UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE CONTADURÍA Y ADMINISTRACIÓN

PROCESAMIENTO DE CORPUS LINGÜÍSTICOS MEDIANTE EL USO DE BASES DE DATOS RELACIONALES

TESIS PROFESIONAL

QUE PARA OBTENER EL TÍTULO DE:

LICENCIADO EN INFORMÁTICA

PRESENTA:

LAURA RENATA CHAVARRÍA CRUZ

ASESOR:
L.I. CARLOS FRANCISCO MÉNDEZ CRUZ

MÉXICO DF 2008

AGRADECIMIENTOS:

A todos los seres que se han cruzado en mí camino y de los cuales he aprendido y he recibido su afecto y hoy ya no están conmigo.

A mi asesor por su compromiso sin importar la hora o el día y por su apoyo en está última etapa universitaria.

Al Programa de Apoyo a Proyectos de Investigación e Innovación Tecnológica (PAPIIT) por el apoyo económico para realizar este proyecto.

A mi familia Fer, Dany, Susy, Sofi, Joana y Karla.

A mis amigos Isaac, Fer, Ceci, Liz, Sandrita, Erika, Martín, Paco, Lina y Noé, por su apoyo incondicional y habilidad para hacer de las peores experiencias, momentos inolvidables.

A mi gran amor, por compartir su vida conmigo.

A todos los que en algún momento, personas cercanas o desconocidas, empujaron la silla de ruedas.

A Dios, por permitirme tener una segunda oportunidad y por poder llenar está hoja.

Índice

Introducción

Este trabajo es parte de un proyecto de investigación iniciado en el Instituto de Ingeniería de la UNAM, en el Grupo de Ingeniería Lingüística. Su finalidad es facilitar el trabajo de investigación a lingüistas y estudiosos de la lengua española, en su labor de conocer diferentes fenómenos lingüísticos, que en el presente dan lugar a la forma de comunicación más común, pero compleja, con la que contamos para compartir un sin número de ideas y sentimientos humanos, que nos permiten interactuar y socializar en una comunidad tan diversa como la que tenemos en nuestro país.

Por otra parte, es un esfuerzo personal que me permitió conocer y practicar los diferentes usos de una carrera tan amplia y útil, llena de cambios e innovaciones constantes como el mismo lenguaje, que en diferentes proporciones, van evolucionando y adaptándose a las necesidades humanas. A través de 4 años de carrera estudié conceptos como lenguaje, palabra, alfabeto, datos, sintaxis, etc. Términos que desde la educación básica empecé a conocer y que en la época en que el teléfono de casa aún era de disco, nunca imagine que en una etapa profesional fueran las misma estructura con la que una computara pudiera entender diferentes líneas de código para desarrollar aplicación que corren cuando el compilador no encuentra “errores de sintaxis” o la ausencia de un “;”.

Esperando que esta investigación pueda proporcionar otros enfoques en el uso de la informática, que limitarse a la creación de sistemas que facilitar los procesos operativos de organizaciones comerciales y comprender la importancia de la misma, el primer capitulo se centra en la investigación documental, inicia con el planteamiento del problema que origina este trabajo. El Marco teórico conceptual es el siguiente tema, en él se estudian términos tanto lingüísticos como técnicos, comienza con las definiciones básicas de lingüística, ingeniería lingüística sus técnicas y recursos.

La siguiente sección de este capitulo introduce a la definición, clasificación y etiquetado de corpus, lo cual nos ayudará a comprender en el siguiente capitulo la metodología de Mark Davis y explicar los objetivos y el trabajo que se realiza en torno al Corpus Histórico del Español de México.

Dentro de los conceptos técnicos se encuentran antecedentes y características de las bases de datos y de SQL, enfocándose principalmente en PostgreSQL, que es el manejador en el que se realizó el caso práctico, revisando sus capacidades de indexación para este proyecto.

La sección 3 del capitulo I establece el objetivo, el cuál está seguido de la hipótesis que intentará responderse al finalizar este esfuerzo.

La sección 5 y 6 hacen referencia a la metodología de desarrollo de software y la justificación de su uso, así como la metodología para el procesamiento y análisis de corpus con las cuales se desarrolló el caso práctico.

El capitulo II se centra en el desarrollo del caso práctico, el cual básicamente contiene la documentación de seguir las etapas del ciclo de vida de desarrollo de software: Análisis, Diseño, Desarrollo y Pruebas.

El capitulo III y en lo personal el más importante ya que es el resultado de todo, contiene las conclusiones, en ellas también se menciona algunos trabajos futuros que de aquí se pueden derivar.

Los últimos capítulos se refieren al Anexo, investigación para seleccionar el ciclo de vida de desarrollo de software, el Glosario y por último la Bibliografía utilizada.

I. Investigación Documental

I.1 Planteamiento del Problema

Existen algunos Corpus en Internet para diversos tipos de estudio. Para investigaciones diacrónicas de la lengua española, en particular, existen el CORDE de la Real Academia Española (http://www.rae.es) y el Corpus del Español de Mark Davies (http://www.corpusdelespanol.org) que permiten obtener en segundos grandes cantidades de datos e información sobre nuestra lengua en diferentes siglos. Sin embargo, no existe un corpus que permita estudiar específicamente el Español de México con las características antes mencionadas.

Actualmente el CHEM (Corpus Histórico del Español de México, http://www.iling.unam.mx/chem/) realiza búsquedas por siglos, mediante una infraestructura limitada que sólo permite realizar búsquedas de palabras, mediante un archivo que agrupa todos los textos correspondientes a un siglo determinado, sin permitir el etiquetado de las palabras buscadas, búsquedas en diferentes o en más de una referencia, así como búsquedas por más de una palabra, por lo cual es conveniente generar la infraestructura que permite realizar las diferentes consultas que permitan un estudio más profundo y completo del corpus.

I.2 Marco Teórico y Conceptual

"Los seres humanos tenemos características importantes que nos diferencian del resto de los seres vivos como la racionalidad y la capacidad de comunicarnos por medio del lenguaje hablado; sin estas capacidades, no habríamos desarrollado nuestro saber ni nuestra cultura." **(Sagan, 1978:107).**

De acuerdo al diccionario de la Real Academia Española se entiende como Lengua el "sistema de comunicación verbal y casi siempre escrito propio de una comunidad humana" **(WB, 01).**

La lengua es el medio natural de comunicación entre personas más eficaz de que disponemos, es difícil imaginar muchas de las actividades humanas que no involucren frases en algún idioma o incluso términos técnicos derivados de diferentes idiomas; la utilizamos de muy diversas maneras, para explicar ideas, conceptos, administrar personal, negociar, convencer, dar a conocer nuestras necesidades, expresar nuestros sentimientos, contar historias, transmitir nuestra cultura o escribir una tesis de titulación. Es para la mayoría de nosotros fundamental en todos los aspectos de nuestras vidas.

De la misma forma, el lenguaje constituye un medio eficaz para registrar, asimilar y representar información, así como parte inseparable en muchos de los aspectos sociales, culturales y políticos de nuestra sociedad, forma parte integrante de nuestra cultura y nos ayuda a tener una identidad propia. Louis Hjellmslev, define lengua como *"una institución de carácter social y nacional considerada en parte como una característica especial de todo un pueblo en un período determinado, y en parte como un conjunto de reglas y estatutos que los individuos están obligados a acatar"* **(Hjellmslev, 1980:64).**

Dada la importancia de la lengua en la sociedad humana es necesario estudiarla. La ciencia que se ocupa de la forma como usamos el lenguaje natural, es decir, entender las frases de otras personas, formular nuestros pensamientos y transmitirlos, se llama ***lingüística*** **(WB, 02).**

Con la finalidad de comprender y estudiar mejor el lenguaje, esté se ha dividido en cinco niveles:

1. Fonética/fonología
2. Morfología
3. Sintaxis
4. Semántica
5. Pragmática

Las diferencias entre los niveles se basan en el enfoque de su análisis.

I.2.1 Fonética/fonología

André Martinet **(Martinet, 1975:32)** nos explica que el término fonética va unido a la aparición de la fonología, antes de que este fenómeno ocurriera "fonética" se empleaba generalmente para designar la ciencia de los sonidos del lenguaje, es decir, estudia la producción y percepción de los sonidos de una lengua. En el siglo XIX los especialistas del

Círculo de Praga[1] (**WB, 03**) se interesaron en los detalles físicos y de está forma aparece la fonología, destinada a estudiar los sonidos de la lengua con métodos lingüísticos, mientras que el término fonética se reserva para el estudio de los sonidos del habla según los métodos de las ciencias naturales.

La fonética es la ciencia que se ocupa de la descripción de los sonidos de las lenguas, se enfoca en los aspectos articulatorios y físicos de estos sonidos, es decir, describe qué órganos orales intervienen en su producción, en qué posición se encuentran y cómo esas posiciones varían los distintos caminos que puede seguir el aire cuando sale por la boca, nariz, o garganta, para que se produzcan sonidos diferentes y también las características de las ondas sonoras como frecuencia o intensidad.

Mientras que la fonética estudia la naturaleza acústica y fisiológica de los sonidos, la fonología describe el modo en que los sonidos funcionan en una lengua particular. “Por ejemplo, los japoneses no pueden distinguir entre [l] y [r]” (**WB, 03**).

La fonología se divide en dos disciplinas:

1) La Fonemática: Analiza los fonemas, su clasificación y combinaciones para formar los significantes de la lengua. Los fonemas son sonidos que permiten distinguir palabras en una lengua, la palabra “casa” por ejemplo, consta de cuatro fonemas (/k/, /a/, /s/, /a/).

2) La Prosodia: Examina la entonación, tonos y acentos.

Con la finalidad de aclarar estos términos, diremos que el fonetista nota todas las diferencias fónicas perceptibles, mientras que el fonólogo intenta descubrir los rasgos fónicos que tienen una función distintiva en la lengua **(Martinet, 1975:71).**

Los problemas en fonética computacional están conectados al desarrollo de sistemas de análisis y síntesis del habla. Aún cuando hay sistemas de reconocimiento de voz (la computadora puede reconocer palabras dichas en el micrófono), el porcentaje de palabras identificadas correctamente es todavía bastante bajo. Entre sistemas de generación de voz hay mucho más progreso: existen algunos que hablan bastante bien, sin "acento de robot", aunque su área de aplicación es bastante restringida.

I.2.2 Morfología

La Morfología estudia los componentes significativos de las palabras; se ocupa de la formación de las palabras a partir de secuencias básicas de fonemas. Se ocupa de la estructura interna de las palabras (sufijos, prefijos, raíces y flexiones) y el sistema de categorías gramaticales de los idiomas (género, número, etcétera). Hay lenguajes que tienen bastantes diferencias con el español. Por ejemplo, en árabe, la raíz contiene tres consonantes. Asimismo, las variantes de una palabra se construyen con la inserción de vocales entre las consonantes (KiTaB - el libro, KaTiB - leyendo) **(WB, 06).**

Los problemas de morfología computacional están relacionados con el desarrollo de los sistemas de análisis y síntesis automático morfológico. Aun el desarrollo de tales módulos es bastante difícil, porque hay que hacer grandes diccionarios de raíces. En general existe la

[1] Escuela fundada en 1930 que focaliza el aspecto sociocomunicativo del lenguaje, “señala que el estudio del lenguaje tiene que ocuparse de los mensajes que se emiten en el código lingüístico”

metodología para tal desarrollo y sistemas funcionando para muchos idiomas. Lo que falta aquí es un estándar. Por eso, frecuentemente, los investigadores crean de nuevo tales módulos, reinventando los ya creados por otros y generando escasos avances.

I.2.3 Sintaxis

La Sintaxis estudia y proporciona las relaciones estructurales y sus reglas entre las palabras y con el fin de regir su combinación y formar oraciones.

Se caracteriza por su toma de partido semántico, expresada en términos de papeles y de partes de la oración que nos permiten darle sentido a nuestras expresiones.

La sintaxis computacional debe tener métodos automáticos para análisis y síntesis, es decir, para construir la estructura de la frase, o generar la frase con base en su estructura. El desarrollo de los generadores es una tarea mucho más fácil, y está más o menos claro qué algoritmos son necesarios en estos sistemas. Al contrario, en el desarrollo de los analizadores sintácticos (se llaman parser), todavía es un problema, especialmente para los idiomas que no tienen un orden de palabras fijo, como en el español (en inglés, el orden de las palabras es fijo. Por eso las teorías basadas en inglés no son fáciles de adoptar al español).

I.2.4 Semántica

La Semántica es la disciplina que se ocupa, del significado de las palabras individuales, por separado o dentro de un contexto, y, de los modos en que tanto las oraciones expresan los significados.

El propósito de la semántica es "entender" la frase, es decir, entender y saber el sentido de todas las palabras y dar las interpretaciones a las relaciones sintácticas.

Definir los sentidos de las palabras en sí, es muy difícil, porque existe polisemia, esto quiere decir que una palabra puede tener más de un significado. Por ejemplo, "gato" es un felino y también un instrumento.

La parte de la semántica que nos ayuda a encontrar todos los sentidos a las palabras se llama lexicografía. Los resultados en este tipo de investigaciones, se encuentran en forma de diccionarios.

Así, los problemas de semántica computacional son muy interesantes, pero todavía no tienen soluciones para la definición de palabras y la construcción de redes semánticas.

I.2.5 Pragmática

La Pragmática es el estudio de cómo el lenguaje es usado para cumplir un objetivo, trata de las relaciones entre la oración y el mundo externo **(Martinet, 1975:29)**. El ejemplo es: si estamos comiendo y yo te pregunto si puedes pasarme la sal, tus contestas que sí y sigues comiendo. Seguramente la respuesta es formalmente correcta, porque de verdad puede pasarme la sal y esa fue la pregunta, pero la intención fue pedir la sal y no preguntar sobre la posibilidad de eso.

I.2.6 Ingeniería Lingüística

En nuestros días tiene lugar un cambio en la utilización de la lengua y que aumentará considerablemente su valor en todos los aspectos de la comunicación. Actualmente las personas en la oficina consultan un diccionario impreso, sin embargo tiene posibilidades de revisar una cantidad impresionante de documentos en cualquier idioma con sólo seleccionar la opción “Traducir página” o “Traducir Artículo”. Dicho cambio es el resultado de los avances en ingeniería lingüística.

La ingeniería lingüística proporciona medios de ampliar y mejorar la utilización de la lengua para hacer de ella una herramienta más eficaz, se basa en su conocimiento y funcionamiento de acuerdo a las investigaciones, las cuales nos ayudan a perfeccionar las técnicas para comprenderla y manipularla. . Utiliza recursos lingüísticos como los diccionarios, bancos de datos y corpus terminológicos elaborados a lo largo del tiempo, que representan la base de conocimiento necesaria para reconocer, validar, comprender y manipular las lenguas utilizando la potencia de las computadoras.

En general trata de mejorar la utilización de los sistemas informáticos, perfeccionando nuestra interacción con ellos asimilando, analizando, seleccionando, utilizando y presentando la información de manera más eficaz y proporcionando medios de generación del lenguaje natural y de traducción.

