UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
POSGRADO EN CIENCIA E INGENIERÍA DE LA COMPUTACIÓN

DESARROLLO DE UN SISTEMA PARA LA ESTANDARIZACIÓN Y NORMALIZACIÓN DE UNA BASE DE DATOS BIBLIOGRÁFICA

T E S I S

QUE PARA OBTENER EL GRADO DE:

MAESTRA EN INGENIERÍA
(COMPUTACIÓN)

P R E S E N T A:

ANA PATRICIA GÓMEZ MAYÉN

DIRECTORA DE TESIS:
DRA. AMPARO LÓPEZ GAONA

México, D.F. 2009

A mi familia y amigos.

Resumen

La importancia de las grandes colecciones de datos radica en la información que es posible obtener a partir de ellas; sin embargo, debido a que la información depende directamente de los datos, es necesario que éstos sean de buena calidad para que la información que se obtiene de ellos sea confiable. En este trabajo se describe la calidad de datos.

La base de datos bibliográfica de la Biblioteca Digital (BiDi) de la UNAM, es una de las fuentes de consulta más importantes dentro de la Universidad e integra datos de publicaciones provenientes de diversas fuentes, lo que ha originado problemas en la calidad de los datos debido, principalmente, a la falta de estandarización de algunos campos. El Sistema de Estandarización de Cadenas (SEC), desarrollado en este trabajo, es una herramienta que permite a los usuarios estandarizar cadenas de caracteres a partir de un archivo de datos o de los datos almacenados en una base de datos relacional, utilizando diferentes algoritmos de estandarización de cadenas y creando diccionarios de *sinónimos* como documentos XML.

SEC proporciona métodos para hacer consultas en una base de datos relacional con SQL consultando previamente un diccionario de sinónimos almacenado en una base de datos nativa XML. El uso de SEC fue probado con los datos de las revistas electrónicas almacenados en la base de datos bibliográfica de BiDi; sin embargo, es una herramienta que puede ser utilizada para el desarrollo de nuevas aplicaciones, lo que lo convierte en un sistema reusable, flexible y extensile que no se limita a resolver los problemas de estandarización en la base de datos de la Biblioteca Digital de la UNAM.

Agradecimientos

En estos últimos años han ocurrido bastantes cambios en mi vida y he tenido la oportunidad de toparme con gente a la que me gustaría agradecerle el tiempo que compartió conmigo; así que, por fin y después de mucho tiempo, una vez concluido este trabajo quiero agradecer al Posgrado en Ciencia e Ingeniería de la Computación y a toda la gente que me ayudó a cumplir con los trámites necesarios para la titulación y que sin su ayuda no lo habría logrado. En especial me gustaría nombrar a Diana, a Lulú y al Dr. Boris Escalante.

Quisiera agradecerle al CONACYT el apoyo económico que me brindó durante este periodo de estudios y de manera especial quiero agradecerle a la Dra. Amparo López Gaona su confianza y la oportunidad de permitirme trabajar a su lado. De igual manera quiero agradecerle al Dr. Alfonso Medina, la M.C. Guadalupe Ibargüengoitia, la Dra. Sofía Galicia y al M.C. Egar García por el tiempo que dedicaron para revisar mi trabajo y hacer comentarios que mejoraron en muchos aspectos esta tesis.

Me gustaría agradecerle también al Ing. René Rodríguez Navarro el tiempo que me proporcionó para dedicarme de lleno a terminar esta tesis, su apoyo y sus palabras. Aquí es importante para mí agradecerle de manera muy especial a Eduardo Lemus por la paciencia que me tuvo y aguantarme en los buenos y los malos momentos.

Finalmente les debo dar las gracias a los amigos que me dieron el apoyo moral para terminar las cosas y la alegría para no entristecerme cuando las cosas no van como uno espera. Mencionaré, por el cariño que les tengo, a Daniel y Abraham; además Bardo tiene el crédito de haber diseñado el logotipo del sistema. A mis compañeros de la maestría que pasaron por las mismas dificultades. A mis amigas Deyanira, Isidra y Alma, por estar conmigo y apoyarme.

Índice general

Introducción

Antecedentes

A lo largo de los años, la capacidad de producir datos de diversas índoles se ha visto beneficiada por el avance de la tecnología, lo que ha provocado que se incremente tanto el número de bases de datos como su contenido; sin embargo, la importancia de las grandes colecciones de datos radica en la información[1] que es posible obtener a partir de ellas, por lo que es necesario procesar los datos para obtener información al mismo ritmo en el que se están generando.

Debido a que la información depende directamente de los datos, se debe destacar la importancia de la calidad de los mismos. Los resultados que se obtienen a través de diversas técnicas o sistemas de extracción de información se pueden ver afectados por datos de mala calidad. Esto a su vez puede ocasionar la obtención de información errada que no refleje la realidad de los datos y, por lo tanto, errores en la toma de decisiones.

Calidad de datos es un término relativamente nuevo y que no posee una definición formal. De manera general una buena calidad de datos depende de las expectativas del usuario final, pero determina la capacidad de un sistema de información para reflejar aspectos de la vida real, de ahí la importancia que se le ha dado al término.

[1] En este trabajo, el término información se refiere al conjunto de datos que tiene un significado dentro de un contexto.

La mejor manera de mantener la buena calidad de los datos en una base de datos relacional es insertando datos de calidad desde un principio. De esta manera se evita que en la base existan datos dudosos que afecten la información que se obtiene de ellos. La idea general es la de crear un *firewall* entre los datos a ingresar y la base de datos, de manera que los datos a ingresar se comparen con los que se encuentran en la base de datos y, si cumplen ciertos parámetros de calidad, pueden ser ingresados a misma.

El problema principal que se puede generar al implementar un *firewall* entre los datos y la Base de Datos (BD), es encontrar reglas adecuadas que permitan ingresar a la base de datos información correcta de manera que los parámetros de calidad establecidos no excluyan datos válidos (Ver Figura 1), originando con esto la pérdida de información.

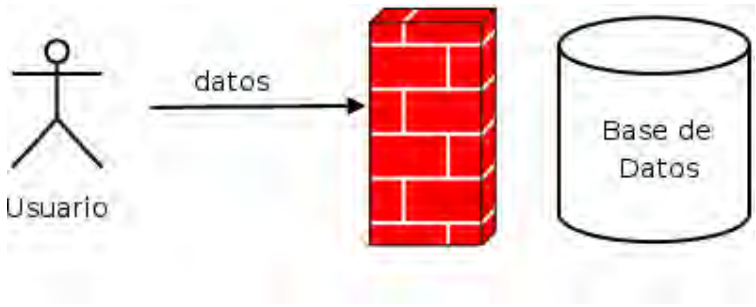


Figura 1: Firewall de inserción de datos

Cuando una BD ya contiene datos, es necesario hacer un análisis del estado actual de la base para determinar si los datos cuentan con una buena calidad que haga confiable la información que se obtiene de ellos. Si no es el caso, es necesario hacer una limpieza de datos, y la estandarización de las cadenas es un aspecto a considerar.

Estandarización de cadenas

Actualmente algunos datos que se representan con cadenas de caracteres pueden representarse de diferentes maneras. En muchos casos, estas distintas representaciones se deben a errores tipográficos en las cadenas; sin embargo, en ocasiones las diferentes representaciones de una cadena son correctas porque todas hacen referencia al mismo elemento. En la Figura 2, cada cuadro contiene ejemplos de algunos de los posibles nombres que puede tener una entidad.

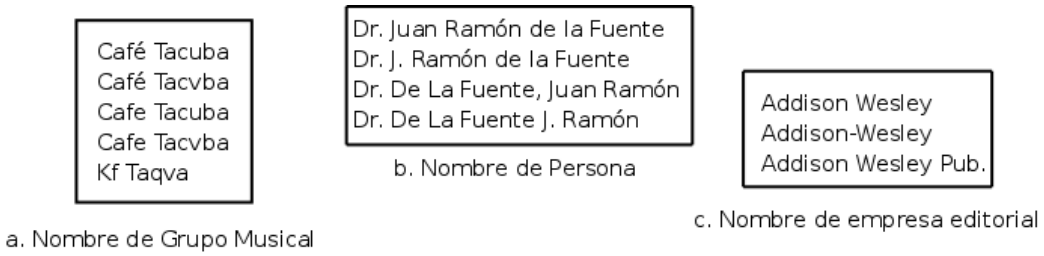


Figura 2: Representación de entidades con distintas cadenas de caracteres

Cuando se deben almacenar datos representados con diferentes cadenas de caracteres en una base de datos relacional, surge la pregunta:*¿Qué cadena se debe almacenar en la base de datos por cada entidad?* En el ejemplo c de la Figura 2, cualquier cadena representa el nombre de la editorial *Addison Wesley*; sin embargo, se debe elegir sólo una cadena de caracteres para estandarizar los valores en la BD y no tener duplicados no exactos, ya que que si se mantienen las inconsistencias, éstas pueden afectar el análisis estadístico que se haga con los datos así como las conclusiones que se formulan a partir de los mismos [6].

El objetivo que se busca es mantener los datos estandarizados en una base de datos relacional y no perder información. Cuando existen dos cadenas de caracteres distintas que representan a una misma entidad, elegir una de ellas para estandarizar la base de datos puede derivar en la pérdida de información. Por ejemplo, supóngase que se tiene una base de datos en donde se encuentran almacenados datos de diferentes grupos musicales y se realiza una estandarización como la mostrada en la Figura 3.

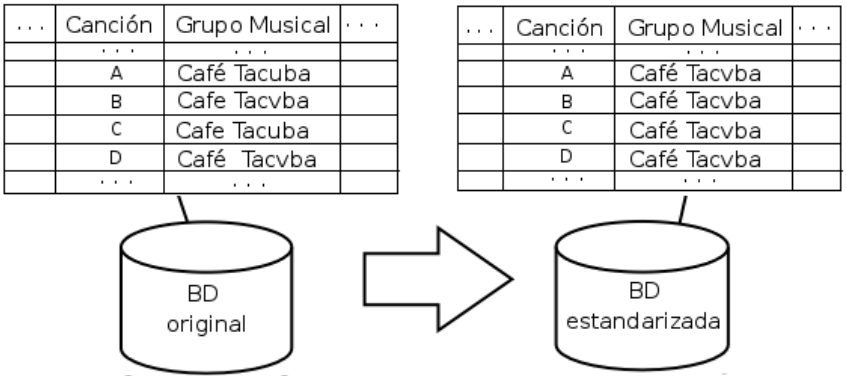


Figura 3: Ejemplo de la estandarización de los datos de la columna `Grupo Musical` de una base de datos.

Aunque se logra el objetivo de estandarizar la base de datos relacional puede existir pérdida de información, ya que todas las cadenas que originalmente se encontraban en la base de datos hacen referencia a una entidad de manera correcta[2], en este caso el grupo musical `'Café Tacvba'`. Cuando se hace una consulta posterior a la estandarización de datos sobre el campo `'Grupo Musical'` utilizando alguna de las cadenas ignoradas durante este proceso, no habrá registros en la base de datos que coincidan en la búsqueda. Ver Figura 4.

Figura 4: Ejemplo de una búsqueda infructuosa en una base de datos estandarizada.

[2]Las cadenas de caracteres mencionadas se consideran correctas porque han sido utilizadas por diferentes medios de comunicación para referirse a dicho grupo musical.

Lo ideal es mantener estandarizada la base de datos relacional y tener métodos de consulta que regresen los mismos resultados, inclusive al usar como término de búsqueda alguna de las cadenas que originalmente se encontraba en la base de datos y no se utilizó para estandarizar la misma.

La estandarización de cadenas permite detectar y eliminar duplicados no exactos. En muchas bases de datos relacionales, el esquema conceptual utilizado no permite que los valores cadena sean fáciles de estandarizar, sobre todo en aquellos campos en los que no se consideraron valores atómicos para representar los atributos de una entidad, por ejemplo las direcciones postales. Las direcciones o domicilios se encuentran compuestos por varios elementos como calle, número, colonia, etc. Si al desarrollar el esquema conceptual de la base de datos no se separaron los campos compuestos en unos más simples, existe una alta probabilidad de encontrar elementos duplicados en ese campo.

Cuando se han utilizado valores atómicos para el desarrollo del esquema conceptual de una BD, la estandarización de cadenas se puede hacer utilizando diversos algoritmos para establecer medidas y obtener indicadores que permitan determinar la similitud entre palabras. Dichos algoritmos dependen en gran medida de los campos a estandarizar y del problema que se quiera resolver.

Una BD con datos estandarizados puede incrementar la eficiencia de las consultas que se ejecutan sobre ella, así como la calidad de la información obtenida, sobre todo cuando las consultas utilizan funciones de agregación sobre los datos. Un ejemplo muy común de este problema es el uso de agrupaciones por un campo en particular, si este campo no se encuentra estandarizado, las agrupaciones no reflejarán la realidad de los datos.

Debido a la importancia que tiene la calidad de datos en sistemas de extracción de información, han surgido empresas que ofrecen herramientas comerciales para hacer la limpieza de los datos en bases de datos específicas; sin embargo, no hay manera de garantizar que las modificaciones que se le hacen a la base de datos como resultado de una limpieza de datos sean permanentes. Las actualizaciones tales como agregar o modificar datos dentro de la base limpia pueden no cumplir con los parámetros de calidad requeridos, por lo tanto se pierde la calidad de los datos en la base.

Bases de datos bibliográficas

Las bases de datos bibliográficas almacenan datos de publicaciones como el nombre de la publicación, el nombre del autor del documento o el nombre de la editorial del mismo. Las bases de datos bibliográficas sólo contienen referencias de las publicaciones, en algunas ocasiones pueden contener un resúmen o extracto de la publicación original pero no el texto completo.

Las bases de datos bibliográficas que integran la información de diversas fuentes son un ejemplo de bases de datos relacionales que son susceptibles a problemas en la calidad de los datos. El problema de calidad en este tipo de bases de datos radica en los problemas de integración en donde se puede destacar:

- La integración de datos considera información que proviene de diversas fuentes, aunque esto no signifique la existencia de un estándar en la presentación de la información.
- Algunos recursos o publicaciones sufren diversas modificaciones a lo largo de su vida, por lo que es necesario hacer actualizaciones constantes en la base de datos.

La base de datos bibliográfica de la Dirección General de Bibliotecas (DGB) de la UNAM, requiere de un sistema de limpieza de datos; sin embargo, debido a las constantes actualizaciones de la base, hacer una limpieza de datos después de cada modificación no es una solución viable, ya que se debe considerar la cantidad de información almacenada y los constantes accesos por parte de los usuarios a los que esta base de datos es sometida pues es una de las fuentes de consulta más importantes dentro de la Universidad.

Entre otros recursos, la base de datos bibliográfica de la DGB almacena la información de las revistas electrónicas que la Universidad contrata con diversos proveedores; por esta misma razón existen campos que no se encuentran estandarizados debido a problemas en la integración de los datos, por ejemplo: el título o el nombre de las editoriales de las publicaciones. Si se pretenden almacenar datos estandarizados, es necesario analizar los datos proporcionados por los diferentes proveedores para elegir cuál será el valor que se debe introducir a la base de datos. Ver Figura 5.

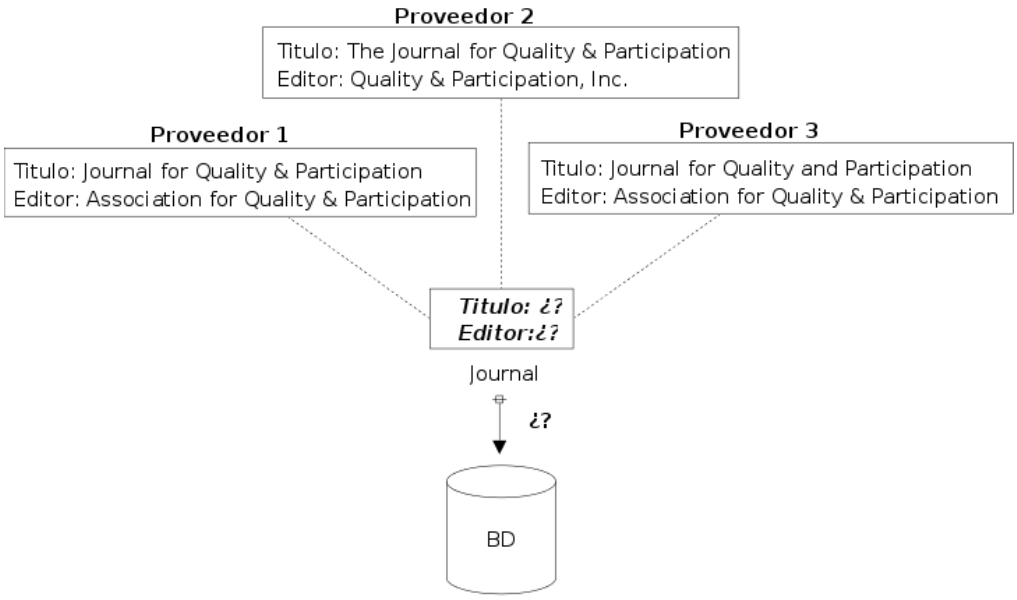


Figura 5: Ejemplo de los problemas de integración en los datos de las revistas electrónicas.

Para aumentar la confiabilidad de la información que se extrae a partir de los datos almacenados en la BD, se debe hacer un preprocesamiento de éstos para estandarizarlos. Este procesamiento de datos no se puede hacer de manera automática puesto que se requiere de personal especializado para llevarlo a cabo con la finalidad de evitar introducir errores en los datos, de ahí la necesidad de contar con un sistema que además de hacer una limpieza de datos pueda mantener esa limpieza tras las modificaciones a la base de datos.

Objetivos

El desarrollo de este trabajo de tesis tiene como finalidad alcanzar los siguientes objetivos:

1. Proporcionar una herramienta que permita a los usuarios estandarizar cadenas de caracteres a partir de un archivo de datos o de los datos almacenados en una base de datos relacional, utilizando diferentes algoritmos de

estandarización de cadenas.

2. Proponer un esquema conceptual normalizado para la representación de la base de datos relacional de la Biblioteca Digital de la Dirección General de Bibliotecas de la UNAM. Dicho esquema debe eliminar la redundancia de los datos almacenados.

3. Proporcionar una herramienta implementada en Java que le permita al usuario utilizar diferentes algoritmos de estandarización de cadenas en la implementación de otros sistemas de software.

4. Definir un esquema XML para la representación de diccionarios de sinónimos.

5. Porporcionar una interfaz entre una base de datos relacional y una base de datos nativa XML, de manera que sea posible realizar consultas SQL en la BD relacional consultando previamente el diccionario almacenado como documento XML en una base de datos nativa XML.

6. Proporcionar una interfaz gráfica para la manipulación de los diccionarios de sinónimos por parte de los usuarios, de manera que la creación de los mismos sea de manera supervisada para evitar introducir errores en los datos y obtener información incorrecta.

Contribución del trabajo

Los datos constituyen uno de los soportes fundamentales para el proceso de toma de decisiones en diferentes ámbitos, por ejemplo: decisiones importantes en las grandes empresas, las cuales pueden ser rutinarias o estratégicas.

Debido a que los datos son intangibles, éstos son susceptibles a varios errores, muchos de los cuales se producen durante la inserción de los datos en la BD y pueden generar la falta de estandarización en los datos, afectando directamente la interoperabilidad de los sistemas que hacen uso de ellos y la información que se obtiene de los mismos.

Cuando las empresas trabajan con datos erróneos es posible que tomen decisiones incorrectas, pierdan clientes y oportunidades y deban aumentar costos en actividades de corrección para cada uno de esos problemas. La falta de estandarización de datos provoca otros errores en la base de datos como la existencia de elementos duplicados.

Este trabajo proporciona una herramienta para llevar a cabo la estandarización de cadenas de caracteres de grandes cantidades de datos basándose en la similitud sintáctica que poseen. Los diccionarios de sinónimos creados a partir del sistema se representan mediante documentos XML, lo cual garantiza que puedan ser utilizados en otras aplicaciones ya que existe una gran varidedad de herramientas disponibles para su procesamiento.

El Sistema de Estandarización de Cadenas (SEC) desarrollado en este trabajo, crea diccionarios de sinónimos basados en un esquema XML propuesto, por lo que es posible realizar búsquedas sobre diferentes diccionarios de sinónimos aunque éstos no hayan sido creados a través de SEC, siempre y cuando cumplan con el esquema establecido.

Con la utilización de SEC se pueden realizar consultas SQL sobre una base de datos relacional consultando previamente un diccionario en una base de datos nativa XML. De esta manera, si se hacen consultas sobre una base de datos relacional en la que que se ha estandarizado la columna A, al buscar cualquier término en A y sus sinónimos (descritos en el diccionario) los resultados deben ser los mismos. Ver Figura 6.

A través de SEC se pueden crear sentencias SQL para estandarizar los valores almacenados en una base de datos relacional, así como consultas en XQuery para hacer búsquedas en los documentos XML.

SEC se encuentra completamente implementado en Java por lo que es un sistema multiplataforma, además de que cuenta con instaladores para Windows XP y Windows Vista, así como un instalador para Unix[3]. Además cuenta con manuales de usuario y de instalación.

[3]El instalador para Windows está desarrollado con NSIS [42] y el instalador para Unix se encuentra desarrollado con IzPack [41].

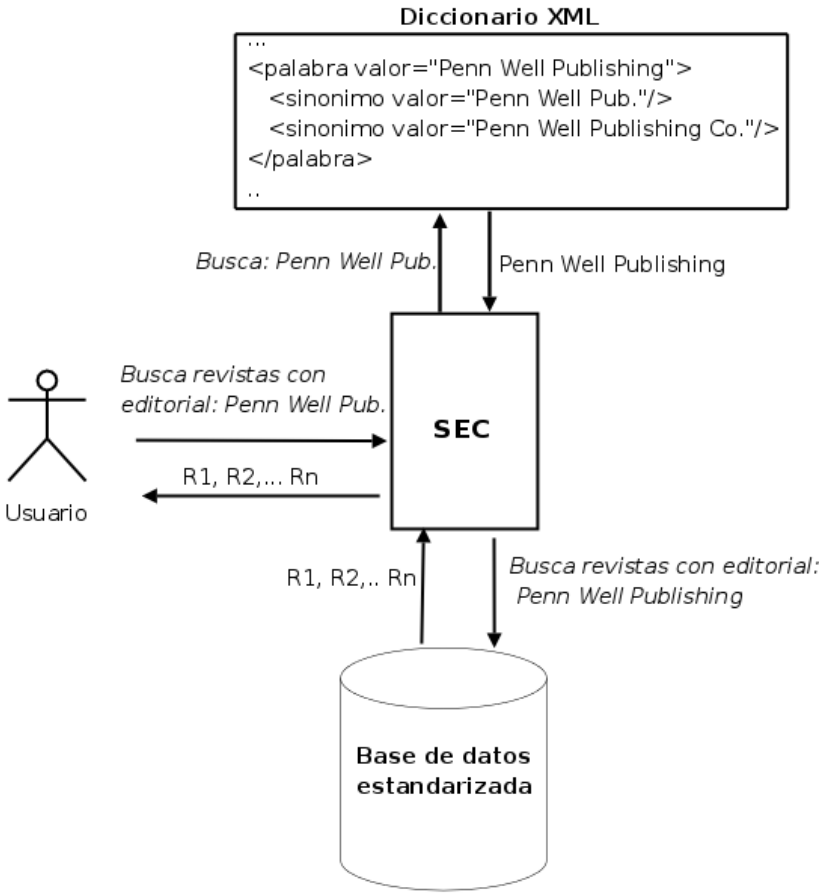


Figura 6: Esquema de búsquedas en una base de datos normalizada utilizando un diccionario de sinónimos a través de SEC.

SEC es una herramienta de apoyo en la creación de aplicaciones para la estandarización de datos ya sea que éstos se encuentren en un archivo de datos o almacenados en una base de datos relacional. SEC proporciona además una interfaz gráfica para la manipulación de los diccionarios de sinónimos, la cual permite crear y modificar los diccionarios de manera independiente a cualquier aplicación. Debido a que los diccionarios de sinónimos se encuentran representados como documentos XML y son almacenados en una base de datos nativa XML, no es necesario modificar los sistemas de información que hacen uso de los datos.

Estructura del documento

En este documento se presenta el proceso de desarrollo del Sistema de Estandarización de Cadenas (SEC) y el proceso de normalización de la Base de Datos Bibliográfica de la Biblioteca Digital de la UNAM, enfatizando que los procesos de estandarización de cadenas y la elaboración de los diccionarios de sinónimos pueden ser utilizados en cualquier colección de datos que requiera una estandarización de cadenas.

En el primer capítulo se presenta a grandes rasgos la Biblioteca Digital (BiDi) de la UNAM y la estructura de la base de datos bibliográfica de esta Institución. Se hace un análisis simple de los datos almacenados para los registros de las revistas electrónicas contratadas por la Universidad, haciendo énfasis en los problemas de duplicidad de registros. Analizando el sistema de consultas del portal de BiDi[4] se muestran algunas inconsistencias en los resultados obtenidos debido a la falta de estandarización de los datos.

En el segundo capítulo se describe la calidad de datos y la importancia de la misma en grandes colecciones de datos. Se presenta la limpieza de datos como un conjunto de técnicas que se utilizan para eliminar elementos duplicados o registros inconsistentes. En este capítulo se describen diversos algoritmos de estandarización de cadenas utilizados en SEC.

En el capítulo 3 se hace un análisis del lenguaje XML y las Bases de datos nativas XML, así como las ventajas de almacenar información en el formato XML. Se describe la base de datos nativa XML *eXist* y los lenguajes de consulta sobre XML *XPath* y *XQuery* para hacer consultas sobre los documentos.

En el capítulo 4 se presenta el Sistema de Estandarización de Cadenas (SEC) desarrollado en este trabajo. Las secciones que conforman este capítulo abarcan desde el análisis de requerimientos hasta la validación del sistema utilizando los datos de las revistas electrónicas almacenados en la base de datos de la Biblioteca Digital.

Por último se presentan las conclusiones obtenidas tras el desarrollo de este trabajo.

[4] http://bidi.unam.mx

Capítulo 1

Biblioteca Digital, UNAM

La Universidad Nacional Autónoma de México (UNAM) cuenta con datos de diversas publicaciones que corresponden a recursos bibliográficos que la comunidad universitaria puede consultar a través de la Dirección General de Bibliotecas (DGB) por medio de la Biblioteca Digital de esta misma institución. La base de datos de la DGB proporciona información útil para toda la comunidad universitaria y externa y representa la fuente de información por medio de la cual es posible consultar y aprovechar los recursos que proporciona la Universidad.

Los datos de los recursos bibliográficos de la UNAM forman la base de datos Bibliográfica de la Universidad y está sujeta a constantes actualizaciones debido a que los datos almacenados deben reflejar la realidad de los recursos. Dado que muchas publicaciones son contratadas por diversos proveedores es necesario hacer una integración de los datos para evitar tener registros duplicados en la base de datos o inconsistencias en la información que se obtiene de ellos.

La Biblioteca Digital (BiDi) es un área que pertenece a la Direccción General de Bibliotecas, fue fundada en mayo de 2001 y desde entonces tiene como misión ofrecer a la comunidad universitaria el acceso a diversos acervos de información en formato digital y responder a las demandas de información de los usuarios de acuerdo con los requerimientos de sus áreas de estudio [34].

1.1. Biblioteca Digital

Existe mucha confusión en torno a la definición de una *biblioteca digital* debido a dos factores principales [3]:

1. Las bibliotecas digitales son el punto de atención de muchas áreas diferentes de investigación y lo que constituye una biblioteca difiere dependiendo de la comunidad de investigadores que se refiere a ella.

2. Existen elementos en Internet que se hacen llamar bibliotecas digitales, aunque desde un punto de vista bibliotecario no lo son.

Aunque en muchos grupos la palabra *biblioteca* ha sido tomada para referirse a una colección de objetos digitales, una *biblioteca digital* no se refiere únicamente a la colección digitalizada de documentos y al uso de herramientas tecnológicas para administrar la información, se trata de un medio que conjuga el ciclo de la creación, diseminación, uso y preservación de los datos, la información y el conocimiento [2], ya que en un principio las bibliotecas digitales tienen los mismos propósitos, funciones y metas que una biblioteca tradicional.

En el entendido de que las bibliotecas digitales, antes que todo son bibliotecas, es posible listar algunas de sus características [3]:

- Incluyen datos de colecciones impresas, digitales y multimedia.
- Permiten el acceso remoto a los recursos de información de otras bibliotecas o repositorios.
- Idealmente proporcionan una visión coherente de toda la información contenida en la biblioteca, sin importar su formato.

Los primeros trabajos en los que se comenzó a dar un acercamiento teórico a la biblioteca digital se publicaron a principios de la década de 1990[1]. Uno de los primeros intentos para crear una biblioteca digital fue el de *Mercury Electronic Library Project* [2]. A este proyecto le siguieron otros que exploraron el uso de

[1] Entre las principales metas que resaltó la biblioteca digital se encontraba el rescate de algunos materiales impresos, por lo que entre los primeros proyectos destacaron aquellos que procuraban que los materiales valiosos fueran digitalizados.

[2] Carnegie Mellon University, 1987-1993

imágenes escaneadas de artículos, el más conocido de los cuales fue el proyecto *Elsevier Science Publishing's Tulip*. Aún cuando estos proyectos no fueron a largo plazo demostraron que eran muy grandes los beneficios potenciales de una biblioteca digital [2].

Actualmente la construcción de una biblioteca digital representa serios retos. La integración de medios digitales a colecciones tradicionales no es cosa sencilla debido, principalmente, a la naturaleza de la información digital. Algunos de los aspectos más significativos a los que se enfrenta el desarrollo de bibliotecas digitales se mencionan a continuación.

1.1.1. Arquitectura técnica

Al diseñar una biblioteca digital se debe considerar la arquitectura técnica que se requiere para almacenar las colecciones digitales, así como el acceso simultáneo de los usuarios de la misma. La arquitectura técnica debe incluir componentes como [3]:

- Redes de alta velocidad y rápida conexión a Internet.
- Bases de datos que soporten diversos formatos digitales.
- Motores de búsqueda de texto completo para crear índices y proporcionar el acceso a las fuentes.
- Funciones de administración de documentos electrónicos, que ayudarán en el manejo integral de los recursos digitales.
- Bases de datos bibliográficas para almacenar las referencias a los metadatos de las colecciones digitales e impresas.

Cuando en la biblioteca digital se permite el acceso a los recursos de otras bibliotecas digitales, se requiere de normas para interactuar y compartir recursos. Sin embargo, entre las diferentes bibliotecas digitales existe una diversidad de estructuras de datos, motores de búsqueda, interfaces, formatos, etc. que no permite la facilidad de compartir recursos entre las bibliotecas [3].

1.1.2. Construcción de las colecciones digitales

La construcción de las colecciones digitales representa uno de los aspectos más importantes en la creación de una biblioteca digital ya que éstas deben ser relevantes y útiles [1]. Las colecciones se pueden formar de la conversión de las colecciones impresas a formato digital, la adquisición de obras digitales originales y el acceso a materiales externos como colecciones de otras bibliotecas.

Cuando los recursos de una biblioteca dependen de la digitalización de los documentos existentes, se requiere definir qué partes de la colección se deben de digitalizar y si las restricciones de derechos de autor permiten la digitalización. Esta decisión afecta directamente en el tamaño y contenido de la colección final de la biblioteca digital.

Metadatos

Los documentos de una biblioteca digital se identifican a través de *metadatos*, que son los datos que describen el contenido y los atributos de un documento en particular.

Los metadatos son parte de una infraestructura de información técnica concreta, incluso hasta cierto punto en el nivel semántico, ya que en un principio estaban pensados para ser independientes del contexto [5]; sin embargo, estas estructuras de representación de la información se basan en la utilización del lenguaje natural por lo que se requiere de herramientas terminológicas para compartir recursos entre bibliotecas digitales.

La necesidad de procesamiento de los recursos digitales, así como la necesidad de herramientas computacionales, hacen que la construcción de una biblioteca digital requiera de trabajo de especialistas en bibliotecología para la selección de recuros, catalogación de los mismos y de científicos de la computación para realizar los procesos de extracción y búsqueda sobre las colecciones.

Los derechos de autor (Copyright)

El concepto actual de *derechos de autor*, basado en la literatura impresa, se viene abajo en el ambiente digital debido a que se pierde el control sobre las

copias. Los objetos digitales son menos rígidos, más fáciles de ser copiados y accesibles a distancia por muchos usuarios de forma simultánea [3].

Los derechos de autor han sido uno de los principales obstáculos a los que se enfrentan muchas bibliotecas digitales, ya que es poco probable poder proporcionar acceso libre a un documento amparado por las leyes de autoría [3] ya que se tendrían que desarrollar mecanismos para administrar el *copyright*, mismos que permitan proporcionar la información sin infringir los derechos de autor.

Algunas de las funciones que la administración de derechos de autor podrían incluir son:

- Administrar las transacciones de los usuarios para permitirles el acceso sólo a un determinado número de copias, cobrarles por cada una o bien trasladar la solicitud de consulta a una editorial

- Proporcionar la situación de *copyright* de cada objeto digital, las restricciones para su uso y las tarifas pertinentes.

1.2. Base de datos bibliográfica

Los datos de los libros y las revistas electrónicas contratadas por la UNAM, se almacenan en una base de datos bibliográfica. Una base de datos bibliográfica es una base de datos referencial, que como su nombre lo indica, es una base de datos que almacena referencias de documentos, es decir: se guarda sólo la información fundamental para describir y permitir la localización de los documentos ya sean impresos o electrónicos.

Dentro de una base de datos bibliográfica, cada registro corresponde a una referencia bibliográfica, para la cual existen distintos campos como título o autores; sin embargo, aún no es posible establecer un estándar para el diseño de los registros bibliográficos de recursos digitales debido la diversidad de los mismos[3].

[3]La UNESCO publicó *The Common Communication Format* para incidir el diseño de registros y bases de datos bibliográficos en diversos entornos geográficos [5]

La base de datos bibliográfica de BiDi almacena información de diversos recursos como:

- Revistas electrónicas.
- Libros electrónicos.
- Bases de datos.
- Bibliotecas Digitales.
- Recursos electrónicos como almanaques, anuarios, atlas, etc.
- Sitios en Internet.

Al igual que la base de datos bibliográfica de la DGB, muchas de las bases de datos bibliográficas disponibles son de naturaleza interdisciplinaria y consecuentemente pueden considerarse fuentes generales de referencia [4].

Para este trabajo de tesis se utilizarán las tablas de la base de datos bibliográfica de BiDi que almacenan la información de las revistas electrónicas, ya que pertenecen al grupo de recursos que se encuentra en constante actualización e integra información de diversas fuentes. Debido a que las revistas son proporcionadas por diversos proveedores, se trabajará también con la información de los proveedores.

1.3. Las revistas de la UNAM

La Universidad contrata el servicio de acceso al texto completo de revistas electrónicas a través de diversos proveedores. Las suscripciones con los proveedores tienen un periodo de tiempo válido. Por mencionar algunos proveedores se tiene:

- **EBSCO Host**. `http://www.ebscohost.com`. Base de datos que ofrece textos completos, índices y publicaciones periódicas académicas que cubren diferentes áreas de estudio como las ciencias y las humanidades.

- **WILEY**. `http://www3.interscience.wiley.com`. Base de datos que ofrece acceso a artículos en texto completo, tablas de contenido y resúmenes de temática multidisciplinaria.

- **JSTOR**.`http://jstor.org`. Ofrece información de revistas arbitradas agrupadas en cinco colecciones retrospectivas: Artes y Ciencias; Negocios; Ecología y Botánica; Salud y Ciencias Generales; Matemáticas y estadística.

- **BioMed Central**.`http://biomedcentral.com`. Colección de revistas científicas arbitradas de biomedicina y ciencias de la salud, permite el acceso al texto completo y resúmenes de artículos de investigación.

Las revistas electrónicas pueden sufrir una serie de modificaciones durante su tiempo de vida, razón por la cual las tablas de datos de las revistas electrónicas se encuentran en constante actualización. Algunos de estos cambios son:

- Una revista puede cambiar de nombre.

- Una revista se puede fusionar con otra y cambia el ISSN de la revista en la que se fusionaron.

- Una revista puede cambiar de nombre y debe cambiar de ISSN como consecuencia de esta modificación.

- Una revista puede cambiar de editorial.

Los cambios que sufren las revistas electrónicas son registrados en bitácoras de los proveedores de las mismas. Periódicamente los proveedores dan a conocer información de los cambios realizados y es cuando la Universidad debe actualizar la información contenida en la base de datos.

Dentro de las actualizaciones que se deben ejecutar sobre la base de datos de BiDi, se incluye eliminar los enlaces a texto completo de los proveedores que ya no proporcionan dicho recurso o agregar nueva información. De cualquier modo, es necesario integrar la información de los diversos proveedores para verificar que la información a eliminar sea la correcta o, en caso contrario, evitar insertar duplicados.

La estructura de la base de datos que corresponde a la información de las revistas electrónicas se muestra en la Figura 1.1.

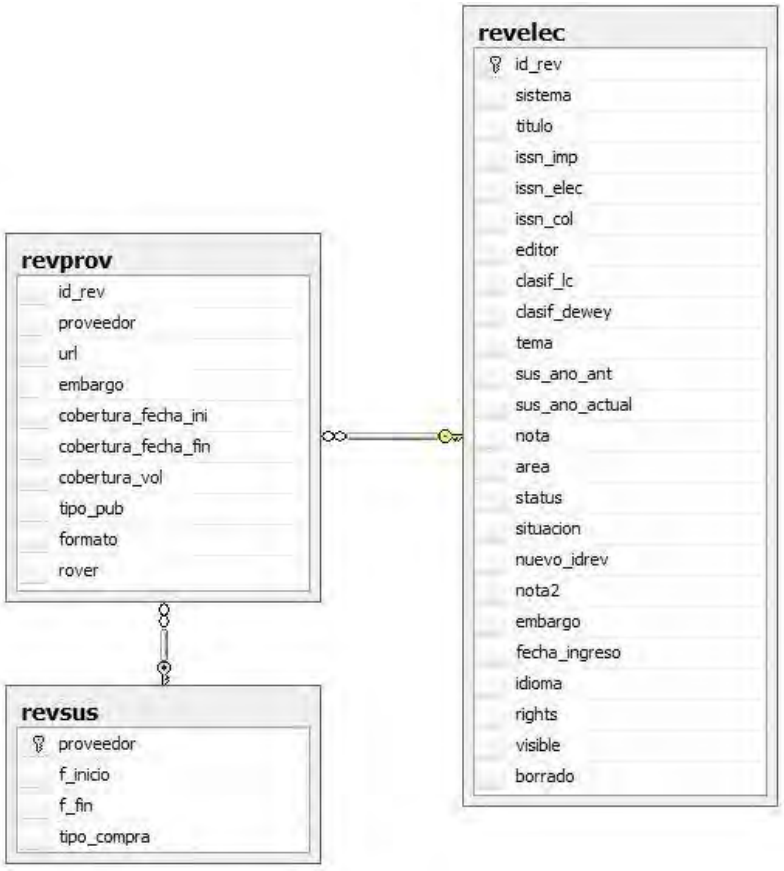


Figura 1.1: Esquema de la base de datos (Revistas electrónicas)

En la estructura se consideran 3 tablas principales: una para la información de las revistas (*revelec*), una para almacenar la información de los proveedores de las revistas (*revprov*) y una para la información de la suscripción que se tiene con cada proveedor (*revsus*). La descripción de los campos de cada una de las tablas en la base de datos relacional de BiDi mencionadas se describen en las Tablas 1.1, 1.2 y 1.3.

Campo	Descripción
id_rev	Identificador único, asignado automáticamente por el sistema va de 1 a n.
sistema	Número del sistema Aleph.
titulo	Título de la revista.
issn_imp	ISSN de la revista en formato impreso.
issn_elec	ISSN de la revista en formato electrónico.
issn_col	ISSN de la colección.
editor	Editor de la revista. Cuando la revista tiene más de un editor, sólo se pone un valor o se separara con el caracter '/'.
clasif_lc	Clasificación *Library of Congress*. Valor que se le asigna a las revistas que pertenecen al Congreso de E. U. A.
clasif_dewey	Clasificación Dewey
tema	Tema de la Revista. Cuando la revista tiene varios temas, estos se separan por ';'.
sus_ano_ant	Número de suscripciones impresas al título del año anterior.
sus_ano_actual	Número de suscripciones impresas al título del año actual.
nota	Observaciones e información adicional.
area	Área temática a la que pertenece: 1.- Ciencias físico-matemáticas e ingenierías; 2.- Ciencias biológicas y de la salud; 3.- Ciencias sociales; 4.- Humanidades y artes.
status	Estatus de la revista: Vigente o no vigente.
situacion	Estado que guarda la publicación: cambios.
nuevo_idrev	Nuevo número de sistema de la revista. No todas las revistas poseen este valor.
nota2	Otras observaciones.
embargo	Embargo de la revista, en caso de existir, se considera en meses.
fecha_ingreso	Fecha en que se ingresa el registro a la base de datos. Asignada por el sistema.
idioma	Idioma en que se encuentra la información.
rights	Derechos de autor, correspondientes a este recurso.
visible	Valor booleano que indica si el resurso es visible al usuario.
borrado	Valor boolean que indica su el recurso está seleccionado para ser borrado.

