Universidad de las Ciencias Informáticas
Facultad 5

Integración de bases de datos relacionales y NoSQL para una Red Social de Aprendizaje en la UCI

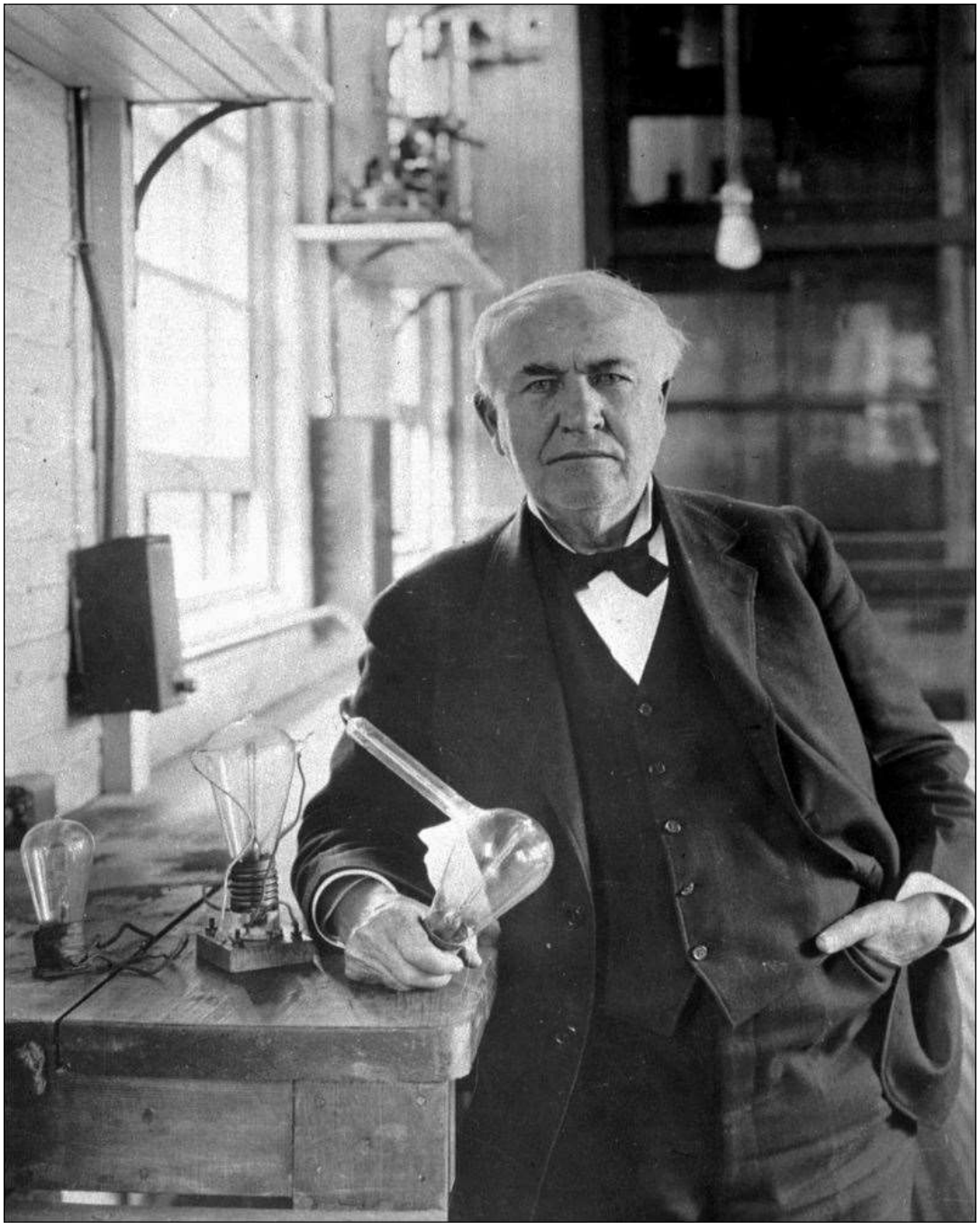
Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Yaikel López González

Tutores: Ing. Marcos Luis Ortíz Valmaseda

Ing. Enelis Blanca Cuba Rondón

"Año 57 de la Revolución"
La Habana, Cuba
Junio 2015



Quien no se resuelve a cultivar el hábito de pensar, se pierde el mayor placer de la vida.

Thomas Alva Edison

Declaración de autoría

Declaro ser el único autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____

Yaikel López González

Autor

_____ _____

Ing. Marcos Luis Ortíz Valmaseda

Tutor

Ing. Enelis Blanca Cuba Rondón

Tutor

Datos de contacto

Nombre y Apellidos: Marcos L. Ortiz Valmaseda
Ciudadanía: cubana
Institución: Universidad de las Ciencias Informáticas (UCI)
Título: Ingeniero en Ciencias Informáticas
Categoría Docente: Asistente
E-mail: mlortiz@uci.cu
Ocupación Actual: Profesor de Sistemas de Bases de Datos

Nombre y Apellidos: Enelis Blanca Cuba Rondón
Ciudadanía: cubana
Institución: Universidad de las Ciencias Informáticas (UCI)
Título: Ingeniero en Ciencias Informáticas
Categoría Docente: Ninguna
E-mail: ebcuba@uci.cu
Ocupación Actual: Jefa del Grupo de Inteligencia Empresarial

Nombre y Apellidos: Yaikel López González
Ciudadanía: cubana
Institución: Universidad de las Ciencias Informáticas (UCI)
E-mail: ylglez@estudiantes.uci.cu

Dedicatoria

Dedico este trabajo a la mujer más valiente y sacrificada que he conocido y conoceré en toda mi vida. A alguien a quien amo con toda la fuerza de mi ser y por quien me levanto todas las mañanas de este mundo. A alguien que nunca se ha puesto por delante de sus hijos pues para ella eso es lo fundamental y lo demás es secundario. A mi amiga, tutora y maestra incondicional de la vida, eres lo que más pienso en todo y a cada momento. Con todo el amor y respeto que mereces este resultado también es tuyo mamá.

A mi papá (mi Dios en la tierra), dudo mucho que haya alguien en este mundo que ame más a su padre como yo amo al mío. Ni viviendo y haciendo cien años por ti te pagaría todo lo que has hecho. No recuerdo en la edad que tengo que me hayas puesto un dedo encima y sin embargo eres la persona que más respeto le tengo. Nunca me he sentido solo ni triste cuando recuerdo el padre que tengo. Podría hacer una tesis solo para resaltar los bellos valores y la dependencia emocional que tengo de tu persona. Gracias de verdad mi viejo, nuca olvides que te adoro y si hoy estoy aquí que no te quepa ni la menor duda que es por ti.

A toda mi familia, tanto materna como paterna pues siempre me han considerado el centro de ambas.

Agradecimientos

A Kathrin por brindarme su amor, su amistad y por sobre todo por ser más que una novia, una escuela. Los momentos más fabulosos en esta universidad los viví a su lado y aunque lejos nunca ha dejado de estar presente. Te quiero y respeto siempre.

A mi tío Silvano y mi tío René, así como a mi abuela Caridad que aunque ya no están hicieron mucho por mí. Dios los tenga en la gloria.

A mis maravillosos hermanos Giselle, Bryan y el recién llegado Lain (mi puchungo), los tres son personas en sus totalidades adorables y nobles.

A mi madrasta Roque por siempre atenderme tan bien y contribuir de manera muy atenta en la elaboración de este tamaño.

A mis tutores Enelis y Marquito, cuando la cosa se puso fea de verdad ambos dieron el paso al frente y nunca en el transcurso de este trabajo dejaron de darme aliento. Discúlpame Enelis si en ocasiones no te hice sentir orgullo y gracias de verdad por todo.

Al tribunal y oponente por la paciencia mostrada en todo momento.

Al fenómeno de mis tías maternas, todas son fuente de alegría y ternura.

A mis compañeros de cuarto Carlitos y Riguito, ya son como hermanos.

A mi amiga Lisaidi y a mis buenos amigos: Hendrik que aunque no se lo menciono mucho le tengo mucha admiración y se que saldrá adelante, a Eduardo, Frank, Jose, Alex mi hermano gracias por tu tiempo, a Gretter, Pedrito, Fredy, Melissa, Orlandito, el pipi, Dany, Alexander, gracias de verdad a todos.

A la real familia: El Niño mi colega, Hairito, Orlandito el hobbie, Eliades y Andito, los quiero.

Resumen

El creciente desarrollo de las tecnologías de la informática y las comunicaciones, unido al volumen cada vez mayor de datos que tienen que procesar las aplicaciones, ha hecho necesaria la implementación y adopción de soluciones informáticas más eficientes para el almacenamiento y gestión de la información. En el caso de las Redes Sociales de Aprendizaje, que se destacan como herramienta de apoyo al proceso formativo en la Educación Superior, son varios los criterios que se han propuesto sobre cómo gestionar los datos y los recursos que manipulan. Particularmente, la integración de Sistemas Gestores de Bases de Datos NoSQL y Sistemas Gestores de Bases de Datos Relacionales, permite garantizar un almacenamiento y acceso eficiente a los datos de la Red, aprovechando las potencialidades de cada uno. En tal sentido, el presente trabajo pretende desarrollar un modelo de datos a partir de la integración de bases de datos NoSQL y relacional para una Red Social de Aprendizaje de la Universidad de Ciencias Informáticas.

Palabras claves: bases de datos relacionales, bases de datos NoSQL, integración de datos, red social de aprendizaje.

Abstract

The constant growth of IT, united to the massive data volume that modern applications need to process, have made a necessity the implementation and adoption of more efficient informatics solutions for the storage and data management. In the case of Social Networks focused in Learning, that highlight as a helpful support tool for the educational process in the Superior Education, the criteria that haven proposed are several, mainly focused in how to manage data and the resources that they manipulate. Particularly, the integration among NoSQL database systems and Relational Database Systems, allow to ensure an efficient data management and data access in the network, taking advantage of the potential of each one of them. In such matter, the present work pretends to develop a data model from the integration of NoSQL databases and RDBMS for a Social Network focused in Learning for the University of Information Sciences.

Keywords: *data integration, NoSQL data bases, relational data bases, social learning network.*

Índice de contenidos

Índice de figuras

Índice de tablas

Introducción

En los últimos años, las Tecnologías de la Informática y las Comunicaciones (TIC) han contribuido a minimizar las distancias entre los seres humanos a través del perfeccionamiento de la comunicación. Hecho que ha sido apoyado por la creación de diferentes plataformas y tecnologías que han permitido la proliferación de los denominados sitios de redes sociales o Webs de redes sociales. Su trascendencia puede apreciarse en estadísticas que reflejan que su uso ha sobrepasado, con creces, al del correo electrónico, y que se han posicionado en la Web por encima de las búsquedas de páginas de interés general y de las aplicaciones de software (1).

Tales sitios de intercambio social son espacios ampliamente conocidos, principalmente por los jóvenes, pero también por un grupo de nuevos usuarios que, progresivamente, están incorporando como una de sus prácticas cotidianas la interacción a través de redes sociales en línea. Las implicaciones de este fenómeno en la comunicación, en la sociedad y en la cultura no son en lo absoluto, insubstanciales, ya que las redes sociales virtuales han demostrado su efectividad como medios de organización ciudadana, más allá de los límites geográficos. En general, constituyen verdaderas plataformas que propician el crear y mantener contactos, publicar y compartir recursos de diferentes tipos, crear comunidades o grupos de interés, agregar contenidos multimedia y mostrar nuestra identidad en la medida en que se desee (1).

La inserción de esas tecnologías en la educación proporciona numerosos recursos a los docentes y, a su vez, familiariza a los estudiantes con herramientas que van a utilizar con frecuencia. La Web 2.0 ofrece aplicaciones interactivas y eficaces para la enseñanza y el aprendizaje que fomentan en los alumnos, la creación de habilidades imprescindibles para el uso de los nuevos entornos educativos. El profesor 2.0 es un guía que conduce a los educandos enseñándoles a adquirir capacidades para que se valgan por sí mismos, y consigan aprender en un mundo constantemente cambiante. Se trata de preparar a los estudiantes para utilizar las herramientas que tendrán que manejar a lo largo de su vida. El uso de las plataformas 2.0 no sólo permite la transmisión de un conocimiento concreto de forma rápida y la colaboración entre personas, sino que, además, desarrolla competencias tecnológicas imprescindibles para operar en contextos diversos y complejos.

Además de los conocimientos tecnológicos, hay que tener en cuenta las habilidades y aptitudes que los alumnos pueden adquirir a través de la educación 2.0. La socialización, el trabajo en equipo o la

importancia de compartir, son elementos que no se pueden enseñar directamente, hay que transmitirlos de manera que se aprendan intuitivamente mediante la interacción con los recursos de los que se disponen. Los nuevos servicios y sistemas educativos permiten aprender "haciendo". Los procesos cognitivos evolucionan a través de la transformación y manipulación de la información, desarrollando lo que se conoce como capacidades cognitivas de alto nivel tales como: el razonamiento, la capacidad de síntesis y análisis, o la toma de decisiones, entre otras (2).

En las redes sociales destinadas al sector educativo circula un flujo de información de manera continua mediante el cual se intercambian grandes volúmenes de datos. Los usuarios tienen la capacidad de introducir datos variables según se ajusten a su perfil en la Red, al tiempo que comparten conocimiento, recursos como fotos, videos y otros con sus similares. Por otro lado, el usuario de la generación de Internet espera tiempos mínimos de respuesta ante cualquier solicitud a los sistemas con los que interactúa, máxime si se trata de una red social para el aprendizaje, donde el intercambio de conocimiento y recursos ocurren de manera simultánea y cada cual se apropia de ellos en el instante que lo necesita, de no ser así perderían interés en la red y no la usarían. La inmediatez en la difusión del conocimiento juega un papel sustancial en la Red, considerando que tiempos mínimos de respuesta para los usuarios conectados, favorece la adquisición del conocimiento que se ha generado en ese preciso instante de tiempo, pues permiten administrar mejor la información y el tiempo.

Debido al bajo aprovechamiento de las estructuras físicas y virtuales para la búsqueda de información y adquisición del conocimiento, así como la ausencia de un carácter colaborativo para la actualización de los contenidos por materia que favorezca la inter e intradisciplinariedad se desea implantar una red social de aprendizaje en la Universidad de Ciencias Informáticas (UCI).

Para llevar a cabo esta empresa es necesario tener claro los diferentes componentes que la conforman y la aplicabilidad de cada uno según los procesos de la universidad. Uno de los componentes más importantes es la capa de datos. La ausencia de modelo de datos traería consigo la inexistencia de información persistente, lo que provocaría que la red perdiera sentido. Dicho modelo debe ser elaborado de la manera más coherente de tal manera que represente los diferentes procesos de la universidad, de no ser así, no cumpliría con el principal objetivo con el que se quiere crear la red.

Para ello es necesaria una correcta definición de la información que será almacenada, pues de no ser así podría traer inconsistencia a la red. Como se trata de un proyecto de gran envergadura, donde se esperan grandes cantidades de tráfico de información concurrente, es necesaria la implementación de una capa de datos que garantice integración, fiabilidad, escalabilidad y además permita reducir los tiempos de respuesta del sistema durante la interacción de los usuarios. Si el modelo no cumple con estos requerimientos, puede traer consigo desinterés por parte de los usuarios a la hora de usar la red y que estos no puedan obtener los beneficios principales por la que fue creada.

Teniendo en cuenta lo anteriormente expuesto se plantea el siguiente **problema de la investigación**: ¿Cómo desarrollar un modelo de datos para una red social para el aprendizaje en la UCI, garantizando la fiabilidad y escalabilidad de los datos y minimizando los tiempos de respuesta del sistema?

La investigación tiene como **objeto de estudio:** los Sistemas de Bases de Datos y como **objetivo general:** el desarrollo de un modelo de datos para una red social de aprendizaje en la UCI, integrando bases de datos NoSQL con una base de datos relacional.

Lo anterior permite definir como **campo de acción:** los modelos de datos para las Redes Sociales de Aprendizaje.

Para resolver el problema científico y darle cumplimiento al objetivo general, se plantearon las siguientes **tareas de investigación**:

1. Análisis de las principales redes sociales de aprendizaje existentes.
2. Estudio y análisis del estado del arte de las Bases de Datos Relacionales y NoSQL.
3. Selección de las tecnologías web y herramientas apropiadas para la implementación del modelo de datos.
4. Modelación de la capa de datos para la Red Social de Aprendizaje en la UCI.
5. Implementación de las funciones para la integración entre el Sistema de Base de Datos Relacional y los sistemas NoSQL.
6. Validación del modelo de datos.

Entre los métodos científicos de investigación utilizados durante el desarrollo de este trabajo, se encuentran:

- **Histórico-lógico:** este método permite establecer la necesaria correspondencia entre los elementos lógicos e históricos, con el fin de analizar la evolución histórica de los conceptos abordados en la investigación. En tal sentido se realizó una minuciosa revisión bibliográfica que permitió analizar la evolución y desarrollo de las bases de datos NoSQL y elaborar una síntesis del estado del arte del tema.
- **Analítico-sintético**: este método permite analizar, comparar y confrontar las diferentes fuentes bibliográficas consultadas, en cuanto a tendencias actuales para el diseño e implementación de modelos dimensionales en base de datos NoSQL. Esto permite identificar los criterios comunes existentes en la literatura y decidir cuáles de ellos usar durante los procesos de diseño e implementación del modelo.
- **Revisión de documentos:** este método permite realizar una exploración exhaustiva de numerosas fuentes de información relacionadas con el tema que se encuentran disponibles en la literatura.