I.2.6.1 Técnicas y recursos

Con lo anterior podemos decir que la ingeniería lingüística es la aplicación de los conocimientos sobre la lengua al desarrollo de sistemas informáticos que puedan reconocer, comprender, interpretar y generar lenguaje humano en todas sus formas. En la práctica, la ingeniería lingüística consiste en una serie de técnicas y recursos lingüísticos, que se aplican, por medio de programas informáticos y que, en el segundo, constituyen una fuente de conocimientos que se puede acceder por medio de programas informáticos **(WB,02).**

A continuación se describen algunas de las múltiples técnicas que se utilizan en ingeniería lingüística **(WB,02).**

- **Identificación y verificación del locutor**. La voz humana es, como las huellas digitales, única para cada individuo. Por esta razón, se puede identificar a una persona determinada, lo que permite comprobar que la persona está autorizada a acceder a un servicio o recurso.
- **Reconocimiento del habla.** Una computadora recibe el sonido de la voz en forma de ondas analógicas, que se analizan para identificar las unidades de sonido (denominadas fonemas) constitutivas de las palabras. Se utilizan modelos estadísticos de fonemas y palabras para reconocer la introducción de datos orales discretos o continuos (ejemplo: software de dictado).
- **Reconocimiento de caracteres e imágenes.** El reconocimiento de textos escritos o impresos exige obtener una representación simbólica de la lengua a partir de la forma espacial de sus símbolos gráficos. En la mayoría de las lenguas esto significa reconocer y transformar caracteres (ejemplo: Escáner). Hay dos tipos de reconocimiento de caracteres:

    - El reconocimiento de imágenes impresas, denominado reconocimiento de caracteres ópticos (ROC, "Reconocimiento Óptico de Caracteres"), el cual

consiste en el reconocimiento de fuentes de imprenta a partir de una serie de una familia de fuentes.
- El reconocimiento de la escritura, que se denomina por lo general reconocimiento inteligente de caracteres (RIC "Reconocimiento Inteligente de Caracteres"). Consiste en técnicas de reconocimiento de palabras que utilizan modelos lingüísticos, tales como léxicos o información estadística sobre la secuencia de palabras.

- **Comprensión del lenguaje natural.** Permite clasificar los textos mediante un análisis inicial, por ejemplo, se centra en partes "interesantes" de un texto con vistas a un análisis semántico más profundo que determine el contenido del texto dentro de un dominio delimitado. Puede utilizarse también, conjuntamente con los conocimientos estadísticos y lingüísticos, para determinar automáticamente las características lingüísticas de las palabras desconocidas, que pueden luego añadirse a los conocimientos del sistema. La combinación de análisis y generación permite la traducción de textos, aunque aún existen deficiencias en estas aplicaciones.
- **Generación de habla.** El habla se genera a partir de plantillas rellenadas, mediante la reproducción de grabaciones o concatenando unidades de habla (fonemas, palabras). El discurso generado tiene que tener en cuenta aspectos como la intensidad, duración y acento para producir una respuesta continua y natural (ejemplo: traductores que proporcionan la pronunciación de las palabras).
- **Léxicos.** Un léxico es un depósito de palabras y de conocimientos sobre ellas. Entre estos se cuentan informaciones sobre la estructura gramatical de cada palabra (morfología), la estructura fonética (fonología), el significado de la palabra en diferentes contextos textuales, etc.
- **Léxicos especializados**. Existen algunos casos especiales de léxicos que se suelen investigar y producir independientemente de los léxicos de carácter general (Ejemplo: Diccionarios electrónicos de la Real Academia Española) :
  - ***Nombres propios***: Los diccionarios de nombres propios son esenciales para una comprensión eficaz de la lengua, al menos para que estos puedan ser reconocidos dentro de su contexto como lugares, objetos, personas o incluso animales.
  - ***Terminología***: En el complejo entorno tecnológico de nuestros días existe una multitud de términos que es preciso registrar, estructurar y poner a disposición de las aplicaciones lingüísticas.
  - ***Redes de palabras (wordnets)***: Las redes de palabras describen las relaciones existentes entre las palabras, por ejemplo, los sinónimos, antónimos, sustantivos colectivos, etc.
- **Corpus:** Un corpus es una muestra amplia de lengua escrita o hablada, que proporciona las bases para:
  - analizar la lengua y determinar sus características
  - entrenar a las máquinas, por lo general para adaptar su comportamiento a circunstancias específicas
  - verificar empíricamente una teoría lingüística
  - ensayar una técnica o aplicación de ingeniería lingüística a fin de determinar su buen funcionamiento en la práctica.

Ya que el esfuerzo de este trabajo se enfoca en el tratamiento, análisis y estudio de corpus lingüísticos, la siguiente sección revisa esta técnica de la lingüística, la cual se apoya además, en el reconocimiento de caracteres e imágenes, léxicos y comprensión del lenguaje natural.

I.2.7 Corpus Lingüísticos

"Los corpus lingüísticos son una recopilación de textos hablados y escritos con la finalidad de realizar cierto análisis lingüístico, como evolución del lenguaje, uso de las palabras, uso de las reglas sintácticas" **(WB,08).**

La palabra corpus proviene del latín, y quiere decir cuerpo, podemos entender un corpus lingüístico como el conjunto de textos, debidamente ordenados, codificados y organizados y que forma un modelo de la realidad lingüística que se requiere observar.

La importancia de los corpus radica en que nos permiten realizar análisis cuantitativos y cualitativos de los datos reales de textos escritos u orales.

I.2.7.1 Tipos de Corpus

Los corpus según su estudio, obtención y temática se pueden clasificar en 11 tipos **(WB,08)** los cuales se mencionan a continuación:

- Según el origen de los textos: Pueden ser escritos u orales.

- Según la especificad de los textos. Se dividen en:
  a. Específicos o especializados: Aportan datos a un tema.
  b. Genéricos: se forman de un género de textos, por ejemplo: revistas o textos poéticos, etc.
     - Canónicos: Textos de un solo autor.
     - Periódico o cronológico. Textos de una época o años determinados.

- Según el lenguaje:
  a. Monolingüe: utiliza sólo un idioma
  b. Multilingüe hace referencia a más de una lengua

- Según la cantidad de texto:
  a. Textual: recoge íntegramente los documentos que los componen.
  b. Corpus de referencia: recoge fragmentos de documentos.
  c. Léxico: Obtiene fragmentos de texto muy pequeños y de longitud constante, a fin de evaluar el léxico del corpus.

- Según la distribución
  a. Equilibrado: Toma proporciones de documentos parecidas para cada uno de los Tipos de documentos.
  b. Piramidal: Textos distribuidos en diferentes estratos o niveles, de tal forma que en el primer estrato se tienen rica variedad temática, pero con muchos textos por cada variedad y en los otros estratos se tiene mayor variedad con menor cantidad.
  c. Monitor: se actualiza constantemente pero mantiene una cantidad de textos constantes, es decir cuando incorpora unos desecha otros.

- Según es estado de la lengua:
  a. Sincrónico: Documentos de 1975 a la fecha.

b. Diacrónico o Histórico: Textos desde los inicios del idioma español a antes de 1975.

- Según la clasificación y anotaciones:
  a. Simple: Guardado en formato ASCII.
  b. Codificado: Textos a los que se les ha añadido electrónica o manualmente etiquetas para reconocer algunos elementos en los documentos.

- Según su propósito:
  a. De dominio público: Son gratuitos o se cobra una cuota para su comercialización.
  b. Corpus privados o restringidos: Su accesibilidad depende del soporte electrónico para el que fueron diseñados. Web, ftp, videos, CD´s

- Según la documentación del corpus:
  a. Documentados: Se tienen registros de los corpus y además es posible usar dicha documentación, ya sea para hacer una búsqueda específica o para conocer de donde provienen los textos.
  b. No Documentados: No tiene registro documentales de los textos.

- Según la variedad dialectal.
  a. Se diferencian los dialectos y variedades lingüísticas.
- Según el género literario: Pueden ser de textos literarios, técnicos de no Ficción (Periódicos)

I.2.7.2 Etiquetado de Corpus

Un corpus se puede etiquetar con información de diversa índole, para lo que se han propuesto formas de anotación diferentes. *"Uno de los estándares más usados hoy día (Burnard 1991; Hockey 1991; Sperberg-McQueen & Burnard 1992), que supone la adopción de un sistema de codificación (mark-up) de documentos ya existente, el Standard Generalised Markup Language (SGML), puesto que es un sistema bastante sencillo, altamente formalizado y de uso común en la comunidad computacional, por lo que facilita el intercambio de recursos lingüísticos entre investigadores"*. **(WB,09)**

SGML permite la codificación de documentos de una forma altamente estructurada, a través de un conjunto de etiquetas definidas por el usuario, que han de ser llevadas a cabo de una forma muy detallada y consistente, en lo que se denomina DTD (Document Type Definition). Este DTD debe acompañar a todo documento SGML para su correcta interpretación por parte de las distintas aplicaciones que pretendan hacer uso de su contenido.

El tipo de anotación más usual es la que identifica partes de la oración, conocida normalmente como etiquetación morfológica (part of speech tagging) y es fundamental para poder hacer más precisas las búsquedas en el corpus, puesto que nos permiten, por ejemplo, seleccionar los usos nominales o verbales de un lema y es también un requisito básico para otros tipos de codificación más sofisticados, como el análisis sintáctico (parsing), y se usan para producir un análisis que identifica las relaciones sintácticas entre los elementos de una oración.

A pesar de los estándares propuestos a los que hacíamos alusión anteriormente, los corpora existentes difieren bastante en el tipo y cantidad de anotación y codificación que poseen.

Leech (1993) propone siete reglas en la anotación de los corpora, reglas que resumimos a continuación:

1. Debe ser posible eliminar las etiquetas añadidas a un texto anotado y recuperar el texto original sin que éste sufra modificación alguna.
2. Debería ser posible también extraer las anotaciones de los textos y almacenarlas de forma independiente, por ejemplo en una base de datos relacional o en líneas paralelas al texto original.
3. El sistema de anotación usado debe estar basado en unas directrices, documentadas y accesibles al usuario final del corpus, de modo que pueda tener acceso tanto a un listado completo de las etiquetas usadas como a las decisiones tomadas en el proceso de etiquetación.
4. Debe ser posible incluir información sobre la autoría de la codificación del texto, de forma que sea posible saber si se ha realizado manualmente y por quién, o si se ha realizado de forma automática con o sin revisión posterior por un lingüista.
5. Se debe hacer al usuario final consciente de que las anotaciones añadidas al corpus no son infalibles, sino que simplemente constituyen una herramienta de ayuda para el análisis. Cualquier anotación que se añada al corpus será, por definición, un acto de interpretación y de análisis del texto, por lo que es susceptible de incorrecciones e inexactitudes.
6. Los sistemas de anotación han de estar basados en la medida de lo posible en principios teóricamente neutrales y sobre los que exista un acuerdo amplio en el seno de la comunidad científica.
7. Ningún sistema de anotación posee, el derecho de ser considerado estándar. Los estándares, cuando existen, se desarrollan por el consenso de los usuarios.

I.2.8 Corpus del Español de México.

Esta tesis como se menciona en la Introducción, forma parte de un proyecto de Investigación del Instituto de Ingeniería de la UNAM, cuyo objetivo es la institucionalización de un corpus diacrónico del español de México. Lo siguiente es tomado del Protocolo del CHEM escrito por el Dr. Alfonso Medina.

Este proyecto surge gracias al apoyo del proyecto CONACyT R37712A ("Desarrollo del Corpus Lingüístico en Ingeniería") y del proyecto extraordinario DGAPA para complementarlo (IX402204, "Constitución de corpus lingüísticos electrónicos"), actualmente se ha constituido un corpus sincrónico abierto de especialidad, El Corpus Lingüístico en Ingeniería, el cual ha involucrado a becarios y servidores sociales, principalmente de la Facultad de Filosofía y Letras y de la Facultad de Ingeniería, que han participado en la digitalización y reconocimiento óptico de caracteres de documentos de las diversas áreas temáticas de la ingeniería. Otros alumnos, principalmente de la Facultad de Contaduría y Administración, han participado en el desarrollo de una base bibliográfica para dicho corpus, la cual permite contabilizar las cantidades y tipos de documentos, así como de las palabras y tipos de palabras, de la colección de textos ya existente. Todo esto permite controlar la representatividad estadística de las áreas temáticas y los tipos de textos que la conforman.

El conocimiento y la infraestructura generados en la constitución de este corpus, así como los resultados de otro proyecto de corpus no electrónico de carácter diacrónico (también financiado por el CONACyT: 30873H “Generación de infraestructura filológica para la investigación y la docencia”) pueden aprovecharse en el desarrollo de otros corpus electrónicos, ahora de naturaleza diacrónica, y de las herramientas para explotarlos. De allí la idea de constituir el Corpus Histórico del Español de México, que aprovecharía los recursos creados desde la filología y la ingeniería.

El objetivo general y más importante de este proyecto es instituir un corpus diacrónico del español de México que pueda ser utilizado por estudiosos del lenguaje mediante una interfaz en internet. Esto implica varios objetivos particulares que se apoyan en lo construido gracias a los proyectos CONACyT y DGAPA mencionados arriba:

1) Utilizar los recursos, la infraestructura y métodos creados en la construcción del Corpus Lingüístico en Ingeniería para generar los recursos y la infraestructura similares necesarios para almacenar el corpus histórico propuesto.

2) Diseñar y desarrollar herramientas para hacer exploraciones específicamente diacrónicas del corpus (si bien se utilizarán las desarrolladas para el Corpus Lingüístico en Ingeniería, éstas son de carácter exclusivamente sincrónico).

3) Diseñar el mapa del corpus por cada siglo (XVI, XVII, XVIII y XIX), tomando en cuenta géneros literarios (prosa, poesía, ensayo, etc.) y temáticos (literatura, gobierno, religión, ciencia, etc.), tipos de textos (libros, artículos, periódicos), autores prominentes, colecciones y archivos disponibles.

4) Localizar, obtener y digitalizar los textos clave para cada siglo del mapa del corpus. Esto incluye tanto los textos ya capturados electrónicamente, con los que se propone inaugurar el corpus (por ej. Company, Documentos lingüísticos de la Nueva España, 1994), como los textos no digitalizados que los expertos juzguen como clave del mapa del corpus (a diseñarse según el objetivo 3).

Investigaciones lingüísticas descansan bajo la premisa de que los datos empíricos que proporcionan los corpus son la mejor manera de conocer diversos fenómenos del lenguaje. Este es el caso, en especial, de los estudios diacrónicos, para los que no se puede contar con los conocimientos y la intuición de hablantes de los estados de lengua que se comparan.

La automatización de las búsquedas de estructuras gramaticales específicas en corpus electrónicos permite ahorrar mucho tiempo y esfuerzo en las investigaciones diacrónicas, que tradicionalmente requerían de la lectura rápida pero atenta de los documentos en papel que se tomaban como muestra.

La institución del CHEM permitirá el aprovechamiento de diversos recursos ya existentes. Por una parte, la metodología que se ha desarrollado para compilar corpus electrónicos podrá aplicarse y adaptarse en la compilación de un corpus diacrónico. Por otra, las ediciones electrónicas de textos mexicanos de los últimos cinco siglos, cuya elaboración requirió grandes esfuerzos y cuidados por parte de sus editores (lo cual implica el factor tiempo), tendrán a su disposición la infraestructura necesaria para que futuras investigaciones puedan aprovecharlas.

La distribución de metas por año reflejan los objetivos particulares del proyecto, el primer año serían *actividades* de instauración y planeación de recursos del corpus, el segundo año, las metas comprenden la revisión del mapa del corpus para evaluar su reformulación, implementar herramientas de etiquetado de partes de la oración y la localización, obtención y digitalización de los textos clave del mapa del corpus, según los participantes expertos.

El último año tendría tres metas principales, de las cuales la que intentamos cubrir con esta tesis es la implementación de las herramientas de análisis diacrónico para el corpus.

