Universidad de las Ciencias Informáticas

Facultad 10

Título: Diseño de Bases de Datos para la Intranet 2.

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

Autores: Yeneirys Hernández Castellanos.
Wendysh Pérez García.

Tutores: Ing. Dunia Suárez Ferreiro.
Ing. Luis Enrique Ramírez Noy.

Ciudad de la Habana, julio 2007
"Año 49 de la Revolución"

"Todos los triunfos nacen cuando nos atrevemos a comenzar".

Eugene Ware

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____ _____

Yeneirys Hernández Castellanos Wendysh Pérez García

_____ _____

Ing. Dunia Suárez Ferreiro Ing. Luis Enrique Ramírez Noy

Datos de Contacto

Luis Enrique Ramírez Noy (noy@uci.cu): Graduado de Ing. Telecomunicaciones y Electrónica en el año 2004 en el ISPJAE. Posee categoría docente de Profesor Instructor y cursa la maestría de Telemática. Ha impartido las asignaturas Sistemas Operativos, Máquinas Computadoras I y II y Teleinformática I y II en la Facultad 6 desde el curso 2004-2005. Ha presentado ponencias en eventos científicos nacionales. Se desempeña como Líder del Proyecto Automatización de LIMS con el CIB.

Dunia Suarez Ferreiro (dsuarezf@uci.cu): Graduada de Ing. Ciencias de la Computación en el año 2005 en la UH. Posee categoría docente de Profesor Instructor Recien Graduado. Ha impartido las asignaturas de Programación II, Programación IV, Graficos por Computadora, XML, HTML y CSS en la Facultad 10 desde el curso 2005-2006. Se desempeña como Líder del Proyecto Intranet 2.

Agradecimientos

El presente trabajo representa la culminación de un largo trayecto de estudios y sacrificios. Por esta razón es una parte importante de nuestras vidas y por eso queremos agradecer a todos los que contribuyeron con su realización.

A nuestra Revolución y a nuestro Comandante en Jefe Fidel por hacer posible que formáramos parte de este gran proyecto.

A nuestros Profesores y Tutores.

A Yeisel.

Yeneirys:

A mi Comandante Fidel Castros Ruz.

A la memoria de mi papá que en el poco tiempo que estuvo conmigo me enseñó a ser una persona correcta.

A mima que ha puesto todo su empeño porque nunca me falte amor lo que me ha convertido en una buena persona.

A Geñito por el cariño y apoyo que me ha dado siempre.

A Yesy mi hermanita buena.

A Mahe que ha sido mi estrellita.

A toda mi familia que tanto me quiere y ayuda.

A Yilian y Yuriel que fueron las personas más importantes para mi en la universidad.

A mis buenos amigos.

Wendysh:

A mis padres y especialmente a mi abuela, por su amor, apoyo y dedicación.

A mi novio Deybis, por su apoyo.

A mis amigos y en especial a Yeisel y Yuri por estar siempre presentes, en los buenos y malos momentos.

A mi tía Marianela por toda la ayuda que me ha brindado.

A toda mi familia por estar siempre a mi lado brindándome apoyo.

A todos mis educadores, por contribuir a mi formación.

Dedicatoria

Dedicamos este trabajo de diploma a todas las personas que de una forma u otra han aportado su granito de arena; brindándonos las enseñanzas, el valor y la confianza que necesitábamos para formarnos como profesionales.

A todos ustedes…

Resumen

Las Intranets Corporativas son herramientas excelentes en la prestación de servicios que facilitan la comunicación e intercambio de información de interés para los miembros de una institución, razón por la cual se ha hecho común que todas las organizaciones quieran contar con una.

En la Universidad de las Ciencias Informáticas (UCI) está en marcha un proyecto para implementar una Intranet Corporativa, donde estén representadas todas las áreas funcionales de dicha entidad. Para manejar la información relacionada con estas áreas se requiere confeccionar una base de datos para cada una de ellas, incluyendo la del servicio de noticias, efemérides y avisos que ofrece la Intranet actual; la que presenta limitaciones a la hora de actualizar y procesar nuevos datos.

Ante la necesidad de contar con aplicaciones que permitan almacenar, procesar y responder a las diferentes peticiones de la comunidad universitaria en general, se hacen las bases de datos Noticia, desarrollada hasta el nivel físico y Facultad que quedó a nivel de modelo conceptual, en espera de aprobación para continuar su implementación. Para su diseño y desarrollo fueron elegidos el Visual Paradimg y el Sistema Gestor de Base de Datos (SGBD), relacional: PostgreSQL.

En el presente documento se exponen una serie de aspectos relacionados con el tema de Base de Datos, que sirven de punto de partida para la selección de los elementos que conforman la propuesta. Así como su descripción, fundamentación y validación.

Palabras clave: Intranet Corporativa, Base de Datos, Sistema Gestor de Base de Datos, PostgreSQL.

Índice

Índice de Tablas

Introducción

La información ha sido siempre un recurso muy valorado. Desde las primeras civilizaciones se han archivado, manipulado y creado grandes cantidades de información en bibliotecas, monasterios o por determinadas personas. En esos lugares se atesoraban conocimientos científicos, de medicina, literatura, etc.

Sin embargo, nunca como ahora, ha tenido tanto valor la información. La explosión de las tecnologías de redes, la aparición de novedosos servicios, el acercamiento entre culturas y las necesidades de comunicación que estas generan, así como el fenómeno de la Globalización ; han hecho que en unos pocos años la información se haya convertido en un eslabón fundamental para el desarrollo de cualquier sociedad .

Millones de bits de información de todo tipo se generan cada día en todo el mundo e igual cantidad es consumida. Este enorme torrente requiere evidente organización y control, tanto para poder ofrecerla como administrarla. Ello ha hecho aparecer lo que se conoce como Sistemas de Información.

Los Sistemas de Información, son manejados a nivel de entidades, empresas u organizaciones. Son un conjunto de elementos (*hardware*, *software* y recursos humanos) que interactúan entre sí con el fin de apoyar determinadas actividades. Los Sistemas de Información realizan cuatro operaciones básicas sobre la información: la entrada, el almacenamiento, el procesamiento, y su presentación.(ANÓNIMO 2006)

Sin embargo el soporte lógico que hace operables los Sistemas de Información y cooperan en la consistencia de los datos que se manejan, así como el acceso a los mismos son las Bases de Datos (BD).

Las Bases de Datos son el soporte fundamental de empresas e instituciones en sus cotidianos procesos de gestión, ya que manejan gran cantidad de información valiosa y diversa, que necesitan organizar para tener disponible en el momento de tomar decisiones y brindar servicios.

Las Bases de Datos son componentes esenciales de los Sistemas de Información, pero son conocidas de mucho antes en las ciencias de la informática. Las bases de datos digitales o al menos automatizadas aparecieron a finales de los años 60 en el mercado del software y constantemente han estado cambiando, siendo mejoradas y utilizadas en más aplicaciones.

El rápido desarrollo de las tecnologías de Bases de Datos, así como de los Sistemas de Información, está condicionado por la expansión de Internet como la red de redes y la "popularización" de las *intranets*. Cualquier empresa puede tener una intranet, que no es más que su propia red de datos interna. De hecho, la implantación de una intranet posibilita la prestación de servicios que conllevan al aumento de la eficiencia en la gestión interna y de la propia actividad de la empresa.

Cuando dicha intranet brinda servicios que requieren el manejo de datos persistentes, necesarios a largo o mediano plazo, puede clasificarse como Intranet Corporativa. En ese punto requerirá de un Sistema de Información y de un soporte confiable de Bases de Datos.

En las Intranets Corporativas se debe manejar tanto la accesibilidad a la información como su seguridad, y al mismo tiempo su consistencia. Muchos de los servicios de este tipo de intranets, pueden brindarse a través de www en la red de redes, por tanto el control de acceso a recursos, y la calidad de los servicios son aspectos vitales.

Por otro lado, internamente se ha de preservar información y documentación referente a la entidad: normativas internas, protocolos de actuación, manuales de referencia, circulares. A través de una Intranet se pueden organizar sistemas de colaboración y flujos de trabajo, dígase elaboración de proyectos y presupuestos, agendas de trabajo, gestión de documentos de producción. También ayudan en el establecimiento de canales comunicación interna como boletines de información, mensajería, foros, sistemas de mensajería.

Por lo tanto, en la concepción de un Sistema de Información para Intranet Corporativa se ha de prestar gran atención a la seguridad, fiabilidad, disponibilidad y consistencia de la información, es decir a sus Bases de Datos.

En la UCI se ha alcanzado un alto nivel de desarrollo en cuanto a tecnologías de la información y las comunicaciones, se cuenta con una Intranet que actualmente es el espacio de información con funcionamiento constante más visitado por la comunidad universitaria dentro de las opciones informativas existentes. Es una herramienta dinámica y flexible que está a disposición y en función de la comunidad universitaria que debe implementar nuevas formas de retroalimentación e intercambio con los usuarios. En la Intranet existen un conjunto de sitios y servicios que contribuyen a organizar la vida diaria de la universidad,

mantener informados a todos de cuanto acontece tanto en el ámbito local, como nacional e internacional, brindar orientaciones precisas de las tareas a que se convocan, prestar servicios vitales como la reservación del pase de los estudiantes, publicar materiales de apoyo a la docencia, mantener servicios de media en línea para la superación de la comunidad universitaria, servicios de mensajería y correo electrónico, entre muchos otros.

La Intranet fue creada en los inicios de la UCI hace cuatro años. Debido a que todos los años ingresan a la universidad aproximádamente 2 000 estudiantes de todo el país, además de trabajadores docentes y de servicios, ha aumentado hasta aproximádamente 15 000, el número de usuarios que acceden a la Intranet desde la residencia o desde el espacio docente; lo que ha provocado problemas en la autogestión de la Intranet y congestión en los servidores. Por otra parte, no existen bases de datos que almacenen, procesen y muestren la información que se solicita de forma íntegra y rápida. Es así como ha surgido la necesidad de implementar un conjunto de bases de datos que resuelvan las deficiencias existentes.

En este contexto surge el proyecto Intranet 2 con la premisa de crear una Intranet con un nuevo diseño, arquitectura y estrategia de gestión de la información. Además de regirse por las normas de una Intranet Corporativa, orientada a servicios que en la actualidad, son las que añaden mayor valor a las organizaciones en sus procesos de negocio.

La nueva Intranet va a contar con gran cantidad de información donde van a estar representadas todas las estructuras administrativas de la UCI, las organizaciones políticas y de masas, los grupos científicos y proyectos de investigación, áreas y proyectos productivos, así como los miembros de la comunidad universitaria. Además prestará un número considerable de servicios de información.

Una Intranet Corporativa debe ser una herramienta eficiente y capaz de manejar gran cantidad de información de manera descentralizada y de forma segura, además de realizar el mayor número de servicios. Para ello, debe estar respaldada por softwares que se encarguen de realizar cada uno de estos procesos y es aquí donde las bases de datos juegan un rol fundamental, surgiendo como **problema científico** el ¿Cómo crear bases de datos que permitan gestionar de forma eficiente toda la información que se manejará en las Facultades y en el servicio de Noticias de la Intranet Corporativa de la UCI?

El **objeto de estudio** es el análisis y diseño de los servicios definidos en la arquitectura de la información de la Intranet Corporativa de la UCI, teniendo como **campo de acción** las bases de datos para las Facultades y publicación de Noticias de la Intranet Corporativa.

Por lo que el **objetivo general** consiste en realizar el diseño de las bases de datos para la gestión de la información correspondiente al servicio de Noticias y las Facultades que forman parte de la Intranet Corporativa de la UCI.

Se plantearon las **preguntas científicas** siguientes:

¿Cuáles son los criterios apropiados para seleccionar un SGBD y un modelo de datos adecuado para realizar el diseño de las bases de datos para la Intranet Corporativa de la UCI?

¿Cómo hacer una propuesta de diseño de BD adecuada para las BD de la Intranet Corporativa de la UCI?

¿Cómo realizar la validación para el diseño de las bases de datos para la Intranet 2?

Para dar cumplimiento a las **preguntas científicas** se definieron una serie de **tareas**:

1. Realizar búsquedas de información en diversas fuentes sobre SGBD y modelos de datos, estudiar sus características, así como seleccionar uno de los SGBD y un modelo de datos de los estudiados para el diseño de las Bases de Datos para la Intranet 2.

2. Realizar una propuesta de diseño para las Bases de Datos de la Intranet 2.

3. Aplicar reglas de integridad, normalizaciones y mecanismos de seguridad a las Bases de Datos de la Intranet 2.

Lo esperado de este trabajo como aplicación práctica es, crear las bases de datos para el servicio de Noticias y para la gestión de la información de las Facultades que van a servir de soporte a la Intranet que es la herramienta que va a brindar a los usuarios los servicios equivalentes a los que ya se realizan en la actualidad y a otros nuevos.

Para la realización de la investigación se utilizaron los métodos teóricos que proporcionaron la interpretación y fundamentación de los diferentes datos recopilados a lo largo de este proceso y los empíricos que posibilitaron la recogida de información relacionada con el objeto de estudio que se está investigando.

El **método teórico** utilizado fue el **analítico_sintético** que permitió analizar teorías, documentos sobre los SGBD más reconocidos por sus prestaciones y utilizados en la actualidad, así como los modelos de datos, buscar sus características fundamentales, ventajas y desventajas, para a partir de ahí seleccionar el SGBD y el modelo de datos adecuado para realizar la propuesta de diseño de las bases de datos para el servicio de Noticias y para la gestión de la información de las Facultades. Con la base de datos de Noticias se van a eliminar los problemas de redundancia e inconsistencias de la información de la Intranet actual y con la de Facultad tanto trabajadores como estudiantes van a contar con la información relacionada con las Facultades.

Otro **método teórico** utilizado fue el **análisis histórico_lógico** el cual permitió el estudio de la trayectoria de diferentes SGBD y de los modelos de datos, cuándo fueron desarrollados y por qué personalidades, así como su evolución en el tiempo. Se realizaron comparaciones entre las versiones antiguas y las más actuales de los SGBD de Software libre lo que facilitó la obtención de elementos de interés para decidirse por uno u otro para el diseño de las bases de datos Noticia y Facultad para la Intranet Corporativa de la UCI.

El **método empírico** utilizado fue la **entrevista** realizada a los especialistas en información y a los periodistas de la UCI, encargados de realizar la arquitectura de la información para su posterior utilización en la realización del diseño de las bases de datos para la Intranet Corporativa de la UCI. Las entrevistas se hicieron con el objetivo de obtener los elementos necesarios para definir cada una de las entidades con sus respectivos atributos y relaciones, así como las funcionalidades que estas debían tener.

El presente trabajo de diploma está estructurado en tres capítulos. Cada uno consta de introducción, desarrollo formado por varios epígrafes y conclusiones.

El primer capítulo titulado “Fundamentación teórica” consiste en un análisis de las tendencias actuales de sistemas similares y un estudio crítico - valorativo de las diferentes tecnologías, metodologías y estrategias de diseño de bases de datos.

El segundo capítulo nombrado “Descripción y análisis de la solución propuesta” consiste en realizar la propuesta del sistema. En el mismo se hace la descripción y fundamentación de la arquitectura, un análisis de optimización de consultas, se muestran los requisitos funcionales y no funcionales del sistema propuesto, el modelo de objetos o diagrama de clases persistentes obtenido a partir del diagrama de clases del diseño, la descripción de las clases, diseño de las Bases de Datos, el diagrama Entidad Relación de las Bases de Datos, y la descripción de las tablas.

En el tercer y último capítulo titulado “Validación del diseño realizado” se hace la validación teórica del diseño que incluye análisis de integridad, normalización de las bases de datos, el análisis de redundancia de información y el análisis de la seguridad, así como la trazabilidad de las acciones realizadas en dichas bases de datos.

Esta investigación está respaldada por una serie de materiales encontrados en internet. Entre los que se encuentran artículos, páginas web, libros, trabajos de diploma, tesis doctorales y en los que se abordan temas sobre bases de datos. Además está apoyada en las teleclases que se imparten en la UCI de Sistemas de Bases de Datos.

Capítulo 1: Fundamentación teórica

En el proceso de construcción de todo sistema informativo automatizado, el diseño de la BD ocupa un lugar importante, a tal punto que ésta puede verse como un proceso relativamente independiente dentro del diseño del sistema y compuesto por una serie de etapas. Es por ello que resulta de interés el estudio de los problemas relacionados con el diseño de las bases de datos y la modelación de la información.

Con el propósito de realizar un conjunto de bases de datos para la Intranet Corporativa de la UCI se hizo un estudio y análisis de diferentes redes empresariales similares existentes en Cuba y en el mundo con el objetivo de ganar conocimientos en cuanto a funcionalidades y beneficios que reporta el uso de estas.

También se realizó una investigación de los modelos de datos, metodologías, tecnologías así como de estrategias que se utilizan para el diseño de bases de datos.

1.1 Intranets Corporativas

La implementación de las Intranets Corporativas a nivel mundial se ha generalizado. Muchas organizaciones reconocen la necesidad de disponer de una infraestructura que permita a sus miembros acceder de forma sencilla e inmediata a toda la información que necesitan para realizar su trabajo de forma eficiente. Estas instituciones tienen soportadas sus redes empresariales (Intranets Corporativas) sobre tecnologías de Microsoft, software privativo y utilizan para desarrollar sus bases de datos el SGBD SQL Server.

En Cuba muy pocas empresas cuentan con Intranets Corporativas y las que han logrado incorporar estas valiosas herramientas también han utilizado software propietario para su implementación y también utilizan como SGBD al SQL Server.

La UCI, que es una universidad que ha alcanzado un gran desarrollo en cuanto a tecnologías de la información, aún no tiene su Intranet Corporativa, debido a que la instauración de la misma es una tarea de varios estudiantes y profesores donde cada uno aporta una parte importante desde su rol. Uno de los roles

es el de diseñador principal de bases de datos. Para el diseño de las bases de datos de la Intranet Corporativa de la universidad se va a utilizar el SGBD PostgreSQL.

