Universidad de las Ciencias Informáticas
Facultad 9

Trabajo de diploma para optar por el titulo de ingeniero en ciencias informáticas.

Titulo: Diseño de la base de datos para el Grupo de Calidad de la facultad 9.

Autor: Yanly Suárez García.
Tutor: Ing. Yusdeny Pérez Mendoza.
Co-tutor: Msc. Asnioby Hernández López.

Ciudad de La Habana, Junio del 2010.
"Año del 52 Aniversario del Triunfo de la Revolución"

Declaración de Autoría

Declaro que yo Yanly Suárez García soy la única autora de este trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI) y a la Facultad (9) para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los ____ días del mes de junio del 2009.

_____	_____
Yanly Suárez García	Yusdeny Pérez Mendoza
Autora	Tutor

_____

Asnioby Hernández López
Co-tutor

"Es una locura seguir haciendo siempre lo mismo y esperar resultados diferentes"

Albert Einstein.

A mi mamá por ser lo más grande que tengo en la vida.

A mi abuela Martha por apoyarme tanto.

A mi novio Luis por su amor y dedicación.

Agradecimientos

A mi mamá por creer en mí, por su apoyo incondicional y su infinito amor, porque este es su sueño hecho realidad pues lo que soy se lo debo a ella. Muchas gracias mami…

A mi novio Luis por soportarme, por comprenderme, por preocuparse más que yo y animarme a seguir aun cuando yo decía que no podía, por darme tanta felicidad en estos últimos años de la carrera, porque si terminé esta tesis es gracias ti. Muchísimas gracias por todo mi amor.

A mis abuelos Martha y Fidel, en especial a mi abuela por siempre estar ahí para mí, por la educación que me diste durante toda mi vida y por quererme tanto. No te olvidaré jamás.

A mi prima Sule por quererme como una hermana y ayudarme tanto.

A mis tías Nina y Tata, mis tíos Lalo, Boby y el Tirro, a las esposas de mis tíos y a todos mis primos gracias por ser la familia maravillosa que son.

A mis suegros por apoyarme y hacerme sentir como una más de la familia, gracias.

A Emelina y Santana por ayudarme tanto cuando los necesité.

A mis amigas Freny, Dayami y Tatiana por apoyarme siempre de lejos y no olvidarse nunca de mí.

A Magdiel, Adrian, Lisbet y Daylenis, gracias por ser mi amigos.

A todo el tribunal por sus recomendaciones y la ayuda que me dieron.

Agradezco a todas las personas con las cuales he compartido estos maravillosos años, a la gente de mi grupo, a todos aquellos que alguna vez me preguntaron cómo me iba en la tesis, a los que me han ayudado de una forma u otra a realizar este sueño…

Muchas Gracias.

Resumen

El almacenamiento de los procesos de gestión de información en el Grupo de Calidad de la facultad 9 es de vital importancia, el mismo es realizado de forma manual, debido a que no existe un sistema informático que gestione eficientemente toda la documentación que se genera en las pruebas y revisiones que se le realizan a los proyectos. En su lugar se cuenta con un repositorio que proporciona muchas desventajas en cuanto a disponibilidad de la información y la forma de la almacenarla, así como no poder llevar el control sobre las tareas que se realizan diariamente en el proyecto.

Como solución a esta problemática se diseñó una base de datos para el Grupo de Calidad de la facultad 9 con el objetivo de lograr un almacenamiento estructurado y más seguro de la información. Durante el desarrollo de la investigación se estudiaron algunos conceptos importantes para el desarrollo de la misma, y algunas herramientas que fueron utilizadas para el diseño y administración de la base de datos. Finalmente se obtuvo un modelo que se ajusta a las necesidades del sistema, el cual ha sido validado teórica y funcionalmente teniendo en cuenta un conjunto de aspectos importantes.

Palabras claves: base de datos, calidad, gestión, información.

Índice

Índice de figuras

Índice de tablas

Introducción

La información siempre ha sido un recurso muy preciado, desde la antigüedad los datos se recopilaban, se estructuraban, se centralizaban y luego se almacenaban en bibliotecas u otros lugares donde se atesoraban. El objetivo de este proceso era poder recuperar esa información en algún momento determinado para hacer uso de ella o de algún dato asociado sin necesidad de volverlos a recopilar.

El flujo constante de información ha hecho necesaria la creación de herramientas que faciliten su uso, manejo y acceso. En una organización, por la variedad de documentos que se generan durante el desarrollo de sus actividades cotidianas, suelen presentarse problemas relacionados con el ordenamiento de datos en bases convencionales, debido a la falta de homogeneidad en la información que se quiere almacenar. Con el incremento del uso de las Tecnologias de la Información y las Comunicaciones (TIC) la automatización de los procesos que se realiza es cada vez mayor permitiendo ejecutar las actividades en menor tiempo y aportando mayor productividad.

Cuba no está ajena a esta realidad, en sectores como la Salud y la Educación ya se avanza en la informatización de sus procesos de negocio, por lo que se ha estimulado el desarrollo de la informática en el país. La Universidad de las Ciencias Informáticas (UCI) se crea en medio de este desarrollo como parte de la batalla de ideas, este centro productivo, cuya misión es producir software y servicios informáticos a partir de la vinculación estudio – trabajo como modelo de formación, es considerada la mayor productora de software en el país.

En cada una de las facultades de la universidad existe un conjunto de trabajadores y estudiantes organizados, que son los encargados de controlar la calidad de cada software que se desarrolla en los proyectos productivos, así nace el Grupo de Calidad de la facultad 9, el mismo para lograr una mayor organización en su labor se dividió en 4 áreas de trabajo que se resumen en: Procedimiento de Inicio de Proyecto, Auditorias y Revisiones, Recursos Humanos y Estrategias de Pruebas.
Toda la información que se genera en cada una de estas áreas se realiza de forma manual ya que el grupo no cuenta con un sistema capaz de gestionar toda esta documentación de manera efectiva. El medio de almacenamiento que existe es un repositorio lo que trae como consecuencia que la información

puede estar duplicada, se pueden perder archivos sin el conocimiento de los integrantes del área de trabajo, la labor a la hora de guardar y actualizar la información se hace muy engorrosa y la actualización se realiza mediante el correo electrónico. Esto conlleva a que la información no esté disponible en todo momento por lo que sea hace muy difícil mantener informados a todos los integrantes del grupo. Además los integrantes de un área determinada no tienen conocimiento de las actividades que se llevan a cabo en las áreas restantes porque no tienen acceso a esa información.

Después de analizar la situación problemática anteriormente planteada, surge el siguiente **problema a resolver** la dificultad en el almacenamiento de la información del Grupo de Calidad de la facultad 9.

Teniendo como o**bjeto de estudio** los procesos de gestión de la información del Grupo de Calidad de la facultad 9 y delimitando el **campo de acción** al diseño de la base de datos para los procesos de gestión de información del Grupo de Calidad de la facultad 9.

Para dar solución al problema identificado, se planteó como **objetivo general** del trabajo, desarrollar el diseño lógico y el modelo físico de la base de datos para el Grupo de Calidad de la facultad 9.

Asumiendo como **idea a defender** que con el diseño de una base de datos para el Grupo de Calidad de la facultad 9 se garantizará el almacenamiento organizado y no duplicado de los procesos de gestión de información.

Las **tareas de la investigación** propuestas para guiar la misma son las siguientes:

1. Caracterizar el estado actual de los procesos de gestión de información del Grupo de Calidad de la facultad 9.
2. Determinar los principales gestores de bases de datos utilizados a nivel nacional y en la UCI.
3. Diseñar la base de datos.
4. Administrar la seguridad de los datos.
5. Validar el diseño realizado.

Métodos de investigación:

Para realizar la investigación se utilizaron diferentes métodos de investigación científica, estos son:

Métodos teóricos:

Método histórico-lógico: En la investigación se analiza el desarrollo de las bases de datos y la importancia que se le atribuye al uso de las mismas en la actualidad, así como los principales modelos que existen para el diseño de bases de datos con el fin de determinar el más indicado para cubrir las necesidades del proyecto.

Método analítico-sintético: Durante la investigación se utilizó para comprender con más exactitud como es el proceso de almacenamiento en las bases de datos, así como la estructura que deben poseer los datos para lograr un trabajo más eficiente.

Métodos Empíricos:

Entrevista: En la investigación se aplicaron entrevistas a los jefes de cada una de las áreas de trabajo del grupo para así definir como se realizan los procesos de gestión de información y poder precisar la información que se almacenará en la base de datos. Se utilizó para una mejor realización del trabajo un muestreo intencional, no probabilístico, utilizando como población para la entrevista los profesores que trabajan con el Grupo de Calidad, de un total de 9 se escogieron los 4 que se encuentran frente a cada una de las áreas de trabajo.

Capítulo I: Fundamentación teórica

1.1 Introducción

Con el desarrollo de este capítulo se presentan un grupo de conceptos a los que se hará referencia en el resto del trabajo, se exponen los temas relacionados con el campo de acción para su mejor compresión. Se abordan aspectos relacionados con las bases de datos, tales como su definición, arquitectura y tipos considerando que actualmente el uso de las bases de datos es fundamental en diferentes esferas de la vida social. Todo este análisis se realiza teniendo en cuenta las necesidades y características del medio donde se aplicará la solución propuesta.

1.2 Conceptos asociados al campo de acción

1.2.1 Gestión

Gestión del latín *gestio: acción* de administrar. Es la actividad profesional destinada a establecer los objetivos y medios para la realización, organización del sistema, elaboración de la estrategia de desarrollo y ejecución de la gestión del personal. El término gestión, por lo tanto, implica al conjunto de trámites que se llevan a cabo para resolver un asunto o concretar un proyecto, se refiere al conjunto de actividades que desarrollan, movilizan y motivan al personal empleado que una empresa necesita para su éxito.

1.2.2 Proceso

La palabra proceso tiene su origen del latín: *processus*. De acuerdo al diccionario de la Real Academia Española (RAE), el concepto hace referencia a la acción de ir hacia adelante, al transcurso del tiempo, al conjunto de las fases sucesivas de un fenómeno natural o de una operación artificial.

Hablar de análisis del proceso, se refiere a las diferentes etapas que componen de una manera ordenada y escalonada la realización de una tarea. En informática, cuando se habla de proceso se refiere a un concepto que se maneja dentro del ámbito de los sistemas operativos, en este contexto, se refiere a las instrucciones que ejecutará el microprocesador mientras lee un programa determinado. Esto también implica a la memoria reservada y a sus contenidos, el estado de ejecución en determinado momento, y la información que permite al sistema operativo planificar. A modo de resumen un proceso no es más que el

conjunto de fases que componen la realización de las actividades tanto en la vida cotidiana como en cualquiera de las ciencias sociales.

1.2.3 Información

La información es un conjunto organizado de datos, que constituye un mensaje sobre un cierto fenómeno. La información permite resolver problemas y tomar decisiones, ya que su uso racional es la base del conocimiento**.** Esta es un fenómeno que aporta significado o sentido a las cosas, ya que mediante códigos y conjuntos de datos, forma los modelos de pensamiento humano**.**

En términos generales, se habla de la información como un conjunto de datos que están organizados y que tienen un significado. De esta manera, si se toman los datos por separado no tendrían un significado mientras que si se agrupan en forma organizada, si. La información es un elemento fundamental en el proceso de la comunicación, porque tiene un significado para quien la recibe, que la va a comprender si comparte el mismo código de quien la envía.

1.2.4 Gestión de la información

La gestión de la información es el proceso que se encarga de suministrar los recursos necesarios para la toma de decisiones, así como para mejorar los procesos, productos y servicios de la organización (6). La gestión de la información no es más que el proceso de analizar y utilizar la información que se ha conseguido y registrado para permitir tomar decisiones documentadas. La información para la gestión es la información necesaria para tomar decisiones de gestión.

Por lo tanto, la gestión de la información implica:

- Determinar la información que se precisa.
- Recoger y analizar la información.
- Registrarla y recuperarla cuando sea necesaria.
- Utilizarla.

1.2.5 Bases de Datos

El término bases de datos fue escuchado por primera vez en 1963, en un simposio celebrado en California, Estados Unidos. Una base de datos se puede definir como un conjunto de información relacionada que se encuentra agrupada ó estructurada.

Desde el punto de vista informático, la base de datos es un sistema formado por un conjunto de datos almacenados en discos que permiten el acceso directo a ellos y un conjunto de programas que manipulen ese conjunto de datos. Cada base de datos se compone de una o más tablas que guarda un conjunto de datos. Cada tabla tiene una o más columnas y filas. Las columnas guardan una parte de la información sobre cada elemento que se quiere guardar en la tabla, cada fila de la tabla conforma un registro.

Características:

- Independencia lógica y física de los datos.
- Redundancia mínima.
- Acceso concurrente por parte de múltiples usuarios.
- Integridad de los datos.
- Seguridad de acceso y auditoría.
- Acceso a través de lenguajes de programación estándar.

En el epígrafe 1.4 se hace una descripción más detallada del concepto de base de datos.

1.3 Procesos de gestión de información del Grupo de Calidad de la facultad 9

1.3.1 Procedimiento de Inicio de Proyecto

Un proyecto es esencialmente un conjunto de actividades interrelacionadas, con un inicio y una finalización definida, que utiliza recursos limitados para lograr un objetivo deseado. En este caso los dos elementos fundamentales de un proyecto son las actividades y los recursos. Las actividades son las tareas que se ejecutan para llegar a un objetivo deseado, los recursos son los elementos utilizados para realizar cada una de estas tareas.

Los integrantes del área de Procedimiento de Inicio de Proyecto tienen como objetivo: garantizar un nivel de calidad básico homogéneo en todos los proyecto de la facultad 9 como principal requerimiento para su certificación. Para que un proyecto sea certificado debe realizar todas las actividades y tener bien definido las valoraciones económicas. El significado de que un proyecto sea certificado es que obtiene la autorización para comenzar a desarrollar el producto que tenían previsto.

1.3.2 Auditorias y Revisiones

Una auditoria en calidad es un examen metódico e independiente, que se realiza para determinar si las actividades y los resultados relativos a la calidad cumplen las disposiciones previamente establecidas y si estas disposiciones se llevan a cabo de forma efectiva y son adecuadas para alcanzar los objetivos establecidos.(ISO 8402) .

Por lo tanto una auditoria es una herramienta que permite evaluar la eficacia de un sistema determinado, sus defectos y los medios de mejora. El área encargada de las auditorias y revisiones tiene como función fundamental revisar los proyectos basándose en lo establecido por la dirección de calidad, cuando se realiza una auditoria a un proyecto primeramente se le notifica al líder del mismo y luego se realiza una reunión de apertura donde se establecen las condiciones y especificaciones, así como la información de interés para todos los implicados.

El proyecto debe facilitar la documentación que tiene hasta la fecha, se debe revisar que el expediente de proyecto cumpla con el formato establecido y a partir de aquí se revisa cada una de las planillas según la fase en la que se encuentra el proyecto, al final de cada auditoria se realiza una lista de no conformidades encontradas y esta se le entrega al jefe del proyecto auditado en una reunión de cierre, esto debe ser informado al grupo de trabajo y los errores deben ser erradicados.