Hay dos vertientes de estrategias o metodologías para este proyecto: una dedicada a la planeación del contenido del corpus para garantizar la variedad de discursos y fuentes necesaria de un corpus representativo y otra relacionada con la infraestructura para el corpus electrónico.

La primera vertiente del proyecto tiene que ver con la planeación del contenido del corpus dada la variedad de textos y documentos producidos en México desde el siglo XVI,

El corazón inicial del corpus estará constituido por los materiales que ha editado la Dra. Concepción Company (del Instituto de Investigaciones Filológicas de la UNAM) y su equipo de trabajo. Esto permitirá tener de manera inmediata un prototipo del corpus.

Por lo que se refiere a la segunda vertiente del proyecto, la construcción de un corpus electrónico requiere de una metodología compleja que involucre a digitalizadores, archivadores, programadores y administradores. Así, una de las estrategias principales de este proyecto consiste en establecer una cadena de procesamiento de documentos que convierta cada texto en una porción electrónica del corpus. El mecanismo de control para saber qué contiene y qué le falta al corpus es una base bibliográfica electrónica. Ésta orientará a los administradores en cuanto a qué género, tipo textual y a qué área temática (entre otros parámetros pertinentes) deberán pertenecer los documentos que se vayan incorporando al corpus para garantizar su representatividad como muestra del español de México escrito en los últimos siglos. En síntesis, todo esto se ha desarrollado para corpus sincrónicos en el marco de los proyectos CONACyT y DGAPA mencionados arriba y se puede resumir de la siguiente manera:

1) **Construcción de una base bibliográfica para controlar el contenido del corpus.** Esta base genera informes que permiten visualizar la distribución de los documentos

del corpus según su área temática y tipo textual. Otros parámetros se agregarán para visualizar las cantidades de palabras por siglo y género literario.

2) **Digitalización de los documentos.** Una vez trazado y revisado el mapa del corpus (por siglo, por género, área temática, tipo textual, etc.), será posible proceder a digitalizar algunos documentos considerados clave.
3) **Etiquetado de los textos.** Una parte importante de la cadena de procesamiento de documentos para incorporarlos al corpus es la aplicación de etiquetas para conservar los formatos de los originales y reflejar la estructura y carácter gramatical de las construcciones lingüísticas de los textos. La primera parte encuentra ya su funcionalidad en la infraestructura técnica que proporciona el Corpus Lingüístico en Ingeniería. La segunda se desarrollaría específicamente para facilitar las búsquedas de carácter diacrónico.
4) **Planeación, diseño y desarrollo de herramientas.** A las herramientas de análisis sincrónico ya disponibles en el Corpus Lingüístico en Ingeniería, se les agregarán aquellas de carácter diacrónico que permitan comparar varios estados de lengua.

I.2.9 Bases de Datos

Existen diferentes definiciones de una base de datos, sin embargo, la mayoría son similares, Date define una base de datos como: "Un sistema de mantenimiento de registros basado en computadores, es decir, un sistema cuyo propósito general es registrar y mantener información" **(Date, 1995:5).**

En general una base de datos es un conjunto de registros almacenados y administrados generalmente mediante computadoras, cuyo objetivo es permitir registrar, eliminar o modificar información, es una tecnología muy útil en diferentes campos de las actividades humanas, particularmente en las comerciales, no obstante son herramientas valiosas en la investigación y educación; un ejemplo de ello es la implementación de Bases de Datos en el estudio de los Corpus Lingüísticos para proporcionar los medios necesarios de almacenamiento para realizar investigaciones rápidas y significativas.

La principal ventaja de las Bases de Datos es que proporcionan un control centralizado o distribuido de sus datos, que permite:

- Disminuir redundancia.
- Evitar la inconsistencia.
- Compartir información.
- Hacer cumplir las normas establecidas.
- Aplicar restricciones de seguridad.
- Conservar la integridad de los datos.
- Equilibrar los requerimientos contradictorios.

Existen diferentes tipos de bases de datos, jerárquicas, distribuidas, orientadas a objetos, sin embargo nos enfocaremos en las bases de datos relacionales que son las que se utilizan en el desarrollo del CHEM.

I.2.9.1 Base de datos Relaciónales

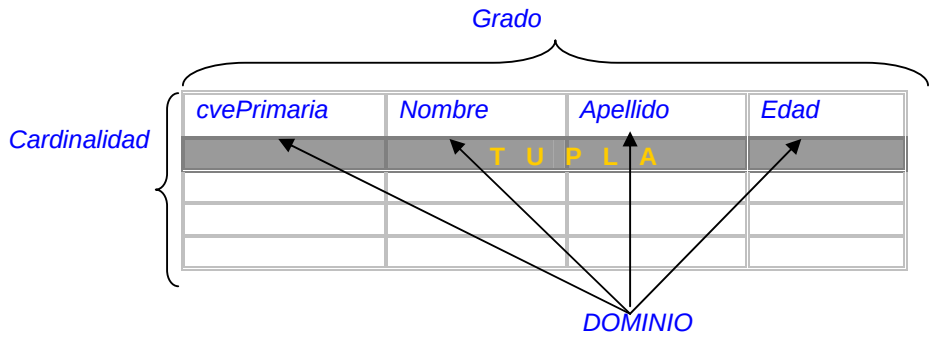
Son bases de datos cuya idea fundamental es el uso de relaciones en términos de la Teoría de Conjuntos. Se desarrollan a partir de 1972 por Edgar Codd, quien propone este modelo de datos como una solución para evitar la redundancia y asegurar la integridad de los datos. **(Date, 1995:5)**

Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados tuplas. Es decir, la representación de los datos se realiza pensando en cada relación como si fuese una tabla compuesta por columnas y renglones. Cada columna está representada mediante un nombre o atributo.

Una tabla posee las siguientes propiedades:

- Cada elemento de la matriz rectangular, representa un ítem de datos elemental.
- Todos los ítems de datos elementales de una columna (en todas las filas) son de la misma clase y, por tanto, están definidos en el mismo dominio de datos y representan una misma propiedad o característica.
- Cada columna de la tabla tiene asignado un nombre único, aunque pueden existir tablas diferentes con columnas de igual nombre.
- Para una tabla todas las filas son diferentes, no se admiten filas duplicadas.
- Tanto las filas como las columnas pueden ser consideradas en cualquier secuencia sin afectar, por ello, ni al contenido de la información ni a la representación semántica de la misma.

Cada tabla o relación cuenta con las siguientes características:



1. **Dominio:** Conjunto de valores para cada una de las columnas.
2. **Tupla:** Cada renglón de la tabla o registro.
3. **Cardinalidad:** Número de tuplas de una relación.
4. **Grado:** Número total de columnas.
5. **Atributo:** Representa el uso de un dominio de la relación.

Estos términos son muy abstractos, en la práctica son sustituidos por otros de uso más común **(WB,10).**

Término relacional formal	Equivalente informal
Relación	Tabla
Tupla	Fila o registro
Cardinalidad	Número de filas o registros
Atributo	Columna o campo
Grado	Número de columnas o campos
Clave primaria	Identificador único

Una característica adicional de las tablas o relaciones de una Base de Datos Relacional es que existen atributos únicos y por lo tanto se pueden utilizar para identificar un registro. No todas las relaciones tiene una llave primaria de un solo atributo; sin embargo cada relación tendrá, alguna combinación de atributos que, tomados en conjunto tiene la propiedad de la identificación única.

Existen diferentes tipos diferentes de llaves:

- **Llave primaria:** Es la columna que identifica como única ese renglón.
- **Llave foránea:** Columna de una tabla que hace referencia a una llave primaria de otra tabla.
- **Llave Candidata:** Es una columna que también puede ser llave primaria en una tabla.
- **Llave Alternativa:** Llaves candidatas que no han sido elegidas.
- **Llave Simple:** Llave compuesta sólo de un atributo.
- **Llave Compuesta:** Llave compuesta de más de un atributo.

I.2.9.2 Integridad

Como se mencionó anteriormente, una de las ventajas de las bases de datos relacionales es mantener la integridad de la información. Por lo tanto es importante satisfacer las siguientes reglas para garantizar la consistencia de los registros **(Castaño, 1999:107)**.

- La primera regla de integridad se aplica a las claves primarias de las relaciones base: ninguno de los atributos que componen la clave primaria puede ser nulo.
- La segunda regla de integridad se aplica a las claves ajenas: si en una relación hay alguna clave ajena, sus valores deben coincidir con valores de la clave primaria a la que hace referencia, o bien, deben ser completamente nulos.
- Otras restricciones: Además de las dos reglas de integridad anteriores, los usuarios o los administradores de la base de datos pueden imponer ciertas restricciones específicas sobre los datos, denominadas reglas de negocio. Por ejemplo, valor máximo y mínimo, lista de valores, etc.

I.2.9.3 Normalización

Normalización es el proceso que consiste en aplicar una serie de reglas con la finalidad de:

- Evitar redundancia
- Problemas de actualización
- Proteger la integridad de los datos

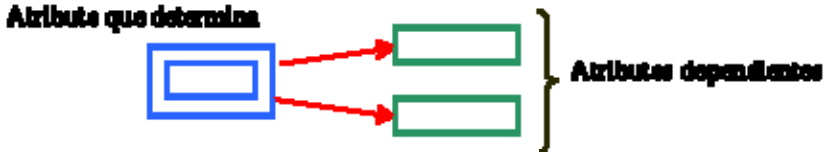
Edgar Codd definió originalmente tres formas de normalizar una relación, mediante el rompimiento de una relación principal en subrelaciones, sin embargo, estás tres reglas eran insuficientes. Una revisión posterior, realizada por Boyce y Codd dio como resultado una nueva 3FN la cuál se llama Forma Normal de Boyce Codd (FNBC). Posteriormente Fagin **(Fagin, R. 1977:54)** definió una Cuarta Forma Normal (4FN) y más adelante otra forma normal llamada Forma Normal de Proyección-Reunión (FN/PR o 5FN) se suma al proceso de normalización.

Se dice que una relación está en una Forma Normal si satisface un conjunto específico de restricciones. **(Date, 1995:267)**

A continuación se explican cada una de las cinco Formas Normales **(Pérez, 2007:129**).

1. Primera forma normal: **"**Una relación R está en primera Forma Normal (1FN) si y sólo si todos los dominios subyacentes sólo contienen valores atómicos." Es decir, una tabla está en Primera Forma Normal sólo si:
   a. Todos los atributos son atómicos. Un atributo es atómico si los elementos del dominio son indivisibles, mínimos.
   b. La tabla contiene una clave primaria
   c. La tabla no contiene atributos nulos
   d. Si no posee ciclos repetitivos

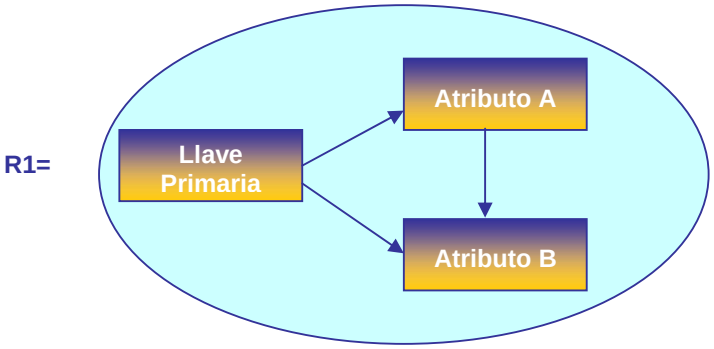
2. Segunda forma normal. Para definirla formalmente se debe entender el concepto de dependencia funcional, el cual consiste de identificar que atributos dependen de otro(s) atributo(s).



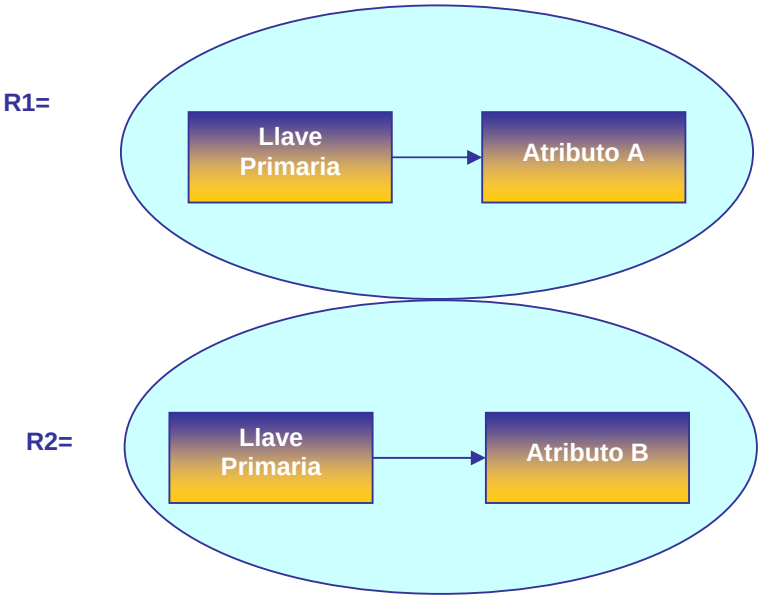
"Una relación R está en 2FN si y sólo si está en 1FN y los atributos no primos dependen completamente de la llave primaria."
Una relación se encuentra en segunda forma normal, cuando cumple con las reglas de la primera forma normal y todos sus atributos que no son claves (llaves) dependen por completo de la clave.

3. Tercera Forma Normal: "Una relación se encuentra en Tercera Forma Normal (3FN) si y sólo si está en 2FN y todo atributo no primo es dependiente no transitivamente de la llave primaria."

Un ejemplo de transitividad es el siguiente:



La dependencia transitiva se rompería al generar una nueva relación de la siguiente forma:



4. Cuarta forma normal: "Un esquema de relaciones R está en 4FN con respecto a un conjunto D de dependencias funcionales y de valores múltiples sí, para todas las dependencias de valores múltiples en D de la forma X->->Y, donde X sea un elemento de la relación<=R y Y<=R, se cumple por lo menos una de estas condiciones:
   - X->->Y es una dependencia de valores múltiples.
   - X es una llave primaria de la relación R."

   Se tienen dependencia multivaluada cuando un valor de una variable está siempre asociado con varios valores de otra u otras variables dependientes que son siempre las mismas y están siempre presentes **(Ruiz, 2001:81)**.

Para que esto sea más claro pondré el siguiente ejemplo: una tabla llamada estudiante que contiene los siguientes atributos: Clave, Especialidad y Curso tal y como se demuestra en la siguiente figura:

Clave	Especialidad	Curso
S01	Sistemas	Natación
S01	Bioquímica	Danza
S01	Sistemas	Natación
B01	Bioquímica	Guitarra
C03	Civil	Natación

En este caso se tienen los siguientes supuestos:

- Los estudiantes pueden inscribirse en varias especialidades y en diversos cursos.
- El estudiante con clave S01 tiene su especialidad en sistemas y Bioquímica y toma los cursos de Natación y danza,
- El estudiante B01 tiene la especialidad en Bioquímica y toma el curso de Guitarra.
- El estudiante con clave C03 tiene la especialidad de Civil y toma el curso de natación.

En esta tabla no existe dependencia funcional porque los estudiantes pueden tener distintas especialidades y un valor único de clave puede poseer muchos valores de especialidades al igual que de valores de cursos. Por lo tanto existe dependencia de valores múltiples **(Ruiz, 2001:85).**

Este tipo de dependencias produce redundancia de datos, como se puede apreciar en la tabla anterior, en donde la clave S01 tiene tres registros para mantener la serie de datos en forma independiente lo cual ocasiona que al realizarse una actualización se requiera de demasiadas operaciones para tal fin.