1.2 Bases de datos

La expresión Bases de Datos es muy antigua, aparece por primera vez a mediados de los años sesenta en una conferencia organizada por el System development Corporation en Santa Mónica (California) para referirse a "un almacén de todos los datos de interés y valor para los usuarios del sistema de información". Debido a las exigencias de los usuarios y el desarrollo constante de las tecnologías este concepto ha evolucionado. Según el Comité de Terminología de la FID (Federación Internacional de Información y Documentación), una base de datos es "un conjunto de datos homogéneos, ordenados de una forma determinada, que se presenta normalmente de forma legible por ordenador y se refiere a una organización, materia o problema determinado" (ANÓNIMO 2006)

En el mundo se han hecho varios e importantes descubrimientos sobre las bases de datos, se han desarrollado diferentes clasificaciones para lograr una mejor utilización de estas tecnologías que va a estar basada en la naturaleza de la información y en la forma en que esta va a ser recuperada.

1.2.1 Clasificación de las bases de datos en cuanto al modo de administración

Una clasificación importante de las bases de datos es en cuanto al modo de administración de datos en el que se analizan las **bases de datos jerárquicas,** que como su nombre lo indica almacenan la información en una estructura de jerarquía. Una de las principales limitaciones de este modelo es su incapacidad de representar eficientemente la redundancia de datos. Las **bases de datos de red** se diferencian ligeramente del jerárquico debido a su modificación en el concepto de nodo pues permiten que un mismo nodo tenga varios padres. Fue una gran mejora con respecto al modelo jerárquico, ya que ofrecía una solución eficiente al problema de redundancia de datos. Las **bases de datos orientadas a objetos** incorporan todos los conceptos importantes del paradigma de objetos: encapsulación (propiedad que permite ocultar la información al resto de los objetos, impidiendo así accesos incorrectos o conflictos), herencia (propiedad a través de la cual los objetos heredan comportamientos dentro de una jerarquía de clases). Polimorfismo (propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos). Las **bases**

de datos documentales permiten la indexación a texto completo y en líneas generales realizar búsquedas más potentes. Las **bases de datos deductivas** permiten hacer deducciones a través de inferencias, basadas principalmente en reglas y hechos que son almacenados en la base de datos(PERRAUT 1997).

El modelo relacional está caracterizado por su única estructura de datos llamada tabla. Las tablas, unidades bidimensionales, están formadas por filas (tuplas) y columnas homogéneas. Las filas representan las entidades (registros) y las columnas los atributos (campos). Los datos almacenados en las tablas se relacionan a través de operaciones basadas en el álgebra y en el cálculo relacional, que implican operaciones lógicas entre conjuntos. Básicamente estas operaciones son tres: la selección que crea un subconjunto de las filas de las tablas que cumplen con una o varias condiciones dadas, creándose una nueva tabla, normalmente temporal; la proyección que es una operación similar a la selección pero en lugar de actuar sobre las filas lo hace sobre las tablas; y por último, la unión que crea una nueva tabla a partir de otras dos o más tablas(PERRAUT 1997).

El modelo de **bases de datos relacionales** es el que se escoge para el diseño de las bases de datos de la Intranet Corporativa de la UCI. Se selecciona por su sencillez, la cual propicia una fácil comprensión a los diseñadores. Cuando se usa este modelo el usuario solo tiene que especificar cuales registros quiere procesar y de cómo hacerlo se encarga el SGBD, esto está dado por la independencia de datos debido a la separación entre el modelo (lógico) y la implementación (física). La utilización del lenguaje SQL proporciona a los que lo utilizan, la facilidad de poder actuar directamente sobre el SGBD; que es la aplicación informática que permite crear, mantener y explotar dichos datos, a través de un conjunto coordinado de programas y procedimientos que suministran los medios necesarios para describirles y manejarles. SQL también puede ser usado desde otros lenguajes de programación de tercera generación, tales como C, para poder acceder a los datos de la base de datos y usarlos para cualquier fin en el programa. Incluye una serie de reglas de integridad y descomposiciones que permiten obtener un buen diseño, libre de redundancias e inconsistencias. Además ofrece la posibilidad de escoger el SGBD de acuerdo a las necesidades que se tengan, pues en la actualidad existen varios SGBD relacionales con excelentes características.

1.3 Sistemas Gestores de Bases de Datos

Los **Sistema de Gestión de Base de Datos** son un tipo de software muy específico dedicado a servir de interfaz entre la base de datos y las aplicaciones que la utilizan (el usuario). Se compone de un lenguaje de definición de datos (LDD), de un lenguaje de manipulación de datos (LMD) y de un lenguaje de consulta (LC). Tiene como propósito general manejar de manera clara, sencilla y ordenada un conjunto de información. Se destacan entre sus funciones la definición de los datos en los distintos niveles de abstracción, la manipulación de los datos, el mantenimiento de la integridad de la BD y el control de la seguridad de dichos datos.(ANÓNIMO 2006)

1.31 Objetivos

Existen distintos objetivos que deben cumplir los SGBD:

- **Abstracción de la información**. Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos; da lo mismo si una base de datos ocupa uno o cientos de archivos, este hecho se hace transparente al usuario. Así, se definen varios niveles de abstracción.
- **Independencia**. La independencia de los datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.
- **Redundancia mínima**. Un buen diseño de una base de datos logrará evitar la aparición de información repetida o redundante. De entrada, lo ideal es lograr una redundancia nula; no obstante, en algunos casos la complejidad de los cálculos hace necesaria la aparición de redundancias.
- **Consistencia**. En aquellos casos en los que no se ha logrado esta redundancia nula, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea.
- **Seguridad**. La información almacenada en una base de datos puede llegar a tener un gran valor. Los SGBD deben garantizar que esta información se encuentre asegurada frente a usuarios malintencionados que intenten leer información privilegiada; frente a ataques que manipulen o destruyan la información o simplemente ante las torpezas de algún usuario autorizado pero despistado. Normalmente, los SGBD disponen de un complejo sistema de permisos a usuarios y grupos de usuarios que permiten otorgar diversas categorías de permisos.

- **Integridad**. Se trata de adoptar las medidas necesarias para garantizar la validez de los datos almacenados; es decir, se trata de proteger los datos ante fallos de *hardware*, datos introducidos por usuarios descuidados o cualquier otra circunstancia capaz de corromper la información almacenada.
- **Respaldo y recuperación**. Los SGBD deben proporcionar una forma eficiente de realizar copias de seguridad de la información almacenada en ellos y de restaurar a partir de estas copias los datos que se hayan podido perder.
- **Control de la concurrencia**. En la mayoría de entornos lo más habitual es que sean muchas las personas que acceden a una base de datos, bien para recuperar información, bien para almacenarla. Es también frecuente que dichos accesos se realicen de forma simultánea. Así pues, un SGBD debe controlar este acceso concurrente a la información, que podría derivar en inconsistencias.
- **Tiempo de respuesta**. Lógicamente, es deseable minimizar el tiempo que el SGBD tarda en darnos la información solicitada y en almacenar los cambios realizados.

1.3.2 Características de los principales SGBD

En la actualidad existen numerosos SGBD entre los que se pueden mencionar DB2, Informix, InterBase, SQL Server y Oracle los cuales se encuentran en el grupo de software privativo y otros como FireBird, MaxDB, SQLite, MySQL y PostgreSQL que pertenecen al grupo de software libre. Entre los SGBD más utilizados mundialmente se encuentran SQL server y Oracle ambos de código cerrado que ofrece buenas prestaciones a los usuarios pero tienen precios elevados, fundamentalmente Oracle.

En nuestro país existe la imperiosa necesidad de utilizar las tecnologías de software libre por cuestiones políticas, sociales y económicas. En algunas instituciones como la UCI se están utilizando en algunas facultades los SGBD MySQL y PostgreSQL ambos de código abierto (open source).

A continuación se ofrece una breve descripción de algunos de ellos con sus características fundamentales.

SQL Server

SQL Server es un sistema de bases de datos relacionales basado en lenguaje SQL, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. Entre sus

características figura el soporte de transacciones, escalabilidad, estabilidad y seguridad. (*Microsoft SQL Server* . 2007)

La versión más reciente y completa en estos momentos es Microsoft SQL Server 2005.

Oracle

Oracle es un sistema de bases de datos relacional fabricado por Oracle Corporation. Se considera a Oracle como uno de los sistemas de bases de datos más completos.

Oracle surge a finales de los 70 bajo el nombre de Rational Software a partir de un estudio de SGBD de George Koch.

Oracle es básicamente una herramienta cliente/servidor para la gestión de bases de datos. 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.(Oracle. Introducion. Monografias.com. 2007)

Es considerado como uno de los sistemas de bases de datos más completos destacando su soporte de transacciones, estabilidad, escalabilidad, además es multiplataforma. La versión más reciente es la 10g Release 2.

MySQL

MySQL es un sistema de gestión de bases de datos, multihilo y multiusuario. MySQL es una idea originaria de la empresa open source MySQL AB y actualmente está en la versión 5.0.(MySQL. 2007)

MySQL Database Server es la base de datos de código fuente abierto más usada del mundo. Su ingeniosa arquitectura lo hace extremadamente rápido y fácil de personalizar.(ANÓNIMO 2006)

Su principal objetivo de diseño fue la velocidad tanto al conectar con el servidor como al realizar consultas. Otra característica importante es que consume muy pocos recursos, tanto de CPU como de memoria. Es bueno en estabilidad y ofrece facilidad de despliegue.

PostgreSQL

El sistema de gestión de base de datos PostgreSQL nace en la universidad de Berkeley - California, en los años 80, como un proyecto académico y actualmente se encuentra en la versión 8.1, siendo permanentemente mantenido por la comunidad Open Source. La coordinación del desarrollo del PostgreSQL es realizada por el Global Development Group, que está formada por un amplio grupo de desarrolladores alrededor del mundo, que permite al PostgreSQL evolucionar constantemente en cuanto a la corrección de errores y la implementación de nuevas funcionalidades. Por su arquitectura de diseño, escala muy bien al aumentar el número de CPUs y la cantidad de RAM. (ANÓNIMO 2006)

Tabla 1. Comparación MySQL vs PostgreSQL

	MySQL	PostgreSQL
Soporte	MySQL AB	Comunidad de desarrolladores
Licencia	GPL y Licencia Comercial	Licencia Berkeley (BSD)
Procedimientos almacenados y disparadores	Basado en el estándar SQL: 2003.	Lenguaje de procedimientos del Postgres es PL/pgSQL[1]
Índices	Una columna, Múltiples columnas, Índices de llave primaria.	
Tipo de datos	Soporta la mayoría de los tipos de datos estándares.	
		Soporte para tipos de datos definidos por el usuario
Interfaz con la BD	Soporta los estándares ODBC y JDBC. Acceso desde C/C++, Java, Perl, Python, PHP, .Net	
Herramientas gráficas	MySQL Query Browser, MySQL Administrator, MySQL Front, phpMyAdmin	Pgadmin III, Phppgadmin
Plataformas soportadas	Windows, Linux, FreeBSD, MacOSX, y Unix.	
Almacenamiento de datos	MyISAM, InnoDB, Memory, NDB	Postgres Storage System
Integridad referencial	Llave primaria, Llave externa.[2]	
ACID	Ambos son ACID compatibles.	
Transacciones	Ambos soportan transacciones.[3]	
Data Warehouse	Soporta Cubos OLAP	No soporta bases de datos multidimencionales

Nota (1): PostgreSQL además soporta otros lenguajes de procedimientos: PL/Tcl, PL/Perl, L/Python.

Nota (2): MySQL solamente tiene soporte para integridad referencial para las el tipo de tabla InnoDB.

Nota (3): MySQL unicamente tiene soporte de transacciones para el tipo de tablas InnoDB.

Tanto MySQL como PostgreSQL son SGBD relacionales muy buenos. El hecho que sean software libre ya constituye una gran ventaja. Ambos tienen una particularidad deseable en cualquier SGBD que es la de ser ACID (atómico, consistente, aislada, durable) compatibles lo que quiere decir que permiten transacciones y esas transacciones van a garantizar que la información se guarde de forma atómica lo que significa que un bloque de transacciones es indivisible, se guarda toda la información o no se guarda nada. Sin embargo, hay peculiaridades interesantes en PostgreSQL que pueden resultar atrayentes para cualquier diseñador de base de datos aunque no tenga mucha experiencia. Una de ellas es que su licencia es mucho más simple que la de MySQL, esta permite cualquier uso siempre que se incluya una copia de la licencia en el producto final. Mientras MySQL utiliza para escribir sus procedimientos almacenados y disparadores el lenguaje SQL que tiene la dificultad que cada sentencia se ejecuta de manera independiente en el servidor, PostgreSQL permite agrupar un bloque de sentencias en el servidor y ejecutar cálculos. Al utilizar PostgreSQL se disipa cualquier dificultad a la hora de seleccionar el tipo de tabla que se va a utilizar debido a que cuenta con un único mecanismo de almacenamiento. Además posee una gran escalabilidad lo cual se demuestra con su gran capacidad para ajustarse al número de CPUs y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole capaz de soportar una mayor cantidad de peticiones simultáneas de manera correcta. Es bastante fiable debido a que su código está bien probado y cada versión tiene menos errores. Usa una estrategia de almacenamiento de filas llamada MVCC (Acceso Concurrente Multiversión) para conseguir una respuesta mucho mejor en ambientes de grandes volúmenes de datos.

Por estas razones, se seleccionó a PostgreSQL como SGBD para desarrollar las bases de datos de la intranet corporativa de la UCI.

1.4 Metodología de diseño de bases de datos

El diseño de bases de datos es un proceso complejo para el cual es necesario tomar decisiones a distintos niveles. Para controlar esta complejidad se desglosa el problema en otros más pequeños, es decir en

subproblemas y se resuelven cada uno de estos problemas de forma independiente, utilizando técnicas específicas. Así el diseño de una base de datos se descompone en diseño conceptual, diseño lógico y diseño físico.

Los requerimientos de usuario son el punto de partida para el diseño conceptual y el resultado es el esquema conceptual de la base de datos, cuyo objetivo es describir el contenido de información de la base de datos y no las estructuras de almacenamiento que se necesitarán para manejar dicha información.

Un **esquema conceptual** es una descripción de alto nivel de la estructura de la base de datos, independientemente del SGBD que se vaya a utilizar para manipularla. Un **modelo conceptual** es un lenguaje que se utiliza para describir esquemas conceptuales.

El diseño lógico parte del esquema conceptual y da como resultado un esquema lógico. Un **esquema lógico** es una descripción de la estructura de la base de datos en términos de las estructuras de datos que puede procesar un tipo de SGBD. Un **modelo lógico** es un lenguaje usado para especificar esquemas lógicos (modelo relacional, modelo de red, etc.). El diseño lógico depende del tipo de SGBD que se vaya a utilizar, no depende del producto concreto.

El diseño físico parte del esquema lógico y da como resultado un esquema físico. Un **esquema físico** es una descripción de la implementación de una base de datos en memoria secundaria: las estructuras de almacenamiento y los métodos utilizados para tener un acceso eficiente a los datos. Por ello, el diseño físico depende del SGBD concreto y el esquema físico se expresa mediante su lenguaje de definición de datos. (MARQUÉS 2001)

Es de vital importancia seguir adecuadamente la metodología de diseño de bases de datos porque de su correcta aplicación dependen que se obtengan buenos resultados. Esta metodología proporciona una secuencia lógica de pasos que deben ser realizados en cada una de las etapas (diseño conceptual, diseño lógico, diseño físico).

1.5 Estrategias de diseño de una base de datos

Hay varias estrategias a seguir para realizar el diseño de una base de datos, puede utilizarse la estrategia de abajo hacia arriba, de arriba a abajo, de dentro a fuera y la estrategia mixta, en dependencia de la información que se tenga o de la complejidad del problema a resolver.

La estrategia **de abajo hacia arriba** parte de todos los atributos y los va agrupando en entidades y relaciones. Es apropiada cuando la base de datos es simple, con pocos atributos.

La estrategia **de arriba a abajo** es más apropiada cuando se trata de bases de datos complejas. Se comienza con un esquema con entidades de alto nivel, que se van refinando para obtener entidades de bajo nivel, atributos y relaciones.

La estrategia **de dentro a fuera** es similar a la estrategia de abajo a arriba, pero difiere en que se parte de los conceptos principales y se va extendiendo el esquema para considerar también otros conceptos asociados con los que se han identificado en primer lugar.

La estrategia **mixta** utiliza las estrategias, de abajo a arriba y de arriba a abajo, con un esquema de divide y vencerás. Se obtiene un esquema inicial de alto nivel, se divide en partes y de cada parte se obtiene un subesquema. Estos subesquemas se integran después para obtener el modelo final.(ANDRÉS 2001)

La estrategia empleada para el diseño de las bases de datos de Noticias y de Facultades fue la **de dentro a fuera** porque en ambos casos se partió de las nociones generales, es decir se tenía la idea básica de lo que se quería obtener y a partir de ahí fue fluyendo el modelo hasta estar totalmente completo, quedando representados cada uno de los objetos del mundo real relacionados con el producto deseado.

1.6 Lenguaje de programación para el modelado del sistema

Lenguaje Unificado de Modelado (UML, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.(Lenguaje Unificado de Modelado 2007)

UML se ha convertido en un estándar que tiene las siguientes características:

- Permite modelar sistemas utilizando técnicas orientadas a objetos.
- Permite especificar todas las decisiones de análisis, diseño e implementación, construyéndose así modelos precisos, no ambiguos y completos.
- Puede conectarse con lenguajes de programación (Ingeniería directa e inversa).
- Permite documentar todos los artefactos de un proceso de desarrollo (requisitos, arquitectura, pruebas, versiones, etc.).
- Cubre las cuestiones relacionadas con el tamaño propias de los sistemas complejos y críticos.

UML es un lenguaje para especificar y no un método o un proceso, se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir el lenguaje en el que está descrito el modelo. (Lenguaje Unificado de Modelado 2007)

1.7 Metodologías de desarrollo de software

Una pregunta importante a la hora de hacer un software es qué metodología de software se va a usar. Teniendo en cuenta que tanto en pequeños como en grandes proyectos es necesario basarse en una metodología para que al final el producto satisfaga las necesidades de los usuarios y/o clientes. Debido a que obtener el conjunto de bases de datos que den soporte a todos los contenidos de la Intranet Corporativa de la UCI es una meta que necesita cierto nivel de organización para distribuir el tiempo y las tareas, así como tener un seguimiento y control de la evolución de las mismas para ser cumplida; se hace necesario utilizar una metodología de desarrollo de software.