Tabla 1.1: Descripción de los valores de la tabla, `revelec`

Campo	Descripción
proveedor	Nombre del proveedor de la suscripción (único).
f_inicio	Fecha de inicio de la suscripción.
f_fin	Fecha de término de la suscripción.
tipo_compra	Tipo de compra de la publicación: contrato, factura, gratuito o licencia.

Tabla 1.2: Descripción de los valores de la tabla `revsus`

Campo	Descripción
id_rev	Identificador único de la revista.
proveedor	Nombre del proveedor con el que se contrata la revista.
url	Dirección electrónica para acceso al recurso electrónico, uno por cada revista para cada proveedor.
embargo	Número de meses en los que la revista no puede ser accesada a pesar de haber sido contratada.
cobertura_fecha_fin	Fecha de término de la cobertura.
cobertura_vol	Desde el volumen y fascículo que se encuentra disponible.
tipo_pub	Indica que tipo de recurso se pueden accesar, por ejemplo: tabla de contenido, resúmen y texto completo.
formato	Formato del texto completo al que hace referencia el url, puede ser HTML, PDF, HTML/PDF u otro.
rover	Valor booleano para usarse en otra aplicación, indica si hay problemas de acceso al texto completo: si es verdadero el acceso es disponible, falso en otro caso.

Tabla 1.3: Descripción de los valores de la tabla `revprov`

1.3.1. Búsquedas en BiDi

Uno de los desafíos a los que se enfrentan los administradores de la base de datos bibliográfica de BiDi es mantener actualizada la información. Dado que la información de un mismo recurso puede provenir de diversas fuentes, no se ha elaborado una integración adecuada lo que ha originado que exista información duplicada o no haya una estandarización de los datos. Esto puede afectar algunos procesos en donde se deben hacer agrupaciones sobre campos no estandarizados como `editorial`.

Una de las dificultades para hacer una correcta actualización de la base de datos radica en la variedad de formatos en la que los proveedores publican las actualizaciones periódicas de las revistas contratadas por la Universidad. En muchos de estos casos, la información se encuentra en diversos formatos como HTML, MARC y archivos de texto plano.

La falta de estandarización de los datos ocasiona errores en los sistemas de información de BiDi. Un ejemplo de lo anterior se muestra en la Figura 1.2, en la cual se puede observar que el nombre de la editorial de los *libros* de BiDi no se encuentra estandarizado, por lo que al listar el nombre de las editoriales éstos se repiten; sin embargo, el sistema funciona de manera adecuada, lo que no es correcto son los datos.

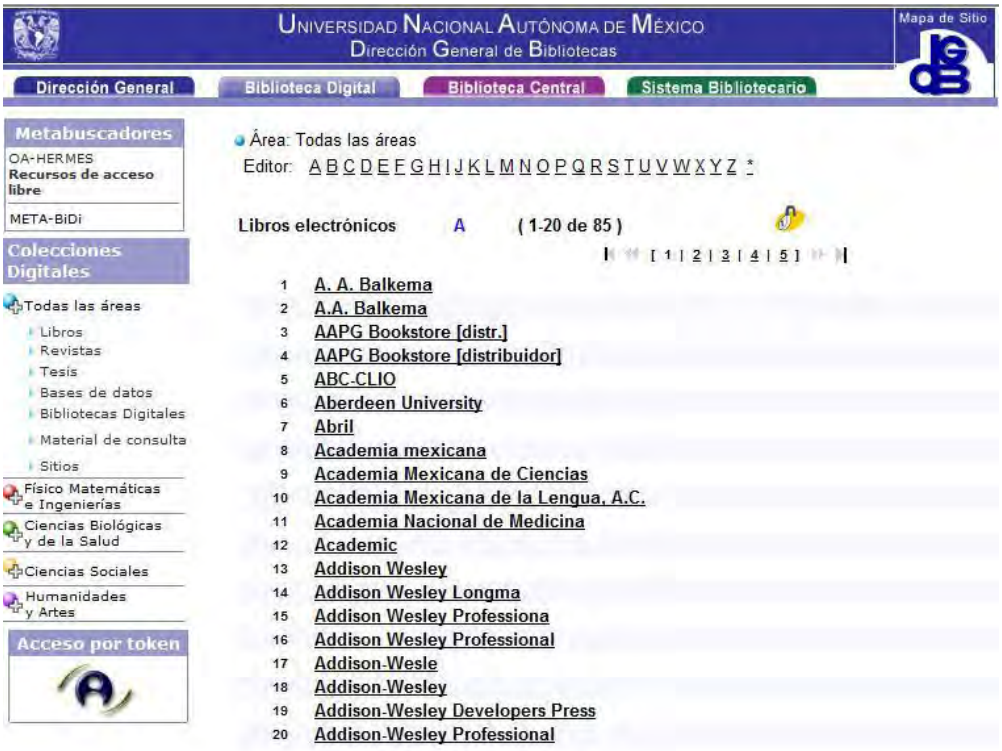


Figura 1.2: Listado por editoriales (Libros)

En el ejemplo que se muestra en la Figura 1.2 se enlistan los nombres de 20 editoriales que deberían ser distintas; sin embargo, se puede notar que existe un subconjunto de 10 elementos en donde aparecen nombres de editoriales duplicados ya que contienen errores tipográficos en su escritura o simplemente no se encuentran estandarizados. A estos elementos se les conoce como duplicados no exactos ya que hacen referencia al nombre de la misma editorial. Si se toman estos 10 elementos seleccionados y se agrupan aquellos duplicados no exactos, la lista se reduce a 4 como se muestra en la Figura 1.3.

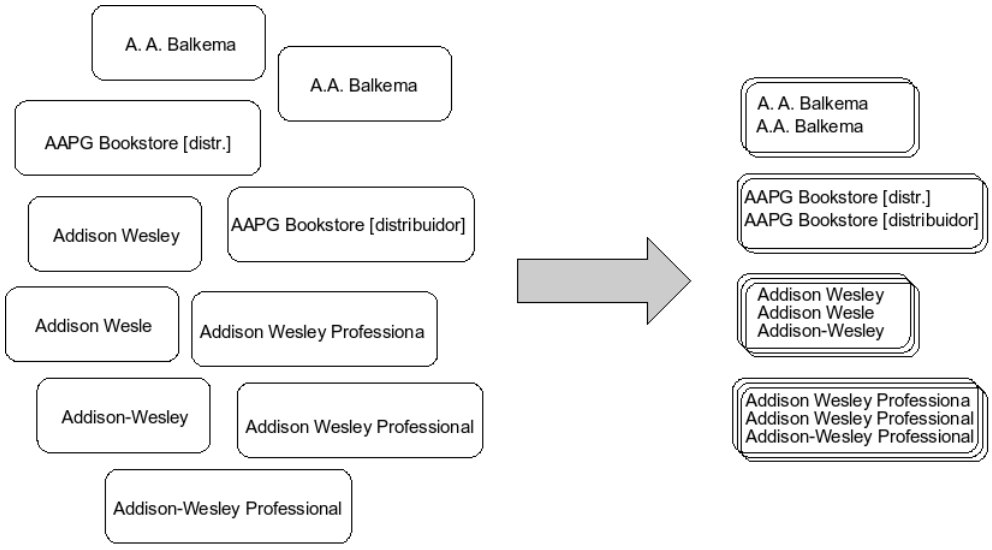


Figura 1.3: Agrupación de editoriales duplicadas en libros.

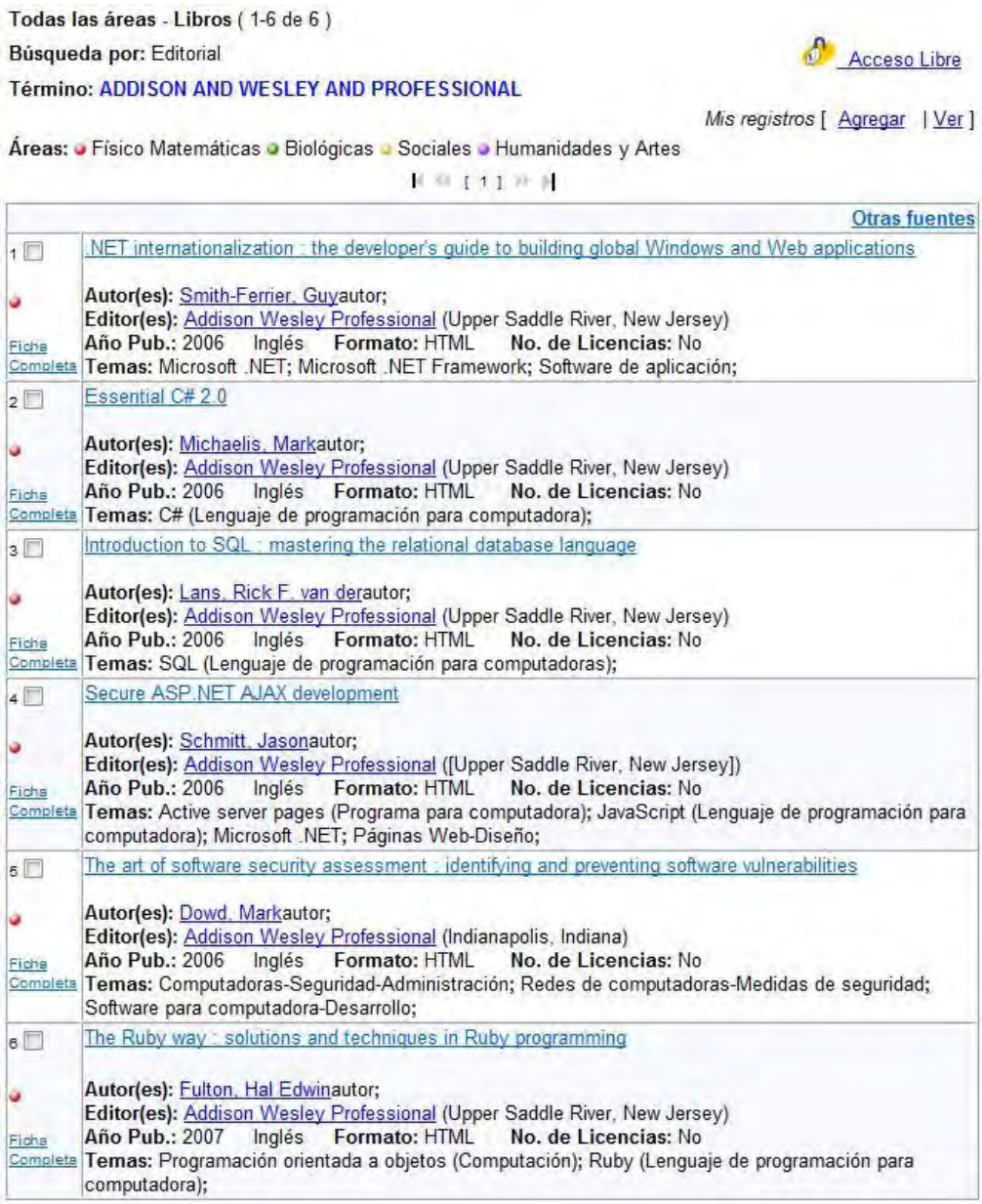
Este problema se debe a falta de estandarización en el campo `editorial` de los libros, lo que afecta a otros procesos del sistema como las consultas por editorial. En la Figura 1.4 se muestra un ejemplo de una búsqueda por editorial a través del portal de BiDi. El término buscado es `Addison Wesley Profesiona`

En la Figura1.5 se realiza la búsqueda de la misma editorial pero con una cadena de caracteres diferente: `Addison-Wesley Professional`. Como se puede ver, los resultados varían en ambas consultas. Finalmente, si realizamos una tercera búsqueda de la misma editorial, se obtienen resultados muy diferentes, como se muestra en la Figura 1.6.

Figura 1.4: Búsqueda por editorial en libros (`Addison Wesley Professiona`).

Figura 1.5: Búsqueda por editorial en libros (`Addison-Wesley Professional`).

Como se puede notar, todos los resultados obtenidos en las diferentes búsquedas hacen referencia a la misma editorial; sin embargo, como el nombre no se encuentra estandarizado, el sistema regresa diferentes resultados. Si se buscan todas las publicaciones de un editor en particular el sistema no devuelve información confiable.



Todas las áreas - **Libros** (1-6 de 6)

Búsqueda por: Editorial

Término: ADDISON AND WESLEY AND PROFESSIONAL

Acceso Libre

Mis registros [Agregar | Ver]

Áreas: Físico Matemáticas Biológicas Sociales Humanidades y Artes

[1]

Otras fuentes

1 Ficha Completa	.NET internationalization : the developer's guide to building global Windows and Web applications **Autor(es):** Smith-Ferrier, Guyautor; **Editor(es):** Addison Wesley Professional (Upper Saddle River, New Jersey) **Año Pub.:** 2006 Inglés **Formato:** HTML **No. de Licencias:** No **Temas:** Microsoft .NET; Microsoft .NET Framework; Software de aplicación;
2 Ficha Completa	Essential C# 2.0 **Autor(es):** Michaelis, Markautor; **Editor(es):** Addison Wesley Professional (Upper Saddle River, New Jersey) **Año Pub.:** 2006 Inglés **Formato:** HTML **No. de Licencias:** No **Temas:** C# (Lenguaje de programación para computadora);
3 Ficha Completa	Introduction to SQL : mastering the relational database language **Autor(es):** Lans, Rick F. van derautor; **Editor(es):** Addison Wesley Professional (Upper Saddle River, New Jersey) **Año Pub.:** 2006 Inglés **Formato:** HTML **No. de Licencias:** No **Temas:** SQL (Lenguaje de programación para computadoras);
4 Ficha Completa	Secure ASP.NET AJAX development **Autor(es):** Schmitt, Jasonautor; **Editor(es):** Addison Wesley Professional ([Upper Saddle River, New Jersey]) **Año Pub.:** 2006 Inglés **Formato:** HTML **No. de Licencias:** No **Temas:** Active server pages (Programa para computadora); JavaScript (Lenguaje de programación para computadora); Microsoft .NET; Páginas Web-Diseño;
5 Ficha Completa	The art of software security assessment : identifying and preventing software vulnerabilities **Autor(es):** Dowd, Markautor; **Editor(es):** Addison Wesley Professional (Indianapolis, Indiana) **Año Pub.:** 2006 Inglés **Formato:** HTML **No. de Licencias:** No **Temas:** Computadoras-Seguridad-Administración; Redes de computadoras-Medidas de seguridad; Software para computadora-Desarrollo;
6 Ficha Completa	The Ruby way : solutions and techniques in Ruby programming **Autor(es):** Fulton, Hal Edwinautor; **Editor(es):** Addison Wesley Professional (Upper Saddle River, New Jersey) **Año Pub.:** 2007 Inglés **Formato:** HTML **No. de Licencias:** No **Temas:** Programación orientada a objetos (Computación); Ruby (Lenguaje de programación para computadora);

Figura 1.6: Búsqueda por editorial en libros (`Addison Wesley Professional`).

Los sistemas de información son cajas negras para el usuario. El usuario espera que la información que le devuelve el sistema sea correcta; sin embargo, la falta de calidad de los datos almacenados ocasionan que se pierda la confiabilidad del sistema porque devuelve información errónea.

1.3.2. Proceso de actualización de revistas electrónicas

La base de datos bibliográfica de BiDi contiene la información de las revistas contratadas por la Universidad y de los diversos proveedores de las mismas. Para actualizar la información se sigue un proceso de actualización que debe ser desarrollado por personal especializado que se encarga de revisar que la información en BiDi coincida con la información proporcionada en las bitácoras de actualización de los proveedores.

Las bitácoras de actualización de los proveedores son archivos con la información completa de las colecciones de revistas contratadas. En estos archivos no se especifica cuáles revistas sufrieron cambios ni cuáles fueron agregadas o eliminadas. La identificación de estos elementos se debe hacer por el personal de BiDi.

Pasos del proceso de actualización

1. Se obtienen los listados de revistas contratadas por cada uno de los proveedores.
2. En cada uno de los archivos se identifica la información relevante que debe estar almacenada en la base de datos de BiDi.
3. Se buscan los títulos que ya se encuentran en la base de datos y se separan aquellos que se encuentran en el archivo pero no en la base para identificarlos como nuevos.
4. De los elementos que ya se encuentran en la base de datos se verifican los datos de las revistas que pudieron haber sufrido algún cambio, como por ejemplo: título, editorial o ISSN.
5. Una vez identificadas las modificaciones a realizar, se hacen los cambios a través de un formulario (Interfaz) campo por campo de cada una de las revistas modificadas.

6. Los datos de las revistas separadas como nuevas se vuelven a comparar con los datos almacenados en la base de datos. Es necesario verificar cada uno de los datos para asegurar no agregar registros duplicados en la base de datos. Este proceso se hace con personal especializado, el cual puede determinar si es viable agregar un nuevo recurso o si se trata de algún recurso ya almacenado en BiDi.

7. Los elementos identificados como nuevos son agregados a la base de datos una vez que se le incluye información propia que la universidad requiere, como el área temática.

Este es un procedimiento manual, ya que no se puede realizar de manera automatizada debido a que es un trabajo especializado. Por este motivo se puede suponer que la base de datos se encuentra desactualizada, ya que actualmente se cuenta con información de aproximadamente 26,000 revistas electrónicas[4].

A pesar del trabajo que implica la actualización de la información, por la cantidad de datos almacenados, el procedimiento se repite para los diferentes proveedores. Si una revista es proporcionada por más de un proveedor, la base de datos puede presentar inconsistencias pues los proveedores no tienen un estándar en la presentación de la información. Algunos problemas importantes son:

- Algunos registros pueden ser identificados como nuevos cuando no lo son y se agrega información duplicada en la base de datos.

- Se pueden realizar varias modificaciones a un registro debido a que no hay un estándar entre los proveedores para escribir algunos datos.

- Debido a que los procesos de actualización de cada proveedor se dividen entre diferentes miembros del equipo para hacer las actualizaciones, por el esquema conceptual de la base de datos no es posible saber si se hace más de una actualización por campo.

[4]Hasta el 12 de mayo de 2009.

1.3.3. Análisis de los datos de las revistas en BiDi

En esta sección se hará un análisis simple de los datos de revistas electrónicas almacenados en la base de datos bibliográfica de BiDi. Como se mencionó anteriormente, debido al proceso de actualización y a la falta de estandarización de los datos por parte de los diferentes proveedores, los campos más susceptibles a errores son: `editor` y `título`.

Uno de los procedimientos de la actualización de la base de datos consiste en la inserción de nuevos registros de revistas. Cuando en este paso se omite insertar la información que identifique a las revistas, es posible que se agreguen elementos duplicados en actualizaciones posteriores.

Los datos que permiten identificar fácilmente los registros de las revistas electrónicas son: el título y el ISSN, ya sea impreso o electrónico. En la Tabla 1.4 se muestra el número de revistas almacenadas en la base de datos de BiDi que carecen de uno o más campos que faciliten su identificación. Además de la existencia de valores nulos (`NULL`) para indicar que no existe un dato, muchos registros tienen en estos campos una cadena vacía[5] para indicar que dicho campo carece de dato.

Campo	**NULL**	ϵ
`sistema`	18,682	172
`issn_imp`	4,831	0
`issn_elec`	9,929	11,908
`editor`	136	9,799
`tema`	7	7

Tabla 1.4: Número de registros con valores faltantes (`NULL` o ϵ) para la identificación de las revistas electrónicas en la tabla revelec.

[5]En las tablas, el valor de la cadena vacía se representará con ϵ.

Debido a las características del ISSN, se puede suponer que cuando se tiene este dato es posible identificar una revista. En la base de datos existen 4,829 registros de revistas que carecen de ISSN tanto impreso como electrónico, por lo que no es posible utilizar este dato para identificar un 19.29 % de registros en la base de datos (Ver Figura 1.7).

Figura 1.7: Gráfica circular en donde se muestra la proporción de revistas en la base de datos que no cuentan con dato en los campos `issn_imp` ni `issn_elec`.

De los registros que cuentan con ISSN, es posible encontrar elementos duplicados en la base de datos bajo la siguiente consideración: Dos registros a y b almacenados en la tabla `revelec` son duplicados si tienen el mismo ISSN tanto impreso como electrónico, para encontrar estos registros se hace la siguiente consulta:

```
SELECT a.id_rev,a.titulo, a.issn_imp, a.issn_elec
FROM revelec as a, revelec as b
WHERE a.issn_imp =b.issn_imp AND
a.id_rev!=b.id_rev AND
a.issn_elec=b.issn_elec;
```

El resultado de la consulta se muestra en la Tabla 1.5, en donde se puede notar que los registros que tienen el mismo valor de dato en los campos `issn_imp` y `issn_elec`, además tienen títulos iguales o muy semejantes entre sí.

id_rev	titulo	issn_imp	issn_elec
19289	Epidemiology	1044-3983	1531-5487
7594	Epidemiology - Baltimore	1044-3983	1531-5487
10374	Human Reproduction	0268-1161	1460-2350
4118	Human Reproduction	0268-1161	1460-2350
2552	Rheumatology	1462-0324	1462-0332
10410	Rheumatology	1462-0324	1462-0332

Tabla 1.5: Lista de revistas de la tabla `revelec` que tienen el mismo ISSN tanto impreso como electrónico.

Para determinar si los registros mostrados en la Tabla 1.5 se encuentran duplicados en la base de datos, se utilizan los datos de los proveedores almacenados en la tabla `revprov` mediante la siguiente consulta:

```
SELECT id_rev,proveedor,cobertura_vol
FROM revprov
WHERE id_rev in(19289,7594,10374,4118,2552,10410);
```

En los resultados de la consulta (mostrados en la Tabla 1.6) se puede notar que los registros de la Tabla 1.5 son proporcionados por diferentes proveedores y cada uno de ellos suministra una cobertura diferente. Con lo anterior, se puede suponer que los registros de la Tabla 1.5 se encuentran duplicados en la BD.

id_rev	proveedor	cobertura_vol
19289	JSTOR	Año 1990-2001- Vol. 1 - 12
7594	SWETS	Año 2001- Vol. 12 No. 1
10374	OUP	Año 1996- Vol. 11 No. 7
4118	PROQUEST	Año 1999- Vol. 14 No. 1
2552	OUP	Año 1996- Vol. 35 No. 10
10410	PROQUEST	Año 1994- Vol. 33 No. 1

Tabla 1.6: Datos de la tabla `revprov` que pertenecen a los registros mostrados en la Tabla1.5.

Cuando se pretende localizar elementos duplicados entre registros que no tengan ISSN electrónico, se puede sospechar que basta con comparar el ISSN impreso mediante la siguiente consulta a la base de datos:

```
SELECT a.id_rev,a.titulo,a.issn_imp,a.issn_elec,a.fecha_ingreso
FROM revelec as a, revelec as b
WHERE a.issn_imp =b.issn_imp AND
a.id_rev!=b.id_rev
ORDER BY  titulo
```

La consulta anterior regresa 57 registros, de los cuales 6 coinciden con los mostrados en la Tabla 1.5. Todos los registros que comparten ISSN impreso se repiten por pares excepto uno, algunos de los registros que se obtienen después de hacer la consulta anterior se muestran en la Tabla 1.7.

id_rev	titulo	issn_imp	issn_elec	ingreso
20980	IMRN - International Mathematics Research Notices	1073-7928	1687-0247	2007-06-27
21086	International Mathematics Research Notices	1073-7928		2007-06-27
21214	Journal of Neurosurgery	0022-3085		2007-06-27
21215	Journal of Neurosurgery: Pediatrics	0022-3085		2007-06-27
21803	Merlyn's pen	0882-2050		2007-06-27
21804	Merlyn's Pen: Middle School Edition	0882-2050		2007-06-27
21805	Merlyn's Pen: Senior Edition	0882-2050		2007-06-27
6484	Review of Central and East European law.	0925-9880		2005-12-16
11865	Review of Central and East European Law	0925-9880	1573-0352	2006-02-20
25055	Revista Mexicana de Sociologia	0188-2503		2007-06-29
19297	Revista Mexicana de Sociología	0188-2503		2007-04-25

Tabla 1.7: Datos de revistas que comparten ISSN impreso con alguna otra en la base de datos.

Se puede notar que muchas revistas revistas mostradas en la Tabla 1.7 tienen como fecha de ingreso del día 27 de junio de 2007. Para verificar que no se trata de registros duplicados, es posible apoyarse en los datos de los proveedores que se encuentran almacenados en la tabla `revprov`, los cuales se muestran en la Tabla 1.8.

id_rev	proveedor	url
21214	EJS	http://ejournals.ebsco.com/direct.asp?JournalID=104815
21215	EJS	http://ejournals.ebsco.com/direct.asp?JournalID=711191
20980	SWETS	http://www.swetswise.com/link/access_db?issn=1073-7928
20986	EBSCO	http://search.ebscohost.com/direct.asp?db=fth&jid= %225X3 %22&scope=site

Tabla 1.8: Datos de los proveedores de las revistas mostradas en la Tabla 1.7

En la Tabla 1.8 es posible identificar algunas revistas que comparten ISSN impreso y además son proporcionadas por el mismo proveedor pero con diferente enlace (`url`). Como ejemplo se pueden tomar las revistas que tienen identificador `21214` y `21215`, ambas son proporcionadas por EJS[6] y tienen tíulos diferentes a pesar de tener el mismo ISSN impreso. Al buscar estos datos en la página de Internet del proveedor se encuentra lo mostrado en la Figura 1.8.

Figura 1.8: Fragmento de la lista de títulos del proveedor EJS (EBSCO) publicada en Internet.

[6]Electronic Journals Service (EJS).

En este caso en particular, aunque parece una inconsistencia el que dos revistas tengan el mismo ISSN impreso, la base de datos de BiDi contiene correctamente los datos que proporciona el proveedor. El que exista un ISSN diferente puede deberse a que la revista cambió de nombre pero no de ISSN; sin embargo, la cobertura al igual que el nombre debió cambiar, aunque esto no se vea reflejado en la base de datos.

Por otro lado, los datos de los proveedores de las revistas que aparecen en la Tabla 1.7 y que no se ingresaron a la base de datos el 27 de junio de 2007 se muestran en la Tabla 1.9.

id_rev	**proveedor**	**url**
6484	EBSCO	http://search.ebscohost.com/direct.asp?db=buh&jid=MZ0&scope=site
11865	SWETS	http://www.swetswise.com/link/access_db?issn=0925-9880
25055	UCLA	http://www.ejournal.unam.mx/rms/rms_index.html
19297	JSTOR	http://www.jstor.org/journals/01882503.html

Tabla 1.9: Datos de los proveedores en la tabla `revprov` relacionados con las revistas de la Tabla 1.7.

Debido a que las revistas agrupadas de la Tabla 1.7, que no se ingresaron el 27 de junio de 2007, tienen el mismo ISSN impreso y las diferencias entre los títulos son mínimas, se puede suponer que los registros se encuentran duplicados en la base de datos; además, como la información proviene de diferentes fuentes es probable que los datos no se encuentren estandarizados.

Como se mencionó anteriormente, existen 4,829 registros en la base de datos que carecen de valor en los campos `issn_imp` y `issn_elec`. En este caso, la única manera de buscar registros duplicados es revisar cada uno de ellos. Para simplificar la tarea se pueden buscar los títulos iguales dentro de la base de datos mediante la siguiente consulta:

```
SELECT a.id_rev,a.titulo, a.issn_imp, a.issn_elec
FROM revelec as a, revelec as b
WHERE a.titulo =b.titulo AND
a.id_rev!=b.id_rev
```

En total, el resultado de la consulta consta de 264 registros de los cuales se tomaron los mostrados en la Tabla 1.10 para su análisis[7]. En este conjunto de datos se observa que las revistas tienen el mismo título pero el valor del ISSN impreso es diferente. Para verificar que no se trata de elementos duplicados, se pueden utilizar los datos de los proveedores almacenados en la tabla `revprov` mostrados en la Tabla 1.11.

id_rev	Título	issn_imp	issn elec
12154	Adweek	AAAA-0249	
3744	Adweek	0199-2864	
12155	Adweek Magazines'Technology Marketing	AAAA-0250	
11380	Adweek Magazines'Technology Marketing	1536-2272	
179	Analyst	0003-2654	1364-5528
5438	Analyst	0741-7918	

Tabla 1.10: Lista de algunos registros que se almacenaron con el mismo título pero tienen diferente ISSN impreso

id_rev	Proveedor	cobertura_vol
12154	PROQUEST	Año 2003- Vol. 44 No. 5
3744	EBSCO	Año 1993- Vol. 43 No. 27
12155	PROQUEST	Año 1997-2003
11380	EBSCO	Año 1993- Vol. 34 No. 27
179	EJS	Año 1997- Vol. 122 No. 1
5438	JSTOR	Año 1874-1883

Tabla 1.11: Datos de los proveedores de las revistas mostradas en la Tabla 1.10

En este caso, como las revistas son proporcionadas por diferentes proveedores y el ISSN impreso no coincide entre las revistas que tienen el mismo título, para verificar que no existen elementos duplicados en la base de datos, es necesario revisar cada uno de los registros por separado y comparar los datos de la base de datos de BiDi con la información que facilitan los proveedores de las revistas.

[7]Una lista larga de algunos resultados de esta consulta se muestran en el Apéndice A.

Si se observan las coberturas de cada uno de los proveedores mostradas en la Tabla 1.11, se puede suponer de manera general que se trata de elementos duplicados y que cada proveedor proporciona una cobertura diferente, asumiendo que se deben actualizar los datos de la BD de BiDi y buscar los valores correctos en las diferentes fuentes de información. Esta suposición se debe verificar por un experto.

Como se puede notar, encontrar elementos duplicados en la base de datos no es una tarea sencilla debido a la constante actualización de los datos de las revistas electrónicas. Hasta ahora se han mencionado casos en donde se pueden hacer conjuntos pequeños para un análisis basado en agrupación de valores de datos iguales en campos como: título o ISSN impreso; sin embargo, no se puede hacer mucho cuando existen datos con valores cadena que son diferentes pero tan semejantes que se puede suponer tratan de representar el mismo valor.

El sistema de búsquedas de revistas electrónicas a través de Internet que proporciona la Biblioteca Digital, permite hacer listas alfabéticas de las revistas electrónicas disponibles por diferentes campos como título, proveedor o editorial (Ver Figura 1.9). Cuando se elige una letra, por ejemplo de los títulos, se listan todos los títulos de las revistas que comienzan con dicha letra.

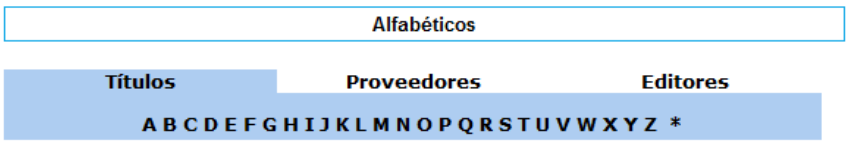


Figura 1.9: Imagen del portal de búsqueda de la Biblioteca Digital para seleccionar una lista de títulos de revistas electrónicas alfabéticamente al seleccionar una letra.

Si se ingresaron registros duplicados a la base de datos, en los cuales el título es el mismo pero se capturó con variaciones en su escritura o con errores tipográficos, no es posible encontrar fácilmente elementos duplicados agrupando por el título. Esto se puede ver en las listas alfabéticas que se muestran a través del portal de BiDi. En la Figura 1.10, se ha solicitado el listado de los títulos que comienzan con la letra **'S'**.

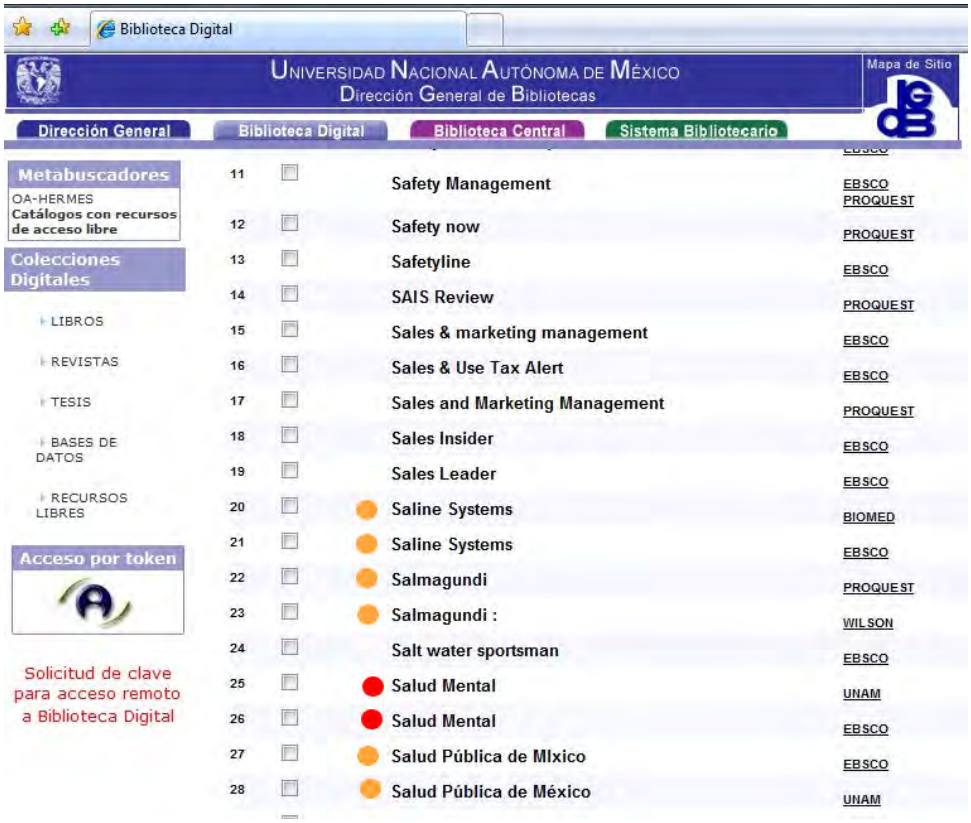


Figura 1.10: Fragmento de la lista alfabética de títulos de revistas electrónicas en BiDi. Listado de títulos que comienzan con la letra 'S'.

En la Figura 1.10 se muestra una lista de 18 títulos que comienzan con la letra 'S'. En esta imagen es posible identificar tres inconsistencias:

1. El título `'Salmagundi:'` tiene un signo de puntuación que lo hace diferente al título `'Salmagundi'`. Es claro que se trata del mismo título pero las representaciones cadena no son las mismas.

2. El título `'Salud Pública de Mlxico'` tiene un error tipográfico. El título correctamente escrito debe ser `'Salud Pública de México'`, que ya se encuentra almacenado en la base de datos.

3. Los títulos `'Saline Systems'` y `'Salud Mental'` aparecen más de una vez como si se tratara de títulos diferentes.

Al hacer la búsqueda por título con el término `'Salud Mental'` en el sistema de consultas de revistas electrónicas de BiDi, se obtienen los resultados mostrados en la Figura1.11. En esta figura se observa que las revistas que tienen el título `'Salud Mental'` están almacenadas en la base de datos con registros diferentes porque no comparten el valor del ISSN impreso o electrónico.

Figura 1.11: Resultados de la búsqueda por título: Salud Mental

Al buscar los datos de las revistas en la información proporcionada por los proveedores se puede notar que ambos registros hacen referencia la misma revista electrónica (Ver Figura 1.12), por lo que se trata de un elemento duplicado en la base de datos.

Figura 1.12: Datos de la revista electrónica titulada 'Salud Mental', proporcionada por los diferentes proveedores.

En la base de datos existen 24,889 títulos diferentes de revistas electrónicas, lo que significa que 135 revistas comparten el título con alguna otra en la base de datos; sin embargo, existe un gran número de revistas en donde los títulos son muy semejantes a otros, pero no son iguales. Algunos se listan a continuación [8]:

- Tool & Equipment Aftermarket Industry Profile United States
- Tool & Equipment Aftermarket Industry Profile: United States
- Staples Inc SWOT Analysis
- Staples, Inc. SWOT Analysis
- Sumitomo Chemical Company Ltd SWOT Analysis
- Sumitomo Chemical Company, Ltd. SWOT Analysis
- Yogurt Industry Profile Europe
- Yogurt Industry Profile: Europe

Para verificar que los registros con títulos semejantes pero no iguales hacen referencia a la misma revista, es necesario revisar los datos almacenados en la base de datos. Si se toman los siguiente títulos:

- Tool & Equipment Aftermarket Industry Profile: United States
- Tool & Equipment Aftermarket Industry Profile United States

Se pueden obtener los identificadores de las revistas electrónicas y la fecha en la que se ingresaron a la base de datos de BiDi (Ver Tabla 1.12).

id_rev	**Título**	**fecha ing**
24355	Tool & Equipment Aftermarket Industry Profile: United States	2007-06-27
16596	Tool & Equipment Aftermarket Industry Profile United States	2006-02-20

Tabla 1.12: Datos de las revistas que tienen dos títulos muy semejantes pero están almacenadas en la base de datos de BiDi como revistas diferentes

[8] En el Apéndice B existe una lista más grande de títulos de revistas electrónicas semejantes pero no iguales, tomados de la base de datos de BiDi.

En la Tabla 1.13 se muestran los datos de ISSN impreso y electrónico, editor, área temática y tema almacenados en la base de datos de BiDi de las revistas electrónicas mencionadas anteriormente.

id_rev	issn_imp	issn_elec	editor	area	Tema
24355	NULL	NULL	Datamonitor Plc	13	Equipo y maquinaría,Gobierno
16596	AAAA-5514	NULL	NULL	13	Equipo y maquinaria, Ingeniería industrial

Tabla 1.13: Datos de las revistas con identificadores 24355 y 16596 en la base de datos de BiDi.

Con los datos mostrados en las Tablas 1.12 y 1.13 no es posible determinar si ambos registros hacen referencia a la misma revista, por lo que se deben revisar en los datos proporcionados por los proveedores de las revistas (Ver Figura1.14). En este caso, se trata de elementos duplicados porque, además de tener el mismo proveedor, al probar los enlaces almacenados en la base de datos, éstos están redireccionados a la misma página.

id_rev	proveedor	url
16596	EBSCO	http://search.ebscohost.com/login.aspx?direct=true&db=buh&jid=YEQ
24355	EBSCO	http://search.ebscohost.com/direct.asp?db=buh&jid=YEQ&scope=site

Tabla 1.14: Datos de los proveedores de las revistas con identificadores 24355 y 16596 en la base de datos de BiDi.

Encontrar registros duplicados de revistas electrónicas en la base de datos de BiDi basándose en el título o en el ISSN no es sencillo, esto se debe principalmente a la falta de datos en la base de datos (muchos campos con valores nulos).

Para saber cuántas editoriales diferentes de revistas electrónicas se encuentran almacenadas en la base de datos, se presentan los mismos problemas que con los títulos de las revistas. En primer lugar, existen 9,935 registros que no tienen un valor en el campo `editor`, lo que equivale a un 39.7 % de los elementos en la base de datos (Ver Figura 1.13).

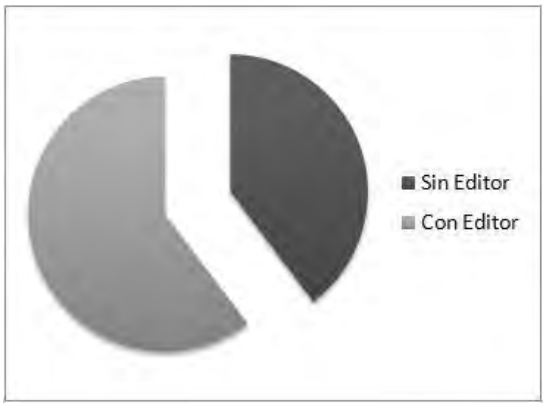


Figura 1.13: Gráfica circular en donde se muestra la proporción de revistas electrónicas que cuentan con valor en el campo `editor` y aquellas que no lo poseen.

La presencia de valores nulos afecta también al sistema de recuperación de información. A través del portal no es posible hacer el listado de las editoriales por orden alfabético ya que ni siquera es posible ver la lista de letras con las que empiezan los nombres de editor (Ver Figura 1.14).

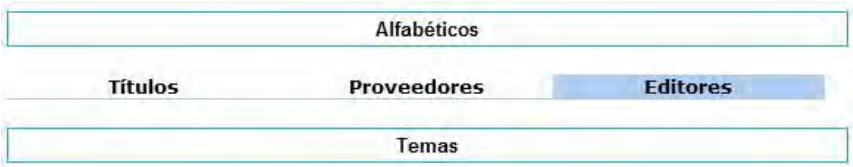


Figura 1.14: Vista del sistema de búsqueda del portal de BiDi. Listado alfabético por editorial.

En la base de datos existen 2,340 nombres de editorial diferentes; sin embargo, no se han contemplado los títulos que son semejantes pero no son iguales. En el Apéndice C se muestra un listado de los nombres de editoriales que son muy semejantes pero que no son iguales, así como el número de revistas que tienen estas editoriales.

1.4. Estandarización de valores

El problema principal encontrado en la base de datos de la DGB es la falta de estandarización de los datos, ya que existen sistemas de recuperación de información que funcionan correctamente; sin embargo, están diseñados para funcionar con datos sin errores, por lo que al corregir la falta de estandarización de los datos y eliminando los valores nulos, se esperaría que los sistemas funcionaran de manera adecuada.