En la tabla anterior Clave determina valores múltiples de especialidad y clave determina valores múltiples de curso, pero especialidad y curso son independientes entre sí.

Las dependencias de valores múltiples se definen de la siguiente manera: Clave ->-> Especialidad y Clave->->Curso; Esto se lee "Clave multidetrmina a Especialidad, y clave multidetermina a Curso"

Para eliminar la redundancia de los datos, se deben eliminar las dependencias de valores múltiples. Esto se logra construyendo dos tablas, donde cada una almacena datos para solamente uno de los atributos de valores múltiples.

Para nuestro ejemplo, las tablas correspondientes son:
Tabla Especialidad

Clave	Especialidad
S01	Sistemas
B01	Bioquímica

C03	Civil

Tabla Curso

Clave	Curso
S01	Natación
S01	Danza
B01	Guitarra
C03	Natación

5. Quinta Forma normal: Una tabla se encuentra en 5FN si:

   - La tabla esta en 4FN
   - No existen relaciones de dependencias no triviales que no siguen los criterios de las claves. Una tabla que se encuentra en la 4FN se dice que esta en la 5FN si, y sólo si, cada relación de dependencia se encuentra definida por las claves candidatas

I.2.9.4 Algebra relacional

Dado que las Base de datos se basan en la Teoría de Conjuntos, las operación que se hacen con las entidades, se realizan con la ayuda del algebra relacional, la cual nos permite obtener las consultas necesarias de la información almacenada en una o varias tablas.

El algebra relacional es un conjunto de operaciones que nos permiten manipular información en un sistema relacional. Estos operadores deberán ser parte del sublenguaje de datos del manejador de base de datos. Se aplica sobre relaciones (tablas) y da como resultado otras relaciones (**García, 2007:24**)

Operadores:

- **Unión** Es un operador tradicional, construye una relación a partir de dos relaciones concatenando todas las tuplas posibles de estas (elimina duplicados), C:=A Unión B.
- **Intersección**: Es un operador tradicional, construye una relación a partir de todos los registros que aparecen en las dos relaciones de entrada, C:=A Intersección B.
- **Diferencia**: Es un operador tradicional, construye una relación a partir de los registros que aparecen en A y no en B, C:=A Diferencia B.
- **Producto** Es un operador tradicional, la relación resultante se compone de todas las combinaciones posibles de las columnas de ambas relaciones de entrada, C:=AB.
- **Selección** Selecciona registros de A de acuerdo a un criterio específico y forma con ellas una nueva relación.
- **Proyección** Selecciona columnas (atributos) de la relación A y forma con ellos una nueva relación.

I.2.9.5 Las 12 reglas de Codd

Para la década de los 80 la difusión de las ventajas que proporcionaban las bases de datos relacionales, ya había alcanzado una importancia considerable, por lo cual comenzaron a aparecer numerosos SGBD que intentaban posicionarse como Sistemas de Bases de Datos relacionales. Sin embargo estos sistemas carecían de muchas características que se consideran importantes en un sistema relacional, perdiendo muchas ventajas de dicho modelo (**García, 2007:89)**.

En 1984 Edgar Codd publicó 12 reglas con las que un verdadero Sistema Manejador de Bases de Datos Relacional debería de cumplir. En la práctica incluso algunos SMBD no las cumplen en su totalidad **(WB, 06)**. Por lo tanto, un sistema podrá considerarse "más relacional" cuanto más siga estas reglas.

1. Regla 0**:** "Para que un sistema se denomine sistema de gestión de bases de datos relacionales, este sistema debe usar (exclusivamente) sus capacidades relacionales para gestionar la base de datos*".* Significa que para que un sistema de Base de Datos se considere relacional, debe cumplir con las siguientes 12 reglas.

2. Regla 1 de la información: "Toda la información en una base de datos relacional se representa explícitamente en el nivel lógico exactamente de una manera: con valores en tablas." Una base de datos Relacional permite almacenar los datos en Tablas, las cuales:
   - Son un conjunto de columnas y renglones formando celdas que permiten almacenar valores atómicos.
   - Cada fila se identifica como única por medio de la llave primaria, lo cual significa que no puede tener 2 filas iguales.
   - Las celdas permiten almacenar valores nulos, los cuales representan valores desconocidos los cuales son diferentes a 0 o a un conjunto vacío.

3. Regla 2 acceso garantizado: "Para todos y cada uno de los datos (valores atómicos) de una BDR se garantiza que son accesibles a nivel lógico utilizando una combinación de nombre de tabla, valor de clave primaria y nombre de columna". Es decir, cualquier dato almacenado en una BDR tiene que poder ser recuperado unívocamente. Para lo cual hay que indicar en qué tabla está, cuál es la columna y cuál es la fila (mediante la clave primaria). Para esto es necesario:
   - Hacer que los atributos clave primaria no puedan ser nulos (NOT NULL).
   - Crear un índice único sobre la clave primaria.
   - No eliminar nunca el índice.

4. Regla 3: Tratamiento sistemático de valores nulos: Los valores nulos (que son distintos de la cadena vacía, blancos o 0) se soportan en los SGBD totalmente relacionales para representar información desconocida o no aplicable de manera sistemática, independientemente del tipo de datos. Es decir, se crean tablas de verdad para las operaciones lógicas.
   - null Y null = null
   - Verdadero Y null = null
   - Falso Y null = Falso
   - Verdadero O null = Verdadero

5. Regla 4: Catálogo dinámico en línea basado en el modelo relacional. “La descripción de la base de datos se representa a nivel lógico de la misma manera que los datos normales, de modo que los usuarios autorizados pueden aplicar el mismo lenguaje relacional a su consulta, igual que lo aplican a los datos normales.” Está regla nos indica que una base de datos tiene la capacidad de almacenar datos o metadatos de todos sus componentes y que estos datos pueden ser consultados por los usuarios, con ello podemos conocer el nombre de columna, tipo de dato, longitud, si acepta valores nulos o si se trata de una llave primaria o foránea mediante el uso de comandos o sentencias SQL.

6. Regla 5: Sublenguaje de datos completo.**:** **“**Un sistema relacional debe soportar varios lenguajes y varios modos de uso de terminal (ej: rellenar formularios, etc.). Sin embargo, debe existir al menos un lenguaje cuyas sentencias sean expresables, mediante una sintaxis bien definida, como cadenas de caracteres y que sea completo, soportando”:

7. Regla 6: Actualización de vistas:”Todas las vistas que son teóricamente actualizables se pueden actualizar por el sistema.” Está regla es confusa, dado que algunos autores mencionan que el problema para su esclarecimiento es definir cuales son las vistas teóricamente actualizables. Por lo tato es importante aclarar que es una vista, según Date es una tabla que no tiene existencia por derecho propio, sino que se deriva de una o varias tablas de base. **(Date, 1995:109)**

   Una tabla de base es una tabla almacenada en la base de datos, es una tabla independiente y que por sí sola tiene existencia en la base de datos. Es decir, tenemos una base de datos de nombre administraciónEscolar en la cual existen las siguientes tablas:
   - Alumno
   - Profesor
   - Materia

   La tabla Alumno contiene los siguientes campos:

   Alumno=

numCuenta	NombreA	ApellidoPA	ApellidoMA	FechaNacimientoA

   La tabla Profesores contiene los siguientes campos:

   Profesor=

Matricula	NombreP	ApellidoPP	ApellidoMP	FechaNacimientoP

   La tabla Materia contiene los siguientes campos:

   Materia=

CveMateria	Nombre	Temario

   Una vista derivada de estas tres tablas sería la siguiente:

vInscripción=

numCuenta	NombreA ApellidoPA ApellidoMA	Materia	NombreP ApellidoPP ApellidoMP

Ahora bien, la regla de actualización de vistas, indica que si un usuario hiciera alguna modificación sobre está vista, todas y cada una de las tablas bases involucradas se tendrían que actualizar dinámicamente, es decir en automático y sin la intervención humana.

8. Regla 7: Inserción, actualización y borrado de alto nivel. "La capacidad de manejar una relación base o derivada como un solo operando se aplica no sólo a la recuperación de los datos (consultas), si no también a la inserción, actualización y borrado de datos." Esto es, el lenguaje de manejo de datos también debe ser de alto nivel, es decir, debe manejar cada uno de los objetos a nivel de conjuntos, es decir cada tabla es un conjunto y mediante una sola operación se puede insertar, modificar o eliminar registros en una o varias tablas.

9. Regla 8: Independencia física de los datos. **"**Los programas de aplicación y actividades de la Terminal permanecen inalterados a nivel lógico en el momento en que se realicen cambios en las representaciones de almacenamiento o métodos de acceso." El modelo relacional es un modelo lógico de datos, y oculta las características de su representación física, es decir, no importa el medio de almacenamiento en el que se encuentre (disco duro, cd, o zip, etc) de tal forma que en el momento en que éste cambie, la base de datos no sufre ninguna alteración en sus datos o información.

10. Regla 10: Independencia lógica de datos: **"**Los programas de aplicación y actividades del terminal permanecen inalterados a nivel lógico en el momento que se realicen cambios a las tablas base que preserven la información." Cuando se modifica el esquema lógico de la base de datos, no tiene porque afectar las aplicaciones, siempre y cuando se conserve la integridad de los datos. Algunos ejemplos de cambios que preservan la información son los siguientes:
    - Añadir un atributo a una tabla base.
    - Sustituir dos tablas base por la unión de las mismas. Usando vistas de la unión puedo recrear las tablas anteriores.

11. Regla 10 Independencia de Integridad: **"**Los limitantes de integridad específicos para una determinada base de datos relacional deben poder ser definidos en el sublenguaje de datos relacional, y almacenables en el catálogo, no en los programas de aplicación." Esta regla quiere decir que todas las restricciones se deben almacenar en el diccionario de datos. Cuando se modifican las restricciones no debe afectar a las aplicaciones.

    El objetivo de las bases de datos no es sólo almacenar los datos, si no también sus relaciones y evitar que estas (limitantes) se codifiquen en los programas. Por tanto en una Base de Datos Relacional se deben poder definir limitantes de integridad. Como parte de los limitantes inherentes al modelo relacional están:
    - Una BDR tiene **integridad de entidad**. Es decir, toda tabla debe tener una clave primaria.
    - Una BDR tiene **integridad referencial**. Es decir, toda clave externa no nula debe existir en la relación donde es primaria.

12. Regla 11: Independencia de Distribución: “Una BDR tiene independencia de distribución.” Es decir, todas las reglas anteriores aplican a todos los elementos de la base de datos distribuida. Esta regla es responsable de tres tipos de transparencia de distribución, es decir de tres cuestiones que el usuario no puede ver:
    - Transparencia de localización. El usuario tiene la impresión de que trabaja con una BD local. (aspecto de la regla de independencia física).
    - Transparencia de fragmentación. El usuario no se da cuenta de que la relación con que trabaja está fragmentada. (aspecto de la regla de independencia lógica de datos).
    - Transparencia de replicación. El usuario no se da cuenta de que pueden existir copias (réplicas) de una misma relación en diferentes lugares.

13. Regla 13: No Subversión: “Si un sistema relacional tiene un lenguaje de bajo nivel (un registro cada vez), ese bajo nivel no puede ser usado para saltarse (subvertir) las reglas de integridad y los limitantes expresados en los lenguajes relacionales de más alto nivel (una relación (conjunto de registros) de cada vez).” Cualquier lenguaje de bajo nivel no se puede oponer a las restricciones y a las reglas de integridad definidas. Normalmente se usa SQL inmerso en un lenguaje anfitrión para solucionar estos problemas y se utiliza el concepto de cursor para tratar individualmente los registros de una relación. En cualquier caso no debe ser posible saltarse los limitantes de integridad impuestos al tratar las tuplas a ese nivel **(Velthuis, 2007:149).**

I.2.10 Características de SQL (Structured Query Language).

Cómo se estudio en la sección anterior el modelo relacional de Bases de Datos fue diseñado por el investigador de la IBM Edgar Codd, sin embargo para que fuera posible manejar de forma fácil la información mediante el uso de tablas, era necesario crear un lenguaje que permitiera la manipulación de la información

Para cumplir este propósito se diseño el lenguaje SQL, el cual es un lenguaje estandarizado de base de datos, que permite realizar tablas y obtener datos de ellas de manera sencilla.

Este lenguaje tiene muchas características muy importantes, el objetivo de este capitulo no es profundizar el las sentencias, sintaxis o administración de la base de datos, sólo se revisarán algunas de las características relevantes para el procesamiento de corpus.

SQL se considera un sublenguaje de programación cuyo propósito es la definición, manipulación y control de datos. Este lenguaje sólo puede ser empleado para manejar base de datos. Para generar una aplicación, es preciso integrar SQL en algún otro lenguaje de programación como JAVA, PHP o cualquier otro que contenga las estructuras adecuadas dependiendo de la función de dicha aplicación.

Un sublenguaje no es un lenguaje de aplicación con características completas. Asimismo, un lenguaje de aplicación con características completas incluye semántica para los procedimientos, en tanto que el SQL no se basa en los procedimientos. No especifica cómo se debe hacer algo, sólo especifica qué se debe hacer. En otras palabras, SQL se interesa en los resultados más que en los procedimientos **(Cornelio, 2002:28)**.

Una de las herramientas lógicas más poderosas de SQL es el reconocimiento de un patrón de consulta, datos parcialmente recordados. Los patrones de consulta juegan un papel importante en el momento de realizar consultas, ya que es común que necesitemos encontrar un texto y que no recordemos exactamente cómo fue ingresado. Con el uso del operador ILIKE podemos comparar patrones y ubicar un texto, independientemente de la posición en que se encuentre, este es uno de los motivos por los que las bases de datos relacionales, son una herramienta clave en el procesamiento y análisis de los corpus **(Beaulieu, 2006:131)**.

Para la definición del patrón de consulta también existen dos tipos de caracteres especiales:

- % (signo de porcentaje) llamado comodín, representa cualquier cantidad de espacios o caracteres en esa posición. Significa que se admite cualquier cosa en su lugar: un caracter, cien caracteres o ningún caracter.

- _ (signo de subrayado) llamado marcador de posición, representa exactamente una posición e indica que puede existir cualquier caracter en esa posición.

Otra característica importante de SQL son las subconsultas, que son aquella consulta de cuyo resultado depende otra consulta, llamada principal, y se define como una sentencia SELECT que esta incluida en la orden WHERE de la consulta principal. Una subconsulta, a su vez, puede contener otra consulta y así hasta un máximo de 16 niveles. Sus particularidades son:

1. Su resultado no se visualiza, sino que se pasa a la consulta principal para su comprobación.

2. Puede devolver un valor único o una lista de valores y dependiendo de esto se debe usar el operador del tipo correspondiente.
3. No puede usar el operador BETWEEN, ni contener la orden ORDER BY.
4. Puede tener una sola columna, que es lo más común, o varias columnas. Este último caso se llama subconsulta con columnas múltiples. Cuando dos o más columnas serán comprobadas al mismo tiempo, deben encerrarse entre paréntesis.

Para combinar grupos con subconsulta debemos incluir en la sentencia SELECT la orden HAVING, que tiene las siguientes características:

1. Funciona como la orden WHERE, pero sobre los resultados de las funciones de grupo, en oposición a las columnas o funciones para registros individuales que se seleccionan mediante la orden WHERE. O sea, trabaja como si fuera una orden WHERE, pero sobre grupos de registros.
2. Se ubica después de la orden GROUP BY.
3. Puede usar una función de grupo diferente a la de la orden SELECT.

