar(11)
primer_apellido varchar(20)
municipio_id (FK) int
manugas_trabajador
persona_trabajador_id (PK)(FK) int
usuario varchar(15)
password varchar(30)
trabajador_sistema bit
rol_id (FK) int
manugas_credencial
id (PK) int
nombre varchar(25)
credencial_padre_id (FK) int
rol_id (FK) int
manugas_provincia
id (PK) int
nombre_largo varchar(25)
nombre_corto varchar(10)
manugas_factura
id (PK) int
create_at datetime
importe float
persona_id (FK) int
persona_trabajador_id (FK) int
manugas_trabajador_manugas_lectura
persona_trabajador_id (PK)(FK) int
id (PK)(FK) int
manugas_direccion
id (PK) int
numero int
sector int
consejo varchar(20)
calle_transversal varchar(20)
calle varchar(20)
ruta_id (FK) int
persona_id (FK) int
libro_id (FK) int
municipio_id (FK) int
manugas_cliente
persona_cliente_id (PK)(FK) int
cambio_metro datetime
codigo varchar(12)
numero_metro varchar(4)
numero_nucleo varchar(4)
cant_personas_nucleo int
observacion text
manugas_clave_lectura_siguiente
id (PK) int
clave_lectura_id (FK) int
clave_lectura_siguiente_id (FK) int
manugas_municipio
id (PK) int
provincia_id (FK) int
nombre varchar(25)
manugas_lectura
id (PK) int
lectura int
update_at datetime
fecha_lectura datetime
create_at datetime
clave_lectura_id (FK) int
ruta_id (FK) int
libro_id (FK) int
municipio_id (FK) int
persona_cliente_id (FK) int
manugas_ruta
id (PK) int
libro_id (PK)(FK) int
municipio_id (PK)(FK) int
folio1 varchar(4)
manugas_clave_lectura
id (PK) int
create_at datetime
update_at datetime
descripcion text
clave int
manugas_libro
id (PK) int
municipio_id (PK)(FK) int
num_libro char(2)

Anexo 3


manugas_persona
id
create_at
update_at
primer_nombre
segundo_nombre
segundo_apellido
ci
primer_apellido
municipio_id
manugas_trabajador
persona_trabajador_id
usuario
password
trabajador_sistema
rol_id
manugas_municipio
id
provincia_id
nombre
manugas_provincia
id
nombre_largo
nombre_corto
manugas_factura
id
create_at
importe
persona_id
persona_trabajador_ic
manugas_direccion
id
numero
sector
consejo
calle_transversal
calle
ruta_id
persona_id
libro_id
municipio_id
manugas_libro
id
municipio_id
num_libro
manugas_rol
id
nombre_rol
descripcion
manugas_cliente
persona_cliente_id
cambio_metro
codigo
numero_metro
numero_nucleo
cant_personas_nucleo
observacion
manugas_clave_lectura_siguient
id
clave_lectura_id
clave_lectura_siguiente_id
manugas_trabajador_manugas_lectura
persona_trabajador_id
id
manugas_ruta
id
libro_id
municipio_id
folio1
manugas_lectura
id
lectura
update_at
fecha_lectura
create_at
clave_lectura_id
ruta_id
libro_id
municipio_id
persona_cliente_id
manugas_credencial
id
nombre
credencial_padre_id
rol_id
manugas_clave_lectur
id
create_at
update_at
descripcion
clave