Entre las metodologías de desarrollo de software más reconocidas se encuentran: RUP que es más adaptable para proyectos de largo plazo, XP para proyectos de corto plazo y MSF que se adapta a proyectos de cualquier dimensión y de cualquier tecnología.

1.7.1 Rational Unified Process (RUP)

El **Proceso Racional Unificado** o RUP (Rational Unified Process), es un proceso de desarrollo de software que junto con el Lenguaje Unificado de Modelado **UML** constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.(ANÓNIMO 2006)

Se caracteriza por la forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo). Es iterativo e incremental, está centrado en la arquitectura y guiado por los casos de uso. Implementa las mejores prácticas en Ingeniería de Software como son: administración de requisitos, uso de arquitectura basada en componentes y verificación de calidad de software.

Incluye artefactos (que son los productos tangibles del software y roles (papel que desempeña una persona en un determinado momento, una persona puede desempeñar distintos roles a lo largo del proceso).

RUP divide el proceso de desarrollo en ciclos, teniendo un producto final al final de cada ciclo. Cada ciclo se divide en fases que finalizan con un hito donde se debe tomar una decisión importante:

- Fase de concepción: se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos.
- Fase de elaboración: se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos.
- Fase de construcción: se concentra en la elaboración de un producto totalmente operativo y eficiente y el manual de usuario.
- Fase de transición: se implementa el producto en el cliente y se entrena a los usuarios. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados.(ANÓNIMO 2006)

1.7.2 Extreme Programing (XP)

Es una de las metodologías de desarrollo de software más exitosas en la actualidad utilizada para proyectos de corto plazo y pequeño equipo. Esta metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto.

Características de XP, la metodología se basa en:

- **Pruebas Unitarias:** se basa en las pruebas realizadas a los principales procesos, de tal manera que adelantándose en algo hacia el futuro, se puedan hacer pruebas de las fallas que pudieran ocurrir. Es como si se adelantara a obtener los posibles errores.
- **Prefabricación:** se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
- **Programación en pares:** una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento. Es como el chofer y el copiloto: mientras uno conduce, el otro consulta el mapa.(SANCHEZ 2004)

1.7.3 Microsoft Solution Framework (MSF)

Esta es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos tecnológicos. MSF se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas.

MSF tiene las siguientes características:

- **Adaptable:** es parecida a un compás, usado en cualquier parte como un mapa, del cual su uso es limitado a un lugar específico.
- **Escalable:** puede organizar equipos tan pequeños entre 3 o 4 personas, así como también proyectos que requieren 50 personas a más.

- **Flexible:** es utilizada en el ambiente de desarrollo de cualquier cliente.
- **Tecnología Agnóstica:** porque puede ser usada para desarrollar soluciones basadas en cualquier tecnología.(SANCHEZ 2007)

Después de examinar las características de XP, MSF y RUP que son las metodologías de desarrollo de software más utilizadas en la actualidad se ha seleccionado a RUP que hace una perfecta combinación con UML que es el lenguaje de modelaje que se va a utilizar en este trabajo. El hecho que divida el proceso en fases, se asignen roles en el equipo de trabajo y se distribuyan las tareas va a permite lograr un mayor control del proyecto, así como aportar la organización y disciplina que se necesita para obtener resultados satisfactorios en el mismo. Además es muy ventajoso que el proceso sea iterativo e incremental pues posibilita que en cada una de las iteraciones se vayan corrigiendo errores lo que implica que al final se obtengan productos de calidad que satisfagan las expectativas y necesidades de los usuarios y/o clientes.

1.8 Herramientas de modelado visual

Existen herramientas que permiten representar un producto en su totalidad, especificar, analizar, diseñar el sistema antes de codificarlo; mediante diagramas que se van desarrollando durante el ciclo de vida del proyecto. Facilitan el desarrollo de un proceso cooperativo en el que todos los agentes tienen sus propias vistas de información (vistas de casos de uso, vista lógica, vista de componentes y vistas de despliegue), utiliza un lenguaje común para todo el equipo de desarrollo facilitando la comunicación.

1.8.1 Rational Rose

Es una herramienta CASE que proporciona mecanismos para realizar Ingeniería Directa e Inversa, es decir, a partir del código de un programa, se puede obtener información sobre su diseño. Brinda facilidades para elaborar la documentación del software que se está desarrollando y posee un gran número de estereotipos predefinidos que agilizan el proceso de modelación. Posibilita capacidades avanzadas de modelado visual para bases de datos. Además permite a los desarrolladores ver cómo accederá la aplicación a la base de datos, de forma que los problemas se pueden escalar antes del desarrollo.

Esta herramienta brinda la posibilidad de construir un modelo de casos de usos bien definido, identificar los objetos y representar cómo interactúan con los diagramas de secuencia y colaboración, organizar los componentes de software y su despliegue para diseminarlos por los distintos nodos de una arquitectura de red, describir la estructura de las clases, generar el código fuente de las clases definidas en el diseño. También se debe decir que una de sus limitaciones es que una vez generado el diagrama de clases persistentes a partir del cual se genera la base de datos del sistema, no existe la posibilidad de que el mismo exporte ese modelo hacia algún SGBD. (ANÓNIMO 2006)

1.8.2 Visual Paradimg

Es un producto que facilita las organizaciones visuales, diseña, integra y despliega su misión en las aplicaciones y sus bases de datos subyacentes y permite crear tipos diferentes de diagramas en un ambiente totalmente visual. Es una herramienta CASE que utiliza UML como lenguaje de modelado.

Esta herramienta es un producto de calidad, soporta aplicaciones Web, es muy fácil de instalar y actualizar, genera código para varios lenguajes, pero se debe señalar que una de sus desventajas es que las imágenes y reportes generados no son de muy buena calidad. El Visual Paradimg ofrece además:

- Un entorno de creación de diagramas para UML 2.0.
- Diseño centrado en casos de uso y enfocado al negocio generando un software de mayor calidad.
- Uso de un lenguaje común para todo el equipo de desarrollo facilitando la comunicación.
- Capacidades de ingeniería directa e inversa.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad en múltiples plataformas.(VIZCAÍNO 2007)

Considerando todas las facilidades brindadas por el Visual Paradimg, que es un producto de calidad, muy fácil de instalar y actualizar, que está disponible en varias ediciones entre las que se encuentran Enterprise, Professional, Community, Standard, Modeler y Personal ofreciéndoles a los usuarios la posibilidad de escoger según sus necesidades y que además es una herramienta de software libre, lo que constituye una poderosa ventaja no hay duda alguna que es la mejor opción para hacer el modelado visual del presente trabajo.

1.9 Tendencias Actuales

Es evidente que todas las empresas tienen como premisa triunfar en el mercado y es normal que quieran tener su propia Intranet Corporativa para realizar sus procesos que por supuesto están sustentados por grandes bases de datos operacionales que se utilizan para el trabajo diario de estas organizaciones pero ya no se conforman dar soporte a los procesos básicos (**sistemas de información para la gestión**) o (**OLTP, On-Line Transactional Processing**) que mantiene el trabajo transaccional diario de los sistemas de información originales, ahora son necesarias nuevas prestaciones de los sistemas de información (**sistemas de información para la toma de decisiones**) que les permitan manejar gran cantidad de información disponible en diferentes fuentes de datos heterogéneas como Internet, Librerías Digitales, Bases de datos antiguas, Sistemas de correo electrónico, etc.

Las empresas y sus directivos necesitan disponer de esta información de manera rápida y eficaz, libre de inconsistencias en el momento preciso para poder tomar buenas decisiones. Con la aparición constante de nuevas infraestructuras de comunicación con potentes y flexibles herramientas de tratamiento de información: Almacenes de Datos *(*Data Warehouse), herramientas OLAP (Online Analytical Processing) y Minería de Datos (Data Mining) mejoran la calidad, cantidad y eficiencia de los datos comerciales, así como el análisis, procesamiento y comunicación de los mismos.

En este capítulo se realizó un estudio con una visión analítica de las Intranets Corporativa a nivel internacional y nacional que presentan similitud con el sistema a desarrollar. Se describieron las características de herramientas, metodologías y estrategias para la elección de las que se usarán en el diseño de las bases de datos de Noticias y de las Facultades, así como conceptos asociados al dominio del problema. Se eligió el modelo relacional para el diseño de las bases de datos y el SGBD PostgreSQL. A demás se seleccionó como metodología de desarrollo de software el RUP, UML como lenguaje de modelado y para el modelado visual el Visual Paradimg.

Capítulo 2: Descripción y análisis de la solución propuesta

El diseño de una base de datos es un proceso complicado que se descompone en tres fases: diseño conceptual, diseño lógico, diseño físico. Desarrollar cada una de ellas implica realizar una serie de pasos. El primer paso para el diseño de una base de datos es la realización del esquema conceptual. En este caso se realizaron dos esquemas conceptuales, uno para la base de datos de Noticias y otros para la de Facultades. Cada uno de estos esquemas se compone de entidades, relaciones, atributos, dominios de atributos e identificadores. Partiendo del esquema conceptual se obtuvo el esquema lógico. También se hizo la descripción y fundamentación de la arquitectura, el análisis de optimización de consultas, la selección de los requisitos funcionales y no funcionales. Además se representan los diagramas de clases persistentes y la descripción de las clases.

2.1 Descripción de la arquitectura y fundamentación

La arquitectura seleccionada para la realización de la Intranet es la arquitectura orientada a servicios (SOA) para el desarrollo del sistema de información de la Intranet Corporativa de la UCI. Se va a estructurar en tres niveles en los que se dividirá la carga en tres partes bien definidas; una capa para presentación, una para negocio y otra para acceso a datos. En esta estructura una capa solo tiene relación con la siguiente.

La capa de acceso datos está compuesta por uno o varios gestores de bases de datos que realizan el almacenamiento de toda la información, recibe solicitudes de almacenamiento o recuperación de datos desde la capa de negocio. En este caso la capa de datos esta formada por un solo SGBD que es PostgreSQL.

PostgreSQL es un sistema de base de datos relacional libre, publicado bajo la licencia BSD. Una de sus características es que garantiza alta concurrencia mediante un sistema denominado MVCC (Acceso Concurrente Multiversión), es decir, permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin generar bloqueos. Además ofrece un conjunto de características técnicas que resultarían interesantes para cualquier diseñador de bases de datos.

- Integridad referencial
- Replicación (soluciones comerciales y no comerciales) que permiten la duplicación de bases de datos maestras en múltiples sitios de replica.
- Interfaces nativas para ODBC, JDBC, C, C++, PHP, Perl, TCL, ECPG, Python y Ruby
- Reglas
- Vistas
- Triggers
- Unicode
- Secuencias
- Herencia
- Outer Joins
- Sub-selects
- Una API abierta
- Procedimientos almacenados
- Soporte nativo SSL
- Lenguajes procedurales
- Respaldo en caliente
- Bloqueo a nivel mejor-que-fila
- Índices parciales y funcionales
- Soporte para consultas con UNION, UNION ALL y EXCEPT
- Herramientas para generar SQL portable para compartir con otros sistemas compatibles con SQL
- Sistema de tipos de datos extensible para proveer tipos de datos definidos por el usuario, y rápido desarrollo de nuevos tipos
- Funciones de compatibilidad para ayudar en la transición desde otros sistemas menos compatibles con SQL. (ANÓNIMO 2006)

El servidor de aplicaciones seleccionado fue Zope el mismo está escrito en el lenguaje de programación Python, en su mayor parte o casi totalmente puede ser manejado mediante una interfaz de usuario basada en páginas web. Usa una base de datos denominada ZODB o Zope Object Database que almacena objetos

ordenados en un sistema similar a un sistema de ficheros, aunque cuenta con múltiples conectores para las bases de datos relacionales habituales.

Como CMS se escogió Plone el cual es un CMS basado en Zope, programado en Python y publicado bajo la licencia GPL. Es un desarrollo basado en código abierto que puede utilizarse como servidor intranet o extranet. Es un Sistema de Publicación de documentos y una herramienta que permite el trabajo en grupo sin importar la situación geográfica.

El servidor web seleccionado fue Apache que es uno de los servidores web más potentes del mercado, hoy en día es muy utilizado encontrándose muy por encima de sus competidores, tanto gratuitos como comerciales. El servidor HTTP Apache es un software de código abierto que ofrece una perfecta combinación entre estabilidad y sencillez. Funciona en múltiples plataformas. Tiene capacidad para servir páginas tanto de contenido estático como de contenido dinámico a través de otras herramientas soportadas que facilitan la actualización de los contenidos mediante bases de datos, ficheros u otras fuentes de información. Apache entre sus características presenta mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido.

2.2 Análisis de optimización de Querys

En bases de datos relacionales el lenguaje de consultas SQL es lo más utilizado para obtener información desde la base de datos. La complejidad que pueden alcanzar algunas consultas puede ser tal, que el diseño de una consulta puede tomar un tiempo considerable, obteniendo no siempre una respuesta óptima.

Existen varias maneras de expresar y calcular la respuesta de una sentencia SQL, cada una de las formas sugiere una estrategia para encontrar la respuesta y por lo tanto algunas podrán optimizarse más que otras.

El problema que se plantea entonces es: encontrar la manera más apropiada de resolver la consulta, sin que, el proceso de determinar este resultado exceda un tiempo razonable y un gasto de recursos adicional.

El objetivo principal de este proceso es, encontrar o bien la mejor alternativa para solucionar la consulta, o de lo contrario, la alternativa que mejor cumpla las características de eficiencia, entre las estudiadas, cuando no sea posible estudiarlas todas.

Se entiende entonces como proceso de optimización de consultas al conjunto de técnicas, algoritmos y reglas que le permiten, al motor de base de datos, elegir una alternativa entre varias sentencias con la cual pueda rescatar los datos de la manera óptima.

Existen distintos métodos para optimizar consultas relacionales, sin embargo el enfoque de optimización basada en costos combinado con heurísticas que permitan reducir el espacio de búsqueda de la solución es el método mayormente utilizado por los motores de base de datos relacionales de la actualidad.

La optimización de consultas en base a costos supone la utilización de una medida de costo que sea común a lo largo del proceso, esta medida debe representar el criterio de minimización en la utilización de recursos del sistema, la medida estándar para bases de datos relacionales es usualmente la cantidad de entrada_salida (E/S) (tanto de disco como de la memoria intermedia). Este enfoque estima un costo que estará determinado por formulas predefinidas y por la información del catálogo inherente a la consulta. Sin embargo el optimizador no siempre escoge el mejor plan, ya que una búsqueda exhaustiva de la estrategia óptima puede consumir demasiado tiempo de proceso, entonces el optimizador escoge una estrategia razonablemente eficiente.(ANÓNIMO 2006)

2.2.1 Aspectos de optimización de las querys

Un aspecto importante en la parte de optimización de las querys relacionado con este trabajo son las relaciones de asociación entre las tablas de la base de datos de Noticias para almacenar todo lo referente a las noticias, avisos y efemérides.

Se normalizaron las tablas y se evitó la sobreutilización del join, debido a que el uso indiscriminado del mismo, alrededor de 10 vinculaciones puede traer demora de ejecución de la consulta a la hora de devolver los datos, e inconsistencia.

1-Se utilizó la función EXPLAIN ANALIZE para ver la cantidad de registros devueltos por la consulta, y el tiempo de ejecución de la misma y así ver de las tantas respuestas, cuál era la más óptima.

Ejemplo:

```
EXPLAIN ANALYZE select * from categoría
```

2- Se filtraron los datos con la utilización de sentencias, restricciones y funciones de agregación como WHERE, SUM, COUNT .etc. Se utilizaron funciones de agregaciones como GROUP BY, seleccionando los atributos necesarios y evitando la utilización del (*) mientras fuese posible.

Ejemplo:

Dado un editor seleccionar todos los avisos editados por este.

```
CREATE OR REPLACE FUNCTION recuperar_aviso_por_editor ("varchar")
  RETURNS SETOF record AS
$BODY$declare
datos record;
begin
for datos in select * from aviso where editor_noticia=$1
loop
return next datos;
end loop;
return;
end$BODY$
 LANGUAGE 'plpgsql' VOLATILE;
```

Una mejor forma para hacer la consulta de acuerdo a los datos que se quieren, es devolver solo los datos que se necesitan, así el tiempo de ejecución es más rápido al restringir más la búsqueda.

```
CREATE OR REPLACE FUNCTION recuperar_aviso_por_editor("varchar")
  RETURNS SETOF record AS
$BODY$declare
datos record;
begin
for datos in select  fecha publicada,
 titulo noticia,  articulo noticia from aviso where editor noticia=$1
loop
return next datos;
end loop;
return;
end$BODY$
  LANGUAGE 'plpgsql' VOLATILE;
```

3. La utilización del PERFORM en PostgreSQL a la hora de comprobar alguna sentencia, permite que una consulta sea ejecutada pero sin devolver ningún dato.

Ejemplo:

```
CREATE OR REPLACE FUNCTION obtener_aviso_por_categoria (int4)
  RETURNS SETOF record AS
$BODY$declare
muestra record;
begin
perform * from categoria where id_categoria=$1;
if not found then
raise exception ' no existe una categoria de ese tipo';
else
for muestra in select * from aviso where id_categoria =$1
loop
return next muestra;
end loop;
return;
end if;
end
```

4. Mediante ciclos utilizados en PostgreSQL, se logró una mayor consistencia de los datos ya que al ser una base de datos muy grande en registros almacenados, la base de datos puede colapsar a la hora de devolver los datos ya que muchos SGBD devuelven los registros al instante pero en PostgreSQL, con el comando NEXT va devolviendo cada registro uno por uno y evita que haya fallos a la hora de devolver los datos.

Una de las ventajas que se logra comprendiendo el proceso de optimización de consultas es la de entender que es lo que hace realmente un optimizador y con esto evitar la construcción de malas consultas.

Algunas consideraciones que ayudan al buen diseño de una base de datos son:

- Seguir los estándares de normalización de base de datos.
- Tratar de maximizar el uso de índices en clúster.
- Tratar de reducir el número de índices secundarios (no en clúster).
- Si existen consultas que involucren más de 4 tablas en un Join se recomienda desnormalizar algunas tablas.
- Para tablas potencialmente grandes se recomienda el particionamiento horizontal de tablas (varias tablas con la misma estructura original pero con datos distribuidos por cierto criterio predefinido).(ANÓNIMO 2006)

2.3 Requisitos funcionales y no funcionales

Requisitos Funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. (CONFERENCIA#3 2006)

R1. Autenticar usuario.