Cada uno de los métodos de investigación descritos anteriormente tuvo una influencia significativa en la construcción teórica, en la fundamentación del problema y la propuesta de solución.

El resto del documento está estructurado de la siguiente manera:

Capítulo 1: Marco teórico de la solución: en este capítulo se realiza un estudio del estado del arte de las redes sociales, haciendo especial énfasis en las de perfil educativo y citando algunos ejemplos. Se abordan las definiciones de bases de datos más reconocidas por la comunidad científica, así como sus clasificaciones en relacional y NoSQL. También se resaltan las ventajas de las bases de datos NoSQL que motivaron su elección para el desarrollo de la solución propuesta. Por otro lado se Describe y justifica la metodología seguida, así como los lenguajes de programación (Python) y de consulta (Cypher) utilizados.

Capítulo 2: Propuesta de la solución: en este capítulo se explica detalladamente el procedimiento que se siguió para la construcción del modelo de integración. Se detalla el Sistema de Actividades Humanas (SAH) propio de la investigación, así como el Sistema de Conocimientos (SCO). Se definen

las entidades (tablas) de la base de datos (BD) con sus respectivos atributos. Se describe el Modelo Entidad-Relación obtenido a partir del SAH y se argumenta la relación entre el problema científico definido y la solución propuesta. Por otro lado se construye un diccionario de datos para tablas más representativas del negocio con el objetivo de facilitar la comprensión semántica de los datos que se almacenarán y se describe el procedimiento que se siguió en la construcción del modelo de integración.

Capítulo 3: Validación de la solución: en este capítulo se valida la propuesta de la solución a través de criterios de especialistas y pruebas de rendimientos con el objetivo de evaluar la integración entre los gestores de bases de datos. Se exponen los requerimientos tanto de software como de hardware necesarios para implantar el modelo de integración. Se muestra un libro de estilo de datos utilizado como estándar durante el desarrollo de la capa de integración.

Capítulo #1. Marco teórico de la solución

1.1. Introducción

El marco teórico es fundamental en el desarrollo de un trabajo de diploma pues provee a la investigación de un basamento teórico imprescindible para la construcción de la solución. Éste proporciona, a cualquier investigador científico, el conjunto de términos, conceptos y categorías de análisis que le permiten enfrentar el problema de interés. Una sólida fundamentación teórica sienta las bases de una solución robusta, agregando seriedad y calidad al trabajo y una merecida confiabilidad por parte de la comunidad científica. En el presente capítulo se analizan un conjunto de enfoques teóricos e investigativos que han sido publicados en el área de gestión de la información de las redes sociales. También se abordan los conceptos fundamentales relacionados con el tema de investigación, tales como: redes sociales, bases de datos y bases de datos NoSQL. Además se argumenta la selección de las herramientas y tecnologías escogidas para el desarrollo del modelo de integración datos.

1.2. Redes sociales

Red, un término que procede del latín *rete*, hace mención a la estructura que tiene un patrón característico. Esta definición permite que el concepto se aplique en diversos ámbitos, como la informática (donde una red es un conjunto de equipos interconectados que comparten información) (3). Social, por su parte, es aquello perteneciente o relativo a la sociedad (el conjunto de individuos que interactúan entre sí para formar una comunidad). Lo social suele implicar un sentido de pertenencia (3).

La noción de red social, por lo tanto, está vinculada a la estructura donde un grupo de personas mantienen algún tipo de vínculo. Dichas relaciones pueden ser amistosas, comerciales, profesionales o de otra índole (3). Según Orihuela, “las redes sociales son los nuevos espacios virtuales en los que nos relacionamos y en los que construimos nuestra identidad” (4).

Citando el material del curso de Formación AOL, *Comunicación en redes sociales*, se entienden estas como “aquellos medios sociales que constan de correo electrónico y protocolos similares para poder enviar mensajes a otros amigos o contactos de la red social; mensajería instantánea para poder

comunicarse con los demás usuarios en tiempo real, de forma similar a los chats; herramientas para poder compartir, bajar o visualizar información, ya sea enlaces, fotos, vídeos, música o información sobre algún perfil; agregación de comentarios personales que permiten a los usuarios expresarse sobre algún tema, indicar el estado de ánimo, contestar a comentarios de otros usuarios, etc" (5).

Para el presente trabajo se adoptará como concepto de Red Social el definido por (6): "Son *webs* que te ofrecen la posibilidad de organizar, en un mismo sitio, tu libreta de contactos personales**,** el correo electrónico, la mensajería instantánea**,** además de proporcionarte espacio para colgar tus fotos, tu música y tus vídeos, herramientas para gestionar tus grupos de discusión, buscar trabajo e incluso para crear tu propio blog". En definitiva**,** todo lo que necesitas para relacionarte a través de tu ordenador. En resumen, una red social es una estructura social formada por personas o entidades conectadas y unidas entre sí por algún tipo de relación o interés común.

Las redes sociales se han convertido, en pocos años, en un fenómeno global. Se expanden como sistemas abiertos en constante construcción de sí mismos, al igual que las personas que las utilizan (2).

1.2.1. Análisis de redes sociales

El análisis de las redes sociales constituye la representación del flujo de una red social a partir de la teoría de grafos, donde los nodos serían los individuos y las aristas las relaciones que les unen. Todo ello conforma un grafo, estructura de datos que permite describir las propiedades de una red social. El término se atribuye a los antropólogos británicos Alfred Radcliffe-Brown y Jhon Barnes y ha sido llevado a cabo por especialidades como la matemática y las ciencias de la computación (2).

Algunas de las redes sociales más conocidas son:

LinkedIn: es una red social para profesionales, donde se puede crear el perfil y colgar el Currículum Vitae (CV), a la vez que contactar con personas con las que has trabajado o del mismo sector y así crear una red de contactos. LinkedIn también posee una red de intercambio de contenidos entre profesionales y un buscador de empleo. Actualmente cuenta con más de 345 millones de usuarios registrados (7).

¿Cómo funciona?

En LinkedIn el usuario puede publicar datos tales como: experiencia laboral, educación, sitio web y recomendaciones. También permite establecer contacto con otros miembros generalmente enfocados en un ámbito profesional específico. Su principal objetivo es fomentar las conexiones profesionales. Estas conexiones pueden estar enfocadas a buscar trabajos así como oportunidades de negocios. Esta red social a menudo es usada como herramienta de los que buscan empleo. El perfil es uno de los elementos básicos de Linkedln, sin embargo los usuarios normalmente no sacan el mayor provecho de él (7).

Facebook: es una red social gratuita que permite conectar a las personas en Internet. Inicialmente se creó para estudiantes de la Universidad de Harvard en Estados Unidos. Era un instrumento que permitía que los alumnos tuvieran contacto entre ellos, intercambiar notas sobre sus cursos e incluso organizar todo tipo de reuniones estudiantiles. En septiembre del 2006 se abrió a toda persona que tuviera un email o correo electrónico, lo que elevó su número de usuarios a 140 millones en ese año (9).

¿Cómo funciona?

Después de crearse una cuenta, el sistema dirá quiénes de los contactos de la libreta de correo electrónico tienen ya un perfil en Facebook. Se selecciona aquellos que se quiere estén en la red de amigos. Facebook les enviará un email para que confirmen la solicitud de amistad. A continuación, Facebook sugiere que se invite al resto de los contactos que aún no tengan perfil, a que se lo creen.

La plataforma de Facebook como la mayor parte de las redes sociales permite publicar y hacer visibles los datos que el usuario quiera. Si no se quiere publicar nada, lo único que aparecerá será el nombre y/o dirección de e-mail. Pero también se puede elegir hacer público la fecha de nacimiento, el lugar de residencia, estado civil, creencias religiosas o políticas, carrera profesional, aficiones, etc.

Después de haber completado los datos personales, Facebook propone **"**elegir una red**"** (por ejemplo, la red "Spain", que actualmente congrega casi un millón de personas) o apuntarse a grupos (como pueden ser, misma empresa, antiguo colegio, o los amantes del vino). Los grupos permiten a los miembros mantenerse al corriente de las novedades, de los eventos, publicar fotos o interactuar con otros usuarios, a través de los foros de discusión.

La gran diferencia entre Facebook y las otras redes sociales, es el dinamismo del sitio: en lugar de un perfil estático que recoja solamente la información que concierne al usuario registrado, se llega a una página de inicio que lo mantiene al corriente de todo lo que han hecho los amigos en Facebook recientemente (si han actualizado su estado, si han agregado fotos, si se han hecho de algún grupo, si han creado un evento...) (10)

1.2.2. Redes sociales educativas

Se pueden definir como grupos de personas relacionadas y conectadas por el interés común en la educación. La alta interrelación entre personas, conocimiento y herramientas que proporcionan espacios comunes para padres, alumnos y profesores donde la imbricación de los agentes educacionales da lugar a una enriquecedora colaboración. Las redes sociales educativas se convierten en entornos de participación y descubrimiento que fomentan la sinergia entre estudiantes y profesores, facilitan el consenso, crean nuevas dinámicas de trabajo fuera y dentro del aula, y permiten el rápido flujo de información, desarrollando así la socialización del conocimiento (2).

1.2.3. Ejemplo de redes sociales educativas

Dentro de las redes sociales más populares en el sector educativo se encuentran:

Redalumnos: es una red social gratuita para mantener en contacto a profesores, alumnos y padres. Es accesible a instituciones públicas y privadas, y permite a los profesores impartir cursos a través de la web. Las opciones de las que dispone son: gestionar plazos de trabajo mediante su calendario, enviar mensajes, importar y exportar ficheros de formato .xls con listados de calificaciones y alumnos, compartir recursos, crear documentos sin paquetes de software adicionales, informar en tiempo real, publicar notas de exámenes, crear y compartir exámenes con autocorrección, mantener el contacto con padres de los alumnos, y crear y compartir tareas (2). (Véase **Figura 1**).

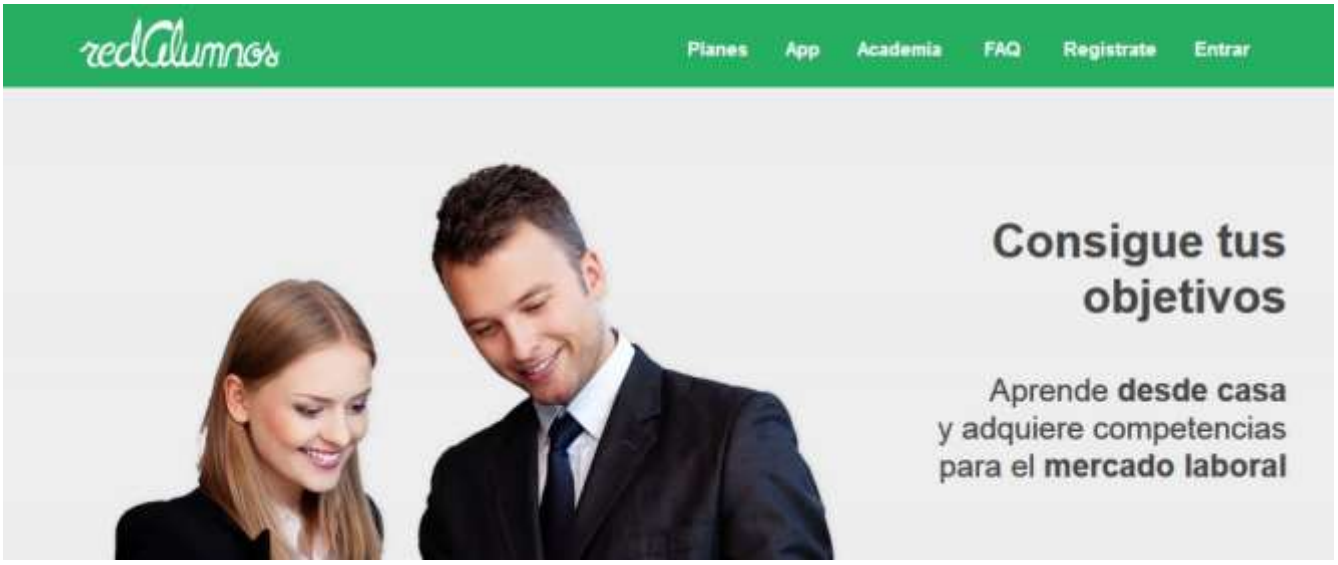


Figura 1. Interfaz de redAlumnos (12).

Educanetwork: esta red social se basa en la consigna de aprender y enseñar en grupo. Pueden crearse grupos que permiten crear cursos; compartir contenidos, como materiales, documentos, vídeos o apuntes; chatear; incluir test multimedia; y organizar eventos. Está disponible en inglés y español, y se presenta como una empresa diferente e innovadora creado por y para personas interesadas en la nueva educación (2).

Diipo: es una red social didáctica y colaborativa para profesores y alumnos del mismo estilo que Edmodo, pero también permite la creación de blogs y proyectos. Ofrece la conexión con otros profesores y permite relacionar las clases con otras que se elijan dentro de su red. Presenta su interfaz de usuario accesible y familiar, con un modelo parecido al de Facebook. Está sólo disponible en inglés (2). (Véase **Figura 2**).

Figura 2. Interfaz de Diipo (14).

Como se puede apreciar, es fundamental en las redes sociales que estas permitan crear perfiles propios a cada persona que se registre y que se establezcan vínculos y relaciones entre ellas según se ajuste al interés de cada cual.

Desde el punto de vista de la implementación, las redes sociales se basan en el principio matemático de la teoría de grafos. Razón por la cual encontrar una tecnología adecuada para almacenar y manipular este tipo de estructura de datos constituye una tarea fundamental durante el desarrollo de la red social de aprendizaje que se pretende implantar en la UCI.

1.3. Bases de Datos

En la actualidad existe una gran polémica alrededor de la gestión de la información digital que se necesita almacenar para su posterior recuperación y análisis por parte de los diseñadores, arquitectos y desarrolladores de aplicaciones informáticas de cualquier tipo. Una base de datos es un conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, una base de datos puede considerarse una colección de datos variables en el tiempo (15).

En (16) se define base de datos como: “Un sistema de mantenimiento de registros basado en computadores cuyo propósito general es registrar y mantener información relacionada con cualquier

cosa que sea significativa para la organización donde el sistema opera". Dicha información normalmente es necesaria para los procesos de toma de decisión inherente a la administración de esa organización.

Es posible considerar que una base de datos es un conjunto de datos almacenados entre los que existen relaciones lógicas y ha sido diseñada para satisfacer los requerimientos de información de una empresa u organización. En una base de datos, además de los datos, también se almacena su descripción (17).

Por un lado están los tradicionales sistemas de gestión de bases de datos relacionales y por otro los prometedores sistemas de bases de datos no relacionales y distribuidos conocidos como NoSQL. Los primeros, dueños de la mayor parte del mercado del almacenamiento de datos, con una robustez innegable y años de explotación en múltiples entornos de gestión de información. Los segundos, emergentes y novedosos, ofrecen sin embargo una nueva forma de pensar en el desarrollo de aplicaciones web orientadas y centradas en el usuario (18).

1.3.1. Bases de Datos Relacionales

Son bases de datos en donde todos los datos visibles al usuario están organizados estrictamente como tablas de valores, y en donde todas las operaciones de la base de datos se realizan sobre estas tablas. Estas bases de datos son presentadas a los usuarios como una colección de relaciones normalizadas de diversos grados que varían con el tiempo (18).

Es una colección de datos interrelacionados que contiene información de una o varias organizaciones relacionadas. La noción de integración tiene que ver con una colección de datos interrelacionados, y la de compartir porque contiene información de una o varias organizaciones (19).

Es una colección de archivos relacionados que permite el manejo de la información de alguna compañía. Cada uno de dichos archivos puede ser visto como una colección de registros y cada registro está compuesto por una colección de campos. Cada uno de los campos de cada registro permite llevar información de algún atributo de una entidad del mundo real (20).

1.4. Sistemas Gestores de Bases de Datos Relacionales (SGBD)

Un Sistema Gestor de Base de Datos es un software diseñado para asistir, mantener y utilizar grandes colecciones de datos. La alternativa al no usar SGBD es almacenar los datos en archivos y utilizar lenguajes de programación para manejarlos, estos últimos son denominados Sistemas de Archivos. El uso de los SGBD tiene muchas ventajas importantes, de allí que su uso es muy extendido en el desarrollo de cualquier tipo de sistema o aplicación automatizada (19).

Un SGBD es un software que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez (15). Su objetivo fundamental consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos, o sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado.

Los programas de aplicación operan sobre los datos almacenados en la base utilizando las facilidades que brindan los SGBD, los que, en la mayoría de los casos, poseen lenguajes especiales de manipulación de la información que facilitan el trabajo de los usuarios (15). Entre los SGBD relacionales más ampliamente utilizados y difundidos en la actualidad se encuentran PostgreSQL, Oracle y Microsoft SQL Server.