Uno de los obstáculos que se deben superar es la pérdida de información durante la estandarización de datos, por lo que este proceso debe ser realizado por personal especializado. En caso contrario, se deben buscar alternativas para estandarizar los datos de campos como `editorial` y permitir que los sistemas de recuperación de información y obtención de estadísticas que utilizan los datos de esta base, obtengan información correcta y confiable.

Una propuesta de solución al problema de estandarización es la creación de diccionarios de manera supervisada, de modo que las correcciones se puedan llevar a cabo por un sistema de estandarización. Además se busca que, para hacer las actualizaciones de la base, se pueda hacer uso de los diccionarios creados de modo que las correcciones se hagan adecuadamente y no se genere pérdida de información.

Para lo anterior se espera que los diccionarios se encuentren disponibles y puedan ser extendidos conforme se obtengan nuevos datos.

1.5. Resumen

En este capítulo se ha presentado la base de datos de la Biblioteca Digital (BiDi) de la Dirección General de Bibliotecas de la UNAM. Esta base de datos proporciona información útil para toda la comunidad universitaria y externa, ya que es una fuente de información por medio de la cual es posible consultar y aprovechar los recursos que brinda la Universidad.

Se dedica una sección a lo que es una *biblioteca digital* debido a que existe mucha confusión sobre este concepto. Una biblioteca digital no se refiere única-

mente a la colección digitalizada de documentos y al uso de herramientas tecnológicas para administrar la información, sino que abarca el ciclo de la creación, diseminación, uso y preservación de los datos, la información y el conocimiento [2]. *Biblioteca Digital* (BiDi) es un área que pertenece a la Dirección General de Bibliotecas, fue fundada en mayo de 2001 y desde entonces tiene como misión ofrecer a la comunidad universitaria el acceso a diversos acervos de información en formato digital.

En este capítulo, se describe la estructura de la base de datos bibliográfica de BiDi, específicamente de la sección que contiene los datos de las revistas electrónicas que contrata la Universidad a distintos proveedores. Una *base de datos bibliográfica* es una base de datos en donde se guarda sólo la información fundamental para describir y permitir la localización de los documentos, ya sean impresos o electrónicos. Se describe el proceso de actualización de los datos, mencionando los problemas de integración de datos a los que se enfrenta el personal de BiDi que debe hacer las actualizaciones.

Por último se muestran algunas inconsistencias en los resultados de consultas realizadas a través del sistema de búsquedas del portal de BiDi. En donde se identifica que las inconsistencias son producto de la falta de estandarización de los datos en la base de datos bibliográfica.

Capítulo 2

Datos y Calidad de Datos

El concepto de *dato* no es fácil de definir, de hecho muchas definiciones coinciden en que *los datos generan información*; sin embargo, no toman en cuenta la estructura inherente a los datos. Para la realización de este trabajo se considera la definición de Redman [9]: Un *dato* o una *colección de datos* consiste en dos componentes interrelacionados: *modelo de datos* y *valores de los datos*.

El *modelo de datos* define el contexto al que pertenecen todos los datos. Generalmente los datos describen una *entidad* o algún objeto del mundo real como `PERSONA`, los atributos y las relaciones describen las características pertinentes de las entidades como `NOMBRE` y `FECHA_NACIMIENTO`. Los *valores de los datos* son asignados a los atributos en el modelo de datos para entidades específicas, por ejemplo: `NOMBRE=Ana María Pérez` es el valor de un dato para una persona en específico. [9]

Debido a que los datos son intangibles, éstos son susceptibles a varios errores. El error en los datos se puede definir como la inexactitud entre el valor del dato y el valor real del atributo de la entidad [6]; muchos errores se generan cuando se hace una integración de datos que provienen de diversas fuentes de información.

La exactitud de los datos para representar objetos del mundo real es un requerimiento fundamental para obtener información verídica mediante sistemas de información, razón por la cual la exactitud de los datos es un aspecto importante de la calidad de la información [16].

Los datos han incrementado su valor a lo largo de los años ya que las empresas los usan cada vez más como apoyo en la toma de decisiones importantes, las cuales pueden ser rutinarias o estratégicas. Cuando las empresas trabajan con datos erróneos es posible que tomen decisiones incorrectas, pierdan clientes y oportunidades y deban aumentar costos en actividades de corrección para cada uno de estos problemas. Muchas empresas aún desconocen que el incremento en costos depende en gran medida de la falta de calidad de los datos con los que trabajan sus sistemas de información [16] y concentran sus esfuerzos en corregir los errores en lugar de prevenirlos. Ver Figura 2.1.

Figura 2.1: Muchas empresas concentran sus esfuerzos en corregir errores en lugar de prevenirlos. Tomado de [29]

Para obtener información confiable a partir de los datos, es necesario que éstos tengan una buena calidad. Los datos son de alta calidad si pueden ser utilizados para las aplicaciones previstas, la toma de decisiones y el planeamiento; además se encuentran libres de defectos y poseen las características deseadas por los usuarios [9]. La calidad de los datos determina la capacidad de un sistema de información para reflejar aspectos de la vida real, por esta razón es que una buena calidad de datos permite la obtención de información confiable, útil para hacer una acertada toma de decisiones.

2.1. Calidad de datos

Calidad de datos es un término que no tiene una definición formal, de manera general se puede decir que la calidad de datos depende de las expectativas del usuario final. La definición establecida por Redman sugiere que la calidad de datos se puede obtener de comparar dos fuentes de datos: *Una colección de datos X es de mayor calidad que una colección de datos Y si X satisface mejor que Y las necesidades del usuario*. [7, 8]

Otra definición es la descrita por Olson [16]: Los datos son de alta calidad si éstos satisfacen los requerimientos para los que se pretenden utilizar. En otras palabras, la calidad de los datos depende de los datos y del uso previsto de los mismos. Para satisfacer el uso previsto, los datos deben ser exactos (*accurate*), oportunos (*timely*), relevantes (*relevant*), completos (*complete*), entendibles (*understood*) y confiables (*trusted*). [16]

2.1.1. Exactitud

Exactitud es el grado en el cual los datos almacenados en una base de datos contienen valores *aceptables* para satisfacer los requerimientos para los que se pretenden usar; es decir, depende directamente del número de errores que contienen los datos de un campo en particular, incluyendo datos que no pueden ser utilizados como los valores `null`.

En la base de datos bibliográfica de BiDi, el campo de los nombres de las editoriales de las revistas electrónicas presenta un grado de exactitud bajo ya que de los `25,024` registros almacenados en la base de datos analizada, `9,935` tienen un valor `null`. Con lo anterior se tiene que casi el 40 % de los nombres de las editoriales de las revistas presentan inexactitud, esto sin contar los datos con valores incorrectos debido a errores tipográficos que contienen.

2.1.2. Actualidad

Los datos contenidos en una base de datos son *actuales* cuando reflejan el estado real en el presente de los objetos representados en la BD.

La base de datos bibliográfica de la DGB debe ser actualizada constantemente debido a que la información referente a las revistas electrónicas cambia de manera periódica. Es necesario consultar los datos proporcionados por cada uno de los proveedores para actualizar los campos que hayan sufrido un cambio como el título de la revista o el nombre de los editores, etc.

Para la actualización de datos, BiDi se enfrenta a dos problemas principales debido a que la mayoría de los datos de las revistas electrónicas contratadas por la UNAM provienen de diferentes proveedores:

1. La integración de datos debido a que la información proporcionada por los proveedores de una misma revista no es consistente.

2. Obtener datos actualizados ya que no todos los proveedores han actualizado sus datos, lo que genera confusión al momento de que los especialistas deben determinar cuál es la información actual.

Dentro del proceso de actualización de la base de datos de BiDi, los especialistas revisan la información de algunos proveedores asignados, por lo que no revisan los datos proporcionados por los otros proveedores de las revistas que deben revisar. Esta es una razón por la que, en muchos casos, en lugar de actualizar la base de datos bibliográfica se tiende a desactualizar. Este es un problema externo que se deriva de la inconsistencia de los datos proporcionados por los diferentes proveedores.

Integración de datos

La integración de bases de datos es el proceso de extraer y combinar datos de múltiples fuentes para formar una nueva fuente de información. Resolver las diferencias estructurales, sintácticas y semánticas entre las fuentes ha sido un problema complejo de integración de datos por muchos años [7].

Una solución sencilla es la de crear un nuevo esquema conceptual para una base de datos que represente la información integrada, utilizando mapeos de las diferentes fuentes de datos a los campos del nuevo esquema. Esta es la solución que adoptó la Biblioteca Digital para almacenar la información de las revistas de sus diferentes proveedores, de manera que los datos que la mayoría de las fuentes comparten se encuentran almacenados en la base de datos bibliográfica de BiDi.

2.1.3. Relevancia

Para determinar si los datos son relevantes no es posible separar los datos del uso que se le pretende dar a los mismos [16]. La base de datos bibliográfica de BiDi es una base de consulta que permite a la comunidad universitaria conocer las publicaciones contratadas por la Universidad a través de diferentes sistemas de información.

Relevancia suele referirse al campo de una entidad que será almacenado en la base de datos. Por ejemplo, el campo `editorial` de las revistas electrónicas es *relevante* para los objetivos de la base de datos bibliográfica de BiDi, por lo que es un campo que se consideró durante el diseño del esquema conceptual de la misma.

Para el diseño del esquema relacional de la base de datos bibliográfica de la DGB, únicamente se consideró información mediante la cual es posible realizar consultas de los datos de las diferentes revistas electrónicas, incluyendo campos que son de uso exclusivo de la DGB como *área temática*. Los valores de los datos son el resultado de la integración de la información que proporcionan los diferentes proveedores excluyendo aquella que no es necesario almacenar o no fue considerada relevante y fue proporcionada por algunos proveedores como el número de páginas.

2.1.4. Completitud

Un esquema de bases de datos está completo cuando representa todas las características relevantes de un problema. En el caso de una base de datos es necesario saber el uso que se le dará a los datos para determinar el esquema conceptual de la BD y garantizar que todos los requerimientos de los sistemas de información que utilicen los datos se pueden llevar a cabo.

En el caso de la base de datos bibliográfica de BiDi, aunque el esquema conceptual es completo (ya que considera el *nombre de la editorial* para cada una de las revistas electrónicas contratadas por la UNAM), la presencia de valores `null` hace que los sistemas de información excluyan esos datos, alterando probablemente los resultados que se deberían obtener al hacer una operación como, por ejemplo, el agrupamiento de revistas por editorial.

2.1.5. Entendible

Esta característica, al igual que las anteriores, depende del uso que se le pretende dar a los datos. Los valores de los datos deben tener un significado para que sean utilizados por los sistemas de información. Por ejemplo, la base bibliográfica de BiDi mantiene un campo para almacenar las áreas temáticas a las que pertenece una revista electrónica, este campo tiene el nombre "`área temática`" y está representado por una cadena de caracteres, en la cual cada posición está ocupada por un dígito porque durante el diseño del esquema conceptual a cada área temática se le asignó un valor numérico de la siguiente manera:

1. Ciencias físico-matemáticas e ingenierías.

2. Ciencias biológicas y de la salud.

3. Ciencias sociales.

4. Humanidades y artes

En la tabla 2.1 se muestra cómo están almacenados unos registros de revistas electrónicas en la base de datos de BiDi. Como se puede ver, es necesario tener un conocimiento del diseño conceptual de la base de datos para determinar el área temática a la que pertenecen las revistas electrónicas que se muestran, de manera que los datos puedan ser utilizados por los sistemas de información para presentar información entendible a los usuarios.

ISSN	**Título**	**Area**
AAAA-6121	1 800 FLOWERS COM Inc SWOT Analysis	23
0149-1210	33 Metalproducing	1
0365-0855	Abstracts of the Papers Communicated to the Royal Society of London	13
0965-2140	Addiction.	23
1130-2887	América Latina Hoy	1234

Tabla 2.1: El campo área temática sólo es entendible por los diseñadores de la BD.

2.1.6. Confiabilidad

Los datos de una BD son *confiables* cuando la información que se obtiene a partir de ellos refleja aspectos del mundo real, por lo que esta propiedad depende mucho de la actualidad de los datos.

2.2. Evaluación de la Calidad de Datos

Es posible hacer una clasificación de la calidad de datos que comprenda el modelo de datos y el valor de los mismos, pero esto depende de las perspectivas y las necesidades del usuario final. Para medir la calidad de los datos es necesario definir los campos que se van a evaluar, clasificarlos, promediarlos y determinar con esto la calidad total de una colección de datos.

Los resultados obtenidos en pequeños estudios con datos de calidad pueden resultar mucho más interesantes y concluyentes que los obtenidos en grandes estudios con datos sin control de calidad o cuestionables en lo que se refiere a este aspecto. [6]

Para poder determinar la calidad de una colección de datos es necesario conocer las necesidades de los usuarios, ya que serán ellos los que al final evalúen la calidad de los datos [9]. Para esto, es necesario hacer un análisis adecuado de requerimientos entre los usuarios para que el modelo conceptual de datos permita que se cumpla con las características de calidad definidas anteriormente.

Cuando se trata de evaluar la calidad de datos que ya se encuentran almacenados en una BD, se pueden encontrar dos tipos de registros: de tipo *perfecto* y los que son simplemente *usables* [9]. Como su nombre lo indica, los registros perfectos son aquellos que sirven para realizar todas las operaciones del sistema, los registros usables sólo se pueden usar en algunas tareas. Ver Figura 2.2.

En la tabla que se muestra en la Figura 2.2, se pueden identificar diferentes tipos de registros en una BD que almacena datos de las ventas de playeras de una tienda de ropa. Los registros *no usables* son aquellos que no se pueden utilizar para obtener información. Con los registros *usables* es posible obtener información no necesariamente confiable . Lo ideal sería que todos los registros en la base de datos

sean perfectos para que la información obtenida a partir de éstos sea confiable. Para que un registro sea perfecto debe ser usable.

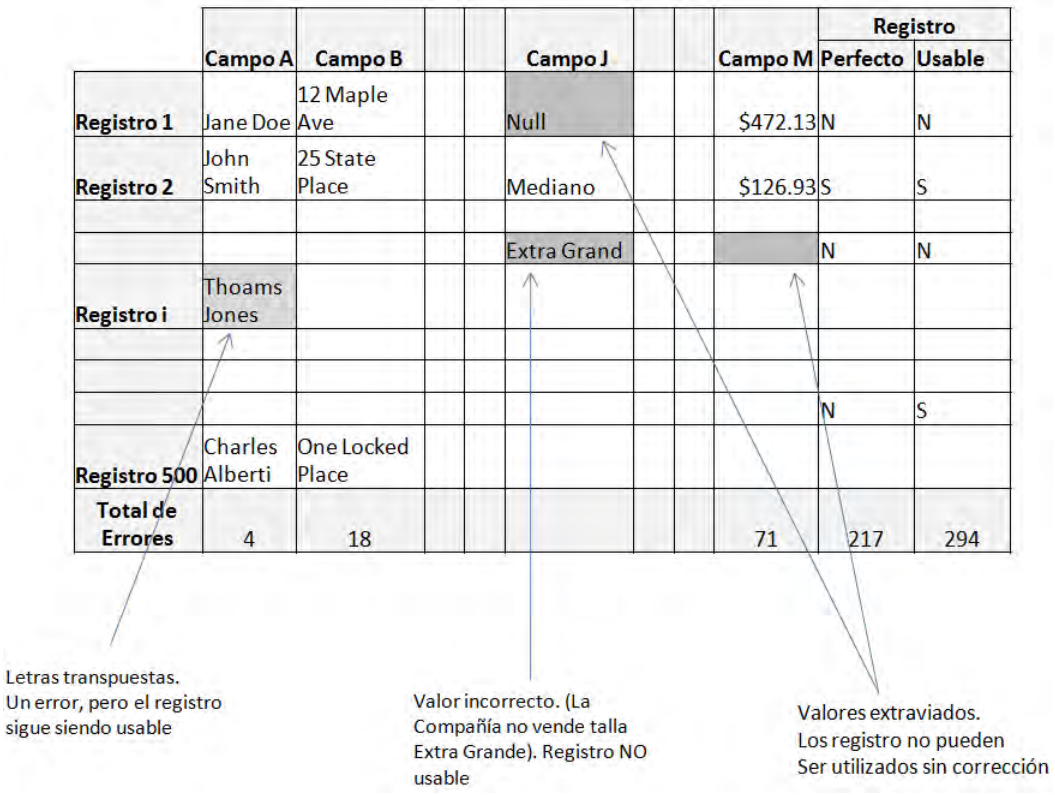


Figura 2.2: Ejemplo de una tabla para estimar errores. Tomado de [9]

Para garantizar la buena calidad de los datos almacenados en una base de datos, es importante validar los datos antes de ser ingresados. De esta manera se evita que en la base existan datos dudosos que afecten la información que se obtiene de ellos. Como se mencionó en la introducción de esta tesis, la idea es crear un *firewall* entre los datos a ingresar y la base de datos, de manera que sean ingresados únicamente si cumplen con parámetros de calidad establecidos.

Cuando en la base de datos ya se encuentran datos almacenados es necesario hacer un análisis del estado actual de la base de datos y, si estos no cumplen con una buena calidad, se debe hacer una *limpieza de datos* para eliminar errores en los mismos.

2.3. Limpieza de datos

Como se mencionó anteriormente, cuando se tiene un conjunto de datos es muy probable que éstos contengan algún tipo de error, ya sea simple o complejo. La solución lógica para resolver este problema es buscar algún método para hacer una limpieza de los datos. Esto es: analizar el conjunto de datos y localizar los errores para ser corregidos posteriormente [26].

El objetivo original de la limpieza de datos fue eliminar los elementos duplicados de una colección de datos, problema que aparece comúnmente en bases de datos que han integrado datos de otras fuentes. La limpieza de datos se ha convertido en una parte integral del proceso de integración de datos [32] pretendiendo además la eliminación de inconsistencias y errores de los datos [26].

La *limpieza de datos* se incluye como una parte de procesos de *data warehouse* o *data mining* por ejemplo. La limpieza de datos es un conjunto de técnicas que se deben aplicar en colecciones de datos susceptibles a errores, ya sea donde se ha realizado previamente una integración de datos o exista una alta probabilidad de ingresar datos con errores como los tipográficos. En estos casos muchos registros pueden hacer referencia a una misma entidad con un formato diferente o pueden estar representados de manera errónea [26]. Los registros duplicados aparecen comúnmente en este tipo de bases de datos. Este problema se conoce como *integración/limpieza* de datos [7].

La limpieza de datos no se puede realizar de manera automatizada sin la asesoría de un experto porque la detección y corrección de inconsistencias requiere de conocimiento especializado, pero aún así se debe automatizar el mayor número de procesos involucrados ya que de otro modo sería imposible mantener la base de datos actualizada. Además, la capacidad de limpieza de datos está limitada por la información disponible que es necesaria para detectar y corregir anomalías de los datos. [32].

Con lo anterior se puede deducir que entre más grande sea una colección de datos, mayor información se puede obtener de la misma; sin embargo, es necesario que los cambios realizados al hacer la limpieza de los datos permitan que éstos cumplan con las propiedades de calidad de datos para que la información que se obtenga de ellos sea confiable.

Actualmente muchas compañías ofrecen herramientas para la limpieza de datos como *Harte-Hanks Data Technologies*, *Innovate Systems Inc.* y *Vality Technololy* las cuales se han centrado en la limpieza de campos de direcciones postales de listas de clientes de diversas compañías [26].

Dado que los datos generan información, se requiere que los datos tengan una calidad adecuada para poder ser utilizados ya que muchos sistemas de información consideran que los datos son correctos por lo que la información que se obtiene a partir de ellos puede ser errónea sin que el usuario final pueda percatarse de este problema.

El proceso de limpieza de datos se define por 3 fases [26]:

1. Definir y determinar los tipos de error posible.
2. Buscar errores.
3. Corregir errores.

Cada una de estas fases constituye por sí sola un problema que puede resolverse utilizando una gran variedad de métodos y tecnologías. Aunque muchos errores que se generan en la integración de datos son tipográficos, existen también errores que involucran las relaciones entre los diferentes campos que son difíciles de corregir.

2.3.1. Detección de errores

En una base de datos es posible encontrar datos con errores que pueden dificultar el acceso a los mismos a través de consultas así como la comprensión de éstos por los usuarios finales. Por ejemplo, en la Tabla 2.2 se muestran algunos datos que se encuentran almacenados en el campo *editorial* de las revistas electrónicas en la base bibliográfica de BiDi.

Nombre Editorial
Blackwell Publishing
Blackwell Publishing Limited
Blackwell Publishing Ltd

Tabla 2.2: Nombres de editoriales de revistas electrónicas almacenados en la base de datos bibliográfica de BiDi.

En la Tabla 2.2, es claro que todos los valores almacenados hacen referencia a la misma editorial; sin embargo, es posible que las inconsistencias entre las cadenas se deban a inconsistencias entre la información que proporcionan los proveedores de las revistas.

Para detectar errores en una colección de datos se utilizan diversos métodos, algunos de los cuales se explican de manera breve en las siguientes secciones. Es importante señalar que los métodos más usados para la detección de errores se basan en análisis estadísticos.

Método de análisis estadístico

Para detectar errores por este método se debe analizar la distribución de los valores de los datos en un campo determinado. Para datos numéricos, es posible identificar cómo se distribuyen los valores de los datos en un campo en particular desde el valor más pequeño hasta el valor más grande. Una vez obtenida la distribución se pueden identificar valores atípicos o que se encuentran fuera de algunos rangos como: media, desviación estándar o considerando valores especiales para cada campo.

En el caso de datos no numéricos como las cadenas de caracteres, este tipo de análisis se puede hacer mediante un conteo de los valores dentro de la base de datos para establecer los rangos mediante el número de ocurrencias de cada valor.

Clustering

Se identifican los registros que se encuentran fuera de un rango haciendo *clusters* basados en la distancia euclidiana. Existen algoritmos que permiten identificar estos valores que se encuentran fuera de rango [26]

Basados en patrones

Se identifican los registros que no concuerdan con algún patrón de valor insertado en la Base de Datos. Un patrón está definido por un grupo de registros que poseen características similares para un porcentaje p, donde p es un valor definido por el usuario [26].

Muchas técnicas se combinan (dividir, clasificar o hacer *clusters*) para determinar los patrones de un campo en particular.

Reglas de asociación

Las reglas de asociación se definen mediante patrones, los registros que no cumplan con las reglas definidas se consideran como posibles candidatos a ser registros con errores [26]. El uso de reglas de asociación es muy semejante a la detección de errores basada en patrones.

2.3.2. Detección de duplicados

En los campos de las tablas de una base de datos se pueden encontrar elementos duplicados que no son exactamente iguales pero que hacen referencia al mismo elemento. En el caso de las cadenas de caracteres estas inconsistencias en los datos pueden ser el resultado de distintos factores, por ejemplo:

1. Algunos de los datos inconsistentes contienen errores tipográficos.

2. Los datos inconsistentes provienen de diversas fuentes, en donde no existe un estándar para la representación de los mismos.

Para identificar los elementos que se encuentran representados con diferentes cadenas de caracteres en la base de datos, se debe elegir una técnica a seguir después de analizar el problema a resolver. En el caso de la base de datos bibliográfica de BiDi, el campo *nombre de editorial* contiene datos inconsistentes como los mostrados en la Tabla 2.3, en donde a pesar de las diferencias entre cada una de las cadenas, éstas son semejantes entre sí.

Nombre Editorial
Addison Wesley
Addison-Wesle
Addison-Wesley
Addison-Wesley.

Tabla 2.3: Algunas inconsistencias en los nombres de las editoriales de los libros de BiDi.

Para estandarizar los valores de los datos de las editoriales almacenados en la base de datos, lo primero es identificar los elementos duplicados. Debido a que los elementos duplicados no exactos están representados por diferentes cadenas de caracteres que son muy semejantes entre sí, el primer paso es encontrar y agrupar los duplicados no exactos buscando la correspondencia entre cadenas.

2.4. Correspondencia de cadenas

El problema de definir una medida que describa la relación que existe entre dos cadenas de caracteres tiene múltiples aplicaciones. Actualmente existen diversos métodos para establecer una medida de diferencia o de *distancia* entre dos cadenas de caracteres; sin embargo, en ocasiones puede ser más útil obtener una medida de *similitud*, en este último caso se utilizan algoritmos de *alineamiento* [28].

Un alineamiento entre dos cadenas de caracteres x y y se obtiene al insertar espacios arbitrariamente en las cadenas de manera que ambas terminen con el mismo número de elementos. Una vez insertados los espacios, es posible establecer una correspondencia directa entre los elementos de x y los de y. Los algoritmos que existen para obtener un alineamiento de cadenas pueden ser adaptados para hacer una transformación y viceversa [28], por lo que es posible obtener medidas

de distancia y de similitud basándose en estos algoritmos.

La *distancia de edición* es una medida común que se obtiene al aplicar una serie de operaciones de edición sobre una cadena de caracteres x para transformarla (editarla) en otra cadena de caracteres y [28], el valor numérico que se obtiene es el número de operaciones de edición básicas que se necesitan para transformar una cadena en la otra [12]. Estas operaciones son las unidades que definen las medidas de similitud ya que se les asigna un costo. El objetivo es minimizar el costo total de convertir una cadena en otra.

Las operaciones de edición básicas que se utilizan para tranformar una cadena de caracteres en otra, permite obtener indicadores de correspondencia entre cadenas. Las operaciones posibles son:

- **Inserción**. Consiste en agregar un caracter en una determinada posición de una cadena. Por ejemplo: sean las cadenas $x = mes$ y $y = mesa$, se utiliza la inserción $Inserta(x,' a', 4)$ para que $x = y$.

- **Eliminación**. Consiste en quitar un caracter determinado de una cadena. Por ejemplo: sean las cadenas $x = Bibbliobytes$ y $y = Bibliobytes$, se utiliza la eliminación en la cadena x para convertirla en la cadena y. De este modo $Elimina(x, 3) = y$.

- **Sustitución**. Consiste en cambiar un caracter por otro diferente en una posición determinada de una cadena. Por ejemplo: sean las cadenas $x = AddisonWesley$ y $y = AddisonWezley$, hay que hacer una sustitución para que $x = y$ de la forma $Sustituye(y, 11, s) = x$.

- **Transposición**. Consiste en intercambiar un caracter con el siguiente o el anterior, existe también la operación en la cual se transpone todo un bloque de caracteres.

Cuando se desea obtener un indicador de diferencia entre dos cadenas de caracteres, generalmente se denomina *medida de distancia* entre cadenas. Matemáticamente, una medida de distancia d es una función que satisface las siguientes condiciones [27]:

A 3.1 $d(x, y) \geq 0 \forall x, y$

A 3.2 $d(x, y) = 0 \iff x = y$

A 3.3 $d(x, y) = d(y, x) \forall x, y$

A 3.4 $d(x, z) \leq d(x, y) + d(y, z) \forall x, y, z$

Los métodos para obtener una correspondencia entre cadenas se dividen en algoritmos de distancia y algoritmos de similitud. Las medidas que se obtienen a partir de los algoritmos para el cálculo de distancia, satisfacen las condiciones anteriores (A 3.1 - A 3.4); sin embargo, no necesariamente se cumplen estas condiciones en el cálculo de una medida de similitud entre cadenas, un ejemplo es la obtención de la medida de similitud local, la cual se describirá más adelante.

Aunque existen muchas definiciones de distancia entre cadenas, una de las más estudiadas es la distancia de edición simple (distancia de Levenshtein) [23]. Existen otras definiciones de *distancia* más complejas que son utilizadas en otros campos, principalmente en biología molecular; sin embargo, la medida de distancia más popular y utilizada por diversos algoritmos para obtener la distancia entre cadenas de caracteres están basados en la *distancia de Levenshtein* [30, 31].

Una variante de la distancia de edición simple es la *distancia de Hamming*, que únicamente regresa el número de sustituciones que se le deben hacer a una cadena de caracteres x para convertila en una cadena y, si x y y tienen el mismo número de caracteres [31].

2.4.1. Distancia de Hamming

Para dos cadenas de caracteres de la misma longitud[1] x y y, la distancia de *Hamming* (1982) se define como el número de caracteres en donde $x_i \neq y_i$. Esto es equivalente al mínimo costo para transformar x en y utilizando únicamente sustituciones a las que se les haya asignado un peso, de manera que se pueda obtener un valor numérico [27]. Por ejemplo, sean las cadenas $x = casa$ y $y = caza$, la distancia de Hamming se define como $DH(x, y) = 1$, ya que basta con hacer la sustitución $Sustituye(x, 3, z)$ para que $x = y$.

[1]Se define como la *longitud de una cadena* el número de caracteres que la conforman, la longitud de la cadena x se representa como $|x|$.

2.4.2. Distancia de edición simple (Levenshtein)

La distancia de edición simple es el mínimo número de operaciones requeridas para transformar una cadena de caracteres en otra, en donde es posible utilizar operaciones de edición como inserción, eliminación o sustitución. La distancia de edición simple se conoce también como *distancia de Levenshtein*, ya que fue el científico ruso Vladimir Levenshtein quien ideó el algoritmo en 1965 [12].

Como se puede notar, debido a las operaciones permitidas para el cálculo de la distancia de edición simple, es posible obtener un valor al comparar cadenas de diferente o de igual longitud. Como es una medida de distancia, se sabe que $DL(x, y) = 0$ sólo si $x = y$, además el valor resultado está acotado por: $0 \leq DL(x, y) \leq max(|x|, |y|)$.

No es posible obtener la distancia de edición simple de manera recursiva utilizando directamemente las operaciones de edición, ya que el tiempo de ejecución crece de manera exponencial. Debido a lo anterior, para hacer el cálculo de la distancia de Levenshtein se utiliza la programación dinámica.

La programación dinámica consiste en resolver un problema dividiéndolo en subproblemas independientes, los cuales se resuelven de manera recursiva para combinar finalmente las soluciones y así resolver el problema original. El algoritmo para el cálculo de la distancia de Levenshtein se basa en el llenado de una matriz D en donde se registran los resultados de las distancias de cada prefijo de una cadena x y cada prefijo de una cadena y [12].

Las columnas de la matriz de distancias D corresponden a cada uno de los caracteres de la cadena x y los renglones de la matriz D corresponden a los caracteres de la cadena y. Los elementos de la matriz $D_{i,j}$ corresponden a los costos de transformar la secuencia $x_{1...i}$ en la secuencia $y_{1...j}$.

La idea básica del algoritmo de programación dinámica para obtener la distancia edición simple, establece que el costo para llegar a la posición $D_{i,j}$ dentro de la matriz de costos se puede calcular con base en el costo de haber llegado a posiciones anteriores, las cuales debieron haber sido calculadas previamente. El algoritmo para el llenado de la matriz de distancias se muestra en el Algoritmo 1.

Algoritmo 1 Distancia de edición simple (Levenshtein)

Entrada: x, y
Salida: Matriz de distancias

```
for i ← 0 to |x| do
    D_{0,i} ← 0
end for
for j ← 0 to |y| do
    D_{i,0} ← i
end for
for i ← 1 to |y| do
    for j ← 1 to |x| do
        if x_i = y_i then
            D_{i,j} ← D_{i-1,j-1}
        else
            D_{i,j} = 1 + min(D_{i-1,j}, D_{i,j-1}, D_{i-1,j-1})
        end if
    end for
end for
```

Se puede notar que obtener el valor $D_{i,j}$ toma tiempo constante. Si $|x| = m$ y $|y| = n$, el tiempo de ejecución es $O(mn)$ ya que se deben hacer mn cálculos. Un ejemplo del llenado de una matriz de distancias con este procedimiento se muestra en la Figura 2.3.

		S	U	R	G	E	R	Y
	0	1	2	3	4	5	6	7
S	1	0	1	2	3	4	5	6
U	2	1	0	1	2	3	4	5
R	3	2	1	0	1	2	3	4
V	4	3	2	1	1	2	3	4
E	5	4	3	2	2	1	2	3
Y	6	5	4	3	3	2	2	2

Figura 2.3: Ejemplo de llenado de una matriz de distancias mediante el cálculo de la distancia de edición simple.

2.4.3. Distancia de edición por bloques

El algoritmo para el cálculo de la distancia de Levenshtein puede ser generalizado para considerar secuencias de palabras en lugar de cadenas de caracteres, de modo que se comparen cada una de las palabras que constituyen la secuencia. Por lo que se pueden considerar a las palabras como unidades indivisibles en un enunciado [24]. En este caso, la notación para denotar la longitud de un enunciado y se representa con $|y|$. Para obtener la distancia de edición por bloques se necesita de un algoritmo de extracción de *tokens* que en la mayoría de los casos propone algoritmos NP-Completos.

En este caso las operaciones comunes como inserción, borrado o sustitución se hacen sobre palabras y no sobre caracteres. En el trabajo realizado por Gabriel Castillo Hernández [24] se utiliza este tipo de distancia para obtener un diccionario terminológico basado en definiciones.

En la Figura 2.4 se muestra el llenado de una matriz de distancias para dos definiciones de caída libre.

	?	caída	libre	movimiento	de	un	cuerpo	en	un	campo	gravitatorio	bajo	la	influencia	de	la	gravedad
?	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
caída	1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
libre	2	1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
descenso	3	2	1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
de	4	3	2	2	1	2	3	4	5	6	7	8	9	10	11	12	13
un	5	4	3	3	2	1	2	3	4	5	6	7	8	9	10	11	12
cuerpo	6	5	4	4	3	2	1	2	3	4	5	6	7	8	9	10	11
sometido	7	6	5	5	4	3	2	2	3	4	5	6	7	8	9	10	11
únicamente	8	7	6	6	5	4	3	3	3	4	5	6	7	8	9	10	11
a	9	8	7	7	6	5	4	4	4	4	5	6	7	8	9	10	11
la	10	9	8	8	7	6	5	5	5	5	5	6	6	7	8	9	10
acción	11	10	9	9	8	7	6	6	6	6	6	6	7	7	8	9	10
de	12	11	10	10	9	8	7	7	7	7	7	7	7	8	7	8	9
la	13	12	11	11	10	9	8	8	8	8	8	8	7	8	8	7	8
gravedad	14	13	12	12	11	10	9	9	9	9	9	9	8	8	9	8	7

Figura 2.4: Ejemplo de llenado de una matriz de distancias mediante el cálculo de la distancia de edición por bloques. Tomado de [24].

A partir de esta alineación, se identifican los pares de palabras que son candidatos a formar parte de un mismo grupo semántico. En el área de recuperación de información, se denomina agrupamiento semántico a un conjunto de palabras semánticamente relacionadas [24].

Al igual que con las cadenas de caracteres, se busca obtener el mínimo costo total de las operaciones aplicadas para convertir una cadena en otra, empleando el algoritmo de distancia de edición por bloques. Las palabras deben ser consideradas como unidades indivisibles en este caso.

2.4.4. Distancia de edición general

La distancia de edición general o distancia de edición ponderada es una generalización de la distancia de edición simple. Como se mencionó anteriormente, la distancia de edición simple es el mínimo número de operaciones requeridas para transformar una cadena de caracteres en otra, en donde es posible utilizar operaciones de edición como inserción, eliminación o sustitución. En el caso del cálculo de la distancia de la edición general, a cada una de las operaciones de edición que se deben aplicar para transformar las cadenas de caracteres se les asigna un costo.

Supóngase que se quiere obtener la distancia de edición general de las cadenas de caracteres x y y, denotada por $DEG(x, y)$, en donde $x = x_1, x_2, \ldots, x_m$ y $y = y_1, y_2, \ldots, y_n$. Sea $DEG_{i,j}$ la distancia de edición general entre los prefijos $x_1, x_2, \ldots, x_i$ y $y_1, y_2 \ldots, y_j$. Se definen los costos de la siguiente manera:

- $w(x_i, y_j)$ el costo de sustituir x_i por y_j si $x_i \neq y_j$.
- $w(x_i, \epsilon)$ el costo de eliminar el caracter x_i.
- $w(\epsilon, y_j)$ el costo de insertar el caracter y_j en una cadena de caracteres.

En la distancia de edición simple (Levenshtein) se asume que todos los costos mencionados anteriormente tienen un valor unitario excepto cuando $x_i = y_j$, ya que en este caso $w(x_i, y_j) = 0$ [30].

Como se trata de una medida de distancia entre cadenas de caracteres, una buena correspondencia entre caracteres debe tener un costo de 0 ya que no agrega valor a la distancia obtenida entre las cadenas comparadas. Generalmente no se

hace distinción entre las operaciones de insertar o borrar un caracter, por lo que $w(x_i, \epsilon) = w(\epsilon, y_j)$ y se denota con W.

En la Figura2.5 se muestra un ejemplo de llenado de la matriz de distancias mediante el cálculo de la distancia de edición general entre dos cadenas cuyo costo mínimo de operaciones de edición es 7, considerando los costos: $W = 4$ y $w(x_i, y_j) = 3$.

		S	U	R	G	E	R	Y
	0	1	2	3	4	5	6	7
0	0	4	8	12	16	20	24	28
S 1	4	0	4	8	12	16	20	24
U 2	8	4	0	4	8	12	16	20
R 3	12	8	4	0	4	8	12	16
V 4	16	12	8	4	3	7	11	15
E 5	20	16	12	8	7	3	7	11
Y 6	24	20	16	12	11	7	6	7

Figura 2.5: Ejemplo de llenado de una matriz de distancias mediante el cálculo de la distancia de edición general.

Al igual que la distancia de Levenshtein si $|x| = m$ y $|y| = n$, la distancia de edición general puede obtenerse en un tiempo $O(mn)$ y $D_{i,j}$ representa el costo mínimo total de operaciones de edición aplicadas entre las cadenas $x_{1...i}$ y $y_{1...j}$. Es importante notar que las operaciones de edición simple utilizadas deben tener un costo mayor a 0 para que la distancia obtenida tenga sentido.

El Algoritmo 2 ejemplifica el cálculo de la distancia de edición general para dos cadenas de caracteres.

Algoritmo 2 Distancia de edición general

Entrada: x y y
Salida: Matriz de distancias

```
g ← w(x_i, ε) = w(ε, y_j)
for i ← 1 to |x| do
    D_{0,i} ← i * g
end for
for j ← 1 to |y| do
    D_{j,0} ← j * g
end for
for i ← 1 to |x| do
    for j ← 1 to |y| do
        costoMatch ← D_{i-1,j-1} + w(x_i, y_j)
        costoInsertar ← D_{i,j-1} + g
        costoBorrar ← D_{i-1,j} + g
        D_{i,j} ← min(costoMatch, costoInsertar, costoBorrar)
    end for
end for
return D_{|x|,|y|}
```

2.4.5. Similitud global entre cadenas

Al obtener una medida de similitud entre dos cadenas de caracteres x y y, se pretende encontrar la mejor correspondencia de caracteres entre los caracteres de x y los caracteres de y, teniendo en cuenta que a un mejor alineamiento corresponderá una mayor medida de similitud.

Una medida de similitud entre cadenas de caracteres, al contrario de una medida de distancia, es 0 si no existe correspondencia alguna entre los caracteres comparados. Los algoritmos para obtener una medida de similitud, al igual que los algoritmos de distancia, se resuelven basándose en el llenado de una matriz de costos mediante programación dinámica.

La asignación de costos para las correspondencias, se definen con la función w la cual se describe de la siguiente manera:

- $w(x_i, y_j)$ el costo de sustituir x_i por y_j si $x_i \neq y_j$

- $w(x_i, \epsilon)$ el costo de eliminar el caracter x_i
- $w(\epsilon, y_j)$ el costo de insertar el caracter y_j en una cadena de caracteres

Una medida de similitud debe ser mayor cuando existe alguna correspondencia de caracteres entre las cadenas comparadas, por lo que se debe considerar un costo positivo M para cuando $x_i = y_j$. En este caso no hay distinción entre agregar o eliminar un caracter, por lo que $w(x_i, \epsilon) = w(\epsilon, y_j)$ y se denota con W. Además $W < 0$ ya que debe afectar de manera negativa en la medida calculada.

En la Figura 2.6 se muestra el llenado de una matriz de costos basándose en el cálculo de la similitud global, considerando $W = -2$, $w(x_i, y_i) = -1$ y $M = 1$.

		G	A	T	C	A	G	G	A	A	C	T	G	A	G
	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22	-24	-26	-28
G	-2	1	-1	-3	-5	-7	-9	-11	-13	-15	-17	-19	-21	-23	-25
A	-4	-1	2	0	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	-22
G	-6	-3	0	1	-1	-3	-3	-5	-7	-9	-11	-13	-15	-17	-19
C	-8	-5	-2	-1	2	0	-2	-4	-6	-8	-8	-10	-12	-14	-16
G	-10	-7	-4	-3	0	1	1	-1	-3	-5	-7	-9	-9	-11	-13
G	-12	-9	-6	-5	-2	-1	2	2	0	-2	-4	-6	-8	-10	-10
A	-14	-11	-8	-7	-4	-1	0	1	3	1	-1	-3	-5	-7	-9
A	-16	-13	-10	-9	-6	-3	-2	-1	2	4	2	0	-2	-4	-6
T	-18	-15	-12	-9	-8	-5	-4	-3	0	2	3	3	1	-1	-3
T	-20	-17	-14	-11	-10	-7	-6	-5	-2	0	1	4	2	0	-2
G	-22	-19	-16	-13	-12	-9	-6	-5	-4	-2	-1	2	5	3	1
C	-24	-21	-18	-15	-12	-11	-8	-7	-6	-4	-1	0	3	4	2
A	-26	-23	-20	-17	-14	-11	-10	-9	-6	-5	-3	-2	1	4	3
C	-28	-25	-22	-19	-16	-13	-12	-11	-8	-7	-4	-4	-1	2	3

Figura 2.6: Ejemplo de llenado de una matriz de alineamiento global

Aunque parece que los pesos asignados a cada una de las operaciones que es posible realizar durante el alineamiento son arbitrarios, los resultados que arroja la obtención de esta medida demuestran que no es así. El Algoritmo 3 muestra los pasos a seguir para obtener el cálculo de la medida de similitud global entre dos cadenas de caracteres.