1.3.3 Recursos humanos

Los recursos humanos son todas las personas con que cuenta una empresa u organización. En el Grupo de Calidad se cuenta con un área encargada de registrar los datos de todos los estudiantes y profesores, así como la cantidad de computadoras que existen y la distribución de cada una de estas personas en las estaciones de trabajo.

1.3.4 Estrategia de Prueba

En cierta manera la estrategia de pruebas vela porque el producto de software que se está construyendo o modificando, reúna los requerimientos de lógica del negocio que el cliente ha pedido realizar mediante el debido contrato de desarrollo de software. El objetivo fundamental de las pruebas es detectar los errores que presenta el software que se está construyendo, las pruebas se deben realizar en cada iteración para de esta forma poder eliminar lo antes posible los defectos encontrados.

¿Qué es una estrategia de pruebas?
Son las líneas guía del equipo de pruebas de un proyecto de desarrollo de software. Estas son escritas por el diseñador de pruebas, en un documento, que será entregado formalmente al director del proyecto, para su posterior revisión y aprobación. Se debe tener claridad en que dicho documento no debe ser visto como un entregable más, éste debe ser visto como un artefacto especialmente creado para reunir las ideas más representativas del proceso de pruebas que se llevará a cabo.

El grupo de Estrategia de Pruebas tiene como objetivo: definir un procedimiento para homogenizar las pruebas en los proyectos productivos de la facultad 9 además este define cómo se efectuará el esfuerzo de prueba contra uno o más aspectos de los sistemas que se desarrollan en los diferentes proyectos.

Roles de pruebas de software:

- Diseñador de pruebas.
- Gestor de prueba.
- Analista de pruebas.
- Verificador.

Estos roles pueden ser desempeñados tanto por profesores como estudiantes que estén a cargo de las pruebas de software dentro del proyecto.

Diseñador de pruebas: Este rol dirige la identificación de las técnicas, herramientas y directrices apropiadas para implementar las pruebas necesarias y proporciona orientación sobre los requisitos a cumplir y los recursos necesarios.

Gestor de prueba: Este rol dirige la planificación y gestión de recursos para la realización de las pruebas.

Analista de pruebas: Este rol identifica y define las pruebas necesarias, supervisa el proceso de prueba, los resultados de cada ciclo de prueba y evalúa la calidad global. El rol también representa a los interesados que no tienen una representación directa o regular en el proyecto.

Verificador: Este rol realiza pruebas y registra los resultados.

1.4 Introducción a las bases de datos

Los datos constituyen una parte esencial en un sistema de información, para organizar y gestionar estos datos en el computador se han desarrollado técnicas cuya evolución ha estado determinada por el desarrollo de las tecnologías de las computadoras. Las bases de datos se han convertido en una de las herramientas más difundidas actualmente en el mundo de la información, siendo utilizadas como fuentes de almacenamiento y recuperación de información en todos los campos a nivel social, científico, económico y político.

Sobre las bases de datos existen diferentes criterios:

- Una base de datos es un conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, que una base de datos puede considerarse una colección de datos variables en el tiempo (1).
- Una base de datos se puede considerar como una especie de armario electrónico para archivar; es decir, es un depósito o contenedor de una colección de archivos de datos computarizados (2).
- Un conjunto de datos almacenados entre los que existen relaciones lógicas y ha sido diseñada para satisfacer los requerimientos de información de una empresa u organización. En una base de datos, además de los datos, también se almacena su descripción (3).

A modo general una base de datos es un conjunto de datos relacionados entre sí que comparten características similares, organizados y estructurados, los cuales son coleccionados y explotados por los sistemas de información de una empresa o negocio en particular.

Una base de datos cuenta con varios componentes:

- **Hardware:** Se refieren a los dispositivos de almacenamiento donde se encuentra la base de datos, así como los periféricos asociados para su utilización como son los controladores de dispositivos, unidades de discos entre otros.

- **Software**: Está constituido por un conjunto de programas que se conoce como Sistema Gestor de Base de Datos (SGBD) que es el encargado de manejar cada una de las solicitudes realizadas por los usuarios.

- **Usuario:** Existen tres grandes grupos de usuarios relacionados con las bases de datos.

  - Programador de aplicaciones: se encarga de crear programas de aplicación que se utilizan en la base de datos.

  - Usuarios finales: acceden a la base de datos por medio de un lenguaje de consultas o un programa de aplicaciones.

  - Administrador de la base de datos: es el encargado del control general del SGBD.

1.5 Arquitectura de una base de datos

La forma de almacenamiento y organización de los datos no tiene que ser del conocimiento de los usuarios, por este motivo las bases de datos debe presentar los datos de forma que los usuarios puedan interpretarlos y modificarlos. La arquitectura ANSI/SPARC (American National Standard Institute - Standards Planning and Requirements Committee) propone tres niveles generales: interno, lógico global y externo.

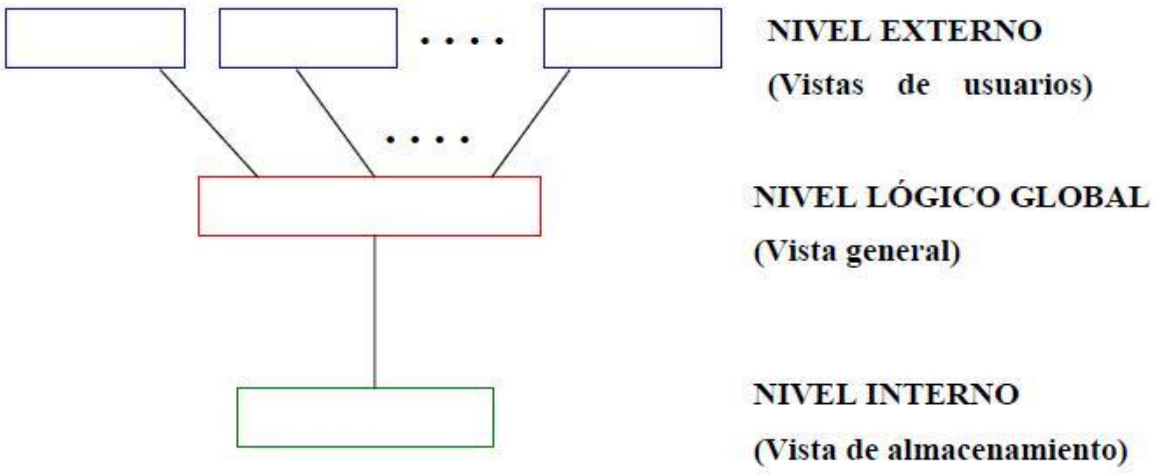

Fig. 1. Arquitectura de un sistema de base de datos (1).

- **El nivel externo:** Es el nivel más cercano a los usuarios, o sea, es el relacionado con la forma en que los datos son vistos por cada usuario individualmente.
- **El nivel lógico global:** Es el nivel medio de abstracción. Se trata de la representación de los datos realizada por la organización, que recoge las vistas parciales de los requerimientos de los diferentes usuarios y las aplicaciones posibles. Se configura como visión organizativa total, e incluye la definición de datos y las relaciones entre ellos.
- **El nivel interno:** Es el nivel más cercano al almacenamiento físico, o sea, es el relacionado con la forma en que los datos están realmente almacenados.

1.6 Fases de diseño de bases de datos

Con el desarrollo de las bases de datos han evolucionado también las metodologías y técnicas de diseño. El diseño de una base de datos se encuentra dividido en tres fases: el diseño conceptual, el diseño lógico y el diseño físico.

En el **diseño conceptual** se debe construir un esquema de la información del entorno que usará la base de datos. A esto se le denomina *esquema conceptual*, al construir el esquema, el diseñador descubre el

significado de los datos del entorno: encuentra entidades, atributos y relaciones. El objetivo es comprender el punto de vista que cada usuario tiene de los datos, su naturaleza y su uso a través de las áreas de aplicación. El esquema conceptual se construye utilizando la información que se encuentra en la especificación de los requisitos de usuario. El diseño conceptual es completamente independiente de los aspectos de implementación, como puede ser el (SGBD) que se vaya a usar, el hardware disponible o cualquier otra consideración física. El esquema conceptual es una fuente de información para el diseño lógico de la base de datos.

En el **diseño lógico** es el proceso de construir un esquema de la información basándose en un modelo de base de datos específico, independiente del SGBD concreto que se vaya a utilizar y de cualquier otra consideración física. En esta etapa, se transforma el esquema conceptual en un esquema lógico que utilizará las estructuras de datos del modelo de base de datos en el que se basa el SGBD que se vaya a utilizar, como puede ser el modelo relacional, el modelo de red, el modelo jerárquico o el modelo orientado a objetos. El esquema lógico es una fuente de información para el diseño físico. Además, juega un papel importante durante la etapa de mantenimiento del sistema, ya que permite que los futuros cambios que se realicen sobre los programas de aplicación o sobre los datos, se representen correctamente en la base de datos.

El **diseño físico** es el proceso de producir la descripción de la implementación de la base de datos, las estructuras de almacenamiento y los métodos de acceso que garanticen un camino eficiente a los datos. Para llevar a cabo esta etapa, se debe haber decidido cuál es el SGBD que se va a utilizar, ya que el esquema físico se adapta a él. Entre el diseño físico y el diseño lógico hay una retroalimentación, debido a que algunas de las decisiones que se tomen durante el diseño físico para mejorar las prestaciones, pueden afectar a la estructura del esquema lógico. El propósito del diseño físico es describir cómo se va a implementar físicamente el esquema lógico obtenido en la fase anterior.

1.7 Clasificación de las bases de datos

Las bases de datos de pueden clasificar según el modo en que estas almacenan la información o sus características. Algunas de estas clasificaciones se muestran a continuación.

1.7.1 Según la variabilidad de los datos almacenados

Base de Datos Estáticas: Son bases de datos de solo lectura, son utilizadas para almacenar datos históricos utilizados posteriormente para el estudio del comportamiento de un conjunto de datos a través del tiempo.

Base de Datos Dinámicas: La información que se almacena en la base de datos se modifica con el tiempo, permitiendo operaciones de actualización y adición en la misma además de las operaciones generales de consultas.

1.7.2 Según el contenido que almacenan

Bases de Datos Bibliográficas: Solo contienen un representante de la fuente primaria que permita localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, edición, titulo, etc. Puede contener un resumen pero nunca el texto completo, como lo indica su nombre el contenido son cifras o números.

Bases de Datos de texto completo: Almacenan las fuentes primarias, como por ejemplo, el contenido de todas las ediciones de una revista científica.

Directorios: Este es el caso de directorios digitales con los números de los usuarios, ya sean privados a estatales en el que se puede encontrar el nombre de dicho usuario y la dirección del mismo. Un ejemplo son las guías telefónicas en formato digital.

1.8 Modelos de bases de datos

Las bases de datos también se pueden clasificar en cuanto al modelo de administración de sus datos. Un modelo de datos es una descripción del contenedor de datos, así como los métodos que se utilizan para el almacenamiento y recuperación de la información. No constituyen elementos físicos sino que permiten la implementación eficiente de un sistema de base de datos. Los modelos de bases de datos se pueden dividir en 2 categorías principales.

1.8.1 Modelos lógicos basados en objeto

Se usan para describir los datos en los niveles conceptual y de visión (4). Los más conocidos son:

1.8.1.1 Modelo Entidad-Relación (ER)

Fue propuesto por Peter Chen en 1976 y desde entonces se viene utilizando de una forma muy global. Está basado en una percepción del mundo real que consta de un conjunto de objetos básicos llamados *entidades* con sus *atributos* y de las *relaciones* que existen entre estos objetos. Se desarrolló para facilitar el diseño de bases de datos permitiendo la especificación de un esquema del universo de discurso que representa la estructura completa de una base de datos. Este modelo es extremadamente útil para hacer corresponder los significados e interacciones del desarrollo del mundo real con un esquema conceptual. Los esquemas de modelos ER usan diagramas para representar la estructura natural de los datos.

Entidad

Una entidad es cualquier tipo de objeto o concepto sobre el que se recoge información: cosa, persona, concepto abstracto o suceso. Las entidades son descritas mediante atributos, si las mismas pueden ser descritas por el mismo tipo de atributos son consideradas como conjuntos y referidas como un conjunto de entidades. El nombre de entidad sólo puede aparecer una vez en el esquema conceptual. Hay dos tipos de entidades: fuertes y débiles. Una *entidad débil* es una entidad cuya existencia depende de la existencia de otra entidad. Una *entidad fuerte* es una entidad que no es débil.

Relación

Es una correspondencia o asociación entre dos o más entidades. Cada relación tiene un nombre que describe su función. Las entidades que están involucradas en una determinada relación se denominan *entidades participantes*. El número de participantes en una relación es lo que se denomina el *grado* de la relación. Una *relación recursiva* es una relación donde la misma entidad participa más de una vez en la relación con distintos papeles. El nombre de estos papeles es importante para determinar la función de cada participación.

La *cardinalidad* con la que una entidad participa en una relación especifica el número mínimo y el número máximo de correspondencias en las que puede tomar parte cada ocurrencia de dicha entidad. La participación de una entidad en una relación es obligatoria si la existencia de cada una de sus ocurrencias requiere la existencia de, al menos, una ocurrencia de la otra entidad participante. Si no, la participación es opcional. Las reglas que definen la cardinalidad de las relaciones son las reglas de negocio.

Atributo

Es una característica de interés o un hecho sobre una entidad o sobre una relación. Los atributos representan las propiedades básicas de las entidades y de las relaciones. Toda la información extensiva es portada por los atributos. Cada atributo tiene un conjunto de valores asociados denominado *dominio*. El dominio define todos los valores posibles que puede tomar un atributo. Puede haber varios atributos definidos sobre un mismo dominio.

Los atributos pueden ser simples o compuestos. Un *atributo simple* es un atributo que tiene un solo componente, que no se puede dividir en partes más pequeñas que tengan un significado propio. Un *atributo compuesto* es un atributo con varios componentes, cada uno con un significado por sí mismo.

Los atributos también pueden clasificarse en *monovaluados* o *multivaluados*. Un *atributo monovaluado* es aquel que tiene un solo valor para cada ocurrencia de la entidad o relación a la que pertenece. Un *atributo multivaluado* es aquel que tiene varios valores para cada ocurrencia de la entidad o relación a la que pertenece.

Por último, los atributos pueden ser derivados. Un *atributo derivado* es aquel que representa un valor que se puede obtener a partir del valor de uno o varios atributos, que no necesariamente deben pertenecer a la misma entidad o relación.

1.8.1.2 Modelo orientado a objetos

Es uno de los modelos más recientes y propio de los modelos informáticos orientados a objetos que trata de almacenar en la base de datos los objetivos completos (estado y comportamiento). Estas bases de datos incorporan todos los conceptos importantes del paradigma de objetos: encapsulamiento (es la propiedad que permite ocultar la información del resto de los objetos), herencia (es la propiedad mediante la cual los objetos heredan el comportamiento dentro de una jerarquía de clases) y el polimorfismo (es la propiedad mediante la cual una operación puede ser aplicada a diferentes tipos de objetos). Aquí los usuarios pueden definir operaciones sobre los datos.