1.4.1. PostgreSQL 9.4

Es un sistema de gestión de bases de datos objeto-relacional (ORDBMS). Basado en el proyecto POSTGRES, de la universidad de Berkeley. Es una derivación libre (*OpenSource*) de este proyecto, y utiliza el lenguaje SQL92/SQL99.

Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluido, más tarde en otros sistemas de gestión comerciales. PostgreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional. A pesar de esto, PostgreSQL no es un sistema de gestión de bases de datos puramente orientado a objetos.

Quizás la principal novedad que trae consigo PostgreSQL 9.4 es que soporta de forma nativa JSON, una de las claves del éxito de la popular base de datos documental MongoDB. Las versiones anteriores de PostgreSQL ya lo hacían, aunque almacenaban los documentos JSON en un formato de texto, que requiere más tiempo para su almacenamiento y recuperación.

El formato estructurado de PostgreSQL para guardar JSON elimina la necesidad de estructurar un documento antes de que sea dirigido a la base de datos, lo que procura una velocidad de ingesta de documentos tan rápida como la de MongoDB, al tiempo que mantiene el cumplimiento de ACID (atomicidad, consistencia, aislamiento y durabilidad), requerido para almacenar datos de forma confiable en bases de datos.

Otras características de esta versión de PostgreSQL son (21):

- Implementa una nueva Aplication Programming Interfaces (API) para *Streaming* Replicación, la cual permite que la confección del esquema de replicación sea menos agresivo en el rendimiento de los servidores implicados.
- La función *Materialized Views* para actualizar los reportes sumarios, son un tipo especial de vista que garantizan la persistencia de los resultados de las consultas en forma de tabla. Son muy útiles a la hora de construir tablas resumen de los datos y mantener los mismos con una actualización frecuente. Cabe destacar que este tipo de vistas, hay que refrescarlas periódicamente para que contengan la versión más actualizada de las tuplas.
- El uso de la nueva función *Alter System Set* posibilita a los administradores modificar el archivo de configuración de PostgreSQL directamente desde la línea de comando de SQL.

1.4.2. Oracle 12

El SGBD Oracle utiliza la arquitectura cliente/servidor. Ha incorporado en su sistema el modelo objeto-relacional, pero al mismo tiempo garantiza la compatibilidad con el tradicional modelo relacional de datos. Así ofrece un servidor de bases de datos híbrido. Es uno de los más conocidos y ha alcanzado un alto nivel de madurez y de profesionalidad. Se destaca por su soporte de transacciones, estabilidad y escalabilidad.

Los tipos objeto de Oracle son tipos de datos definidos por el usuario que permiten modelar entidades complejas del mundo real en una estructura que trata cada entidad como una unidad atómica simple en la base de datos.

A partir de la versión 10g del año 2004, se añade a los servidores la capacidad de funcionar según el paradigma de "*Grid*" (o rejilla) y se ofrecen mejoras en la administración e integración de algunos elementos que previamente no funcionaban correctamente juntos.

Características fundamentales de Oracle (22):

- Entorno cliente/servidor.
- Gestión de grandes bases de datos.
- Usuarios concurrentes.
- Alto rendimiento en transacciones.
- Sistemas de alta disponibilidad.
- Gestión de la seguridad.
- Autogestión de la integridad de los datos.
- Opción distribuida.
- Portabilidad.
- Compatibilidad.
- Replicación de entornos.

1.4.3. Microsoft SQL Server 2014

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

Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son *Oracle*, *Sybase ASE*, *PostgreSQL*, *Interbase*, *Firebird* o *MySQL* (23).

Algunas características de Microsoft SQL Server son (24):

- Facilidad de instalación, distribución y utilización.
- Posee una gran variedad de herramientas administrativas y de desarrollo que permiten mejorar la capacidad de instalar, distribuir, administrar y utilizar SQL Server.
- Puede utilizarse el mismo motor de base de datos a través de plataformas que van desde equipos portátiles que ejecutan Microsoft Windows 95 ó 98 hasta grandes servidores con varios procesadores que ejecutan Microsoft Windows NT, Enterprise Edition.
- Almacenamiento de datos.
- Incluye herramientas para extraer y analizar datos resumidos para el proceso analítico en línea (OLAP, Online Analitycal Processing). SQL Server incluye también herramientas para diseñar gráficamente las bases de datos y analizar los datos mediante preguntas en lenguaje normal.
- SQL Server se integra con el correo electrónico, internet y Windows, permitiendo una comunicación local.

1.4.4. SGBD seleccionado

Como SGBD se seleccionó PostgreSQL en su versión 9.4. El principal motivo de esta elección fue la capacidad que ofrece de tratar estructuras de datos externas desde el mismo gestor, lo que es posible gracias a sus funciones *Foreign Data Wrappers.* Por otro lado se valoró también el hecho de ser una herramienta de código abierto y multiplataforma, además de su elevada tendencia de uso en las aplicaciones web actuales que gestionan grandes volúmenes de información.

1.4.5. SGBD NoSQL

Los SGBD NoSQL difieren del modelo clásico del Sistemas Gestores de Bases de Datos Relacionales (RDBMS) en aspectos importantes como la escalabilidad y la flexibilización en el acceso a los datos. El término fue acuñado en 1998 por Carlo Strozzi y resucitado en 2009 por Eric Evans. El propio Evans sugiere mejor referirse a esta familia de BD de nueva generación como “Big Data”. Los datos almacenados no requieren estructuras fijas como tablas, normalmente no soportan operaciones

JOIN, ni garantizan completamente ACID, y habitualmente escalan bien horizontalmente. Los sistemas NoSQL se denominan a veces "no sólo SQL" para subrayar el hecho de que también pueden soportar lenguajes de consulta de tipo SQL (25).

1.4.6. Por qué NoSQL?

Son varias las razones que sustentan el surgimiento y desarrollo de los sistemas NoSQL como mecanismo de apoyo a los RDBMS. Entre estas cuentan:

Tendencia 1: Tamaño: se han creado más datos en los últimos dos años que en todos los años anteriores (1018 Exabytes por año).

Tendencia 2: Conectividad: los datos están cada vez más entrelazados y conectados: de documentos de texto a Wikis a *Resource Description Framework (*RDF) a grafos gigantes conectados.

Tendencia 3: Datos semiestructurados: los datos son cada vez más desestructurados.

Tendencia 4:Arquitectura: el software orientado a servicios actual donde cada servicio tiene su *back-end* (25).

1.4.7. Diferencia entre RDBMS y NoSQL

La principal diferencia radica en cómo guardan los datos (por ejemplo, almacenamiento de un recibo): En un RDBMS se tendría que partir la información en diferentes tablas y luego usar un lenguaje de programación en la parte servidora para transformar estos datos en objetos de la vida real. En NoSQL, simplemente se guarda el recibo: NoSQL es libre de *schemas*[1], no se diseña sus tablas y su estructura por adelantado.

[1] El propósito del estándar XML Schemas es definir la estructura de los documentos XML que estén asignados a tal esquema y los tipos de datos válidos para cada elemento y atributo. En este sentido las posibilidades de control sobre la estructura y los tipos de datos son muy amplias (44).

Las bases de datos relacionales tradicionales permiten definir la estructura de un esquema que demanda reglas rígidas y garantizan ACID. Las aplicaciones web modernas presentan desafíos muy distintos a las que presentan los sistemas empresariales tradicionales (ej. sistemas bancarios): datos a escala web, alta frecuencia de lecturas y escrituras, y cambios de esquema de datos frecuentes. Las aplicaciones sociales (no bancarias) no necesitan el mismo nivel de ACID (25).

1.4.8. Ventajas de Bases de Datos NoSQL

Hay algunas necesidades concretas que las bases de datos SQL no cubren suficientemente y por lo tanto se han desarrollado soluciones NoSQL para poder satisfacerlas. Muchas veces las bases de datos no relacionales no cubren todas las restricciones impuestas por ACID o el esquema de datos que hay que pre-definir para cumplir el estándar SQL y de esta manera pueden cometer otros requisitos que las bases de datos relacionales no desempeñan. Algunos ejemplos son:

Escalabilidad Horizontal

Las bases de datos relacionales son extremadamente difíciles de escalar a más de una máquina o muy costosa la solución de escalabilidad. Aunque típicamente están diseñadas para aprovechar bien el incremento de recursos de una misma máquina, escalar la base de datos distribuyéndola en varias máquinas es un proceso complejo y en muchos casos sin solución en tiempo razonablemente corto.

Si la base de datos está compuesta por un único nodo de datos y lo que se busca es escalar horizontalmente para aumentar el rendimiento de la misma, entonces la técnica utilizada es la replicación. Esto consiste en tener réplicas (llamadas esclavos) de la base de datos principal (llamada maestro) donde las escrituras se deben hacer todas en el equipo maestro y las lecturas se pueden distribuir entre los esclavos.

Esta solución para escalabilidad tiene ciertos problemas. No permite escalar las escrituras, todas las escrituras se deben hacer en la misma máquina, y por tanto están limitadas a las escrituras que pueda hacer una máquina. Otro problema es que la replicación no es en tiempo real. Esto invalida la C de ACID, ya que la base de datos deja de estar en un estado consistente, debido a que si se escribe un registro nuevo y se consulta en una de las réplicas, es posible que todavía no se haya replicado y por tanto ya se ha perdido la consistencia de los datos.

En caso de querer además escalar las escrituras, se tiene que utilizar una técnica llamada "*sharding"* o particionamiento horizontal. Utilizando esta técnica, la base de datos se divide en varias partes donde cada porción se guarda en una máquina distinta.

De esta manera dependiendo de la tabla, o el fragmento de tabla al que se quiera escribir, se escribe o se lee de una máquina u otra.

Esta solución incrementa exponencialmente la complejidad del sistema, y al mismo tiempo, hace que se pierda en la mayoría de las veces la A de ACID, debido a que las transacciones dejan de ser atómicas por el hecho de que una consulta escriba en varias particiones simultáneamente.

Como consecuencia de la complejidad de escalar horizontalmente, y la capacidad finita de la escalabilidad vertical, es fácil entender por qué se han desarrollado proyectos que intentan crear una base de datos horizontalmente escalable. Ya que igualmente hay que renunciar a algunas de las restricciones ACID para escalar bases de datos relacionales, se entiende que crear un nuevo sistema de almacenamiento de datos pensado desde un principio para escalar horizontalmente pueda sacrificar desde el inicio algunas de las propiedades ACID.

Éste es el caso de las soluciones NoSQL. Diseñadas para escalabilidad horizontal masiva, renuncian a algunas de las ventajas de bases de datos tradicionales. Sin embargo, estas ventajas igualmente se tendrían que renunciar si se intenta escalar la misma base de datos relacional a los niveles requeridos, por lo que no se considera gran pérdida.

Al ser diseñadas con este propósito específico, sabiendo desde un principio que sacrificarían las ventajas de una solución ACID no tienen la desventaja que acarrean los sistemas diseñados para cumplirlo, y que luego intentan escalarse más allá de lo que fueron diseñados para hacer (26).

Sin esquema de datos

La mayoría de las aplicaciones evolucionan con el tiempo. Las distintas necesidades de negocio imponen el almacenamiento de datos diversos, o que se necesite guardar datos adicionales. Los cambios en los esquemas de bases de datos relacionales son procesos extremadamente complicados.

En una base de datos tradicional, añadir una columna requiere añadir ese campo a todas las filas existentes, que fácilmente pueden ser cientos de millones de filas. Este proceso puede dejar inoperativa a la base de datos durante un tiempo considerable, y en algunas bases de datos hasta necesita hacerse cuando la base de datos no brinde servicios.

Hay algunos negocios que pueden permitirse estar inoperativos durante una hora en ciertos horarios, pero hay otros que funcionan las 24 horas del día. Algunos ejemplos son los grandes portales web como Facebook, Twitter, o Google, que no se pueden permitir ningún tiempo de caída.

Las bases de datos NoSQL son *Schema-less*, lo que significa que no tienen un esquema de datos pre-definido. Cada registro de la tabla puede tener campos distintos, la base de datos no limita ni controla cuantos campos ni de qué tipo se pueden guardar en cada registro.

Por ejemplo en una base de datos no relacional se guardan registros de tipo “persona”. Estos registros tienen los campos “nombre” y “apellido”. Después de varios millones de registros, se puede añadir el campo “teléfono” a cualquier registro, sin necesidad de variar el esquema de la base de datos, ya que este no tiene esquemas (26).

1.4.9. Arquitectura de Bases de Datos NoSQL

A menudo ofrecen solo garantías de consistencia débiles, como por ejemplo consistencia eventual, o transacciones restringidas a elementos de datos simples. Emplean una arquitectura distribuida, donde los datos se guardan de modo redundante en distintos servidores, a menudo usando tablas *hash* distribuidas. Suelen ofrecer estructuras de datos sencillas como arregles asociativos o almacenes de pares clave-valor (25). (Véase **Figura 3**).

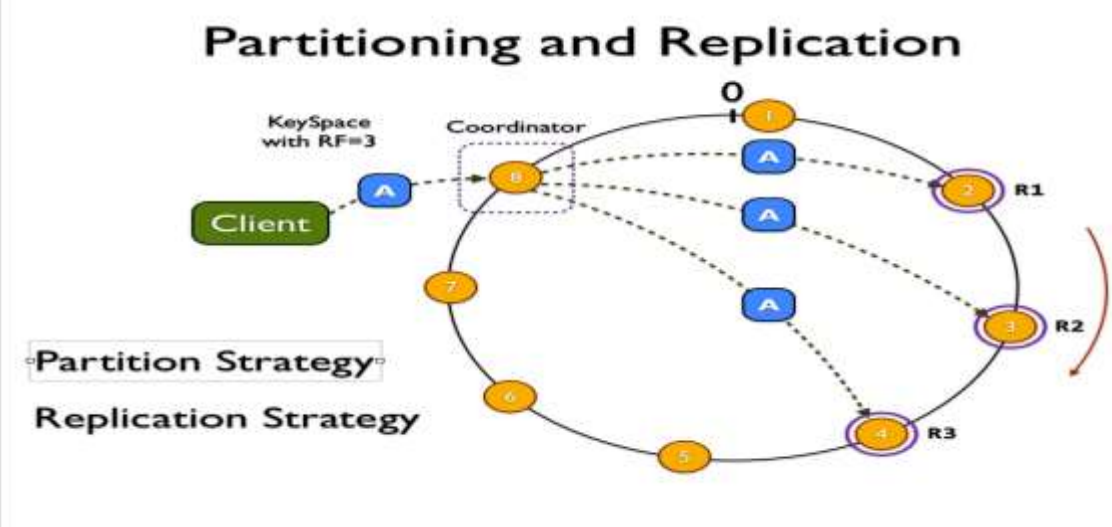


Figura 3. Arquitectura Apache Cassandra (27).

1.4.10. Tipos de Bases de datos NoSQL

Actualmente existen alrededor de 150 sistemas de bases de datos NoSQL. Pero a pesar de que todas se denominan NoSQL, en realidad hay diferentes tipos. Independientemente de que existen aproximaciones diferentes para clasificar las bases de datos NoSQL, en general se considera que existen cuatro tipos diferentes: orientadas a documentos, orientadas a columnas, de clave-valor y orientadas a grafos (28).

Orientadas a documentos

Son aquellas que gestionan datos semiestructurados, es decir documentos. Estos datos son almacenados en algún formato estándar como puede ser XML, JSON o BSON. Se pueden utilizar en gran cantidad de proyectos, incluyendo muchos que tradicionalmente funcionarían sobre bases de datos relacionales.

Dentro de esta categoría se incluyen:

MongoDB 3.0: probablemente la base de datos NoSQL más famosa del momento. En octubre del año pasado, MongoDB conseguía 150 millones de dólares en financiación, convirtiéndose en una da

las *startups* más prometedoras. Algunas compañías que actualmente utilizan MongoDB son *Foursquare* o *eBay*.

CouchDB 1.2: es la base de datos orientada a documentos de Apache. Una de sus interesantes características es que los datos son accesibles a través de una API tipo REST[2]. Este sistema es utilizado por compañías como *Credit Suisse* y la BBC.

Orientadas a columnas

Este tipo de bases de datos están pensadas para realizar consultas y agregaciones sobre grandes cantidades de datos. Funcionan de forma parecida a las bases de datos relacionales, pero almacenando columnas de datos en lugar de registros.

Dentro de esta categoría se incluyen:

Apache Cassandra 2.1.5: incluida en esta sección, aunque en realidad sigue un modelo híbrido entre orientada a columnas y clave-valor. Es usada por Netflix, eBay, Facebook, Ooyala, Zonar Systems, Instagram y otras.

HBase 1.0: escrita en Java y mantenida por el Proyecto Hadoop de Apache, se utiliza para procesar grandes cantidades de datos. Usa *Hadoop Distributed File System* (HDFS) como su sistema de almacenamiento de archivos. La usan Tumblr, Facebook, Yahoo! y otras.

De clave valor

---

[2] REST (40), *Representational State Transfer*, es un tipo de arquitectura de desarrollo web que se apoya totalmente en el estándar HTTP.

REST nos permite crear servicios y aplicaciones que pueden ser usadas por cualquier dispositivo o cliente que entienda HTTP, por lo que es increíblemente más simple y convencional que otras alternativas que se han usado en los últimos diez años como SOAP y XML-RPC.