Algoritmo 3 Cálculo de similitud global entre cadenas (alineamiento global)

Entrada: x y y
Salida: Matriz de distancias

```
g ← w(x_i, ε) = w(ε, y_i)
for i ← 0 to |x| do
    D_{i,0} ← i * g
end for
for j ← 0 to |y| do
    D_{0,j} ← j * g
end for
for i ← 1 to |x| do
    for j ← 1 to |y| do
        costoMatch ← D_{i-1,j-1} + t(i, j)
        costoInsertar ← D_{i,j-1} + g
        costoBorrar ← D_{i-1,j} + g
        D_{i,j} ← max(costoMatch, costoInsertar, costoBorrar)
    end for
end for
return D_{|m|,|n|}
```

En donde se tiene, para las cadenas de caracteres $x = x_1, x_1, \ldots, x_m$ y $y = y_1, y_2, \ldots, y_n$ la función t:

$$t(i,j) \begin{cases} w(x_i, y_j) & \text{si } x_i \neq y_j \\ M & \text{si } x_i = y_j \end{cases}$$

Como se puede notar, el algoritmo es semejante al algoritmo para obtener la distancia de edición general; sin embargo, como el Algoritmo 3 pretende obtener una medida de similitud utiliza la función max en lugar de min. El algoritmo toma un tiempo de ejecución acotado por $O(mn)$ si $|x| = m$ y $|y| = n$.

2.4.6. Similitud local entre cadenas (Smith-Waterman)

Dos cadenas de caracteres x y y pueden ser muy diferentes entre ellas pero contener regiones de alta similitud, lo cual puede llegar a ser más importante en algunas aplicaciones. El reto es encontrar un par de regiones, una por cada cadena de caracteres a comparar, en las cuales exista un alto grado de similitud entre ellas [28]. La medida que se obtiene es llamada *medida de similitud local*.

Es importante señalar que el concepto de similitud local entre cadenas queda definido exclusivamente en términos de medida de similitud debido a que se busca maximizar un valor de semejanza, contrario a los objetivos de las medidas de distancia en donde se trata de minimar.

En muchas aplicaciones biológicas, una medida de similitud local puede ser más útil que una medida de similitud global, particularmente cuando se pretende hallar relaciones evolutivas entre diferentes proteínas y series de ADN o ARN en donde sólo algunas secciones de las cadenas contienen información útil para este propósito [28].

De manera formal, para obtener una medida de similitud local entre dos cadenas de caracteres x y y se pretende encontrar las subcadenas $x' \in x$ y $y' \in y$, cuya similitud entre sí sea el máximo que puede tener cualquier subcadena de x y de y.

Un alineamiento global entre dos cadenas de caracteres se puede ver influenciado por regiones de alta similitud; sin embargo, es posible que el mejor alineamiento (el que devuelve la medida más alta de similitud global) no considere las regiones de alta similitud. En biología molecular, un buen alineamiento entre cadenas debe respetar en mayor medida regiones completas, sin separar cada uno de los caracteres, razón por la cual, para identificar las regiones de alta similitud es más efectivo buscarlas explícitamente [28].

La obtención de la medida de similitud local, al igual que la medida de similitud global, se basa en el llenado de una matriz de costos. De manera general, si $|x| = m$ y $|y| = n$, encontrar todas las subcadenas de x y y toma $O(m^2)$ y $O(n^2)$ respectivamente. Por lo tanto, encontrar las subcadenas de x y y se encuentra acotado por $\Theta(m^2n^2)$, ésto sin obtener los alineamientos entre las subsecuencias, por lo que el cálculo del alineamiento local óptimo se lograría en $O(m^3n^3)$.

Temple Smith y Michael Waterman (1981) lograron una mejora significativa en la obtención del alineamiento local óptimo que toma $O(mn)$, restringiendo el esquema de puntuación al asumir que el alineamiento global óptimo entre dos cadenas vacías es 0. De esta manera no se toma en cuenta el acumulado del prefijo a ignorar por tener una contribución negativa y se inicia el cálculo nuevamente desde 0, lo que no ocurre en el alineamiento global.

Para el llenado de la matriz de costos, también se requiere una función para asignar un costo a los prefijos de las cadenas comparadas. La asignación de costos se basan en una función w en donde se tiene:

- $w(x_i, y_j)$ el costo de sustituir x_i por y_j si $x_i \neq y_j$
- $w(x_i, \epsilon)$ el costo de eliminar el caracter x_i
- $w(\epsilon, y_j)$ el costo de insertar el caracter y_j en una cadena de caracteres

A diferencia del alineamiento global, en las recurrencias implementadas el valor 0 actúa como reinicio del cálculo de la medida de similitud. Se consideran costos negativos cuando $x_i \neq y_j$, este valor no se acumula y en su lugar se agrega un 0. No hay distinción entre agregar y eliminar un caracter, por lo que $w(x_i, \epsilon) = w(\epsilon, y_j)$ y se denota con W. Otra característica importante es inicializar el primer renglón y columna de la matriz de la siguiente manera: $(x_i, 0) = 0 \; \forall i = 0 \ldots m$ si $|x| = m$ y $(0, y_j) = 0 \; \forall j = 0 \ldots m$ si $|y| = m$.

Al igual que para el cálculo de la medida se similitud global, se debe considerar un costo positivo para cuando haya una correspondencia entre los caracteres de las cadenas comparadas (*match*) y se denotará con M. El Algoritmo 4 muestra los pasos a seguir para el llenado de la matriz en donde se tiene:

$$t(i,j) \begin{cases} w(x_i, y_j) & \text{si } x_i \neq y_j \\ M & \text{si } x_i = y_j \end{cases}$$

Algoritmo 4 Cálculo de similitud local entre cadenas (Smith-Waterman)

Entrada: x y y
Salida: Matriz de distancias

$m \leftarrow |x|$
$n \leftarrow |y|$
$g \leftarrow w(x_i, y_j)$
for $i \leftarrow 0$ **to** m **do**
  $D_{i,0} \leftarrow 0$
end for
for $j \leftarrow 0$ **to** n **do**
  $D_{0,j} \leftarrow 0$
end for
for $i \leftarrow 1$ **to** m **do**
  for $j \leftarrow 1$ **to** n **do**
    $costoMatch \leftarrow D_{i-1,j-1} + t(i,j)$
    $costoInsertar \leftarrow D_{i,j-1} + g$
    $costoBorrar \leftarrow D_{i-1,j} + g$
    $D_{i,j} \leftarrow max(0, costoMatch, costoInsertar, costoBorrar)$
  end for
end for
return D

El algoritmo toma un tiempo de ejecución de $O(mn)$. Una vez obtenida la matriz de costos, el valor del alineamiento local óptimo se encuentra en la celda con el valor máximo. En la Figura 2.7 se muestra el llenado de una matriz para la obtención del valor de similitud local entre dos cadenas de caracteres, en donde los pesos son:

a) $W = -2$, $w(x_i, y_j) = -1$ y $M = 1$

b) $W = -1$, $w(x_i, y_j) = -2$ y $M = 2$

		G	A	T	C	A	G	G	A	A	C	T	G	A	G
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	1	0	0	0	0	1	1	0	0	0	0	1	0	1
A	0	0	2	0	0	1	0	0	2	1	0	0	0	2	0
G	0	1	0	1	0	0	2	1	0	1	0	0	1	0	3
C	0	0	0	0	2	0	0	1	0	0	2	0	0	0	1
G	0	1	0	0	0	1	1	1	0	0	0	1	1	0	1
G	0	1	0	0	0	0	2	2	0	0	0	0	2	0	1
A	0	0	2	0	0	1	0	1	3	1	0	0	0	3	1
A	0	0	1	1	0	1	0	0	2	4	2	0	0	1	2
T	0	0	0	2	0	0	0	0	0	2	3	3	1	0	0
T	0	0	0	1	1	0	0	0	0	0	1	4	2	0	0
G	0	1	0	0	0	0	1	1	0	0	0	2	5	3	1
C	0	0	0	0	1	0	0	0	0	0	1	0	3	4	2
A	0	0	1	0	0	2	0	0	1	1	0	0	1	4	3
C	0	0	0	0	1	0	1	0	0	0	2	0	0	2	3

		X	Y	A	X	B	A	C	S	L	L
	0	0	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0
A	0	0	0	2	1	0	2	1	0	0	0
X	0	2	1	1	4	3	2	1	0	0	0
A	0	1	0	3	3	2	5	4	3	2	1
B	0	0	0	2	2	5	4	3	2	1	0
C	0	0	0	1	1	4	3	6	5	4	3
S	0	0	0	0	0	3	2	5	8	7	6
T	0	0	0	0	0	2	1	4	7	6	5
V	0	0	0	0	0	1	0	3	6	5	4
Q	0	0	0	0	0	0	0	2	5	4	3

Figura 2.7: Ejemplo de llenado de dos matrices de costos utilizando el algoritmo para el cálculo de similitud local entre cadenas.

2.4.7. Similitud semiglobal entre cadenas

En un alineamiento semiglobal se pretende contabilizar el alineamiento de dos cadenas completas como se hace en el alineamiento global, con la diferencia de ignorar algunos de los espacios en blanco insertados al inicio o al final de las cadenas de caracteres al hacer el alineamiento. De esta manera el mejor alineamiento se encuentra en un segmento compacto y con alta similitud respecto a las cadenas completas.

El cálculo de la medida de similitud semiglobal es muy parecido al cálculo de la medida de similitud global. Cuando se comparan dos cadenas de caracteres x y y en donde $|x|$ varía considerablemente de $|y|$, esta característica contribuye a la obtención de un valor negativo si se hace el cálculo de la medida de similitud utilizando algoritmo de similitud global, inclusive cuando alguna de las cadenas se encuentre contenida en la otra.

La implementación del algoritmo de similitud semiglobal se puede lograr modificando el algoritmo de alineamiento global. El primer cambio consiste en inicializar la matriz de similitud con valores 0 y evitar la penalización por los espacios al inicio de las cadenas a comparar. Para eliminar los espacios al final basta con elegir la celda con el cálculo máximo que se puede encontrar en la columna m o en el renglón n de la matriz de costos D.

En este caso, al igual que en el alineamiento local se debe inicializar la matriz de la siguiente manera: $D_{x_i,0} = 0 \; \forall i \in [0, m]$ si $|x| = m$ y $D_{0,y_j} = 0 \; \forall j \in [0, n]$ si $|y| = n$.

Al igual que en los cálculos de medidas anteriores se requiere de una función w para la asignación de costos durante la correspondencia de las cadenas a comparar. Las operaciones que se pueden realizar con esta función son:

- $w(x_i, y_j)$ el costo de sustituir x_i por y_j si $x_i \neq y_j$
- $w(x_i, \epsilon)$ el costo de eliminar el caracter x_i
- $w(\epsilon, y_j)$ el costo de insertar el caracter y_j en una cadena de caracteres

Como se trata de una medida de similitud, se debe considerar un costo positivo M cuando exista una correspondencia entre caracteres de las cadenas a comparar. El Algoritmo 5 muestra los pasos seguir para obtener la medida de similitud semiglobal, en donde la función t se define:

$$t(i,j) \begin{cases} w(x_i, y_j) & \text{si } x_i \neq y_j \\ M & \text{si } x_i = y_j \end{cases}$$

Al igual que en los algoritmos anteriores, el tiempo de ejecución del algoritmo del cálculo de la similitud semiglobal es de $O(mn)$ y tampoco se hace distinción entre las operaciones para insertar o eliminar un caracter, por lo que $w(x_i, \epsilon) = w(\epsilon, y_j)$ y se denota con W.

Un ejemplo del llenado de una matriz de costos mediante el algoritmo de alineamiento semiglobal se muestra en la Figura 2.8 con los valores:

$$W = -2, w(x_i, y_j) = -1 \text{ y } M = 1$$

Algoritmo 5 Cálculo de similitud semiglobal entre cadenas

Entrada: x y y
Salida: Matriz de distancias

$g \leftarrow w(x_i, y_j) \forall x_i \neq y_j$
for $i \leftarrow 0$ **to** $|x|$ **do**
  $D_{i,0} \leftarrow 0$
end for
for $j \leftarrow 0$ **to** $|y|$ **do**
  $D_{0,j} \leftarrow 0$
end for
for $i \leftarrow 1$ **to** $|x|$ **do**
  for $j \leftarrow 1$ **to** $|y|$ **do**
    $costoMatch \leftarrow D_{i-1,j-1} + t(i,j)$
    $costoInsertar \leftarrow D_{i,j-1} + g$
    $costoBorrar \leftarrow D_{i-1,j} + g$
    $D_{i,j} \leftarrow max(costoMatch, costoInsertar, costoBorrar)$
  end for
end for
return $max(D_{|x|,0} \cdots D_{|x|,i} \cdots D_{|x|,n} D_{0,|y|}, D_{j,|y|} \ldots D_{|x|,|y|})$

		C	A	G	C	A	C	T	T	G	G	A	T	T	C	T	C	G	G
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	1	-1	-1	1	-1	1	-1	-1	-1	-1	-1	-1	-1	1	-1	1	-1	-1
A	0	-1	2	0	-1	2	0	0	-2	-2	-2	0	-2	-2	-1	0	-1	0	-2
G	0	-1	0	3	1	0	1	-1	-1	-1	-1	-2	-1	-3	-3	-2	-1	0	1
C	0	1	-1	1	4	2	1	0	-2	-2	-2	-2	-3	-2	-2	-4	-1	-2	-1
G	0	-1	0	0	2	3	1	0	-1	-1	-1	-3	-3	-4	-3	-3	-3	0	-1
T	0	-1	-2	-1	0	1	2	2	1	-1	-2	-2	-2	-2	-4	-2	-4	-2	-1
G	0	-1	-2	-1	-2	-1	0	1	1	2	0	-2	-3	-3	-3	-4	-3	-3	-1
G	0	-1	-2	-1	-2	-3	-2	-1	0	2	3	-1	-1	-3	-4	-4	-5	-2	-2

Figura 2.8: Llenado de una matriz de costos utilizando el algoritmo para el cálculo de similitud semiglobal entre cadenas.

2.4.8. Subsecuencia común más larga

Una manera de determinar si dos cadenas x y y son semejantes, es identificando si alguna de ellas es subcadena de la otra, es decir $x \in y$. Formalmente, si se tiene la cadena de caracteres $x = x_1, x_2, \ldots x_m$, se puede decir que la cadena $z = z_1, z_2, \ldots, z_k$ es una subsecuencia de x si existe una secuencia estrictamente creciente $i_1, i_2 \ldots, i_k$ de índices de x en donde $\forall j = 1, 2, \ldots, k$ se tiene $x_i = z_j$ [22]. Por ejemplo, $z = bcdb$ es una subsecuencia de $x = abcbdab$ con sus índices correspondientes $(2, 3, 5, 7)$.

Muchas aplicaciones en biología molecular requieren hacer un alineamiento entre secuencias de ADN de diversos organismos, algunas características de un organismo se pueden determinar si la secuencia de ADN de éste tiene una subsecuencia en común con el ADN de otro organismo, tratando en mayor medida que las subsecuencias encontradas no se encuentren interrumpidas por espacios. Si se tienen dos cadenas de caracteres x y y, se dice que la cadena de caracteres z es una *subsecuencia común* de x y y si z es una subsecuencia de ambas [22].

Una manera de encontrar la subsecuencia común más larga entre dos cadenas de caracteres x y y es encontrar todas las subsecuencias posibles de x y comparar cada una de ellas con todas las subsecuencias posibles de y para elegir el mejor resultado. Cada subsecuencia de x corresponde a un subconjunto de índices de $1, 2, \ldots, m$ de x, por lo que existen 2^m subsecuencias de x por lo que el tiempo de ejecución crece de manera exponencial.

Para darle solución al problema con un costo de tiempo de ejecución menor, se puede usar programación dinámica mediante el llenado de una matriz de costos como en los algoritmos anteriores. La matriz de costos D debe ser de dimensiones $m \times n$ si $|x| = m$ y $|y| = n$. Si $m = 0$ o $n = 0$, la longitud de la subsecuencia común más larga debe ser 0 [22].

Las soluciones de cada recursión que se utilizan para el cálculo de la longitud de la subsecuencia común más larga son:

$$D_{i,j} = \begin{cases} 0 & \text{si } m = 0 \text{ o } n = 0 \\ D_{i-1,j-1} + 1 & \text{si } i, j > 0 \text{ y } x_i = y_j \\ max(D_{i,j-1}, D_{i-1,j}) & \text{si } i, j > 0 \text{ y } x_i \neq y_j \end{cases}$$

Basándose en las funciones recursivas definidas, se pueden distinguir $\Theta(mn)$ subproblemas distintos, y es posible utilizar programación dinámica para resolver cada uno y encontrar una solución general [22].

Una vez llena la matriz de costos, la longitud de la subsecuencia común más larga se encontrará en la posición $D_{m,n}$. El Algoritmo 6 describe el proceso para obtener la subsecuencia común más larga entre las cadenas de caracteres x y y. Las longitudes de las subsecuencias encontradas se almacenan en la matriz de costos D, las trayectorias para localizar la subsecuencia se almacenan en la matriz T.

Algoritmo 6 Cálculo de subsecuencia común más larga

Entrada: x y y
Salida: Matriz de distancias

```
m ← longitud(x)
n ← longitud(y)
for i ← 1 to m do
    D_{i,0} ← 0
end for
for j ← 1 to n do
    D_{0,j} ← 0
end for
for i ← 1 to m do
    for j ← 1 to n do
        if x_i = y_i then
            D_{i,j} ← D_{i-1,j-1} + 1
            T_{i,j} ←↖
        else if D_{i-1,j} ≥ D_{i,j-1} then
            D_{i,j} ← D_{i-1,j}
            T_{i,j} ←↑
        else
            D_{i,j} ← D_{i,j-1}
            T_{i,j} ←←
        end if
    end for
end for
return D y T
```

La matriz T sirve para obtener la ruta por la que se consiguió la subsecuencia común más larga. El tiempo de ejecución es de $O(mn)$ ya que se recorre una vez cada elemento de la matriz. En la Figura 2.9 se puede observar una representación de la Matriz D señalando las trayectorias a seguir para cada uno de los valores indicados en la matriz T.

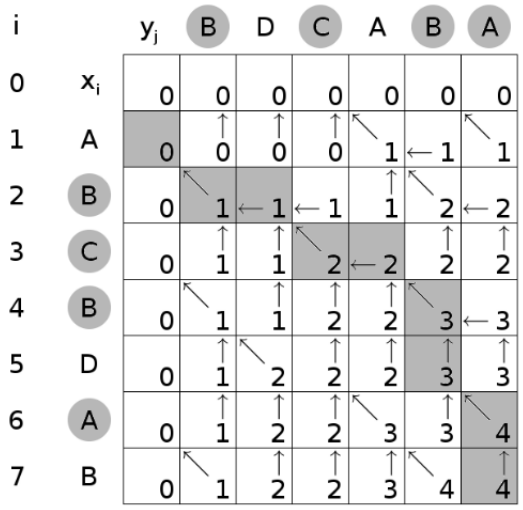


Figura 2.9: Representación de la matriz D junto con las trayectorias de la matriz T tras aplicar el algoritmo 6 a dos cadenas de caracteres.

2.4.9. Selección de los parámetros

Para los algoritmos de correspondencia entre cadenas mostrados en las secciones anteriores, es importante señalar que el resultado depende en gran medida de la selección de los pesos de las operaciones de edición simple que son utilizadas. Para la asignación de estos pesos se deben considerar las características de las cadenas que van a ser comparadas.

En el caso de los algoritmos de similitud, el valor de una correspondencia de caracteres debe ser mayor que el valor de una sustitución o una eliminación. De esta manera, la medida de similitud aumenta cuando existe un alineamiento de caracteres idénticos entre las cadenas; sin embargo, también se debe considerar

que las operaciones de sustitución o eliminación deben influir de manera negativa en el resultado de una medida de simlitud, por lo que se recomienda que para dos cadenas $x = x_1, x_2, \ldots, x_m$ y $y = y_1, y_2, \ldots, y_n$, los pesos para las operaciones de edición simple cumplan con:

$$w(x_i, \epsilon) < 0 \text{ y } w(\epsilon, y_i) < 0$$

$$w(x_i, y_i) < 0 \; \forall x_i \neq y_i.$$

En algunas aplicaciones puede ser más recomendable hacer una sustitución durante el proceso de alineamiento de cadenas en lugar de una eliminación. En biología molecular, por ejemplo, se pueden tener las correspondencias:

A		_	A
C		C	_

En este caso, el alineamiento de lado izquierdo puede tener una puntuación mayor que el mostrado a la derecha para una determinada aplicación. Con lo anterior se puede definir la regla de asignación de pesos:

$$2W < w(x_i, y_j) < M$$

Para definir reglas para la asignación de pesos como la mostrada anteriormente, es posible suponer otros ejemplos de alineamiento y establecer cuál sería el más deseado. En el campo de la biología, un emparejamiento entre aminoácidos con propiedades físicas o químicas similares, con tamaño, carga o hidrofobicidad semejante, suele recibir una mejor puntuación que el emparejamiento entre otros aminoácidos que no son tan parecidos [10].

En la base de datos de BiDi existen muchos registros de revistas electrónicas que se encuentran duplicados. Uno de los campos en donde se encuentran más valores repetidos es el título de las revistas. Una de las características de los títulos duplicados es que son muy semejantes a pesar de no ser cadenas iguales. Por ejemplo:

```
MyTravel PLC SWOT Analysis
MyTravel, PLC SWOT Analysis
Nabors Industries Inc SWOT Analysis
Nabors Industries, Inc. SWOT Analysis
```

La diferencia entre las cadenas mostradas radica en algunos signos de puntuación, aunque se puede notar que se trata de títulos repetidos. Al igual que en estos casos, muchos títulos de revistas electrónicas tienen una gran similitud con otros en la base de datos, algunos de los cuales se encuentran listados en el Apéndice B. También se pueden considerar casos en donde se ha cambiado una palabra pero el título sigue siendo el ”mismo”, por ejemplo:

```
Journal of Psychiatric & Mental Health Nursing
Journal of psychiatric and mental health nursing
```

En el caso de los nombres de las editoriales de las revistas electrónicas almacenados en la base de datos, muchos valores muestran la misma característica: los nombres de las editoriales repetidas son muy semejantes entre sí pero no son iguales. En el caso de las editoriales hay más diferencias entre cadenas que algunos signos de puntuación, por ejemplo:

```
American Association for the Advancement of Science
American Association for the Advancement of Science.AAAS

Cambridge University Press
Cambridge University Press (CUP)

IEEE
Institute of Electrical and Electronic Engineers

Miller Freeman Inc.
Miller Freeman, Inc.
```

Para buscar los valores más parecidos en los campos título y editorial, se pueden utilizar algoritmos de correspondencia de cadenas, como los descritos en esta capítulo, ya sea estableciendo una medida de similitud o una medida de distancia.

Independientemente del método de agrupación seleccionado, es necesario establecer un rango para determinar en qué momento se considera que una cadena es suficientemente similar a otra para agruparlas o suficientemente diferente para descartar una agrupación. Para el desarrollo de este trabajo, el rango se establece a través de porcentajes utilizando como base la longitud de la cadena con menor número de caracteres.

Aún haciendo la suposición de que muchos valores repetidos en la base de datos difieren únicamente en algunos signos de puntuación, es importante establecer un rango que permita agrupaciones exitosas. Por ejemplo, no se puede suponer que es suficiente con que dos cadenas sean semejantes al 50 % para ser agrupadas, ya que se pueden generar grupos de valores que no representan el mismo elemento como:

```
Bank Investment Consultant
Bank Investment Services Report
```

Es importante señalar que los métodos que proporciona el Sistema de Estandarización de Cadenas (SEC), desarrollado en este trabajo, son únicamente una herramienta para simplificar la tarea de la elaboración de los diccionarios de sinónimos; sin embargo, la contribución del sistema consiste en la introducción del uso de diccionarios de sinónimos en una base de datos relacional utilizando una base de datos nativa XML.

La elaboración de los diccionarios de sinónimos debe ser un proceso supervisado por un experto, ya que existen casos en donde las cadenas son completamente diferentes y no pueden ser agrupadas utilizando alguno de los algoritmos descritos anteriormente; sin embargo, hacen referencia a las mismas editoriales, por ejemplo:

```
IEEE
Institute of Electrical and Electronic Engineers

INFORMS
INFORMS - Institute for Operations Research
```

Para que estas cadenas sean agrupadas se puede suponer que es necesario establecer un rango muy pequeño si se utilizan algoritmos de distancia o muy grande

si se utilizan algoritmos de similitud. Independientemente del método a utilizar, si se logra establecer un rango que agrupe estos pares de cadenas, es muy probable que no funcione para hacer agrupaciones exitosas con el resto de los valores.

En la Tabla 2.4 se muestran algunos resultados de comparar pares de cadenas utilizando los algoritmos de distancia de edición simple (DES) y distancia de edión general (DEG). El porcentaje que se muestra en la tabla es respecto a la longitud de la cadena más corta.

Título	**DES**	**%**	**DEG**	**%**
CRH PLC SWOT Analysis CRH, PLC SWOT Analysis	1	4.76	4	6.35
Canadian Journal of Criminology and Criminal Justice Canadian Journal of Criminology & Criminal Justice	3	6	11	7.33
Criminal Law Criminal Law Forum	6	50	24	66.67
Criminology and Public Policy Criminology & Public Policy	3	11.11	11	13.58
MyTravel PLC SWOT Analysis MyTravel, PLC SWOT Analysis	1	3.85	4	5.7
American Association for the Advancement of Science American Association for the Advancement of Science.AAAS	5	9.8	20	13.07
Bank Investment Consultant Bank Investment Services Report	13	50	44	56.41
Nabors Industries Inc SWOT Analysis Nabors Industries, Inc. SWOT Analysis	2	5.71	8	7.62
Cambridge University Press Cambridge University Press (CUP)	6	23.08	24	30.77
Miller Freeman Inc. Miller Freeman, Inc.	1	5.26	4	7.02
IEEE Institute of Electrical and Electronic Engineers	45	–	56	–
INFORMS INFORMS - Institute for Operations Research	32	–	57	–

Tabla 2.4: Resultados de medida de distancia para un grupo de títulos de revistas electrónicas. La distancia de edición general se obtiene considerando los valores igualdad=0, reemplazo=3, espacio=4

En la tabla 2.5 se muestran algunos ejemplos de las cadenas que se han comparado mediante los algoritmos de similitud.

Título	**SG**	**%**	**SL**	**%**	**SSG**	**%**	**SCL**	**%**
CRH PLC SWOT Analysis CRH, PLC SWOT Analysis	20	95.24	20	95.24	20	95.24	20	95.24
Canadian Journal of Criminology and Criminal Justice Canadian Journal of Criminology & Criminal Justice	46	92	46	92	46	92	20	40
Criminal Law Criminal Law Forum	6	50	12	100	12	100	12	100
Criminology and Public Policy Criminology & Public Policy	23	85.19	23	85.19	23	85.19	14	51.85
MyTravel PLC SWOT Analysis MyTravel, PLC SWOT Analysis	25	96.15	25	96.15	25	96.15	25	96.15
American Association for the Advancement of Science American Association for the Advancement of Science.AAAS	46	90.2	51	100	51	100	51	100
Bank Investment Consultant Bank Investment Services Report	6	23.08	16	61.54	8	30.77	–	–
Nabors Industries Inc SWOT Analysis Nabors Industries, Inc. SWOT Analysis	33	94.29	33	94.29	33	94.29	34	97.14
Cambridge University Press Cambridge University Press (CUP)	6	23.08	26	100	26	100	26	100
Miller Freeman Inc. Miller Freeman, Inc.	18	94.74	18	94.74	18	94.74	19	100
IEEE Institute of Electrical and Electronic Engineers	-40	-	2	50	2	50	–	–
INFORMS INFORMS - Institute for Operations Research	-29	-	7	100	7	100	7	100

Tabla 2.5: Resultados de medida de distancia para un grupo de títulos de revistas electrónicas. La medida de similitud global (SG) considera los valores igualdad=1, reemplazo=-1, espacio=-1

En este trabajo se estandarizarán los valores de la base de datos de BiDi para los campos de `titulo` y `editor` de las revistas electrónicas, por lo que se deben hacer dos diccionarios de sinónimos: uno para los títulos y otro para los nombres de las editoriales. Debido a que las cadenas duplicadas tanto de los títulos como de los nombres de las editoriales tienen una secuencia de caracteres semejante, se sabe que al utilizar el algoritmo de similitud semiglobal para hacer las agrupaciones de palabras y además utilizar un porcentaje alto para identificar el rango de similitud se pueden agrupar los casos como:

```
Cambridge University Press
Cambridge University Press (CUP)
```

```
INFORMS
INFORMS - Institute for Operations Research
```

Sin embargo, no es viable utilizar estos parámetros para la creación de los diccionarios de sinónimos en BiDi porque se agrupan cadenas que hacen referencia a diferentes elementos como:

```
HFN: The Weekly Newspaper for the Home Furnishings Network
Home Furnishings

Hampton Roads International Security Quaterly
International Security

Harvard Men's Health Watch
Harvard women's health watch (Print)

Health News
Health News Naturally
Health news & review
```

Si se elabora el diccionario de sinónimos utilizando el algoritmo de similitud global y además se considera un rango de 90 % de similitud para agrupar palabras, las cadenas anteriormente mencionadas no son agrupadas; sin embargo, algunas cadenas tampoco serán agrupadas a pesar de tener una gran semejanza. Esto se debe a que ambas cadenas son de longitudes muy cortas y el cambio en cualquier caracter afecta de manera drástica el porcentaje de similitud. Por ejemplo:

```
Criminology and Public Policy
Criminology & Public Policy
```

En cambio las siguientes cadenas sí son agrupadas:

```
Canadian Journal of Criminology and Criminal Justice
Canadian Journal of Criminology & Criminal Justice
```

Como se puede notar, las longitudes de las cadenas afectan drásticamente la agrupación de los elementos. Por esta razón, se destaca la importancia de que la creación de los diccionarios de sinónimos a través del sistema debe ser un proceso supervisado.

2.5. Resumen

En este capítulo se define el concepto de *calidad de datos* y se describe la importancia que tiene en las grandes bases de datos y cómo los datos han incrementado su valor a lo largo de los años, ya que las empresas los usan cada vez más como apoyo en la toma de decisiones importantes que pueden ser rutinarias o estratégicas.

El concepto de calidad de datos es un término que no tiene una definición formal, aunque de manera general se puede decir que la calidad de datos depende de las expectativas del usuario final. La definición establecida por Redman sugiere que la calidad de datos se puede obtener de comparar dos fuentes de datos: *Una colección de datos X es de mayor calidad que una colección de datos Y si X satisface mejor que Y las necesidades del usuario* [7, 8].

Para medir la calidad de los datos, se han establecido propiedades que los datos deben de cumplir, las cuales se describen en este capítulo. Estableciendo las características que deben tener los datos para ser considerados *datos de calidad*, es posible definir métodos para localizar datos con errores ya sean simples o complejos.

Para eliminar errores en una base de datos se debe hacer una *limpieza de datos*. Esto es, analizar el conjunto de datos y localizar los errores para ser corregidos posterioremente. La presencia de registros duplicados es un error que se presenta en las bases de datos relacionales. Los datos de tipo cadena pueden estar duplicados en la base de datos aunque no se trate de cadenas exactamente iguales ya que pueden hacer referencia al mismo elemento. Estas inconsistencias generalmente son el resultado de la integración de datos de diversas fuentes, como sucede en la base de datos de la Biblioteca Digital de la UNAM.

Para estandarizar los valores de una base de datos, lo primero es identificar los elementos duplicados no exactos. Una característica de este tipo de registros almacenados en la base de datos de BiDi, es que las cadenas de caracteres de los elementos duplicados son muy semejantes entre sí. Una manera de encontrar aquellos elementos duplicados es utilizando algoritmos basados en la correspondencia entre cadenas.

En este capítulo se describen algunos algoritmos de correspondencia entre cadenas para encontrar los duplicados no exactos entre las diferentes cadenas de caracteres almacenadas en la base de datos de BiDi, específicamente en los campos de título y editorial de las revistas electrónicas. El objetivo es establecer un nivel de similitud entre un par de cadenas x y y para considerar una correspondencia exitosa. Dependiendo del nivel de correspondencia entre las cadenas comparadas se definen los conceptos de *similitud* y *distancia*.

Para los algoritmos de correspondencia entre cadenas, es importante señalar que el resultado depende en gran medida de la selección de los pesos de las operaciones de edición simple que son utilizadas. Para la asignación de estos pesos, se deben considerar las características de las cadenas que van a ser comparadas. En este capítulo se describen de manera general algunas consideraciones para la asignación de los parámetros.

Analizando algunas cadenas almacenadas en la base de datos de BiDi, se definen los parámetros que se utilizarán para la creación de los diccionarios de sinónimos de los campos título y editorial de la sección de revistas electrónicas de la base de datos para hacer la estandarización de valores.

Capítulo 3

XML y Bases de Datos

El lenguaje de marcas extensible XML (*eXtensible Markup Language*) es un estándar avalado por el W3C[1] para marcar documentos, derivado del lenguaje estándar de marcas generalizadas SGML (*Standard Generalized Markup Language*) desarrollado en IBM a finales de 1970 [15].

XML, al igual que SGML, es un lenguaje de *meta-marcas* para documentos de texto. Los datos se incluyen en un documento XML como cadenas de texto y se rodean mediante marcas de texto que las describen. La unidad básica de datos y marcas de XML se denomina *elemento* y la especificación XML define la sintaxis de texto exacta que deben seguir las marcas a lo largo del documento [15].

En el desarrollo de SGML aparece el concepto de DTD (*Document Type Definition*), un procedimiento para expresar una gramática de documentos, basado en una especificación de reglas que permite validarlos automáticamente y hacer que la estructura de cada tipo de documento quede especificada de forma estricta haciéndolo un *documento válido* [13].

El problema de SGML residía en que era tan complicado que casi ningún software lo ha implantado completamente. Los programas que lo hicieron o que se basaron en los distintos subconjuntos de SGML eran normalmente incompatibles entre sí [15]. El mayor éxito de SGML fue HTML, que es una aplicación simplificada de SGML, ya que limita el uso a un conjunto restringido de etiquetas

[1] Consorcio World Wide Web

únicamente para describir páginas Web. La excesiva simplificación de SGML que supuso HTML acabó mostrando sus carencias. Cuando la Web se desarrolló plenamente se evidenció la necesidad de volver al concepto de *documento válido* en el que está basado SGML [13].

En febrero de 1998 apareció *XML 1.0* como una versión *liviana* de SGML que conservaba la mayor parte de su eficacia, pero en donde sus creadores recortaron muchas opciones que habían demostrado ser redundantes, complicadas para su implantación o confusas para los usuarios finales [15]. El siguiente estándar de XML tiene nuevas funcionalidades que se describirán a lo largo de este capítulo.

XML es un lenguaje que se puede ampliar ya que no tiene un conjunto fijo de etiquetas, permitiendo a los desarrolladores crear los elementos que necesiten cuando los necesiten.

La gramática para documentos XML es lo suficientemente específica como para permitir el desarrollo de analizadores sintácticos de XML [15]. Los documentos que satisfacen dicha gramática se dice que están *bien formados*. En un documento XML bien formado, las marcas también describen la semántica del documento ya que, además de describir la estructura del mismo, nos permite saber qué elementos se asocian entre sí.

XML es muy útil como formato de datos cuando debe haber comunicación entre aplicaciones o integración de información de diversas fuentes porque los documentos XML son textos que se pueden leer con cualquier heramienta que pueda leer un archivo de texto [15]. Es por eso que XML se ha convertido en una herramienta útil para el tratamiento de la información, principalmente porque permite manipular los datos de manera estructurada. Un sistema que procese documentos XML puede acceder al contenido y a la estructura de los documentos [36].

Con XML es posible representar datos de bases de datos relacionales, así como muchas clases de datos estructurados [21]. A continuación se listan las diferencias que existen entre datos XML y datos relacionales según Martin [13] :

- *Metadatos*. Los datos relacionales presentan estructuras regulares y homogéneas lo que permite usar metadatos sin ningún problema, mientras en XML los datos son heterogéneos e irregulares con estructuras que deben

describirse caso a caso, de forma que estos metadatos se acaban describiendo en el propio documento.

- *Anidamiento*. Los documentos XML contienen distintos niveles de anidamiento, que son irregulares e impredecibles, mientras que los datos relacionales son *planos* al estar organizados a partir de tablas.

- *Jerarquía*. En XML existe una jerarquía y un orden intrínseco que no se da en la estructura relacional en donde carece de relevancia.

- *Densidad*. Los datos relacionales son densos (a cada columna se le asigna un valor) y los inexistentes se declaran como tales (con `null`), en cambio los datos en XML son dispersos y la información que no existe sencillamente carece de elemento. Como consecuencia XML es más libre que el modelo relacional a la hora de enfrentarse con datos faltantes.

Comparado al almacenamiento de los datos en una base de datos, la representación XML puede parecer poco eficiente, puesto que los nombres de las etiquetas se repiten por todo el documento; sin embargo, una representación XML presenta ventajas significativas cuando se utiliza para el intercambio de datos [11] tales como:

- La presencia de las etiquetas a lo largo de un documento XML hace que el mensaje sea autocomentado, es decir, no se tiene que consultar una DTD para comprender el significado del texto, lo que lo hace relativamente legible y usable por el usuario final [36].

- El formato del documento no es rígido, algunas etiquetas pueden ser reconocidas o ignoradas permitiendo al formato de los datos evolucionar con el tiempo sin invalidar las apliacciones existentes.

- Debido a que el formato XML está ampliamente aceptado, hay una gran variedad de herramientas disponibles para ayudar a su procesamiento, incluyendo software de búsqueda y herramientas de bases de datos.

Durante el diseño de las bases de datos relacionales se crean esquemas conceptuales que se usan para restringir qué información se puede almacenar en ellas así como qué tipos de datos se ingresan en las tablas [21]. Por otro lado, las marcas permitidas en una determinada aplicación XML se pueden documentar en un

esquema XML (*XML Schema*), y si un documento XML coincide con este esquema se dice que es un *documento válido*; sin embargo, no todos los documentos XML tienen que ser válidos. Muchas veces es suficiente con que el documento esté bien formado [15].

3.1. Estructura de los Documentos XML

Un documento XML tiene una estructura física y lógica. Físicamente el documento está formado por un conjunto de elementos. La estructura lógica de un documento XML consiste en la jerarquía que tienen los elementos dentro del documento, de manera que el *marcado* describe la estructura de la información y el *texto* su contenido [13].

El marcado corresponde a las instrucciones que el analizador XML debe procesar y se encuentran delimitadas por los caracteres '<'y '>', estas marcas son también llamadas *etiquetas*. XML no tiene ningún conjunto de etiquetas predefinidas por lo que se pueden crear nuevas etiquetas de acuerdo a los datos que van a ser representados. Las etiquetas se utilizan en pares y tienen la forma `<nombre></nombre>`, donde *nombre* es el nombre del elemento que se está representando. Normalmente, el nombre de las etiquetas refleja el tipo de contenido del elemento [15].

En los documentos XML es posible insertar comentarios, es decir, texto informativo que debe ser ignorado por el procesador XML. Los comentarios en XML tienen el siguiente formato:

```
<!-- Esto es un comentario -->
```

Todo documento XML contiene uno o más elementos. Un *elemento* en XML es una estructura compuesta por una etiqueta inicial, una etiqueta final y la información entre las etiquetas. Todos los elementos deben estar bien delimitados e identificados por un nombre llamado *identificador genérico* [13]. Por ejemplo, sea el siguiente elemento `libro`:

```
<libro> Esta es una novela.. </libro>
```

El *contenido de un elemento* es cualquier cosa contenida entre sus etiquetas de inicio y final, y puede constar de texto como de otros elementos. En el caso de un elemento sin contenido se denomina *elemento vacío* [13].

Los elementos de XML pueden tener atributos, que son una manera de incorporar información relacionada acerca de sí mismos, describiendo sus propiedades [13]. Un *atributo* es un par de valores de nombre añadido a la etiqueta de inicio del elemento. Los nombres se separan de los valores mediante signos de igual y un espacio en blanco opcional. Los valores se escriben entre signos de comillas dobles. Por ejemplo, el siguiente elemento `capitulo` tiene un atributo con nombre `titulo` cuyo valor es `XML`

```
<capitulo titulo="XML"> Contenido </capitulo>
```

Los atributos son bastante limitados en su estructura porque el valor del atributo es simplemente un texto sin diferenciar [15]. Se debe señalar que cada elemento no puede tener más de un atributo con un nombre determinado.