- El sistema debe ser capaz de autenticar los usuarios por el dominio.
- El sistema debe ser capaz de identificar los grupos de usuarios en dependencia al rol que desempeñan.

Esta información requiere:

a-) Nombre de usuario.

b-) Contraseña.

R2. Adicionar noticias.

- El sistema debe ser capaz de adicionar las noticias por categorías, confeccionadas por los periodistas o encontradas en los diferentes medios de comunicación.

Esta información requiere:

a-) Categoría.

b-) Título.

c-) Resumen.

d-) Texto de la noticia.

e-) Fotos.

f-) Fuentes.

g-) Archivos.

h-) Fecha de publicación.

i-) Fecha de retirada.

j-) Autor.

R3. Actualizar noticias.

- El sistema debe ser capaz de actualizar las noticias diariamente en dependencia de las diferentes categorías.

  Esta información requiere:

  a-) Categoría.
  b-) Título.
  c-) Resumen.
  d-) Texto de la noticia.
  e-) Fotos.
  f-) Fuentes.
  g-) Archivos.
  h-) Fecha de publicación.
  i-) Fecha de retirada.
  j-) Autor.

R4. Aprobar noticias.

- El sistema debe ser capaz de decidir si la noticia cumple con los requisitos para ser publicada.

  Esta información requiere:

  a-) Categoría.
  b-) Autor o fuente.
  c-) Tema.
  d-) Palabras en el título.
  e-) Fecha de publicación.
  f-) Fecha de caducidad.
  g-) Palabras claves.

R6. Adicionar Avisos.

- El sistema debe ser capaz de adicionar avisos creados por los responsables de aéreas que pueden ser de interés para ciertos usuarios o para todos en general.
- El sistema debe ser capaz de identificar los avisos según sean ser docentes, eventuales o de otras categorías.
- Pueden estar enmarcados de importancia o de prioridad.

  Esta información requiere:

  a-) Autor.
  b-) Nivel de prioridad.
  c-) Categorías.
  d-) Fechas de publicación.
  e-) Categoría.
  f-) Título.
  g-) Texto.
  h-) Fecha de retirada.
  i-) Solicitud de área.

R7. Actualizar avisos.

- El sistema debe ser capaz de actualizar los avisos según vayan caducando en la fecha de vigencia.
- Debe ser capaz de identificar los avisos por categorías y nivel de prioridad.

  Esta información requiere:

  a-) Contenido nuevo.
  b-) Categoría.
  c-) Nivel de prioridad.
  f-) Título.
  g-) Texto.
  h-) Fecha de publicación.
  i-) Fecha de retirada.
  j-) Autor.
  k-) Solicitud de área.

R8. Aprobar avisos.

- El sistema debe ser capaz de aprobar los avisos que serán publicados a partir de que cumplan con los requisitos para la publicación.
  Esta información requiere:
    a-) Autor.
    b-) Título.
    c-) Nivel de prioridad.
    d-) Categorías.
    e-) Fechas de publicación.
    f-) Fecha de retirada.
    g-) Solicitud del área.

R10. Adicionar efemérides.

- El sistema debe ser capaz de adicionar efemérides de acuerdo al día en curso.
  Esta información requiere:
    a-) Texto.
    b-) Fecha.
    c-) Título.
    e-) Aprobación.
    f-) Fecha de publicación.
    g-) Archivos.

R11. Actualizar efemérides.

- El sistema debe ser capaz de actualizar diariamente las efemérides en correspondencia a la fecha.
  Esta información requiere:
    a-) Texto.
    b-) Fecha.

c-) Título.

e-) Aprobación.

f-) Fecha de publicación.

g-) Archivos.

R12. Aprobar efemérides.

- El sistema debe ser capaz de aprobar las efemérides que serán publicadas diariamente.
  Esta información requiere:
  a-) Texto.
  b-) Fecha.
  c-) Título.
  e-) Aprobación.
  f-) Fecha de publicación.
  g-) Archivos.

R14. Enviar solicitud de publicación SW.

- El sistema debe ser capaz enviar las solicitudes confeccionadas por los diferentes Resp. de aéreas.
  Esta información requiere:
  a-) Descripción del sitio.
  b-) Nombre del solicitante.
  c-) Clasificación del sitio.

R15. Aprobar solicitud de publicación SW.

- El sistema debe ser capaz de aprobar los sitios web que se publicarán.
  Esta información requiere:
  a-) Descripción del sitio.
  b-) Clasificación.

R16. Publicar Sitios Web.

- El sistema debe ser capaz de publicar los sitios web de las diferentes aéreas de la universidad de acuerdo con la aprobación del consejo editor.
  Esta información requiere:
    a-) Descripción del sitio.
    b-) Clasificación.

R17. Enviar solicitud de publicación de páginas Informativas.

- El sistema debe ser capaz de recibir las solicitudes enviadas por los diferentes responsables de aéreas para publicar las páginas informativas, estas recogen la información de determinado usuario, área o servicio perteneciente a la universidad.

  Esta información requiere:
    a-) Título.
    b-) Fuente (Área de la universidad).
    c-) Resumen del contenido de la página (Texto).

R18. Aprobar solicitud de publicación de páginas Informativas.

- El sistema debe ser capaz de enviar la aprobación de la solicitud de publicación de páginas Informativas.
  Esta información requiere:
    a-) Título.
    b-) Fuente (Área de la universidad).
    c-) Resumen del contenido de la página (Texto).
    d-) Respuesta.

R19. Publicar Páginas informativas.

- El sistema debe ser capaz de publicar las páginas informativas que han sido aprobadas.
  Esta información requiere:
    a-) Título.
    b-) Fuente (Área de la universidad).

c-) Página a publicar.

R20. Adicionar usuario.

- El sistema debe ser capaz de adicionar usuario en dependencia del rol a desempeñar.
  Esta información requiere:
  a-) Nombre del usuario.
  b-) Fecha de creación.
  c-) Área a la pertenece.
  d-) Rol.

R21. Modificar usuario.

- El sistema debe ser capaz de modificar los usuarios, en dependencia del rol que desempeñe en el momento.

  Esta información requiere:
  a-) Nombre del usuario.
  b-) Fecha de creación.
  c-) Área a la pertenece.
  d-) Rol.

R22. Eliminar usuario

- El sistema debe ser capaz de eliminar usuario y rol correspondiente.
  Esta información requiere:
  a-) Nombre del usuario.
  b-) Fecha de creación.
  c-) Área a la pertenece.
  d-) Rol.

R23. Buscar información del contenido del portal.

- El sistema debe ser capaz de realizar una búsqueda y recopilación de cualquier información perteneciente a la universidad existente en el portal.
  Esta información requiere:
  a-) Información que se va a buscar.
  b-) Palabras claves.
  c-) Autor.

R24. Mostrar estadísticas del portal.

- El sistema debe ser capaz de mostrar las estadísticas del portal, usuario que han visitados, sitios más visitados, información más buscada y servicios más utilizado.
  Esta información requiere:
  a-) Usuarios.
  b-) Sitios visitados.
  c-) Servicios.

Requerimientos no funcionales del sistema

Los requerimientos no funcionales son propiedades o cualidades que el producto puede tener. Debe pensarse en esas propiedades como las características que hacen al producto activo, usable, rápido o confiable.(CONFERENCIA#3 2006)

Entre las diferentes categorías de los requerimientos no funcionales se encuentran:

1. Usabilidad.

- El portal podrá ser accedido por todos los usuarios de la UCI.
- Debe tener un diseño sencillo.
- La navegación debe ser fácil para el usuario.
- No se debe recargar de contenido que resulte molesto.

2. Portabilidad.

- El sistema será multiplataforma.

3. Seguridad.

- El sistema se regirá por el reglamento disciplinario de navegación de la universidad, así como la seguridad que ofrece PLONE.
- Debe estar protegida con la seguridad de Linux. Los usuarios que accedan a la aplicación solo tienen acceso según el rol que cumplan dentro de esta.

4. Implementación.

- Utilizar el servidor Apache para remplazar el Zope Web Server. Para manejar las BD utilizar PostgreSQL en reemplazo de ZopeDB. Usar CMS Plone y el lenguaje que trae incorporado Python para el desarrollo de la aplicación.

5. Apariencia.

- La apariencia del sistema debe ser lo más accesible y usable posible, cumplir las normas de la UCI, en dependencia del rol que cumpla el usuario.

6. Software.

Cliente:

- Cualquier sistema operativo con interfaz gráfica y red.
- Navegador Web.

Servidor:

- Linux cualquier distribución (Debian, Ubuntu, KDE, etc.).
- Servidor de BD PostgreSQL en reemplazo de ZopeDB.
- Apache para remplazar el Zope Web Server.

7. Hardware.

Para el desarrollo:

- Pentium IV o superior.
- 256 de memoria RAM o superior.
- Disco duro con capacidad de 80 GB o superior.

Para la explotación:

Clientes:

- Pentium IV o superior.
- 240 de RAM o superior.

8. Legales.

- El sistema contara con normas de de estilos y pautas de diseño para cada uno de los servicios de la intranet aprobadas por el consejo de dirección de la Universidad.

9. Confiabilidad.

- El sistema solo será visitado por los usuarios que cuenten con acceso al dominio UCI.

10. Interfaz.

- La interfaz del sistema será muy legible.
- También será fácil de usar.
- Debe regirse por el manual de estilo de la Universidad.

11. Ayuda y documentación en línea.

- El sistema cuenta con ayuda para navegar en el portal, una guía de estilos y pautas de diseño.

2.4 Diagrama de clases persistentes

Las clases persistentes son aquellas cuyos objetos deben ser almacenados en algún repositorio como una base de datos relacional. Estos objetos deben contener sólo los datos de las entidades que se modelan y sus relaciones con otros objetos persistentes.

La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo. Lo contrario son las clases temporales (transient) que son manejadas y almacenadas por el sistema en tiempo de ejecución por lo que dejan de existir cuando termina el programa.

Es responsabilidad del diseñador definir cuáles clases son las que deben ser persistentes.

El diagrama de clases persistentes toma como base la notación del diagrama de estructura de UML, aunque se añaden nuevas características usando estereotipos y notas. Se representan las clases preliminares, las relaciones entre las clases persistentes, y los atributos de las clases. En la siguiente figura se muestra el diagrama de clases persistentes para las bases de datos Noticia y Facultad.

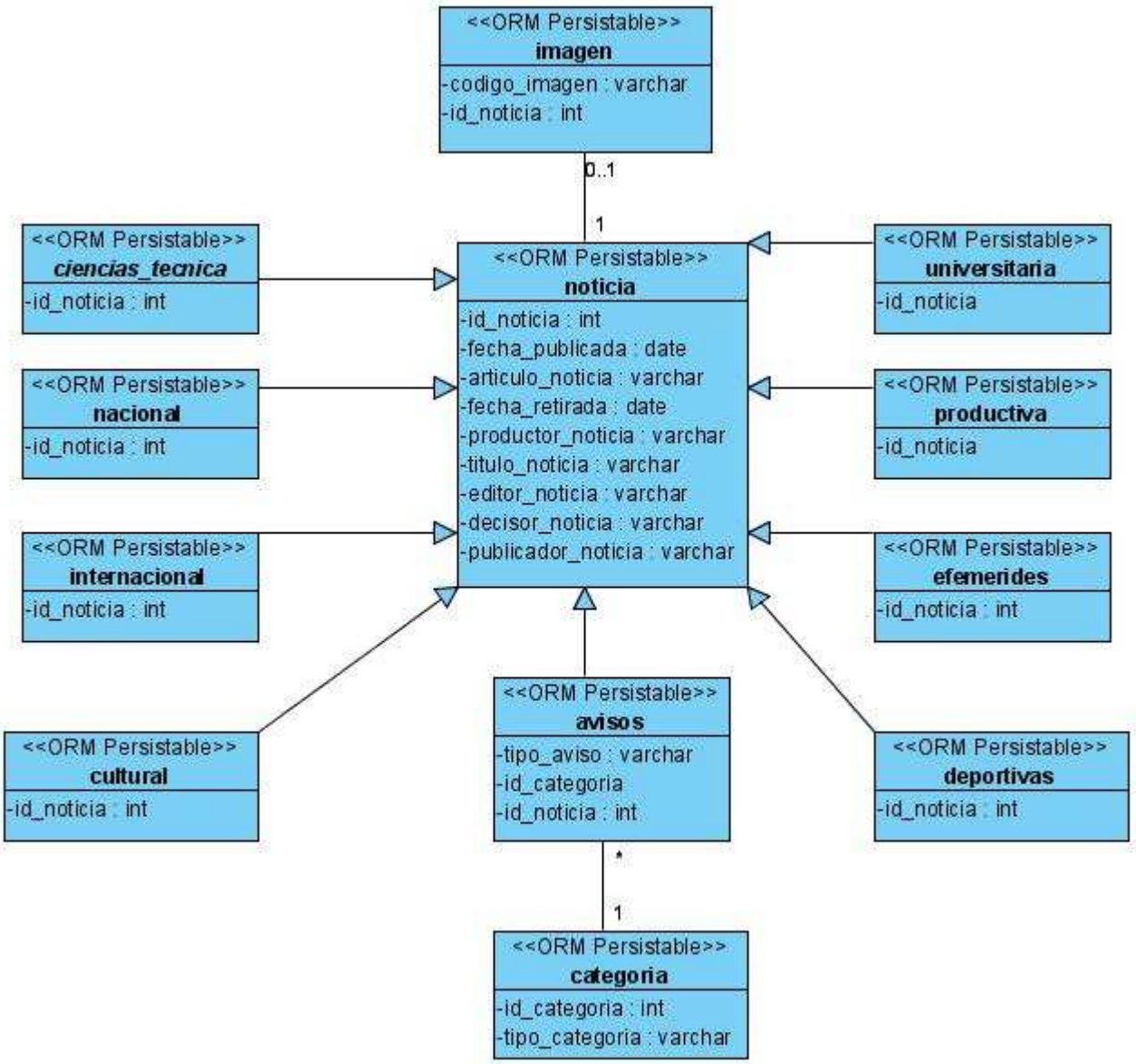

Figura 1. Diagrama de clases persistentes BD Noticia

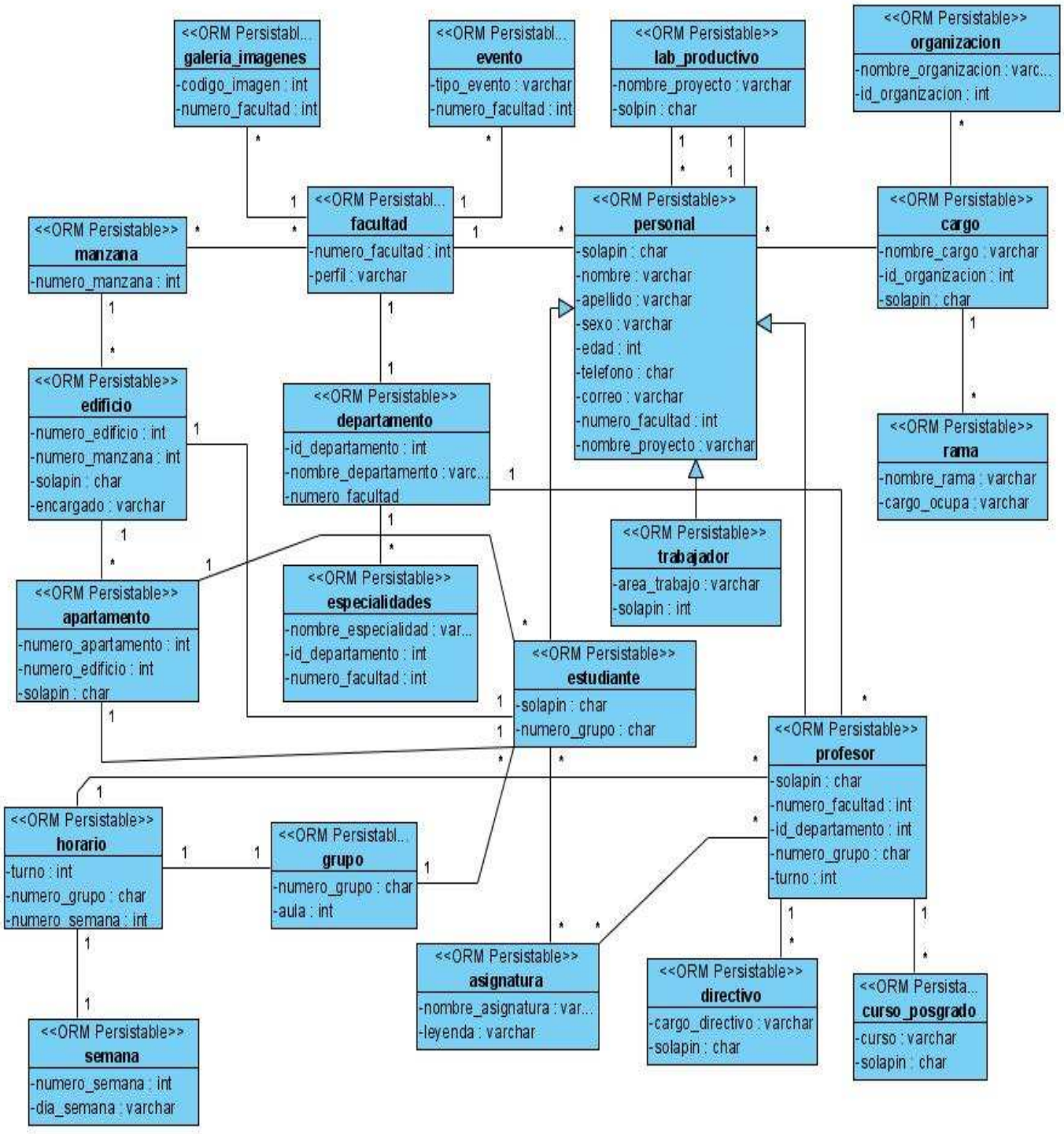

Figura 2. Diagrama de clases persistentes BD Facultad

2.4.1 Descripción de las clases

Tabla 2. Noticia

Nombre: Noticia	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
titulo_noticia	varchar
articulo_noticia	varchar
fecha_retirada	date
productor_noticia	varchar
editor_noticia	varchar
decisor_noticia	varchar
publicador_noticia	varchar
Para cada responsabilidad:	
Nombre:	Insertar_noticia (fecha_publicada, título_noticia,articulo_noticia,fecha_retirada,productor_noticia,editor_noticia, decisor_noticia,publicador_noticia)
Descripción:	Inserta una noticia con los datos pasados como parámetro.
Nombre	Actualizar_ noticia(título_noticia)
Descripción	Modifica el titulo de la noticia con el dato pasado por el parámetro.
Nombre	Eliminar_noticia(id_noticia)
Descripción	Elimina una noticia con los datos pasados como parámetro
Nombre	Recuperar_noticia_por_titulo(título_noticia)
Descripción	Muestra los datos de la noticia pasada como parámetro.