REST se definió en el 2000 por Roy Fielding, coautor principal también de la especificación HTTP. Se podría considerar REST como un *framework* para construir aplicaciones web respetando HTTP.

Por lo tanto REST es el tipo de arquitectura más natural y estándar para crear APIs para servicios orientados a Internet.

Estas son las más sencillas de entender. Simplemente almacenan tuplas que contienen una clave y su valor. Cuándo se quiere recuperar un dato, simplemente se busca por su clave y se recupera el valor.

Dentro de esta categoría se incluyen:

DynamoDB 2.8: desarrollada por Amazon, es una opción de almacenamiento que se puede usar desde los servicios web de Amazon. La utilizan el Washington Post y Scopely.

Redis 2.8: desarrollada en C y de código abierto, es utilizada por Craiglist y StackOverflow (a modo de caché).

Orientadas a grafos

Basadas en la teoría de grafos, utilizan nodos y aristas para representar los datos almacenados. Son muy útiles para guardar información en modelos con muchas relaciones, como redes y conexiones sociales.

Dentro de esta categoría se incluyen:

InfiniteGraph 2.0: escrita en Java y C++ por la compañía Objectivity. Tiene dos modelos de licenciamiento: uno gratuito y otro de pago.

Neo4j 2.0.1: base de datos de código abierto, escrita en Java por la compañía Neo Technology. Utilizada por compañías como HP, Infojobs o Cisco.

1.5. Soluciones NoSQL adoptadas

Redis 2.8: un servidor de caché de memoria que conoce la estructura de los datos que alberga o dicho de otra forma, "*una caché de datos con estructura*". Su nombre es un acrónimo de Servidor de Diccionario Remoto (29). Es una solución de código abierto de almacenamiento de datos NoSQL basado en una estructura de Llave-Valor (*key-value*).

Además de actuar como servidor de estructuras de datos, Redis aporta otras funcionalidades muy interesantes (29):

- Es capaz de manejar altos niveles de concurrencia, por defecto está establecido en 10000, pero puede ser fácilmente cambiado en la configuración. En este caso, la cantidad real de clientes que puede atender simultáneamente viene dada por la cantidad de descriptores de archivo que pueda manejar el sistema.
- Expiración de claves basada en tiempo.
- Sistema de Publicación y Subscripción a colas de mensajes.
- Operaciones Atómicas y Transacciones.
- Persistencia periódica de la memoria a disco, para recuperación ante caídas.
- Replicación Maestro-Esclavo, para en el futuro implementar un *Clúster.*
- Protocolo abierto, y uso desde decenas de lenguajes de programación.

Actualmente Redis soporta cinco tipos de estructuras de datos (29):

- Cadenas de caracteres (*Strings*).
- Listas (*Lists*).
- Conjuntos (*Sets*).
- Conjuntos Ordenados (*Sorted Sets*).
- Tablas Hash (*Hashes*).

Redis en su versión 2.8 fue seleccionada en la presente investigación para tratar las tablas que más lectura presentan, con el objetivo de aprovechar su rapidez para poner a disposición de los usuarios los datos más frecuentemente usados.

Neo4j 2.1: base de Datos de Grafos. Una colección de nodos (entidades) y aristas (relaciones) que conectan pares de nodos. Se asocian propiedades (*key-value pairs*) sobre nodos y propiedades. Las relaciones conectan dos nodos y tanto nodos como relaciones pueden alojar un número arbitrario de pares clave/valor. Puede ser considerado como un *key-value store*, con soporte completo para relaciones. Sin esquemas, permite el diseño de modelos de datos de abajo a arriba (25).

Neo4j tiene las siguientes características (30):

- No hay esquema.
- Transacciones ACID.
- Puede contener billones de nodos y relaciones.
- Rápido recorriendo relaciones, este tipo de consultas se conoce como transversales.
- Lenguaje de consultas propio, *Cypher.*
- Alta disponibilidad, instalación en diferentes maquinas con balanceador de carga.
- Multilenguaje, proporciona una *API Rest* pudiendo utilizarse desde cualquier lenguaje.

Neo4j en su versión 2.1 se utilizará para modelar el grafo con las respectivas entidades y relaciones correspondientes al negocio del presente proyecto. Esta elección se basó en que es un tipo de gestor de bases de datos NoSQL de código abierto robusto que posee una interfaz amigable para tratar las estructuras de datos.

1.6. Lenguaje de programación Python

Es un lenguaje de programación de propósito general muy fácil de aprender, con una sintaxis característica que hace que los programas escritos en él sean muy legibles. Es de código abierto y ampliamente utilizado por empresas como Google o la NASA. Fue creado en 1991 por Guido van Rossum, un programador holandés (31).

Técnicamente, el lenguaje Python tiene las siguientes características (31):

- Es dinámico, en lugar de estático. No es necesario declarar explícitamente el tipo de las variables: es el intérprete quien hace automáticamente la asignación, lo que no ocurre en lenguajes como Fortran o C.
- Tiene tipado fuerte, en lugar de débil. Una vez que una variable tiene un tipo asignado, no se producirán conversiones implícitas, sino que se lanzará un error, al contrario de lo que sucede en JavaScript o en PHP.
- Soporta diversos paradigmas de programación tales como: la programación por procedimientos, la programación orientada a objetos (POO), la programación funcional, etc.

Python ofrece numerosas ventaja para el mundo científico e ingenieril (31):

- La documentación de Python es abundante, de fácil acceso y está muy bien explicada. Es muy sencillo de aprender de forma autodidacta.
- Existen multitud de bibliotecas implementadas para Python relacionadas con la física o la ingeniería.
- Es libre y se puede instalar en Linux, Windows y Mac.

El lenguaje Python se utilizará en el presente trabajo para crear desde PostgreSQL los distintos tipos de funciones que actúan sobre el grafo diseñado. Con el propósito de embeber código Python dentro del gestor se utilizará la extensión de PL/Python; y para establecer la conexión entre el gestor PostgreSQL y Neo4j se utilizó el módulo py2neo.

1.7. Lenguaje de consulta Cypher

Lenguaje de consulta declarativo que implementa Neo4j que toma ciertos elementos de SQL y SPARQL[3]. A diferencia de lenguajes imperativos y que describen un recorrido a través de un grafo para acceder a un conjunto de nodos, Cypher permite definir patrones del grafo y condiciones de manera declarativa. Dicho de otro modo, Cypher se focaliza en describir *qué* es lo que se quiere obtener en vez de *cómo*, dejando un espacio para que la optimización de la consulta sea realizada por el motor de la base de datos (32).

Es un lenguaje propio de Neo4j para realizar consultas. Es la forma declarativa de realizar consultas, porque luego está la forma nativa que es utilizando la API de java de Neo4j. Cypher es a Neo4j lo que T-SQL a SQL Server. Es bastante intuitivo y fácil de entender. Consultas que resultan largas en SQL, con Cypher quedan más simples y cortas (30).

Para hacer consultas con Cypher hay que tener en cuenta los siguientes aspectos (33):

[3] El Servicio "SPARQL" es un Protocolo Simple y un Lenguaje de Consulta para *Resource Description Framework* que permite obtener también los resultados en formato RDF para poder reutilizarlos en las aplicaciones **(41)**.

- Tipos de consultas: se pueden dividir en consultas de lectura y de escritura. Algunas veces, para escribir sobre la base de datos es necesario hacer lecturas, por lo cual las consultas pueden ser también una combinación de consultas de lectura y escritura.
- Inicio de la consulta: el inicio de una consulta depende de su tipo. En caso de que sea solo escritura, lo único que se puede hacer es utilizar como inicio una cláusula CREATE o CREATE UNIQUE para crear nuevos nodos. En cualquier otro caso (una consulta de lectura o una híbrida) se deberá comenzar con una cláusula START, la cual identificará el punto de partida obtenido del análisis hecho sobre el grafo cuando se aplican los patrones sobre ellos.
- Patrones: son formaciones de caminos sobre el grafo que se pueden identificar de acuerdo a relaciones y nodos específicos. Estos caminos tienen un punto (o varios puntos) de partida (tomados en la aplicación de la cláusula START y tienen límites, los cuales pueden o no ser explícitos (estar definidos por un identificador). Los patrones empiezan con la cláusula MATCH seguida del patrón a aplicar sobre la base de datos.
- Retorno: al realizar una transacción sobre la base de datos, cuando se lee el grafo se debe usar la cláusula RETURN para devolver los valores que se han consultado.
- Separación de una consulta de lectura y una de escritura: con la cláusula WITH se hace una separación de la transacción en una parte de lectura y una de escritura.

El lenguaje Cypher es usado en la presente aplicación para hacer tipos de búsquedas específicos en el grafo.

1.8. Metodología de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos software. Constituyen guías en las que se indican paso a paso todas las actividades a realizar para lograr el producto informático de calidad. Describe además qué personas deben participar en el desarrollo de cada una de las actividades y qué papel deben desempeñar. Además detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla (34).

En el contexto de los sistemas de información, el término metodologías suele generar equívocos a menudo. Es frecuente que se espere de ellas resultados que, en realidad, no pueden dar. En

concreto, se suele esperar de ellas lo mismo que proporcionan, por ejemplo, los algoritmos en matemáticas, es decir, una solución segura a un problema bien planteado (35).

Por desgracia, en el desarrollo de sistemas de información no existe nada parecido a los algoritmos. ¿Para qué sirve entonces una metodología en este contexto? La experiencia indica que una metodología sirve, exactamente, para que el resultado final de un proyecto documental se deba en lo más posible a la planificación consciente y, en lo menos posible, al azar o al método de ensayo y error (35).

En sistemas de información documentales, una metodología debería contemplar, como mínimo, tres grupos de elementos o aparatos conceptuales (35):

- Aparato conceptual
- Aparato instrumental
- Aparato procedimental

El primer aparato, o grupo de elementos conceptuales, tiene la misión de proporcionar a los responsables de desarrollo de sistemas de información unas bases conceptuales mínimas que faciliten su entendimiento de todo el proyecto y que faciliten, así mismo, la comunicación entre los diferentes actores involucrados en el proceso. Por tanto, en el aparato conceptual se definen las entidades básicas que intervienen en el proyecto y se proporcionan puntos de vista estratégicos.

El aparato instrumental es el responsable de proveer los instrumentos de análisis y de diseño, es decir, es aquella parte de la metodología que, precisamente, a veces se ha confundido, incorrectamente, con un algoritmo.

Finalmente, el aparato procedimental establece las fases y los procedimientos básicos, señalando sus objetivos, así como identifica y describe los productos que deben obtenerse de cada fase de análisis, incluido el producto final. Así pues, el proceso de diseño de un sistema de información debe ajustarse siempre al siguiente ciclo de vida que, por otro lado, es universal para todo sistema de información:

- Análisis.

- Diseño.
- Implantación.

Para el desarrollo del presente trabajo se siguió la “Metodología de análisis de sistemas de información y diseño de bases de datos documentales.” propuesta por el Dr. Lluís Codina de la Universidad Pompeu Fabra de Barcelona (35).

La metodología propone la planificación como herramienta de éxito para reducir al mínimo posible los riesgos debidos a la improvisación. Por otro lado, importa señalar que es el resultado básicamente, de la utilización de tres tradiciones científicas y académicas distintas: (35)

- La tradición del análisis de sistemas, proveniente de las ciencias informáticas.
- La tradición de la teoría de sistemas y de la metodología general de resolución de problemas. Concretamente, se ha utilizado teoría general de sistemas adaptada a problemas de información (Baiget, 1986; Curras, 1988) y aportaciones de la SSM (Soft System Metodology), una metodología elaborada principalmente, pero no únicamente, por Checkland (Checkland, 1981; Checkland y Scholes, 1990; Lewis, 1994; Underwood, 1996).
- La tradición, naturalmente, de los métodos y procedimientos de trabajo de las ciencias de la documentación.

Una vez expuestas estas consideraciones de tipo meta-metodológicas, se presentan las fases que la metodología define:

1. **Análisis**
    - ✓ Análisis del sistema de actividades humanas.
    - ✓ Análisis del sistema de conocimiento.
2. **Diseño**
    - ✓ Diseño del modelo conceptual.
    - ✓ Determinación de los procedimientos de tratamiento documental (descripción, análisis e indexación documental, etc.) si es el caso.

3. Implantación

- ✓ Elaboración del presupuesto y del calendario de implantación, en su caso.
- ✓ Selección del soporte informático (software y hardware) de acuerdo con los requerimientos expresados en el modelo conceptual de la base de datos producido en la fase 2a y de acuerdo con los requerimientos expresados en 2b.
- ✓ Instalación, pruebas de rendimiento y reelaboración, en su caso, de los puntos previos de este ciclo de vida.
- ✓ Elaboración del libro de estilo de la base de datos.
- ✓ Carga de datos, formación de usuarios y promoción del producto.

Aunque expresado en fases y enumeradas secuencialmente el proceso parece estrictamente lineal, en realidad, el proceso de diseño también tiene mucho de circular, porque aunque siempre se empieza por la fase de análisis y se sigue con la de diseño, llegados a la fase 2b, por ejemplo, es posible que el diseñador desee considerar de nuevo algunos aspectos de 2a, o que necesite aclarar mejor algunas cuestiones de 1 b, etc., idea que recoge la siguiente figura:

Figura 4. El ciclo de vida de un sistema de información como un proceso circular.

En este sentido, es notorio que la metodología no excluye totalmente el procedimiento del ensayo y error, como ya se advirtió, sino que lo integra como un modo natural de refinar el producto.

1.9. Consideraciones parciales

En este capítulo se realizó una investigación de las redes sociales de los que se obtuvo que el análisis de los datos se realiza a partir de la teoría de grafos, resaltando la necesidad de utilizar una base de datos que permita aplicarla. Además se argumentó la selección tanto de las herramientas usadas para la gestión de las bases de datos como de los lenguajes de programación y consulta a usar. PostgreSQL 9.4 fue elegido para el tratamiento de los datos relacionales, Neo4j para almacenar y manipular los datos en forma de grafos de la Red Social y Redis para guardar y permitir el acceso rápido a los datos que requieren lectura más frecuente. Como lenguaje de programación, se seleccionó Python, específicamente el módulo py2neo para la conexión a Neo4j y Cypher, como lenguaje de consultas sobre grafos. Por último se describió la metodología de desarrollo a seguir durante el análisis del sistema de información y el diseño de la base de datos documental.

Capítulo #2: Análisis y diseño del modelo de datos

2.1 Introducción

En el presente capítulo se abordarán las fases de análisis y diseño del modelo de datos que se pretende implementar. Para ello se analizan los artefactos generados en cada una de las fases de la metodología usada, y se describen minuciosamente las actividades realizadas en ambas.

2.2 Fase análisis

El objetivo de esta fase es explorar y entender las propiedades del sistema de información para el cual se va a implementar el modelo de datos. Con tal propósito se definen detalladamente las características y funcionalidades. En aras de realizar un análisis minucioso, el sistema objeto será dividido en (35):

- Un sistema de actividades humanas (SAH)
- Un sistema de conocimientos (SCO)

2.2.1 Sistema de actividades humanas

El sistema de actividades humana (SAH) representa el conjunto formado por personas y cosas que justifica la existencia del sistema de información. En él los futuros usuarios desarrollarán actividades que dependen del sistema de información (35).

SAH de la investigación

Aprender de forma informal no significa que la docencia tradicional esté “pasada de moda”, ni que sea inútil en el contexto actual. Sin embargo, la sociedad de hoy demanda un uso intensivo de las TICs pero con un enfoque innovador, que no deje espacio al aburrimiento y a la imposición del estudio como un deber, en lugar de la propia necesidad de aprender con un estilo personalizado. No cabe duda que las redes sociales se imponen entonces como la herramienta idónea para crear el contexto apropiado que demanda hoy el proceso de enseñanza-aprendizaje.

Desde el año 2007, en la Facultad 5 de la Universidad de Ciencias Informáticas se desarrollan soluciones de carácter científico docente que utilizan el paradigma de la Web 2.0. Por un lado se implementan aplicaciones con un enfoque social, propicias para compartir información y conocimiento; al tiempo que en el Grupo de Investigación de Visualización y Realidad Virtual se trabajaba en entrenadores, juegos serios, laboratorios virtuales y otros productos que incluyen un alto contenido educativo. Estos resultados permiten corroborar que se ha logrado establecer una sinergia entre los conocimientos informáticos y los de corte pedagógico y didáctico. Hecho que ha sido posible gracias a la incorporación de profesores con perfiles diversos que se complementan entre sí, formando un equipo multidisciplinar (36).

Como consecuencia de esa colaboración ha surgido la idea de crear una red social de aprendizaje para la universidad, que teniendo como actores principales a estudiantes y profesores, sea capaz de satisfacer las demandas educativas de cada una de sus facultades. De dichas personas se deben conocer sus nombre y apellidos, sexo, correo, fecha de nacimiento y categoría. De los estudiantes específicamente se debe conocer su grupo, grado y proyecto en caso de pertenecer a alguno y de los profesores, su categoría científica y docente, así como la asignatura que imparten. Las personas podrán interactuar mediante el uso de blogs, desafíos y recomendaciones.

Los blogs deben contener comentarios e imágenes, y tener asociados una fecha y texto. En el caso de las imágenes deben poseer nombre, descripción y dirección física.