Los documentos XML tienen una estructura fija compuesta por un prólogo y un cuerpo. En el prólogo se especifican las características del documento en sí, como la versión de XML a la cual pertenece la especificación del tipo o estructura al cual debe ajustarse para ser válido. En el cuerpo de un documento XML se incluyen los datos o la información propiamente dicha, formada por el contenido en sí del documento. [13].

3.1.1. Prólogo

Aunque no es obligatorio, un documento XML puede comenzar con unas líneas que describen la versión XML[2] que será usada y el tipo de documento entre otras cosas. Estas líneas forman el prólogo del documento, el cual contiene de manera general [36]:

- Una declaración XML, que es la sentencia que declara al documento como un documento XML.

[2]La versión utilizada especifica la sintaxis que puede ser utilizada a lo largo del documento [36]

- Una declaración de tipo de documento que enlaza el documento con su Definición de Tipo de Documento (DTD) en un archivo externo o la DTD puede estar incluida en la declaración.

- Uno o más comentarios o instrucciones de procesamiento.

Todo documento XML debe comenzar con un encabezado, que es la primera sentencia del documento y está encerrada entre `<?xml ?>`. Dentro de esta sentencia se agrega la versión de XML a la cual se ajusta el documento, por ejemplo para la versión 1.0 debe quedar de la siguiente forma:

```
<?xml version="1.0"?>
```

Aunque en estos momentos ya ha aparecido la versión `1.1`, el uso de `1.0` es muy general. La declaración de la versión del XML debe usarse, ya que en un futuro puede asumirse que un documento que no tiene una versión especificada dentro del prólogo es de la última versión, con lo que pueden surgir problemas y errores en su procesamiento [13].

En la declaración XML, se puede especificar la codificación del documento como `UTF-8` ó `ISO-8859-1`. Por ejemplo, en la siguiente declaración:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

Dentro del prólogo existe un atributo `standalone` que indica si se necesita de un documento externo (DTD o esquema XML) para definir la estructura del documento, requiere los valores `"yes"` o `"no"`, por ejemplo:

```
<?xml version="1.0" encoding="ISO-8859-1"
                            standalone="no"?>
```

Para indicar en el prólogo que una DTD se incorpora al documento XML, se debe agregar una *declaración del tipo de documento* en donde se encuentra la información sobre la DTD mediante un identificador público `PUBLIC` que hace referencia a dicha DTD, o mediante un Identificador Universal de Recursos `(URI)` precedido por la palabra `SYSTEM`. Por ejemplo:

```
<!DOCTYPE MENSAJE SYSTEM "mensaje.dtd">
```

3.1.2. Cuerpo del documento

El cuerpo de un documento XML está formado por uno o más elementos. Cuando se desea agregar caracteres de marcado XML en el contenido de un elemento se deben utilizar los elementos predefinidos. Los elementos predefinidos permiten representar caracteres especiales, de manera que no sean interpretados como marcado por el procesador XML. Los elementos predefinidos se muestran en la Tabla 3.1.

Elemento	Caracter
&amp;	&
&lt;	¡
&gt;	¿
&apos;	’
&quot;	”

Tabla 3.1: Tabla de elementos prefinidos en XML.

Cuando se requiere utilizar muchos elementos predefinidos para escribir un documento XML, se puede utilizar una sección CDATA, que permite especificar datos utilizando cualquier caracter sin que se interprete como marcado y permite que caracteres especiales no rompan la estructura del documento[3]. Por ejemplo, el siguiente fragmento en XML:

```
<capitulo titulo="HTML">
  <ejemplo>
    &lt;etiqueta&gt;
       Contenido
    &lt;/etiqueta&gt;
  </ejemplo>
</capitulo>
```

[3]Lo único que no puede aparecer en una sección CDATA es el delimitador final de la sección: `CDATA,]]>`.

Se puede reescribir como:

```
<capitulo titulo="HTML">
  <ejemplo>
     <![CDATA[
     <etiqueta>
       Contenido
     </etiqueta> ]]>
  </ejemplo>
</capitulo>
```

Instrucciones de procesamiento

Las *instrucciones de procesamiento* es un mecanismo que permite a los documentos XML contener instrucciones específicas para las aplicaciones que los van a utilizar, sin que éstas formen parte de los datos del documento. El analizador XML al detectarlas se limita a pasar esa información a la aplicación que realiza la llamada, indicándole simplemente el modo de administrar los datos del documento [13].

Una instrucción de procesamiento se delimita mediante '<?'y '?>' aunque existe una excepción cuando se utiliza en el prólogo del documento XML. Las instrucciones de procesamiento empiezan con un identificador (acompañado de un valor) denominado *objetivo* que, siguiendo las mismas reglas de los nombres de elementos y atributos, se usa para identificar la aplicación a la cual se dirige [13]. En el siguiente ejemplo, la instrucción de procesamiento se usa para indicar a la aplicación que el documento se debe mostrar con una determinada hoja de estilo:

```
<?xml-stylesheet type="text/xsl" href="h-estilo.xsl"?>
```

3.2. Documentos XML bien formados

Los documentos XML deben seguir una estructura estrictamente jerárquica respecto a las etiquetas que delimitan sus elementos. El cuerpo del documento debe contener un solo elemento raíz y los elementos contenidos en éste deben estar correctamente anidados. Es así como en los documentos XML las etiquetas

se estructuran en forma de árbol n-ario, con nodos para los elementos como se muestra en la Figura 3.1.

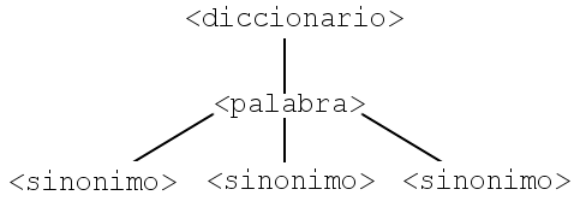


Figura 3.1: Árbol generado de un XML

Para que un documento XML se encuentre bien formado, todos los elementos con contenido deben estar correctamente *cerrados* de acuerdo a la jerarquía. Para el árbol mostrado en la Figura 3.1 se tiene el siguiente documento XML de ejemplo:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <diccionario>
    <palabra valor="triunfo">
      <sinonimo valor="victoria"/>
      <sinonimo valor="conquista"/>
    </palabra>
  </diccionario>
```

Para que un documento XML esté bien formado debe cumplir varias reglas, algunas de ellas son [15]:

- Toda etiqueta de inicio debe tener una etiqueta de cierre coincidente.
- Debe haber exactamente un elemento raíz
- Los valores de atributos deben estar entre comillas.
- Un elemento no puede tener dos atributos con el mismo nombre.
- Los comentarios y las instrucciones de procesamiento no pueden estar dentro de las etiquetas.

Si un documento escrito en XML no está bien formado, no se considera un documento XML, por lo que el analizador al detectarlo, procede a notificarlo e interrumpe su trabajo [13].

3.3. Documento XML válido

Escribir un documento XML bien formado generalmente no es suficiente para crear una aplicación por sencilla que sea, ya que de alguna forma se tiene que limitar o controlar en mayor o menor medida el tipo de datos a incorporar [13].

Un documento XML *válido* es aquel que además de estar bien formado se ajusta a una estructura definida, la cual se puede escribir como una DTD o como un esquema XML del W3C. Aunque no es necesario que a un documento XML se le asocie una estructura, su uso es más que recomendable ya que debe existir un mecanismo para asegurar la conformidad del documento cuando vaya a ser intercambiado, ya que de otro modo las aplicaciones que utilicen los documentos tendrán que detectar la estructura de cada uno y se puede correr el riesgo de generar inconsistencias al carecer de dichas especificaciones [15].

En XML se tiene que hacer la distinción entre *documento válido* y *documento bien formado*, en este último no hay restricciones sobre el número o tipo de contenidos de elementos. En un documento válido, además de estar bien formado se deben respetar las restricciones de la estructura establecidas por la definición de un esquema XML o una DTD [13]. Un documento XML puede estar bien formado y no ser un documento válido. Cuando existen documentos que no tienen una DTD asociada y se encuentran bien formados, no es posible decir que son válidos o no.

3.3.1. Definición del Tipo de Documento

La Definición de Tipo de Documento (DTD) aparece en el desarrollo de SGML para explicar, mediante una sintaxis formal, qué elementos pueden aparecer en un documento XML y en dónde, así como el contenido y los atributos de los mismos [15]. Si un documentoXML tiene una declaración del tipo de documento y el documento satisface la DTD que indica la declaración de tipo de documento, entonces se dice que el documento es válido.

La validación de un documento opera sobre el principio de que todo lo que no está permitido está prohibido, por lo que un documento XML válido bajo una DTD puede omitir elementos presentados en la DTD. Las DTD proporcionan la capacidad de realizar una validación básica de los siguientes elementos en documentos XML [15]:

- Anidamiento de elementos.
- Limitaciones de ocurrencia de elementos.
- Atributos permitidos.
- Tipos de atributos y valores predeterminados.

Una DTD puede estar incluida en el documento XML como una declaración de tipo de documento o puede definirse en un archivo diferente para poder ser compartida por varios documentos. En el siguiente ejemplo se hace referencia a una DTD externa:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE dicc SYSTEM "http://unam.mx/ejemplo.dtd">
<diccionario>
  <palabra valor="Cualquier palabra"/>
</diccionario>
```

Una DTD es una colección de declaraciones de elementos (`ELEMENT`), atributos (`ATTLIST`), entidades (`ENTITY`) y notaciones (`NOTATION`) a partir de las cuales se describe la validez de un documento.

Declaraciones de elementos

Los elementos son la base del marcado en los documentos XML [37]. Todos los elementos usados en un documento válido, deben declararse en la DTD de un documento con una declaración de elemento [15]. Las declaraciones de elementos deben comenzar con `<!ELEMENT` seguidas por el identificador genérico del elemento que se declara, por ejemplo:

```
<!ELEMENT revista(titulo, issn)>
```

En el ejemplo anterior se está definiendo que el elemento <revista> debe contener un elemento <titulo> y un elemento <issn>, los cuales también deben estar definidos en la DTD. Un elemento que cumpla esta declaración de tipo es, por ejemplo:

```
<revista>
 <titulo>Canadian Journal of Microbiology</titulo>
 <issn>1480-3275</issn>
</revista>
```

La especificación de contenido puede ser de 4 tipos:

1. EMPTY. Puede no tener contenido.

   ```
   <!ELEMENT nota EMPTY>
   ```

2. ANY. Puede tener cualquier contenido, no se suele utilizar ya que es conveniente estructurar adecuadamente los documentos XML [37].

   ```
   <!ELEMENT anexo ANY>
   ```

3. MIXED. Puede tener caracteres de tipo datos o una mezcla de caracteres y subelementos definidos en la especificación de contenido mixto.

   ```
   <!ELEMENT enfasis (#PCDATA)>
   <!ELEMENT parrafo (#PCDATA|enfasis)>
   ```

   En el ejemplo anterior, el elemento <enfasis> sólo debe contener datos de caracter (#PCDATA) y el elemento <parrafo> debe contener datos de caracter (#PCDATA) ó un elemento de tipo <enfasis>.

4. ELEMENT. Sólo puede contener sub-elementos que se hayan definido en la especificación de contenido, por ejemplo:

   ```
   <!ELEMENT mensaje (remitente, destinatario, texto)>
   ```

Modelos de contenido

Un modelo de contenido es un patrón que establece los subelementos que debe contener un elemento y el orden en que deben aparecer. Un modelo de contenido sencillo puede tener un solo tipo de sub-elemento. En el siguiente ejemplo cada elemento <aviso> sólo puede contener un elemento <parrafo>:

```
<!ELEMENT aviso (parrafo)>
```

Además de que existen reglas sintácticas para definir el orden y la anidación de los elementos[4], cada elemento definido en el modelo de contenido puede llevar un indicador de frecuencia, que siguen directamente a un identificador. Los indicadores de frecuencia se listan en la Tabla 3.2

?	Opcional (0 o 1 vez)
*	Opcional y repetible (0 o más veces)
+	Necesario y repetible (1 o más veces)

Tabla 3.2: Tabla de indicadores de frecuencia.

Un ejemplo del uso de los indicadores de frecuencia es:

```
<!ELEMENT aviso (titulo?, (parrafo+, grafico)*)>
```

En el ejemplo se define que un elemento <aviso> puede o no contener un título, además debe contener al menos un elemento párrafo (puede contener más) pero no necesariamente requiere contener un elemento <grafico>, aunque si lo hay puede ser más de uno.

Declaraciones de listas de atributos

Una declaración de lista de aributos permite añadir información sencilla y desestructurada a los elementos de un documento; se declara una lista de atributos porque puede existir más de un atributo por elemento [13].

[4]Las reglas para la definición de una DTD están disponibles en la especificación.

Una declaración de lista de atributos empieza con <!ATTLIST seguido del identificador del elemento al que se aplica. Posteriormente viene el nombre del atributo, su tipo y su valor por defecto. Por ejemplo, sea la siguiente DTD:

```
<!ELEMENT mensaje (remitente, destinatario, texto)>
<!ATTLIST mensaje prioridad(normal | urgente) normal>
<!ELEMENT texto (#PCDATA)>
<!ATTLIST texto idioma CDATA #REQUIRED>
```

En el ejemplo anterior, el elemento <mensaje> tiene un atributo con nombre prioridad, el cual puede tener el valor normal o urgente. Si el atributo no es especificado toma el valor normal que es el valor por defecto. El atributo idioma del elemento <texto> no tiene valor por defecto y es obligatorio especificar este atributo, por eso aparece la sentencia #REQUIRED. Para especificar que es posible omitir un atributo sin que tome un valor por defecto, se usa la sentencia #IMPLIED, por ejemplo:

```
<!ATTLIST imagen url CDATA #IMPLIED>
```

Las entidades son abreviaturas de texto que se utilizan cuando éstas son referenciadas por un elemento en el documento XML, cuando el analizador sintáctico encuentra una referencia a una entidad reemplaza la referencia por el contenido. Las entidades pueden ser:

- Internas o Externas
- Analizadas o No analizadas
- Generales o Parámetro

Entidades internas

Las entidades internas se encuentran definidas en la sección de la DTD del documento XML, una vez que el analizador sintáctico reemplaza a la entidad por su contenido, ésta pasa a ser parte del documento XML y puede ser analizada por el procesador XML. Por ejemplo:

```
<!DOCTYPE texto[
<!ENTITY frase "Piensa, cree, sueña y atrévete">
]
<texto>
  <titulo>
     Alguna vez Walt Disney dijo: &frase;
  </titulo>
</texto>
```

Entidades externas

Las entidades externas obtienen su contenido en cualquier otro sitio del sistema, ya sea de otro archivo del disco duro, una página web o un objeto de una base de datos. Se hace referencia al contenido de una entidad externa mediante la palabra `SYSTEM` seguida de un URI (Universal Resource Identifier) [37].

```
<!ENTITY intro SYSTEM "http://ejemplo.com/intro.xml">
```

Cuando el contenido de la entidad es un archivo que no debe ser interpretado como si fuera texto XML, se dice que se trata de una entidad general y externa, tal es el caso de los archivos `MPG` o las imágenes `GIF`, por ejemplo:

```
<!ENTITY logo SYSTEM "http://ejemplo.com/logo.gif">
```

Entidades parámetro

Las entidades párametro son aquellas que sólo pueden usarse en la DTD y no en el documento XML. Se pueden utilizar para agrupar ciertos elementos de la DTD que se repiten mucho. Para hacer referencia a una etidad parámetro se utiliza el símbolo `'%'` en lugar de `'&'`, tanto para declararlas como para usarlas. Por ejemplo:

```
<!DOCTYPE texto[
<!ENTITY % elemento-frase "<!ELEMENT FRASE (#PCDATA)>">
 ...
 %elemento-frase;
 ]>
```

Si además fuera una entidad externa, se puede escribir como:

```
<!DOCTYPE texto[
<!ENTITY % elemento-frase SYSTEM "frase.ent">
 ...
%elemento-frase;
]>
```

DTD para un diccionario de sinónimos

Para este trabajo de tesis se utilizarán documentos XML para representar diferentes diccionarios de sinónimos. La DTD asociada a los documentos generados en este trabajo es la siguiente:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!ELEMENT diccionario (palabra)+>
<!ELEMENT palabra (sinonimo)*>
<!ATTLIST palabra valor PCDATA #REQUIRED>
<!ELEMENT sinonimo (#PCDATA)>
<!ATTLIST sinonimo valor PCDATA #REQUIRED>
```

Con la DTD definida podemos tener el siguiente documento XML válido.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <diccionario>
    <palabra valor="triunfo">
      <sinonimo valor="victoria"/>
      <sinonimo valor="conquista"/>
    </palabra>
    <palabra valor="amanecer"/>
    <palabra valor="amplificar">
      <sinonimo valor="ampliar"/>
    </palabra>
  </diccionario>
```

3.3.2. Esquemas XML (XML Schema)

Como se mencionó en la sección anterior, las DTD proporcionan la capacidad de realizar una validación básica de un documento XML; sin embargo, hay muchas cosas que no indica la DTD. En particular, no indica lo siguiente [15]:

- Cuál es el elemento raíz del documento XML.
- La cantidad de instancias de cada tipo de elemento que aparece en el documento.

Es decir, con una DTD se puede describir la estructura de un documento XML, pero presenta importantes limitaciones a la hora de definir del contenido permitido del mismo [13]. Por ejemplo, la DTD no permite especificar tipos de datos y no es extensible, por lo que una vez definida no es posible añadir vocabularios. Por ello y con el objetivo de superar las carencias de las DTD surgió la idea de generalizar las DTD utilizando la sintaxis XML a la hora de definir y validar las características de un documento a través de los esquemas XML.

Los esquemas XML pueden implantar reglas mucho más específicas que las DTD sobre el contenido de los elementos y los atributos. Además de tipos simples integrados (entero, decimal, cadena), se pueden incluir restricciones más explícitas sobre el número y la secuencia de los hijos de un elemento. [15].

Un esquema XML es un documento XML que contiene una descripción formal de lo que comprende un documento XML válido. Su objetivo es definir una clase de documentos XML [38], estableciendo qué elementos puede contener, cómo están organizados y qué atributos y de qué tipo pueden ser estos elementos.

Los requisitos perseguidos por los esquemas XML son los siguientes [13]:

- Usar una sintaxis XML y ajustarse a las DTD que validan esquemas XML. Las DTD utilizadas deben estar ligadas al analizador que valida esquemas XML.
- Tener la máxima libertad para soportar tipos de datos a la hora de especificar elementos y atributos.

- Utilizar espacios de nombres

Debido a que uno de los objetivos de XML es facilitar el intercambio de información, es necesario que los documentos se diseñen para ser compartidos, por esta razón se hace uso de los *espacios de nombres*. En XML, los espacios de nombres (*namespace*) tienen dos propósitos [15]:

1. Distinguir entre elementos y atributos de distintos vocabularios con distintos significados que puedan compartir el mismo nombre.

2. Agrupar todos los elementos y atributos relacionados de una sola aplicación XML para que el software pueda reconocerlos con facilidad.

Las DTD no proporcionan un soporte explícito para los espacios de nombres, en cambio un esquema XML usa espacios de nombres internamente para diversos propósitos. El vocabulario del esquema XML está en su propio espacio de nombres y los componentes usados dentro del esquema XML (grupos, atributos y tipos de datos) también pueden tener espacios de nombres [15].

Una de las tareas más importantes al construir un esquema XML consiste en determinar los espacios de nombres involucrados, ya que al poder existir varios en un mismo esquema XML, la forma como éstos se declaren determina la capacidad para incluir nuevos esquemas XML (o sus elementos) en un documento [13]

Un esquema XML es un documento XML que incorpora declaraciones y definiciones de los elementos y atributos que se van a representar en el documento instancia. Una vez declarados, los elementos y atributos pueden ser objeto de referencias dentro de distintos contextos o ámbitos [13]. Todo esquema XML contiene un solo elemento raíz `xsd:schema`. Este elemento contiene declaraciones para todos los elementos y atributos que pueden aparecer en un documento instancia válido.

Es importante señalar que en un esquema XML las declaraciones de elementos, atributos, etc. y la definición de los tipos de datos que soporta cada documento no se diferencian entre ellos. Para definir la estructura y los tipos de datos que soporta cada documento instancia, se usan los elementos de tipo simple (`simpleType`) y tipo complejo (`complexType`).

Un tipo de dato simple puede ser predefinido (incorporado al espacio de nombres de los esquemas XML) o derivado, es decir, definido por el autor del esquema XML. Entre los tipos de datos predefinidos se distinguen los tipos primitivos como `cadena`, `entero`, `decimal`, etc.

A partir de los datos simples existe la posibilidad de obtener nuevos tipos de datos simples, ya sea por definición o por derivación. Para definir nuevos datos simples se usa el elemento `SimpleType` que constituye la representación de un tipo simple en un documento, identificándolo con un nombre[5] [13].

En un esquema XML, los elementos, atributos y la mayoría de sus componentes pueden ser globales o sólo estar referidos a otros componentes del mismo (locales). Un componente global puede referenciar a componentes globales de uno o más espacios de nombres y utilizarse en cualquier documento instancia del esquema XML, cosa que no tiene sentido en un componente local [13]. Los tipos de datos complejos (`ComplexType`) pueden integrar distintos elementos mediante los elementos `xsd:element` y `xsd:attribute`.

El elemento `xsd:element`

Todos los elementos que aparecen en un documento instancia del esquema XML deben aparecer en éste con un elemento `xsd:element`, cuyos atributos más sigificativos son [13]:

- **`maxOccurs`** y **`minOccurs`**. Determinan el número máximo y mínimo de veces que el elemento puede aparecer en un documento instancia. Sólo se aplica a elementos de tipo local.
- **`name`**. Especifica el nombre usado para referenciar un tipo de elemento tanto en el esquema XML al que pertenece como en un documento instancia.
- **`ref`**. Hace referencia a un elemento global ya declarado. Es mutuamente excluyente con `name`.
- **`type`**. Especifica el tipo de dato del elemento, con un valor nombre que hace referencia a un tipo global, simple o complejo.

[5] La sintaxis a seguir para la construcción de tipos de datos simples y complejos se describe en la especificación de esquemas XML

El elemento `xsd:attribute`

El elemento `xsd:attribute` permite declarar los atributos de un elemento. Este elemento tiene a su vez una serie de atributos que establecen restricciones a las propiedades de este elemento, su alcance depende de su ubicación dentro del esquema XML. Por ejemplo, un atributo global se declara como hijo del elemento raíz `xsd:schema`.

Los atributos de un elemento `xsd:attribute` son muy parecidos a los del elemento `xsd:element`, la diferencia más importante es que los atributos son siempre de tipo simple.

Esquema XML para un diccionario de sinónimos

Se propone un esquema XML para representar diccionarios de sinónimos.

```
<xsd:schema
        xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <xsd:annotation xml:lang="es">
      Ejemplo de diccionario de sinonimos
   </xsd:annotation>
   <xsd:element name="diccionario"
                   type="tipoDiccionario"/>
     <xsd:complexType name="tipoDiccionario">
        <xsd:element name="palabra"
                   type="tipoPalab"/>
     </xsd:complexType>
     <xsd:complexType name="tipoPalab">
        <xsd:element name="sinonimo"
                   type="tipoSin" minOccurs="0"/>
           <xsd:attribute name="valor"
                type="xsd:string" use="required"/>
     </xsd:complexType>
     <xsd:complexType name="tipoSin">
        <xsd:attribute name="valor"
                type="xsd:string" use="required"/>
     </xsd:complexType>
</xsd:schema>
```

En resumen, existen muchas capacidades de validación que se pueden realizar con un esquema XML que no se pueden realizar con una DTD, o que mediante una DTD se pueden hacer de manera menos estricta [13], por ejemplo:

- **Tipos de datos**. A través de un esquema XML se pueden controlar los tipos de datos que puede contener un elemento o atributo.
- **Aspectos de restricción**. Establece límites para el valor de los datos dependiendo de su tipo.
- **Cardinalidad**. Controla el número de apariciones permitidas dentro de un elemento.
- **Opción**. Limita los valores a los de una lista de valores dada.
- **Secuencia**. Define el orden en el que se pueden utilizar los elementos.
- **Valores predeterminados**. Proporciona valores que se utilizan cuando no se especifca ningún otro valor.

Entre las características de los esquemas XML se encuentra:

- Agrupación `xsd:group`. Permite la agrupación de elementos y atributos para hacer referencias a ellos.
- Incluir esquemas `xsd:include`. Es posible incluir otros esquemas con el mismo espacio de nombres.
- Importar esquemas `xsd:import`. Para incluir elementos de esquemas con distintos espacios de nombres.
- Redefinir esquemas `xsd:redefine`. Similar a `include` pero permite modificar los elementos incluidos.

XML está desempeñando un papel cada vez más importante en el intercambio de datos entre aplicaciones. Aunque en esta sección se ha descrito de manera breve la estructura y sintaxis de los esquemas XML, es posible encontrar toda la documentación respectiva en la página de W3C `http://www.w3.org/XML/Schema`.

3.4. Procesamiento de XML

Actualmente se ha incrementado el número de aplicaciones que utilizan XML para intercambiar, transmitir y almacenar datos, destacando aquellas de extracción de información sobre documentos XML [21]. Como se mencionó anteriormente, una de las ventajas de los documentos XML es que básicamente son texto y se pueden utilizar editores de texto para crear o modificar documentos XML.

Es posible procesar un documento XML de diferentes maneras: considerando el documento como un texto sin formato, como un flujo de eventos, como un árbol o como una serie de cualquier otra estructura. Actualmente existen muchas herramientas para procesar documentos XML. Por ejemplo, para extraer datos de un documento XML se pueden utilizar las API SAX[6] y DOM[7], basadas en eventos y árboles respectivamente.

Existen algunos lenguajes de consulta sobre colecciones de datos XML como XPath y XQuery. Estos lenguajes permiten localizar datos específicos en un documento XML sin la necesidad de conocer su estructura y devolver todos los elementos que cumplen ciertas condiciones de consulta. En esta sección se dará una descripción de estos lenguajes de consulta así como de las API SAX y DOM mencionadas anteriormente.

3.4.1. Procesamiento basado eventos

Cuando un analizador XML lee un documento se desplaza desde el inicio hasta el final del mismo, puede detenerse para recuperar recursos externos como una DTD pero construye un conocimiento del documento a medida que se desplaza por él. Los analizadores basados en eventos informan de eventos de análisis incrementales a medida que se producen [15].

Algunos eventos de análisis son la lectura de etiquetas de inicio de un elemento, la lectura del contenido de un elemento y la lectura de las etiquetas finales del elemento. La lista y estructura de eventos puede hacerse más compleja a medida que se añaden opciones como espacios de nombres, atributos, comentarios,

[6] Simple API for XML (SAX)
[7] Document Object Model (DOM)

instrucciones de procesamiento y entidades; sin embargo, el mecanismo básico es bastante sencillo y generalmente muy eficiente [15].

Un aspecto importante del procesamiento basado en eventos es que los documentos XML pueden ser muy grandes, ya que no es necesario guardar estos datos en memoria. La API basada en eventos más usada para procesar documentos XML es SAX (*Simple API for XML*). Originalmente, SAX se definió como una API Java y estaba principalmente destinada para analizadores escritos en Java; actualmente es usada por muchos lenguajes de programación orientados a objetos como C++.

3.4.2. Procesamiento basado en árboles

En ocasiones, para extraer datos de un documento XML es más fácil descomponer este proceso en dos fases [13]:

1. Construir mediante una API una estructura de datos en forma de árbol que describa el documento.

2. Buscar los datos en la estructura creada en la primera fase.

Al trabajar con un modelo de árbol, todo el documento XML representado se encuentra siempre disponible, con el inconveniente de que puede ocupar una gran cantidad de memoria. Este modelo de uso de memoria hace que el procesamiento basado en árboles no sea apropiado para aplicaciones que tratan con documentos muy grandes o que necesitan ejecutar algún procesamiento intermedio sobre un documento antes de analizarlo completamente [15].

DOM es la API basada en árboles más común [15]. En el fondo es un conjunto de interfaces abstractas que se especifican en módulos[8], permitiendo que las implementaciones admitan partes de DOM sin tener que admitirlas todas.

DOM es una de las tecnologías disponibles con mayor soporte para aplicaciones que requieren de un acceso aleatorio a distintas partes del documento XML en momentos diferentes, o aplicaciones que necesitan modificar la estructura de un documento XML en el momento.

[8]Los requerimientos en los que se basan los módulos se encuentran en `http://www.w3.org/TR/DOM-Requirements`

3.4.3. XML Path Language (XPath)

XPath es un lenguaje de consulta de documentos XML que se basa en expresiones de ruta para la localización de elementos [21]. XPath es un lenguaje declarativo que proporciona una sintaxis y un modelo de datos para poder localizar y dirigirse a una parte de un documento XML dado, incorporando algunas funciones propias de un lenguaje de propósito general [13].

Para seleccionar una partes de un documento XML, XPath construye internamente un árbol de nodos llamado *árbol XPath*, en donde las hojas están formadas por valores indivisibles como cadenas, números, valores booleanos, etc.

Los nodos de un árbol XPath pueden ser de siete tipos:

- Raíz
- Elemento
- Atributo
- Texto
- Comentario
- Instrucción de procesamiento
- Espacios de nombres.

Las secciones CDATA, las referencias a entidad y las declaraciones de tipo de documento no forman parte de los nodos de un árbol XPath, ya que XPath opera sobre un documento XML después de que todos estos elementos se hayan combinado en el documento [15].

En el árbol XPath, cada nodo intermedio contiene listas ordenadas de nodos hijos; por ello, además de la raíz, sólo pueden tener hijos los siguientes nodos: elemento, comentario, texto e instrucción de proceso; mientras que los nodos atributo y los nodos espacios de nombres sólo describen a su nodo padre, por lo que se asume que no contienen ningún nodo [13].

Una expresión de ruta en XPath se crea a partir de pasos sucesivos de localización. Cada paso de localización se evalúa como relativo a un nodo determinado en el documento denominado *nodo contexto* [15]. Una ruta XPath selecciona un conjunto de nodos relativos al nodo contexto y devuelve como resultado el conjunto de elementos del XML seleccionados por la ruta.

La sintaxis de XPath está orientada tanto a definir partes del documento como a proporcionar rutas hacia los elementos definidos en el documento XML. XPath utiliza la barra '/' seguida de una lista con los nombres de los elementos hijo, que en su conjunto describen un recorrido a través del documento. Los elementos seleccionados son aquellos que se ajustan al recorrido expresado por la secuencia, que actúa como patrón para identificar los nodos de un documento [13]. La ruta de localización más simple es la que selecciona el nodo raíz del documento, utilizando '/'.

XPath permite definir rutas de ubicación incorporando una serie de abreviaciones que simplifican las rutas de localización de nodos a través de una sintaxis abreviada. Por ejemplo:

- `child::elemento`. Selecciona los hijos con etiqueta `elemento` del nodo contexto
- `child::*`. Selecciona todos los hijos del nodo contexto
- `child::text()`. Selecciona todos los hijos que contienen texto del nodo contexto
- `attribute::*`. Selecciona todos los atributos del nodo contexto
- `descendant::elemento`. Selecciona todos los hijos del nodo contexto con etiqueta `elemento`
- `/`. Selecciona el elemento raíz del documento XML
- `child::elemento[attribute::valor="valorX"]`. Selecciona los hijos con etiqueta `elemento` del nodo contexto que además tengan un atributo `valor=ValorX`
- `child::*[self::capitulo or self::apendice]`. Selecciona los hijos del nodo contexto con etiqueta `capitulo` ó `apendice`.

Sea el documento XML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <diccionario>
    <palabra valor="triunfo">
      <sinonimo valor="victoria"/>
      <sinonimo valor="conquista"/>
    </palabra>
    <palabra valor="amanecer"/>
    <palabra valor="amplificar">
      <sinonimo valor="ampliar"/>
    </palabra>
  </diccionario>
```

Y sea la expresion XPath:

```
/diccionario/palabra/sinonimo
```

La expresión devuelve todos los elementos `<sinonimo>` de cualquier elemento `<palabra>`, dando como resultado:

```
 <sinonimo valor="victoria"/>
 <sinonimo valor="conquista"/>
 <sinonimo valor="ampliar"/>
```

Los predicados en XPath se incluyen dentro de la expresion de ruta XPath utilizando corchetes, por ejemplo en la siguiente expresion:

```
/diccionario/palabra[@valor="triunfo"]/sinonimo
```

La expresión regresa únicamente los elementos `<sinonimo>` de los elementos `<palabra>` que tengan el atributo `valor` en donde `valor="triunfo"`. Aplicado al documento XML que representa el diccionario de sinónimos se obtiene:

```
 <sinonimo valor="victoria"/>
 <sinonimo valor="conquista"/>
```

Cada vez que se utiliza / se selecciona un subconjunto de elementos de un conjunto dado por un nodo contexto, de manera que el conjunto cada vez se hace más pequeño. Cuando se usa / se dice que se utiliza un `eje`. Cada paso de localización XPath se desplaza por un eje desde el nodo contexto [15].

Para verificar todas las características de XPath se recomienda leer las especificaciones[9].

Entre las características que posee este lenguaje de consulta se encuentran:

- La selección de predicados puede seguir cualquier paso en la ruta.
- Una expresión XPath puede saltar varios niveles de nodos mediante el uso de //.

3.4.4. XQuery

XQuery es un lenguaje diseñado para realizar consultas sobre colecciones de datos expresadas en XML. Su principal función es extraer información de un conjunto de datos organizados como un árbol n-ario de etiquetas XML. En este sentido XQuery es independiente del origen de los datos [17].

XQuery es una extensión de XPath, por lo que cualquier expresión sintácticamente válida en XPath que además se ejecute de manera exitosa tanto en XPath como en XQuery, devuelve el mismo resultado en ambos lenguajes [18]. Las expresiones de XQuery pueden ser construidas a partir de palabras clave, símbolos y operandos. En general, los operandos de una expresión son otras expresiones y dentro de las expresiones se pueden identificar las *literales* y las *variables*:

- Una *literal* representa un valor atómico (entero, decimal o cadena de caracteres, por ejemplo) y es la expresión más sencilla de XQuery.
- Una *variable* es un nombre que empieza con el signo '$'que se asocia a un valor y que se usa dentro de una expresión para representarlo.

[9]Las especificaciones de XPath están disponibles en la página de W3C `http://www.w3.org/TR/xpath`

Los documentos XML pueden tener estructuras muy complejas, por lo que las expresiones deben estar bien definidas para que sea posible evaluar cualquier expresión. XQuery puede procesar documentos XML con tipos de datos simples y complejos, además de poder procesar esquemas XML y DTD y también debe ser capaz de trabajar sin ellos. Cuando XQuery se enfrenta a tipos de datos derivados de esquemas XML, debe estar en condiciones de manejar los tipos predefinidos accesibles en toda la consulta y los tipos importados para la consulta desde un esquema XML específico [13].

Para simplificar la semántica de XQuery se definen operadores con operaciones implícitas. Por ejemplo, los operadores aritméticos como (+) aplicados a un elemento, automáticamente extraen su valor numérico; por otro lado, los operadores de comparación como (=) aplicados a una secuencia de valores, automáticamente iteran buscando valores que satisfagan la condición de comparación [13].

Es importante señalar que XQuery se centra más en la recuperación de información que en la actualización de documentos existentes; además, como una consulta XQuery tiene como entrada y salida documentos XML, no se trata sólo de un lenguaje de consulta que opera sobre documentos sino que también genera documentos XML a demanda de la consulta correspondiente [13].

XQuery difiere de SQL porque en el modelo de datos XML existen los conceptos de jerarquía y orden que no están presentes en el modelo relacional de bases de datos. En XQuery, el orden en el que se encuentran los elementos dentro del documento XML es importante y determinante, no así en el modelo relacional sobre el que se sustenta SQL [17].

Una consulta XQuery está formada de dos partes: un prólogo (*query prolog*) y un cuerpo (*query body*). El prólogo consta de una serie de declaraciones que definen el entorno para el procesado del cuerpo. El cuerpo de una consulta XQuery consta de una expresión cuyo valor proporciona el resultado de la consulta. El prólogo se usa cuando el cuerpo depende de los espacios de nombres, esquemas XML o funciones, cuando esto sucede la consulta depende en gran medida de lo especificado en él [13].

Para identificar los datos de entrada existen dos funciones básicas:

- **`doc()`**, que devuelve un documento completo identificándolo con una URI. Por ejemplo: `doc(diccionario.xml)` .

- **`collection()`** que devuelve cualquier secuencia de nodos asociada a una URI.

XQuery utiliza expresiones basadas en expresiones de ruta XPath para localizar datos en un documento XML. Primero se debe determinar el documento en el que se efectuará la búsqueda con la función `doc()`. Si la consulta requiere el uso de variables, éstas deben definirse en el prólogo de la consulta XQuery, de esta forma son accesibles desde cualquier punto de la consulta [13].

Sentencias FLWOR

Cuando en una consulta se combinan los datos de una o más fuentes, una vez localizados los nodos solicitados por la consulta XQuery, se debe crear el resultado reestructurando los datos para satisfacer la consulta mediante el uso de constructores. Para efectuar estas tareas existe una serie de sentencias llamadas *FLWOR* (leído como *flower*). La capacidad de construir nuevos objetos XML es una de las funcionalidades más importantes de XQuery [13].

FLWOR es un nombre que proviene las siglas de *For*, *Let*, *Where*, *Order* y *Return*. Estas sentencias juegan un papel similar al de las instrucciones `SELECT-FROM-WHERE` de SQL, al asociar valores a las variables para obtener resultados [13]. El uso de estas sentencias debe tener un orden y seguir un conjunto de reglas para que una consulta XQuery tenga sentido.

En XQuery, el término *tupla* se refiere a una combinación de variables asociadas a una expresión FLWOR [13]. En la Figura 3.2 se muestra gráficamente el orden el el que se ejecuta cada sentencia FLWOR. Es importante señalar que la sentencia **`Return`** es obligatoria.

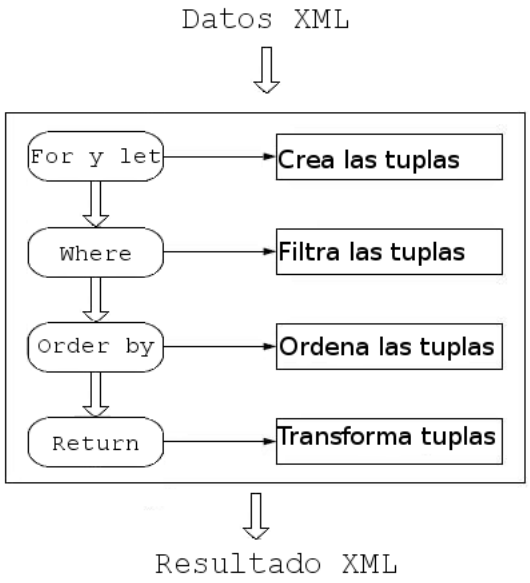


Figura 3.2: Orden en el que se ejecutan las sentencias FLWOR.Tomado de [17]

Las sentencias **`For`** y **`Let`** permiten crear tuplas que serán usadas durante la consulta, estas tuplas pueden usarse cuantas veces se desee incluso dentro de otras sentencias. La sentencia **`where`**, a diferencia de `For` y `Let`, sólo puede declararse una vez en cada consulta. Esta sentencia elimina tuplas que no satisfacen una determinada condición, de manera que `Return` sólo evalúa las tuplas que regresa la sentencia `where`. Por ejemplo, sea el siguiente documento XML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <diccionario>
    <palabra valor="triunfo">
      <sinonimo valor="victoria"/>
      <sinonimo valor="conquista"/>
    </palabra>
    <palabra valor="amanecer"/>
    <palabra valor="amplificar">
      <sinonimo valor="ampliar"/>
    </palabra>
  </diccionario>
```

La consulta XQuery

```
<diccionario>
{for $b in doc(urlBase)//palabra
  where ($b/@valor="victoria") or
   (some $a in $b/sinonimo satisfies
              ($a/@valor="victoria"))return $b}
 </diccionario>
```

regresa lo siguiente:

```
  <diccionario>
    <palabra valor="triunfo">
      <sinonimo valor="victoria"/>
      <sinonimo valor="conquista"/>
    </palabra>
  </diccionario>
```

Este resultado se debe a que, a través de la consulta, se busca la palabra `victoria` en el atributo `valor` de los elementos `palabra` y en cada uno de los subelementos `sinonimo`.

La sentencia **`Order by`** ordena las tuplas resultantes según un criterio dado antes de evaluar la sentencia `Return`. En una consulta XQuery sólo puede existir una sentencia `Order by`. La sentencia **`Return`** construye el resultado de la consulta para una tupla dada, después de haber sido filtrada por la sentencia `where` y, si se solicitó, ordenada por la sentencia `Order by`.