1.8.2 Modelos lógicos basados en registros

Se utilizan para describir los datos en los niveles conceptual y físico. Además, para especificar la estructura lógica global de la base de datos y para proporcionar una descripción al nivel más alto de la implementación (4).

1.8.2.1 Modelo relacional

Es el modelo de base de datos más utilizado actualmente para modelar problemas reales y administrar datos dinámicos. Su idea fundamental es el uso de las relaciones que podrían considerarse en forma lógica como un conjunto de datos llamados ¨tuplas¨. En este modelo la forma de almacenamiento de los datos no es relevante lo cual tiene como ventaja que es más fácil de entender y utilizar por un usuario ocasional de la base de datos. La información puede ser almacenada o recuperada mediante consultas que proporcionan una amplia flexibilidad y poder para administrar la información. Durante su diseño las bases de datos relacionales pasan por un proceso conocido como normalización de una base de datos.

1.8.2.2 Modelo de datos en red

Su gran diferencia con el modelo jerárquico es la modificación del concepto de nodo, en este caso un mismo nodo puede tener varios padres. Este modelo fue una gran mejora con respecto al jerárquico ya que proporcionaba una solución eficiente al problema de la redundancia, pero la dificultad que significa almacenar la información en una base de datos de red ha significado que sea utilizado por los programadores más que por los usuarios finales.

1.8.2.3 Modelo jerárquico

Como su nombre lo indican estas bases de datos almacenan la información en una estructura jerárquica. En este modelo los datos se organizan de forma similar a un árbol en donde un nodo padre de información puede tener a varios nodos hijos. El nodo que no tiene padre es conocido como raíz, y los nodos hijos como hojas. Estas bases de datos son especialmente útiles para aplicaciones que manejan un gran volumen de información y datos muy compartidos permitiendo crear estructuras estables y de gran rendimiento. Su gran limitante es la imposibilidad que tienen de representar la redundancia.

1.9 Sistema Gestor de Bases de Datos (SGBD)

Existen varias definiciones sobre los SGBD, algunas se muestran a continuación:

- El software que permite la utilización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez es lo que se conoce como un Sistema Gestor de Base de Datos (SGBD) (1).
- Un Sistema de Gestión de Bases de Datos consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a esos datos. El Objetivo primordial de un SGBD es proporcionar un entorno que sea a la vez conveniente y eficiente para ser utilizado al extraer y almacenar información de la base de datos (4).
- "El sistema de gestión de la base de datos es una aplicación que permite a los usuarios definir, crear y mantener la base de datos, y proporciona acceso controlado a la misma." (5).

Un SGBD no es más que un tipo de aplicación informática o software que actúa como interfaz entre el usuario y la base de datos. Funciona como un intermediario para que el usuario pueda comunicarse con la base de datos en términos abstractos y de una forma práctica y eficiente, abstrayéndolo del modo físico de almacenamiento de la información en la computadora, y del método de acceso empleado, permitiéndole definir, crear y mantener la base de datos, y proporcionándole el acceso controlado a la misma.

1.9.1 Objetivos de los SGBD:

- **Independencia de los datos:** Capacidad de modificar el esquema físico o lógico de una base de datos sin la necesidad de realizar cambios en las aplicaciones que se sirven de ellas.
- **Minimización de la redundancia:** Un buen diseño de una base de datos lograra evitar la aparición de información repetida o redundante. Lo ideal es lograr una redundancia nula aunque en algunos casos la complejidad de los cálculos hace necesaria la aparición de redundancia.
- **Integración y sincronización de las bases de datos:** La integración consiste en garantizar una respuesta a los requerimientos de diferentes aspectos de los mismos datos por diferentes usuarios, de forma que, aunque el sistema almacene la información con cierta estructura y cierto tipo de representación, debe garantizar entregar al programa de aplicación datos que solicita y en la forma en que lo solicita. Está vinculada a la sincronización, que consiste en la necesidad de garantizar el acceso múltiple y simultáneo a la base de datos, de modo que los datos puedan ser compartidos por diferentes usuarios a la vez. Están relacionadas, ya que lo usual es que diferentes

usuarios trabajen con diferentes enfoques y requieran los mismos datos, pero desde diferentes puntos de vista.

- **Integridad de los datos:** Consiste en garantizar la no contradicción entre los datos almacenados de modo que, en cualquier momento del tiempo, los datos almacenados sean correctos, es decir, que no se detecte inconsistencia entre los datos. Se trata de proteger los datos almacenados ante fallos de hardware, datos entrados por usuarios descuidados o ante cualquier situación que pueda corromper la información.
- **Seguridad y recuperación:** Su objetivo es garantizar el acceso autorizado a los datos, garantizando que esta información este siempre asegurada ante cualquier intento de acceso no autorizado, ya sea por error del usuario o por mala intención. Normalmente los SGBD cuentan con un complejo sistema de permisos de usuarios, de grupos de usuarios permitiendo así otorgar categorías de permisos. Debe disponer de métodos que garanticen la restauración de la base de datos al producirse alguna falla técnica, interrupción de la energía eléctrica, etc.
- **Facilidad de manipulación de la información:** El SGBD debe contar con la capacidad de una búsqueda rápida por diferentes criterios, permitiendo que los usuarios planteen sus necesidades de forma simple.
- **Control centralizado:** Este es uno de los objetivos principales de los SGBD, permite controlar de manera sistemática y única los datos que se almacenan en la base de datos así como el acceso a ella. Por lo que debe existir el administrador de la base de datos que puede considerarse parte integrante del SGBD.

1.9.2 Principales Gestores de Bases de Datos

El desarrollo que ha experimentado el software de base de datos ha sido extraordinario por la facilidad de manejar la información, suministrando a los usuarios las herramientas necesarias que le permitan manipular de forma abstracta los datos con la existencia de múltiples entornos de desarrollo y soluciones informáticas.

Los SGBD son un software específico dedicado a servir de interfaz entre el usuario, la base de datos y las aplicaciones que la utilizan. Dentro de los SGBD más potentes de software propietario se encuentran Oracle y Microsoft SQL Server, y dentro del software libre esta PostgreSQL. En el caso de MySQL presenta el uso de una *licencia dual* o doble licenciamiento.

1.9.2.1 Sistema Gestor de Bases de Datos Oracle

Es un sistema de administración de bases de datos, básicamente una herramienta cliente/servidor para la gestión de bases de datos creado por Oracle Corporation. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hacen que sólo se vea en empresas muy grandes y multinacionales, por norma general. Es considerado como uno de los sistemas de bases de datos más completos destacando su: soporte de transacciones, estabilidad y escalabilidad. La tecnología Oracle actualmente se encuentra en casi todas la industrias alrededor del mundo, esta es la primera compañía que desarrolla software para empresas que tienen cien por ciento activado por Internet. Las últimas versiones de Oracle han sido certificadas para poder trabajar bajo GNU/Linux debido a la gran competencia que sufre hoy con las ofertas del software libre.

Características:

- Es multiplataforma.
- Un aceptable soporte.
- Alta seguridad.
- Su elevado precio.
- Las licencias Personal Oracle, son excesivamente caras.
- La necesidad de ajustes, una mala configuración de Oracle puede resultar desesperadamente lento.

1.9.2.2 Sistema Gestor de Bases de Datos Microsoft SQL Server

Es un sistema de gestión de base de datos relacionales basado en el lenguaje Transact-SQL, este gestor es capaz de poner a disposición de muchos usuarios de forma simultánea grandes cantidades de datos. Es nombrado SQL porque hace uso de este lenguaje para la definición y manejo de los datos y Server porque cuenta con una parte servidora que es responsable de atender a los procesos clientes, que son los que realizan peticiones al servidor.

Características:

- Escalabilidad, estabilidad. seguridad.
- Soporta procedimientos almacenados.

- Incluye un potente entorno gráfico de administración.
- Permite administrar información de otros servidores de datos.
- Permite trabajar en modo cliente-servidor.
- Es un SGBD propietario.
- Las funciones definidas por el usuario tienen algunas restricciones.

1.9.2.3 Sistema Gestor de Bases de Datos MySQL

Es un SGBD relacional, uno de los más usados en el mundo debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración. Desde enero de 2008 una subsidiaria de la Sun Microsystems desarrolla MySQL como software libre bajo un esquema de licenciamiento dual. Por un lado se ofrece bajo licencia GPL para cualquier uso compatible con esta licencia y por otro las empresas que quieran incorporarlos en productos privativos deben comprar una licencia específica que les permita este uso.

Características:

- Integración perfecta con el lenguaje PHP.
- Mejor control de acceso de usuarios.
- Gran portabilidad entre sistemas.
- No tiene límites en los tamaños de los registros.
- No es viable para su uso con grandes bases de datos, a las que se acceda continuamente, ya que no implementa una buena escalabilidad.

1.9.2.4 Sistema Gestor de Bases de Datos PostgreSQL

Es un sistema de bases de datos relacional de software libre que está catalogado como la mejor y más avanzada base de datos Open Source del mundo. Iniciado como un proyecto universitario en la década de 1980, ha llevado varios períodos de desarrollo y cuenta con una arquitectura confiable,

estabilidad de procesamiento e integridad en el manejo de datos. Es un verdadero proyecto de software abierto, ya que su desarrollo no es dirigido por una empresa sino que es administrado por una comunidad.

PostgreSQL es un sistema relacional orientado a objetos, ya que incluye características como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Es liberado bajo la licencia BSD, la cual permite generar versiones comerciales del producto.

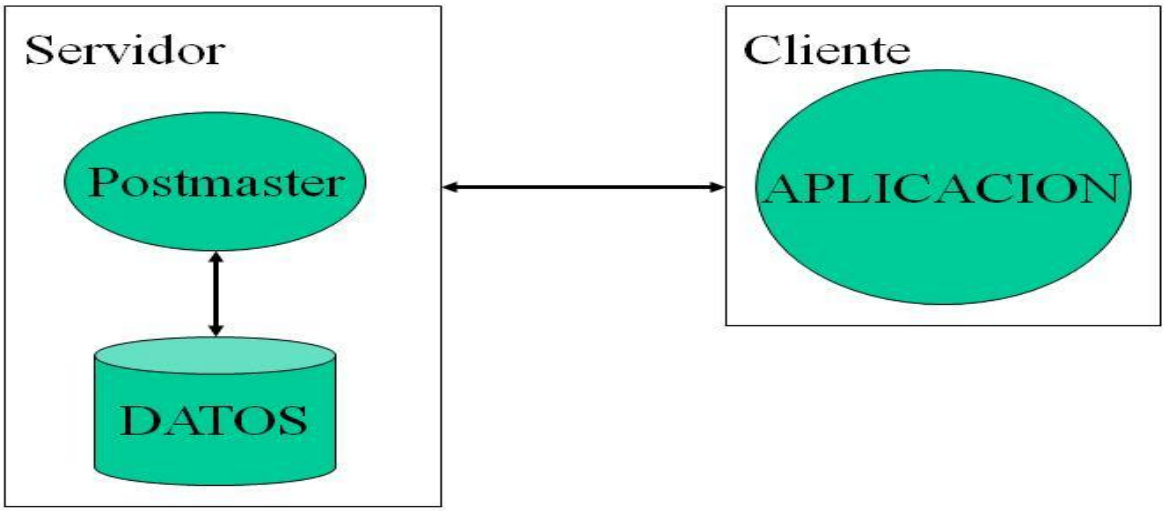

Fig. 2. Arquitectura de PostgreSQL (21).

Características:

- MVCC: Control de Concurrencia Multi-Versión (Multi-Version Concurrency Control), es la tecnología que usa para evitar bloqueos innecesarios.
- Multiplataforma.
- Soporta claves foráneas.
- Posee buenas interfaces de instalación y administración.

- Write Ahead Logging (Escritura anticipada del registro): incrementa la dependencia de la base de datos al registro de cambios antes de que estos sean escritos en la base de datos. Esto garantiza que en el hipotético caso de que la base de datos se caiga, existirá un registro de las transacciones a partir del cual podremos restaurar la base de datos.
- Cumple con las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad).
- Ha sido diseñado para mantenerse estable con grandes volúmenes de datos y transacciones.
- Es altamente configurable, con lo cual puede personalizarse de acuerdo al escenario donde será utilizado.

1.10 Sistema Gestor de Base de Datos seleccionado

Luego de realizar un análisis sobre las características de cada uno de los gestores expuestos anteriormente, se define PostgreSQL en su versión 8.0 u otra superior como gestor para la base de datos del Grupo de Calidad de la facultad 9. Las razones por la que se escogió este manejador y no otro son: utiliza la integridad referencial además es multiplataforma, se distribuye bajo una licencia que permite utilizar el producto sin restricciones, necesita pocos recursos para su funcionamiento y es un gestor muy fácil de aprender.

Criterios de comparación	Oracle	MySQL	SQL Server	PostgreSQL
Sistemas Operativos	Windows, Mac OS X, Linux y GNU/UNIX	Windows, Mac OS X, Linux, BSD y GNU/UNIX	Windows	Windows, Mac OS X, GNU/Linux, BSD y UNIX
Consumo de recursos	Necesita gran cantidad de recursos	Consume pocos recursos	Consume pocos recursos.	Consume pocos recursos
Tipo de licenciamiento	Propietario	GPL o Propietario	Propietario	BSD
Integridad referencial	Si	No	Si	Si

Fig. 3. Comparación entre gestores de bases de datos.

1.10.1 Herramienta de administración y desarrollo para PostgreSQL

Para la administración y desarrollo con PostgreSQL se utilizará la herramienta pgAdmin III, esta es una aplicación gráfica para gestionar PostgreSQL, siendo la más completa y popular con licencia Open Source. Está escrita en C++ usando la librería gráfica multiplataforma wxWidgets, lo que permite que se pueda usar en Linux, FreeBSD, Mac OS X y Windows. Es capaz de gestionar versiones a partir de la de PostgreSQL 7.3 ejecutándose en cualquier plataforma.

Está diseñada para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita enormemente la administración. La conexión al servidor puede hacerse mediante el protocolo TCP/IP.

1.11 Herramientas CASE (Computer Aided Software Engineering)

Estas herramientas permiten el diseño de un software antes de su codificación, esto se realiza mediante diagramas que permiten que todo el equipo de desarrollo comprenda lo que se quiere hacer realmente y que el trabajo que se realice sea de forma homogénea. Algunas de estas herramientas tienen un valor económico muy alto y requieren altos costos de entrenamiento para el personal. Por otra parte, algunas herramientas CASE no ofrecen o evalúan soluciones potenciales para los problemas relacionados con sistemas, o simplemente no llevan a cabo ningún análisis de los requerimientos de la aplicación.
Entre las principales herramientas se encuentran:

1.11.1 Embarcadero ER / Studio

Es una herramienta de modelado de datos fácil de usar para el diseño y construcción de bases de datos a nivel físico y lógico. Direcciona las necesidades diarias de los administradores, desarrolladores y arquitectos que construyen y mantienen aplicaciones de bases de datos grandes y complejas. Principales funcionalidades:

- Capacidad fuerte en el diseño lógico.
- Construcción automática de base de datos.