I.2.10.1 Índices

El índice es un instrumento que aumenta la velocidad de respuesta de la consulta, mejorando su rendimiento y optimizando su resultado.

El índice tiene un funcionamiento similar al índice de un libro, guardando parejas de elementos: el elemento que se desea indexar y su posición en la base de datos. Para buscar un elemento que esté indexado, sólo hay que buscar en el índice dicho elemento para, una vez encontrado, devolver el registro que se encuentre en la posición marcada por el índice.
Los índices pueden ser creados usando una o más columnas, proporcionando la base tanto para búsquedas rápidas al azar como de un ordenado acceso a registros eficiente.

La identificación del índice a usar está relacionada con las columnas que participan en las condiciones de la orden WHERE. Si la columna que forma el índice está presente en alguna de las condiciones éste se activa.

De acuerdo al análisis realizado para seleccionar la solución técnica, el cual se puede ver en el siguiente capitulo, el manejador de bases de datos a utilizar para el procesamiento de corpus es PostgreSQL, por lo tanto en está sección me centrare en los índices que proporciona, B-tree, R-Tree, Hash, y GiST. Cada tipo del índice utiliza un algoritmo diferente que se satisface los diferentes tipos de queries **(The PostgreSQL Global Development Group, 2005, Capitulo 11.2)**.

Los b-trees pueden manejar queries de un rango de datos que se pueden clasificar con cierto orden. El comando CREATE INDEX crea un índice b-tree, que ayuda en las situaciones más comunes.

PostgreSQL considerará el uso de un índice b-tree siempre que una columna puesta en un índice esté implicada en una comparación usando uno de estos operadores: >, <, <=, =, >=. Construcciones equivalentes para combinar esos operadores, como BETWEEN e IN, pueden ser implementadas con una búsqueda indexada por medio de B-tree. Un operador IS NULL no equivale a un operador = y no es indexable **(The PostgreSQL Global Development Group, 2005:Capitulo 11.2)**.

El planificador puede usar también un indice b-tree en queries que incluyen comodines mediante búsquedas con el operador LIKE, si los patrones son una constante anclada al inicio de una cadena. Por ejemplo: LIKE 'foo%', pero no en los casos '%bar'.

Es también posible utilizar los índices b-tree para el operador ILIKE y el ~ *, pero solamente si el patrón comienza con caracteres que no son afectados por la conversión upper o lower case.

Los índices R-tree están diseñados para satisfacer queries de dos dimensiones. Para crear un índice del R-tree, se utiliza el comando:

CREATE INDEX nombreIndice ON nombreTabla USING rtree nombreColumna;

PostgreSQL considera el uso de un índice R-Tree siempre que una columna indexada esté involucrada en una consulta donde se usen alguno de los siguientes operadores: `<<, &<, &>, >>, <<|, &<|, |&>, |>>, ~, @, &&`

El tercer tipo de índice es el Hash, el cual sólo puede manejar simples comparaciones de igualdad. Postgres considera el uso de este tipo de índices cuando una consulta solo incluye el operador =. El comando para generar un índice HASH es el siguiente:

CREATE INDEX *name* ON *table* USING hash (*column*);

El índice GiST es una infraestructura dentro de la cual se incluyen diferentes estrategias de indexación que pueden ser implementados. Es un balance, un método de acceso estructurado y funciona como una platilla base en la que se implementan arbitrariamente esquemas como el B-Tree, R-tree o cualquier otro esquema **(The PostgreSQL Global Development Group, 2005:Capitulo 11.2)**.

Una ventaja es que permite el desarrollo de tipos de datos personalizados con un método de acceso apropiado por un experto en el domino de los tipos de datos las bases de datos más bien que por un experto en base de datos.

Después de revisar los diferentes índices, es importante mencionar que en el desarrollo del CHEM, se utilizaran índices B-tree dado el tipo de operadores que se utilizaran en las consultas, además de revisar una nota aclaratoria del capitulo 11 de la documentación de Postgres donde menciona que los índices hash no son mejores que los B-Tree y necesitan ser reconstruidos cuando ocurre algún problema en la base de datos. De igual manera, el mismo capitulo no recomienda el uso de los R-Tree y recomienda una migración a los índices GiST.

I.2.10.2 Indices Clustered y nonclustered

Los índices clustered son útiles para las columnas que son muy consultadas por algún rango o accedidas de forma ascendente o descendente, ayuda a reducir las búsquedas por páginas ya que cuando se recuperan los datos de la tabla la localización en la misma página es mucho más rápida pues se tiene el punto de inicio y todos los datos consecutivos.

En consecuencia la ejecución del comando CLUSTERE obliga el ordenamiento físico de los datos en el mismo orden que se haya elegido para el índice ya que sólo se puede tener un orden físico y un índice CLUSTERE por tabla. Así que es impórtate elegir cuidadosamente sobre que índice se ejecutará este comando y es necesario volver a ejecutarlo para mantener el orden. En resumen, este tipo de índices no ayudan a las búsquedas sin rango **(WB,12)**.

Los índices nonclustered son muy útiles cuando el usuario requiere de múltiples manera de acceder a la información. Si no se especifica el tipo de índice al crearlo, por default se crea un nonclustered.

I.2.10.3 Cuando no debe de crearse un índice

Debe de tenerse en mente que aunque los índices son muy útiles para mejor el performance de consultas, estos consumen espacio en disco e incrementan el costo de mantenimiento. Algunos puntos que se deben tener en cuenta para crear un índice son los siguientes **(Cornelio, 2002:28)**:

1. Cuando modifique datos en una columna que se encuentra dentro de un índice, también se actualiza el índice.
2. Dar mantenimiento a los índices requieres de tiempo y recursos por esto no se deben crear índices que no se usen.
3. Los índices sobre columnas que contienen información duplicada dan pocos beneficios, es mejor no crear un índice en este tipo de columnas.
4. Existen dos tipos de índices, los índices clustered que modifican el orden en que se almacenan los datos físicamente, esto es se ordenan por el índice generado y los índices nonclustered que no modifican el orden de los datos.

I.2.10.4 PL/SQL

Es un lenguaje portable, procedural y de transacción muy potente y de fácil manejo, con las siguientes características fundamentales (**Pérez, 2008:87)**:

- Incluye todos los comandos de SQL.
- Es una extensión de SQL, ya que este es un lenguaje no completo dado que no incluye las herramientas clásicas de programación. Por eso, PL/SQL amplia sus posibilidades al incorporar las siguientes sentencias:
    - Control condicional
    - Ciclos
    - Incorpora opciones avanzadas en:
        - Control y tratamiento de errores llamado excepciones
        - Manejo de cursores
    - Estructura del bloque de código: La organización del bloque de código de PL/SQL, compuesto por cuatro secciones DECLARE, BEGIN, EXCEPTION y END.

I.2.10.5 Manejo de cursores

El conjunto de filas resultantes de una consulta con la sentencia SELECT, puede estar compuesto por ninguna, una o varias filas, dependiendo de la condición que define la consulta.

Para poder procesar individualmente cada fila de la consulta debemos definir un cursor (que es un área de trabajo en memoria) y contiene los datos de las filas de la tabla consultada por la sentencia SELECT. Los cursores se utilizan porque generalmente los lenguajes de programación son procedurales y no disponen de ningún mecanismo para manipular conjuntos de datos en una sola instrucción. Debido a ello, las filas deben ser procesadas de forma secuencial por la aplicación. Un cursor puede verse como un iterador sobre la colección de filas que habrá en el set de resultados **(Pérez, 2008:257)**.

I.2.10.6 Disparadores

El disparador es un bloque de código que se activa cuando se pulsa una determinada tecla u ocurre cierto evento, como puede ser **(Cornelio, 2002:107)**:

- Mover el cursor hacia o desde un campo, registro, bloque o forma.
- Realizar una consulta.
- Validar un dato.
- Hacer una transacción al insertar, modificar o eliminar registros de la base de datos.

I.3 Objetivo

Investigar, planear, diseñar y desarrollar las herramientas de exploración diacrónicas de corpus, en base a las herramientas desarrolladas en el Instituto de Ingeniería de la UNAM y en la Metodología propuesta por Mike Davis, mediante el uso de Bases de Datos relacionales.

I.4 Hipótesis

El uso de Base de Datos Relacionales en la explotación del Corpus Histórico del Español de México, permitirá optimizar y agilizar el procesamiento de los documentos comprendidos entre el siglo XVI y XIX con la finalidad de recuperar rápidamente concordancias a partir de la posición de cada grama, estableciendo así las bases para su análisis.

I.5 Metodología de Desarrollo de Software.

La Ingeniería de Software es la rama de la Ingeniería que crea y mantiene las aplicaciones de software aplicando tecnologías y prácticas de las ciencias computacionales, manejo de proyectos, ingeniería, el ámbito de la aplicación, y otros campos. Uno de sus objetivos ha sido encontrar procesos o metodologías predecibles y repetibles que mejoren la calidad y productividad en el desarrollo de software **(Schach, 2006:27)**.

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos de software. Son como un libro de recetas de cocina, en el que se van indicando paso a paso todas las actividades a realizar para lograr el producto deseado. Además detallan la información que se debe producir como resultado de una actividad y la información necesaria para comenzarla.

En general la Ingeniería de Software requiere ejecutar las siguientes actividades: Análisis de requerimientos, Especificación de requerimientos, Diseño y Arquitectura, Desarrollo, Pruebas, Documentación y Mantenimiento; para lo cual ha generado diferentes modelos o paradigmas de desarrollo como apoyo en la realización de software, a los cuales también se les conoce como Ciclo de Vida de Desarrollo.

I.5.1 Justificación

Para el desarrollo de este proyecto se compararon los ciclos de vida en Cascada, en Espiral, de Prototipos y en V como se muestra en el **Anexo I** y dado sus características, el ciclo seleccionado es el de **Cascada** con la siguiente justificación **(Pfleeger, 2002:163).**

- Las especificaciones son claras dado que se cuenta con una versión inicial, la cual necesita ser modificada con la finalidad de que los archivos puedan ser separados por siglos.
- Dado que no se cuenta con la experiencia en el desarrollo de sistemas y por lo tanto en estimación de tiempo, se realizará la especificación de requerimientos para poder generar un plan de trabajo.
- Dado que las actividades son secuenciales y solo se cuenta con dos recurso humanos (asesor y tesista), la administración del proyecto debe ser sencilla, además de que no se podrán realizar actividades en paralelo.

I.6 Metodología de Mark Davis

Uno de los problemas principales que se oponen a la creación de grandes corpus es la necesidad de nivelar el tamaño, rapidez y una avanzada ingeniería de búsqueda que permita un gran alance de queries **(Davis, 2003:23)**.

Existen tres opciones en la exploración y almacenamiento de corpus **(Davis, 2003:25)**:

- Una opción sería poner un número de niveles de anotaciones dentro del propio corpus textual, es decir, etiquetar cada palabra dentro del texto.
- Otra propuesta podría usar un producto como Microsoft Search que indexa y busca el corpus. Esto resultaría en queries muy rápidos, pero se limitarían a búsquedas de palabras y frases principalmente, con poca o ninguna posibilidad de anotaciones tradicionales como lema y partes del lenguaje.
- La tercera propuesta se basa en el uso de n-grama o palabras almacenada en grandes bases de datos relacionales. Esta propuesta permitiría un tamaño ilimitado de búsquedas rápidas y un gran número de anotaciones del corpus, como lema, partes del lenguaje, sinónimos, frecuencia y listas personalizadas. Además de un diseño modular que permitiría en un futuro agregar más anotaciones. Esta es la propuesta de Mark Davis.

I.6.1 Arquitectura del Corpus.

En términos del diseño general del corpus, la propuesta es única ya que hay pocas anotaciones dentro del corpus textual. Los corpus son almacenados como trozos de palabras de texto en una base de datos relacional. Este esquema indexado permite encontrar palabras y frases exactas rápidamente, generalmente en menos de un segundo en el corpus entero y regresar los ejemplos relevantes. Una limitación que Mark Davis resalta es que la ingeniería de búsqueda del texto completo sólo trabaja bien con palabras y frases exactas, sin embargo, y dado que los manejadores de datos permiten el uso de la sentencia LIKE, esto es posible.

Las anotaciones para el Corpus del Español no residen en la base de datos textual, por el contrarío están en bases de datos separadas que contienen tablas de todo los distintos gramas en el corpus, las tablas son muy grandes **(Davis, 2003:23)**.

I.6.2 Arquitectura de Anotaciones

Dado que el etiquetado es un objetivo que no se alcanzará en este trabajo, es importante conocer las propuestas dado que debe contemplarse en el diseño de la base de datos para su almacenamiento futuro.

Hay dos posibles propuestas para la cuestión de dónde y cómo poner la información del POS (Part of Speech o partes del lenguaje) y el lema en la base de datos relacional **(Davis, 2003:25)**.

Una opción es ponerlos dentro de la tabla que contiene los n-gramas y la información de frecuencias. Otra es separarlas de esta tabla y usar sentencias JOIN de SQL para unir la tabla de anotación y la de n-gramas y frecuencia.

La ventaja de tener la anotación dentro de la tabla de n-gramas y frecuencias es que la información contextual puede ser usada para resolver ambigüedades, Sin embargo usando simples modificaciones SQL, podríamos buscar información contextual para llenar o modificar los valores del POS y el lema, en base a los valores de las palabras de los otros cuadros. Además la recuperación de las palabras, así como el contexto de las mismas se puede realizar mediante la búsqueda de su posición en el texto.

Los investigadores de lenguajes como el inglés, que tiene una morfología relativamente pobre, podrían objetar la idea de tener información del POS y el lema en tablas donde ésta sería removida de su contexto original. La preocupación podría ser que no halla suficiente información en la base de datos de los n-gramas para desambiguar exitosamente los casos de polisemia. Para un lenguaje como el español esta preocupación es infundada, porque el español es un lenguaje morfológicamente fuerte que puede ser exitosamente desambiguado **(Davis, 2003:27)**.

Mark Davis propone otra solución para un lenguaje como el español, que consiste de quitar las anotaciones de la tabla de frecuencias y de n-gramas y ponerlas en una tabla separada que muestre el contexto inmediato de la palabra. El único problema en la asignación de partes del lenguaje y el lema ocurriría en aquellos casos relativamente pocos en que:

- Una palabra es poli semánticas.
- Ambos significados son altamente frecuentes.
- El contexto no es suficientemente rico para desambiguar el múltiple significado.

I.6.3 Uso de n-gramas y frecuencias para anotar formas desconocidas.

Existen dos ventajas importantes de la propuesta de las bases de datos relacionales además de la rapidez, complejidad y profundidad de los queries. La primera es la forma en que la arquitectura de la base de datos puede ser usada para asistir el desarrollo de un diccionario para el corpus. La segunda ventaja es el diseño modular, que permite un número ilimitado de tipos de anotaciones, sin decremento en su desempeño. Y la tercera y más importante ventaja es la posibilidad usar subqueries para realizar múltiples pruebas en los datos que se extraen de los textos, así como hacer diferentes tipos de búsqueda como por siglo, por anotaciones, referencia, obtener la frecuencia de aparición, etc **(Davis, 2003:29)**.

I.6.4 Justificación.