No es necesario que una consulta XQuery contenga alguna sentencia FLWOR. Como se mencionó anteriormente, una expresión XPath es una consulta válida en XQuery. Por ejemplo, para el documento XML siguiente:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <diccionario>
    <palabra valor="triunfo">
      <sinonimo valor="victoria"/>
      <sinonimo valor="conquista"/>
    </palabra>
  </diccionario>
```

La expresión XPath:

```
/diccionario/palabra/sinonimo
```

es una consulta XQuery válida.

Expresiones condicionales

La sentencia `where` de una consulta permite filtrar las tuplas que aparecerán en el resultado, mientras que una expresión condicional permite crear como resultado otra estructura de nodos que dependa de los valores de las tuplas filtradas.

En XQuery se utilizan expresiones condicionales **`if-then-else`**. A diferencia de la mayoría de los lenguajes, la cláusula `else` es obligatoria y debe aparecer siempre en la expresión condicional. El motivo de esto es que toda expresión en XQuery debe devolver un valor. Si no existe ningún valor a devolver al no cumplirse la cláusula if, devolvemos una secuencia vacía con *else()* [17]. Por ejemplo, sea el documento XML:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <diccionario>
    <palabra valor="triunfo">
      <sinonimo valor="victoria"/>
      <sinonimo valor="conquista"/>
    </palabra>
    <palabra valor="amanecer"/>
    <palabra valor="amplificar">
      <sinonimo valor="ampliar"/>
    </palabra>
  </diccionario>
```

Y sea la sentencia XQuery:

```
<palabras>
{for $b in doc(urlBase)//palabra
 let $c:=$b/sinonimo
 return
 if(count($c)=0)
  then <sin_sinonimos>{$b/@valor}</sin_sinonimos>
  else()
 </palabras>
```

regresa lo siguiente:

```
  <palabras>
    <sin_sinonimos>amanecer</sin_sinonimos>
  </palabras>
```

Ya que se buscan los elementos `<palabra>` que no tengan subelementos `sinonimo`, esto se da en la condición `if(count($c)=0)`. La consulta sólo regresa el valor del elemento `palabra` si cumple con la condición. Debido a que en XQuery toda sentencia **`if`** debe ir acompañada de la sentencia **`else`**, se incluye la línea `else()` para que no se haga alguna acción cuando el elemento `palabra` tenga subelementos `sinonimo`.

3.5. Bases de datos nativas XML

Un documento XML es autodescriptible y universal, por lo que su uso se ha extendido en diversas aplicaciones, sobre todo para aquellas que están enfocadas al intercambio de información; sin embargo, existen desventajas de usar XML en lugar de bases de datos para el almacenamiento de grandes cantidades de datos [11]. Por ejemplo:

- En los documentos XML pueden existir elementos completamente duplicados. Lo que afecta la exploración y análisis del documento además de que el tamaño del documento se incrementa de manera innecesaria para su exploración.

En una base de datos relacional, no pueden existir duplicados de registros exactamente iguales, ya que los sistemas manejadores de bases de datos no permiten este tipo de inconsistencias. En el caso de los archivos XML no hay manera de garantizar la exclusión de elementos duplicados, ya que es posible manipularlos a través de un editor de texto.

Contener o no elementos duplicados en una base de datos relacional, depende en gran medida del diseño del esquema conceptual de la base y de los métodos de inserción de registros.

- La comparación física del tamaño entre las bases de datos nativas XML y las bases de datos relacionales no es práctica.

Por definición, una *Base de Datos Nativa* (NXD) por sus siglas en inglés, puede ser tanto un documento XML como un tipo de dato XML. Un tipo de dato XML es un tipo especializado contenido en una base de datos relacional. Una base de datos nativa XML es escencialmete cualquier método de almacenar datos de XML como un documento XML [11]. Un documento XML procesado por un navegador (*browser*) es una base de datos nativa XML. Además utilizando tipos de datos XML en bases de datos relacionales, como Oracle o Postgres es posible simular una base de datos nativa almacenando el XML en un campo de tipo texto.

Las bases de datos nativas XML respetan la estructura de los documentos y es posible hacer consultas sobre dicha estructura para recuperar el documento tal y como fue insertado originalmente. Ejemplos de bases de datos son:

- XQengine
- eXist
- Xindice

Las bases de datos nativas XML, como las otras bases de datos soportan transacciones, acceso multiusuario, lenguajes de consulta, etc. Las BD nativas XML se caracterizan por emplear como unidad lógica de almacenamiento el documento XML y preservar el orden del documento, las instrucciones de procesamiento, los comentarios, las secciones CDATA y las entidades además de soportar lenguajes de consulta XML.

El almacenamiento de los documentos en las bases de datos nativas XML es en colecciones. Las colecciones juegan en las bases de datos nativas el papel de las tablas en las BD relacionales. Los documentos se suelen agrupar, en función de la información que contienen, en colecciones que a su vez pueden contener otras colecciones.

La mayoría de las BD XML soportan uno o más lenguajes de consulta como XQuery. Es posible permitir la creación de índices para acelerar las consultas realizadas frecuentemente. A cada documento XML se le asocia un identificador único por el que será reconocido dentro del repositorio.

3.6. eXist

eXist es un esfuerzo de Open Source por tener un desarrollo de un sistema de base de datos nativa XML, el cual puede ser fácilmente integrado en aplicaciones que utilizan XML en una variedad de escenarios posibles. La base de datos está completamente desarrollada en Java y puede ser implementada de distintas formas, ya sea en un proceso de servidor, adentro del motor de un servlet o directamente embebido en un aplicación [19].

eXist proporciona un almacenamiento que se encuentra en colecciones jerárquicas. Usando una sintaxis extendida de XPath, los usuarios pueden hacer consultas en distintas partes de una colección jerárquica o incluso a todos los documentos contenidos en la base de datos XML.

El motor de búsqueda de eXist implementa de manera eficiente el procesamiento de consultas basadas en índices, utilizando algoritmos de unión de rutas (*path join*). Un gran número de consultas de expresión de rutas es procesado únicamente con información del índice.

eXist provee un número de extensiones a XPath para procesar consultas de texto completo de manera eficiente, incluyendo búsquedas de plabras clave, consultas de proximidad de términos de búsqueda o expresiones regulares.

3.6.1. Indexado y almacenamiento XML

Un gran número de implementaciones de lenguajes de consulta XML están basadas en recorridos de árbol para evaluar las expresiones de ruta; sin embargo, se convierte en algo muy ineficiente cuando se trata de colecciones de documentos muy grandes. Por ejemplo, consideremos una expresión XPath que selecciona los títulos de todas la figuras en una colección de libros `/libro//figura/titulo`.

En un recorrido convencional, el procesador de la consulta tiene que seguir cada ruta que empiece con `libro` para entonces checar los descendientes `figura`, porque no hay manera de determinar la posible ubicación del elemento `figura` por anticipado. Esto implica que un gran número de nodos que no son `figura` tengan que ser accesados para probar si el nodo es el elemento que se busca o si la etiqueta es `figura`.

Por lo anterior, las estructuras basadas en índice son necesarias para trabajar eficientemente con consultas muy grandes y sin restricción en las colecciones de documentos.

Para hacer más rápido el proceso de expresiones de ruta basadas en relaciones estructurales, un esquema de indexado debe soportar la identificación rápida de tales relaciones entre dos nodos como las relaciones padre-hijo. La necesidad de recorrer un documento de un subárbol debe estar limitada a casos especiales donde la información contenida en los índices no es suficiente para procesar la expresión.

Un esquema de numeración asigna un identificador único a cada nodo en el documento del árbol lógico, por ejemplo: Recorre el documento del árbol en orden de niveles y debe proporcionar mecanismos para determinar de manera óptima la relación estructural entre dos nodos e identificar todas las ocurrencias en el documento o en la colección de documentos. Los identificadores generados son usados en los índices como una referencia al nodo actual.

El esquema de numeración implementado en eXist provee una extensión al esquema de numeración que modela un árbol de documentos como un árbol de orden k (k-ario) llamado XISS, en el cual un identificador único es asignado a cada nodo mediante un recorrido de nivel. Ver Figura 3.3

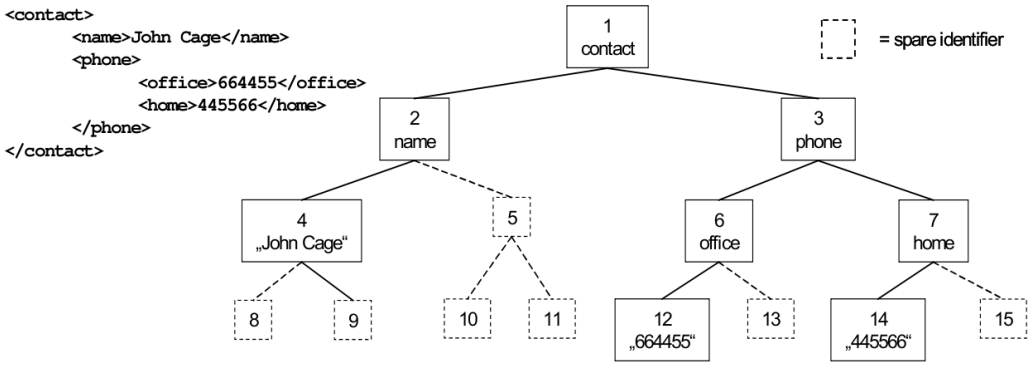


Figura 3.3: Arbol XISS. Tomado de [19]

Para accesar de manera fácil al nodo padre de cada elemento, el árbol XISS debe cumplir con la condición de ser un *árbol completo*. De manera que varios nodos son agregados para que se cumpla la fórmula: $parent_i = [(i-2)/k] + 1$, lo que significa un acceso rápido a los ancestros de cada nodo.

A pesar de la ventaja de accesos que se tienen con el árbol XISS, se encuentra una limitación en el tamaño del documento. Para corregir este problema en eXist se borra parcialmente la restricción de completez para crear un esquema alternativo: El documento ya no es visto como un árbol de orden k completo, en lugar de esto el número de hijos que un nodo puede tener es calculado para cada nivel del árbol, de tal modo que para dos nodos x y y de un árbol el tamaño(x)=tamaño(y) si nivel(x)=nivel(y), donde el tamaño(n) es el numero de hijos del nodo n y el nivel(m) es la longitud de la ruta desde el nodo raíz del árbol al nodo m [19]. Ver Figura 3.4.

Con este esquema alternativo de eXist, se permite un indexado de documentos más grandes que con el esquema XISS, ya que se necesitan menos identificadores vacíos que son colocados en el esquema original para garantizar la completez del árbol. Esto también hace posible dejar identificadores entre nodos existentes para evitar un reordenamiento frecuente de identificadores de nodo en actualizaciones posteriores del documento. Es importante mencionar que no es posible realizar actualizaciones parciales de un documento XML en eXist, ya que está adaptado a

documentos estáticos y que rara vez son actualizados. Ya que el indexado se hace completamente cada vez.

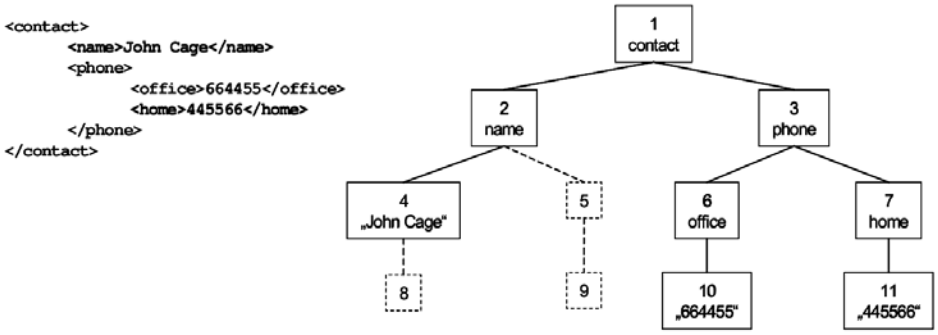


Figura 3.4: Arbol de eXist. Tomado de [19]

eXist fue diseñado para proveer una implementación completa del lenguaje de consulta XPath y su esquema de numeración se enfoca en un subconjunto limitado de consultas de expresión de ruta, concentrándose en un soporte eficiente para un nodo y sus hijos de manera que para accesar al nodo padre de un elemento, simplemente se calcula su identificador único y se busca en el índice [19].

Con el esquema de numeración de eXist, cualquier nodo en un documento XML puede servir como un punto de inicio para una expresión XPath. Por ejemplo: los nodos seleccionados por una primera expresión XPath pueden ser procesados por una segunda expresión, ésto es una característica importante porque permite que múltiples consultas de expresión de ruta sean embebidas dentro de la expresión XQuery

3.6.2. Organización de índices y datos

Actualmente eXist utiliza cuatro archivos de índices en el núcleo de la base de datos XML nativa.

- `collections.dbx`. Este archivo administra la jerarquía de la colección.
- `dom.dbx`. Colecciona los nodos en un archivo paginado y asocia identificadores de nodo únicos a los nodos actuales.

- `elements.dbx` Indexa los elementos y los atributos
- `words.dbx` Mantiene un seguimiento de la ocurrencia de palabras y es utilizado por las extensiones de la búsqueda de texto completo.

Todos los índices están basados en la estructura de los árboles B+, los índices de los elementos, atributos y palabras clave están organizados por colección y no por documento. Por ejemplo, todas las ocurrencias de un elemento `seccion` dentro de una colección serán almacenadas como una sola entrada en el índice de elementos, lo que permite mantener pequeño el número de páginas internas de los árboles B+ y optimizar las consultas sobre colecciones completas. [19].

Los usuarios normalmente consultan sobre colecciones completas o sobre varias colecciones vistas como una sola. En este caso sólo se requiere hacer la búsqueda sobre un índice relevante que refleje los datos de toda la colección, ésto provoca una ganancia en el rendimiento de consultas que se ejecutan sobre múltiples colecciones.

El archivo `collections.dbx` administra la jerarquía de las colecciones y mapea los nombres de una colección en objetos de la colección. Las descripciones del documento son almacenadas siempre con el objeto colección al que pertenecen y se le asigna un identificador único a cada colección y documento durante el indexado. [19].

El archivo `dom.dbx` representa el componente central de la arquitectura de eXist. Consiste en un archivo en el cual todos los nodos del documento se almacenan de acuerdo al modelo de objetos descrito en el documento W3C [20]. El almacenamiento de los datos es respaldado por un árbol `B+` multiraíz en el mismo archivo.

Sólo los elementos de un nivel superior son indexados por el arbol B+. Los atributos, los nodos de texto y los elementos de niveles bajos o inferior de la jerarquía de nodos del documento son escritos en las páginas de datos sin añadir una llave en el árbol B+. El acceso a este tipo de nodos se obtiene recorriendo el padre del nodo disponible más cercano que se encuentra en el árbol, ya que el acceso directo a estos nodos es raramente requerido. El motor de búsquedas procesará la mayor parte de las expresiones XPath sin accesar a `dom.dbx`. Ver Figura 3.5.

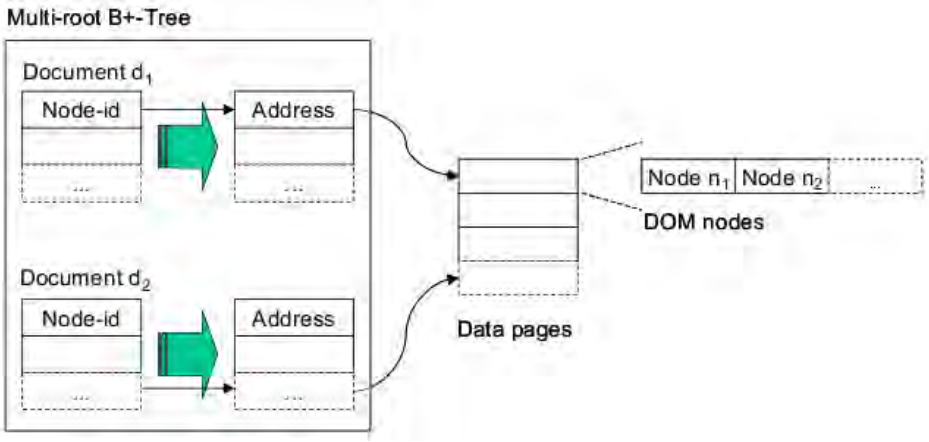


Figura 3.5: Arbol B+ de multi-raíz. Tomado de [19]

La implementación DOM se basa completamente en el esquema de numeración para determinar las relaciones entre los nodos, por ejemplo: para mantener el padre de un nodo, el identificador único de éste es calculado a partir del identificador del nodo hijo mediante la búsqueda de índices. De esta manera el tamaño de almacenamiento de un documento `dom.dbx` es más pequeño que el tamaño de la fuente original para documentos grandes.

Debido a que los nodos son almacenados en el orden que aparecen en el documento, se requiere de un solo índice de búsqueda para serializar un documento, a partir del cual eXist genera un flujo de eventos SAX. Ver figura 3.6

Los nombres de elementos y atributos son mapeados a identificadores de nodo únicos en el archivo `elements.dbx`. Cada entrada en el índice consiste de una llave (par de `(id-coleccion, id-nombre)`) y un arreglo de valores que contiene una lista ordenada de identificadores de documento e identificadores de nodo, los cuales corresponden a elementos y atributos.

El archivo `words.dbx` corresponde a un índice invertido que representa una estructura de datos común y es usada para asociar una palabra o frase con el conjunto de documentos en los cuales ha sido encontrada y la posición exacta dónde se encontró. El índice invertido de eXist, difiere de los sistemas tradicionales de

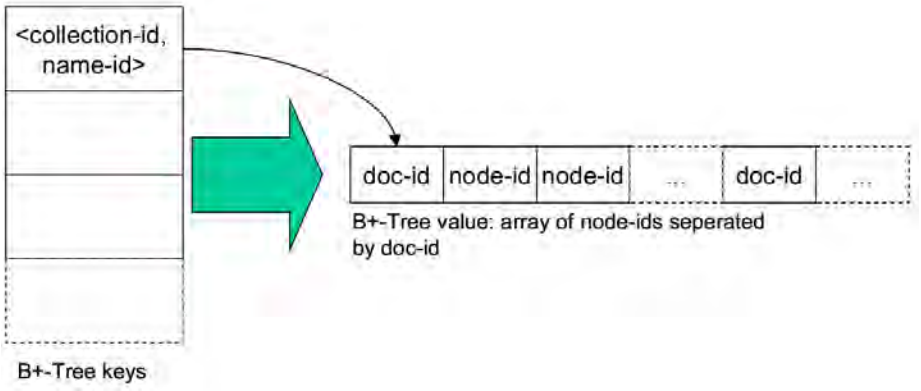


Figura 3.6: Paginación en eXist. Tomado de [19]

recuperacion de información en los que, en lugar de almacenar la posición de la palabra, se usan únicamente identificadores nodo para dar seguimiento a la ocurrencia. Por default, eXist indexa todos los nodos de texto y los valores atributo identificando las palabras clave del texto. En `words.dbx`, las palabras clave extraídas son mapeadas a una lista ordenada de documento e identificadores únicos de nodo. El archivo sigue la misma estructura que `elements.dbx` usando como llaves los pares `(id-coleccion, id-llave)`. Cada entrada en la lista de valores apunta a un texto o atributo nodo donde la palabra clave ocurre. [19]

3.7. Resumen

En este capítulo se describe de manera breve lo que es el lenguaje de marcas extensible XML (*eXtensible Markup Language*). XML es muy útil como formato de datos cuando debe haber comunicación entre aplicaciones o integración de información de diversas fuentes, porque su estructura lo hace autodescriptible y relativamente legible al usuario, además de que son textos que se pueden leer con cualquier herramienta que pueda leer un archivo de texto.

Con XML es posible representar datos de bases de datos relacionales, así como muchas clases de datos estructurados [21], pero existen diferencias entre los documentos XML y bases de datos relacionales que se mencionan a lo largo de este capítulo.

Un documento XML debe estar bien formado para ser considerado un documento XML. A lo largo de este capítulo se presenta una breve descripción a las Definiciones de Tipo de Documento(DTD) y a los esquemas XML (*XML Schema*) para que los documentos XML sean validados; además se mencionan algunas técnicas de procesamiento de documentos XML y se da una breve introducción a las API SAX y DOM.

Para extraer datos de colecciones XML, se resalta la necesidad de contar con lenguajes de consulta que eviten al usuario conocer de manera previa la estructura de los documentos. Se dedica una sección a los lenguajes de consulta XML *XPath* y *XQuery*. XQuery es una extensión de XPath por lo que cualquier expresión sintácticamente válida en XPath, que además se ejecute de manera exitosa tanto en XPath como en XQuery, devuelve el mismo resultado en ambos lenguajes.

En la sección dedicada a XQuery de este capítulo, se describe de manera general cómo se hacen las consultas en XQuery, además se da una breve introducción a las sentencias FLWOR que permiten combinar datos de diversas fuentes en una consulta y crear el resultado reestructurando los datos.

En este capítulo se le dedica una sección a las bases de datos nativas XML. Una base de datos nativa XML es escencialmente cualquier método de almacenar datos XML . Las bases de datos nativas respetan la estructura de los documentos XML y permiten hacer consultas sobre dicha estructura para recuperar los documentos tal y como fueron insertados originalmente.

Las bases de datos nativas XML, como otras bases de datos soportan transacciones, acceso multiusuario, lenguajes de consulta, etc. En este capítulo se describe de manera breve la base de datos nativa XML *eXist*. eXist utiliza estructuras basadas en índices para trabajar de manera eficiente las consultas muy grandes y sin restricción en las colecciones de documentos.

Capítulo 4

Desarrollo del Sistema

Como se menciona en el título de esta tesis, el objetivo principal de este trabajo es desarrollar un sistema para la estandarización y normalización de una base de datos bibliográfica, para lo cual se utiliza un respaldo de la base de datos bibliográfica de la Biblioteca Digital de la Dirección General de Bibliotecas, UNAM.

Debido a que los datos de las revistas electrónicas contratadas por la Universidad son proporcionados por diferentes proveedores, son datos que representan el mayor reto para ser estandarizados debido a que se requiere hacer una integración de los mismos. En este trabajo únicamente se consideran las tablas que contienen la información de las revistas electrónicas.

Para lograr los objetivos planteados, el trabajo se dividió en dos etapas:

1. Normalización del esquema conceptual de la base de datos bibliográfica de BiDi para los datos correspondientes a las revistas electrónicas que fueron contratadas por la Universidad a través de diferentes proveedores.

2. Desarrollo de un sistema de software para la estandarización de los datos de tipo `cadena` almacenados en una base de datos relacional.

Como se mencionó en el Capítulo 1, la base de datos con la que se trabajará únicamente almacena los datos referentes a los recursos que proporciona la Biblioteca Digital, cuya estructura se ha ido modificando a lo largo de la historia de BiDi para agregar nuevos recursos para los usuarios o corregir problemas en el

diseño orginal.

La normalización del diseño de la base de datos debe ser un proceso independiente a la estandarización de los valores contenidos en la misma, además de ajustarse a sus características así como a la información que se pretende obtener con los datos almacenados. El objetivo del diseño de las bases de datos relacionales es la generación de un conjunto de esquemas relacionales que permita almacenar la información sin redundancias innecesarias, pero que permita recuperar fácilmente esa información. [21].

En el caso de los datos de las revistas electrónicas contratadas, se debe tomar en cuenta que los datos almacenados cambian periódicamente debido al ciclo de vida de las revistas, por ejemplo:

- Las revistas sufren cambios de manera periódica como: cambio de nombre, de editorial, de proveedor o simplemente ya no se encuentran dentro del tiempo establecido en el contrato y no pueden ser consultadas pues ya no será permitido por el proveedor.
- Debido a que los datos de las revistas son proporcionados por diversos proveedores, es necesario llevar a cabo una integración de datos, evitando agregar registros duplicados resultado de un problema de estandarización de campos entre los diversos proveedores.

La falta de estandarización de valores en la base de datos tiene dos fuentes principales:

1. Problemas de integración de la información proporcionada por los diferentes proveedores de datos, debido a que no hay un estándar en la representación de datos como el nombre del editor.
2. Errores de *dedo* introducidos a la base por el personal de BiDi al hacer cambios a la información de manera manual como actualizaciones o inserciones.

Cualquiera que sea el origen de la falta de estandarización de los valores en una base de datos, ésta afecta la calidad de los datos en la base y puede afectar la confiabilidad de la información que se extrae de la misma.

El proceso de normalización de la base de datos así como la estandarización de valores se describen en las siguientes secciones.

4.1. Normalización de la BD

Se utilizó un respaldo de la base de datos bibliográfica de la Biblioteca Digital de la UNAM, cuya última actualización fue realizada el 29 de junio de 2007[1]. En esa fecha se contaban con 25,024 registros de revistas electrónicas de los cuales sólo un registro se encuentra marcado como `borrado`.
En el esquema original(ver Figura 4.1[2]) se observan 3 tablas principales:

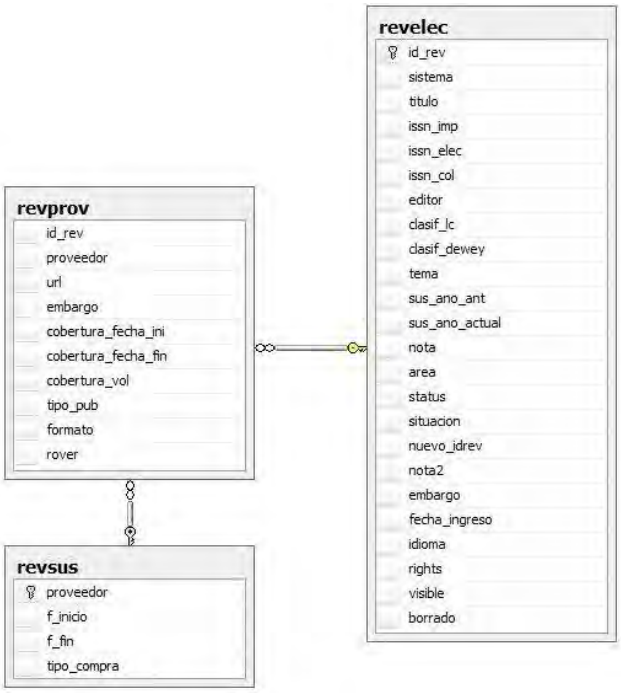


Figura 4.1: Esquema relacional de la base de datos (Revistas electrónicas)

[1]Se acordó con los responsables de BiDi utilizar este respaldo de la base de datos

[2]Para facilitar la lectura y la visualización del esquema, la Figura4.1 es la misma Figura 1.1 presentada en el Capítulo 1.

- **revsus**. Contiene la información de los proveedores y las suscripciones que han establecido con la Universidad.
- **revelec**. Contiene la información de todas las revistas electrónicas contratadas.
- **revprov**. Es la tabla que relaciona a los proveedores con las revistas.

Al analizar los datos almacenados en las tablas, es posible identificar *datos no relevantes*, es decir, aquellos que no son compartidos por todas las revistas y sólo un pequeño porcentaje los contiene; además de que no son útiles para realizar consultas en el sistema de la Biblioteca Digital.

La base de datos tiene almacenada la información de 25,024 revistas electrónicas y sólamente un registro se encuentra marcado como `borrado`, lo que equivale a un 0.003 %. Este registro tiene esta marca desde un respaldo de mayo de 2006, por lo que no es necesario seguir almacenando este dato. De hecho no es necesario almacenar datos de revistas eliminadas.

Al hacer el diseño original de la base de datos no se consideró establecer campos que pueden tomar más de un valor como *multivaluados*, por ejemplo: los editores de una revista. El campo `editor` en la base de datos es de tipo `VARCHAR` con una longitud máxima de 300 caracteres. Cuando una revista tiene más de un editor se pueden realizar dos acciones:

1. Los nombres de ambos editores se escriben en la misma cadena separándolos por un caracter especial, por ejemplo:
   ```
   Universitat de les Illes Balears ; Universitat
   Autonoma de Barcelona
   ```
2. Se almacena sólo el nombre de un editor, aquel que el especialista considere el más relevante.

En el primer caso, al hacer una búsqueda por cualquier editor, el sistema regresará al menos un resultado. En el segundo caso, existe pérdida de información y puede suceder que un usuario no encuentre un registro válido si busca el nombre del editor que no se encuentra almacenado en la base de datos aunque sí sea un editor de la revista.

El campo `nota2` es un campo que está pensado para poner anotaciones pero que no ha sido utilizado en todos los registros de la base de datos, por lo que debe ser eliminado.

Revistas que son publicadas en más de un idioma, presentan el mismo problema que al almacenar el editor: no se consideró un campo multivaluado. En este caso se almacena un valor por cada registro en la base de datos, de manera que los datos almacenados en la base de datos no reflejan la realidad de las publicaciones.

El campo `tema` también debe ser multivaluado; sin embargo sólo se permite almacenar una cadena de caracteres ya que este campo es de tipo `VARCHAR` de una longitud máxima fija de 120 caracteres. Para registrar varios temas, éstos son separados por comas (`,`). Además al ser un campo compuesto es susceptible a varios errores como los que se muestran en la Tabla 4.1

ISSN	**Tema**
0957-4212	Administración y contaduría,Economía
0148-4182	Administración y contaduría, Economía
0013-4120	Administración y contaduría, Inversiones
1087-0148	Administración y contaduría, Inversiones
	Economía,Administración y conatduría
0004-5578	Economía, Administración y contaduría
0120-3592	Economía,Administración y contaduría

Tabla 4.1: Ejemplos de duplicados no exactos en el campo `tema`

En la Tabla 4.1 se muestran ejemplos de valores del campo `tema` que aparecen como únicos porque presentan diferencias tipográficas en su escritura. Al hacer un listado de los temas con la consulta:

```
SELECT DISTINCT(tema) FROM revelec
```

Todas las cadenas de la columna **Tema** que aparecen en la Tabla 4.1 aparecen en el resultado de la consulta porque estos datos no se encuentran estandarizados.

Al analizar la Base de Datos se puede observar que existe la dependencia funcional[3] $editor \rightarrow formato$ ya que todas las revistas proporcionadas por un proveedor tienen el mismo formato.

Finalmente, el campo `clasif_lc` contiene la clasificación que tienen estas revistas en *Library of Congress* las cuales son proporcionadas por el Congreso de los Estados Unidos. Únicamente 2,944 revistas están en el Congreso, lo que significa que un 11.76 % de registros en la base tienen el campo `clasif_lc` distinto de `NULL`; el resto de los registros (88.24 %) no deberían tener este campo.

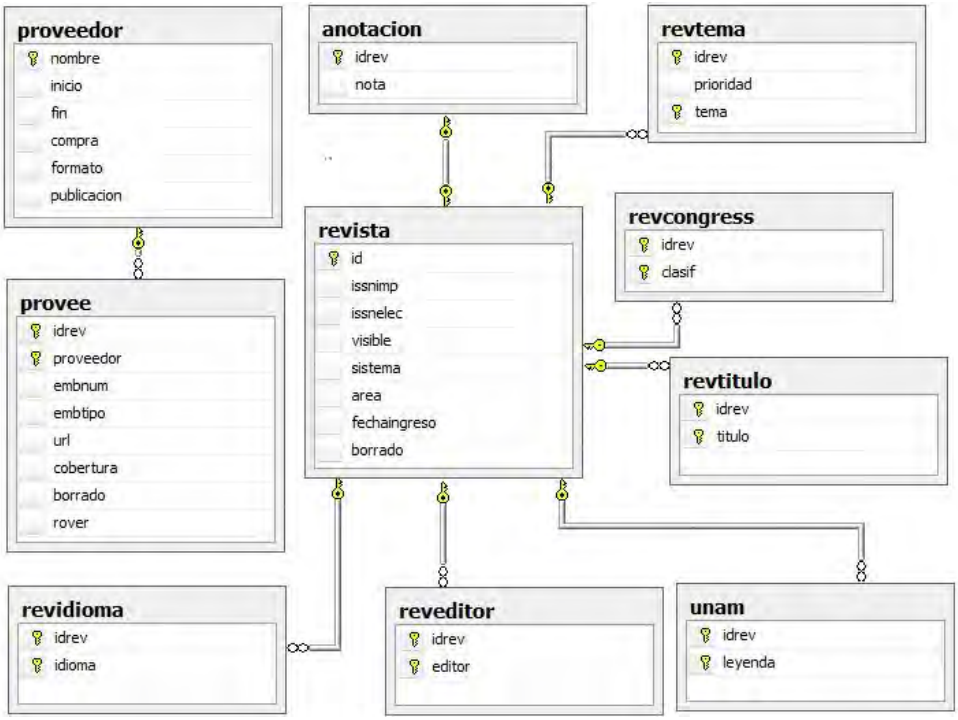


Figura 4.2: Esquema relacional propuesto (Revistas electrónicas)

En conclusión se puede decir que el esquema original de la base de datos no es un sistema adecuado, ya que no implica normalización ni la existencia de va-

[3] Una dependecia funcional es un tipo de restricción que construye una generalización del concepto de clave [21].

lores multivaluados. El esquema que se propone se muestra en la Figura 4.2. Con el esquema propuesto es posible representar valores multivaluados en los campos `titulo`, `idioma` y `tema`; además se separa la información de las revistas del Congreso y las revistas editadas por la UNAM de manera que la cantidad de registros con valores `NULL` disminuye de manera significativa.

En la Tabla 4.2 se muestra un ejemplo de cómo afectó la normalización de la estructura del esquema relacional los datos almacenados en el campo `editor`[4]; se muestra un fragmento del conteo de los editores de las revistas agrupados por nombre, con la consulta:

```
SELECT editor, COUNT(id_rev) FROM revelec GROUP BY editor
```

Editorial	**Total**
ϵ	9799
Brill Academic Publisher	6
Brill Academic Publishers	49
Institute of Electrical and Electronic Engineers	2
Institute of Electrical and Electronics Engineers	59
Institute of Electrical & Electronics Engineers	1
Organisation for Economic Cooperation and Development	6
Organisation for Economic Cooperation & Development	54
Sage Publications	18
SAGE Publications	52
Sage Publications, Inc.	1
Sage Publications (London)	5
Sage Publications, Ltd.	1
NULL	136
(2342 registros)	

Tabla 4.2: Conteo del editores agrupado por nombre en la base de datos con el esquema original.

[4]Para migrar los datos primero se separaron las cadenas con el valor de más de un editor en adenas para cada uno de los editores

Haciendo la consulta equivalente sobre la BD con el nuevo esquema conceptual:

```
SELECT editor, idrev FROM reveditor GROUP BY editor
```

El resultado de la consulta se muestra en la Tabla 4.3.

Editorial	**Total**
Brill Academic Publisher	6
Brill Academic Publishers	49
Institute of Electrical and Electronic Engineers	2
Institute of Electrical and Electronics Engineers	59
Institute of Electrical & Electronics Engineers	1
Organisation for Economic Cooperation and Development	6
Organisation for Economic Cooperation & Development	54
Sage Publications	18
SAGE Publications	52
Sage Publications, Inc.	1
Sage Publications (London)	5
Sage Publications, Ltd.	1
(2340 registros)	

Tabla 4.3: Conteo del editores agrupado por nombre en la base de datos con el esquema conceptual normalizado propuesto.

Una vez normalizado el esquema de la base de datos, es posible hacer una estandarización de los valores de los datos almacenados. Mediante la estandarización se pretende eliminar duplicados no exactos.

4.2. SEC

Para la estandarización de valores se implementó el Sistema de Estandarización de Cadenas (SEC), el cual permite estandarizar un conjunto de cadenas de caracteres basándose en algoritmos de alineamiento mediante el uso de diccionarios de sinónimos creados en XML. Ver Figura 4.3

Figura 4.3: Logotipo del Sistema de Estandarización de Cadenas (SEC)

La estandarización de valores no puede ser un proceso automatizado completamente, ya que debe ser supervisado por un experto para no eliminar información. En el caso de la base de datos bibliográfica de BiDi, los valores que pueden ser estandarizados son: `título` y `editor`.

Para llevar a cabo la estandarización de datos en la BD, se requiere de una estructura externa que funcione como un diccionario de sinónimos, con la finalidad de mantener todos los posibles valores que puede tomar un dato antes de ser estandarizado.

Las estructuras que representan a los diccionarios de sinónimos deben ser externas a la base de datos para que no alteren el esquema de la misma agregando más tablas para almacenar datos propios del diccionario y con la finalidad de que los diccionarios creados puedan ser utilizados en diferentes aplicaciones.

Las ventajas de manejar estructuras tipo diccionario a través del sistema y que éstos sean documentos XML se listan a continuación:

- Al tener un diccionario de sinónimos no se requiere procesar nuevamente los datos para hacer modificaciones a la base de datos (como insertar o eliminar datos), *con la seguridad de que las agrupaciones hechas para el diccionario se realizaron bajo la supervisión de un especialista*.
- Es posible mantener la estructura de un diccionario de sinónimos bajo la estructura de un documento XML, almacenando todos los valores posibles que puede tomar el nombre de un dato.
- Al guardar los diccionarios de sinónimos como documentos XML es posible procesarlos posteriormente utilizando una gran variedad de herramientas

que ya existen para ello.

4.2.1. Requerimientos del sistema

El Sistema de Estandarización de Cadenas debe cumplir los siguientes requerimientos:

- El sistema debe permitir la estandarización de cadenas de caracteres provenientes de fuentes como archivos de datos y datos almacenados en una base de datos relacional.
- El sistema debe proporcionar la estandarización por medio de diferentes algoritmos de alineamiento de cadenas, mencionados en el Capítulo 2.
- El sistema debe permitir que el usuario indique qué campos desea estandarizar y los parámetros que utilizará independientemente de que los datos provengan de un archivo de texto o de una BD.
- El sistema debe proporcionar una interfaz gráfica de usuario para la manipulación de los diccionarios de manera intuituva.
- El sistema debe permitirle al usuario probar la estandarización de cadenas con diferentes parámetros de los algoritmos para obtener distintos resultados y darle al usuario la posibilidad de escoger el que más le convenga.

4.2.2. Diseño

El desarrollo del sistema se divide en 4 módulos principales:

1. Definición del esquema XML para los diccionarios de sinónimos, así como de la estructura de datos para representarlos, la cual debe ser utilizada por el Sistema de Estandarización de Cadenas.
2. Implementación de los algoritmos de alineamiento y establecer métricas de comparación de distancia y de similitud.
3. Desarrollo de la Interfaz de Usuario (GUI) para la manipulación de los diccionarios creados, con el propósito de que su creación sea un proceso supervisado.

4. Conexión a la base de datos relacional y almacenamiento de los diccionarios de sinónimos en una base de datos nativa XML para hacer las consultas a la base de datos en Postgres, consultando previamente la base de datos nativa XML.

Cada uno de éstos módulos se describen en las siguientes secciones:

Diccionario de Sinónimos

Los diccionarios de sinónimos creados mediante el Sistema de Estandarización de Cadenas deben ser documentos XML, por lo que su definición requiere de dos partes:

- Definición del esquema XML

- Diseño de la estructura para su representación en el sistema

Definición del esquema XML

Como se mencionó en el Capítulo 3, un esquema XML define qué elementos puede contener un documento XML, cómo están organizados y qué atributos y de qué tipo pueden tener sus elementos.

Para la representación de un diccionario de sinónimos se identifican 2 elementos principales: las palabras y los sinónimos de las mismas. Las palabras deben tener un valor que será la *llave* de los elementos contenidos en el diccionario y cada uno de los sinónimos tiene un valor.

Al ordenar jarárquicamente los elementos que pertenecen a un diccionario de sinónimos, se obtiene el diagrama de la Figura 4.4.

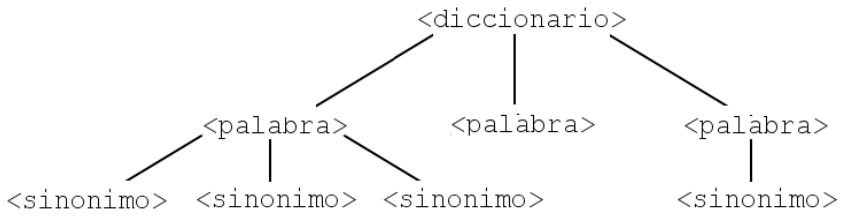


Figura 4.4: Esquema del anidamiento de elementos para la construcción de un diccionario de sinónimos.