En el caso de los desafíos se debe registrar la persona a la que está dirigido, el estado, la nota máxima, la fecha en que fue creado, una etiqueta y su descripción. El desafío tendrá una respuesta formada por el texto con la información, la persona que lo respondió y la URL[4] del archivo. Esta respuesta va a poder ser evaluada por la persona que asignó el desafío, por quien la contestó y por la comunidad en general.

[4] **URL** es una sigla del idioma inglés correspondiente a Uniform Resource Locator (Localizador Uniforme de Recursos). Representa la secuencia de caracteres que sigue un estándar y que permite denominar recursos dentro del entorno de Internet para que puedan ser localizados (43).

Las recomendaciones se llevan a cabo entre las personas. De ellas se registra la prioridad, la persona a la que fue dirigida y la información a mostrar. Estas recomendaciones recibirán notificaciones, que poseerán un título, tipo y recurso.

Las personas además podrán tener una galería de imágenes, imágenes sueltas y un currículum. La galería está caracterizada por tener una o muchas imágenes, que estas a su vez deben tener un nombre, una descripción y una dirección física. El currículum tiene un nombre, archivo, debe reflejar las habilidades computacionales de la persona, si ha certificado algún rol, si ha sido referenciada, los estudios que ha realizado la persona (con su inicio, fin, título de graduado y la institución en la que se graduó), así como la experiencia laboral, en la que se debe informar el período en que estuvo desarrollando un rol, la ocupación, el lugar en el que trabajó y la dirección de dicho centro (Véase **Figura 8**).

2.2.2 Sistema de Conocimiento

El sistema de conocimiento (SCO) está formado por la clase de documentos o por los tipos de entidad sobre los cuales el sistema de información debe mantener alguna clase de registros (35).

El SCO del presente proyecto se han definido las siguientes entidades:

Referentes al funcionamiento general de las redes sociales:

- **Expcompu**: es una tabla para almacenar clasificadores relacionados con la experiencia o experticia que tienen los usuarios en determinadas herramientas de la computación.
- **Persona**: es todo usuario que interactúa con la Red. De esta se deriva por herencia (profesor y estudiante).
- **Profesor**: son los profesores que forman parte de la red.
- **Estudiante**: estudiantes que son parte de la red.
- **Currículo**: es el currículo de todas las personas, tanto estudiantes, profesores como de cualquier otro tipo de persona. Incluye todo aquello que has hecho, estudiado y se quiera mostrar, habilidades, etc... De ahí se derivan algunas como son la Experiencia Laboral que se

recoge en "exptrabajo" y Estudios "estudios" que almacenan en partes distintas dentro el currículo.

- **Exptrabajo**: experiencia laboral tanto de estudiantes, profesores o de cualquier otro tipo de persona que esté registrado en la red.
- **Estudios**: recoge las especificaciones de estudios cursados por los distintos tipos de usuarios de la red.
- **Galería**: es la organización que hacen las personas de sus imágenes. Funciona como una estructura de carpetas donde se almacenan las imágenes que comparte y postea cada persona.
- **Imagen**: son los datos de las imágenes en sí.
- **Blog**: es aquel conjunto de textos y medias (imágenes, videos, etc....). Lo mismo para el nanoblog. En un blog se almacenan artículos, comentarios, imágenes y cada elemento tiene su tabla. Una persona puede comentar un blog directamente o puede comentar un artículo que se encuentra en ese blog.
- **Nanoblog**: conjunto de comentarios cortos asociados a cada persona.
- **Artículo**: almacena la información completa de un artículo del blog.
- **Comentario**: almacena la información relacionada con un artículo, blog o nanoblog, siempre teniendo en cuenta que las personas son las que realizan los comentarios.
- **Recomendación**: esta tabla se refiere a las recomendaciones que se ofrecen a las personas de nuevos conocimientos, retos, tareas, etc...
- **Notificación**: almacena la notificación que le llega a la persona sobre algún nuevo artículo de los blog que ella sigue, una persona que se le recomiende para que la agregue como colega, etc...
- **Tarea**: está relacionada con la organización del tiempo: calendario compartido, etc...
- **Evento**: es un evento en el calendario, que emite o no una notificación.

Entidades de la parte "educativa":

- **Desafío**: es todo lo relacionado a las tareas dentro de la red social, se refiere tanto a las tareas que se pone una persona para sí mismo, a como las tareas que son orientadas por un profesor u otro estudiante. Representan actividades que persiguen el cumplimiento de una meta.
- **Respuestas**: almacenan las respuestas a los desafíos.
- **Evaluación**: la evaluación tanto de los profesores o autores de desafíos, como la autoevaluación a la respuesta (puede que no esté bien la pregunta o no sea relevante). Es útil para ambos partes, y permite que tanto el que pregunte como los que responden reciban una evaluación.

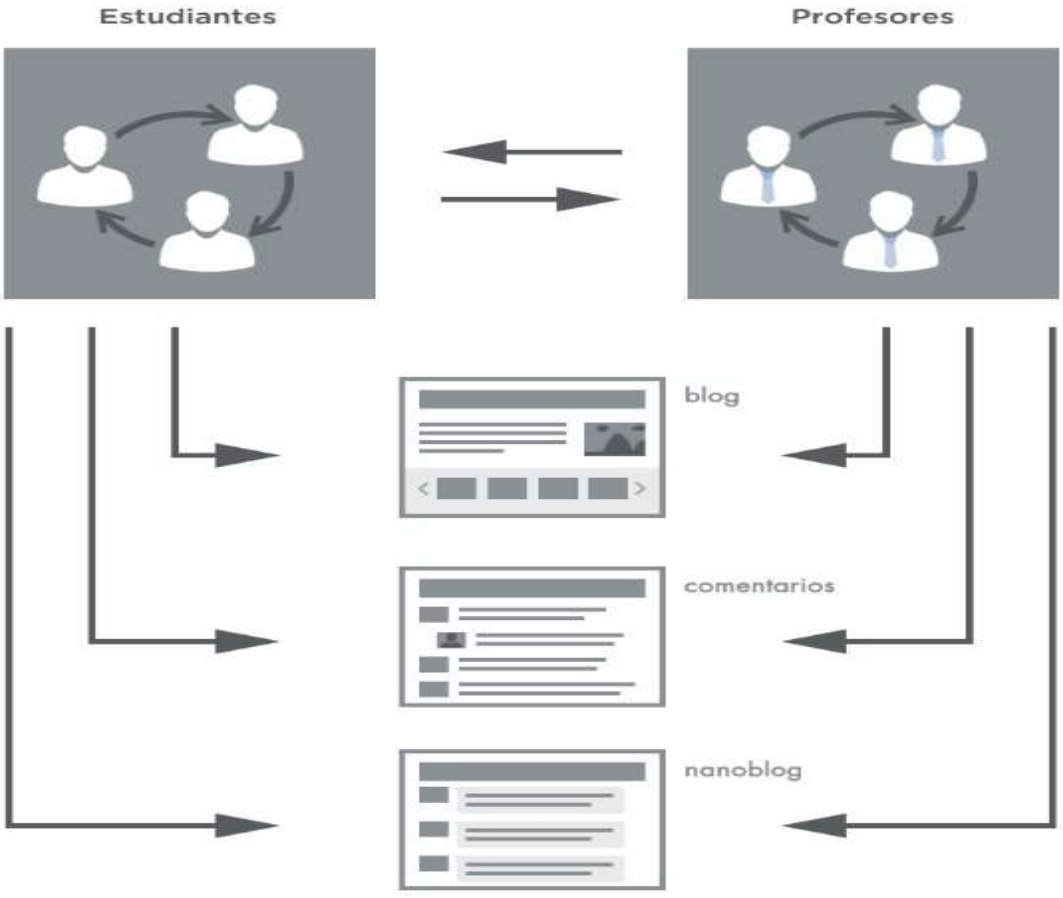


Figura 5. SAH de la investigación.

2.3 Patrones de bases de datos

Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software. En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares. De un patrón se debe tener en cuenta: su nombre, el problema (cuando aplicar un patrón), la solución (descripción abstracta del problema) y las consecuencias (costos y beneficios) (37).

En el modelo desarrollado sigue el patrón Node Edge Directed Graph (Modelo de base de datos de objeto gráfico dirigido). Dicho patrón se enfoca en que tantos los nodos como las aristas del grafo son parte de la información (Véase **Figura 6**).

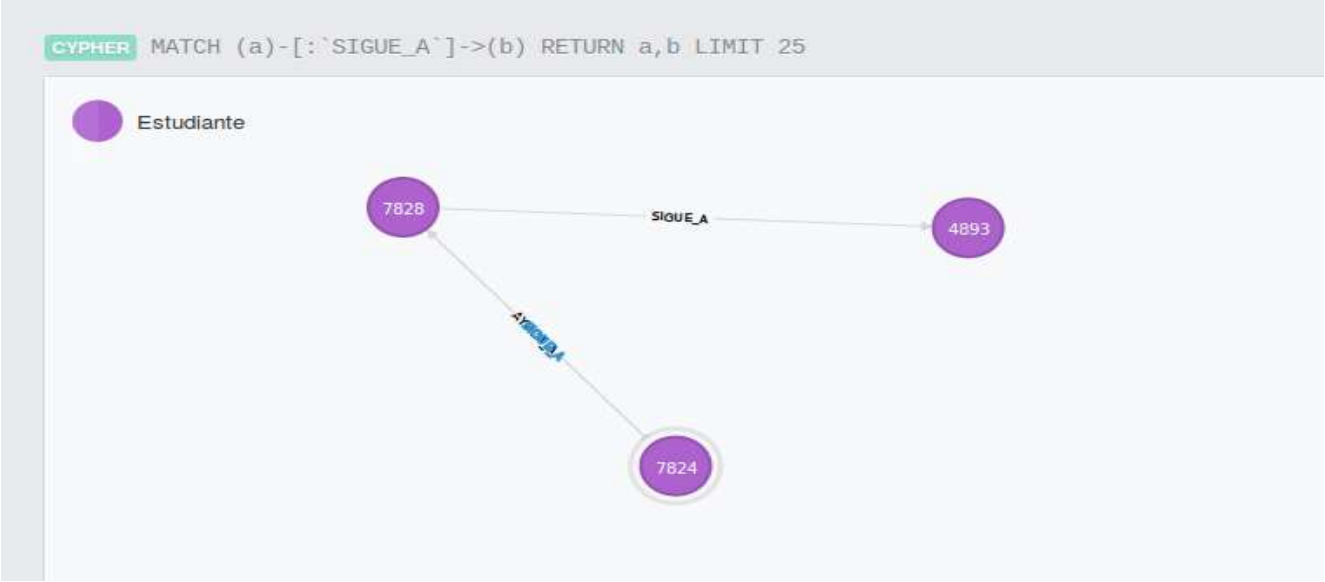


Figura 6. Ejemplo del uso del patrón.

2.4 Entidades de la bases de datos

En bases de datos, una entidad es la representación de un objeto o concepto del mundo real que se guarda en una estructura de almacenamiento. Una entidad se describe en la estructura de la base de datos a través de un modelo de datos (38). Estas entidades fueron seleccionadas a partir del análisis del sistema de relaciones humanas.

A continuación se representan las entidades del proyecto con sus respectivos atributos:

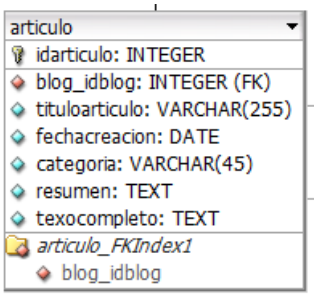


Figura 7. Entidad artículo.

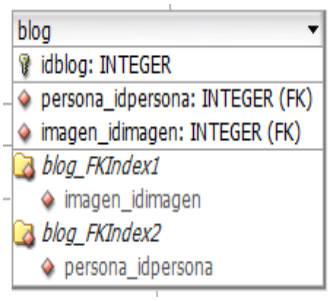


Figura 8. Entidad blog.

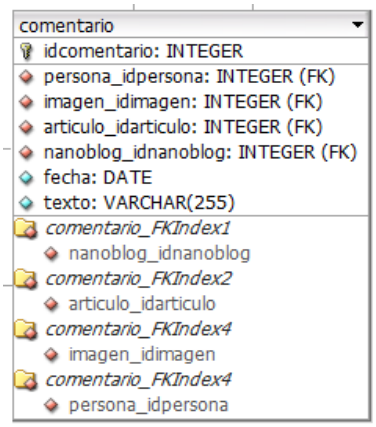


Figura 9. Entidad comentario.

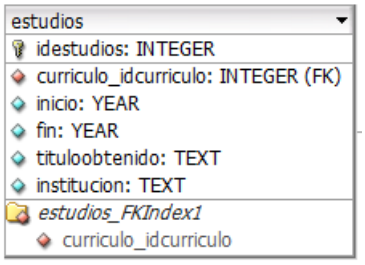


Figura 10. Entidad estudios.

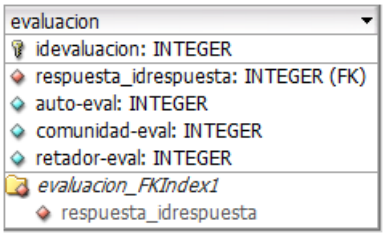


Figura 11. Entidad evaluación.

Figura 12. Entidad eventos.

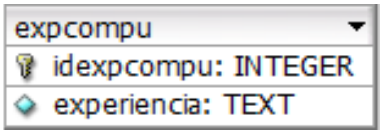


Figura 13. Entidad Expcompu

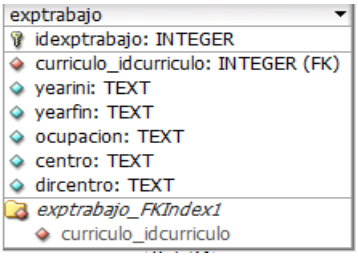


Figura 14. Entidad exptrabajo.

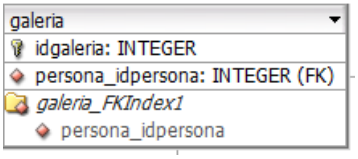


Figura 15. Entidad galeria.

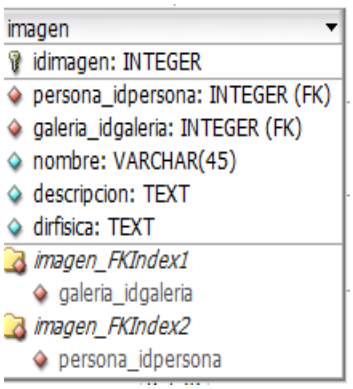


Figura 16. Entidad imagen.

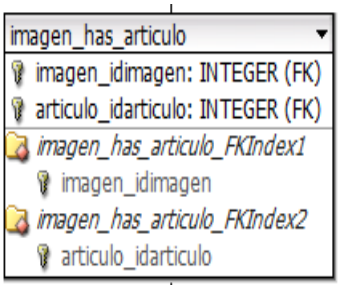


Figura 17. Entidad imagen_has_articulo.

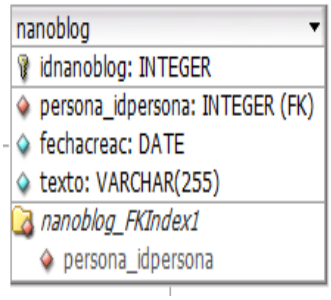


Figura 18. Entidad nanoblog.

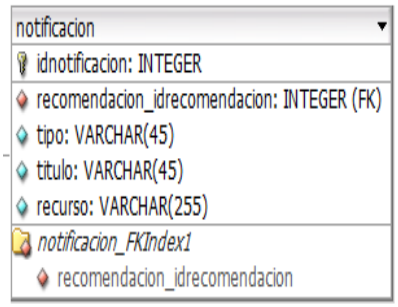


Figura 19. Entidad notificación.

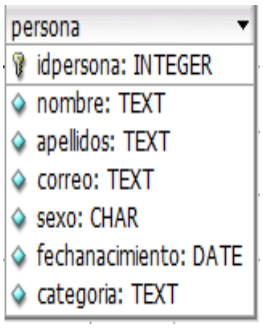


Figura 20. Entidad persona.

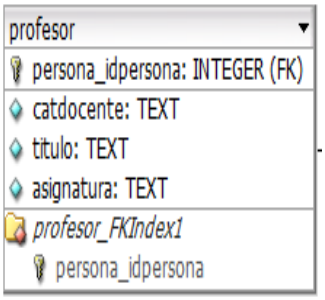


Figura 21. Entidad profesor.

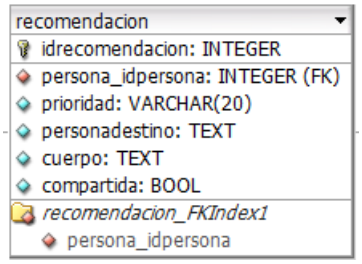


Figura 22. Entidad recomendación.

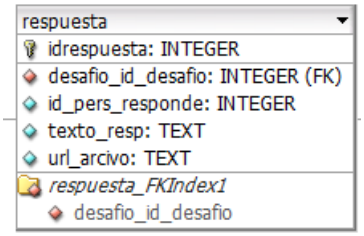


Figura 23. Entidad respuesta.