Haciendo un resumen del análisis anterior, las ventajas de la propuesta de la base de datos de n-gramas relacional y la justificación de porque se sigue esta propuesta, es que las tablas son simplemente una parte que pueden ser unidas a otras tablas dentro de la misma base de datos y no hay limite para todas las anotaciones que pueden ser aplicadas a el corpus, como sinónimos, partes del lenguaje, lemas, traducciones entre lenguajes, etimologías o listas personalizadas, y todas pueden ser unidas con una simple sentencia JOIN de SQL

II. Caso Práctico

II.1 Análisis del Sistema

II.1.1 Especificación de Requerimientos

II.1.1.1 Sistema Actual

Actualmente se cuenta con un sistema de lectura de archivos

II.1.1.2 Limitaciones del Sistema Actual

- Únicamente se puede realizar búsquedas por siglo y por tokens (palabras).
- Todos los archivos pertenecientes a un siglo, se deben concatenar para poder localizar la palabra deseada.
- No puede obtenerse la frecuencia.
- No se tiene una base de datos con la información de la referencia a la que pertenece el archivo.
- No se da a conocer al usuario el corpus donde se localiza la palabra.

II.1.1.3 Sistema Propuesto

- El sistema propuesto procesará cada texto de forma independiente y se guardará la información de cada uno en la tabla de referencia.
- Los textos serán procesados para dividirse en palabras y cada una será almacenada en la base de datos, incrementando la frecuencia de aparición en cada siglo, del XVI-XIX.
- El usuario podrá realizar consultas por siglo y palabra y por referencia y recuperar las concordancias y la frecuencia de aparición de cada una.
- La Base de datos queda abierta para que posteriormente puedan agregarse las etiquetas de cada palabra, mediante sentencias sql o bien mediante el contexto de la misma. De igual forma se crearán las tablas de Bigramas y Trigramas.

II.1.1.4 Objetivos del Sistema Propuesto

Explorar los textos del Corpus Histórico del Español de México, del siglo XVI-XIX, mediante la búsqueda de palabras y recuperación de concordancias, de tal manera que se pueda analizar el uso de la misma y su frecuencia de aparición en este período histórico de la lengua o en cada uno de los siglos.

II.1.1.5 Beneficios del Sistema Propuesto

- Se podrán hacer consultas de cada texto en forma independiente.
- Se obtendrá la frecuencia de aparición de cada palabra por siglo y posteriormente se podrá mostrar la frecuencia total del período diacrónico.
- Se podrán hacer búsquedas por siglo o por referencia.
- El sistema quedará abierto para que en otra versión se puedan llenar las tablas de trigramas y bigramas e incrementar la búsqueda por este tipo de n-gramas.
- De igual forma el sistema queda abierto para hacer búsquedas por etiquetado.

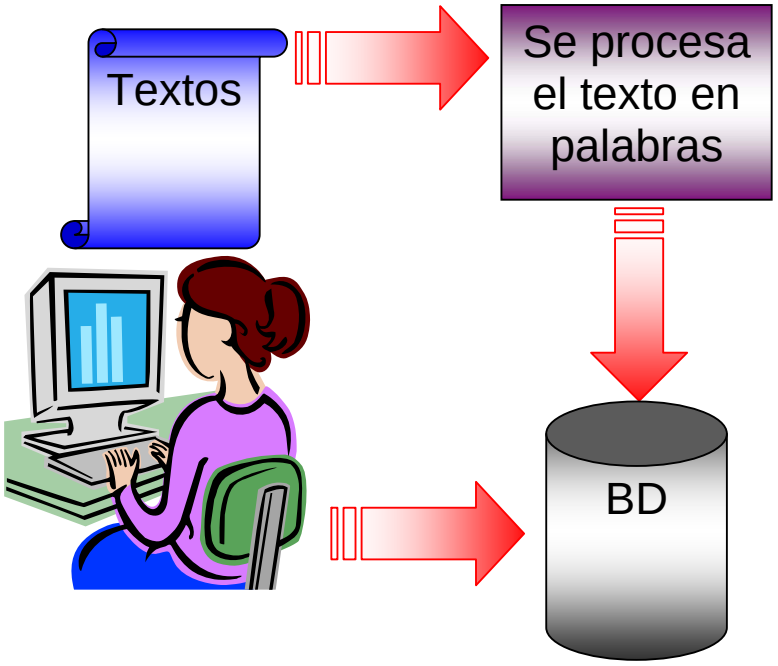
II.1.1.6 Alcance del Sistema

- El sistema podrá procesar los textos de forma independiente.
- Con las palabras contenidas en cada uno de los textos, se llenará la tabla de unigramas y se incrementará la frecuencia de aparición por siglo.
- Se llenará la tabla de referencia, con la información obtenida de los textos y posteriormente se podrá consultar directamente la base de datos Bibliográfica, como se espera en el protocolo del proyecto (Ver Antecedentes del Corpus Histórico del Español de México).
- Se incrementará la frecuencia de aparición de cada palabra en cada uno de los en la que se presenta.
- Se crea la base de datos con las tablas Bigramas y Trigramas y se agrega una columna para el etiquetado.
- Al final de esta versión solo se podrán recuperar las palabras y sus concordancias y la frecuencia de aparición de la tabla de unigramas.

II.1.1.7 Acotaciones al Sistema

- El sistema no llenaras las tablas de Bigramas y Trigramas ni las columnas de POS.
- No se realizarán búsquedas por Bigrama o trigrama ni por etiquetado.

II.1.1.8 Diagrama de Contexto



II.1.1.9 Reglas de Negocio

Los corpus se encuentran en formato txt Unicode

II.1.1.10 Eventos Externos

Evento	Descripción	Origen
Corpus	Los textos que conforman el corpus del español de México deben estar debidamente seleccionados y clasificados.	Textos escritos en Nueva España y el México del siglo XIX, trascritos por filólogos o lingüistas.

II.1.1.11 Eventos por Tiempo

Evento	Descripción	Condición de Tiempo
N/A		

II.1.1.12 Entradas

Entrada	Descripción	Origen
Textos del Corpus	Son los textos que forman el corpus del español de México y que serán procesados para separarse en palabras.	Los ingresa el Administrador de la Base de Datos.
Nombre del texto o referencia	Es el nombre del archivo que se va a procesar.	Los ingresa el administrador de la Base de Datos.
Siglo	Siglo al que pertenece la referencia (debe ser entre el XVI y XIX)	Las ingresa el usuario final. Los ingresa el Administrador de la Base de datos.
Palabra consultada	Es la palabra que el usuario quiere analizar en el corpus.	La ingresa el usuario final.

II.1.1.13 Salidas

Salida	Descripción	Destino
Unigramas	Son las palabras que forman un texto o referencia	Tabla de Unigramas de la Base de Datos.
Nombre del texto o Referencia	Es el nombre del archivo que se va a procesar.	Tabla de Referencia de la Base de Datos.
Concordancias	Son las palabras que forman el contexto de la palabra buscada.	Pantalla.
Frecuencia	Es la frecuencia de aparición de una palabra en un siglo.	Pantalla y Tabla de unigramas
Offset	Es la posición de cada palabra en el texto.	Tabla de unigramas

II.1.1.14 Relaciones de Entradas y Salidas

Entrada	Relación (Eventos y/o Requerimientos)	Salida
Textos del Corpus	El Administrador de la base de datos Ingresa los textos para llenar la tabla de unigramas.	Unigramas
Textos del Corpus	Al procesar los textos, se evalúa que la palabra no esté repetida, en caso de que se encuentre registrada en la base de datos, se incrementa la frecuencia en el siglo al que pertenece en la tabla de unigramas.	Frecuencia
Textos del Corpus	Al procesar los textos se obtiene la posición de la palabra y se llena la tabla de unigramas	Offset
Nombre del Texto o referencia	Es el nombre del archivo que se va a procesar, es ingresado por el Administrador de la base de datos y se almacena en la tabla de referencia.	Nombre del texto o Referencia
Siglo	Es el siglo al que pertenece el texto o referencia, es ingresado por el administrador de la base de datos y se almacena en la tabla de referencia	Siglo
Palabra Consultada	El usuario final ingresa la palabra que quiere consultar y el sistema da como resultado la misma palabra y el contexto en base a una ventada	Concordancias

	definida.	

II.1.1.15 Relaciones precedentes

Relación	Precedencia
Nombre del Texto o referencia y Siglo –textos del Corpus	Primero se llena la tabla de referencia con el nombre del texto y siglo.
Textos del Corpus-Palabra consultada	Se debe llenar primero la base de datos para que se puedan realizar consultas.
Texto del Corpus-Concordancia	Para obtener las Concordancias e necesario haber obtenido la posición de las palabras en el texto para recuperar el contexto.
Palabra consultada-Concordancias	Se debe ingresar una palabra para que se muestre una concordancia.

II.1.1.16 Requerimientos de Interfaces Externas

Interfase	Origen	Destino	Descripción
Base de Datos Bibliográfica	Base de datos externa	Base de datos de n-gramas	Por el momento no aplica, aunque se espera que en versiones futuras la tabla de referencia sea sustituida por la base de datos Bibliográfica.

II.1.1.17 Alternativas de Solución Técnica

Continuar con el proyecto con la tecnología actualmente utilizada.

La solución técnica actual consiste en agrupar todos los textos correspondientes a un determinado siglo en un solo archivo. Dicho archivo es procesado por medio de clases de tecnología Java y desglosado en palabras con la finalidad de tener un archivo Índice que acumule todas las palabras y el offset o posición en la que se localiza. Finalmente se obtiene un archivo Índice de los 4 siglos correspondiente al periodo diacrónico.

El usuario ingresa la palabra que desea analizar y selecciona el siglo en el que desea buscar mediante una interfaz en Internet creada con Java Script, el resultado es la palabra subrayada y las concordancias encontradas en dicho archivo, en caso de que se localice alguna. Esto se hace mediante el uso de hashtable, es decir, tablas que relacionan una clave con un valor, donde la clave, en el caso del corpus, es la palabra y el valor es el conjunto de offset o posiciones donde se localizó en el texto.

Las opciones para continuar con está forma de exploración son:

01. Continuar explorando el corpus sin contar con la referencia bibliográfica en la que se localiza la palabra analizada por el investigador o usuario final.
02. El código puede ser modificado, para que cuente la cantidad de offset del archivo índice y obtenga la frecuencia de presentación de cada palabra por siglo.
03. Se puede etiquetar el archivo, mediante algún estándar que no sea XML con la finalidad de que se muestre en pantalla al obtener las concordancias o bien, se deberá modificar el código para que ignore las etiquetas.
04. No se puede generar búsquedas por etiqueta, o por frecuencia.

Los recursos necesarios para esta alternativa son:

05. Herramientas de desarrollo de tecnología JAVA.
06. Servidor de Aplicaciones
07. El esfuerzo en horas hombres para agrupar los archivos del Corpus Histórico del Español de México por siglos.
08. Esfuerzo en horas hombres de programación para obtener la frecuencia.
09. Esfuerzo en horas hombres para etiquetar los archivos.
10. Esfuerzo en horas hombres para agrupar los textos en un solo archivo por siglo.

Riesgos previstos:

El principal riesgo detectado radica en la dificultad que se presenta al crecer el corpus, ya que cuando se agregué un nuevo texto, este tendrá que ser agregado al condensado y procesado nuevamente, lo cual implica la búsqueda por palabra en todo el archivo para recuperar las concordancias y esto puede volverse muy lento. Además del retrabajo que implica.

1. Generar y modificar el código actual con la finalidad de implementar el Corpus Histórico del Español de México mediante el uso de Bases de Datos Relacionales.

    1.1. La solución consiste en modificar el código actual, de tal forma que los archivos sean procesados y desglosados por palabras, y éstas almacenadas en una base de datos, la cual simularía ser un hashtable, de tal forma que las palabras no se almacenen repetidas y los offset se inserten en otra columna.
    1.2. El etiquetado se pude realizar por medio de condiciones en sql, pero esto sería posible a partir de la tabla de bigramas y trigramas para evitar ambigüedades, dado que en dichas tablas se almacenaría el contexto de las mismas.
    1.3. Se pueden hacer búsquedas más complejas como por palabra, etiqueta, siglo, o la combinación de las tres opciones.
    1.4. Se pueden calcular frecuencia por siglo, por archivo y el total de aparición en el periodo diacrónico.
    1.5. La recuperación de la información se realizaría brincando dentro del archivo directamente a la posición de la palabra buscada.

    Los recursos necesarios para está alternativa son:

    1.6. Herramientas de desarrollo de tecnología JAVA.
    1.7. Servidor de Aplicaciones
    1.8. Manejador de Bases de Datos.
    1.9. Esfuerzo en horas hombres de programación.
    1.10. Esfuerzo en horas hombres para generar las reglas lógicas en SQL de etiquetado.

    Riesgos previstos.

    El principal riesgo es el tiempo que puede llevar el generar las sentencias lógicas para etiquetar las palabras, así como contar con los especialistas para revisar y generar dichas reglas.
    Otro riesgo es el tiempo de programación, ya que se tienen que facilitar las herramientas para procesar los archivos.

II.1.1.17.1 Criterios de Selección

- Costo: Es importante y dado que es un proyecto de Investigación, que el costo no se incremente al presupuesto otorgado al Instituto de Ingeniería y que se puedan aprovechar los recursos existentes.
- Capacidad de almacenamiento. Dado que el proyecto del Corpus Histórico del Español de México se espera que crezca continuamente, la capacidad de almacenamiento debe ser capaz de soportar los datos de los textos que lo conforman. De acuerdo a los trabajos de Mark Davis, el volumen del Corpus del Español es de 100 millones de palabras de textos pertenecientes a los siglos 1200 al 1900. Las anotaciones se realizaron en tablas con secuencias de 1, 2, 3 y 4 n-gramas.

- Beneficios Académicos: Dado que éste proyecto de desarrollo, también es un proyecto de Tesis, es importante considerar los beneficios académicos que la solución técnica proporcione, sin dejar de considerar los beneficios antes mencionados.

II.1.1.17.2 Solución Técnica

Dado la importancia de analizar el Corpus del Español de México de una forma más optima y profundizando, es conveniente proporcionar a los investigadores la información necesaria que facilite su trabajo, es por esto que el etiquetado es una parte fundamental en el estudio de las partes del lenguaje y la aplicación, muchas veces empírica, del mismo. La búsqueda de diferentes criterios es también un factor importante, los cuales deben permitir hacer análisis más específicos. Por lo tanto la solución técnica seleccionada es: “Generar y modificar el código actual con la finalidad de implementar el Corpus Histórico del Español de México mediante el uso de Bases de Datos Relacionales”, el cual es factible dado que el Instituto cuenta con las herramientas necesarias de desarrollo como manejadores de bases de datos y dado que se utilizará tecnología Java, también se cuentan con herramientas de desarrollo gratuitas o bien, se puede trabajar con línea de comando. Por otro lado se cuenta con el equipo de lingüistas adecuado que pueden ayudar en la generación y revisión de sentencias lógicas para programar el etiquetado de los textos.

Por lo tanto y dado que no habría ningún aumento en el costo de esta opción dado que ya se cuenta con ellos se desarrollará bajo la siguiente plataforma:

- Lenguaje de Programación: Java
- Manejador de Base de Datos: PostgresSQL
- Servidor de Aplicaciones: Tomcat

II.1.1.18 Requerimientos del Ambiente Productivo

Requerimientos de Hardware

- Dell Servidor Power Edge 2850
- 2 procesadores Xeon a 3.8 GHz/ 2MB cache
- 4 GB de memoria RAM , 4 discos de 300GB

Requerimientos de Software

- Manejador de Base de Datos: PostgresSQL 8.1
- Lenguaje de Programación: Java 1.5
- Servidor de Aplicaciones: Tomcat 5.5.9
- Suse Linux Entreprice Server 9.