- Ingeniería inversa de base de datos.
- Un Repositorio para el modelado.
- Documentación basada en HTML.

Soporta el muy popular SQL y base de datos de escritorio, incluyendo: Oracle, Microsoft Access, Microsoft SQL Server entre otras.

1.11.2 Visual Paradigm

Es una herramienta CASE que utiliza UML como lenguaje de modelado, soporta el diagrama entidad-relación. Está disponible en varias ediciones: Enterprise, Professional, Community, Standard, Modeler y Personal, que permite escoger la más idónea para el usuario según su necesidad. Esta herramienta se distribuye bajo licencia gratuita y comercial, entre las principales características que presenta Visual Paradigm están: producto de calidad, multiplataforma, soporta aplicaciones web, varios idiomas, fácil de instalar y actualizar, compatibilidad entre ediciones, soporta a miles de usuarios trabajando sobre el mismo proyecto y permite que cada uno vea los cambios realizados en tiempo real.

Esta herramienta tiene habilitado UML 2.1, un estándar ampliamente utilizado actualmente en las empresas para el modelado de software. Permita obtener el diagrama Entidad-Relación a partir del diagrama de clases persistentes o viceversa. Tiene la capacidad de crear el esquema de clases a partir de una base de datos y crear la base de datos a partir del esquema de clases. A partir de un modelo relacional en SQL Server, MySQL u otro, es capaz de desplegar todas las clases asociadas a las tablas.

Posibilita la generación de bases de datos y la transformación de diagramas de Entidad-Relación en tablas de base de datos. Además la documentación es generada en varios formatos como JPG, XML, Excel y en el caso de la base de datos genera un script con las tablas.

1.12 Herramienta CASE seleccionada

A partir de un análisis realizado a las principales herramientas CASE anteriormente expuestas se concluyó que para el modelado de la base de datos para el Grupo de Calidad de la facultad 9 se utilizará Visual Paradigm para UML en su versión 3.4. Este software brinda múltiples ventajas al usuario como son: es fácil de instalar y actualizar, es un software multiplataforma. Visual Paradigm permite realizar todo tipo de

diagramas de clases, generar código desde diagramas y generar documentación. Además facilita la generación de bases de datos, transformación de diagramas de Entidad-Relación en tablas de base de datos e ingeniería inversa.

1.13 Conclusiones Parciales.

Los SGBD han evolucionado considerablemente desde sus inicios, mejorando cada vez más los servicios que brindan, lográndose con ellos que la información sea almacenada de forma estructurada y no duplicada. Además que garantizan el acceso controlado a la información.

En el mundo existen diversas herramientas para el diseño y procesamiento de las bases de datos, las que se van ampliando y perfeccionando con el tiempo. Definir cuál de ellas utilizar es cada vez más difícil. Por tal motivo en este capítulo se analizaron los principales modelos de bases de datos que ayuden al diseño eficiente de la misma, los gestores más utilizados en el mundo y en la UCI, y las herramientas de modelado que ayuden en este trabajo.

Capítulo II: Descripción de la solución propuesta.

2.1 Introducción

En el presente capítulo se describen los principales procesos que se llevan a cabo para el diseño de la base de datos, entre estos se encuentran: la selección de los requisitos del sistema, y la propuesta de diseño de la base de datos, para lograr este objetivo se desarrolla el modelo lógico y físico de la base de datos ya normalizada.

2.2 Requisitos funcionales y no funcionales del sistema

2.2.1 Requisitos funcionales (RF)

Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir, más específicamente son acciones que el sistema debe ser capaz de realizar.

RF1. Gestionar usuarios. El sistema debe permitir gestionar los datos referentes a los usuarios.

RF1.1. Adicionar usuario: El sistema debe permitir crear nuevos usuarios.

RF1.2. Modificar información del usuario: El sistema debe permitir modificar la información de los usuarios.

RF1.3. Eliminar usuario: El sistema debe permitir eliminar los usuarios.

RF2. Autenticar usuario: El sistema debe permitir que los usuarios se autentiquen en el sistema, una vez verificado el rol que poseen se les permitirá un acceso al sistema.

RF3. Gestionar estudiantes.

RF3.1. Adicionar estudiante: El sistema debe permitir adicionar nuevos estudiantes que se integren al Grupo de Calidad de la facultad 9.

RF3.2 Modificar datos de estudiante: El sistema debe permitir modificar la información de los estudiantes.

RF3.3. Eliminar estudiante: El sistema debe permitir eliminar a un estudiante.

RF4. Gestionar profesor.

RF4.1. Adicionar profesor: El sistema debe permitir adicionar nuevos profesores que se integren al Grupo de Calidad de la facultad 9.

RF4.2 Modificar datos de profesor: El sistema debe permitir modificar la información de los profesores.

RF4.3. Eliminar profesor: El sistema debe permitir eliminar a los profesores.

RF5. Gestionar computadora.

RF5.1. Adicionar computadora: El sistema debe permitir adicionar nuevas computadoras que se le asignen al Grupo de Calidad de la facultad 9.

RF5.2 Modificar datos de computadora: El sistema debe permitir modificar la información de las computadoras.

RF5.3. Eliminar computadora: El sistema debe permitir eliminar las computadoras que causen bajo o por algún motivo ya no pertenezcan al grupo.

RF6. Mostrar reporte: El sistema debe permitir opciones de mostrar un reporte con todos los estudiantes, profesores y computadoras del Grupo de Calidad de la facultad 9.

RF7. Crear planilla de procedimiento de inicio de proyecto: El sistema debe permitir crear una nueva planilla para cada proyecto que se inicie.

RF7.1. Adicionar proyecto certificado: El sistema debe permitir que se adicionen los proyectos que se certifiquen en el proceso de inicio de proyecto a los proyectos que ya existen en la facultad.

RF8. Gestionar planilla de caso de pruebas: El sistema debe permitir gestionar las planillas de los casos de pruebas de cada proyecto.

RF8.1. Adicionar planilla de caso de pruebas: El sistema debe permitir adicionar una nueva planilla de casos de prueba.

RF8.2 Modificar planilla de caso de pruebas: El sistema debe permitir modificar la información dentro de las planillas de caso de pruebas.

RF8.3. Eliminar planilla de caso de pruebas: El sistema debe permitir eliminar una planilla de caso de prueba en caso de ser necesario.

RF9. Noticias: El sistema debe brindar la opción de publicar noticias que sean de interés para los integrantes del Grupo de Calidad de la facultad 9 como para los demás usuarios que hagan uso de la aplicación.

RF11. Mostrar información: El sistema debe permitir que todo usuario que este autenticado en el sistema pueda ver cualquier información que está registrada.

RF12. Gestionar paquete de revisiones: El sistema debe permitir que se gestionen de forma conjunta todas las planillas que se encuentran dentro del paquete de revisiones.

RF12.1. Gestionar planilla de acciones correctivas: El sistema debe permitir gestionar las planillas de acciones correctivas.

RF12.1.1 Adicionar planilla de acciones correctivas: El sistema debe permitir adicionar las nuevas planillas de acciones correctivas.

RF12.1.2Modificar planilla de acciones correctivas: El sistema debe permitir modificar la información de las planillas de acciones correctivas.

RF12.1.3. Eliminar planilla de acciones correctivas: El sistema debe permitir eliminar una planilla de acciones correctivas en caso de ser necesario.

RF12.2. Gestionar planilla de reunión de apertura: El sistema debe permitir gestionar las planillas de reunión de apertura.

RF12.2.1. Adicionar planilla de reunión de apertura: El sistema debe permitir adicionar la nueva planilla de reunión de apertura.

RF12.2.2. Modificar planilla de reunión de apertura: El sistema debe permitir modificar la información de las planillas de reunión de apertura.

RF12.2.3. Eliminar planilla de reunión de apertura: El sistema debe permitir eliminar una planilla de reunión de apertura en caso de ser necesario.

RF12.3. Gestionar planilla de reunión de cierre: El sistema debe permitir gestionar las planillas de reunión de cierre.

RF12.3.1. Adicionar planilla de reunión de cierre: El sistema debe permitir adicionar la nueva planilla de reunión de cierre.

RF12.3.2. Modificar planilla de reunión de cierre: El sistema debe permitir modificar la información de las planillas de reunión de cierre.

RF12.3.3. Eliminar planilla de reunión de cierre: El sistema debe permitir eliminar una planilla de reunión de cierre en caso de ser necesario.

RF12.4. Gestionar evaluación de desempeño de los revisores: El sistema debe brindar la opción de adicionar más de una planilla de este tipo porque es una para cada revisor que participa en la revisión.

RF12.4.1. Adicionar evaluación de desempeño de los revisores: El sistema debe permitir adicionar evaluación de desempeño de los revisores que participan en las revisiones.

RF12.5. Gestionar informe final: El sistema debe permitir que se gestione la planilla de informe final luego de cada revisión realizada.

RF12.5.1. Adicionar planilla de informe final: El sistema debe permitir adicionar una nueva planilla de informe final.

RF12.5.2. Modificar planilla de informe final: El sistema debe permitir modificar la información de las planillas de informe final que se realiza después de cada revisión.

RF12.5.3. Eliminar planilla de informe final: El sistema debe permitir eliminar una planilla de informe final en caso de ser necesario.

RF12.6. Mostrar lista de chequeo 1.0: El sistema debe permitir que se consulte la lista de chequeo que debe estar disponible para todos los usuarios.

RF12.7. Mostrar lista de chequeo 2.0: El sistema debe permitir que se consulte la lista de chequeo que debe estar disponible para todos los usuarios.

RF12.7. Gestionar paquete de seguimiento: El sistema debe brindar la opción de gestionar el paquete de planillas que se encuentran dentro del seguimiento que se le realizan a las no conformidades encontradas en la revisión.

RF12.7.1 Modificar registro de evaluaciones del proyecto: El sistema debe permitir modificar la información de la planilla registro de evaluaciones del proyecto.

RF12.7.2 Modificar registro de no conformidades: El sistema debe permitir modificar la información de la planilla registro de no conformidades, que son las deficiencias encontradas durante la revisión.

RF12.7.3 Modificar seguimiento a las no conformidades: El sistema debe permitir modificar la información de la planilla seguimiento a las no conformidades.

RF12.7.4 Modificar registro de evaluaciones: El sistema debe permitir modificar la información de la planilla registro de evaluaciones.

RF12.7.5 Modificar plan de evaluaciones del proyecto: El sistema debe permitir modificar la información de la planilla plan de evaluaciones del proyecto.

RF12.7.6 Eliminar plan de evaluaciones del proyecto: El sistema debe permitir eliminar la planilla que contiene el cronograma de las evaluaciones que se le realizaran al proyecto.

2.2.2 Requisitos no funcionales (RNF)

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido y confiable.

RNF Restricciones en el diseño y la implementación:

- Se trabajará con la herramienta CASE Visual Paradigm para UML en su versión 3.4, utilizada para modelar los artefactos que se generen en cada uno de los flujos de trabajo.
- Se utilizará como Gestor de Base de Datos PostgreSQL en su versión 8.4 o alguna superior.

RNF de Usabilidad:

- Reducir el tiempo de respuesta a 10 segundos de las peticiones por parte de los usuarios a la base de datos.
- La base de datos tendrá gran disponibilidad y confiabilidad en cuanto a la información que se almacene.

RNF de Seguridad:

- La información almacenada o manejada desde la base de datos debe estar protegida de acceso no autorizado y divulgación.
- La base de datos será independiente de la aplicación.

RNF de Portabilidad:

- La base de datos podrá ser usada en los sistemas operativos Windows y GNU/Linux, por las características que presenta el SGBD utilizado.

2.3 Control de concurrencia a la base de datos

La mayoría de las bases de datos se utilizan en entornos multi-usuario, en los que muchos clientes utilizando la misma aplicación, o muchas aplicaciones cada una con uno o muchos clientes acceden a la misma base de datos. Cada una de esas aplicaciones enviará consultas al gestor, y normalmente cada hilo de ejecución será una transacción diferente.

El acceso simultáneo descrito puede dar como resultados información inconsistente o simplemente incorrecta, dependiendo de la mala o buena suerte que se tiene en la intercalación de las lecturas y escrituras simultáneas. Esta problemática ha llevado a diseñar e implementar diferentes estrategias de control de concurrencia, que se encargan de evitar todos esos problemas, de modo que los desarrolladores de las aplicaciones pueden “olvidarse” de ellos al escribir su código.

A diferencia de la mayoría de otros sistemas de bases de datos que usan bloqueos para el control de concurrencia, PostgreSQL mantiene la consistencia de los datos mediante un sistema denominado MVCC (Multi-Version Concurrency Control). Esto permite que mientras se consulta una base de datos, cada transacción ve una imagen de los datos (una versión de la base de datos) como si fuera tiempo atrás, sin tener en cuenta el estado actual de los datos que hay por debajo. Esto evita que la transacción vea datos inconsistentes que pueden ser causados por la actualización de otra transacción concurrente en la misma fila de datos, proporcionando *aislamiento transaccional* para cada sesión de la base de datos.

Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases, eliminando la necesidad del uso de bloqueos explícitos.

2.3.1 Aislamiento transaccional:

Aislamiento transaccional: Las transacciones están aisladas entre sí. Es decir, aunque en general hay muchas transacciones ejecutándose en forma concurrente, las actualizaciones de una transacción dada están ocultas ante las demás, hasta que esa transacción sea confirmada.

El nivel predeterminado es Serializable, ya que si se especifica alguno de los otros tres, la implementación tiene la libertad de asignar algún nivel superior (en donde "superior" está definido en términos del ordenamiento Serializable > Lectura repetible > Lectura confirmada > Lectura no confirmada).

Si todas las transacciones son ejecutadas al nivel de aislamiento Serializable queda garantizado que la ejecución intercalada de cualquier conjunto de transacciones concurrentes es seriable. Sin embargo, si alguna transacción se ejecuta a un nivel de aislamiento menor, la seriabilidad podría ser violada de varias formas diferentes. El estándar define tres formas específicas en las que se puede violar la seriabilidad: lectura sucia, lectura no repetible y fantasmas:

- **Lecturas sucias:** La transacción TI realiza una actualización sobre alguna fila, luego la transacción T2 recupera esa fila y la transacción TI termina con una instrucción deshacer. Entonces, la transacción T2 ha visto una fila que ya no existe, y que en cierto sentido nunca existió (debido a que la transacción TI efectivamente nunca fue ejecutada).
- **Lecturas no repetibles:** La transacción TI recupera una fila, luego la transacción T2 actualiza esa fila y después la transacción TI recupera nuevamente la "misma" fila. La transacción TI ha recuperado la "misma" fila dos veces, pero ve dos valores diferentes en ella.
- **Lecturas fantasmas:** La transacción TI recupera el conjunto de todas las filas que satisfacen alguna condición (por ejemplo, todas las filas de proveedores que satisfacen la condición de que la ciudad es París). La transacción 72 inserta entonces una nueva fila que satisface la misma condición. Si la transacción TI repite ahora su petición de recuperación, verá una fila que antes no existía, un "fantasma".