Basándose en el diagrama de la Figura4.4 se puede definir el siguiente esquema XML para la representación de diccionarios de sinónimos:

```
<xsd:schema
       xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 <xsd:element name="diccionario"
                          type="tipoDiccionario"/>
    <xsd:complexType name="tipoDiccionario">
       <xsd:element name="palabra"
                        type="tipoPalab"/>
    </xsd:complexType>
    <xsd:complexType name="tipoPalab">
       <xsd:element name="sinonimo"
                        type="tipoSin minOccurs="0"/>
            <xsd:attribute name="valor"
                  type="xsd:string" use="required"/>
    </xsd:complexType>
    <xsd:complexType name="tipoSin">
         <xsd:attribute name="valor"
                type="xsd:string" use="required"/>
    </xsd:complexType>
</xsd:schema>
```

Con el esquema anterior, un ejemplo de diccionario de sinónimos que cumple con el esquema es el siguiente:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
  <diccionario>
    <palabra valor="triunfo">
      <sinonimo valor="victoria"/>
      <sinonimo valor="conquista"/>
    </palabra>
    <palabra valor="amanecer"/>
    <palabra valor="amplificar">
      <sinonimo valor="ampliar"/>
    </palabra>
  </diccionario>
```

Diseño de la estructura

El diseño de la estructura de datos estuvo basado en la estructura de árbol n-ario mostrada en la Figura4.4. Para implementar esta estructura de datos se implementó la clase `EntradaArbol`, los objetos de esta clase están formados por una `Palabra` y una lista de `Sinónimos`. El diagrama de clases para la implementación de los elementos de un árbol se muestran en la Figura 4.5.

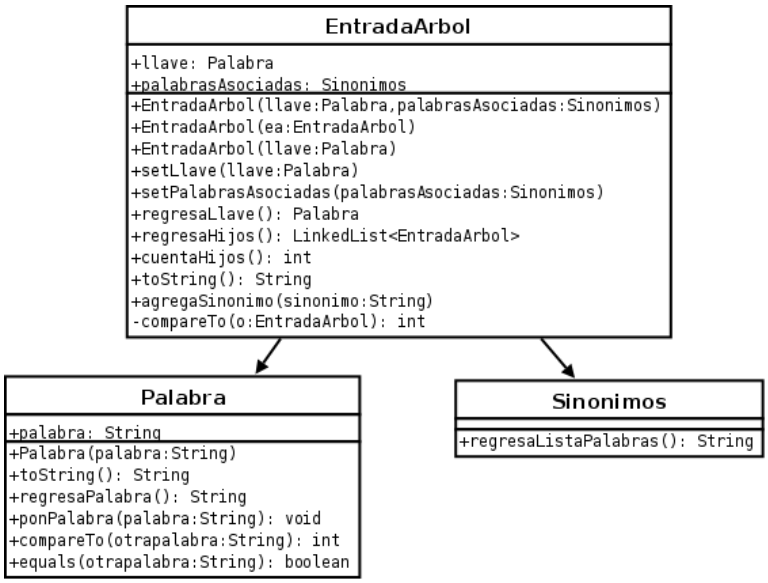


Figura 4.5: Esquema de clases de los objetos que llenarán el diccionario de sinónimos.

Para cumplir el requerimiento de proporcionar al usuario la posibilidad de manipular gráficamente los diccionarios creados a través de SEC, se considera la implementación de un árbol dinámico. En la Figura 4.6 se muestran el diagrama de las clases `ModeloArbol` y `Tesauro`, clases en donde se define cómo se modificará la estructura del árbol al hacer una operación sobre él y las reglas para realizar estas operaciones, respectivamente.

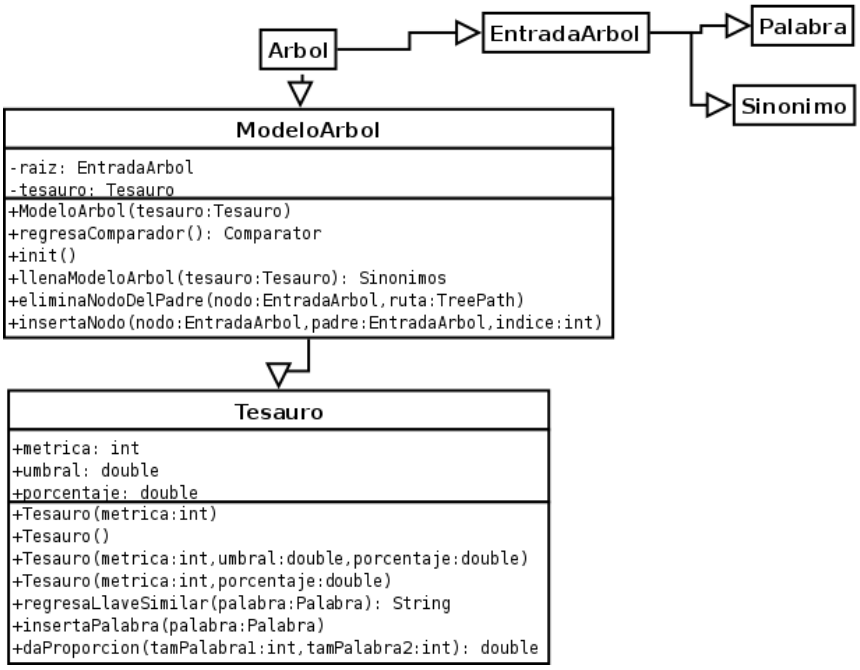


Figura 4.6: Diagrama de clases del Diccionario de Sinónimos

Medidas entre cadenas

Todos los algoritmos de comparación de cadenas de caracteres mencionados en el Capítulo 2 (exceptuando el de la *distancia de edición por bloques*) se implementan en la clase `Distancia`. Esta clase proporciona los métodos de obtención de métricas de distancia o de similitud. Cada objeto de esta clase tiene un atributo llamado `umbral` que es el porcentaje de distancia o de similitud establecido por el usuario para determinar si dos palabras son *semejantes* o no.

Los métodos de la clase `Distancia` se muestran en la Figura 4.7.

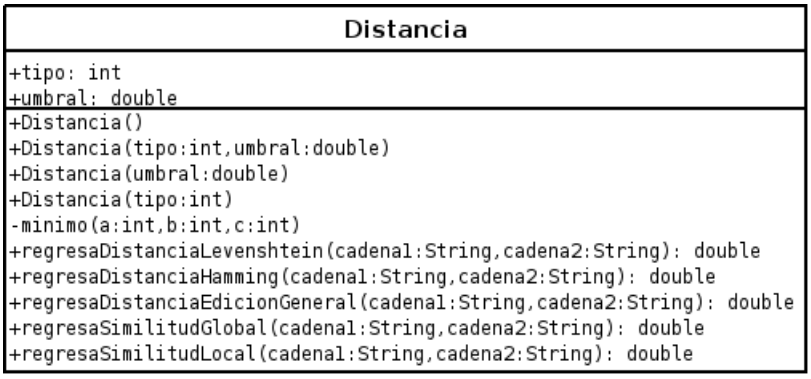


Figura 4.7: Diagrama de la clase `Distancia`

Base de Datos

El sistema debe tener la posibilidad de obtener los datos de una base de datos relacional y de guardar los diccionarios de sinónimos XML en una base de datos nativa XML para hacer más eficientes las consultas sobre grandes colecciones de documentos.

Para realizar la estandarización de los datos de la BD bibliográfica de BiDi, SEC interactúa con 2 tipos de bases de datos:

1. Una base de datos relacional en Postgres, que es en donde se encuentra instalada la base de datos bibliográfica de la Biblioteca Digital

2. Una base de datos nativa XML eXist para almacenar los diccionarios creados a través de SEC.

SEC permite hacer operaciones sobre eXist o sobre una base de datos relacional a través de su interfaz. Para establecer la conexión con la base de datos relacional, los parámetros como el nombre de usuario, el password y el driver se definen en un archivo de propiedades. Las operaciones que se pueden realizar a través de SEC son:

- Obtener los datos de una base de datos relacional estableciendo una conexión con parámetros definidos en un archivo de propiedades, solicitándole al usuario datos como el nombre de la BD, la tabla y la columna a estandarizar.

- Crear diccionarios XML y almacenarlos en un archivo o en eXist.

- Hacer consultas sobre eXist indicando el nombre de la colección y el valor de un dato a buscar en el diccionario.

- Crear consultas SQL de actualización de la base de datos relacional para estandarizar valores utilizando un diccionario XML almacenado en eXist. Para esto, SEC hace dos conexiones secuenciales, primero se conecta a e-Xist para obtener los valores que puede tomar un dato y después se conecta a Postgres para hacer las actualizaciones.

Interfaz de Usuario

Además de ser una herramienta para la implementación de otras aplicaciones de software que proporciona métodos para la comparación de cadenas de caracteres y la creación de documentos XML, SEC cuenta con una interfaz gráfica para la creación y manipulación de diccionarios de sinónimos.

Para el diseño de la interfaz gráfica se consideraron los siguientes puntos:

- Debe permitir al usuario establecer los parámetros necesarios para obtener datos de una base de datos relacional. Estos parámetros son: nombre de la base de datos, nombre de la tabla y nombre de la columna de donde se obtendrá el conjunto de datos.

- Debe permitir al usuario elegir el algoritmo de alineamiento que será utilizado para la creación de los diccionarios de sinónimos.

- Debe permitir al usuario crear diferentes diccionarios de sinónimos modificando los parámetros para comparar todos los diccionarios creados de modo que el usuario pueda elegir el que más se adapte a sus necesidades.

- Debe permitir hacer consultas con la base de datos relacional y con la base de datos eXist para obtener información.

4.3. Descripción del Sistema

La implementación del sistema se encuentra desarrollada completamente en Java, incluyendo la Interfaz Gráfica de Usuario. Las actividades que puede realizar el usuario se encuentran bien definidas:

1. Crear un diccionario de sinónimos a partir de un archivo o de una base de datos relacional indicando las columnas a estandarizar.

2. Importar un diccionario de sinónimos, el cual es un documento XML que cumple con el esquema propuesto para crear un objeto de la clase Diccionario que puede ser modificado nuevamente por el usuario.

3. Consultar un diccionario de sinónimos haciendo consultas a una base de datos en eXist a través de XQuery, para esto es necesario que el documento creado se haya ingresado previamente a la base de datos nativa XML.

4. Hacer consultas en la base de datos en Postgres. Esta opción se implementó para dar la oportunidad al usuario de consultar los campos estandarizados, de manera que si busca un término X ó todos los sinónimos de X, siempre regresen los mismos resultados.

   El método que realiza esta acción devuelve un objeto de la clase `ResultSet` porque no es posible mostrar el resultado de esta acción a través de la interfaz gráfica.

El esquema general de casos de uso se muestra en la Figura 4.8

4.3.1. Instalación

Para la instalación del sistema se implementaron dos instaladores: uno para Linux mediante el uso de IzPack[5] y un instalador para Windows utilizando NSIS[6]

[5] `http://izpack.org` [41]
[6] `http://nsis.sourceforge.net` [42]

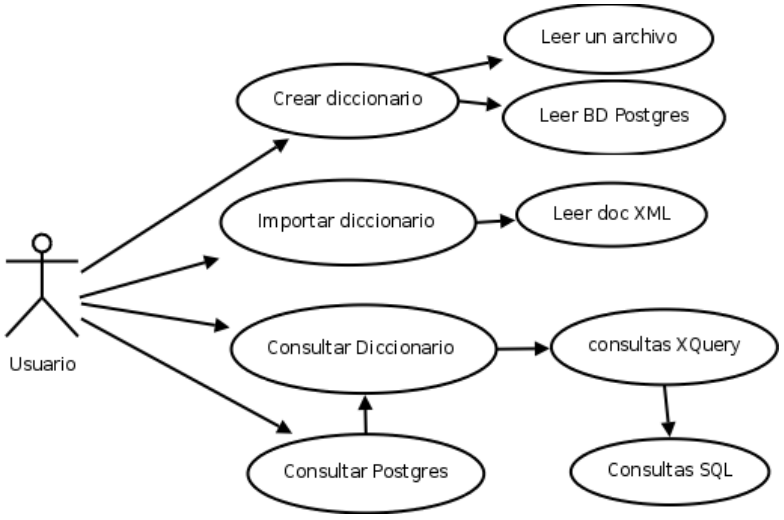


Figura 4.8: Diagrama de casos de uso del sistema

Izpack

A través de Izpack es posible crear instaladores en archivos `jar` para cualquier plataforma ya que está implementado en Java. IzPack es un proyecto SourceForge y está distribuido bajo la licencia Apache Software License 2.0 [41].

NSIS

A través de NSIS es posible crear instaladores para Windows, que son capaces de instalar, desinstalar y configurar los parámetros del sistema. NSIS se basa en archivos de comandos y es compatible con Windows 95, Windows 98, Windows ME, Windows NT, Windows 2000, Windows XP, Windows Server 2003 y Windows Vista [42].

4.3.2. Inicio

Cuando se ejecuta el sistema, aparece en pantalla una ventana de inicio en donde se muestran las características generales del software desarrollado como nombre y versión. Ver Figura 4.9

Figura 4.9: Imagen de bienvenida al sistema

La interfaz gráfica del sistema cuenta con un espacio de trabajo y una barra de menú. En la barra de menú se despliegan los menús y cada una de las operaciones que es posible realizar a través de SEC, éstas se describirán en las siguientes secciones. En el espacio de trabajo se mostrarán todas las ventanas de configuración para crear diccionarios de sinónimos así como las ventanas en donde se muestran los resultados.

4.3.3. Crear un diccionario de sinónimos

SEC permite crear diccionarios de sinónimos utilizando algoritmos de alineamiento de cadenas de caracteres. El criterio de agrupación se puede dividir entre algoritmos de distancia o de similitud. A través de SEC es posible elegir el algoritmo que será utilizado para crear el diccionario de sinónimos.

A través del sistema es posible crear un diccionario de sinónimos con datos provenientes de una base de datos relacional o de un archivo de texto. Para crear un diccionario es necesario seleccionar la opción deseada del menú `Inicio` como se muestra en la Figura 4.10.

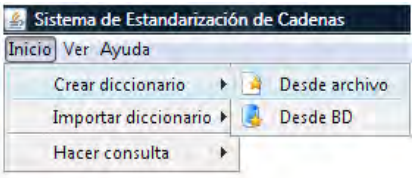


Figura 4.10: Selección de operación: crear diccionario de sinónimos

Una vez seleccionada la fuente de los datos, aparecerá en pantalla una ventana en donde se deben indicar los datos necesarios para la creación de un diccionario de sinónimos. El diccionario creado por el sistema es sólo una base, mediante la cual el usuario puede modificar la estructura para que el proceso de creación del diccionario sea un proceso supervisado y las estructura satisfaga las necesidades del usuario, evitando insertar errores en la base.

Entrada de datos desde archivo

Para crear un diccionario de sinónimos a partir de un archivo es necesario especificar algunos parámetros como la ubicación del archivo de datos, el algoritmo a utilizar y el peso de las operaciones del algoritmo seleccionado. En la Figura 4.11 se muestra la ventana en la que se asignan los parámetros para crear un diccionario de sinónimos a partir de los datos de un archivo utilizando algoritmos de distancia.

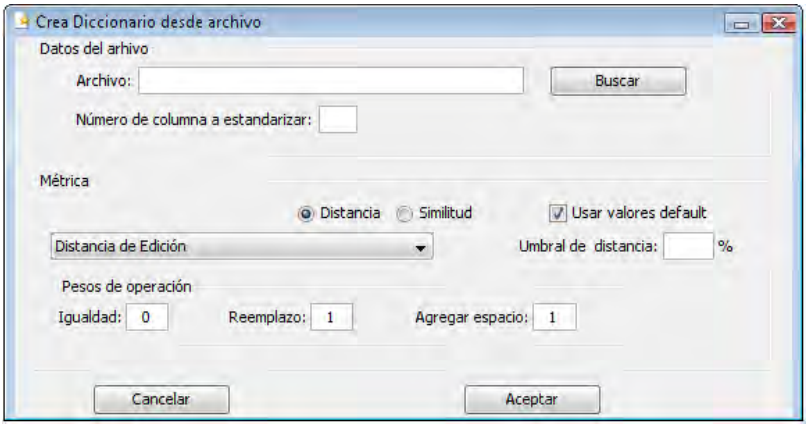


Figura 4.11: Ventana para asignar parámetros para la creación de un diccionario de sinónimos con los datos de un archivo

Cuando los datos provienen de un archivo de texto es necesario que éste tenga un formato especial, es decir que se encuentre en columnas en donde una columna contiene datos a estandarizar. Los archivos de este tipo pueden ser documentos en hojas de Cálculo.

Entrada de datos desde BD relacional

Para obtener los datos de una base de datos es necesario dar los parámetros como el nombre de la base, de la tabla y de la columna a estandarizar. Es importante señalar que no deben existir valores nulos porque no será posible obtener una métrica con cualquier algoritmo seleccionado.

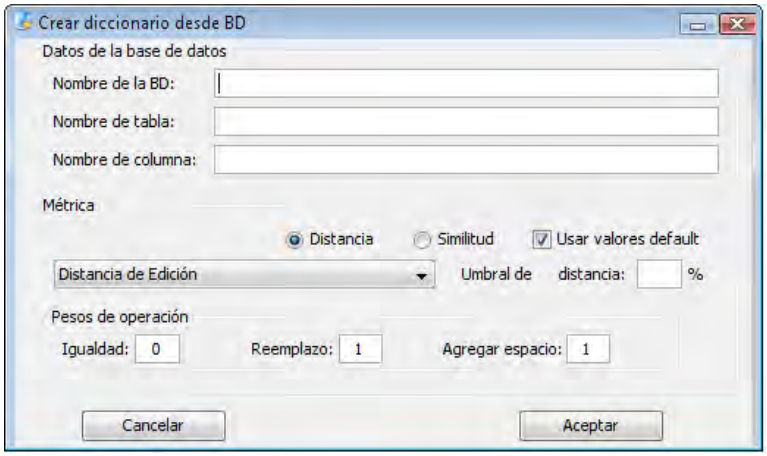


Figura 4.12: Ventana para asignar parámetros para la creación de un diccionario de sinónimos con los datos de una BD

Asignación de parámetros

Para crear un diccionario de sinónimos a través de SEC es necesario indicar la fuente de los datos, ya sea un archivo de texto o provengan de una base de datos relacional. También es necesario definir el algoritmo de alineamiento de cadenas que será usado, así como los parámetros de los mismos. Como se mencionó en el Capítulo 2, la asignación de parámetros afecta el resultado obtenido y éstos deben ser elegidos de acuerdo a las cadenas que van a ser comparadas.

El Sistema de Estandarización de Cadenas permite que el usuario modifique los parámetros de creación de diccionarios para que pueda crear diferentes diccionarios y seleccione aquel que satisfaga sus necesidades. Una de las ventajas del sistema es que todos los diccionarios de sinónimos pueden estar siempre en

pantalla, de manera que puedan ser comparados por el usuario.

Para definir los parámetros de los algoritmos de lineamiento de cadenas, se deben indicar los pesos de las operaciones de edición simple que se van a utilizar. En la Figura 4.12 se muestra la ventana en la que se asignan los parámetros para crear un diccionario de sinónimos a partir de los datos de una base de datos utilizando algoritmos de distancia.

4.3.4. Diccionario de sinónimos

Una vez que se han introducido los parámetros necesarios para la creación de un diccionario, se muestra en pantalla una ventana con el árbol creado. Este árbol representa el diccionario creado por el sistema con los parámetros indicados por el usuario, además es una estructura dinámica que permite que el usuario arrastre cada uno de los nodos hoja a una nueva posición.

Un ejemplo del árbol creado se muestra en la Figura 4.13, en donde se ejemplifica la creación de un diccionario de sinónimos con los valores de las editoriales de las revistas electrónicas de BiDi.

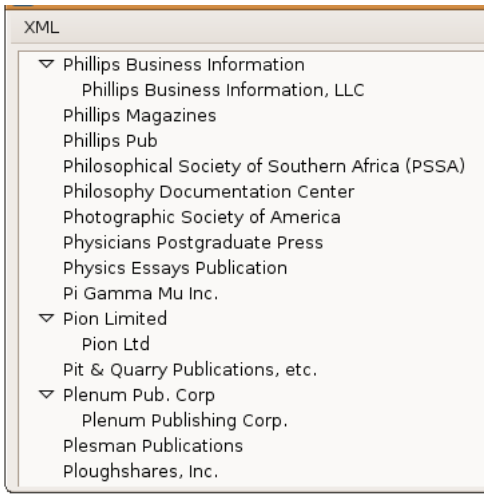


Figura 4.13: Diccionario creado con los datos de las editoriales de las revistas electrónicas de BiDi.

Exportar a documento XML

Una vez que se ha creado un diccionario de sinónimos es posible exportarlo a un documento XML . La ventaja de esta opción radica en que los documentos XML creados podrán ser utilizados en otras aplicaciones.

Para exportar un diccionario de sinónimos se debe especificar el nombre del documento XML que será creado. Un fragmento de un diccionario de sinónimos convertido a documento XML es el siguiente:

```
<diccionario>
 <palabra valor="Phillips Business Information">
   <sinonimo
        valor="Phillips Business Information, LLC"/>
 </palabra>
 <palabra valor="Phillips Magazines"/>

...

 <palabra valor="Pion Limited">
   <sinonimo valor="Pion Ltd"/>
 </palabra>
 <palabra valor="Plenum Pub. Corp">
   <sinonimo valor="Plenum Publishing Corp."/>
 </palabra>
...

</diccionario>
```

Exportar a eXist

A través del Sistema de Estandarización de Cadenas es posible enviar directamente un diccionario de sinónimos a una base de datos nativa XML, como un documento XML. La base de datos utilizada es eXist. Debido a que eXist almacena los datos XML en colecciones se le debe especificar al sistema la coleccion de datos y la base de datos en donde se almacenará el diccionario.

4.3.5. Búsquedas

SEC permite realizar búsquedas sobre los diccionarios de sinónimos almacenados en la base de datos nativa XML eXist. Las búsquedas se pueden realizar sobre el diccionario de sinónimos o sobre una base de datos relacional indicando previamente los parámeros que se requieren para cada caso.

Sobre diccionario

Las búsquedas sobre los diccionarios son búsquedas sobre documentos o colecciones de documentos XML almacenados en eXist. Las búsquedas que se pueden hacer sobre los diccionarios de sinónimos almacenados en eXist pueden ser de tres tipos:

- Consulta los sinónimos de una cadena específica.
- Consulta las palabras que tengan un determinado número de sinónimos asosciados en el diccionario.
- Consulta las palabras y sus sinónimos que contengan una cadena de caracteres en particular.

Para hacer una consulta sobre un diccionario de sinónimos es necesario indicar el nombre de la base de datos, si es que existe, y el nombre de una colección de documentos XML. También se debe especificar el término de búsqueda.

El tipo de búsqueda se debe seleccionar de entre las opciones posibles. En la Figura 4.14 se muestra el resultado de una consulta por sinónimo, el término de búsqueda se busca tanto en los elementos `palabra` como en los subelementos `sinonimo`. En este tipo de búsqueda si no hay una palabra o sinónimo que coincida con el término de búsqueda no se regresará nada.

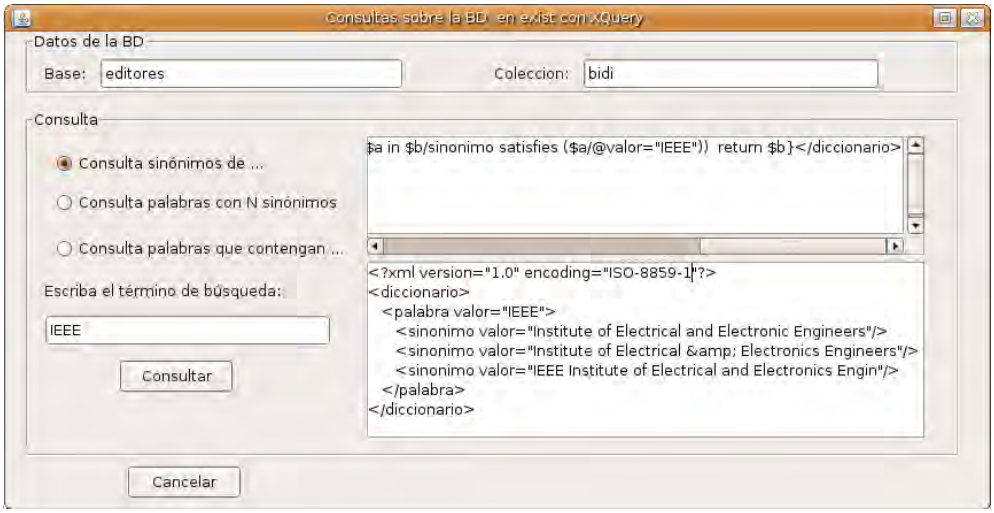


Figura 4.14: Ventana para hacer consultas sobre un diccionario de sinónimos en eXist. Resultados de buscar los sinónimos de IEEE

En la Figura 4.14 se muestra el resultado de hacer la consulta de los sinónimos de `IEEE` de acuqerdo a como se encuentran almacenados en la base de datos. La consulta XQuery es:

```
<diccionario>
  {for $b in doc(URI)//palabra where
             ($b/@valor="IEEE") or
             (some $a in $b/sinonimo
              satisfies($a/@valor="IEEE"))
        return $b}
</diccionario>
```

Y los resultados son:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<diccionario>
 <palabra valor="IEEE">
  <sinonimo
valor="Institute of Electrical and Electronic Engineers"/>
  <sinonimo
valor="Institute of Electrical &amp; Electronics Engineers"/>
```

```
  <sinonimo
valor="IEEE Institute of Electrical and Electronics Engin"/>
 </palabra>
</diccionario>
```

A través de SEC también es posible buscar palabras con determinado número de sinónimos. En este caso se debe elegir la opción correspondiente. En la Figura4.15 se muestra una consulta de este tipo a través del sistema.

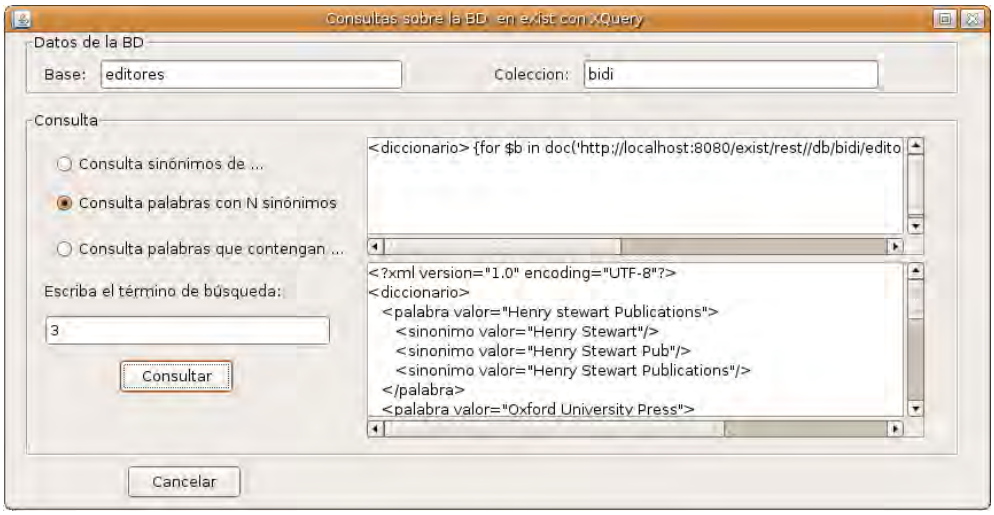


Figura 4.15: Ventana para hacer consultas sobre un diccionario de sinónimos en eXist. Resultados de buscar las palabras que tienen un número determinado de sinónimos, en este caso 3.

En el ejemplo de mostrado en la Figura4.15, la consulta XQuery realizada a la base de datos eXist es:

```
<diccionario>
  {for $b in doc(URI)//palabra
   let $c := $b/sinonimo
   where count($c)=3
   return $b}
</diccionario>
```

Algunos de los resultados encontrados en el diccionario de nombres de editorial en la base de datos en eXist es:

```
<?xml version="1.0" encoding="UTF-8"?>
<diccionario>
 <palabra valor="Henry stewart Publications">
    <sinonimo valor="Henry Stewart"/>
    <sinonimo valor="Henry Stewart Pub"/>
    <sinonimo valor="Henry Stewart Publications"/>
 </palabra>
...
 <palabra valor="World Scientific">
    <sinonimo
      valor="World Scientific Publishing Co. Pte. Ltd."/>
    <sinonimo valor="World Scientific Publishing Co"/>
    <sinonimo valor="World Scientific Publishing Company"/>
 </palabra>
..
</diccionario>
```

Finalmente, es posible buscar palabras contenidas en el diccionario de sinónimos. En la Figura 4.16 muestra un ejemplo de la ventana en donde es posible realizar este tipo de búsqueda en la base de datos nativa XML.

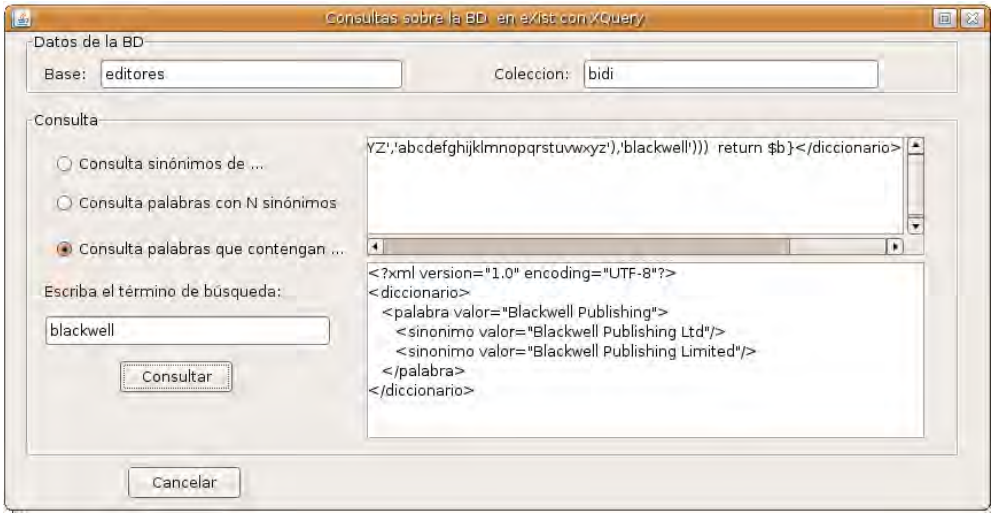


Figura 4.16: Ventana para hacer consultas sobre un diccionario de sinónimos en eXist. Resultados de buscar las palabras que contienen `blackwell`.

En este caso, se debe hacer una conversión de caracteres para ignorar las letras minúsculas, en este caso la consulta XQuery utilizada es:

```
<diccionario>
 {for $b in doc(URI)//palabra
  where contains(translate
       ($b/@valor,'ABCDEFGHIJKLMNOPQRSTUVWXYZ',
       'abcdefghijklmnopqrstuvwxyz'),'blackwell')
    or (some $a in $b/sinonimo satisfies
       (contains(translate
       ($a/@valor,'ABCDEFGHIJKLMNOPQRSTUVWXYZ',
       'abcdefghijklmnopqrstuvwxyz'),'blackwell')))
  return $b}
</diccionario>
```

El resultado a esta consulta es el siguiente documento XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<diccionario>
   <palabra valor="Blackwell Publishing">
      <sinonimo valor="Blackwell Publishing Ltd"/>
      <sinonimo valor="Blackwell Publishing Limited"/>
   </palabra>
</diccionario>
```

Las consultas al diccionario de sinónimos mostradas en esta sección son ejemplos simples de los métodos que SEC proporciona para el desarrollo de nuevas aplicaciones con Java, en donde la estandarización de cadenas y el uso de un diccionario de sinónimos es útil para mantener la calidad de los datos en una base de datos relacional.

Consultas a una base de datos Relacional

Mediante el sistema, también es posible realizar consultas a una base de datos relacional con consultas SQL, consultando de manera previa el diccionario de sinónimos en eXist. Los métodos proporcionados por SEC permiten recuperar un objeto de la clase `ResultSet` después de ejecutar la consulta en la base de datos relacional.

El tipo de consulta que se puede realizar a través del sistema pueden funcionar para una base de datos relacional que no se encuentre estandarizada. En la Figura 4.17 se muestra una consulta de este tipo:

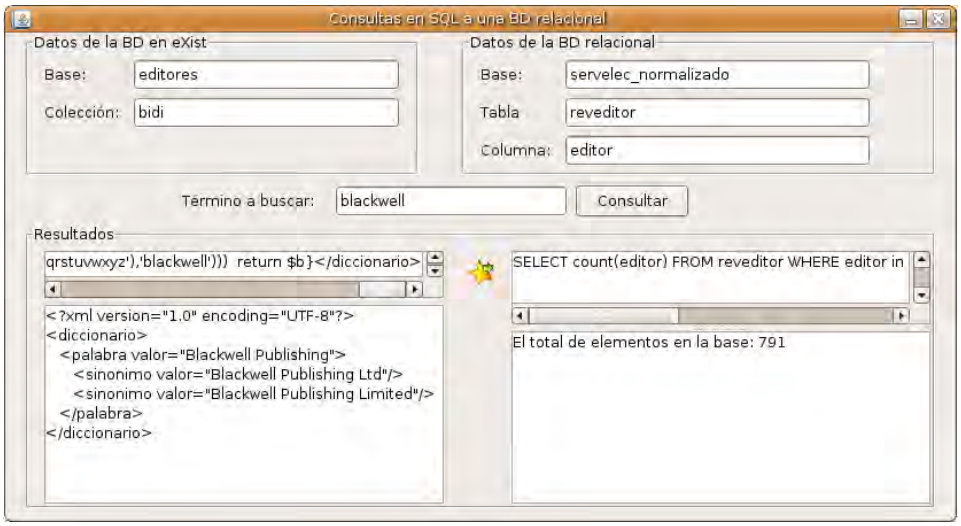


Figura 4.17: Ventana para hacer consultas sobre una base de datos relacional y eXist. Resultados de buscar el término `blackwell`.

Debido a que los métodos de SEC regresan objetos de la clase `ResultSet` para desarrollar otras aplicaciones, en la Interfaz gráfica de SEC sólo se muestra el número de registros que contienen dicho término en la base de datos relacional. La consulta XQuery que se utiliza para en este caso es la siguiente:

```
<diccionario>
 {for $b in doc(URI)//palabra
  where contains(translate
   ($b/@valor,'ABCDEFGHIJKLMNOPQRSTUVWXYZ',
    'abcdefghijklmnopqrstuvwxyz'),'blackwell')
  or (some $a in $b/sinonimo satisfies
     (contains(translate
     ($a/@valor,'ABCDEFGHIJKLMNOPQRSTUVWXYZ',
     'abcdefghijklmnopqrstuvwxyz'),'blackwell')))
 return $b}</diccionario>
```

Posteriormente se obtiene un documento XML con las palabras y sinónimos que contienen el término de búsqueda, el cual en este caso es:

```
<?xml version="1.0" encoding="UTF-8"?>
<diccionario>
   <palabra valor="Blackwell Publishing">
      <sinonimo valor="Blackwell Publishing Ltd"/>
      <sinonimo valor="Blackwell Publishing Limited"/>
   </palabra>
</diccionario>
```

Con los datos recuperados se hace una consuta SQL, por ejemplo:

```
SELECT count(editor)
FROM reveditor
WHERE editor in ('Blackwell Publishing',
'Blackwell Publishing Ltd',
'Blackwell Publishing Limited')
```

Es importante señalar que el Sistema de Estandarización de Cadenas proporciona un conjunto de métodos que permiten implementar aplicaciones para mantener la calidad de los datos en una base de datos relacional. Además, el sistema proporciona métodos para la creación de consultas de actualización en SQL a partir de los diccionarios de sinónimos, permitiendo la actualización de una base de datos relacional, una vez que se han creado los diccionarios de sinónimos.

4.4. Resultados

Para estandarizar los valores de la base de datos para los campos *título* y *editorial*, se crearon dos diccionarios de sinónimos: uno para los títulos y otro para los nombres de las editoriales. Ambos diccionarios se crearon utilizando el algoritmo de similitud global con los parámetros siguientes[7]

$$M = 1, W = -1 \text{ y } w(x_i, y_j) = -1$$

Debido al número de registros en la base de datos, el diccionario de sinónimos de los títulos de las revistas electrónicas se creó en una colección de documentos XML. Las consultas a este diccionario son posibles a través del sistema porque

[7]La notación se describe en el Capítulo 2, sección 2.4.5.

eXist permite buscar elementos sobre colecciones de datos.

Utilizando el algoritmo de similitud global para crear los diccionarios de sinónimos, estableciendo un rango de similitud del 90 %, es posible agrupar correctamente títulos como:

```
Albertson's Inc SWOT Analysis
Albertson's, Inc. SWOT Analysis

Potato Chips Industry Profile Asia Pacific
Potato Chips Industry Profile: Asia-Pacific
```

En general se agrupan correctamente muchos títulos de revistas en donde los signos de puntuación marcan la diferencia entre las cadenas comparadas; sin embargo, muchos títulos no se agrupan correctamente, por ejemplo:

```
Dana Corporation SWOT Analysis
Danaher Corporation SWOT Analysis

Iceland Country Reviews
Ireland Country Reviews

Iceland Economic Competitiveness
Ireland Economic Competitiveness

iTV Gambling Industry Profile: Italy
iTV Gaming Industry Profile: Italy
```

Lo anterior subraya la necesidad de que la creación de diccionarios de sinónimos a través del sistema, utilizando los algoritmos de alineamiento entre cadenas debe ser un proceso supervisado.

En la base de datos existen 25,024 títulos de revistas electrónicas. Como en cualquier conjunto grande de datos, la revisión de los diccionarios de sinónimos creados por el sistema requiere tiempo; sin embargo, es un proceso que sólo se debe realizar una vez. Si los diccionarios de sinónimos están creados, se pueden utilizar para hacer actualizaciones a la base de datos y mantener su calidad de datos.

En la Tabla 4.4 se muestra el número de títulos diferentes de revistas electrónicas, así como el número de nombres de editorial diferentes en la base de datos, antes y después de la estandarización de valores.

Campo	Original	Estandarizada
Título	24,889	22,903
Editorial	2341	2211

Tabla 4.4: Número de títulos y nombres de editoriales que no se repiten en la base de datos antes y después de la estandarización de valores.

Al hacer la siguiente consulta en la base de datos original:

```
SELECT DISTINCT(editor)
FROM revelec
ORDER BY editor;
```

Algunos de los resultados obtenidos serían:

```
Alexander Graham Bell Association for the Deaf

Alliance Communication Group
Alliance Communication Group
Alliance Communications Group

Allured Publishing Corporation

Blackwell Publishing
Blackwell Publishing Limited
Blackwell Publishing Ltd

IEEE
Institute of Electrical and Electronic Engineers
Institute of Electrical and Electronics Engineers
IEEE Institute of Electrical and Electronics Engin
```

Con el nuevo esquema conceptual de la base de datos, la consulta equivalente a la consulta anterior, es:

```
SELECT DISTINCT(editor)
FROM reveditor
ORDER BY editor;
```

El conjunto de resultados mostrados anteriormente se reduce a:

```
Alexander Graham Bell Association for the Deaf
Alliance Communication Group
Allured Publishing Corporation
Blackwell Publishing
IEEE
```

Como se muestra en la Tabla4.4, el número de títulos diferentes en la base de datos se reduce al igual que el número de nombres de editoriales diferentes.

4.5. Evaluación

Mediante la normalización del esquema conceptual de la base de datos, se eliminan campos que no eran utilizados en la base de datos, por ejemplo el campo `borrado` o `nota2`.

Los datos almacenados en los campos `titulo` y `editorial` se estandarizaron mediante el uso de diccionarios de sinónimos creados a través del Sistema de Estandarización de Cadenas. Estos métodos de estandarización de cadenas implementados en SEC no granatizan que los agrupamientos de palabras se realicen correctamente por lo que se deben realizar de manera supervisada.

De un total de 24,889 títulos de revistas diferentes en la base de datos, después de la estandarización se obtuvo un total de 22,903 títulos diferentes, lo que significa que alrededor de un 8 % eran registros claramente duplicados como:

```
Activating the Unemployed: A Comparative
    Appraisal of Work - Oriented Policies -
    International Social Securities Series
Activating the Unemployed: A Comparative
    Appraisal of Work- Oriented Policies -
    International Social Securities Series
```

```
Attention to Politics & Other Facts in Students Grades 9 12
ttention to Politics & Other Facts in Students Grades 9-12

Average Performance Scores for the Arts 1997
Average Performance Scores for the Arts: 1997

Institutional Economics Its Place in Political
   Economy Vol 2
Institutional Economics: Its Place in Political
   Economy Vol. 2
```

Algunos otros ejemplos se listan en el Apéndice B. En la Figura 4.18 se muestra la proporción de revistas con títulos semejantes que fueron estandarizadas a través de SEC.

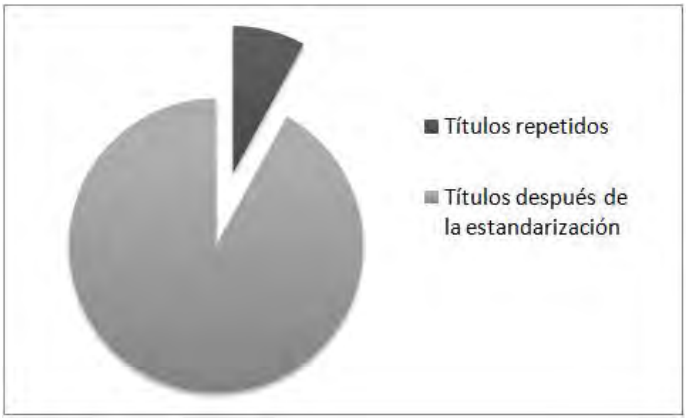


Figura 4.18: Gráfica en donde se muestra la proporción de los títulos de revistas que fueron estandarizados.

En la Figura 4.19 se muestra la proporción de nombres de editoriales semejantes que fueron estandarizados a través del Sistema de Estandarización de Cadenas.

Figura 4.19: Gráfica en donde se muestra la proporción de los nombres de editoriales semejantes que fueron estandarizadas.

En la creación de los diccionarios de sinónimos, los algoritmos de alineamiento dieron mejores resultados con cadenas de longitud larga que al comparar cadenas de longitud corta, ya que en el segundo caso un solo caracter que no tenga una correspondencia puede alterar drásticamente el porcentaje de similitud de las cadenas.

Una de las ventajas es que una vez creados los diccionarios de sinónimos, al ser documentos XML, pueden ser utilizados por diferentes aplicaciones, ya que existe un gran número de herramientas para procesar documentos XML. Además SEC porporciona un conjunto de métodos para mejorar las consultas que se realizan sobre la base de datos. Por ejemplo, si se busca a través del sistema la editorial:

```
Institute for Electronics and Elect Engineerings
```

En la base de datos estandarizada, no habrá registros que cumplan con esta condición porque el valor almacenado para esta editorial es: `IEEE`; sin embargo, mediante el uso de los métodos proporcionados por SEC, es posible consultar previamente el diccionario de sinónimos para obtener el valor que fue utilizado para estandarizar los datos en la base de datos y hacer una consulta a la base de datos relacional con el mismo.

4.6. Resumen

En este capítulo se presenta el Sistema de Estandarización de Cadenas (SEC) desarrollado. SEC proporciona un conjunto de métodos para estandarizar los datos de campos de una BD relacional a través del uso de diccionarios de sinónimos representados con documentos XML.

Para la creación de los diccionarios de sinónimos, SEC proporciona una serie de métodos para hacer agrupamientos de cadenas, todos ellos basados en algoritmos de correspondencia entre cadenas de caracteres. Es importante señalar que la posibiblidad de utilizar los algortimos de alineamiento de cadenas a través del sistema, únicamente simplifica la tarea de la creación de los diccionarios.

En este capítulo se describe la estructura del Sistema de Estandarización de Cadenas y se muestran algunos resultados obtenidos, con los datos de la base de datos bibliográfica de la Biblioteca Digital de la DGB. Se presentan algunos ejemplos de cadenas que no fueron agrupadas correctamente utilizando los algoritmos de correspondencia de cadenas, lo que permite destacar la importancia de que la creación de diccionarios a través del sistema sea un proceso supervisado.

SEC proporciona métodos para interactuar entre una base de datos relacional y una base de datos nativa XML, además de métodos para hacer la estandarización de datos en una base de datos relacional consultando los diccionarios de sinónimos creados.

Una de las ventajas de que los diccionarios de sinónimos sean documentos XML es que pueden ser utilizados en otras aplicaciones ya que existen muchas herramientas para el procesamiento de XML.

El Sistema de Estandarización de Cadenas es una herramienta que proporciona métodos para mantener una base de datos ralacional normalizada y estandarizada. Para una base de datos que ya tiene información como lo es la base de datos bibliográfica de BiDi, la creación de los diccionarios de sinónimos requiere del análisis de un experto para no eliminar información de la base de datos.

Conclusiones

Una vez cumplidos los objetivos planteados al inicio de este trabajo, es posible concluir que SEC es una herramienta de software útil para la estandarización de cadenas de caracteres de grandes colecciones de datos.

Existen muchos algoritmos que permiten el agrupamiento de cadenas y varias aplicaciones como los correctores de ortografía, pero SEC permite crear estructuras como documentos XML los cuales pueden ser procesados por diferentes herramientas y cuentan con las ventajas que tiene cualquier documento XML como el ser autodescriptibles.

SEC utiliza una base de datos nativa XML y permite hacer consultas eficientes utilizando XQuery. Además el Sistema de Estandarización de cadenas es un sistema flexible al proporcionarle al usuario la posibilidad de elegir entre los algorimos de alineamiento más importantes para la creación de los diccionarios de sinónimos.

SEC es una herramienta que puede ser utilizada para el desarrollo de nuevas aplicaciones, de manera que es posible utilizar todos los métodos del sistema para la solución de otros problemas que incluyan la manipulación de cadenas de caracteres. Por lo tanto, se puede decir que SEC es un sistema reusable, flexible y extensible que no se limita a resolver los problemas de estandarización de datos en la base de datos de la Biblioteca Digital de la UNAM.

La interfaz gráfica con la que cuenta SEC permite la fácil manipulación de los diccionarios de sinónimos que se crean o que se importan para ser modificados a través del sistema. Además la interfaz de usuario contempla un espacio de trabajo para que todos los resultados obtenidos puedan ser vistos por el usuario al mismo

tiempo, permitiendo que sea posible utilizar diferentes parámetros para comparar los diccionarios creados.

Al estar implementado completamente en Java, SEC es un sistema multiplataforma y permite especificar los parámetros de conexión a una base de datos relacional mediante el uso de un archivo de propiedades, permitiendo que sea posible que SEC interactúe con diversos manejadores de bases de datos.

SEC cuenta actualmente con manuales de usuario y de instalación en línea en el sitio `http://uxmcc3.iimas.unam.mx/agomez/sec`. Debido a que estos manuales se encuentran en Internet, SEC se convierte en un sistema accesible para diversos usuarios; además es una herramienta útil para la implementación de otros sistemas de software que requieran del uso de los algoritmos de estandarización de cadenas.

SEC actúa como una interfaz entre una base de datos nativa XML y una base de datos relacional, de manera que sea posible hacer consultas sobre la base de datos relacional consultando previamente el diccionario en XML. Este tipo de interacción permite que se puedan desarrollar sistemas de búsqueda sobre datos estandarizados pero contemplando todos los valores posibles que pueda tener una cadena (previamente establecido en el diccionario).

El uso de SEC no requiere de conocimiento previo del usuario ya que permite ajustar los pesos que se les da a cada una de las operaciones de los algortimos de alineamiento, además de manipular de manera gráfica los diccionarios creados para buscar una mejor solución. Algunas características de SEC son:

- Es software libre.
- Conjunta diversos algoritmos de aineamiento para la creación de diccionarios de sinónimos basándose en el uso de algoritmos de similitud y de distancia.
- Proporciona una herramienta a los desarrolladores para utilizar los algoritmos implementados para el desarrollo de nuevas aplicaciones.
- SEC cuenta con una interfaz gráfica que asemeja un escritorio de trabajo, en

donde el usuario tiene de manera simultánea las herramientas y los resultados de la creación de diccionarios de sinónimos con diferentes parámetros.

- Mediante el archivo de propiedades, es posible cambiar los datos para que SEC se conecte a una base de datos relacional y establecer qué sistema manejador de bases de datos será utilizado.
- SEC es una herramienta multiplataforma, lo que significa que corre en los sistemas operativos Windows y Linux y cuenta con una amplia documentación tanto sobre el funcionamiento como la instalación.

Glosario

A

ADN.ADN. Abreviatura de ácido desoxirribonucleico. Molécula constituída por dos cadenas complementarias de nucleótidos que forman una doble hélice. El ADN contiene y transmite la información genética de la mayor parte de los organismos excepto en algunos tipos de virus.`http://www.inmegen.gob.mx/`

D

Data Warehouse.Un Data Warehouse es una integración de información que proviene de diferentes sistemas de información y se convierte en cimiento para la toma de decisiones y el análisis de la información.

Data mining. La minería de datos (Data mining) es un conjunto de técnicas que permiten explorar grandes bases de datos, con el objetivo de encontrar tendencias o reglas que expliquen el comportamiento de los datos en un determinado contexto. La minería de datos involucra técnicas de diferentes disciplinas como estadística, reconocimiento de patrones, redes neuronales, etc. [14]

DOM. Document Object Model (DOM) es un API que genera un arbol del documento XML sobre el que se puede hacer cualquier tipo de recorrido.

F

Firewall. Conjunto de programas de protección y dispositivos especiales que ponen barreras al acceso exterior a una determinada red privada. Es utilizado para proteger los recursos de una organización de consultas externas no autorizadas [43].

H

HTML. HTML es un formato no propietario basado en SGML desarrollado para la publicación de hipertexto en la *World Web Wide*. Un documento HTML puede ser creado en un editor de texto ya que se trata de texto plano.

I

Indexar. Acción de registrar ordenadamente información para elaborar su índice.

Información. Conjunto de datos que adquiere sentido o valor dentro de un contexto.

ISSN. El ISSN (*Intertional Standard Serial Number*) es el código internacional normalizado que permite la identificación de cualquier publicación en serie (incluyendo la electrónica) independientemente del país de publicación, idioma o alfabeto. `http//www.issn.org`

M

MARC.Un registro MARC es un registro catalográfico legible por máquina (*MAchine-Readable Cataloging*), el cual presenta los datos de una ficha de catálogo de biblioteca.`http://www.loc.gov/marc`

Metadato. Término acuñado por Jack Myers en la década de los 60 para describir conjuntos de datos. Es un dato sobre el dato, ya que proporciona información

mínima necesaria para identificar un recurso. Puede incluir información descriptiva sobre el contexto, calidad y características del dato [33].

S

SAX. Simple API for XML (SAX) es un analizador de XML basado en eventos que realiza un recorrido secuencial sobre el documento XML `http://www.saxproject.org` ***W***

W3C. El Consorcio World Wide Web (W3C) es un consorcio internacional donde las organizaciones miembro, personal a tiempo completo y el público en general, trabajan conjuntamente para desarrollar estándares Web.`http://www.w3c.org`.

Lista de algunas revistas almacenadas en la base de datos que tienen el mismo título pero distinto ISSN.

id_rev	**Título**	**issn_imp**	**issn elec**
12154	Adweek	AAAA-0249	
3744	Adweek	0199-2864	
12155	Adweek Magazines'Technology Marketing	AAAA-0250	
11380	Adweek Magazines'Technology Marketing	1536-2272	
179	Analyst	0003-2654	1364-5528
5438	Analyst	0741-7918	
18964	Appliance Design	AAAA-8233	
11713	Appliance Design	1552-5938	
4911	Archives of Pathology & Laboratory Medicine	0363-0153	
246	Archives of Pathology & Laboratory Medicine	0003-9985	1543-2165
10598	Asia Monitor: China & North East Asia Monitor	1470-5184	
10723	Asia Monitor: China & North East Asia Monitor	1474-5615	
12157	Automotive Design & Production	AAAA-0253	
11392	Automotive Design & Production	1536-8823	
9281	Automotive Industries	1099-4130	
4349	Automotive Industries	0273-656X	
8953	Bank Investment Consultant	1088-730X	
12239	Bank Investment Consultant	AAAA-0353	
9237	Bank Investment Services Report	1097-9905	
8603	Bank Investment Services Report	1076-3082	
9275	Bank Loan Report	1099-3398	
12240	Bank Loan Report	AAAA-0354	
11514	Baseline	1541-3004	
12243	Baseline	AAAA-0357	
5700	BC Business	0849-481X	
5679	BC Business	0829-481X	
12372	Better Nutrition	AAAA-0490	
5076	Better Nutrition	0405-668X	
16889	Biology Bulletin	AAAA-5972	
8162	Biology Bulletin	1062-3590	1608-3059

id_rev	Título	issn_imp	issn elec
12538	Biotech Equipment Update	AAAA-0685	
12245	Biotech Equipment Update	AAAA-0359	
12246	Biotech Financial Reports	AAAA-0360	
12539	Biotech Financial Reports	AAAA-0686	
12373	Black History Bulletin	AAAA-0491	
1694	Black History Bulletin	0028-2529	
11007	Business 2.0	1524-9824	
11428	Business 2.0	1538-1730	
5392	Business Week	0739-8395	
436	Business Week	0007-7135	
12567	CAD CAM Update	AAAA-0716	
12430	CAD CAM Update	AAAA-0556	
10835	Canadian Journal of Microbiology	1480-3275	
467	Canadian Journal of Microbiology	0008-4166	
9860	Career Development International	1362-0436	
12164	Career Development International	AAAA-0260	
500	Case Western Reserve Journal of International Law	0008-7524	
497	Case Western Reserve Journal of International Law	0008-7254	
6123	CFO Alert	0894-4822	
9152	CFO Alert	1094-8961	
5218	Chemical Business	0731-8774	
7252	Chemical Business	0970-3136	
7558	Commercial Property News	1043-1675	
7517	Commercial Property News	1042-1675	
12601	Compensation & Benefits Report	AAAA-0753	
5361	Compensation & Benefits Report	0738-1034	
13209	Compost Science & Utilization	AAAA-1597	
8291	Compost Science & Utilization	1065-657X	
13170	Computer Security Update	AAAA-1393	
12257	Computer Security Update	AAAA-0371	
19071	Construction Bulletin	AAAA-8503	
629	Construction Bulletin	0010-6720	
3645	Construction Digest	0194-2476	
11661	Construction Digest	1550-1396	
7033	Consumer Policy Review	0961-1134	
12167	Consumer Policy Review	AAAA-0264	

id_rev	Título	issn_imp	issn elec
12382	Contract	AAAA-0500	
11193	Contract	1530-6224	
11260	Control Solutions International	1532-1274	
8548	Control Solutions International	1074-2328	
13187	Critical Reviews in Computed Tomography	AAAA-1573	
7472	Critical Reviews in Computed Tomography	1040-8371	
12457	Customer Interface	AAAA-0590	
11233	Customer Interface	1531-4472	
12265	Defined Contribution News	AAAA-0379	
11161	Defined Contribution News	1529-5729	
687	Design Engineering	0011-9342	
4727	Design Engineering	0308-8448	
12384	DISAM Journal of International Security Assistance Management	AAAA-0502	
13172	DISAM Journal of International Security Assistance Management	AAAA-1395	
12387	Early American Life	AAAA-0505	
720	Early American Life	0012-8155	
12145	Electronic Engineering Design	AAAA-0239	
774	Electronic Engineering Design	0013-4902	
9236	Electronic Publishing	1097-9190	
19016	Electronic Publishing	AAAA-8353	
781	Emergency Medicine	0013-6654	
7408	Emergency Medicine	1035-6851	1442-2026
12270	Emerging Markets Economy	AAAA-0384	
12666	Emerging Markets Economy	AAAA-0831	
848	Farmers Weekly	0014-8466	
849	Farmers Weekly	0014-8474	
17575	Fiber Optic Technology	AAAA-6658	
19066	Fiber Optic Technology	AAAA-8452	
8411	Fleet Owner	1070-194X	
5221	Fleet Owner	0731-9622	
10936	Food Engineering	1522-2292	
3622	Food Engineering	0193-323X	
894	Foundation News & Commentary	0015-8976	
8607	Foundation News & Commentary	1076-3961	
12726	GUI Program News	AAAA-0899	
12283	GUI Program News	AAAA-0398	
9252	Healthcare Purchasing News	1098-3716	
4476	Healthcare Purchasing News	0279-4799	
13192	Health Policy and Planning	AAAA-1579	
4113	Health Policy and Planning	0268-1080	1460-2237

id_rev	Título	issn_imp	issn elec
12122	IEE Proceedings A-Science Measurement and Technology	AAAA-0097	
2970	IEE Proceedings A-Science Measurement and Technology	0143-702X	
12123	IEE Proceedings G-Circuits Devices and Systems	AAAA-0099	
6876	IEE Proceedings G-Circuits Devices and Systems	0956-3768	
6682	IEE Proceedings H-Microwaves Antennas and Propagation	0950-107X	
2975	IEE Proceedings H-Microwaves Antennas and Propagation	0143-7097	
12443	Imaging Update	AAAA-0576	
12754	Imaging Update	AAAA-0929	
12764	Industrial Environment	AAAA-0940	
13233	Industrial Environment	AAAA-1622	
12192	Information Management Journal	AAAA-0292	
11346	Information Management Journal	1535-2897	
11314	Intellectual Property & Technology Law Journal	1534-3618	
12194	Intellectual Property & Technology Law Journal	AAAA-0294	
10990	Intelligent Enterprise	1524-3621	
12304	Intelligent Enterprise	AAAA-0420	
10593	International Securities Finance	1470-4005	
12196	International Securities Finance	AAAA-0296	
8462	In Vitro Cellular & Developmental Biology - Animal	1071-2690	
11581	In Vitro Cellular & Developmental Biology - Animal	1543-706X	1071-2690
12197	Irish Journal of Management	AAAA-0297	
11938	Irish Journal of Management	1649-248X	
1197	Journal of Business	0021-9398	
8586	Journal of Business	1075-6124	
1237	Journal of Dairy Science	0022-0302	
11018	Journal of Dairy Science	1525-3198	
11812	Journal of Financial Econometrics	1568-4636	
10834	Journal of Financial Econometrics	1479-8409	1479-8417
5971	Journal of Humanistic Education & Development	0890-0493	
5306	Journal of Humanistic Education & Development	0735-6846	
12199	Journal of Management History	AAAA-0301	
9689	Journal of Management History	1355-252X	
3618	Journal of Mental Health Counseling	0193-1830	
7443	Journal of Mental Health Counseling	1040-2861	
6649	Journal of Public Health	0943-1853	1613-2238
11986	Journal of Public Health	1741-3842	1741-3850

	Título		**Título**
67	Event Dropout Rates for Those in Grades	100	Journal of southeast European and Black
	10-12 Ages, 15-24		Sea studies
68	Event Dropout Rates for Those in Grades	101	Kimberly Clark Corporation SWOT
	10-12, Ages 15-24		Analysis
69	Exchange	102	Kimberly-Clark Corporation SWOT
70	Exchange.		Analysis
71	Expedia Inc SWOT Analysis	103	Korn Ferry International SWOT Analysis
72	Expedia, Inc. SWOT Analysis	104	Korn/Ferry International SWOT Analysis
73	fluxx com AG SWOT Analysis	105	London Bridge Software Holdings PLC
74	fluxx.com AG SWOT Analysis		SWOT Analysis
75	Frito Lay SWOT Analysis	106	London Bridge Software Holdings, PLC
76	Frito-Lay SWOT Analysis		SWOT Analysis
77	FSB : Fortune Small Business	107	Louisiana Pacific Corporation SWOT
78	FSB: Fortune Small Business		Analysis
79	Genencor International Inc SWOT Analysis	108	Louisiana-Pacific Corporation SWOT Analysis
80	Genencor International, Inc. SWOT Analysis	109	Lowe's Companies Inc SWOT Analysis
81	Gold Industry Profile Global	110	Lowe's Companies, Inc. SWOT Analysis
82	Gold Industry Profile: Global	111	Lumenis Ltd SWOT Analysis
83	GrafTech International, D574 Ltd. SWOT	112	Lumenis, Ltd. SWOT Analysis
	Analysis	113	Luminar PLC SWOT Analysis
84	GrafTech International,D574 Ltd. SWOT	114	Luminar, PLC SWOT Analysis
	Analysis	115	Lydall Inc SWOT Analysis
85	Greif Bros Corporation SWOT Analysis	116	Lydall, Inc. SWOT Analysis
86	Greif Bros. Corporation SWOT Analysis	117	Lynx PLC SWOT Analysis
87	Institutional Economics Its Place in Political	118	Lynx, PLC SWOT Analysis
	Economy Vol 2	119	Man Group PLC SWOT Analysis
88	Institutional Economics: Its Place in Political	120	Man Group, PLC SWOT Analysis
	Economy Vol. 2	121	Managed Healthcare Industry Profile Global
89	Journal of Consumer Behaviour	122	Managed Healthcare Industry Profile:
			Global
90	Journal of consumer behaviour.	123	Manor Care Inc SWOT Analysis
91	Journal of Physics A - Mathematical and	123	Manor Care Inc SWOT Analysis
	Theoretical	124	Manor Care, Inc. SWOT Analysis
92	Journal of Physics A: Mathematical and	125	Manpower Inc SWOT Analysis
	Theoretical	126	Manpower, Inc. SWOT Analysis
93	Journal of Professional Counseling Practice	127	Marine Biology Research
	Theory & Research	128	Marine biology research
94	Journal of Professional Counseling: Practice,	129	Marine Industry Profile Global
	Theory & Research	130	Marine Industry Profile: Global
95	Journal of Psychiatric & Mental Health	131	Marine Ports & Services Industry Profile
	Nursing		Global
96	Journal of psychiatric and mental health	132	Marine Ports & Services Industry Profile:
	nursing		Global
97	Journal of Public Affairs	133	Marks & Spencer Group PLC SWOT
98	Journal of public affairs.		Analysis
99	Journal of Southeast European & Black Sea	134	Marks & Spencer Group, PLC SWOT
	Studies		Analysis

	Título		Título
135	Marriott International Inc SWOT Analysis	171	Micron Technology Inc SWOT Analysis
136	Marriott International, Inc. SWOT Analysis	172	Micron Technology, Inc. SWOT Analysis
137	Materials Industry Profile Global	173	Milacron Inc SWOT Analysis
138	Materials Industry Profile: Global	174	Milacron, Inc. SWOT Analysis
139	Matsushita Electric Industrial Co Ltd SWOT Analysis	175	MINEBEA CO Ltd SWOT Analysis
		176	MINEBEA CO., Ltd. SWOT Analysis
140	Matsushita Electric Industrial Co., Ltd. SWOT Analysis	177	Mini Vehicles Industry Profile Japan
		178	Mini Vehicles Industry Profile: Japan
141	Maxillofacial Implants Industry Profile United States	179	Mitsubishi Heavy Industries Ltd SWOT Analysis
142	Maxillofacial Implants Industry Profile: United States	180	Mitsubishi Heavy Industries, Ltd. SWOT Analysis
143	MBIA Inc SWOT Analysis	181	Mitsubishi Tokyo Financial Group Inc SWOT Analysis
144	MBIA, Inc. SWOT Analysis		
145	McCormick & Company Inc SWOT Analysis	182	Mitsubishi Tokyo Financial Group, Inc. SWOT Analysis
146	McCormick & Company, Inc. SWOT Analysis	183	Mitsui & Co Ltd SWOT Analysis
		184	Mitsui & Co., Ltd. SWOT Analysis
147	Medco Health Solutions Inc SWOT Analysis	185	Mitsui Chemicals Inc SWOT Analysis
		186	Mitsui Chemicals, Inc. SWOT Analysis
148	Medco Health Solutions, Inc. SWOT Analysis	187	Mitsui Sumitomo Insurance Co Ltd SWOT Analysis
149	MedComSoft Inc SWOT Analysis	188	Mitsui Sumitomo Insurance Co Ltd. SWOT Analysis
150	MedComSoft, Inc. SWOT Analysis		
151	MedImmune Inc SWOT Analysis	189	MM Group Ltd SWOT Analysis
152	MedImmune, Inc. SWOT Analysis	190	MM Group Ltd. SWOT Analysis
153	Medtronic Inc SWOT Analysis	191	mmO2 PLC SWOT Analysis
154	Medtronic, Inc. SWOT Analysis	192	mmO2, PLC SWOT Analysis
155	MEED Middle East Economic Digest	193	Molex Inc SWOT Analysis
156	MEED: Middle East Economic Digest	194	Molex, Inc. SWOT Analysis
157	Merck & Co Inc SWOT Analysis	195	Molson Inc SWOT Analysis
158	Merck & Co., Inc. SWOT Analysis	196	Molson, Inc. SWOT Analysis
159	Merrill Lynch & Co Inc SWOT Analysis	197	MONY Group Inc SWOT Analysis
160	Merrill Lynch & Co., Inc. SWOT Analysis	198	MONY Group, Inc. SWOT Analysis
161	Metal & Glass Containers Industry Profile Global	199	Morality & Economics - De Moribus Est Disputandum
162	Metal & Glass Containers Industry Profile: Global	200	Morality & Economics De Moribus Est Disputandum
163	MG Rover Group Ltd SWOT Analysis	201	Mylan Laboratories Inc SWOT Analysis
164	MG Rover Group, Ltd. SWOT Analysis	202	Mylan Laboratories, Inc. SWOT Analysis
165	Michael Foods , Inc. SWOT Analysis	203	MyTravel PLC SWOT Analysis
166	Michael Foods Inc SWOT Analysis	204	MyTravel, PLC SWOT Analysis
167	Microgaming Ltd SWOT Analysis	205	Nabors Industries Inc SWOT Analysis
168	Microgaming, Ltd. SWOT Analysis	206	Nabors Industries, Inc. SWOT Analysis
169	Microgen PLC SWOT Analysis	207	National Grid Transco PLC SWOT Analysis
170	Microgen, PLC SWOT Analysis	208	National Grid Transco, PLC SWOT Analysis

	Título		Título
209	National Index of Public Effort to Fund Higher Education 1930 96	243	Nuon N V SWOT Analysis
		244	Nuon N.V. SWOT Analysis
210	National Index of Public Effort to Fund Higher Education, 1930-96	245	Oddbins Ltd SWOT Analysis
		246	Oddbins, Ltd. SWOT Analysis
211	Neuroscience & Biobehavioral Reviews	247	Office Depot Inc SWOT Analysis
212	Neuroscience and Behavioral Physiology	248	Office Depot, Inc. SWOT Analysis
213	Nicor Inc SWOT Analysis	249	Office Electronics Industry Profile Global
214	Nicor, Inc. SWOT Analysis	250	Office Electronics Industry Profile: Global
215	NIKE Inc SWOT Analysis	251	Omnicom Group Inc SWOT Analysis
216	NIKE, Inc. SWOT Analysis	251	Omnicom Group Inc SWOT Analysis
217	Nippon Express Co Ltd SWOT Analysis	252	Omnicom Group, Inc. SWOT Analysis
218	Nippon Express Co., Ltd. SWOT Analysis	253	OXIS International Inc SWOT Analysis
219	Nippon Meat Packers Inc SWOT Analysis	254	OXIS International, Inc. SWOT Analysis
220	Nippon Meat Packers, Inc. SWOT Analysis	255	PACCAR , Inc. SWOT Analysis
221	Nippon Mining Holdings Inc SWOT Analysis	256	PACCAR Inc SWOT Analysis
		257	Pagers Industry Profile United States
222	Nippon Mining Holdings, Inc. SWOT Analysis	258	Pagers Industry Profile: United States
		259	Paint & Body Equipment Aftermarket Industry Profile United States
223	Nippon Sheet Glass Company Limited SWOT Analysis		
		260	Paint & Body Equipment Aftermarket Industry Profile: United States
224	Nippon Sheet Glass Company, Limited		
	SWOT Analysis	261	palmOne Inc SWOT Analysis
225	Nissan Motor Co Ltd SWOT Analysis	262	palmOne, Inc. SWOT Analysis
226	Nissan Motor Co., Ltd. SWOT Analysis	263	Panalpina World Transport (Holding) Ltd. SWOT Analysis
227	NMT Group PLC SWOT Analysis		
228	NMT Group, PLC SWOT Analysis	264	Panalpina World Transport Holding Ltd SWOT Analysis
229	Novellus Systems Inc SWOT Analysis		
230	Novellus Systems, Inc. SWOT Analysis	265	Paper & Forest Products Industry Profile Global
231	Noven Pharmaceuticals Inc SWOT Analysis		
232	Noven Pharmaceuticals, Inc. SWOT Analysis	266	Paper & Forest Products Industry Profile: Global
233	Noveon International Inc SWOT Analysis	267	Pennon Group PLC SWOT Analysis
234	Noveon International, Inc. SWOT Analysis	268	Pennon Group, PLC SWOT Analysis
235	Novo Nordisk A S SWOT Analysis	269	Pensions Industry Profile United Kingdom
236	Novo Nordisk A/S SWOT Analysis	270	Pensions Industry Profile: United Kingdom
237	NTL Inc SWOT Analysis	271	Pep Boys - Manny, Moe & Jack SWOT Analysis
238	NTL, Inc. SWOT Analysis		
239	Nu Skin Enterprises Inc SWOT Analysis	272	Pep Boys Manny Moe & Jack SWOT Analysis
240	Nu Skin Enterprises, Inc. SWOT Analysis	273	Pepsi Bottling Group Inc SWOT Analysis
241	Nuance Communications Inc SWOT	274	Pepsi Bottling Group, Inc. SWOT Analysis
242	Nuance Communications, Inc. SWOT Analysis	275	PepsiCo Inc SWOT Analysis
		276	PepsiCo, Inc. SWOT Analysis

Nombres de editores de revistas repetidos en la base de datos cuyos nombres de editorial no son iguales, pero sí son muy semejantes.

Editor	**Total**		
ACS American Chemical Society	1	British Psychological Society	2
ACS American Chemical Society	1	British Psychological Society	2
Adis International	1	BrunnerRoutledge	1
Adis International Limited	1	Brunner / Routledge	1
Advanstar Communications	2	Business Information	
Advanstar Communications Inc.	1	Business Information Group	
Alliance Communication Group	11	Cambridge University Press	
Alliance Communication Group	1	Cambridge University Press	
Alliance Communications Group	2	Cambridge University Press (CUP)	
American Association for the Advancement of Science	5	Capitol Publications, inc	361
American Association for the Advancement of Science. AAAS	48	Capitol Publications, Inc	405
American Association of Critical Care Nurses	1	Carfax International	1
American Association of Critical-Care Nurses	1	Carfax International Publishers	9
American Dental Hygienists Association	7	Carfax Pub.	2
American Dental Hygienists Associations	2	Carfax Pub. Co.	3
American Library Association	1	Carfax Publishing Company	1
American Library Association / Booklist Publications	1	CSIRO Publishing	7
American Psychiatric Publishing Group	1	CSIRO Publishing	2
American Psychiatric Publishing Group	1	CtC Press	2
American Society of International Law	1	CTC Press	1
American Society of International Law	1	Datamonitor	13
American Society of Mechanical Engineers	1	Datamonitor	3
American Society of Mechanical Engineers (ASME)	1	Editorial Ciencias M edicas	1
Annual Reviews	1	Editorial Ciencias Médicas	2
Annual Reviews Inc.	1	EDP Sciences	1
Baywood Pub. Co.	1	EDP Sciences	3
Baywood Publishing Company	1	Excerpta Medica	1
Baywood Publishing Company, Inc.	1	Excerpta Medica, Inc.	1
Berkeley Electronic Press	1	Guilford Publications	2
Berkeley Electronic Press	11	Guilford Publications Inc	1
BioScientifica Ltd	113	Guilford Publications Inc.	1
BioScientifica Ltd	11	Gulf Publishing	1
Blackwell Publishing	21	Gulf Publishing Company	1
Blackwell Publishing Limited	21	Haworth Press	1
Blackwell Publishing Ltd	171	Haworth Press Inc	1
BMJ Publishing	11	Henry Stewart	1
BMJ Publishing Group	11	Henry Stewart Pub	1
Braybrooke Press	21	Henry stewart Publications	1
Braybrooke Press Ltd.	11	Henry Stewart Publications	1
Brill Academic Publisher	1	Horn Book	1
Brill Academic Publishers	1	Horn Book Inc.	2

Editor	Total		
Human Kinetics Publishers	1	Lippincott Williams & Wilkins	1
Human Kinetics Publishers, Inc.	1	Lippincott Williams & Wilkins (LWW)	1
Idea Group Pub	1	Love Pub. Co.	2
IEEE	1	Love Publishing Co.	1
Electrical and Electronics Engineers	3	Meister Pub. Co.	1
Imprint Academic	1	Meister Pub. Co. (etc.)	3
Imprint Academic	5	M E$Sharpe Inc	1
Indian Institute of Management	1	M.E. Sharpe Inc.	2
Indian Institute of Management (IIMB)	1	Miller Freeman	12
INFORMS	1	Miller Freeman Inc.	2
INFORMS - Institute for Operations Research	1	Miller Freeman, Inc.	2
INFORMS: Institute for Operations Research	1	Miller Freeman Publications	1
Institute of Electrical and Electronic Engineers	1	Mosby-Year Book	8
Institute of Electrical and Electronics Engineers	1	Mosby Year Book Inc	1
Institute of Electrical and Electronics Engineers	1	National Council of Teachers of Mathematics	2
Institute of Electrical & Electronics Engineers	1	National Council of Teachers of Mathematics	2
Intellect Limited	1	Nelson Publishing	2
Intellect Ltd.	1	Nelson Publishing Ltd.	33
International Union of Crystallography	1	New York Botanical Garden Press	1
International Union of Crystallography	1	New York Botanical Garden	3
Internet Scientific Publications	1	Oxford University Press	1
Internet Scientific Publications LLC	1	Oxford University Press	1
IOS Press	2	PBI Media	2
IOS Press	1	PBI Media LLC	63
IPC Science and Technology	1	Penn Well Publishing	1
IPC Science and Technology Press	1	Penn Well Publishing Co.	1
IUPAC	1	Pergamon	1
IUPAC	1	Pergamon Press	1
John Benjamins	3	Phillips Business Information	4
John Benjamins BV	1	Phillips Business Information, LLC	1
John Benjamins Publishing Co.	1	Plenum Pub. Corp	1
Johns Hopkins University	3	Plenum Publishing Corp.	64
Johns Hopkins University Press	1	Polygon Media Ltd	9
John Wiley & Sons	1	Polygon Media Ltd.	8
John Wiley & Sons Inc.	1	Portland Press Limited	1
John Wiley & Sons, Inc	1	Portland Press Ltd	1
John Wiley & Sons, Inc.	1	Primedia Business Magazines & Media Inc.	1
John Wiley & Sons, Inc. / Engineering	2	Primedia Business Magazines & Media, Inc.	1
John Wiley & Sons Ltd	10	Project Innovation	2
John Wiley & Sons, Ltd	2	Project Innovation, Inc.	1
Kappa Delta Pi	1	Quality Pub.	1
Kappa Delta Pi (etc.)	1	Quality Publishing	2
Lawrence Erlbaum Associates	1	Royal Society of Chemistry	1
Lawrence Erlbaum Associates, Inc.	1	Royal Society of Chemistry	1
Lippincott Williams & Wilkins	1	Sage Publications	1

Editor	Total		
SAGE Publications	1	Taylor & Francis	2
Sage Publications, Inc.	1	Taylor & Francis Group	1
Sage Publications, Ltd.	16	Taylor & Francis Health Sciences	1
Schnell Pub. Co.	2	Taylor & Francis Ltd	1
Schnell Publishing Company Inc.	1	Taylor & Francis Ltd.	1
Scholastic	1	Time Inc.	5
Scholastic Inc.	1	Time, Inc.	1
SEPM Society for Sedimentary Geology	3	Trade Press Pub	78
SEPM Society for Sedimentary Geology.	2	Trade Press Pub. Co.	1
S Karger AG	1	University of Illinois Press	2
S Karger AG	4	University of Illinois Press	2
Slack	3	VNU eMedia	1
Slack, Inc.	1	VNU eMedia, Inc.	1
Society for General Microbiology	1	VSP / International Science Publishers	1
Society for General Microbiology	1	VSP International Science Publishers	1
Society for Industrial and Applied Mathematics	1	Walter de Gruyter	8
Society for Industrial & Applied Mathematics	1	Walter de Gruyter & Co	1
Springer Science+Business Media	1	Walter de Gruyter GmbH & Co	2
Springer Science & Business Media B.V.	2	Walter de Gruyter GmbH & Co. KG.	1
Springer Science+Business Media B.V	2	W B Saunders	3
Springer Science+Business Media B.V.	13	W.B. Saunders	2
Springer Verlag	2	Whurr Publishers	1
Springer-Verlag	12	Whurr Publishers Ltd	3
Statistics Canada	1	World Scientific	1
Statistics Canada = Statistique Canada	1	World Scientific Publishing Co	1
Taylor and Francis	6	World Scientific Publishing Company	1

Bibliografía

[1] Saul Martínez Equihua. *Biblioteca Digital. Conceptos, recursos y estándares*. Alfagrama Ediciones 2007. ISBN: 978-987-13-0523-0.

[2] Georgina Araceli Torres Vargas. *Biblioteca Digital*. UNAM 2005. ISBN: 9703224717.

[3] Gary Cleveland. *Bibliotecas Digitales: definiciones, aspectos por considerar y retos*. Biblioteca Universitaria, Julio-Diciembre 2001, Vol. 4. No. 2. UNAM. Traducción: Gonzalo Lara Pacheco.

[4] William Saffady. *Informática Documental para Bibliotecas*. Ediciones Díaz de Santos 1986. ISBN:8486251478. Traducido por Andrés Magaña García.

[5] Roberto Garduño Vera. *Enseñanza virtual sobre la organización de recursos informativos digitales*. UNAM, 2005. ISBN: 9703231500.

[6] Albert Bonillo Martín. *Sistematización del proceso de depuración de los datos en estudios con seguimientos*. Proyecto de Investigación para Doctorado de Psicopatología Infantojuvenil. Departament de Psicobiologia i Metodologia de les Ciences de la Salut. Universitat Autonoma de Barcelona. Bellaterra, Septiembre 2003.

[7] Ángeles, María del Pilar. *Management of Data Quality when Integrating Data with Known Provenance*. Tesis de Doctorado. Doctor of Philosophy. Heriot-Watt University. School of Mathematical and Computing Sciences. Edinburg, UK. 2007

[8] Redman, Thomas. *Data Quality for the Information Age*. Boston, MA. London. Artech House, 1996.

[9] Redman, Thomas. *Data Quality: The Field Guide*. Boston, MA. London. Digital Press, 2000.

[10] Setubal C, Meidanis J, *Introduction to Computational Molecular Biolog*, PWS Publishing, First Edition, 33-103 (1997)

[11] Gavin Powell. *Beginning XML Databases*. Wiley Publishing, Inc. Indianapolis, Indiana, 2007.

[12] Drozdek, Adam. *Estructura de datos y algoritmos en Java*. Thomson Learning. México 2007.

[13] Martín, Gregorio. *Curso de XML. Introducción al lenguaje de la Web*. Prentice Hall. -madrid, España, 2005.

[14] Jiawei Han, Micheline Kamber. *Data Mining. Concepts and Techniques*. Academic Press, 2001.

[15] Rusty Harold, Elliotte. *XML imprescindible*. Anaya Multimedia. España 2005.

[16] Olson, Jack E. *Data Quality. The Accuracy Dimension*. Amsterdam. Morgan Kaufmann Publishers, 2003.

[17] Brundage, Michael. *XQuery: The XML Query Language*. Addison Wesley, 2004.

[18] `http://http://www.w3.org/TR/xquery`

[19] Wolfgang Meier. *eXist: An Open Source Native XML Database*. Darmstadt University of Technology. `http://www.exist-db.org/webdb.pdf`

[20] A.Le Hors, P.Le Hegaret, G. Nicol, J. Robie, M. Champion and S. Byrne. *Document Object Model (DOM) Level 2 Core Specification Version 1.0*. `http://www.w3.org/TR/DOM-Level-2-Core/`

[21] Abraham Silverschatz, Henrry F. Korth. *Fundamentos de Bases de Datos*. Cuarta Edición. Mac Graw Hill 2002.

[22] Cormen T, Leiserson C, Rivest R, Stein C. *Introduction to Algorithms.* McGraw-Hill Science/Engineering/Math, Second Edition, 350-356 (2003)

[23] Navarro Gonzalo, *A guided tour to approximate string matching*, ACM COMPU-TING SURVEYS, Volume 33, Issue 1, 31-88 (2001).

[24] Castillo Hernández, Gabriel. Tesis de maestría: *Algoritmo revisado para la extracción automática de agrupamientos semánticos*. Unidad Académica de los ciclos profesionales y de posgrado del Colegio de Ciencias y Humanidades, UNAM. México 2002

[25] Schmid, Joachim. *The Main Steps to Data Quality*. FUZZY! Informatik AG, Eglosheimer Str. 40, 71636 Ludwigsburg, Germany. Joachim.Schmid@fazi.de. P. Perner (Ed.): ICDM 2004, LNAI 3275, 2004. Springer-Verlag Berlin Heidelberg 2004

[26] Maletic, Jonathan I. & Marcus, Andrian. *Cleansing: Beyond Integrity Analysis*. Division of Computer Science. The Department of Mathematical Sciences. The University of Memphis. Junio 2000.

[27] Graham A., Stephen. *String Searching Algorithms*. World Scientific, 1994. ISBN: 9810237030

[28] Gusfield, Dan. *Algorithms on strings, trees and sequences*. Cambridge University Press, 1997.

[29] Briseño, Ana María. Presentación: *Administración de Proyectos*, TIDAP. Noviembre de 2003.

[30] Sushmita Mitra, Tinku Acharya. *Data Mining*. John Wiley and Sons, 2003.

[31] Burkhard, Stephan. Kärkkäinen. *One-Gapped q-Gram Filters for Levenshtein Distance*. Combinational pattern matching:13th annual symposium, CPM 2002, Fukuoka, Japan. July 3-5 2002: proceedings.

[32] Müller, Jeiko & Freytag Johann-Cristoph. *Problems, Methods, and Challenges in Comprehensive Data Cleansing*. Humboldt-Universität zu Berlin. Berlin Germany.

[33] Senso, José A. *El concepto de metadato. Algo más que descripción de recursos electrónicos*. Ci. Inf., Brasilia, v.32, n. 2, p. 95-106, maio/ago. 2003.

[34] Biblioteca Digital, Dirección General de Bibliotecas, UNAM. `http://bidi.unam.mx`

[35] Extensible Markup Language (XML) http://www.w3.org/XML

[36] Extensible Markup Language (XML) 1.0. W3C Recommendation 10-Feb-98. http://www.w3.org/TR/REC-xml-19980210.pdf

[37] Reino Romero, Alfredo. Introducción a XML en Castellano. 26 Enero 2000. http://www.uclv.edu.cu/nosotros/Documentos/Intro%20al%20XML.

[38] W3C. XML Schema. http://www.w3.org/XML/Schema

[39] W3C. XML Path Language (XPath) Version 1.0. Noviembre 1999. http://www.w3.org/TR/xpath

[40] W3C. XSL Transformation (XSLT) Version 1.0. Noviembre 1999. http://www.w3.org/TR/xslt

[41] IzPack Documentation. http://izpack.org/

[42] Nullsoft Scriptable Install System. http://nsis.sourceforge.ne

[43] Glosario. Coordinación General de Servicios Informáticos. Dirección de Cómputo y Comunicaciones. Instituto Politécnico Nacional (IPN). http://www.dcyc.ipn.mx/dcyc/glosario.aspx

Índice alfabético