Tabla 3. Efemérides

Nombre: efemérides	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
título_noticia	varchar
articulo_noticia	varchar

fecha_retirada	date
productor_noticia	varchar
editor_noticia	varchar
decisor_noticia	varchar
publicador_noticia	varchar
Para cada responsabilidad:	
Nombre:	Buscar_efeméride(fecha_publicada)
Descripción:	Busca los datos de la efeméride en la fecha que se pasa como parámetro.

Tabla 4. Ciencias técnicas

Nombre: ciencias_tecnicas	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
titulo_noticia	varchar
articulo_noticia	varchar
fecha_retirada	date
productor_noticia	varchar
editor_noticia	varchar
decisor_noticia	varchar
publicador_noticia	varchar
Para cada responsabilidad:	
Nombre:	Buscar_efemeride(titulo_noticia)
Descripción:	Busca las noticias de tipo efemérides con los datos pasados como parámetro.
Nombre	Mostrar_efemeride()
Descripción	Muestra los datos de las efemérides

Tabla 5. Cultural

Nombre: cultural	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
titulo_noticia	varchar
articulo_noticia	varchar
fecha_retirada	date
productor_noticia	varchar

editor_noticia	varchar
decisor_noticia	varchar
publicador_noticia	varchar
Para cada responsabilidad:	
Nombre:	buscar_noticia_cutural(titulo_noticia)
Descripción:	Busca la información de las noticias culturales con los datos pasados como parámetro.
Nombre	mostrar_noticia_cultural()
Descripcion	Muestra la información de este tipo de noticias

Tabla 6. Deportiva

Nombre: deportiva	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
titulo_noticia	varchar
articulo_noticia	varchar
fecha_retirada	date
productor_noticia	varchar
editor_noticia	varchar
decisor_noticia	varchar
publicador_noticia	varchar
Para cada responsabilidad:	
Nombre:	buscar_noticia_deportiva(titulo_noticia)
Descripción:	Muestra los datos de la noticia deportiva con los datos pasados como parámetro.

Tabla 7. Internacional

Nombre: internacional	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
titulo_noticia	varchar
articulo_noticia	varchar
fecha_retirada	date
productor_noticia	varchar
editor_noticia	varchar

decisor_noticia	varchar
publicador_noticia	varchar
Para cada responsabilidad:	
Nombre:	insertar_noticia_internacional(fecha_publicada, titulo_noticia,articulo_noricia,fecha_retirada_productor_noticia,editor_noticia, decisor_noticia,publicador_noticia)
Descripción:	Inserta noticias internacionales con los datos pasados como parámetro.

Tabla 8. Nacional

Nombre: nacional	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
titulo_noticia	varchar
articulo_noticia	varchar
fecha_retirada	date
productor_noticia	varchar
editor_noticia	varchar
decisor_noticia	varchar
publicador_noticia	varchar
Para cada responsabilidad:	
Nombre:	insertar_noticia_nacional(fecha_publicada, titulo_noticia,articulo_noricia,fecha_retirada_productor_noticia,editor_noticia, decisor_noticia,publicador_noticia)
Descripción:	Inserta noticias nacionales con los datos pasados como parámetro.

Tabla 9. Productiva

Nombre: productiva	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
titulo_noticia	varchar

articulo_noticia	varchar
fecha_retirada	date
productor_noticia	varchar
editor_noticia	varchar
decisor_noticia	varchar
publicador_noticia	varchar
Para cada responsabilidad:	
Nombre:	Insertar_noticia_productiva(fecha_publicada, titulo_noticia,articulo_noricia,fecha_retirada_productor_noticia,editor_noticia, decisor_noticia,publicador_noticia)
Descripción:	Inserta noticia productivas con los datos pasados como parámetro

Tabla 10. Universitaria

Nombre: universitaria	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
titulo_noticia	varchar
articulo_noticia	varchar
fecha_retirada	date
productor_noticia	varchar
editor_noticia	varchar
decisor_noticia	varchar
publicador_noticia	varchar
Para cada responsabilidad:	
Nombre:	Insertar_noticia_universitaria(fecha_publicada, titulo_noticia,articulo_noricia,fecha_retirada_productor_noticia,editor_noticia, decisor_noticia,publicador_noticia)
Descripción:	Inserta noticia universitaria con los datos pasados como parámetro

Tabla 11. Avisos

Nombre: avisos
Tipo de clase (entidad)

Atributo	**Tipo**
id_noticia (llave)	int
fecha_publicada	date
titulo_noticia	varchar
articulo_noticia	varchar
fecha_retirada	date
productor_noticia	varchar
editor_noticia	varchar
decisor_noticia	varchar
publicador_noticia	varchar
tipo_aviso	varchar
id_categoria	int
Para cada responsabilidad:	
Nombre:	Recuperar_aviso_por_editor(editor_noticia)
Descripción:	Muestra los avisos en correspondencia con los datos pasados como parámetro.
Nombre	obtener_aviso_por_tipo(tipo_aviso)
Descripción	Selecciona todos los avisos de acuerdo al tipo pasado como parámetro.

Tabla 12. Categoría

Nombre: categoría	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_categoria (llave)	int
tipo_categoria	varchar
Para cada responsabilidad:	
Nombre:	eliminar_categoria(id_categoria)
Descripción:	Elimina una categoría con el dato pasado como parámetro

Tabla 13. Imagen

Nombre: imagen	
Tipo de clase (entidad)	
Atributo	**Tipo**
código_imagen	varchar
id_noticia	int
Para cada responsabilidad:	
Nombre:	insertar_image_noticia(código_imagen,id_noticia)
Descripción:	Inserta una imagen de la noticia con los datos pasados como parámetro

Tabla 14. Facultad

Nombre: facultad	
Tipo de clase (entidad)	
Atributo	**Tipo**
numero_facultad (llave)	int
perfil	varchar
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 15. Lab productivo

Nombre: lab_productivo	
Tipo de clase (entidad)	
Atributo	**Tipo**
nombre_proyecto (llave)	varchar
solapin	char
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 16. Manzana

Nombre: manzana	
Tipo de clase (entidad)	
Atributo	**Tipo**
numero_manzana (llave)	int
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 17. Edificio

Nombre: edificio	
Tipo de clase (entidad)	
Atributo	**Tipo**
numero_edificio (llave)	int
numero_manzana	int
solapin	char
encargado	varchar

Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 18. Apartamento

Nombre: apartamento	
Tipo de clase (entidad)	
Atributo	**Tipo**
numero_apartamento (llave)	int
numero_edificio	int
solapin	char
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 19. Galería_imágenes

Nombre: galería_imagenes	
Tipo de clase (entidad)	
Atributo	**Tipo**
código_imagenes (llave)	int
numero_facultad	int
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 20. Horario

Nombre: horario	
Tipo de clase (entidad)	
Atributo	**Tipo**
turno	int
numero_grupo	char
numero_semana	int
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 21. Semana

Nombre: semana	
Tipo de clase (entidad)	
Atributo	**Tipo**
numero_semana	int
dia_semana	varchar
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 22. Evento

Nombre: evento	
Tipo de clase (entidad)	
Atributo	**Tipo**
tipo_evento (llave)	varchar
numero_facultad	int
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 23. Departamento

Nombre: departamento	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_departamento	int
nombre_departamento	varchar
numero_facultad	int
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 24. Especialidades

Nombre: especialidades	
Tipo de clase (entidad)	
Atributo	**Tipo**
nombre_especialidad (llave)	varchar
id_departamento	int

numero_facultad	int
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 25. Directivos

Nombre: directivos	
Tipo de clase (entidad)	
Atributo	**Tipo**
solapin	char
cargo_directivo	varchar
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 26. Personal

Nombre: personal	
Tipo de clase (entidad)	
Atributo	**Tipo**
solapin (llave)	char
nombre	varchar
apellido	varchar
sexo	varchar
edad	int
teléfono	char
correo	varchar
numero_facultad	int
nombre_proyecto	varchar
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 27. Estudiante

Nombre: estudiante	
Tipo de clase (entidad)	
Atributo	**Tipo**
solapin (llave)	char

nombre	varchar
apellido	varchar
sexo	varchar
edad	int
teléfono	char
correo	varchar
numero_grupo	int
numero_apartamento	int
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 28. Trabajador

Nombre: trabajador	
Tipo de clase (entidad)	
Atributo	**Tipo**
solapin (llave)	char
nombre	varchar
apellido	varchar
sexo	varchar
edad	int
teléfono	char
correo	varchar
area_trabajo	varchar
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 29. Grupo

Nombre: grupo	
Tipo de clase (entidad)	
Atributo	**Tipo**
numero_grupo	int
aula	int
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 30. Asignatura

Nombre: asignatura	
Tipo de clase (entidad)	
Atributo	**Tipo**
nombre_asignatura	varchar
leyenda	varchar
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 31. Curso_postgrado

Nombre: curso_posgrado	
Tipo de clase (entidad)	
Atributo	**Tipo**
solapin	char
curso	varchar
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 32. Profesor

Nombre: profesor	
Tipo de clase (entidad)	
Atributo	**Tipo**
solapin (llave)	char
nombre	varchar
apellido	varchar
sexo	varchar
edad	int
teléfono	char
correo	varchar
numero_facultad	int
id_departamento	int
numero_grupo	int
turno	int
Para cada responsabilidad:	
Nombre:	(4)

Descripción:	(5)

Tabla 33. Rama

Nombre: rama	
Tipo de clase (entidad)	
Atributo	**Tipo**
nombre_rama	varchar
nombre_cargo	varchar
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 34. Organización

Nombre: organizacion	
Tipo de clase (entidad)	
Atributo	**Tipo**
id_organizacion (llave)	int
nombre_organizacion	varchar
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

Tabla 35. Cargo

Nombre: cargo	
Tipo de clase (entidad)	
Atributo	**Tipo**
nombre_cargo (llave)	varchar
solapin	char
id_organizacion	int
Para cada responsabilidad:	
Nombre:	(4)
Descripción:	(5)

2.5 Diagrama Entidad Relación de la BD

Diseñadores de sistemas, analistas y programadores desde hace mucho tiempo han buscado un sistema que les permita visualizar una base de datos, sus tablas, el contenido de éstas, las claves y las interrelaciones con otras tablas. Los diagramas Entidad-Relación (E-R) permiten efectuar esta representación de forma sencilla, mediante entidades con sus respectivos atributos e interrelaciones. Esto resulta bastante interesante porque permite tratar mediante digramas E-R problemas concretos de la realidad. Aunque es cierto que a la hora de construir modelos sobre realidades concretas es cuando surgen los problemas. Las entidades tienen muchos atributos diferentes, de los cuales se debe aprender a elegir solo los necesarios. Puede decirse lo mismo de las interrelaciones. Además, no siempre está perfectamente claro qué es un atributo y qué una entidad; o que ventajas se obtendrán si se tratan a ciertos atributos como entidades y viceversa.

El diagrama E-R de la base de datos de Noticias es bastante sencillo, el mismo consta de doce entidades interrelacionadas, cada una tiene su respectivo conjunto de atributos y estos a su vez tienen sus dominios. La base de datos de Facultad es un poco más compleja está formada por veinticinco entidades igualmente interrelacionadas que poseen atributos con su respectivo dominio.

Los pasos seguidos para realizar el diagrama E-R fueron los siguientes:

- Reunirse con los analistas del sistema y dejar claros los parámetros y objetivos del proceso a modelar.
- Identificar los conjuntos de entidades útiles para modelar el proceso.
- Identificar los conjuntos de interrelaciones y determinar su grado y tipo (1:1, 1: n o m: n).
- Trazar un primer diagrama E-R.
- Identificar atributos y dominios para los conjuntos de entidades e interrelaciones.
- Seleccionar las claves principales para los conjuntos de entidades.
- Verificar que el modelo resultante cumple el planteamiento las peticiones de los clientes que en este caso es lo representado por los analistas.
- Una vez obtenida la versión correcta del modelo traducir el diagrama a un modelo lógico.

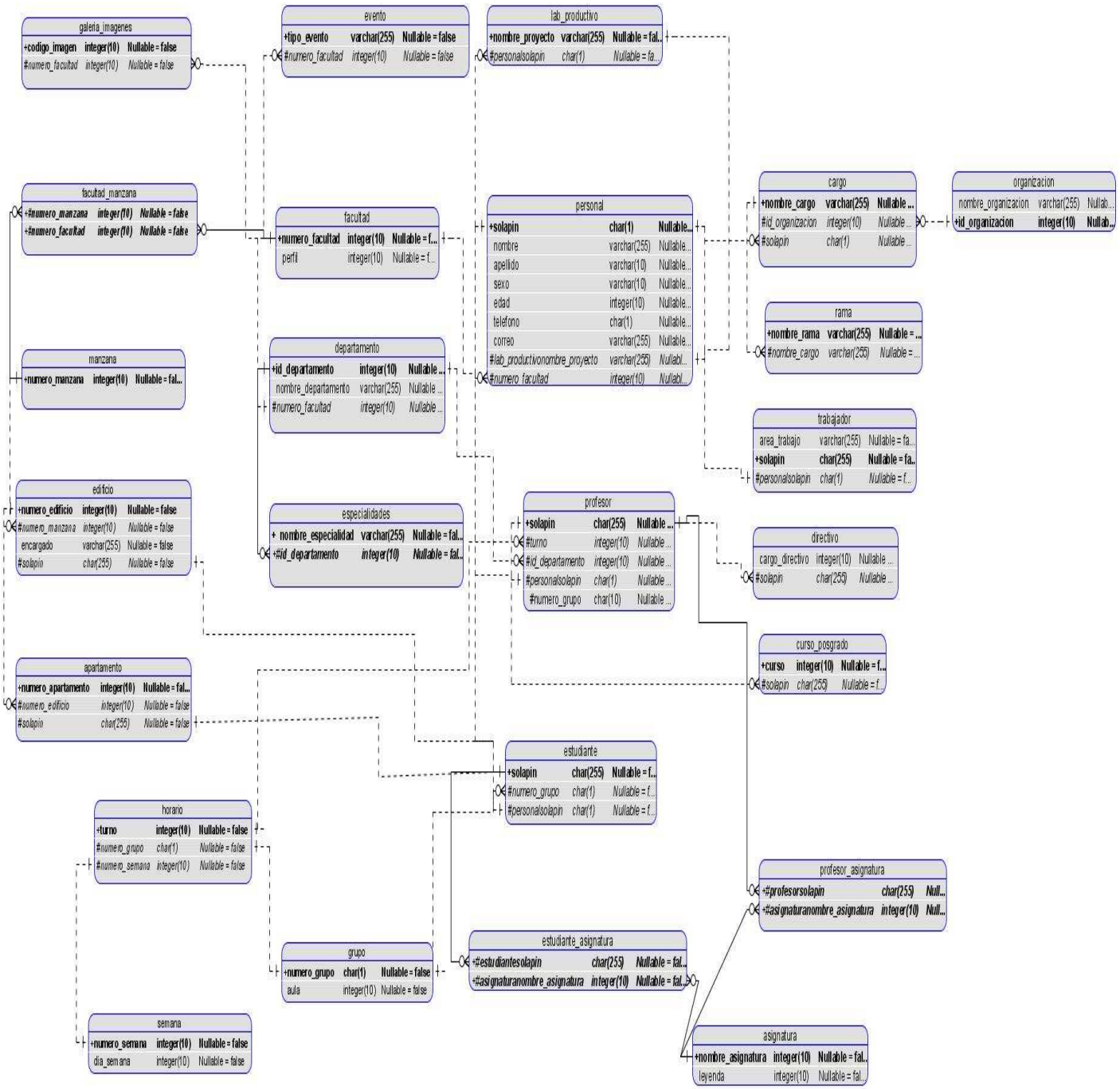

Figura 3. Diagrama de Entidad_Relación BD Facultad

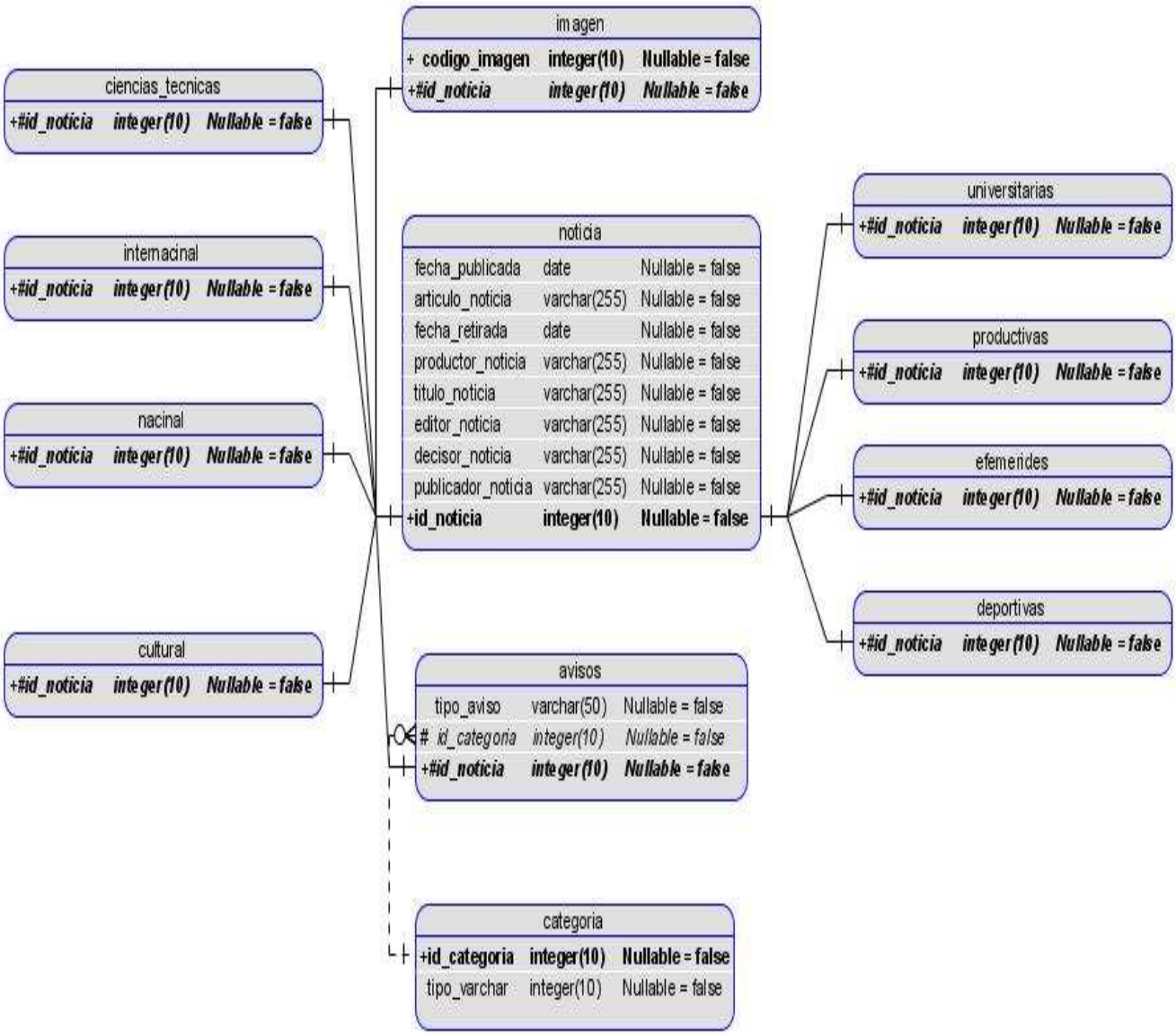

Figura 4. Diagrama de Entidad_Relación BD Noticia

2.5.1 Descripción de las tablas

Tabla 36. Noticia

Nombre: noticia		
Atributo	**Tipo**	**Descripción**
id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Título de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia
productor_noticia	varchar	El nombre del periodista que se encargara de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargara de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la noticia en el sitio