Los cuatro niveles de aislamiento y sus correspondientes acciones se describen a continuación.

Nivel de aislamiento	Lectura sucia	Lectura no repetible	Lectura fantasma
Lectura no confirmada	**Posible**	**Posible**	**Posible**
Lectura confirmada	**No posible**	**Posible**	**Posible**
Lectura repetible	**No posible**	**No posible**	**Posible**
Serializable	**No posible**	**No posible**	**No posible**

Tabla 1. Niveles de aislamiento de PostgreSQL (Manual de PostgreSQL).

Nota: Postgres ofrece lectura confirmada y niveles de aislamiento serializables.

La base de datos para el Grupo de Calidad de la facultad 9 será accedida mediante un canal a través de la aplicación web que se está desarrollando, la concurrencia a la base de datos no será significativa y el

control de la misma se realizará mediante el sistema de control de la concurrencia multiversión que implementa PostgreSQL.

2.4 Normalización de los datos

La normalización de la base de datos es el proceso mediante el cual se dividen las relaciones insatisfactorias distribuyendo sus atributos en relaciones mucho más pequeñas que presentan características deseables. El proceso de normalización es aplicable a los modelos entidad-relación y a los modelos relacionales. Este proceso permite, una vez terminado, que en las bases de datos no existan datos repetidos en las tablas, que se proteja la integridad de los datos.
Provee diferentes ventajas, como:

- Evita anomalías en la actualización.
- Mejora la independencia de los datos, permitiendo realizar extensiones de la base de datos, afectando muy poco, o nada, a los programas de aplicación existentes que accedan la base de datos.

El proceso de normalización tiene un nombre y una serie de reglas para cada fase y cada una se realizan en orden:

2.4.1 Primera Forma Normal (1FN)

La 1FN constituye el primer nivel en el proceso de normalización y representa el más elemental de todos. Durante el proceso de normalización, se puede comprobar que satisface la (1FN), si se demuestra que no hay presencia de atributos multivaluados o atributos compuestos, o la combinación de estos. Cada uno de los atributos de la relación debe tener un valor atómico. Cada una de las tablas deben contener una clave primaria y esta no contiene atributos nulos.
Para llevar a 1FN se crea una relación con todos los atributos y se señala la llave primaria.

La base de datos para el Grupo de Calidad de la facultad 9 cumple con la 1FN, ya que los datos de las relaciones involucradas son atómicos, o sea, no son ni multivaluados, ni compuestos y cada una de las tablas obtenidas cuenta con una clave primaria.

Nota: Ver sección 2.5 Diagrama Entidad-Relación de la base de datos.

2.4.2 Segunda Forma Normal (2FN)

Antes de pasar a la 2FN, es necesario abordar algunas definiciones previas:

Definición: **Dependencia Funcional (DF):**

Dada una relación R, se dice que el atributo Y de R es funcionalmente dependiente del atributo X de R, si y sólo si, cada valor X en R tiene asociado a él, precisamente, un valor de Y en R en cualquier momento del tiempo.

X ----> Y.

Definición: **Dependencia funcional completa**

El atributo Y es funcionalmente dependiente y completamente del atributo X, si es funcionalmente dependiente de X y no es funcionalmente dependiente de algún subconjunto de X.

Se representa: X ==> Y
Para que una relación este en 2FN debe cumplir con: primeramente debe encontrarse en 1FN y además todos los atributos que no forman parte de la llave primaria, dependen de manera total de los atributos que forman la llave primaria. Es decir se eliminan las dependencias parciales y se crean nuevas relaciones con los atributos que dependen de parte de la llave con su nuevo determinante.

Para llevar a 2FN:

- En la relación original se mantienen todos los atributos que dependen completamente de la llave.
- Crear una relación para los atributos que dependan de parte de la llave.

La base de datos para el Grupo de Calidad de la facultad 9 cumple con la 2FN, pues ya está en 1FN y cada atributo no primo de la relación es completamente dependiente de la clave primaria. La 2FN se aplicó a las relaciones que tenían claves primarias compuestas por dos o más atributos.
Nota: Ver sección 2.5 Diagrama Entidad-Relación de la base de datos.

2.4.3 Tercera Forma Normal (3FN)

Una relación está en 3FN si está en 2FN y si, y sólo si, los atributos no llaves son independientes de cualquier otro atributo no llave primaria. Esto es lo mismo que decir que se deben eliminar las dependencias transitivas de atributos no llaves respecto a la llave primaria, estando ya la relación en 2FN.

Definición: **Dependencia transitiva:**

Sean A, B y C conjuntos de atributos de una relación R. Si B es dependiente funcionalmente de A y C lo es de B, entonces C depende transitivamente de A.

Para llevar a 3FN:

- En la relación original se mantienen los atributos no llaves que no dependen transitivamente de la llave primaria.
- Crear una relación para los atributos no llaves que dependen transitivamente de la llave primaria.

Una vez llevada cada una de las relaciones a 2FN se pasó a eliminar las dependencias transitivas para obtener la 3FN. Para ello, se eliminaron los atributos de cada relación que dependían transitivamente y se pusieron en nuevas relaciones con una copia del atributo o los atributos no clave de los que dependen.

Nota: Ver sección 2.5 Diagrama Entidad-Relación de la base de datos.

Después de realizado este proceso la base de datos para el Grupo de Calidad de la facultad 9 ha quedado normalizada hasta la 3FN por lo que se considera que se desarrolló un adecuado diseño. La base de datos está en 1FN puesto que se puede asegurar que cada tupla contiene exactamente un valor para cada atributo de las tablas. También se cumple que los esquemas de relación están en 2FN, porque se encuentran en 1FN, y además todos los atributos que no son claves en las tablas, dependen totalmente de la clave primaria. Y finalmente se puede plantear que se encuentra en 3FN, porque está en 2FN, y además no existen dependencias transitivas entre llaves candidatas y atributos no primos.

2.5 Diagrama Entidad-Relación de la base de datos

Luego de realizado el diseño lógico de la base de datos para el Grupo de Calidad de la facultad 9 y normalizado el mismo quedo el siguiente diseño físico de la base de datos que a continuación se muestra con la descripción de cada una de las tablas.

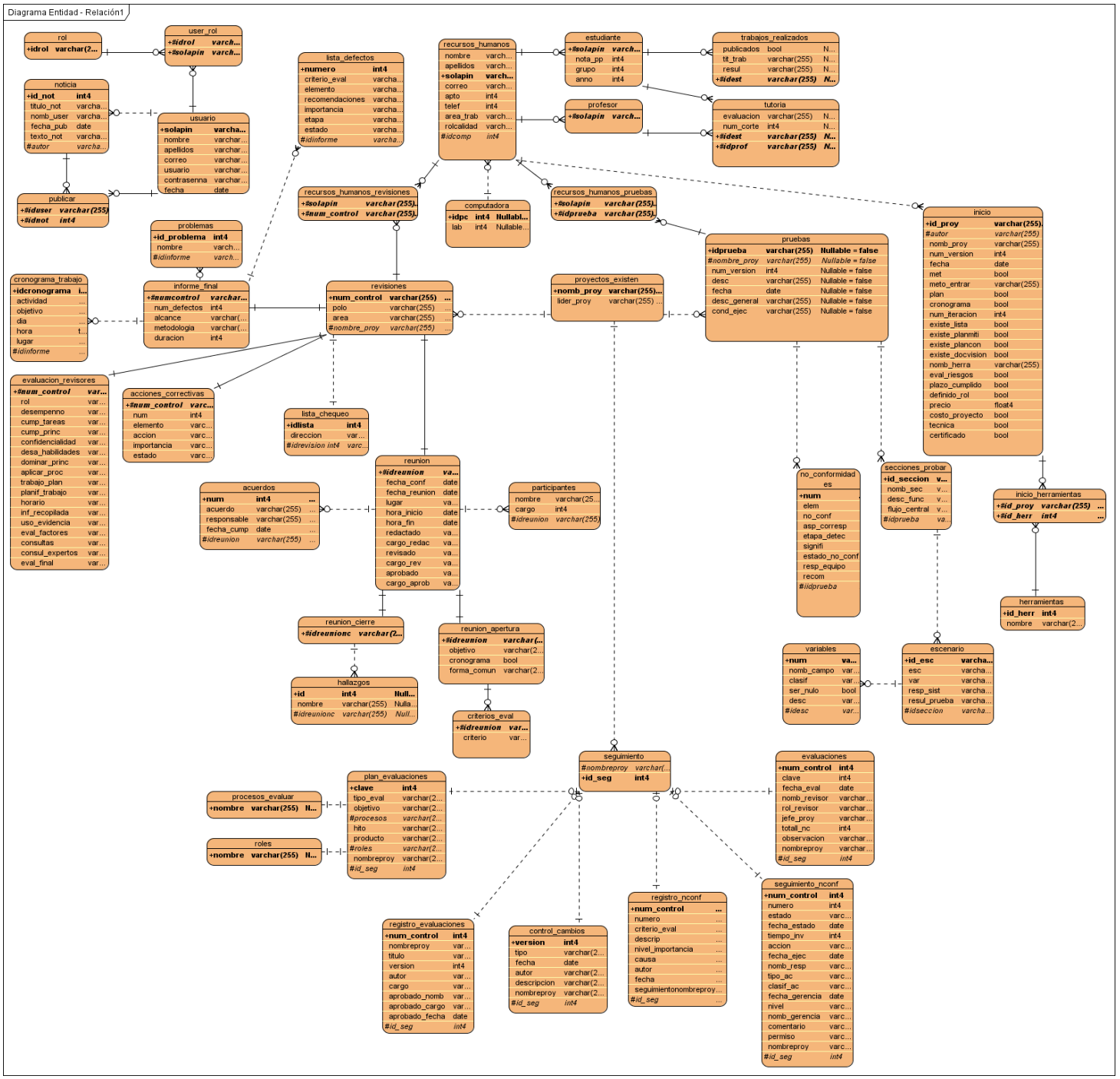

Fig. 4. Diagrama Entidad-Relación de la base de datos para el Grupo de Calidad de la facultad 9.

2.6 Conclusiones parciales

El diseño de la base de datos que se obtuvo responde a las necesidades planteadas por el Grupo de Calidad de la facultad 9, a partir de los requisitos funcionales que se describieron anteriormente. La descripción del diagrama entidad-relación como solución final con cada una de las tablas que componen el diseño de la base de datos ya normalizada.

Capítulo III: Validación del diseño realizado

3.1 Introducción

En el presente capítulo se realiza la validación teórica y funcional del diseño de la base de datos. Primeramente se analizan aspectos como la integridad, el análisis de la redundancia de la información y la seguridad de la base de datos. Lo segundo se basa en las pruebas realizadas a la misma para comprobar su funcionamiento a través de las pruebas de carga que se le realicen a la base de datos.

3.2 Validación teórica del diseño

Para realizar un buen diseño de una base de datos hay que tener en cuenta varios aspectos importantes como: la integridad de los datos, la redundancia de la información y sobre todo la seguridad en la base de datos, la cual es la base principal de los aspectos anteriormente mencionados, permitiendo controlar el acceso a los datos y que los mismos no se corrompan como resultado de acciones no controladas.

3.2.1 Análisis de la integridad de los datos

El termino integridad de los datos se refiere a la corrección o completitud de estos en una base de datos. Cuando se modifica el contenido de la base de datos con sentencias **SELECT**, **INSERT** o **UPDATE**, la integridad de los datos puede verse afectada de diferentes maneras, por ejemplo se pueden añadir datos inválidos, se pueden modificar datos ya existentes tomando valores incorrectos o eliminándolos. La integridad de los datos se contempla en diferentes niveles. Las restricciones de dominio y entidades definen las reglas para el mantenimiento de la integridad de las relaciones individuales. Las relaciones de integridad referencial aseguran que se mantengan las asociaciones necesarias entre las relaciones.

3.2.1.1 Integridad de entidades

La primera regla de integridad se aplica a las claves primarias de las relaciones: ninguno de los atributos que componen la clave primaria puede ser nulo. Por definición, una clave primaria es un identificador irreducible que se utiliza para identificar de modo único las tuplas. Que es irreducible, significa que ningún subconjunto de la clave primaria sirve para identificar las tuplas de modo único.

En el caso de la base de datos para el Grupo de Calidad de la facultad 9 cada entidad tiene definida su llave primaria, que no puede ser nula. Definir correctamente el identificador principal para cada registro, es de vital importancia, ya que esto representa la principal herramienta que utiliza el servidor de base de datos para seleccionar la información que se necesite.

3.2.1.2 Integridad de dominio

Una restricción de integridad de dominio es una regla que define valores válidos para los atributos de las diferentes tablas de una base de datos. Algunas tareas que aseguran la integridad de dominio son: definir los tipos de datos correctamente para cada campo y definir campos no nulos (NOT NULL) para evitar resultados inconsistentes. Para velar por la integridad de dominio en la base de datos, no se definieron atributos nulos y se precisaron correctamente cada uno de los tipos de datos para cada uno de los atributos.

3.2.1.3 Integridad referencial

La regla de integridad se aplica a las claves foráneas: si en una relación hay alguna clave foránea, sus valores deben coincidir con los valores de la clave primaria a la que hace referencia. Cuando se define una columna como clave foránea, las filas de la tabla pueden contener en esa columna o bien el valor nulo (ningún valor), o bien un valor que existe en la otra tabla. Eso es lo que se denomina integridad referencial y consiste en que los datos que referencian otros (claves foráneas) deben ser correctos. La integridad referencial hace que el SGBD se asegure de que no haya en las claves foráneas valores que no estén en la tabla principal.

¿Cuándo se pueden producir errores en los datos?

- Cuando se inserta una nueva fila en la tabla secundaria y el valor de la clave foránea no existe en la tabla principal.
- Cuando se modifica el valor de la clave principal de un registro que tiene 'hijos'.
- Cuando se modifica el valor de la clave foránea, el nuevo valor debe existir en la tabla principal.
- Cuando se desea borrar una fila de la tabla principal y ese registro tiene 'hijos'.

Asociada a la integridad referencial están los conceptos de actualizar los registros en cascada y eliminar registros en cascada.

- **Actualizar registros en cascada:** Esta opción le indica al SGBD que cuando se cambie un valor del campo clave de la tabla principal, automáticamente cambiará el valor de la clave foránea de los registros relacionados en la tabla secundaria.
- **Eliminar registros en cascadas:** Esta opción le indica al SGBD que cuando se elimina un registro de la tabla principal automáticamente se borran también los registros relacionados en la tabla secundaria.