Requerimientos de Comunicaciones

- SSH

II.1.1.19 Requerimientos de Documentación

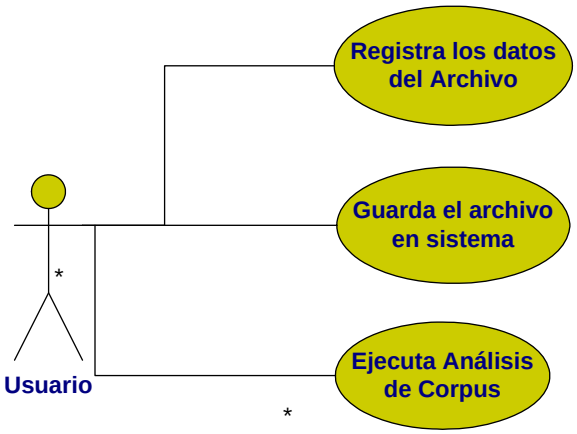
Se requiere la siguiente documentación del sistema:

- **Análisis:** Especificación de Requerimientos y Casos de Uso
- **Diseño:** Diseño arquitectónico, Diagrama de Clases, Diagrama de Base de Datos y Diccionario de datos.
- **Documentación:** Manual de Usuarios

II.1.2 Casos de Uso

Caso de Uso	Guardar Corpus
Descripción	Permite que un usuario pueda registrar los datos de un archivo y almacenarlo dentro del sistema, para después ser procesado.
Complejidad	Baja

II.1.2.1 Caso de Uso Guardar Corpus



Flujo Normal de Eventos	
Flujo Principal	1. Un usuario ingresa el siglo al que pertenece el archivo, los cuales pueden ser XVI, XVII XVIII y XIX. 2. Selecciona el archivo que desea procesar 3. Guarda su archivo en el sistema 4. Ejecuta el análisis del corpus a. El sistema toma el nombre del archivo y el siglo al que pertence y lo guarda en la base de datos. b. Separa el archivo en palabras, y obtiene la posición de cada una dentro del archivo. c. Calcula el número de veces que se presento la palabra en el archivo. d. Calcula el número de veces que se presenta la palabra en el siglo.
Flujos Alternos	1.- En caso de presentar problemas de comunicación se enviará un mensaje indicando que no se puede procesar el archivo.

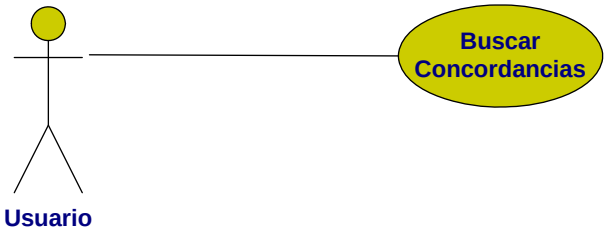
Precondiciones	
Precondición Uno	El corpus debe ser analizado y aprobado como Corpus Histórico del Español en México para ser ingresado a la base de datos
Precondición Dos	El Usuario debe tener permisos para guardar archivos listos para procesar.

Postcondiciones	
Postcondición Uno	NA
Postcondición Dos	NA

Puntos de extensión	
NA	

Caso de Uso	Buscar Concordancias
Descripción	Permite al usuario ingresar el siglo, la palabra y/o corpus en donde desea realizar la búsqueda, para, a partir de la palabra, recuperar las concordancias de un texto.
Complejidad	Alta

II.1.2.2 Caso de Uso Buscar Concordancias



Flujo Normal de Eventos	
Flujo Principal	1.-El usuario ingresa la palabra que desea buscar. 2.-El usuario selecciona: • El siglo en el cual realizara la búsqueda (XVI, XVII, XVIII, XIX). • El documento en el cual realizara la búsqueda. • La venta de caracteres con que se mostrarán las concordancias. 3.-El sistema realiza la búsqueda de los criterios ingresados y obtiene los offsets y la frecuencia con que se presentan en el documento. 4.-El sistema abre el corpus donde se realizará la búsqueda y obtiene las concordancias. 5.-El sistema muestra las concordancias encontradas de la palabra buscada y la frecuencia de aparición.
Flujos Alternos	Si el sistema no encuentra la palabra solicitada se manda un mensaje informativo. Si la ventana no es numérica, el sistema envía un mensaje de error. Si la ventana tiene una longitud menor que la palabra, el sistema manda un mensaje de error. Si el usuario no selecciona un documento, el sistema busca en todos los archivos pertenecientes al siglo seleccionado.

Precondiciones	
Precondición Uno	Previamente se ejecuto el análisis del corpus.
Precondición Dos	NA

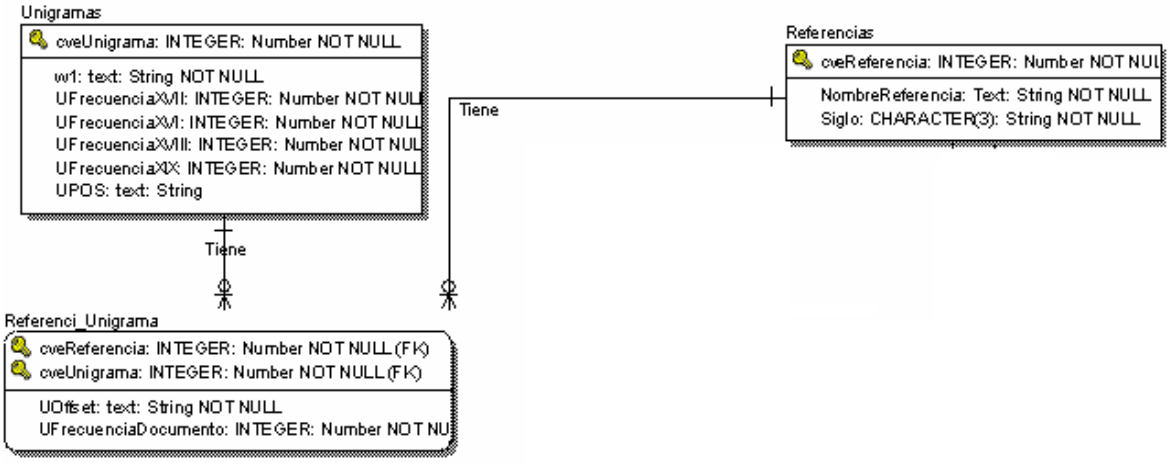
Postcondiciones	
Postcondición Uno	NA
Postcondición Dos	NA

Puntos de extensión	
NA	NA

II.2 Diseño

II.2.1 Modelo de Base de Datos

II.2.1.1 Diagrama Entidad Relación



II.2.1.2 Diccionario de Datos

Nombre de la Entidad	Referencias				
Objetivo de la Entidad	Almacena el siglo y el nombre del Documento de donde se obtendrán los unigramas				
Tipo de Entidad					
Cantidad de Registros Esperados	5000				
Nombre Campo	PK	FK	Tipo de Dato	Ejemplo	Descripción
CveReferencia	X		Integer	1	Clave de la referencia a la que pertenecen los unigramas
NombreReferencia			Text	Cartas de sacerdotes	Nombre de la referencia donde se recuperaran las concordancias
Siglo			CHARACTER	XVI	Siglo al que pertenece la referencia

Nombre de la Entidad	Unigrama				
Objetivo de la Entidad	Almacena los unigramas obtenidos de una referencia y su offset				
Tipo de Entidad					
Cantidad de Registros Esperados	Unigrama				
Nombre Campo	PK	FK	Tipo de Dato	Ejemplo	Descripción
CveUnigrama	X		Integer	1	Clave del Unigrama y campo indexado
W1			Text	Sacerdote	Unigrama
UFrecuenciaXVI			Integer	30	Total de veces en el que se presento el unigrama en el siglo XVI
UFrecuenciaXVII			Integer	10	Total de veces en el que se presento el unigrama en el siglo XVII
UFrecuenciaXVIII			Integer	100	Total de veces en el que se presento el unigrama en el siglo XVIII
UFrecuenciaXIX			Integer	20000	Total de veces en el que se presento el unigrama en el siglo XIX

Nombre de la Entidad	Referenci_Unigrama				
Objetivo de la Entidad	Relaciona la tabla Unigrama y Referencia para recuperar el unigrama del documento especificado.				
Tipo de Entidad	Transitiva				
Cantidad de Registros Esperados					
Nombre Campo	PK	FK	Tipo de Dato	Ejemplo	Descripción
CveReferencia	X	X	Integer	1	Clave del documento al que pertenecen a la referencia. Se obtiene de la tabla Referencias
CveUnigrama	X	X	Integer	1	Clave del unigrama. Se obtiene de la tabla Unigrama
UOffset			Text	1 20 50 10 300 502 203	Posición que ocupa el unigrama dentro de la referencia.
UFrecuenciaDocumento			Integer	40	Total de veces que se presento el unigrama en el documento.

II.2.2 Script de Creación

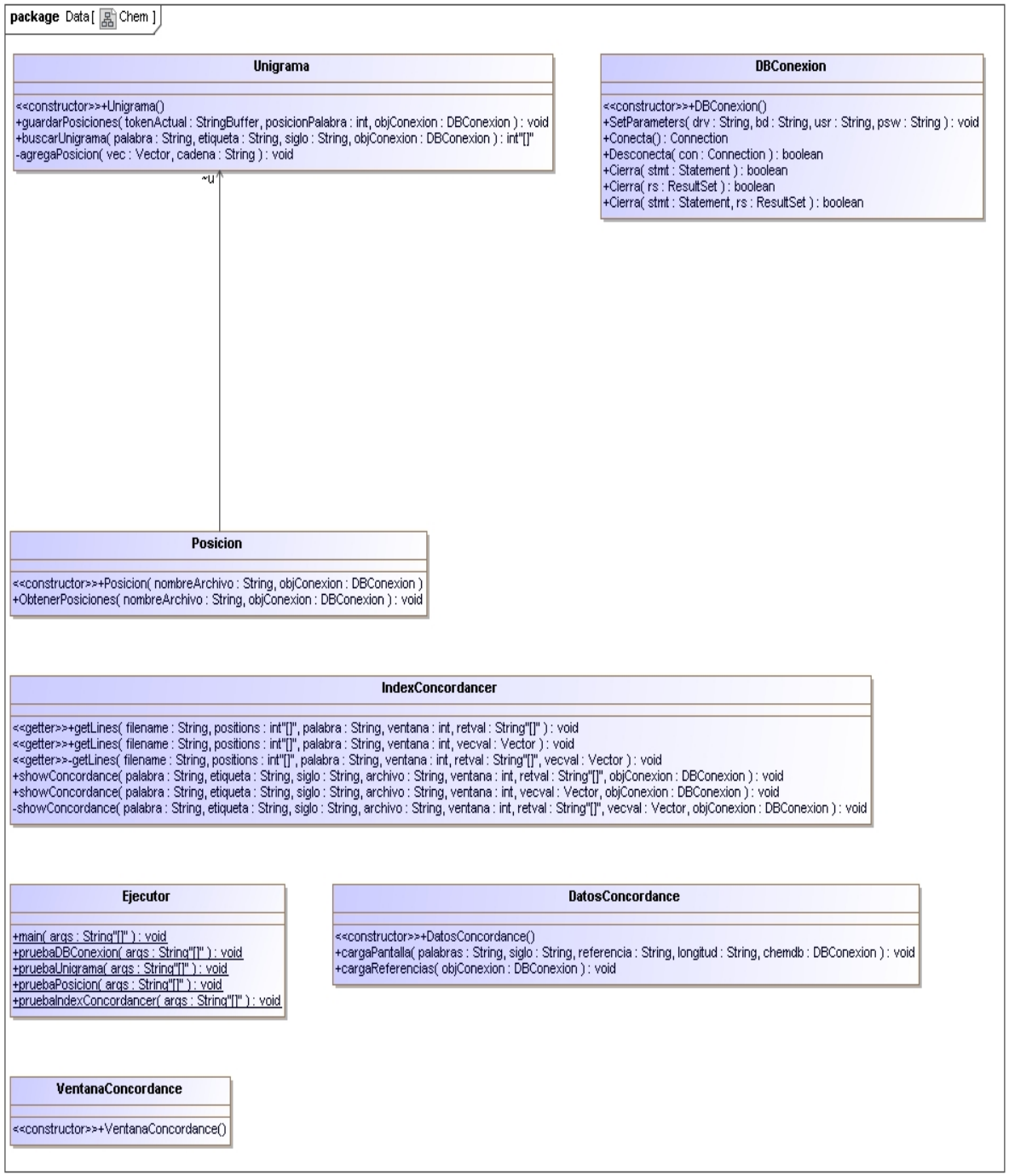
```
--CREATE DATABASE postgres WITH TEMPLATE = template0 ENCODING = 'SQL_ASCII';

--ALTER DATABASE postgres OWNER TO postgres;

--\connect postgres

SET client_encoding = 'SQL_ASCII';
SET check_function_bodies = false;
SET client_min_messages = warning;

CREATE TABLE referenci_unigrama (
    cvereferencia integer NOT NULL,
    uoffset text NOT NULL,
    ufrecuenciadocumento integer,
    cveunigrama integer DEFAULT nextval(('public.sunigrama'::text)::regclass) NOT NULL
);


ALTER TABLE public.referenci_unigrama OWNER TO postgres;

--
-- TOC entry 1200 (class 1259 OID 24689)
-- Dependencies: 5
-- Name: referencias; Type: TABLE; Schema: public; Owner: postgres; Tablespace:
--

CREATE TABLE referencias (
    cvereferencia integer NOT NULL,
    nombrereferencia character varying NOT NULL,
    siglo character varying NOT NULL
);


ALTER TABLE public.referencias OWNER TO postgres;

--
-- TOC entry 1201 (class 1259 OID 24694)
-- Dependencies: 5
-- Name: sunigrama; Type: SEQUENCE; Schema: public; Owner: postgres
--

CREATE SEQUENCE sunigrama
    INCREMENT BY 1
    NO MAXVALUE
    NO MINVALUE
    CACHE 1;


ALTER TABLE public.sunigrama OWNER TO postgres;
```

```
--
-- TOC entry 1574 (class 0 OID 0)
-- Dependencies: 1201
-- Name: sunigrama; Type: SEQUENCE SET; Schema: public; Owner: postgres
--

SELECT pg_catalog.setval('sunigrama', 23998, true);

COMMENT ON SEQUENCE sunigrama IS 'Secuencia de unigrama';

CREATE TABLE unigramas (
    cveunigrama integer NOT NULL,
    w1 text NOT NULL,
    ufrecuenciaxvii integer DEFAULT 0,
    ufrecuenciaxvi integer DEFAULT 0,
    ufrecuenciaxviii integer DEFAULT 0,
    ufrecuenciaxix integer DEFAULT 0,
    upos text,
    ulema character varying,
    utfono character varying
);
ALTER TABLE public.unigramas OWNER TO postgres;

--
-- TOC entry 1577 (class 0 OID 0)
-- Dependencies: 1203
-- Name: COLUMN unigramas.ulema; Type: COMMENT; Schema: public; Owner: postgres
--

COMMENT ON COLUMN unigramas.ulema IS 'Campo que guarda el lema de las palabras';

--
-- TOC entry 1578 (class 0 OID 0)
-- Dependencies: 1203
-- Name: COLUMN unigramas.utfono; Type: COMMENT; Schema: public; Owner: postgres
--

COMMENT ON COLUMN unigramas.utfono IS 'Campo que guarda la transcripción fonológica
';
--
-- TOC entry 1544 (class 2606 OID 24714)
-- Dependencies: 1199 1199 1199
-- Name: referenci_unigrama_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres;
Tablespace:
--

ALTER TABLE ONLY referenci_unigrama
    ADD CONSTRAINT referenci_unigrama_pkey PRIMARY KEY (cvereferencia, cveunigrama);

-
-- TOC entry 1547 (class 2606 OID 24716)
-- Dependencies: 1200 1200
```

```
-- Name: referencias_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres;
Tablespace:
--

ALTER TABLE ONLY referencias
    ADD CONSTRAINT referencias_pkey PRIMARY KEY (cvereferencia);

--
-- TOC entry 1553 (class 2606 OID 24720)
-- Dependencies: 1203 1203
-- Name: unigramas_pkey; Type: CONSTRAINT; Schema: public; Owner: postgres;
Tablespace:
--

ALTER TABLE ONLY unigramas
    ADD CONSTRAINT unigramas_pkey PRIMARY KEY (cveunigrama);

--
-- TOC entry 1545 (class 1259 OID 24722)
-- Dependencies: 1199
-- Name: xif5referenci_unigrama; Type: INDEX; Schema: public; Owner: postgres; Tablespace:
--

CREATE INDEX xif5referenci_unigrama ON referenci_unigrama USING btree (cvereferencia);
CREATE UNIQUE INDEX xpkreferenci_bigramas ON referenci_bigramas USING btree
(cvereferencia, cvebigrama);

CREATE UNIQUE INDEX xpkreferencias ON referencias USING btree (cvereferencia);
--
-- TOC entry 1554 (class 1259 OID 24731)
-- Dependencies: 1203
-- Name: xpkunigramas; Type: INDEX; Schema: public; Owner: postgres; Tablespace:
--

CREATE UNIQUE INDEX xpkunigramas ON unigramas USING btree (cveunigrama);

--
-- TOC entry 1560 (class 2606 OID 24757)
-- Dependencies: 1199 1203 1552
-- Name: referenci_unigrama_cveunigrama_fkey; Type: FK CONSTRAINT; Schema: public;
Owner: postgres
--

ALTER TABLE ONLY referenci_unigrama
    ADD CONSTRAINT referenci_unigrama_cveunigrama_fkey FOREIGN KEY (cveunigrama)
REFERENCES unigramas(cveunigrama);

REVOKE ALL ON SCHEMA public FROM PUBLIC;
REVOKE ALL ON SCHEMA public FROM postgres;
GRANT ALL ON SCHEMA public TO postgres;
GRANT ALL ON SCHEMA public TO PUBLIC;
```