Tabla 37. Efemérides

Nombre: efemérides		
Atributo	**Tipo**	**Descripción**
id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Título de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia
productor_noticia	varchar	El nombre del periodista que se encargará de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargará de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la noticia en el sitio

Tabla 38. Ciencias_técnica

Nombre: ciencias_tecnica		
Atributo	**Tipo**	**Descripción**
id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Titulo de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia
productor_noticia	varchar	El nombre del periodista que se encargará de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargará de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la noticia en el sitio

Tabla 39. Cultural

Nombre: cultural		
Atributo	**Tipo**	**Descripción**
id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Título de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia
productor_noticia	varchar	El nombre del periodista que se encargará de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargará de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la noticia en el sitio

Tabla 40. Deportiva

Nombre: deportiva		
Atributo	**Tipo**	**Descripción**
id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Título de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia
productor_noticia	varchar	El nombre del periodista que se encargará de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargará de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la noticia en el sitio

Tabla 41. Internacional

Nombre: internacional		
Atributo	**Tipo**	**Descripción**
id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Título de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia
productor_noticia	varchar	El nombre del periodista que se encargará de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargará de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la noticia en el sitio

Tabla 42. Nacional

Nombre: nacional		
Atributo	**Tipo**	**Descripción**

id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Título de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia
productor_noticia	varchar	El nombre del periodista que se encargará de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargará de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la noticia en el sitio

Tabla 43. Productivas

Nombre: productivas		
Atributo	**Tipo**	**Descripción**
id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Título de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia
productor_noticia	varchar	El nombre del periodista que se encargará de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargará de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la noticia en el sitio

Tabla 44. Universitarias

Nombre: universitarias		
Atributo	**Tipo**	**Descripción**
id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Título de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia

productor_noticia	varchar	El nombre del periodista que se encargará de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargará de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la noticia en el sitio

Tabla 45. Categoría

Nombre: categoría		
Atributo	**Tipo**	**Descripción**
id_categoria	int	Identificador de la categoría
tipo_categoria	varchar	Define la categoría que corresponde al aviso

Tabla 46. Imagen

Nombre: imagen		
Atributo	**Tipo**	**Descripción**
código_imagen	varchar	Identificador de la imagen
id_noticia	int	Define la noticia con la cual se corresponde la imagen

Tabla 47. Avisos

Nombre: avisos		
Atributo	**Tipo**	**Descripción**
id_noticia	int	Identificador de la noticia
fecha_publicada	date	La fecha en la que fue publicada la noticia
titulo_noticia	varchar	Título de la noticia
articulo_noticia	varchar	El contexto de la noticia
fecha_retirada	date	La fecha en la que será retirada la noticia
productor_noticia	varchar	El nombre del periodista que se encargará de elaborar la noticia
editor_noticia	varchar	El nombre del compañero que se encargará de editar la noticia
decisor_noticia	varchar	El nombre del compañero que decidirá si la noticia será publicada o no
publicador_noticia	varchar	El nombre del compañero que publicará la

		noticia en el sitio
tipo_aviso	varchar	Identificador del aviso
id_categoria	int	Define la categoría con la cual se corresponde el aviso

Tabla 48. Facultad

Nombre: facultad		
Atributo	**Tipo**	**Descripción**
numero_facultad	int	Identificador de la facultad
perfil	varchar	El nombre del perfil que corresponde a la facultad

Tabla 49. Lab productivo

Nombre: lab_productivo		
Atributo	**Tipo**	**Descripción**
nombre_proyecto	varchar	Identificador del proyecto productivo
solapin	char	Identificador del personal

Tabla 50. Manzana

Nombre: manzana		
Atributo	**Tipo**	**Descripción**
numero_manzana	int	Identificador de la manzana donde se encuentran los edificios de una o mas facultades

Tabla 51. Manzana_facultad

Nombre: manzana_facultad		
Atributo	**Tipo**	**Descripción**
numero_manzana	int	Identificador de la manzana
numero_facultad	int	Identificador de la facultad

Tabla 52. Edificio

Nombre: edificio		
Atributo	**Tipo**	**Descripción**
numero_edificio	int	Identificador del edificio
numero_manzana	int	Identificador de la manzana
solapin	char	Identificador del estudiante jefe de edificio
encargado	varchar	Persona encargada de cuidar el edificio.

Tabla 53. Apartamento

Nombre: apartamento		
Atributo	**Tipo**	**Descripción**
numero_edificio	int	Identificador del edificio
numero_manzana	int	Identificador de la manzana
solapin	char	Identificador del jefe de apartamento

Tabla 54. Galería_imágenes

Nombre: galería_imagenes		
Atributo	**Tipo**	**Descripción**
código_imagenes	int	Identificador de la imagen
numero_facultad	int	Identificador de la facultad

Tabla 55. Horario

Nombre: horario		
Atributo	**Tipo**	**Descripción**
turno	int	Define la hora en comienza y termina una asignatura
numero_grupo	char	Identificador del grupo
numero_semana	int	Identificador de la semana

Tabla 56. Semana

Nombre: semana		
Atributo	**Tipo**	**Descripción**
numero_semana	int	Identificador de la semana
dia_semana	varchar	Da la posibilidad de conocer que día de la semana se impartió una asignatura

Tabla 57. Evento

Nombre: evento		
Atributo	**Tipo**	**Descripción**
tipo_evento	varchar	Identificador del evento
numero_facultad	int	Identificador de la facultad

Tabla 58. Especialidades

Nombre: especialidades		
Atributo	**Tipo**	**Descripción**
nombre_especialidad	varchar	Identificador de la especialidad
id_departamento	int	Identificador del departamento
numero_facultad	int	Identificador de la facultad

Tabla 59. Directivos

Nombre: directivos		
Atributo	**Tipo**	**Descripción**
solapin	char	Identificador de la persona que es directivo
cargo_directivo	varchar	Identifica el cargo del personal del departamento

Tabla 60. Personal

Nombre: personal		
Atributo	**Tipo**	**Descripción**
solapin	char	Identificador del personal
nombre	varchar	El nombre del personal
apellido	varchar	Los apellidos del personal
sexo	varchar	El sexo del personal
edad	int	La edad del personal
teléfono	char	El teléfono del personal

correo	varchar	El correo del personal
numero_facultad	int	Identificador de la facultad
nombre_proyecto	varchar	Identificador del proyecto

Tabla 61. Estudiante

Nombre: estudiante		
Atributo	**Tipo**	**Descripción**
solapin	char	Identificador del personal que es estudiante
nombre	varchar	El nombre del estudiante
apellido	varchar	Los apellidos del estudiante
sexo	varchar	El sexo del estudiante
edad	int	La edad del estudiante
teléfono	char	El teléfono del estudiante
correo	varchar	El correo del estudiante
numero_grupo	char	Identificador del estudiante
numero_apartamento	int	Identificador del apartamento donde vive el estudiante

Tabla 62. Trabajador

Nombre: trabajador		
Atributo	**Tipo**	**Descripción**
solapin	char	Identificador del personal que es trabajador
nombre	varchar	El nombre del trabajador
apellido	varchar	Los apellidos del trabajador
sexo	varchar	El sexo del trabajador
edad	int	La edad del trabajador
teléfono	char	El teléfono del trabajador
correo	varchar	El correo del trabajador
area_trabajo	varchar	El área de trabajador

Tabla 63. Estudiante_asignatura

Nombre: estudiante_asignatura		
Atributo	**Tipo**	**Descripción**
solapin	char	Identificador del personal
nombre_asignatura	varchar	Identificador de la asignatura

Tabla 64. Grupo

Nombre: grupo		
Atributo	**Tipo**	**Descripción**
numero_grupo	char	Identificador del grupo
aula	int	El número de aula

Tabla 65. Asignatura

Nombre: asignatura		
Atributo	**Tipo**	**Descripción**
nombre_asignatura	varchar	Identificador de la asignatura
leyenda	varchar	Define la frase o la letra que define el nombre de la asignatura

Tabla 66. Profesor_asignatura

Nombre: profesor_asignatura		
Atributo	**Tipo**	**Descripción**
solapin	char	Identificador del profesor que imparte la asignatura
nombre_asignatura	varchar	Identificador de la asignatura que imparte el profesor en ese semestre

Tabla 67. Curso_posgrado

Nombre: curso_posgrado		
Atributo	**Tipo**	**Descripción**
solapin	char	Identificador del personal
curso	varchar	El nombre del curso

Tabla 68. Profesor

Nombre: profesor		
Atributo	**Tipo**	**Descripción**
solapin	char	Identificador del personal que es profesor
nombre	varchar	El nombre del profesor
apellido	varchar	Los apellidos del profesor
sexo	varchar	El sexo del profesor
edad	int	La edad del profesor
teléfono	char	El teléfono del profesor
correo	varchar	El correo del profesor

id_departamento	int	Identificador del departamento donde viven los profesores
numero_facultad	int	Identificador de la facultad donde imparte clase un profesor
numero_grupo	int	Identificador del grupo donde da clase el profesor
turno	int	Define la hora en comienza y termina una asignatura

Tabla 69. Rama

Nombre: rama		
Atributo	**Tipo**	**Descripción**
nombre_rama	varchar	Define la rama que representa una persona en ese cargo
nombre_cargo	varchar	Identificador del cargo que ocupa en esa rama.

Tabla 70. Organización

Nombre: organización		
Atributo	**Tipo**	**Descripción**
id_organizacion	int	Identificador de la organización
nombre_organizacion	varchar	El nombre de la organización

Tabla 71. Cargo

Nombre: cargo		
Atributo	**Tipo**	**Descripción**
nombre_cargo	varchar	Identificador del cargo
solapin	char	Identificador del personal que ocupa un cargo
id_organizacion	int	Identificador de la organización a la que pertenece ese cargo

Tabla 72. Departamento

Nombre: departamento		
Atributo	**Tipo**	**Descripción**
id_departamento	int	Identificador del departamento
nombre_departamento	varchar	El nombre del departamento
numero_facultad	int	Identificador de la facultad

2.6 Modelo relacional

Partiendo del esquema conceptual se obtuvo el esquema lógico que es básicamente la descripción de la estructura de la base de datos en términos de las estructuras de datos que puede procesar un tipo de SGBD.

La metodología que se siguió para el diseño lógico en el modelo relacional estuvo compuesta por varios pasos que se detallan a continuación.

1. Se convirtieron los esquemas conceptuales en esquemas lógicos.
2. Derivar un conjunto de relaciones (tablas) para cada esquema lógico.
3. Validar cada esquema mediante la normalización.
4. Dibujar el diagrama E-R.
5. Definir las restricciones de integridad.

2.6.1 Modelo relacional para la Base de datos de noticias

Noticia (**id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

Imagen (**código_imagen**, id_noticia)

Avisos (tipo_aviso, id_categoria, **id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

ciencia_tecnicas (**id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

cultural (**id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

deportiva (**id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

efemérides (**id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

internacional (**id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

nacional(**id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

productiva (**id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

universitaria (**id_noticia**, fecha_publicada, titulo_noticia, articulo_noticia, fecha_retirada, productor_noticia, editor_noticia, decisor_noticia, publicador_noticia)

categoría (**id_categoria**, tipo_categoria)

2.6.2 Modelo relacional para la Base de datos de facultades

facultad (**numero_facultad**, perfil)

lab_productivo (**nombre_proyecto**)

manzana (**numero_manzana**)

manzana_facultad (**numero_manzana**, **numero_facultad**)

edificio (**numero_edificio**, numero_manzana, solapin)

apartamento (**numero_apartamento**, numero_edificio, solapin)

galería_imagenes (**código_imagen**, numero_facultad)

horario (**turno**, numero_grupo, numero_semana)

semana (**numero_semana**, dia_semana)

evento (**tipo_evento**, numero_facultad)

departamento (**id_departamento**, nombre_departamento, **numero_facultad**)

especialidades (**nombre_especialidad**, id_departamento, numero_facultad)

directivos (**cargo_directivo**, solapin)

personal (**solapin**, numero_facultad, nombre_proyecto, nombre, apellidos, sexo, edad, teléfono, correo)

estudiante (**solapin**, numero_grupo, numero_apartamento, nombre, apellidos, sexo, edad, teléfono, correo)

trabajador (**solapin**, area_trabajo, nombre, apellidos, sexo, edad, teléfono, correo)

profesor (**solapin**, id_departamento, numero_facultad, numero_grupo, turno, nombre, apellidos, sexo, edad, teléfono, correo)

estudiante_asignatura (**nombre_asignatura**, **solapin**)

grupo (**numero_grupo**, aula)

asignatura (**nombre_asignatura**, leyenda)

profesor_asignatura (**solapin**, **nombre_asignatura**, semestre)

rama (**rama_ocupada**, nombre_cargo)

organización (**id_organizacion**, nombre_organizacion)

cargo (**nombre_cargo**, id_organizacion, solapin)

En este capítulo han sido expuestos los diagramas de clases persistentes realizados para las bases de datos Noticia y Facultad. Después de realizados los diagramas de clases persistentes se obtuvieron los diagramas de E-R donde se plasmó la representación de las clases y sus asociaciones. Además se construyeron tablas para mostrar los atributos y sus responsabilidades.

Capítulo 3: Validación del diseño realizado

Cuando se quiere obtener un buen diseño de bases de datos es necesario validar dicho diseño, o sea, garantizar la integridad, realizar la normalización, eliminar redundancias e implementar mecanismos de seguridad. Con el propósito de lograr el objetivo de este capítulo se aplicaron reglas de integridad, se llevaron a tercera forma normal las bases de datos Noticia y Facultad. Una vez terminado este proceso se hizo un análisis de redundancias para eliminar toda posibilidad de existencia de las mismas y se crearon mecanismos para evitar el acceso a la base de datos de Noticias por personas no autorizadas y para hacer un seguimiento de las acciones que se llevan a cabo en dichas bases de datos.

3.1 Integridad

La integridad de datos se refiere al estado de corrección y completitud de los datos ingresados en una base de datos. Los SGBD relacionales deben encargarse de mantener la integridad de los datos almacenados en una base de datos con respecto a las reglas predefinidas o restricciones.(ANÓNIMO 2006)

Para lograr la integridad de las base de datos Noticia y Facultad fue necesario aplicarle las restricciones de integridad (datos requeridos, restricciones de dominio, restricciones de validación, integridad de entidades, integridad referencial y las reglas de dominio) al modelo E_R para garantizar que los datos de esta estructura fuesen correctos.

Las restricciones de datos requeridos se refieren a que hay atributos que deben contener valores en todo momento es decir, no admiten nulos.(CHAVES., CARLOS ALBERTO GARCÍA 2007b)

En la base de datos Noticia no se definió como nulo el valor de ninguno de los atributos, debido a que se conocían los valores de los atributos y/o eran necesarios dichos valores.

Las restricciones de dominio se refieren a que todos los atributos tienen un dominio asociado que es el conjunto de valores que puede tomar. (CHAVES., CARLOS ALBERTO GARCÍA 2007b)

En las bases de datos Noticia y Facultad los atributos de cada tabla tienen un dominio que no es más que el tipo de dato que tiene cada uno de los atributos, es decir, en cada atributo a la hora de insertar un registro

este debe ser consistente con el tipo de dato de la columna en caso contrario este lanzará un error de no consistencia de los datos.