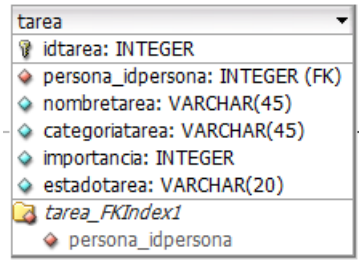


Figura 24. Entidad tarea.

2.5 Modelo Entidad-Relación (MER) de la base de datos

Cuando se utiliza una base de datos para gestionar información, se está plasmando una parte del mundo real en una serie de tablas, registros y campos ubicados en un ordenador; creándose un modelo parcial de la realidad. Antes de crear físicamente estas tablas en el ordenador se debe realizar un modelo de datos. Se suele cometer el error de ir creando nuevas tablas a medida que se van necesitando, haciendo así el modelo de datos y la construcción física de las tablas simultáneamente. El resultado de esto acaba siendo un sistema de información parcheado, con datos dispersos que terminan por no cumplir adecuadamente los requisitos de integridad necesarios (39).

El modelo de datos más extendido es el denominado Entidad-Relación (E/R). En el modelo E/R partiendo de una situación real, se definen entidades y relaciones entre dichas entidades (Para consultar el MER del negocio (39) (Véase **Figura 25**):

- **Entidad**: objeto del mundo real sobre el que se quiere almacenar información. Las entidades están compuestas de atributos que son los datos que definen el objeto. De entre los atributos habrá uno o un conjunto de ellos que no se repite; a este atributo o conjunto de atributos se le llama clave de la entidad. En toda entidad siempre hay al menos una clave, que en el peor de los casos, estará formada por todos los atributos de la tabla. Ya que pueden haber varias claves y se necesita elegir una, se hará atendiendo a las siguientes normas:

    - ✓ Que sea única.
    - ✓ Que se tenga pleno conocimiento de ella.
    - ✓ Que sea mínima, ya que será muy utilizada por el gestor de base de datos.

- **Relación**: asociación entre entidades, sin existencia propia en el mundo real que se está modelando, pero necesaria para reflejar las interacciones existentes entre entidades. Las relaciones pueden ser de tres tipos:

    - ✓ **Relaciones 1-1:** las entidades que intervienen en la relación se asocian una a una.
    - ✓ **Relaciones 1-n:** una ocurrencia de una entidad está asociada con muchas (n) de otra.
    - ✓ **Relaciones n-n:** cada ocurrencia, en cualquiera de las dos entidades de la relación, puede estar asociada con muchas (n) de la otra y viceversa.

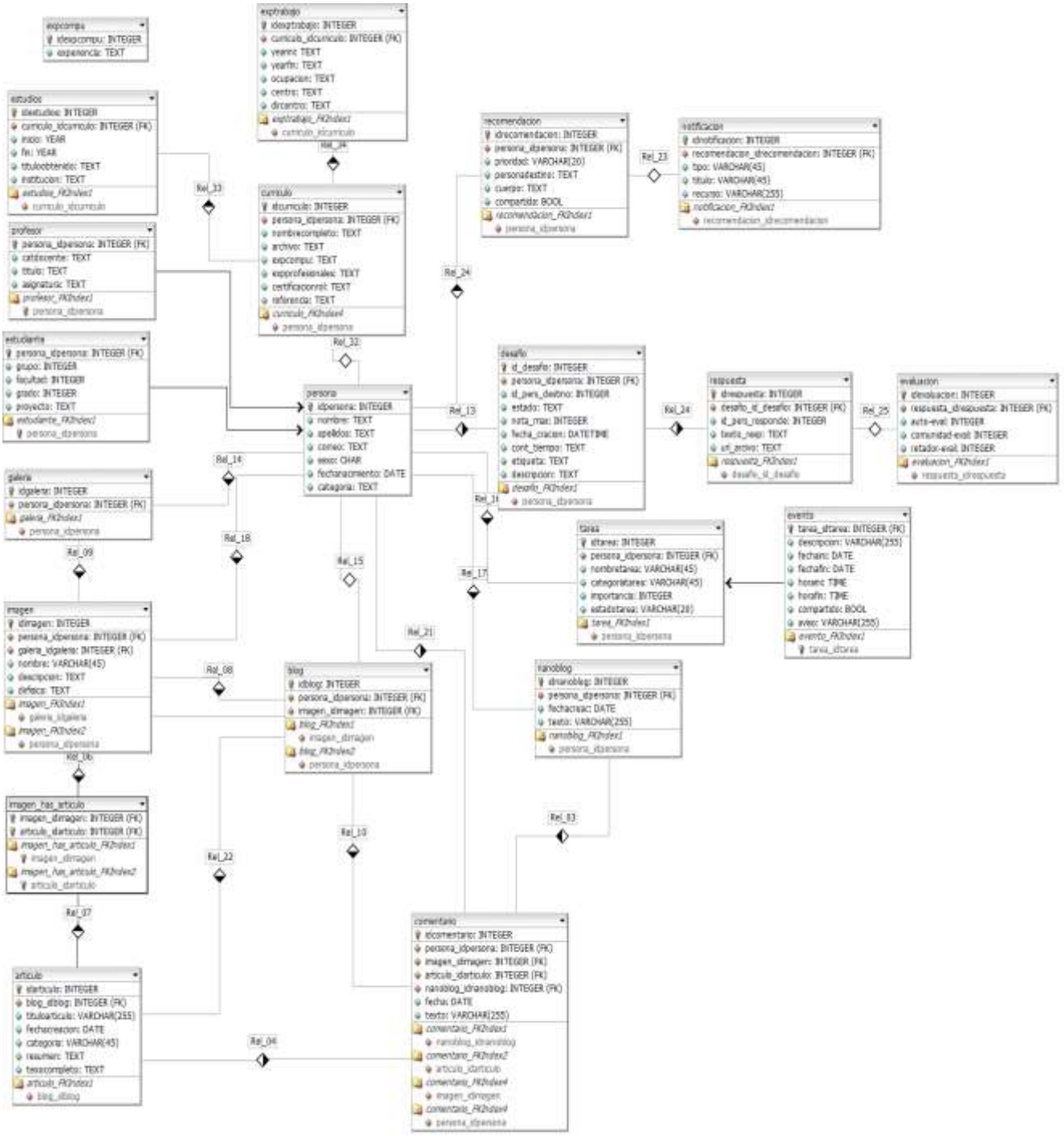


Figura 25. Modelo Entidad Relación.

2.6 Diseño de la base de datos

El propósito de la fase de diseño es obtener un una propuesta de tratamiento documental. Este establece criterios y orientaciones sobre el proceso de descripción y de representación del contenido semántico de los documentos o entidades de los que tratará la base de datos (35).

2.6.1 Modelo Esencial

Consiste básicamente en una descripción textual, puede incluir, si el documentalista lo considera necesario, diagramas o gráficos que faciliten su comprensión. El Modelo Esencial no debe ser muy extenso, sino, que tal como indica su nombre, debe consistir únicamente en una descripción que recoja los aspectos esenciales de la naturaleza y de las actividades del SAH. Además, como una base de datos documental no persigue el modelado de esas actividades, probablemente cinco o seis párrafos deberían ser suficientes para aportar el conocimiento necesario para los objetivos perseguidos (35).

2.6.2 Propósito de la base de datos

El propósito de la base de datos diseñada en el presente trabajo es gestionar el flujo de información que genera la interacción tanto de estudiantes como de profesores de la UCI a través de una red social colaborativa de aprendizaje. Teniendo en cuenta la concepción del negocio, se identifican como usuarios a personas representadas por profesores y estudiantes de la UCI.

2.7 Diccionario de datos

Es una herramienta que ayuda al diseñador de una base de datos a garantizar la calidad, fiabilidad, consistencia y coherencia de la información almacenada en la base de datos; de tal manera que el diccionario de datos influirá notablemente en el rendimiento y la calidad global del sistema de información (35).

El diccionario de datos consiste en la lista detallada de cada uno de los campos que forman los distintos modelos de registro de la base de datos. A cada campo de cada modelo de registro se le aplica una parrilla de análisis que contempla, como mínimo, los siguientes aspectos (35):

- Etiqueta: nombre del atributo.
- Dominio: se refiere al conjunto del que un campo puede obtener sus valores.
- Tipo: se refiere al tipo de dato que admite el campo. Los tipos de datos suelen ser: numérico, alfanumérico, fechas y lógico.
- Indexación: si debe ser indexado o no.
- Lengua: puede ser, o bien la lengua del documento, o bien la del centro de documentación. Eso significa, en el caso de un documento escrito en inglés, que el título estaría en inglés, pero los descriptores en castellano, siempre de acuerdo con el diccionario de datos precedente.
- Controles de validación: restricciones para que los valores cumplan con lo que demanda el atributo.

A continuación se representa en la **Tabla 1** un diccionario de datos para algunos de los atributos más relevantes de la base de datos:

Etiqueta	Dominio	Tipo	Indexación	Lengua	Controles de Validación
idarticulo	Almacena los identificadores de todos los artículos	Entero	Si	Del centro de la documentación	No puede quedar vacío y es único
idblog	Almacena los identificadores de todos los blog	Entero	Si	Del centro de la documentación	No puede quedar vacío y es único
tituloarticulo	Almacena los datos de los nombres de los artículos	Alfanumérico	No	Del centro de la documentación	Se recogen en dependencia al negocio
fechacreacion	Almacena los datos de las fechas de creación del	Fecha	no	Del centro de la documentación	No admite valores fuera de rango

	artículo				
categoria	Describe el tipo de artículo	Alfanumérico	no	Del centro de la documentación	No admite valores fuera de rango y se elige de una lista disponible
resumen	Registra el resumen del artículo	Alfanumérico	no	Del centro de la documentación	Se recogen en dependencia al negocio
textocompleto	Almacena el articulo completo	Cadena de caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
idpersona	Almacena los identificadores de todos las personas del negocio	Numérico	si	Del centro de la documentación	No puede quedar vacío y es único
idimagen	Almacena los identificadores de todos las imagenes del negocio	Numérico	si	Del centro de la documentación	No puede quedar vacío y es único
idcurriculo	Almacena los identificadores de todos los currículos del negocio	Numérico	si	Del centro de la documentación	No puede quedar vacío y es único
nombrecompleto	Almacena el nombre del curriculum	Cadena de Caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
archivo	Guarda el texto del currículo	Cadena de Caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
expcompu	Almacena las áreas de experiencia en computación de la persona	Cadena de Caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio

expprofesionales	Almacena las áreas de experiencia profesional de la persona	Cadena de Caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
certificacionrol	Almacena los roles certificados de las personas del negocio	Cadena de Caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
iddesafio	Almacena los identificadores de todos los desafios	Numérico	si	Del centro de la documentación	No puede quedar vacío y es único
nota_max	Almacena la nota obtenida en el desafio	Numérico	no	Del centro de la documentación	Se recogen en dependencia al negocio
cont_tiempo	Cantidad actual de tiempo consumido	Numérico	no	Del centro de la documentación	Se recogen en dependencia al negocio
idestudiante	Almacena los identificadores de todos los estudiantes	Numérico	si	Del centro de la documentación	No puede quedar vacío y es único
grupo	Grupo actual del estudiante	Alfanumérico	no	Del centro de la documentación	Se recogen en dependencia al negocio
facultad	Facultad a la que pertenece el estudiante	Numérico	no	Del centro de la documentación	Se recogen en dependencia al negocio
grado	Año que cursa el estudiante	Numérico	no	Del centro de la documentación	Se recogen en dependencia al negocio
proyecto	Proyecto al que pertenece el estudiante	Cadena de Caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
idevaluacion	Almacena los identificadores	Numérico	si	Del centro de la documentación	No puede quedar vacío y

	de todas las evaluaciones posibles que otorga el sistema				es único
auto-eval	Es la evaluación al desafío que se da a sí misma la persona	Numérico	no	Del centro de la documentación	Se elige de una lista de valores posibles
comunidad-eval	Es la evaluación que le da la comunidad a la respuesta del desafío de la persona	Numérico	no	Del centro de la documentación	Se elige de una lista de valores posibles
retador-eval	Es la evaluación que le da el retador a la respuesta del desafío de la persona	Numérico	no	Del centro de la documentación	Se elige de una lista de valores posibles
idtarea	Almacena los identificadores de todas las tareas que se programan en el sistema	Numérico	si	Del centro de la documentación	No puede quedar vacío y es único
descripción	Descripción de la tarea programada	Cadena de caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
fechaini	Fecha de inicio de la terea	Fecha	no	Del centro de la documentación	Se recogen en dependencia al negocio
fechafin	Fecha de fin de la tarea	Fecha	no	Del centro de la documentación	Se recogen en dependencia al negocio
horaini	Hora de inicio de la tarea	Fecha	no	Del centro de la documentación	Se recogen en dependencia

					al negocio
horafin	Hora de fin de la tarea	Fecha	no	Del centro de la documentación	Se recogen en dependencia al negocio
compartido	Especifica si la tarea está compartida o no	booleano	no	Del centro de la documentación	Se elige entre True y False
aviso	Especifica el mensaje de notificación de la tarea	Cadena de caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
experiencia	Especifica los datos de todas las experiencias posibles del sistema	Cadena de caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
idexptrabajo	Almacena los identificadores de todas las posibles experiencias de trabajo que se gestionan en el sistema	Numérico	si	Del centro de la documentación	No puede quedar vacío y es único
yearini	Año de inicio del trabajo descrito por la persona	Cadena de caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
yearfin	Año de fin del trabajo descrito por la persona	Cadena de caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
ocupacion	Cargo que ocupó la persona en el trabajo especificado	Cadena de caracteres	no	Del centro de la documentación	Se recogen en dependencia al negocio
centro	Nombre del centro de trabajo que	Cadena de caracteres	No	Del centro de la documentación	Se recogen en dependencia al negocio

	recogido en el curriculo				

Tabla 1. Diccionario de datos.

2.8 Descripción funcional del sistema

La descripción funcional debe incluir los siguientes elementos:

- Qué clase de información se tratará y cómo entrará la información en el sistema.
- Qué procesos documentales se llevarán a cabo.
- Qué servicios y productos generará el sistema, y/o a qué aplicaciones podrá dar soporte.

En el primer punto se debe describir en qué consisten las entradas del sistema. En el segundo punto se debe proporcionar una idea sobre los procesos de tratamiento documental que la base de datos automatiza, y en el tercer punto se debe explicar en qué consisten las salidas del sistema (35).

Para desarrollar el modelo de datos se realizaron los siguientes pasos:

1. Se identificaron todas las tablas del sistema y sus relaciones y se conformó el modelo relacional de la base de datos.
2. Luego se incluyó PL/Python en PostgreSQL para la integración con Neo4j a través del módulo py2neo, el cual brinda una API para todo el trabajo con el gestor. Este paso se desarrolló para darle respuesta a una parte del problema, de tal manera que el modelo permite una manipulación eficiente de los datos almacenados en este tipo de estructuras, debido a que la información se trabajará como un grafo, representando a las personas como nodos y las relaciones entre ellas como aristas.
3. Se crearon las funciones en PostgreSQL para la creación y búsqueda de nodos en Neo4j, además para la creación de las relaciones.
4. Luego por medio del *Foreign Data Wrappers (*FDW) de Redis, se instaló la extensión, garantizando que PostgreSQL pudiese leer los sets de datos referentes a los comentarios creados en Redis, permitiendo la reducción en los tiempos de respuesta.

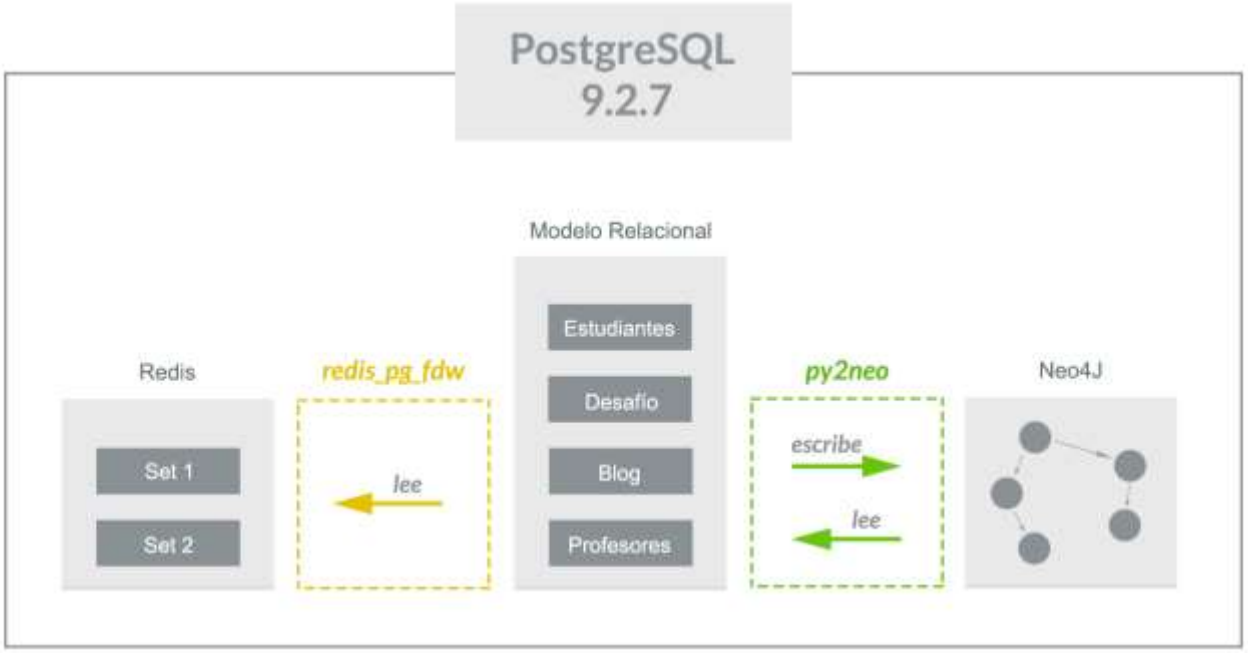


Figura 26. Descripción de la solución.