Para mantener esta integridad referencial en la base de datos se utilizaron: las llaves foráneas o externas, las cuales obligan a que los valores introducidos en las columnas marcadas por esta restricción correspondan a valores en la tabla referenciadas, así como permite realizar acciones en caso de actualización o eliminación de los valores a los que se hacen referencia, para esto, se definió que al realizar la actualización o eliminación de datos, las mismas se realizarían en cascada. Puntualizar que a la hora del administrador de la aplicación eliminar debe tener un previo conocimiento de las dependencias que existen entre las tablas, debido a que eliminar una tabla intermedia podría producir errores en la base de datos.

3.2.2 Análisis de la redundancia de la información

La redundancia de datos es aquella información duplicada o almacenada varias veces en la misma base de datos. Esto dificulta la tarea de modificación de datos y es el motivo más frecuente de inconsistencia de los mismos. Además requiere un mayor espacio de almacenamiento, que influye en un mayor coste y tiempo de acceso a los datos. La redundancia siempre debe evitarse, aunque en proyectos grandes es imposible evitarla al cien por ciento, lo que a veces es deseable por cuestiones de rendimiento.

Con un buen diseño de base de datos se logrará evitar la aparición de información repetida o redundante. Lo ideal es lograr una redundancia nula; no obstante, en algunos casos la complejidad de los cálculos hace necesaria la aparición de redundancias. La base de datos para el Grupo de Calidad de la facultad 9

está normalizada hasta la 3FN, o sea libre de redundancias, ya que se elimina casi completamente la presencia de datos repetidos innecesariamente.

3. 2.3 Seguridad en la base de datos

La seguridad en la base de datos consiste en las acciones que toma el diseñador al momento de crear esta para evitar cualquier ataque o acceso no autorizado con la intención de modificar o difundir la información almacenada. La seguridad constituye una de las características más importantes por la que se debe velar al diseñar la base de datos. En el Grupo de Calidad de la facultad 9, los datos que se almacenan en la base de datos constituyen la parte fundamental del funcionamiento del mismo, ya que una pérdida de esta información ocasionaría una gran demora en el trabajo debido a que este depende directamente de la información almacenada.

El SGBD a utilizar para diseñar las bases de datos para el Grupo de Calidad de la facultad 9- PostgreSQL- incluye formas de restringir el acceso al sistema como por ejemplo la definición de roles y usuarios y la configuración del archivo pg_hba.conf.

Para la base de datos del Grupo de Calidad de la facultad 9 se definió el usuario admincalidad, este se utiliza para la conexión entre el servidor de la aplicación web y el servidor de base de datos. En PostgreSQL no es recomendable el uso del usuario postgres que viene por defecto, el mismo es dueño de todas las bases de datos que hay en el servidor. El objetivo de crear este usuario admincalidad es que tiene todos los permisos sobre la base de datos calidad, es decir la base de datos a la que se va a conectar la aplicación y no sobre ninguna otra ni sobre el servidor.

La autenticación del cliente se controla mediante un archivo de configuración, que tradicionalmente se denomina pg_hba.conf. El formato general del archivo pg_hba.conf es un conjunto de registros, uno por línea. Las líneas en blanco se ignoran, al igual que cualquier texto después del carácter # comment. Un registro se compone de una serie de campos que están separados por espacios y fichas. Los campos pueden contener espacios en blanco.

En el archivo pg_hba.conf, cada registro especifica un tipo de conexión, el nombre de la base de datos, el nombre de usuario, la dirección IP del cliente y el método de autenticación que se utilizará para las conexiones.

Existen cuatro tipos de conexiones diferentes, ellas son:

- Local: Este registro coincide con los intentos de conexión usando los sockets de dominio UNIX.
- Host: Este registro coincide con los intentos de conexión realizados a través de TCP / IP.
- Hostssl: Este registro coincide con los intentos de conexión realizados a través de TCP / IP, pero sólo cuando se realiza la conexión con el cifrado SSL. Para hacer uso de esta opción debe ser el servidor integrado con soporte SSL. Además, SSL debe estar activado a la hora de inicio del servidor.
- Hostnossl: Este tipo de registro tiene el comportamiento contrario de hostssl, sólo coincide con los intentos de conexión realizados a través de TCP / IP que no utilizan SSL.

Hay que especificar el nombre de la base o las bases de datos a las que puede acceder el usuario definido por el administrador de la base de datos. Además a través de qué dirección IP se va a conectar a la base de datos, especificando el método de conexión que utilizará. Existen diferentes métodos de conexión, cada uno de ellos proporcionan diferentes ventajas, estos son:

1. **trust (confianza):** Permite una conexión de manera incondicional. Este permite que cualquiera pueda conectarse al servidor de base de datos PostgreSQL y autenticarse como cualquier usuario sin necesidad de contraseña ni otra forma de autenticación.
2. **reject (rechazar):** Rechaza la conexión de manera incondicional. Esto es útil para "filtrar" a determinados hosts de un grupo, por ejemplo la línea reject bloquea un host en especifico, pero las siguiente línea permite que los host restantes de esa red si se puedan conectar.
3. **md5:** Se necesita una contraseña cifrada en MD5 para realizar la autenticación.
4. **password (contraseña):** Se necesita una contraseña no cifrada para la autenticación. Dado que la contraseña se envía en texto claro sobre la red, esto no debe ser utilizado en redes no seguras.
5. **gss:** Utiliza GSSAPI para autenticar al usuario. Esto sólo está disponible para las conexiones TCP / IP.
6. **sspi:** Utiliza SSPI para autenticar al usuario. Sólo está disponible en Windows**.**

7. **krb5:** Utiliza Kerberos V5 para autenticar al usuario. Esto sólo está disponible para las conexiones TCP / IP.
8. **ident:** Obtiene el nombre de usuario en el sistema operativo del cliente y verifica si coincide con el nombre de la base de datos solicitada. Además se le puede especificar el parámetro map=nombre_map donde nombre_map deberá aparecer registrado en el archivo pg_ident.conf del servidor de PostgreSQL, especificando que nombre de usuario tiene permisos y con qué usuario de postgres se puede autenticar en el servidor
9. **ldap:** Autenticar con un servidor LDAP.
10. **radius (radio):** Autenticar con un servidor RADIUS.
11. **cert:** Autenticar mediante certificados de cliente SSL.
12. **pam:** Autenticación mediante la Pluggable Authentication Modules (PAM) servicios prestados por el sistema operativo.

En el caso de la base de datos para el Grupo de Calidad de la facultad 9 este archivo quedó configurado de la siguiente manera:

Tipo de conexión	Base de Datos	Usuario	Dirección IP	Método de conexión
host	**calidad**	**admincalidad**	**10.34.8.8/32**	**md5**
host	**all**	**postgres**	**255.0.0.0/8**	**md5**

Tabla 2. Archivo de configuración pg_hba.conf.

También se configuró el archivo postgresql.conf, en este fichero se pueden cambiar todos los parámetros de configuración que afectan el funcionamiento y comportamiento de PostgreSQL. Los parámetros más importantes que fueron modificados en el archivo son:

- **listen_addresses:** Especifica las direcciones TCP / IP a través de las cuales el servidor puede escuchar desde aplicaciones cliente. El valor toma la forma de una lista con nombres de host y direcciones IP numéricas separada por comas. La entrada especial * corresponde a todas las interfaces IP. Si la lista está vacía, el servidor no escucha en ninguna interfaz IP, en cuyo caso se utilizan los sockets de dominio UNIX para conectarse a él. El valor por defecto es localhost, que permite las conexiones locales. Este parámetro sólo se puede establecer en el arranque del servidor.

- **port:** El puerto TCP que el servidor escucha de forma predeterminada es 5432. Tener en cuenta que el número de puerto que se utiliza es el mismo para todas las direcciones IP. Este parámetro sólo se puede establecer en el arranque del servidor.
- **max_connections:** Determina el número máximo de conexiones simultáneas al servidor de base de datos. El valor predeterminado es típicamente 100 conexiones, pero este valor puede variar. Este parámetro sólo se puede establecer en el arranque del servidor.

El archivo porgresql.conf quedo configurado de la siguiente manera:

listen_addresses	port	max_connections
"*"	**5800**	**100**

Tabla 3. Archivo de configuración postgresql.conf.

3.3 Validación funcional

Para comprobar que la base de datos para el Grupo de Calidad de la facultad 9 cumple con los requisitos funcionales definidos, es necesario realizarle pruebas antes de dar por terminado el proceso de diseño. El propósito de estas pruebas es simular una carga de producción real y observar cómo se comporta la base de datos bajo esta carga. Esto permite solucionar los problemas de rendimiento, antes de poner la base de datos en marcha.

3.3.1 Herramientas utilizadas para el desarrollo de las pruebas

3.3.1.1 Advanced Data Generator 2

Advanced Data Generator 2 es una poderosa herramienta para generar los llamados "datos de prueba" en las bases de datos para aplicaciones. Incluye una biblioteca de datos de la vida real y tiene varias características que hacen que los datos de prueba sean lo más real posible.

Puede conectarse a cualquier fuente de datos ODBC o ADO, lo que hace el Advanced Data Generator una herramienta genérica para todos los motores de base de datos. Puede generar datos directamente en la base de datos a través de tablas, procedimientos almacenados o la sentencia SQL (INSERT). Posee compatibilidad con algunos tipos de datos como el entero, float, varchar y los booleanos. Esta herramienta se utilizó para poblar de datos la base de datos y poder realizar las pruebas de volumen y carga a la misma, facilitando el llenado con datos reales y en un tiempo muy breve.

3.3.1.2 Apache Jmeter 2.3

Apache Jmeter es una herramienta de carga diseñada para realizar Pruebas de Rendimiento, desarrollada en Java, permite testar el funcionamiento y el rendimiento de diversos recursos de software, entre ellos: objetos de Java, servlets, scripts de Perl, bases de datos o servidores FTP. Apache Jmeter también permite la ejecución de pruebas distribuidas entre distintos ordenadores, para realizar pruebas de rendimiento.

El Apache Jmeter incluye una interfaz gráfica de usuario que facilita el diseño de las pruebas. Esta interfaz además de aportar un cómodo entorno de trabajo, también permite guardar y alterar tanto los test desarrollados como los componentes que lo integran. Gracias a esto se pueden reutilizar las pruebas o módulos de las mismas en el desarrollo de nuevas pruebas.

La aplicación tiene dos entidades básicas. Por un lado tiene el plan de pruebas que representa una prueba y por el otro los elementos que representan las diferentes etapas o acciones que se usaran para componer las pruebas. El Apache Jmeter basa su funcionamiento en elementos integrados dentro de una estructura de árbol. De esta forma de trabajo el entorno gráfico es apropiado para ver y editar los elementos que componen un plan de prueba. Otro detalle que se debe tener muy presente, es que el orden en el árbol (orden descendente) determina el orden de ejecución de los elementos dentro de la prueba.

Nota: Apache Jmeter requiere Java Runtime Environment (JRE 6) y el driver p*ostgresql-8.4-701* para la conexión a la base de datos.

3.3.2 Ventajas de la automatización de las pruebas

El proceso de elaboración de pruebas automáticas no es un proceso complejo y sí necesario en cada uno de las líneas de producción. Varias son las ventajas por las cuales el equipo de calidad y de desarrollo del software, necesitan adoptar esta política.

- Confiable: Las pruebas realizan exactamente las mismas operaciones cada vez que se ejecutan, de tal modo se elimina en gran medida el error humano.

- Repetible: Se puede probar cómo el software reacciona bajo ejecución repetida de las mismas operaciones.
- Programable: Se pueden programar las pruebas de manera sofisticada y automatizada.
- Entendible: Se puede construir un test de pruebas que cubra cada característica del uso de la aplicación.
- Reutilizable: Se puede reutilizar pruebas en diversas versiones de un uso, aun realizando cambios en la interfaz de usuario.
- Software de una calidad mejor: Porque se pueden funcionar más pruebas en menos tiempo y con pocos recursos.
- Rápido: Usuarios humanos son mucho más lentos que usuarios automatizados, por tanto el funcionamiento de las pruebas en herramientas se realiza de manera más rápida.
- Reducción de costos: Como el número de los recursos para la prueba de la se reduce. Las inversiones asociadas disminuyen.

3.3.3 Pruebas de rendimiento

La escalabilidad y disponibilidad son factores clave en los mercados actuales. Las modernas tecnologías de desarrollo hacen que las nuevas aplicaciones sean capaces de servir a usuarios rápidamente y en entornos en constante cambio. Los usuarios esperan un alto grado de servicio y también esperan que este grado sea mantenido constantemente sin tener en cuenta las circunstancias. Es por esto, que las pruebas de rendimiento son parte fundamental en el proceso de desarrollo de una aplicación.

Las pruebas de rendimiento se le realizan a un sistema informático para validar su uso, ejemplo de ellas son: pruebas de volumen, carga, stress, etc. Las que más impactan en el desarrollo de bases de datos son las pruebas que tienen relación con el rendimiento, capacidad y concurrencia. En este sentido las pruebas que se realizaron a la base de datos para el Grupo de Calidad de la facultad 9 serán las de volumen y carga.

3.3.3.1 Pruebas de Volumen

Las pruebas de volumen son pruebas típicas de entornos que utilicen bases de datos. Las mismas se realizan para analizar el comportamiento del sistema o base de datos con volúmenes de datos

almacenados lo más similar posible a los esperados en la explotación real del sistema. Para el sistema en cuestión la base de datos se pobló con datos introducidos a través del uso de la herramienta Advanced Data Generator.

En las pruebas de volumen de agregaron un total de 99028 de tuplas a razón de más de 400 por segundo, quedando distribuidas de la siguiente manera:

Nombre de tabla	Cantidad de tuplas	Nombre de tabla	Cantidad de tuplas
acciones_correctivas	5000	profesor	100
acuerdos	1000	proyectos_existen	8
computadora	100	pruebas	5000
control_cambios	5000	publicar	100
criterios_eval	1000	recursos_humanos	100
cronograma_trabajo	5000	recursos_humanos_pruebas	5000
escenario	1500	recursos_humanos_revisiones	5000
estudiante	100	registro_evaluaciones	5000
evaluacion_revisores	1000	registro_nconf	5000
evaluaciones	1000	reunion	5000
hallazgos	1000	reunion_apertura	5000
herramientas	100	reunion_cierre	5000
informe_final	5000	revisiones	5000
inicio	100	rol	100
inicio_herramientas	100	roles	100
lista_chequeo	5000	secciones_probar	3000
lista_defectos	5000	seguimiento	1000
no_conformidades	5000	seguimiento_nconf	5000
noticia	100	trabajos_realizados	100
participantes	20	tutoria	100
plan_evaluaciones	5000	user_rol	100
problemas	5000	usuario	100
procesos_evaluar	1000	variables	1000
Cantidad Total	109 028		

Tabla 4. Cantidad de datos introducidos.

Al introducir los datos no se presentaron problemas de límite de capacidad, ni de volumen de datos. Tampoco se detectaron desbordamientos de matrices, columnas, atributos, tipos de datos, ni peticiones excesivas de memoria. No se detectaron problemas con los tipos de datos definidos en el diseño de la

base de datos. Lo anteriormente planteado garantiza que el gestor utilizado y el diseño de las estructuras de la base de datos soportan completamente el almacenamiento de los niveles de información requeridos para la puesta en producción de la base de datos.