Ejemplo:

```
CREATE TABLE aviso
(
      id_noticia int4 NOT NULL,
      fecha_publicada date NOT NULL,
      titulo_noticia varchar(50) NOT NULL,
      articulo_noticia text NOT NULL,
      fecha_retirada date NOT NULL,
      productor_noticia varchar(50) NOT NULL,
      editor_noticia varchar(50) NOT NULL,
      decisor_noticia varchar(50) NOT NULL,
      publicador_noticia varchar(50) NOT NULL,
      id_categ int4 NOT NULL,
      tipo_aviso varchar(50) NOT NULL,
      CONSTRAINT aviso_id_categ_fkey FOREIGN KEY (id_categ) REFERENCES categoria
(id_categoria) ON UPDATE CASCADE ON DELETE CASCADE,
      CONSTRAINT comprobar CHECK (fecha_retirada > fecha_publicada)
)
INHERITS (noticia)
```

La integridad de entidades se refiere a que a que el identificador de una tabla no pude ser nulo por lo tanto las claves primarios de las relaciones no admiten nulos. (CHAVES., CARLOS ALBERTO GARCÍA 2007b)

En la base de datos Noticia ningún atributo llave de la relación es nulo es decir en cada relación existe un atributo que lo identifica por lo tanto este no puede ser nulo, este valor es el que identifica a cada tupla de la relación. Un ejemplo es en la relación noticia con la tabla imagen donde la cardinalidad es de 1 a 0.1, podría pasarse la llave foránea para cualquier tabla de las dos pero más factible sería llevar la llave primaria de noticia como foránea de imagen de modo que para cada imagen exista una noticia, porque si fuera de otro modo en la llave noticia se iba a tener muchos valores nulos y eso provocaría la utilización de más memoria a la hora de devolver datos y no sería óptima esa base de datos.

Ejemplo:

```
CREATE TABLE noticia
(
Id_noticia int4 NOT NULL,
fecha_publicada date NOT NULL,
titulo_noticia varchar (50) NOT NULL,
articulo_noticia text NOT NULL,
fecha_retirada date NOT NULL,
productor_noticia varchar (50) NOT NULL,
editor_noticia varchar (50) NOT NULL,
decisor_noticia varchar (50) NOT NULL,
publicador_noticia varchar (50) NOT NULL,
CONSTRAINT "Noticia_pkey" PRIMARY KEY (id_noticia)
)
WITHOUT OIDS;
```

La integridad referencial plantea que una llave foránea enlaza cada tupla de la relación hijo con la tupla de la relación padre que tiene el mismo valor en su llave primaria. La integridad referencial dice que si una clave ajena tiene un valor (si es no nula), ese valor debe ser uno de los valores de la clave primaria a la que referencia. (CHAVES., CARLOS ALBERTO GARCÍA 2007b)

Para que se cumpla la integridad referencial hay que tener en cuenta varios aspectos sobre las llaves foráneas, como es el caso en que la participación de la entidad hijo en la relación sea total, la clave ajena no admite nulos, en cambio si la relación es parcial, la clave debe aceptar nulos.

Cuando se quiere borrar una tupla que está siendo referenciada por otra tupla a través de una clave ajena hay varias formas posibles para hacerlo. Una sería establecer que no se puedan borrar tuplas que están siendo refenciadas por otras tuplas. Otra sería propagar que no es más que borrar la tupla deseada y propagar el borrado a todas las tuplas que le hacen referencia. Otra forma sería anular que consiste en borrar la tupla deseada y todas las referencias que tenía se ponen automáticamente a nulo, aunque esto solo es posible si la clave ajena acepta nulos. También podría utilizarse la vía de valor por defecto en la que una vez borrada la tupla deseada todas las referencias toman automáticamente el valor por defecto aunque este opción solo es válida si se ha especificado un valor por defecto para la clave ajena.

En el caso que se quiera modificar la clave primaria de una tupla que esta siendo referenciada por otra tupla a través de una clave ajena se puede hacer de la misma forma que cuando se quiere borrar una tupla que esta siendo referenciada por otra tupla a través de la clave ajena.

La integridad referencial en la base de datos de Noticias se manifiesta a la hora de devolver cualquier dato de la tabla, es decir cualquier atributo no primo de la relación se debe hacer referencia primero a el atributo llave tal que esta sea igual donde tiene la referencia. Aquí se utiliza para este tipo de relación la sentencia JOIN, INNER JOIN.

Ejemplo:

```
CREATE OR REPLACE FUNCTION avisos_de_una_categoria (int4)
RETURNS SETOF record AS
$BODY$declare
var record;
begin
for var in select tipo_aviso,tipo_categoria from aviso inner join categoria on
categoria.id_categoria=aviso.id_categ where categoria.id_categ=$1
loop
return next var;
end loop;
return;
end$BODY$
LANGUAGE 'plpgsql' VOLATILE;
ALTER FUNCTION avisos_de_una_categoria(int4) OWNER TO postgres;
```

A la hora de borrar una tupla por ejemplo si se borra una tupla en la tabla Facultad, es decir si se borra la llave primaria que es número _ facultad donde quiera que esté ese número _ facultad de referencia debe borrarse, por ejemplo en departamento, si se elimina una facultad, se tienen que eliminar todos los departamentos que pertenecen a esa facultad. Para eso PostgreSQL que es el SGBD que se va a utilizar cuenta con una opción de eliminación en cascada (Cascade) de modo que si se eliminan en una, elimina todas las referencias en la otra relación. Así mismo ocurre a la hora de actualizar o modificar en una tabla si se activa la opción de CASCADE actualiza en las relaciones a las que hace referencia esa entidad. A la hora de insertar datos en una relación que tuviera llave foránea se tendría que verificar que el valor que se fuera a insertar estuviera presente en la relación a la que se hace referencia.

Ejemplo:

```
CREATE TABLE aviso
(
      id_noticia int4 NOT NULL,
      fecha_publicada date NOT NULL,
      titulo_noticia varchar(50) NOT NULL,
      articulo_noticia text NOT NULL,
      fecha_retirada date NOT NULL,
      productor_noticia varchar(50) NOT NULL,
      editor_noticia varchar(50) NOT NULL,
      decisor_noticia varchar(50) NOT NULL,
      publicador_noticia varchar(50) NOT NULL,
      id_categ int4 NOT NULL,
      tipo_aviso varchar(50) NOT NULL,
      CONSTRAINT aviso_id_categ_fkey FOREIGN KEY (id_categ) REFERENCES categoria
(id_categoria) ON UPDATE CASCADE ON DELETE CASCADE,
```

Las reglas de negocio exigen que cualquier operación que se realice sobre los datos deba cumplir las restricciones que impone el propio funcionamiento de la organización.(CHAVES., CARLOS ALBERTO GARCÍA 2007b)

Cualquier operación que se realice en la base de datos, la cual ha sido creada para la UCI por lo que sus datos deben estar en correspondencia con la universidad, por ejemplo a la hora de insertar las facultades no deben haber más de 10 facultades, ni más departamentos de los que han sido declarados en la universidad, es decir los datos deben cumplir las restricciones que impone el funcionamiento de la universidad. Estos datos pueden ser comprobados de distintas maneras o con los llamados CHECK o TRIGGERS.

Ejemplo:

```
CREATE TABLE  facultad(
numero_facultad  int4 NOT NULL,
perfil varchar(50) NOT NULL
CONSTRAINT comprobar  CHECK (numero_facultad >=1 and numero_facultad <=10)

)
```

Existen dos técnicas fundamentales para el trabajo en bases de datos relacionales una de ellas es la de diseño descendente, análisis o descomposición que es precisamente la técnica de la normalización.

3.2 Normalización de la base de datos

La normalización es un proceso mediante el cual se descompone una relación que no cumple con una serie de requisitos en relaciones más pequeñas que si las cumplan. Tiene como objetivo eliminar redundancias y anomalías de inserción, eliminación y actualización.

Durante el proceso de la normalización se fue comprobando que cada relación (tabla) cumpliera con un conjunto de reglas basadas en la clave primaria y las dependencias funcionales. Se llevaron a cabo una serie de pasos donde cada paso responde a una forma normal.

3.2.1 Primera forma normal (1FN)

Para llevar cada relación a 1FN se eliminaron los grupos repetitivos. Un grupo repetitivo es el atributo o el conjunto de atributos que tiene múltiples valores para cada tupla de la relación.

Para eliminar los grupos repetitivos lo que se hizo fue poner cada uno de ellos en una relación aparte, heredando la clave primaria de la relación en la que se encontraban.

Aunque existe otra forma que consiste en repetir los atributos con un solo valor para cada valor del grupo repetitivo. De este modo, se introducen redundancias ya que se duplican valores, estas redundancias son eliminadas después mediante las restantes formas normales.

Después de hacer esto con cada una de las relaciones se logró que la base de datos quedara en 1FN. Todos los dominios de la misma contienen valores atómicos, es decir, sin grupos repetitivos. En las relaciones de las tablas se ve que tienen un solo valor en la intersección de cada fila con cada columna.(CHAVES., CARLOS ALBERTO GARCÍA 2007a)

3.2.2 Segunda forma normal (2FN)

Una relación está en segunda forma normal si, y sólo si, está en 1FN y, además, cada atributo que no está en la clave primaria es completamente dependiente de la clave primaria. (GARCÍA 2007)

Después de llevar todas las relaciones a 1FN para pasarlas a 2FN se eliminaron las dependencias parciales de las claves primarias. Para ello, se eliminaron los atributos que son funcionalmente dependientes de cada relación y se colocaron en una nueva relación con una copia de su determinante (los atributos de la clave primaria de los que dependen). La 2FN se aplica a las relaciones que tienen claves primarias compuestas por dos o más atributos.

Luego de este proceso las bases de datos quedaron en 2FN porque aparte de estar en 1FN que es uno de los requisitos para estar en 2FN, se logró que cada atributo no primo de la relación fuese completamente dependiente de la clave primaria. De esta forma se evitan anomalías a la hora de realizar actualizaciones.

3.2.3 Tercera forma normal (3FN)

Una relación está en tercera forma normal si, y sólo si, está en 2FN y, además, cada atributo que no está en la clave primaria no depende transitivamente de la clave primaria. (CHAVES., .CARLOS ALBERTO GARCÍA)

Una vez llevada cada una de las relaciones a 2FN se pasó a eliminar las dependencias transitivas para obtener la 3FN con el objetivo de eliminar las redundancias que aún quedaban de la 2FN para evitar sufrir anomalías frente a las actualizaciones. Para ello, se eliminaron los atributos de cada relación que dependían transitivamente y se pusieron en nuevas relaciones con una copia de sus determinantes (el atributo o atributos no clave de los que dependen).

Finalmente las bases de datos quedaron en 3FN debido a que cumplían con la primera condición, estar en 2FN y además cada atributo que no está en la clave primaria no depende transitivamente de la clave primaria. La dependencia es transitiva si existen las dependencias, siendo, atributos o conjuntos de atributos de una misma relación.

3.3 Análisis de redundancias

Se dice que hay redundancia de datos cuando la misma información es almacenada varias veces en la misma base de datos. Esto es siempre algo que se debe evitar, la redundancia dificulta la tarea de modificación de datos, y es el motivo más frecuente de inconsistencia de datos. Además requiere un mayor espacio de almacenamiento, que influye en mayor coste y mayor tiempo de acceso a los datos.

Después de llevar las bases de datos de Noticias y de Facultades a 3FN quedaron libres de redundancias aunque en caso que se quisiera agrupar en relaciones más grandes alguna de las bases de datos o ambas se podría utilizar la técnica de **Síntesis Relacional** que precisamente consiste en agrupar las relaciones en otras más grandes a conveniencia del proceso de negocio.

3.4 Análisis de la seguridad de la base de datos

La seguridad es un punto esencial en las bases de datos para evitar ataques, impedir cualquier acceso no autorizado, con la intensión de modificar, usar y/o difundir información almacenada en las bases de datos. El SGBD a utilizar para diseñar las bases de datos de la Intranet Corporativa de la UCI incluye formas de restringir el acceso al sistema. Esta función se denomina **control de acceso** y se pone en prácticas creando cuentas de usuarios y contraseñas para que el SGBD controle el proceso de entrada al sistema.

El administrador de la base de datos (ABD) juega un papel importante en la seguridad de la base de datos porque es quien otorga privilegios a los usuarios, los clasifica, así como a los datos. Las órdenes de los ABD incluyen las siguientes acciones:

1. Creación de cuentas.

2. Concesión de privilegios.

3. Revocación de privilegios.

La creación de cuentas sirve para controlar el acceso al SGBD en general, la concesión y revocación de privilegios para controlar autorizaciones discrecionales. Estas autorizaciones constituyen un importante mecanismo de seguridad.

Para garantizar la seguridad de la base de datos de Noticias se utilizó el control de acceso discrecional que consiste en prevenir de accesos no autorizados a la base de datos y está basado en los derechos de acceso o privilegios y mecanismos para darle al usuario tales privilegios. El acceso discrecional es un modo de restringir el acceso a la información basado en privilegios. Existen dos niveles de asignación de privilegios:

• **Nivel de cuenta:** En este nivel el administrador especifica los privilegios particulares que tiene cada usuario, independiente de las tablas de la BD (CREATE TABLE, CREATE VIEW, ALTER, MODIFY, SELECT). Estas son autorizaciones a nivel de usuario dígase: usuario, cuenta, programa.

En la base de datos de Noticias existe un grupo que tiene el rol de administrador que es el que otorga privilegios para crear tablas, crear vistas.

Ejemplo:

```
CREATE DATABASE intranet
  WITH OWNER = postgres
       ENCODING = 'UNICODE'
       TABLESPACE = pg_default;
GRANT TEMPORARY ON DATABASE "Intranet" TO public;
GRANT ALL ON DATABASE "Intranet" TO otros;
GRANT CREATE ON DATABASE "Intranet" TO prueba;


ALTER TABLE seleccionar_datos_noticia OWNER TO usuario;
GRANT ALL ON TABLE seleccionar_datos_noticia TO usuario;
GRANT SELECT, UPDATE, INSERT ON TABLE seleccionar_datos_noticia TO GROUP
administradores;
```

Ejemplo:

```
CREATE TABLE categoria
(
  id_categoria int4 NOT NULL,
```

```
  tipo_categoria varchar(50) NOT NULL,
  CONSTRAINT categoria_pkey PRIMARY KEY (id_categoria)
)
WITHOUT OIDS;
ALTER TABLE categoria OWNER TO postgres;
GRANT ALL ON TABLE categoria TO postgres;
GRANT SELECT, UPDATE, INSERT, DELETE ON TABLE categoria TO GROUP postgres;
```

Las vistas son poderosas herramientas, ocultando partes de la base de datos a ciertos usuarios. El usuario no sabrá que existen aquellos atributos que se han omitido al definir una vista.

Ejemplo:

```
CREATE OR REPLACE VIEW seleccionar_datos_noticia AS
SELECT noticia.titulo_noticia, noticia.articulo_noticia, noticia.fecha_publicada
FROM noticia;
```

• **Nivel de relación:** En este nivel se controlan los privilegios para tener acceso a cada relación o vista individual. Cada tabla de BD tiene asignada una cuenta propietario, que tiene todos los privilegios sobre esa tabla (DELETE, INSERT, UPDATE, SELECT) y se encarga de otorgarlos al resto de cuentas. Estas son autorizaciones a nivel de sujeto dígase: relación, columna, tupla, vista.

En la base de datos Noticia aparte de la cuenta del sistema hay un usuario publicador con el rol de administrador que tiene otorgados todos los privilegios a nivel de cuenta, es decir posee las autorizaciones de selección (SELECT), inserción (INSERT), actualización UPDATE y borrado (DELETE).

Ejemplo:

```
GRANT UPDATE ON noticia TO prueba
```

3.5 Trazabilidad de las acciones

La trazabilidad es un sistema que deja rastro de las acciones que el usuario realiza con los datos. Por ejemplo, si un usuario elimina un registro, queda constancia del registro que se eliminó, quien lo eliminó y cuando. De esta forma, puede hacerse un seguimiento de todas las acciones que se han realizado sobre un registro concreto. O todas las acciones que ha realizado un usuario concreto. Pudiendo conformar un

histórico de las acciones sobre los datos. La trazabilidad puede realizarse sobre tablas o sobre procesos. Incluso pueden crearse trazabilidades sobre errores o conflictos de la aplicación, *(logs)*. (GALIANO 2006)

Los logs son un excelente mecanismo que tienen los SGBD para seguir las trazas en bases de datos. Estos son usados para registrar datos o información sobre (quién, qué, cuándo, dónde) convirtiéndose en un registro oficial de eventos durante un período de tiempo en particular, de esta manera el ABD tendrá un efectivo archivo de evidencias que podrá utilizar en cualquier momento que necesite.

En el SGBD PostgreSQL que fue el que se utilizó para implementar la base de datos Noticia los logs se encuentran en archivos de texto en la carpeta pg_log en ellos se registra la fecha, hora y acción realizada en la base de datos.

Ejemplo:

2007-04-16 19:56:33 LOG: database system was interrupted at 2007-04-16 14:22:43 Hora est. Or. (EE.UU. y Canadá
2007-04-16 19:56:33 LOG: checkpoint record is at 0/C78F68
2007-04-16 19:56:33 LOG: redo record is at 0/C78F68; undo record is at 0/0; shutdown FALSE
2007-04-16 19:56:33 LOG: next transaction ID: 1997; next OID: 25422
2007-04-16 19:56:33 LOG: database system was not properly shut down; automatic recovery in progress
2007-04-16 19:56:34 LOG: record with zero length at 0/C78FA8
2007-04-16 19:56:34 LOG: redo is not required
2007-04-16 19:56:35 LOG: database system is ready

Los logs son el mecanismo que trae por defecto el SGBD para garantizar la trazabilidad de las acciones, sin embargo se pueden utilizar los *triggers* (desencadenadores) que son un tipo de procedimiento almacenado que va a entrar en vigor antes (BEFORE), después (AFTER) o en lugar de (INSTEAD OF) efectuarse una operación cualquiera de modificación de datos sobre una tabla específica, que puede ser DELETE, INSERT o UPDATE.

Con el objetivo de seguir las trazas de las acciones realizadas en la base de datos de Noticias se hizo un trigger llamado Registro que se activa después (AFFTER) de ejecutar cualquiera de las siguientes operaciones: DELETE, INSERT o UPDATE sobre una relación. Este resultado se almacena en una tabla llamada de igual forma (Registro). En dicha tabla se acumulan las operaciones realizadas por los usuarios, la fecha y hora, la tabla en que se ejecutó la operación y el nombre del usuario que la llevó a cabo.