2.8.1 Pasos para la interacción con la aplicación

A continuación se detallan algunos casos de uso que representan la interacción entre los componentes de la aplicación. Dicha integración permitirá dotar al modelo de datos de los requerimientos de escalabilidad y la reducción de los tiempos de respuesta. El ejemplo a seguir muestra el flujo completo de la inserción de un estudiante, creación de relaciones, etc.

Creación de un estudiante y su correspondiente nodo en Neo4j.

1. Almacenamiento de los datos en la tabla estudiante usando la sentencia INSERT:

INSERT estudiante ();

2. Creación del nodo correspondiente al estudiante en Neo4j usando la función create_nodo_estudiante_db (), a la cual se le pasa como parámetro el identificador de dicho estudiante:

SELECT create_nodo_estudiante_db (8);

3. Comprobar que el estudiante fue creado satisfactoriamente en Neo4j. Para ello usa la aplicación web Browser, con la que se puede consultar el grafo y verificar que el estudiante fue insertado como un nodo en dicho grafo.

Creación de una relación entre dos estudiantes.

Para la creación de una relación entre dos estudiantes, primero se verifica que los estudiantes ya hayan sido creados en la base de datos. Esta verificación es realiza a través de la propia función de creación de la relación, preguntando por el id de ambos.

Por ejemplo, para crear la relación AYUDA_A, se usa la función crear_relacion_est_ayuda_a_est (), a la cual se le pasa como parámetro, un arreglo de enteros, con los id de los estudiantes entre los cuales se creará dicha relación:

SELECT crear_relacion_est_ayuda_a_est ('{1,3}');

Luego, se puede comprobar de igual forma, usando el Browser, que la relación entre ambos estudiantes fue creada satisfactoriamente.

Uso de Redis desde PostgreSQL

Para el uso de Redis desde PostgreSQL, básicamente se creó una tabla foránea en PostgreSQL, la cual puede ser consultada de la siguiente forma:

SELECT * from redis_db0;

Usando el FDW de Redis se pueden consultar remotamente los SET que están visibles en Redis. Vale que aclarar que la versión actual del FDW de Redis sólo permite lecturas, por lo que no se puede escribir en Redis directamente desde PostgreSQL, sólo consultar los datos (40).

Búsquedas en Neo4j desde PostgreSQL

Para la búsqueda de nodos y relaciones en Neo4j desde PostgreSQL, se usaron varias funciones. Por ejemplo, para la búsqueda de todos los profesores guías y los profesores que son jefes, fueron usadas respectivamente las funciones:

SELECT todos_profesores_guias_db ();

SELECT todos_profesores_jefes ();

Ambas devuelven un ResultSet de Python y hacen uso de Cypher, el lenguaje de consultas de Neo4j, y por medio de py2neo se envía la consulta correspondiente a Neo4j.

En caso de que se necesite agregar una nueva función que responda a un nuevo requerimiento, los pasos serían los siguientes:

1. Usar el browser de Neo4j para crear la consulta de Cypher usando la cláusula MATCH.
2. Crear una función en PostgreSQL, usando PL/Python. Esta función contendrá básicamente:

    - El establecimiento de la conexión al grafo en Neo4j.
    - La variable que contendrá el ResultSet, que provendrá de la ejecución de la consulta de Cypher.
    - La sentencia de Cypher que fue anteriormente probada en el browser.

De esta forma se garantiza que todas las búsquedas en Neo4j, se hagan desde PostgreSQL, sin necesidad de usar el browser; lo cual será muy útil para los desarrolladores que usarán el modelo de datos.

2.9 Consideraciones parciales

En este capítulo se enunció el SAH y el SCO del presente trabajo, necesarios para un correcto diseño del modelo de datos relacional. Se definieron y describieron las tablas que componen la base de datos, así como sus atributos.. Se formuló el diccionario de datos y se presentó la secuencia de pasos que se siguieron para la elaboración del modelo de datos. También se implementaron funciones para integrar Neo4j, Redis y PostgreSQL, lo que permitió aprovechar las potencialidades de cada gestor. Neo4j por su parte, al ser una base de datos orientada a grafos permite una manipulación eficiente de los datos almacenados en este tipo de estructuras. Redis, basado en el almacenamiento en tablas de hashes, permite mejorar tiempos de respuestas, mientras que PostgreSQL posee la capacidad de manejar estructuras de datos externas desde el propio gestor.

Capítulo #3: Implantación del modelo de datos.

3.1 Introducción

En este capítulo se valida la propuesta de la solución a través de criterios de especialistas y pruebas de rendimientos con el objetivo de demostrar la integración entre los tres gestores de bases de datos. Se exponen los requerimientos tanto de software como de hardware necesarios para ejecutar el modelo de integración. Se muestra un libro de estilo de datos utilizado como estándar para el desarrollo del modelo de integración.

3.2 Selección del sistema informático

A continuación se presentan los requerimientos de software y de hardware necesarios para interactuar con el modelo de integración.

Requerimientos de software:

- PostgreSQL 9.4
- Redis 2.8.19
- Neo4j 2.1.0-M01
- py2neo 2.0.8
- CentOS Linux 7 para 64 bits
- Foreign Data Wrapper para Redis
- PL/Python
- hiredis 0.12.1
- PgAdmin 1.18

Requerimientos de hardware:

- Para PostgreSQL

- ✓ Servidor con 2 o más CPU de 64 bits
- ✓ Espacio en disco duro 250 gigabytes
- ✓ Conexión Ethernet de 100 Mb/s o más
- ✓ Memoria RAM 4 GB o superior

- Para Redis
  - ✓ Servidor con 2 o más CPU de 64 bits
  - ✓ Espacio en disco duro 250 gigabytes
  - ✓ Conexión Ethernet de 100 Mb/s o más
  - ✓ Memoria RAM 4 GB o superior

- Para Neo4j
  - ✓ Servidor con 2 o más CPU de 64 bits
  - ✓ Espacio en disco duro 500 gigabytes
  - ✓ Conexión Ethernet de 100 Mb/s o más
  - ✓ Memoria RAM 8GB o superior

3.3 Validación del modelo

Para validar el funcionamiento del modelo se aplicaron pruebas de integración. Este tipo de pruebas verifican que los componentes de la aplicación funcionan correctamente actuando en conjunto. En este caso se evalúa la interacción entre dos o más unidades del software.

Prueba: 1	**Tipo de Prueba:** Integración
Artefactos implicados: Funcion crear_nodo_profesor_db()	
Entrada: SELECT crear_nodo_profesor_db(4);	

Salida:

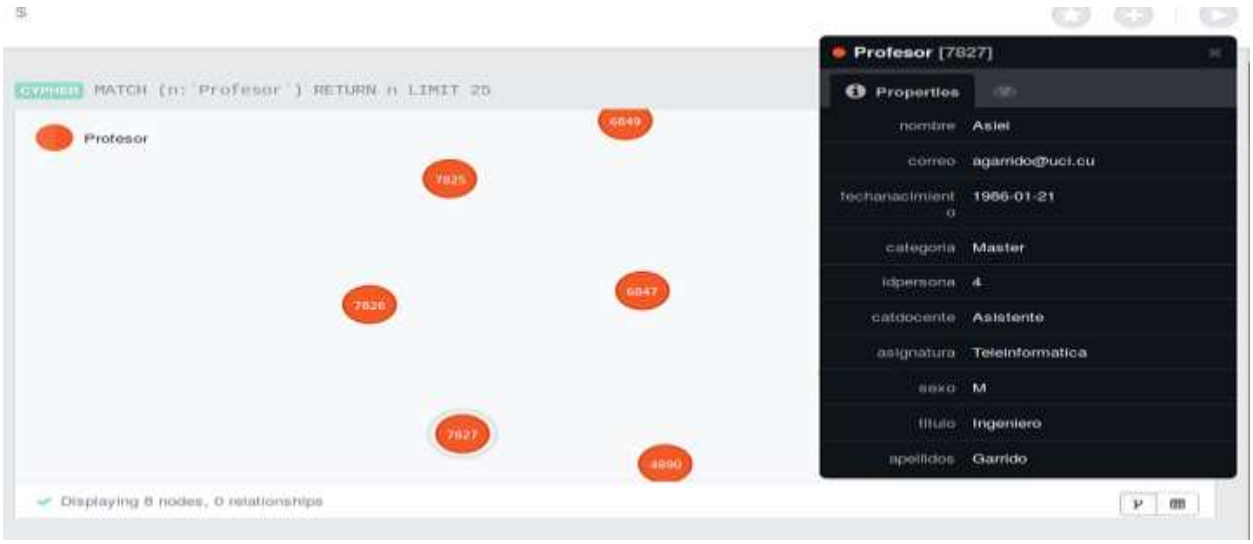


Tabla 2. Prueba 1. Creación de un nodo profesor.

Prueba: 2	**Tipo de Prueba:** Integración
Artefactos implicados: Funcion crear_nodo_profesor_db()	
Entrada: SELECT crear_nodo_profesor_db(3);	
Salida:	

Tabla 3. Prueba 2. Creación de un nodo profesor.

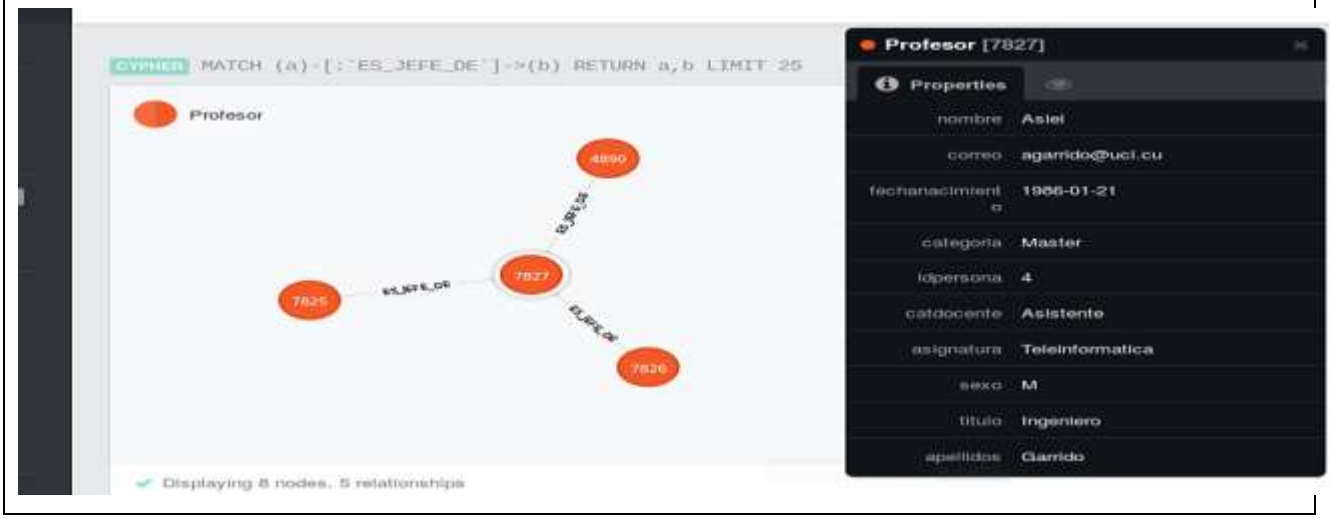
Prueba: 3	**Tipo de Prueba:** Integración
Artefactos implicados: Neo4j's Browser	
Entrada: MATCH (person:Profesor)-[:ES_JEFE_DE]-(jefes) RETURN person.nombre, jefes	
Salida: 	

Tabla 4. Prueba 3. Relación ES_JEFE_DE.

Prueba: 4	**Tipo de Prueba:** Integración
Artefactos implicados: Funcion crear_nodo_estudiante_db()	
Entrada: SELECT crear_nodo_estudiante_db(8);	
Salida: 	

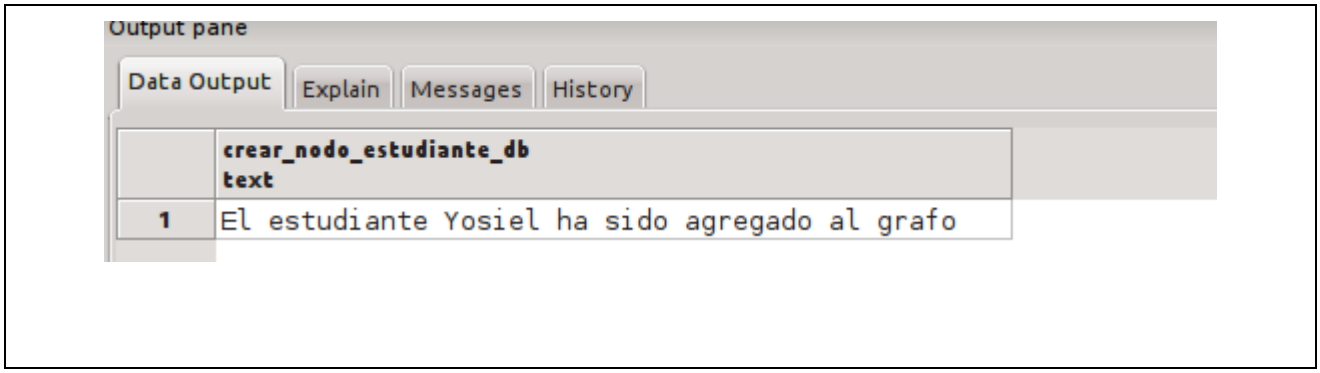


Tabla 5. Prueba 5. Creación de un nodo estudiante.

3.4 Cambios o ajustes necesarios

Una vez ejecutada la máquina virtual se debe seguir el siguiente procedimiento para operar correctamente con el sistema:

1. Verificar el IP de la máquina virtual con el comando hostname –I.
2. Según el IP que retornó la máquina virtual, entrar al fichero de configuración de Neo4j:

nano /opt/neo4j/conf/neo4j-server.properties

Y asignarle el IP que retornó la máquina virtual.

3. Luego iniciar el servicio Neo4j con el fichero:

systemctl start neo4j-service.service

4. Comprobar que el servicio esté funcionando correctamente con el comando:

systemctl status neo4j-service.service

El servicio está funcionando correctamente si el comando anterior devuelve el siguiente resultado:

neo4j-service.service - LSB: The Neo4J graph database server. See http://neo4j.org
Loaded: loaded (/etc/rc.d/init.d/neo4j-service)
Active: active (exited) since lun 2015-06-08 22:33:13 CDT; 13min ago

Process: 559 ExecStart=/etc/rc.d/init.d/neo4j-service start (code=exited, status=0/SUCCESS)
jun 08 22:32:54 localhost.localdomain neo4j-service[559]: WARNING! You are using an unsupported Java runtime.
jun 08 22:32:54 localhost.localdomain neo4j-service[559]: * Please use Oracle(R) Java(TM) 7 to run Neo4...om:
jun 08 22:32:54 localhost.localdomain neo4j-service [559]:

5. Iniciar el servicio Neo4j en el navegador poniendo la URL:

http://{IP de la máquina virtual}:7474/browser/

6. Configurar el servicio PostgreSQL, pare ello se introduce en la máquina virtual el comando:

nano /var/lib/pgsql/data/pg_hba.conf

7. Incluir en la sección de IPv4, en host all all /32 md5 el IP físico de la máquina en la cual se está trabajando.
8. Reiniciar el servicio PostgreSQL con el comando:

systemctl restart postgresql.service

9. Comprobar que el sistema de PostgreSQL esté activado usando el comando:

systemctl status postgresql.service

10. Ejecutar el PgAdmin y cargar la base de datos.

3.5 Libro de estilo de la base de datos

Según la metodología, el libro de estilo de la base de datos es necesario desarrollarlo para aquellos usuarios que una vez terminada el modelo deseen incorporar nuevos módulos o deseen trabajar con la misma (35).