3.3.3.2 Pruebas de Carga

Una prueba de carga se ejecuta para comprender el comportamiento de una aplicación ante una carga determinada. Esta carga puede ser el número de usuarios esperado ejecutando o un número de transacciones durante un tiempo determinado. El resultado de esta prueba dará el tiempo de respuesta de todas las transacciones críticas.

El escenario que se creó para la realización de las pruebas fue una estación cliente con el Apache Jmeter configurado directamente con el servidor de base de datos. Se considera necesario aclarar que el servidor de base de datos utilizado para las pruebas no posee todas las prestaciones de un servidor profesional debido a que se utilizó una estación cliente con características mejoradas. Esto afecta la calidad de las pruebas pero su objetivo es dar una idea del rendimiento de la solución.

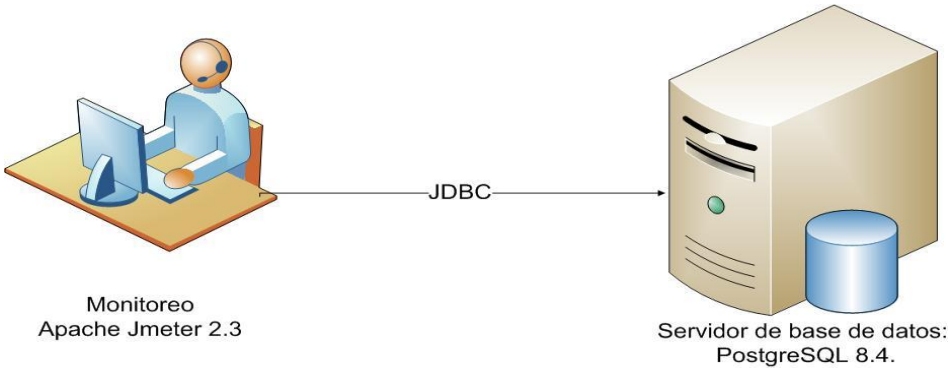

Fig. 5. Configuración para las pruebas de carga.

Características del servidor de base de datos y la estación cliente:

- Sistema Operativo: Windows XP Profesional, Versión 2002, Service Pack 3.
- Microprocesador: Intel Pentium IV.
- Velocidad: 3.0 GHz

- Memoria RAM: 1.0 GB.

La configuración del Apache Jmeter quedo de la siguiente forma:

- Primeramente se creó un grupo de hilos donde se definieron la cantidad de usuarios que realizarían peticiones al servidor.
- Seguidamente se configuró el acceso a la base de datos en la herramienta en el archivo Configuración de la conexión JDBC en el cual se detalla el nombre de la variable que será usada para las peticiones, la dirección url a través de la cual se conecta a la base de datos, con que usuario y contraseña se accede a la misma.
- Luego de configurada la conexión se pasó a definir la petición que se le realizaría a la base de datos en el archivo Petición JDBC.

Variables que se tuvieron en cuenta en los resultados obtenidos:

- **Media:** tiempo promedio de respuesta de todas las peticiones.
- **Mediana:** tiempo promedio de la mitad de los resultados, la otra mitad tomara un tiempo entre la mediana y el valor máximo.
- **Mínimo:** mínimo tiempo de respuesta de una petición.
- **Máximo:** máximo tiempo de respuesta de una petición.

Prueba No 1: Obtener los problemas que tuvo cada proyecto, el número de defectos, la metodologia usada, y la cantidad de revisiones de aquellos proyectos donde la metodología utilizada no es RUP y la cantidad de defectos encontrados sea mayor que 10.

- Tablas involucradas: problemas, proyectos existen, informe final, revisiones.
- Cantidad total de filas en la agregación: 15008
- Cantidad total de filas recuperadas: 2688.
- Consulta:

```
SELECT public.problemas.nombre, public.proyectos_existen.nomb_proy,
public.informe_final.num_defectos, public.informe_final.metodologia, public.revisiones.polo
FROM public.proyectos_existen INNER JOIN public.revisiones ON (public.proyectos_existen.nomb_proy
= public.revisiones.nombre_proy) INNER JOIN public.informe_final ON (public.revisiones.num_control
```

```
public.informe_final.numcontrol)  INNER JOIN public.problemas ON (public.informe_final.numcontrol =
public.problemas.idinforme)
WHERE public.informe_final.metodologia! = 'RUP' AND public.informe_final.num_defectos > 10
ORDER BY  public.revisiones.polo
```

- Cantidad de usuarios:
  - 40 usuarios concurrentes

	Media (ms)	**Mínimo (ms)**	**Máximo(ms)**	**Mediana (ms)**
Resultados	641	203	1219	593

  - 7 usuarios concurrentes

	Media (ms)	**Mínimo (ms)**	**Máximo (ms)**	**Mediana**
Resultados	133	125	156	125

  - Gráfico de relación entre las pruebas

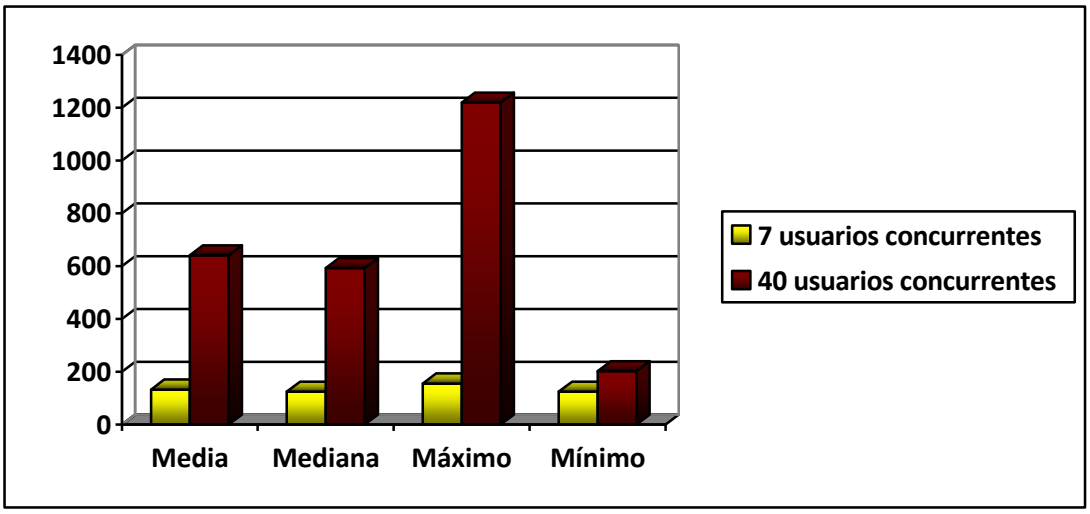

Fig. 6. Representación de la prueba 1.

Prueba No 2: Obtener el nombre y número de solapín de los estudiantes y profesores que hayan coincidido en revisiones, donde el total de no conformidades sea mayor que 3.

- Tablas involucradas:, recursos_humanos, revisiones, reunion, informe_final.
- Cantidad total de filas en la agregación: 15100
- Cantidad total de filas recuperadas: 1430.
- Consulta:

```
SELECT  public.recursos_humanos.nombre, public.recursos_humanos.solapin
FROM  public.revisiones  INNER JOIN public.recursos_humanos_revisiones ON
(public.revisiones.num_control = public.recursos_humanos_revisiones.num_control)  INNER JOIN
public.recursos_humanos ON (public.recursos_humanos_revisiones.solapin =
public.recursos_humanos.solapin) INNER JOIN public.reunion ON (public.revisiones.num_control =
public.reunion.idreunion)  INNER JOIN public.informe_final ON (public.revisiones.num_control =
public.informe_final.numcontrol)
WHERE
public.reunion.fecha_reunion BETWEEN '2010-03-01' AND '2010-03-30' AND
public.informe_final.num_defectos > 3
```

- Cantidad de usuarios:
  - 40 usuarios concurrentes

	Media (ms)	Mínimo (ms)	Máximo (ms)	Mediana (ms)
Resultados	2042	47	4688	2235

  - 7 usuarios concurrentes

	Media (ms)	Mínimo (ms)	Máximo (ms)	Mediana (ms)
Resultados	536	78	1188	547

  - Gráfico de relación entre las pruebas

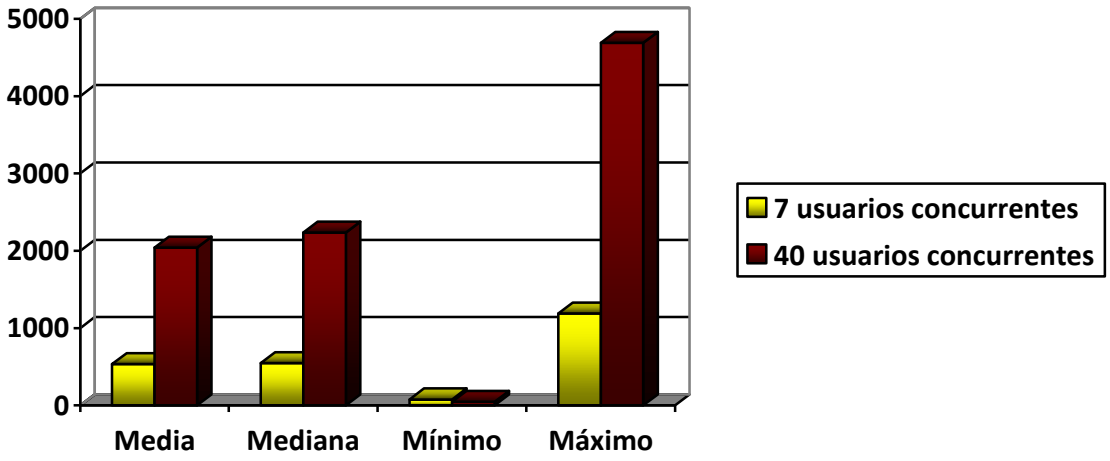

Fig. 7. Representación de la prueba 2.

Luego de realizadas las pruebas de carga a la base de datos para el Grupo de Calidad de la facultad 9 se concluyó que las mismas arrojaron resultados satisfactorios, proporcionando un tiempo de respuesta muy

bueno en comparación con la gran cantidad de información almacenada para la realización de las pruebas, esta es muy superior a la generada por el proyecto. Esto se puede calcular aproximadamente a partir de 280 tuplas generadas a la semana, teniendo en cuenta que se realicen dos auditorias y dos revisiones a los proyectos de la facultad, lo que daría como resultado un total de 13440 tuplas en el periodo de un año.

3.4 Conclusiones Parciales.

Con la validación teórica del diseño realizado se chequearon una serie de parámetros, aspectos y consideraciones importantes a tener en cuenta para realizar un correcto diseño de una base de datos. Se analizó la seguridad, realizando un control de acceso a la base de datos, haciendo uso en primer lugar de la definición de usuarios de acceso que permite PostgreSQL y además de los archivos de configuración de conexión que tiene el SGBD. Este es el aspecto fundamental que se analizó en la validación teórica.

Para la validación funcional se procedió a realizar pruebas de carga y volumen a la base de datos. Se puede concluir que las pruebas de volumen fueron satisfactorias porque se pobló la base de datos con información similar a la generada en nueve años de vida del proyecto y las pruebas de carga arrojaron resultados muy buenos en cuanto al tiempo de respuesta en consideración con las condiciones del servidor no profesional para estas. Las herramientas utilizadas durante el desarrollo de estas pruebas fueron de gran ayuda para la validación funcional de la base de datos para el Grupo de Calidad de la facultad 9.

Conclusiones

Durante el desarrollo del presente trabajo investigativo se adquirieron un conjunto de conocimientos que posibilitaron el avance del mismo. Para darle solución al problema planteado en la investigación, se realizaron cada una de las tareas planteadas. Al terminaron se arrojaron las siguientes conclusiones:

- El estudio realizado para la investigación permitió conocer la realidad existente en la forma de almacenamiento de los procesos de gestión de información del Grupo de Calidad de la facultad 9.
- La selección de las herramientas de desarrollo (CASE-Visual *Paradigm*, SGBD- *PostgreSQL*) permitió dotar al Grupo de Calidad de la facultad 9 de una estructura para almacenar la información correspondiente a sus procesos de gestión de información.
- Se realizo el diseño lógico y físico de una base de datos que cumple con las reglas de normalización hasta la 3FN, por lo que no existirán problemas de inconsistencias en los datos.
- Se analizó el funcionamiento de la base de datos, y la misma respondió satisfactoriamente a todas las pruebas.

Se ha cumplido el objetivo de la investigación, pues se obtuvo una base de datos que permite el almacenamiento de toda la información relacionada con del Grupo de Calidad de la facultad 9, permitiendo un mayor control de la misma.

Recomendaciones

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

- Mejorar las prestaciones del hardware utilizado en el servidor de la base de datos para lograr un mayor rendimiento.
- Proporcionar un mantenimiento y soporte regular a la base de datos enfocados a su mejor funcionamiento.
- Realizar pruebas de stress a la base de datos para analizar su funcionamiento ante algunas condiciones anormales que puedan producirse.

Bibliografía Referenciada

1. 1. Mato G. Lic.Rosa M. *Diseño de Bases de Datos.* 1999.

2. Date, Christopher J. *Introduccion a los sistemas de bases de datos.* 7ma ed, Mexico: Editorial Pearson Educacion S.A, 2001, ISBN 968-444-419-2.

3. Marqués A. María M. *Diseño de bases de datos,* 2001. [Disponible en]: http://www3.uji.es/~mmarques/f47/apun/node68.html.

4. Korth, Henry F, Silberschatz, Abraham. *Database System Concepts*, McGraw-Hill, New York, NY, 1991, ISBN 0071008047.

5. Marqués Andrés, María Mercedes. *Apuntes de Ficheros y Bases de Datos.* 2001. [Disponible en]: http://www3.uji.es/~mmarques/f47/apun/apun.html.

6. Capote M. Lic. Belina, González M. Dr. Diego, Rodríguez D., Lic. Emma. *La gestión de información como herramienta fundamental en el desarrollo de los centros toxicológicos.* 2003. [Disponible en]: http://bvs.sld.cu/revistas/aci/vol11_2_03/aci030203.htm.

7. Zequera R. Karen M. *PRIMICIA, Plataforma de Televisión Informativa. Rol de Diseñador de Bases de Datos,* Universidad de las Ciencias Informaticas, 2009.

8. Esteves D. Msc Gabino, Ochoa H. Ing. Eduardo. *Programa orientador "Gestión de la información"*. Universidad Michoacana de San Nicolás de Hidalgo. 2001.

9. Álvarez, Sara. *Modelos de bases de datos.* 2007. [Disponible en]: http://www.desarrolloweb.com/articulos/modelos-base-datos.html.

10. Suavita. G. Francisco. *Conceptos de Bases de Datos*, 2009 [Disponible en]: http://fransuavg.blogspot.com/search?q=modelos.