Ejemplo:

```
CREATE OR REPLACE FUNCTION registro()
  RETURNS "trigger" AS
$BODY$declare
tabla varchar;
 BEGIN
    select into tabla TG_RELNAME;
     IF (TG_OP = 'DELETE') THEN
        INSERT INTO registro SELECT 'D', now(), user,tabla;
         RETURN OLD;
     ELSIF (TG_OP = 'UPDATE') THEN
        INSERT INTO registro SELECT 'U', now(), user,tabla;
        RETURN NEW;
     ELSIF (TG_OP = 'INSERT') THEN
        INSERT INTO registro SELECT 'I', now(), user,tabla;
```

```
            RETURN NEW;
      END IF;
            RETURN NULL;
end$BODY$
```

En este capítulo se hizo la validación teórica del diseño, para lograrlo se tuvieron en cuenta una serie de aspectos. Uno de ellos consistió en aplicar reglas de integridad como son las restricciones de datos requeridos, restricciones de dominio, integridad de entidades, integridad referencial y las reglas de negocio. También se normalizaron las bases de datos Noticia y Facultad hasta 3FN para eliminar redundancias e inconsistencias. Se utilizaron mecanismos para garantizar la seguridad, tales como el control de acceso y las vistas, así como mecanismos para seguir la trazabilidad de las acciones como los logs y los triggers.

Conclusiones

La creación de una nueva Intranet para la UCI es una alternativa necesaria para propiciar el buen funcionamiento de la universidad en general, porque a través de esta se mantendrá intercomunicada la comunidad universitaria e informada de todo cuando acontece o acontecerá. Los usuarios también podrán contar con una gran cantidad de servicios. Para que la Intranet tenga todas las funcionalidades requeridas y sea posible representar todas las áreas funcionales de la UCI debe estar respaldada por bases de datos que almacenen, procesen y respondan a las diferentes peticiones de los usuarios.

Como resultado de esta investigación, se diseñó una base de datos para el servicio de Noticias que ofrece la Intranet actual que se pondrá en práctica en la nueva intranet. Dicha base de datos es totalmente funcional, respecto a rendimiento, integridad, seguridad, libre de redundancias e inconsistencias. También permite seguir la trazabilidad de las acciones que se realizan sobre ella. Además se hizo una propuesta de diseño de base de datos para las Facultades.

Para realizar este trabajo fue necesario darle solución a un grupo de problemas relacionados con la organización del proceso, siguiendo una serie de pasos definidos por los estándares de diseño de bases de datos. Se seleccionó a RUP como metodología de desarrollo de software. Para la descripción del diseño se empleó UML sobre Visual Paradimg. El Gestor de Bases de Datos relacional seleccionado fue PostgreSQL.

Recomendaciones

- Analizar la propuesta de diseño realizada para la base de datos Facultad y en caso que sea aceptada que se implemente.
- Realizar el levantamiento de requisitos necesarios para continuar con el diseño de las restantes bases de datos de las diferentes áreas funcionales de la UCI.
- Hacer un Almacén de Datos para controlar los procesos de negocio una vez terminadas las bases de datos de Intranet Corporativa de la UCI.

Referencias Bibliográficas

1. ANÓNIMO. Sistemas de Información, 2006.

2. PERRAUT, R. P. *ANALISIS DE FUENTES DE INFORMACION DE ESTUDIOS DE TRADUCCION: CREACION DE UNA BASE DE DATOS*, 1997. p.

3. Sistema de gestión de base de datos. Disponible en: http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_base_de_datos.

4. Microsoft SQL Server . . 2007. [Disponible en: http://es.wikipedia.org/wiki/Microsoft_SQL_Server.

5. MySQL. . 2007. [Disponible en: http://es.wikipedia.org/wiki/MySQL

6. Oracle. Introducion. Monografias.com. , 2007.

7. MySQL 2007, Disponible en: http://www.software-shop.com/Productos/MySQL/mysql.html

8. PostgreSQL. . 2007, Disponible en: http://www.http-peru.com/postgresql1.php

9. MARQUÉS, M. M. *Metodología de diseño de bases de datos.* , 2001. [Disponible en: http://www3.uji.es/~mmarques/f47/apun/node81.html.

10. ANDRÉS, M. M. M. *Ciclo de vida de las aplicaciones de bases de datos. Diseño de la base de datos*, 2001. [Disponible en: http://www3.uji.es/~mmarques/f47/apun/node67.html

11. Lenguaje Unificado de Modelado. 2007. [Disponible en: http://es.wikipedia.org/wiki/Lenguaje_Unificado_de_Modelado

12. ANONIMO. *Proceso Unificado del Rational.* Disponible en: http://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational.

13. SANCHEZ, M. A. M. *Metodologías De Desarrollo de Software. Extreme Programing (XP)*, 2004. [Disponible en: http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

14. ---. *Metodologias de desarrollo de Software. Microsoft Solution Framework (MSF).* 2007. [Disponible en: http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html

15. ANONIMO. *Rational Unified Process* Disponible en: http://es.wikipedia.org/wiki/Rational_Unified_Process

16. VIZCAÍNO, A. *Visual Paradimg*, 2007. [Disponible en: http://alarcos.infcr.uclm.es/per/fgarcia/isoftware/doc/LabTr1_VP.pdf

17. ANONIMO.*PostgreSQL.TiendaLinux.com* Disponible en: http://soporte.tiendalinux.com/portal/Portfolio/postgresql_ventajas_html.

18. El Optimizador de Consultas de Sybase Adaptive Server Enterprise. Disponible en: http://macine.epublish.cl/tesis/index-Definici.html.

19. CONFERENCIA#3. *Técnicas de Recopilación de Información. Flujo de trabajo Captura de requisitos. Continuacion.* . Conferencia 3, 2006. p.

20. *Modelo de Datos Relacional* Disponible en: http://macine.epublish.cl/tesis/index-Conclusi.html.

21. *Diccionario Informático. Definición de Integridad de datos* Disponible en: http://www.alegsa.com.ar/Dic/integridad%20de%20datos.php.

22. CHAVES., C. A. G. *Diseño de base de datos relacionales. VIII Diseño lógico de bases de datos. Normalización. Tercera Forma Normal (3raFN)* Disponible en: http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm

23. ---. *Diseño de base de datos relacionales. VIII Diseño lógico de bases de datos. Normalización. Primera Forma Normal (1raFN)* 2007a. [Disponible en: http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm

24. GARCÍA, C. A. *Diseño de base de datos relacionales. VIII Diseño lógico de bases de datos. Normalización. Segunda Forma Normal (2daFN)*, 2007. [Disponible en: http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm

25. ---. *Diseño de base de datos relacionales.Diseño lógico de bases de datos . definir restricciones de integridad.* , 2007b. [Disponible en: http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm.

26. GALIANO, J. L. M. *Generadores de código*, 2006.

Bibliografía

1. ANÓNIMO. Sistemas de Información, 2006.

2. *Modelo de Datos Relacional* Disponible en: http://macine.epublish.cl/tesis/index-Conclusi.html.

3. *Diccionario Informático. Definición de Integridad de datos* Disponible en: http://www.alegsa.com.ar/Dic/integridad%20de%20datos.php.

4. ANDRÉS, M. M. M. *Ciclo de vida de las aplicaciones de bases de datos. Diseño de la base de datos.* Disponible en: http://www3.uji.es/~mmarques/f47/apun/node67.html. .

5. ANONIMO. *Rational Unified Process*

Disponible en: http://es.wikipedia.org/wiki/Rational_Unified_Process

6. ---. *Base de datos. Modelo de bases de datos.* Disponible en: http://es.wikipedia.org/wiki/Base_de_datos.

7. ---. *Sistema de gestión de base de datos.* Disponible en: http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_base_de_datos. .

8. ---. *Microsoft SQL Server .* Disponible en: http://es.wikipedia.org/wiki/Microsoft_SQL_Server.

9. ---. *Oracle. Introducion. Monografias.com.* Disponible en: http://www.monografias.com/trabajos25/oracle/oracle.shtml. .

10. ---. *MySQL.* Disponible en: http://es.wikipedia.org/wiki/MySQL

11. ---. *MySQL* Disponible en: http://www.software-shop.com/Productos/MySQL/mysql.html.

12. ---. *PostgreSQL.* Disponible en: http://www.http-peru.com/postgresql1.php

13. ---. *Lenguaje Unificado de Modelado.* Disponible en: http://es.wikipedia.org/wiki/Lenguaje_Unificado_de_Modelado.

14. ---. *Proceso Unificado del Rational.* Disponible en: http://es.wikipedia.org/wiki/Proceso_Unificado_de_Rational .

15. ---. *Metodologias de desarrollo de Software. Microsoft Solution Framework (MSF).* Disponible en: http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html .

16. ---. *PostgreSQL.TiendaLinux.com* Disponible en: http://soporte.tiendalinux.com/portal/Portfolio/postgresql_ventajas_html. .

17. ---. *El Optimizador de Consultas de Sybase Adaptive Server Enterprise.* Disponible en: http://macine.epublish.cl/tesis/index-Definici.html.

18. CONFERENCIA#3. Técnicas de Recopilación de Información. Flujo de trabajo Captura de requisitos. Continuacion. . Conferencia 3, 2006. p.

19. CHAVES., C. A. G. *Diseño de base de datos relacionales.Diseño lógico de bases de datos . definir restricciones de integridad.* Disponible en: http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm .

20. MARQUÉS, M. M. *Metodología de diseño de bases de datos.* Disponible en: http://www3.uji.es/~mmarques/f47/apun/node81.html. .

21. PERRAUT., R. P. *ANALISIS DE FUENTES DE INFORMACION DE ESTUDIOS DE TRADUCCION: CREACION DE UNA BASE DE DATOS. 1997.* 2006.

22. SANCHEZ, M. A. M. *Metodologías De Desarrollo de Software. Extreme Programing (XP)* Disponible en: http://www.informatizate.net/articulos/metodologias_de_desarrollo_de_software_07062004.html .

23. VIZCAÍNO. *Visual Paradimg* Disponible en: http://alarcos.inf-cr.uclm.es/per/fgarcia/isoftware/doc/LabTr1_VP.pdf .

24. GARCÍA, C. A. *Diseño de base de datos relacionales. VIII Diseño lógico de bases de datos. Normalización. Segunda Forma Normal (2daFN)*, 2007. [Disponible en: http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm

25. ---. *Diseño de base de datos relacionales.Diseño lógico de bases de datos . definir restricciones de integridad.* , 2007b. [Disponible en: http://www.mailxmail.com/curso/informatica/disenobasesdatosrelacionales/capitulo8.htm

26. GALIANO, J. L. M. *Generadores de código*, 2006.

Glosario de Términos

Algebra relacional: lenguaje de consulta procedural.

Aplicación: programa con el cual el usuario final interactúa a través de una interfaz y que realizan tareas útiles para éste.

Atributo (columna o campo): cada una de las características que posee una entidad, y que agrupadas permiten distinguirla de otras entidades del mismo conjunto.

C: es un lenguaje de programación orientado a la implementación de Sistemas Operativos, es el más popular para crear software de sistemas, aunque también se utiliza para crear aplicaciones.

C++: es un lenguaje que abarca tres paradigmas de la programación: la programación estructurada, la programación genérica y la programación orientada a objetos. Las principales características del C++ son el soporte para programación orientada a objetos y el soporte de plantillas o programación genérica (*templates*).

Cardinalidad (número de filas o registros): número de posibles relaciones que una entidad determinada puede tener sobre la asociación. Se especifica por la cantidad máxima y mínima de instancias de la asociación.

Clave o Llave: es un conjunto de atributos que identifican de forma unívoca una entidad.

Clave candidata: es cada una de las claves mínimas existente en un conjunto de entidades.

Clave principal (primaria): es una clave candidata elegida de forma arbitraria, que se usa siempre para identificar una entidad.

Código abierto: (del inglés ***open source***) es el término con el que se conoce al software distribuido y desarrollado libremente.

Código cerrado: es el código fuente que no se encuentra disponible para cualquier usuario, es decir no se hace público. Se le llama así en contraposición al código abierto.

Consultas o querys: petición al SGBD para que procese un determinado comando SQL. Esto incluye tanto peticiones de datos como creación de bases de datos, tablas, modificaciones, inserciones, etc.

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

Dominio: es el conjunto de valores que tiene un atributo.

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

Escalabilidad: propiedad deseable de un sistema, una red o un proceso, que indica su habilidad para, o bien manejar el crecimiento continuo de trabajo de manera fluida, o bien para estar preparado para hacerse más grande sin perder calidad en los servicios ofrecidos. En general, también se podría definir como la capacidad del sistema informático de cambiar su tamaño o configuración para adaptarse a las circunstancias cambiantes.

Especialización: es el proceso según el cual se crean varios tipos de entidades a partir de uno. Cada una de los conjuntos de entidades resultantes contendrá solo algunos de los atributos del conjunto original.

Estrategia: proceso regulable, conjunto de las reglas que aseguran una decisión óptima en cada momento.

Generalización: es el proceso según el cual se crea un conjunto de entidades a partir de otros que comparten ciertos atributos.

Grado de una relación: número de atributos en la relación.

Herramientas CASE: (Ingeniería de Software Asistida por Ordenador) son diversas aplicaciones informáticas que pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software en tareas

como el proceso de realizar un diseño del proyecto, calculo de costes, implementación de parte del código automáticamente con el diseño dado, compilación automática, documentación o detección de errores entre otras.

Identificador: un identificador de una entidad es un atributo o conjunto de atributos que determina de modo único cada ocurrencia de esa entidad.

Índices: instrumento que aumenta la velocidad de respuesta de la consulta, mejorando su rendimiento y optimizando su resultado.

Internet: es la gran colección de redes que están unidas y ubicadas en todo el mundo; así que los usuarios de cualquier red pueden contactar usuarios en cualquiera de las otras redes, obtener información y/o hecer uso de diferentes servicios.

Interrelación: es una correspondencia o asociación entre dos o más entidades. Cada relación tiene un nombre que describe su función.

Java: es un lenguaje de programación orientado a objetos. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel como punteros.

Lenguaje de definición de datos: lenguaje de definición de datos, es el lenguaje proporcionado por el Sistema Gestor de Base de Datos para llevar a cabo tareas de definición de estructuras de la base de datos.

Lenguaje de consulta: lenguaje en el que el usuario solicita información de la base de datos. Los lenguajes de consulta se pueden clasificar en procedurales y no procedurales.

Lenguaje de manipulación de datos: permite a los usuarios manipular los datos de la base de datos, realizar consultas, inserciones, eliminaciones y modificaciones.

Metodología: conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal.

Modelo de datos: conjunto de conceptos que sirven para describir la estructura de una base de datos: los datos, las relaciones entre los datos y las restricciones que deben cumplirse sobre los datos.

.NET: es un proyecto de Microsoft para crear una nueva plataforma de desarrollo de software con énfasis en transparencia de redes, con independencia de plataforma y que permita un rápido desarrollo de aplicaciones.

Optimizador de consulta: Este módulo determina la estrategia óptima para la ejecución de las consultas.

Perl: es un lenguaje de programación. Estructuralmente, Perl está basado en un estilo de bloques como los del C o AWK, y fue ampliamente adoptado por su destreza en el procesado de texto y no tener ninguna de las limitaciones de los otros lenguajes de script.

PHP: es un lenguaje de programación usado frecuentemente para la creación de contenido para sitios web con los cuales se puede programar las páginas html y los códigos de fuente.

Privilegios: Privilegio (etimológicamente *ley privada*, o sea, no general sino relativa a un individuo específico) el permiso para realizar una actividad garantizado por otra persona.

Python: es un lenguaje de programación, que actualmente se desarrolla como un proyecto de código abierto, administrado por la Python Software Fundación. Python es un lenguaje interpretado, lo que ahorra un tiempo considerable en el desarrollo del programa, pues no es necesario compilar ni enlazar.

Replicación: proceso de copia y mantenimiento de bases de datos múltiples que hacen un sistema distribuido. Consiste en dos servidores: uno maestro y uno esclavo. El maestro registra todas las consultas que provocan cambios en la base de datos y las almacena en un fichero de registro binario. El servidor esclavo se conecta al maestro, lee las condiciones en el fichero binario y las ejecuta contra su copia local.

Servidor: aplicación informática o programa que realiza algunas tareas en beneficio de otras aplicaciones llamadas clientes.

Software: componente intangible de una computadora, es decir, un conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica.

Software libre: es el software que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente.

Software privativo: se refiere a cualquier programa informático en el que los usuarios tienen limitadas las posibilidades de usarlo, modificarlo o redistribuirlo (con o sin modificaciones), o cuyo código fuente no está disponible o el acceso a éste se encuentra restringido.

Tecnología: conjunto de teorías y de técnicas que permiten el aprovechamiento práctico del conocimiento científico.

Transacción: conjunto de acciones llevadas a cabo por un usuario o un programa de aplicación, que acceden o cambian el contenido de la base de datos. Las transacciones representan eventos del mundo real. Estas transacciones se deben realizar sobre la base de datos para que esta siga siendo un fiel reflejo de la realidad.

Tupla (fila o registro): corresponde a una fila de una tabla.

Unicode: estándar industrial cuyo objetivo es proporcionar el medio por el que un texto en cualquier forma o idioma pueda ser codificado para el uso informático.

WWW: World Wide Web, o simplemente Web, es el universo de información accesible a través de Internet, una fuente inagotable del conocimiento humano. Usando el Web, se tiene acceso a millones de páginas de información. La exploración en el Web se realiza por medio de un software especial denominado Browser o Explorador.

Listado de Abreviaturas

ABD	Administrador de la Dase de Datos
ACID:	Analysis Console for Intrusion Databases
a.n.e	antes de nuestra era.
API	Application Programming Interface
BD	Base de Datos
CASE	Computer-Aided Software Engineering
CMS	Content management system
CPU	Central Processing Unit
E-R	Entidad - Relación
E/S	Entrada_Salida
FID	Federación Internacional de Información y Documentación
1FR	Primera Forma Normal
2FN	Segunda Forma Normal
3FN	Tercera Forma Normal

GPL	General Public License
HTTP	Hypertext Transfer Protocol
JDBC	Java Database Connectivity
LC	Lenguaje de Consulta
LDD	Lenguaje de Definición de Datos
LMD	Lenguaje de Manipulación de Datos
MSF	Microsoft Solution Framework
MVCC	Multiversión Concurrency Control
ODBC	Open DataBase Conectivity
OLAP	Online Analytical Processing
OLTP	On-Line Transactional Processing
RAM	Random Access Memory
RUP	Rational Unified Process
SGBD	Sistema Gestor de Base de Datos.
SOA	Service-Oriented Architecture
SQL	Structured Query Language

SSL	Secure Sockets Layer
UCI	Universidad de Ciencias Informáticas
UML	Unified Modeling Language
XP	xtreme Prog
ZODB	Zope Object Data