- Todas las funciones se deben escribir en minúscula y deben llamarse de acuerdo a la acción que realiza, (Véase **Figura 27**).
- Las variables deben escribirse con minúscula y deben llamarse de acuerdo a lo que almacenan, (Véase **Figura 28**).
- Todas las funciones deben tener al menos un comentario, (Véase **Figura 29**).
- El nombre de las tablas y de los atributos se definen en minúscula, (Véase **Figura 30**).
- Las funciones en Python se deben escribir usando PEP8[5], (Véase **Figura 31**).

```
CREATE OR REPLACE FUNCTION crear_nodo_estudiante_db(idestudiante integer)
  RETURNS text AS
$BODY$
from py2neo import Graph
from py2neo.cypher import CypherTransaction

graph = Graph("http://neo4j:neo4j@10.8.45.136:7474/db/data")
tx = graph.cypher.begin()
plan = plpy.prepare("SELECT * FROM estudiante WHERE idpersona = $1", ["integer"])
```

Figura 27. Código de una función.

```
graph = Graph("http://neo4j:neo4j@10.8.45.136:7474/db/data")
tx = graph.cypher.begin()
plan = plpy.prepare("SELECT * FROM estudiante WHERE idpersona = $1", ["integer"])
record_estudiante = plpy.execute(plan, [idestudiante])
sentencia = 'CREATE (est:Estudiante {nombre:{nombre}, idpersona:{idpersona}, grupo:{grupo}, facultad:{facultad}, grado:{grado}, proyecto:
```

Figura 28. Declaración de variables.

[5] PEP8 Es una guía de estilo adoptada por la comunidad de usuarios de Python que facilita la lectura del código y la consistencia entre programas de distintos usuarios (42).

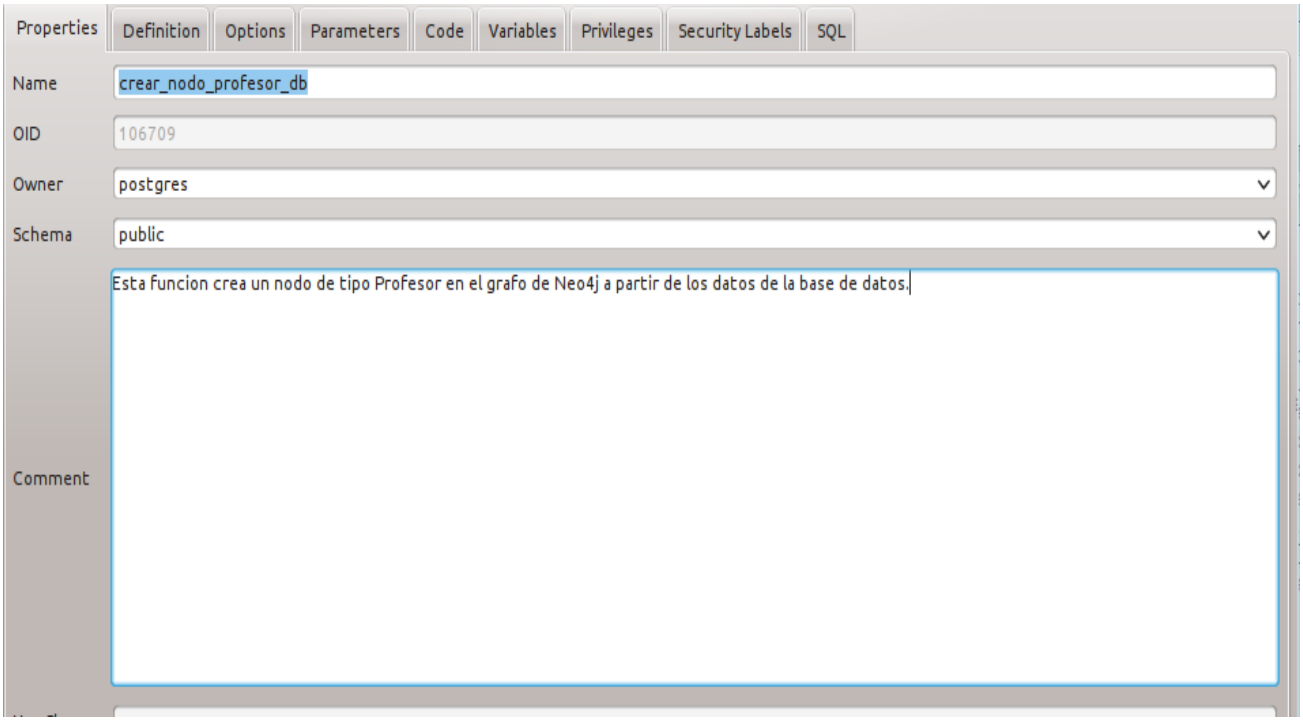


Figura 29. Comentarios en una función.

```
CREATE TABLE estudiante
(
  idpersona integer NOT NULL,
  grupo integer NOT NULL,
  facultad integer NOT NULL,
  grado integer NOT NULL,
  proyecto text,
  nombre text,
  apellido text,
  CONSTRAINT estudiante_pkey PRIMARY KEY (idpersona)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE estudiante
  OWNER TO postgres;
```

Figura 30. Nombre de tablas y atributos.

```
CREATE OR REPLACE FUNCTION crear_nodo_profesor_db(idprofesor integer)
  RETURNS text AS
$BODY$
from py2neo import Graph
from py2neo.cypher import CypherTransaction

graph = Graph("http://neo4j:neo4j@10.8.45.136:7474/db/data")
tx = graph.cypher.begin()
plan = plpy.prepare("SELECT * FROM profesor WHERE persona_idpersona = $1", ["integer"])
record_profesor = plpy.execute(plan, [idprofesor])
sentencia = "CREATE (person:Profesor {nombre:{nombre}, catdocente:{catdocente}, titulo:{titulo}, asignatura:{asignatura}, idpersona:{idper
tx.append(sentencia, parameters={'nombre': record_profesor[0]['nombre'],
                                 'catdocente': record_profesor[0]['catdocente'],
                                 'titulo': record_profesor[0]['titulo'],
                                 'asignatura': record_profesor[0]['asignatura'],
                                 'idpersona': record_profesor[0]['idpersona'],
                                 'apellidos': record_profesor[0]['apellidos'],
                                 'correo': record_profesor[0]['correo'],
                                 'sexo': record_profesor[0]['sexo'],
                                 'fechanacimiento': record_profesor[0]['fechanacimiento'],
                                 'categoria': record_profesor[0]['categoria']})
prof= tx.commit()
if(prof):
 print("El profesor es %s\n", prof)
else:
 print("El profesor %s no esta dentro del grafo.%\n", prof)
 return prof
$BODY$
  LANGUAGE plpythonu VOLATILE
  COST 100;
```

Figura 31. Ejemplo de PEP8.

3.6 Consideraciones parciales

En este capítulo se expusieron los requerimientos de software y de hardware necesarios para de interactuar con el sistema. Se fijó el libro de estilo de que contiene las regulaciones para la construcción de la solución. Se ilustraron los ajustes y configuraciones necesarios para interactuar con la aplicación. Se mostraron los resultados de las pruebas de validación realizadas al modelo. Dichos resultados reflejan que el modelo de datos implementado garantiza un alto rendimiento en la recuperación y procesamiento de los datos.

Conclusiones generales

En el presente trabajo se desarrolló un modelo de datos para la gestión de la información de una Red Social de Aprendizaje usando tres gestores de bases de datos, cada uno encargado de almacenar una parte de la información. Dos de ellos NoSQL (Neo4j y Redis) y uno relacional (PostgreSQL) permitiendo que el modelo de datos garantice la flexibilidad, reducción en tiempos de respuesta y la escalabilidad de los datos.

Para evaluar el modelo propuesto se le realizaron pruebas de integración. Los resultados obtenidos demostraron que la integración de los tres gestores de base de datos generan tiempos de respuestas adecuados para ese tipo de aplicaciones, lo que confirma la eficiencia del modelo.

Recomendaciones

Para futuros trabajos se recomienda:

1. Integrar otros gestores de bases de datos NoSQL al modelo propuesto con el objetivo de aumentar sus prestaciones y garantizar una gestión más eficiente de los datos de la red social de aprendizaje.
2. Agregar al modelo propuesto nuevas funcionalidades de optimización de consultas que minimicen los tiempos de respuesta a los usuarios de la red social de aprendizaje

Referencias bibliográficas

1. Arroyo, Sonia Santana. Revista Cubana de Información en Ciencias de la Salud [ESP]. [En línea] Editorial de Ciencias Médicas, 02 de 2015. [Citado el: 10 de 03 de 2015.] http://www.acimed.sld.cu/index.php/acimed/article/view/98/47.

2. Ponce, Isabel. Monográfico:Redes Sociales [ESP]. [En línea] 17 de 04 de 2012. [Citado el: 2015 de 04 de 04.] http://recursostic.educacion.es/observatorio/web/es/internet/web-20/1043-redes-sociales?showall=1.

3. Definición de Red Social [ESP]. [En línea] 2015. http://definicion.de/red-social/.

4. Orihuela, José Luis. La hora de las redes sociales [ESP]. [En línea] 14 de 10 de 2008. [Citado el: 2015 de 04 de 06.] www.ecuaderno.com/2008/10/14/la-hora-de-las-redes-sociales.

5. Eumed.net [ESP]. [En línea] (http://www.eumed.net/libros-gratis/2012b/1220/concepto_red_social.html)..

6. Informática para novatos [ESP]. [En línea] 11 de 2012. http://www.informaticaparanovatos.com/.

7. ¿Cómo funciona LinkedIn? [ESP]. [En línea] [Citado el: 01 de 04 de 2015.] http://tecnologia.uncomo.com/articulo/como-funciona-linkedin-6482.html.

8. Linkedln Imágenes [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] https://www.google.com.cu/search?q=linkedin&biw=1366&bih=657&source=lnms&tbm=isch&sa=X&ei=4uKKVcSPHYi5-QGRgoLwAw&ved=0CAYQ_AUoAQ.

9. ¿Qué es Facebook? [ESP]. [En línea] [Citado el: 11 de 04 de 2015.] http://quees.la/facebook/.

10. La red social más famosa [ESP]. [En línea] [Citado el: 03 de 04 de 2015.] http://www.plusesmas.com/nuevas_tecnologias/articulos/internet_email/que_es_y_como_funciona_facebook/123.html.

11. Facebook Imágenes [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] https://www.google.com.cu/search?q=facebook&biw=1366&bih=657&source=lnms&tbm=isch&sa=X&ei=hOSKVeagKcPS-QGR-bvQBA&sqi=2&ved=0CAYQ_AUoAQ.

12. redAlumnos Imágenes [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] https://www.google.com.cu/search?q=redalumnos&biw=1366&bih=657&source=lnms&tbm=isch&sa=X&ei=9-OKVd64IcOz-AGE9oK4CA&sqi=2&ved=0CAYQ_AUoAQ.

13. Edmodo Imágenes [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] https://www.google.com.cu/search?q=edmodo&biw=1366&bih=657&source=lnms&tbm=isch&sa=X&ei=FeWKVbi0Hsqy-AG1v4HYBw&ved=0CAYQ_AUoAQ.

14. DIIPO Imágenes [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] https://www.google.com.cu/search?q=diipo&biw=1366&bih=657&source=lnms&tbm=isch&sa=X&ei=buWKVcG9IMro-AGP9J7ICA&ved=0CAgQ_AUoAQ.

15. GARCIA, ROSA MARIA MATO. *Diseño de Bases de Datos.* 1999.

16. L., José Tomás Cadenas. Conceptos Fundamentales de Base de Datos [ESP]. [En línea] http://ldc.usb.ve/~jtcadenas/BasesDeDatos/Conceptos%20Fundamentales.pdf.

17. Pérez, Oniel Boada. *Base de Datos (BD) Relacional para el Sistema.* 2012.

18. Yanes Enríquez, Osmel y Gracia del Busto, Hansel. Revista Digital de las Tecnologías de la Información y las Comunicaciones[ESP]. [En línea] 2012. [Citado el: 07 de 04 de 2015.] http://revistatelematica.cujae.edu.cu/index.php/tele/article/viewFile/74/74.

19. J., Ramakrishnan R. y Gehrke. *Database Management Systems.Third Edition. McGraw-Hill Higher Education..* 2003.

20. Chávez, Marco Antonio Cruz. Conceptos básicos de bases de datos [ESP]. [En línea] [Citado el: 20 de 02 de 2015.] http://www.gridmorelos.uaem.mx/~mcruz//cursos/miic/bd1.pdf.

21. Características, limitaciones y ventajas de PostgreSQL [ESP]. [En línea] [Citado el: 05 de 04 de 2015.] http://postgresql-dbms.blogspot.com/p/limitaciones-puntos-de-recuperacion.html.

22. Velasco, Roberto Hernando. El SGBDR Oracle[ESP]. [En línea] 2014. http://www2.rhernando.net/modules/tutorials/doc/bd/oracle.html.

23. Ortega, Diego Martin Ramos. Monografías.com [ESP]. [En línea] [Citado el: 10 de 05 de 20.] http://www.monografias.com/trabajos73/microsoft-sql-server/microsoft-sql-server.shtml.

24. Características de Microsoft SQL Server [ESP]. [En línea] [Citado el: 04 de 04 de 2015.] http://www.shica19.tripod.com/sql.html.

25. Artaza, Diego Lz. de IpiñaGlz .de. Bases de Datos No Relaciones(NoSQL) [ESP]. [En línea] 04 de 07 de 2012. [Citado el: 10 de 01 de 2015.] http://es.slideshare.net/dipina/nosql-cassandra-couchdb-mongodb-y-neo4j.

26. González, Ismael Gómez. *Propuesta de modelo dimensional y procesos de optimización en bases de datos NoSQL.* 2012.

27. Arquitectura de Apache Cassandra Imágenes [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] https://www.google.com.cu/search?q=arquitectura+de+apache+cassandra&biw=1024&bih=611&source=lnms&tbm=isch&sa=X&ei=y5CLVYzSJczu-AGEr4GICA&ved=0CAYQ_AUoAQ.

28. Bases de Datos NoSQL. Elige la opción que mejor se adapte a tus necesidades. [ESP]. [En línea] Genbetadev, 27 de 01 de 2014. http://www.genbetadev.com/bases-de-datos/bases-de-datos-nosql-elige-la-opcion-que-mejor-se-adapte-a-tus-necesidades.

29. Ojeda, Pablo. SpeaklnBytes [ESP]. [En línea] 20 de 03 de 2014. [Citado el: 30 de 01 de 2015.] http://speakinbytes.com/2014/03/noob-te-presento-a-redis/.

30. Sanchez, Jorge. Xurxo Developer [ESP]. [En línea] 30 de 03 de 2014. [Citado el: 15 de 01 de 2015.] http://xurxodeveloper.blogspot.com/2014/03/neo4j-una-base-de-datos-nosql-orientada.html.

31. Cano, Juan Luis. Introducción a Python para científicos e ingenieros [ESP]. [En línea] 16 de 03 de 2012. [Citado el: 02 de 25 de 2015.] http://pybonacci.org/2012/03/16/introduccion-a-python-para-cientificos-e-ingenieros/.

32. Hernández, Daniel. Bases de datos de grafo [ESP]. [En línea] [Citado el: 10 de 04 de 2015.] http://daniel.degu.cl/cursos/la-web-de-los-datos/taller-3-bases-de-datos-de-grafo.

33. Nicolas2324. Haciendo querys sobre grafos en Neo4j con Cypher [ESP]. [En línea] 10 de 09 de 2013. [Citado el: 15 de 04 de 2015.] https://nicolas2324.wordpress.com/2013/09/10/haciendo-querys-sobre-grafos-en-neo4j-con-cypher/.

34. Metodologías de desarrollo de software [ESP]. [En línea] SOME RIGHTS RESERVED, 30 de 12 de 2006. [Citado el: 04 de 04 de 2015.] http://www.um.es/docencia/barzana/IAGP/Iagp2.html.

35. Codina, Luis. *Metodología de análisis de sistemas de información.* Barcelona : s.n.

36. Alonso, Lidiexy. *Entorno Social de Aprendizaje para la Universidad de las Ciencias Informáticas.* 2014.

37. Tedeschi, Nicolás. ¿Qué es un patrón de diseño? [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] https://msdn.microsoft.com/es-es/library/bb972240.aspx.

38. Diccionario de Informática y Tecnología [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] http://www.alegsa.com.ar/Dic/entidad.php.

39. El Modelo de Datos Entidad-Relación [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] http://basededatos.umh.es/e_r.htm.

40. Redis Cookbook - Practical Techniques for Fast Data Manipulations. [aut. libro] Tiago Macedo y Fred Oliveira. *Redis Cookbook - Practical Techniques for Fast Data Manipulations.* Estados Unidos : O´REALLY, 2011.

41. Marqués, Asier. Conceptos sobre APIs REST [ESP]. [En línea] 11 de 04 de 2013. [Citado el: 05 de 02 de 2015.] http://asiermarques.com/2013/conceptos-sobre-apis-rest/.

42. SPARQL [ESP]. [En línea] https://transparencia.gijon.es/page/12217-servicio-sparql.

43. PEP 8 - La guía de estilo para Python [ESP]. [En línea] [Citado el: 10 de 04 de 2015.] https://bioinf.comav.upv.es/courses/linux/python/estilo.html.

44. Definición de URL [ESP]. [En línea] [Citado el: 10 de 06 de 2015.] http://definicion.de/url/.

45. ¿Qué es XML Schema? [ESP]. [En línea] 15 de 04 de 2004. [Citado el: 04 de 10 de 2015.] http://mat21.etsii.upm.es/mbs/mechxml/que_es_xml_schema.htm.

46. Dave Page, Andrew Dunstan. GitHub Redis FDW for PostgreSQL 9.1+. [En línea] [Citado el: 10 de 01 de 2015.] https://github.com/pg-redis-fdw/redis_fdw.