11. Bartle, Phil. *Información para la gestión y gestión de la información*. 2007 [Disponible en]: http://www.scn.org/mpfc/modules/mon-miss.htm.

12. Portal en español sobre PostgreSQL. [Disponible en]: http://www.postgresql-es.org/

13. Burbano P. Ing. Diego J. *Análisis comparativo de bases de datos de código abierto vs código cerrado.* 2006. Quito, Ecuador.

14. Portal de PostgresSQL. Manual de Postgres. [Disponible en]: http://www.postgresql.org/files/documentation/pdf/8.3/postgresql-8.3-A4.pdf

15. Fernandez P. Yordanis. *Plataforma de VideoWeb. Rol de Diseñador de Bases de Datos,* Universidad de las Ciencias Informaticas, 2009.

16. *Libros en pantalla de SQL 2008.* [Disponible en]: http://msdn.microsoft.com/es-es/library/ms184276.aspx
17. Sicilia. Miguel. A. *Control de concurrencia en bases de datos relacionales.* 2008. [Disponible en]: http://cnx.org/content/m18939/latest/

18. Burbano P. Ing. Diego J. *Análisis comparativo de bases de datos de código abierto vs código cerrado.* 2006. Quito, Ecuador.
19. Portal en español sobre PostgreSQL. [Disponible en]: http://www.postgresql-es.org/
20. Equipo de Desarrollo PostgreSQL. [Disponible en]: http://www.ibiblio.org/pub/Linux/docs/LuCaS/Postgresqles/web/navegable/todopostgresql/postgres.html.
21. Lotero. Hernán A. *Implementando Soluciones cliente/servidor con PostgreSQL.* [Disponible en]: http://www.umanizales.edu.co/programs/ingenieria/congresolinux/dia1/lotero/presenta.ppt
22. Guía de Ubuntu. [Disponible en]: http://www.guia-ubuntu.org/index.php?title=PgAdmin_III.

Bibliografía Consultada

1. Mato G. Lic. Rosa M. *Diseño de Bases de Datos.* 1999.
2. Korth, Henry F, Silberschatz, Abraham. *Database System Concepts*, McGraw-Hill, New York, NY, 1991, ISBN 0071008047.
3. Esteves D. Msc Gabino, Ochoa H. Ing. Eduardo. *Programa orientador "Gestión de la información"*. Universidad Michoacana de San Nicolás de Hidalgo. 2001.
4. Zequera R. Karen M. PRIMICIA, *Plataforma de Televisión Informativa. Rol de Diseñador de Bases de Datos*, Universidad de las Ciencias Informáticas, 2009.
5. Álvarez, Sara. *Modelos de bases de datos.* 2007. [Disponible en]: http://www.desarrolloweb.com/articulos/modelos-base-datos.html.
6. Suavita. G. Francisco. *Conceptos de Bases de Datos*, 2009 [Disponible en]: http://fransuavg.blogspot.com/search?q=modelos.
7. Bartle, Phil. *Información para la gestión y gestión de la información*. 2007 [Disponible en]: http://www.scn.org/mpfc/modules/mon-miss.htm.
8. León, H. A. Rolando, Coello. G. Sayda. *El Paradigma Cuantitativo de la Investigación Científica*. Universidad de las Ciencias Informáticas, Ciudad de la Habana, 2002, ISBN: 959-16-0343-6.
9. *Capítulo 12: Bases de datos. Concepto, características, funcionalidades*. 2009. [Disponible en]: http://www.mailxmail.com/curso-informatica-administracion-publica-3/bases-datos-concepto-caracteristicas-funcionalidades.
10. *Capitulo 10: Tipos de bases de datos.* 2001. [Disponible en]: http://www.mailxmail.com/curso-introduccion-access/tipos-base-datos.
11. Masip. David. *Que es Oracle*. 2002. [Disponible en]: http://www.desarrolloweb.com/articulos/840.php.
12. Sánchez R. Juan E. *PostgreSQL: la alternativa de código abierto a Oracle*. 2009. [Disponible en]: http://blogs.antartec.com/opensource/2009/04/postgresql-la-alternativa-de-codigo-abierto-a-oracle/.
13. Valmaseda O. Marcos L. *Modelo de negocio basado en la filosofía del Software Libre para una Empresa Cubana de Soporte*, UXI, Revista de software libre de la UCI, No. 8 Vol. 02, [Disponible en] : http://softwarelibre.uci.cu/revista-uxi-1/volumen-2/uxi08_v02.pdf/view.

14. *Embarcadero lanza ER/Studio 7.0 con soporte para el modelado y análisis de datos*. 2005. [Disponible en] : http://www.financialtechmag.com/000_estructura/index.php?id=24&idb=57&ntt=3777&sec=12&vn=1.
15. Álvarez, Sara. *Introducción a este concepto y características especiales*. 2007. [Disponible en]: http://www.desarrolloweb.com/articulos/modelos-base-datos.html.
16. Burbano P. Ing. Diego J. *Análisis comparativo de bases de datos de código abierto vs código cerrado.* 2006. Quito, Ecuador.
17. *Modelo Vista Controlador.* [Disponible en]: http://www.webtutoriales.com/tutoriales/programacion/modelo-vista-controlador.54.html.
18. *Manual de profesores. Sistema de Bases de Datos.* 2003-2004. Universidad de las Ciencias Informáticas.
19. Andrés M. María M. *El modelos entidad relación.* 2001. [Disponible en]: http://www3.uji.es/~mmarques/f47/apun/node83.html.
20. *¿Qué es una base de datos?-Definición de base de datos.* [Disponible en]: http://www.masadelante.com/faqs/base-de-datos.
21. Martínez C. Juan C. *Definiciones de información.* [Disponible en]: http://www.eumed.net/flechas/Definfo.htm.
22. Lanzillotta Analía. *Definición y significado de Información.* [Disponible en]: http://www.mastermagazine.info/termino/5366.php.
23. Guglielmetti Marcos. *Definición y significado de Proceso.* [Disponible en]: http://www.mastermagazine.info/termino/6377.php.
24. Rebolledo S. Gustavo. *Gestión, Calidad y Agregación de valor en información.* [Disponible en]: http://b3.bibliotecologia.cl/ar-gestion.htm.
25. Rementeria P. *Ariel. Conceptos de gestión.* Universidad de Santiago de Chile. [Disponible en]: http://www.gurusonline.tv/es/conteudos/drucker4.asp.
26. Portal en español sobre PostgreSQL. [Disponible en]: http://www.postgresql-es.org/
27. Equipo de Desarrollo PostgreSQL. [Disponible en]: http://www.ibiblio.org/pub/Linux/docs/LuCaS/Postgresql-es/web/navegable/todopostgresql/postgres.html.

Anexos

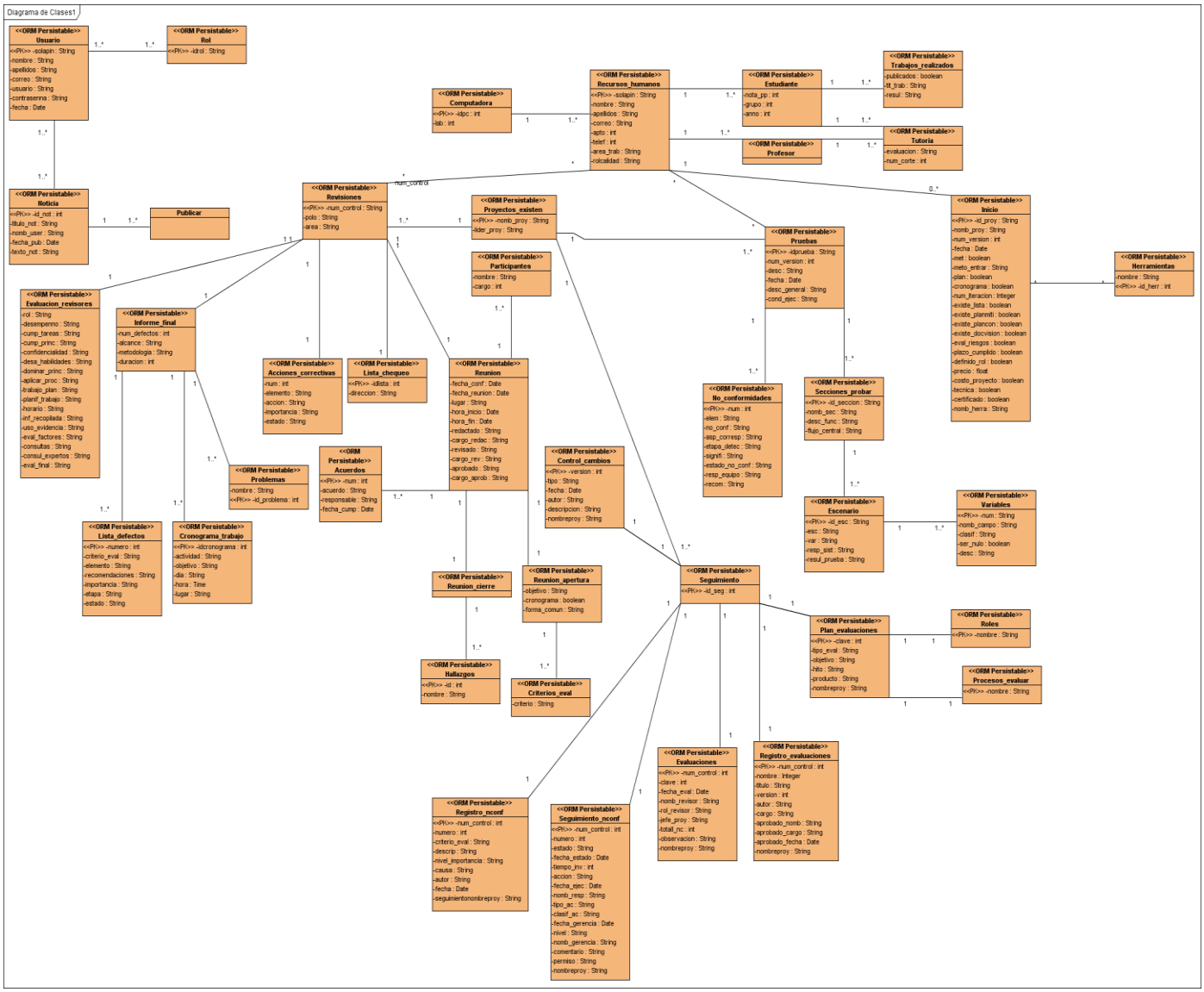

Fig. 8. Diagrama de clases persistentes de la base de datos para el Grupo de Calidad de la facultad 9.

Glosario de Términos

El propósito de este glosario es definir con exactitud y sin ambigüedad algunas terminologías manejadas en el documento.

1. **Atributo compuesto:** pueden ser divididos en pequeñas partes, las cuales representan atributos básicos con existencia independiente.
2. **Atributo multivaluado:** un atributo es multivaluado cuando para una misma entidad puede tomar varios valores diferentes, con independencia de los valores que puedan tomar el resto de los atributos.
3. **API:** (Application Programming Interface). Es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.
4. **CASE:** (Computer-Aided Software Engineering, Ingeniería de Software Asistida por Ordenador). Diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.
5. **GNU:** Proyecto para la creación de un sistema operativo de libre distribución. Como núcleo del sistema se suele usar el de Linux, dando lugar al conjunto llamado "GNU/Linux".
6. **GSSAPI:** es un API para usar sistemas de seguridad de forma genérica. GSSAPI es un estándar de la IETF que aborda el problema que implica la existencia en la actualidad de muchos sistemas de seguridad similar pero incompatible. El sistema de seguridad más común que se utiliza es Kerberos.
7. **IP:** (Internet Protocol, Protocolo de Internet). Protocolo no orientado a conexión usado tanto por el origen como por el destino para la comunicación de datos a través de una red de paquetes conmutados.
8. **ISO:** Organización Internacional para la Estandarización, nacida tras la Segunda Guerra Mundial el 23 de febrero de 1947, es el organismo encargado de promover el desarrollo de normas internacionales de fabricación, comercio y comunicación para todas las ramas industriales a excepción de la eléctrica y la electrónica. Su función principal es la de buscar la estandarización de normas de productos y seguridad para las empresas u organizaciones a nivel internacional.

9. **Java:** Lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90.
10. **JDBC:** (Java Database Connectivity) es un API para trabajar con bases de datos desde Java, independientemente de la base de datos a la que se acceda.
11. **Licencia BSD:** La licencia BSD es la licencia de software otorgada principalmente para los sistemas BSD (*Berkeley Software Distribution*). Es una licencia de software libre permisiva, tiene menos restricciones en comparación con otras como la GPL estando muy cercana al dominio público. La licencia BSD al contrario que la GPL permite el uso del código fuente en software no libre.
12. **Licencia GPL:** La licencia GPL se aplica al software de la FSF (Free Software Foundation) y el proyecto GNU y otorga al usuario la libertad de compartir el software y realizar cambios en él. Dicho de otra forma, el usuario tiene derecho a usar el programa, modificarlo y distribuir las versiones modificadas pero no tiene permiso de realizar restricciones propias con respecto a la utilización de ese programa modificado.
13. **LDAP:** (Lightweight Directory Access Protocol). Es un protocolo a nivel de aplicación que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red.
14. **Linux:** Núcleo o kernel del sistema operativo libre denominado GNU/Linux. Lanzado bajo la licencia pública general (GPL - General Public License) de GNU y desarrollado gracias a contribuciones provenientes de todo el mundo.
15. **SQL:** (Structured Query Language). Es un lenguaje de acceso a las Bases de Datos, permite especificar todas las operaciones sobre la base de datos como por ejemplo: Inserción, Borrado, Actualización. Utiliza características de álgebra y cálculo relacional permitiendo de esta forma realizar consultas a la base de datos de forma sencilla.
16. **SSL:** El protocolo SSL es un sistema diseñado y propuesto por Netscape Communications Corporation. Proporciona sus servicios de seguridad cifrando los datos intercambiados entre el servidor y el cliente con un algoritmo de cifrado simétrico.
17. **SSPI:** (Security Support Provider Interface). Es una API utilizada por los sistemas Microsoft Windows para realizar una variedad de operaciones relacionadas con la seguridad.

18. **TCP:** (Transmission Control Protocol, Protocolo de Control de Transmisión). Protocolo de comunicación orientado a conexión y fiable ubicado en el nivel de transporte del modelo OSI, es uno de los protocolos fundamentales en Internet.
19. **Tupla:** Se define como una función finita que mapea (asocia unívocamente) los nombres con algunos valores. Una tupla es una secuencia ordenada de objetos.
20. **URL:** (Uniform Resource Locator, Localizador Uniforme de Recurso): Es una secuencia de caracteres, de acuerdo a un formato estándar, que se usa para nombrar recursos como documentos e imágenes en Internet para su localización.
21. **UML**: (Unified Modeling Language, Lenguaje Unificado de Modelado). Es un lenguaje de modelado de sistemas de software.
22. **UNIX:** Es un sistema operativo de tiempo compartido, controla los recursos de una computadora y los asigna entre los usuarios. Permite a los usuarios correr sus programas.