II.2.3 Modelado de Clases

II.2.3.1 Diagrama de Clases



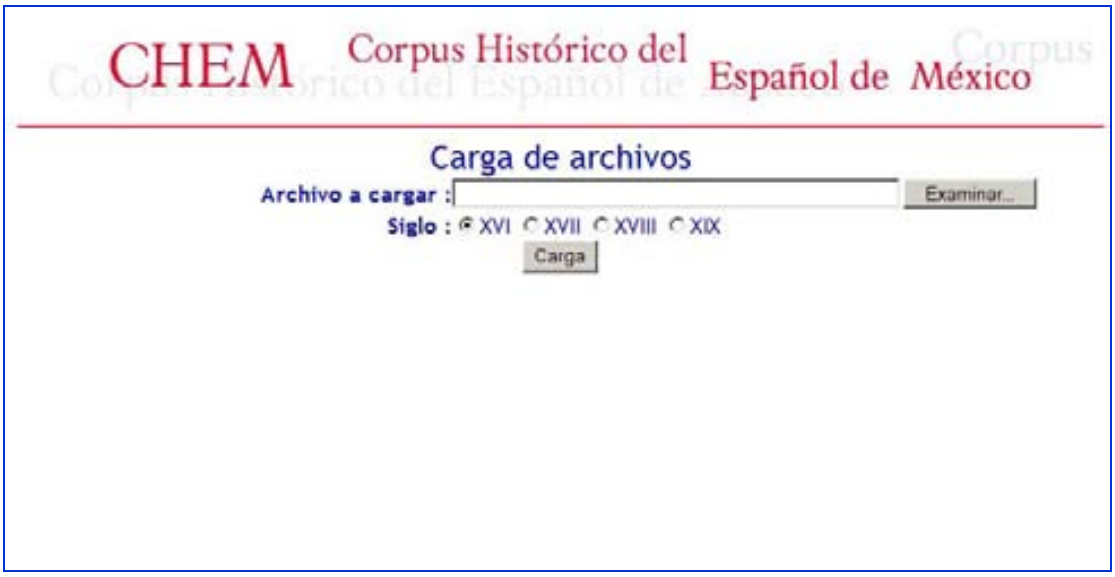
II.2.3.2 Clases por Caso de Uso

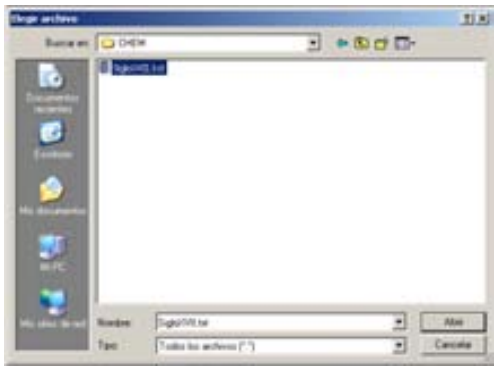
Caso de Uso	Nombre de la clase	Clase de Prueba	Paquete al que pertenece
Guardar Corpus	DBConexion.java	Ejecutor.java	Chem.concordance
Guardar Corpus	Unigrama.java	Ejecutor.java	Chem.concordance
Guardar Corpus	Posicion.java	Ejecutor.java	Chem.concordance
Buscar Concordancias	IndexConcordancer.java	Ejecutor.java	Chem.concordance
Buscar Concordancias	VentanaConcordance.java		Chem.concordance
Buscar Concordancias	DatosConcordance.java		chem.display

II.3 Desarrollo

Como resultado del desarrollo, se muestran las pantallas de la aplicación así como el contenido de la base de datos.

Una de las funcionalidades del CHEM es permitir la carga de archivos a procesar, para esto se ingresa a la pantalla Upload.jsp http://localhost:8080/chem1_0/upload.jsp la cual se muestra a continuación.





Se da clic en el botón examinar para buscar el archivo que se va a procesar, se selecciona el archivo y siglo al pertenece.

Al dar clic en el botón Cargar, confirma si es correcto el siglo seleccionado.

Por último indica el resultado de cargar el archivo y da la opción para regresar a la primera pantalla y cargar otro archivo.

Al finalizar el procesamiento del corpus, tenemos el siguiente resultado en la base de datos.

En la tabla de Referencias se almacena el nombre del archivo y el siglo al que pertenece.

	cvereferencia [PK] int4	nombrereferencia varchar	siglo varchar
1	1	SigloXVI.txt	XVI
2	2	SigloXVII.txt	XVII
3	3	SigloXVIII.txt	XVIII
4	4	SigloXIX.txt	XIX
5	5	Archivo Gramatical de la Lengua Espa	XVII
*			

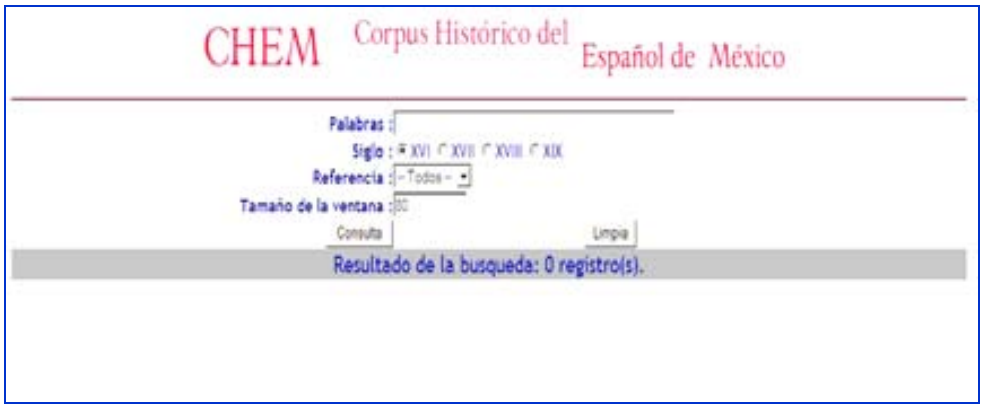
En la tabla referenci_unigrama se almacena el offser o la posición de la palabra en el documento, la frecuencia con que se encontró en el documento y las claves que lo relacionan con las tablas de referencia y frecuencia.

cvereferencia [PK] int4	uoffset text	ufrecuenciadocumento int4	cveunigrama [PK] int4
1	1 225885 298734	3	1
1	8	1	2
1	14 67810 70233 92892 102980 106287 106463 114	66	3
1	21 83 94 156 274 291 363 485 538 571 760 80	4325	4
1	24 67820 70243 92902 102990 103020 104981 106	94	5
1	33 67829 70252 92911 102999 104960 106306 164	61	6
1	41 67837 70260 92919 106314 164630 168364 206	8	7
1	51 67847 70270 92929 106324 164640 168374 206	8	8
1	56 67852 70275 92934 106329 164645 168379 206	8	9
1	61 5437 67797 77500 99155 113025 181754 19707	26	10
1	67 67863 70286 103034 104995 174251 178587 18	44	11
1	73 67869 70292 92959 105001 106373 164748 168	55	12
1	86 67770 96495 112706 114646 116882 117898 28	14	13
1	97 67781 96506 112717 114481 114657 115199 11	11	14
1	106 526 1318 1518 2388 5224 5610 6344 7538 7	356	15
1	109 67903 106412 119149 135456 178670 187058	13	16
1	119 67913 106422 164769 168503 178680 187068	13	17
1	126 67920 106429 164776 168510 178687 187075	20	18
1	129	1	19
1	141 42371 76391 146705 186021 279416	6	20
1	149 18498 131995 134506	4	21
1	159	1	22
1	168 333 383 464 516 710 750 840 1084 1118 11	3627	23
1	170	1	24
1	181 320 344 402 441 505 691 725 734 1247 135	2136	25
1	183	1	26
1	193	1	27
1	203 787 812 1238 1257 2651 2902 3190 3499 36	1665	28

La tabla de unigrama almacena las palabras y la frecuencia total con que se presentan en cada siglo.

cveunigrama [PK] int4	w1 text	ifrecuenciaxvi int4	ifrecuenciaxv int4	frecuenciaxvi int4	ifrecuenciaxi int4	upos text	ulema varchar	utfono varchar
1	1	3	3	8	0			
2	1525	0	1	0	0			
3	ciudad	115	66	66	29			
4	de	5081	4325	4163	1086			
5	MÃ©xico	71	94	55	21			
6	A.G.I	30	61	0	0			
7	Patronato	0	8	0	0			
8	184	0	8	1	0			
9	ramo	0	8	0	0			
10	2	16	26	63	3			
11	Carta	12	44	22	9			
12	autÃ³grafa	47	55	57	18			
13	Rodrigo	8	14	0	0			
14	Albornoz	2	11	0	0			
15	al	351	356	229	97			
16	emperador	0	13	0	0			
17	Carlos	2	13	1	0			
18	V	9	20	4	1			
19	proponiendo	0	1	0	0			
20	mejores	0	6	0	0			
21	formas	1	4	0	0			
22	gobierno	1	1	0	0			
23	y	3234	3627	2405	692			
24	soluciones	0	1	0	0			
25	a	2034	2136	1632	465			
26	distintos	1	1	2	0			
27	problemas	1	1	0	0			
28	en	1703	1665	1550	424			
29	la	2509	1656	2054	518			
30	Nueva	17	98	1	0			
31	EspaÃ±a	36	107	15	1			
32	Sacra	0	18	0	0			
33	Ã§esÃirea	0	3	0	0			
34	catholica	15	15	13	5			
35	majestad	1	422	3	3			
36	Con	32	16	13	3			
37	Lope	0	12	0	0			
38	Samaniego	0	13	0	0			
39	que	3897	4060	3304	918			
40	aquj	0	61	0	0			

La segunda pantalla del CHEM permite recuperar concordancias.



En esta pantalla se escribe la palabra y se selecciona el siglo en el que se va a buscar.

Se puede buscar en un archivo particular o bien en todos los archivos almacenados en ese siglo y se selecciona el tamaño de la ventana, es decir, el número de palabras que se mostraran a la derecha e izquierda de la palabra buscada. Al dar clic en consultar se muestran todas las concordancias encontradas.

CHEM Corpus Histórico del Español de México

Palabras :
Siglo : XVI XVII XVIII XIX
Referencia : -- Todos --
Tamaño de la ventana : 100
Consulta Limpia

Resultado de la busqueda: 17 registro(s).

Archivo Gramatical de la Lengua Española.txt	gua oral) que se fue depositando, ordenadamente,	y	a veces con comentarios, en setenta y cinco cajas
Archivo Gramatical de la Lengua Española.txt	nadamente, y a veces con comentarios, en setenta	y	cinco cajas de cartón. El resultado es una muestr
Archivo Gramatical de la Lengua Española.txt	na a un comentario oído en el autobús refractada	y	ordenada por el criterio de nuestro mayor gramáti
Archivo Gramatical de la Lengua Española.txt	dos miembros del actual equipo editorial (Bosque	y	Millán) concibieron el proyecto de la edición int
Archivo Gramatical de la Lengua Española.txt	n la frase «pasó del camarín a la vecina saleta»	y	clasificarla bajo la categoría «La preposición» d
Archivo Gramatical de la Lengua Española.txt	tra cosa distinta es la explotación del Archivo,	y	ahí sí que los medios informáticos pueden prestar
Archivo Gramatical de la Lengua Española.txt	bras que aquí se contienen, por partes de ellas,	y	con toda la combinatoria que proporciona una base
Archivo Gramatical de la Lengua Española.txt	antener la más absoluta fidelidad al pensamiento	y	a la obra de Salvador Fernández Ramírez, al tiemp
Archivo Gramatical de la Lengua Española.txt	las de los investigadores actuales. Sin embargo,	y	como declaraba tristemente Alexander Cruden en el
Archivo Gramatical de la Lengua Española.txt	obre hombre pecador no puede hacer nada perfecto	y	completo». Al poner estos materiales a disposi
Archivo Gramatical de la Lengua Española.txt	o editorial es que contribuyan a que se complete	y	profundice la descripción gramatical de nuestra l

II.4 Pruebas

II.4.1 Casos de Prueba: Guardar Corpus Procesar

A continuación se muestran los casos de pruebas de integración para verificar el funcionamiento del sistema.

Caso de Prueba	Datos de Entrada	Resultado Esperado	Resultado Real	Notas
Se procesa un archivo	Siglo XVI.txt	Se obtienen todas las palabras del archivo Identifica como palabra completa los números con puntos decimal Respeta las palabras formadas con algún carácter que no sea alfabético Obtiene los offsets de cada palabra Inserta las palabras adecuadamente. Calcula la frecuencia de cada palabra por siglo Calcula la frecuencia de cada palabra por documento	Se proceso el archivo adecuadamente y se obtuvieron todos los offsets y las frecuencias.	Ninguna
Se procesa un archivo	Siglo XVII.txt	Se obtienen todas las palabras del archivo Identifica como palabra completa los números con puntos decimal Respeta las palabras formadas con algún carácter que no sea alfabético Obtiene los offsets de cada palabra Inserta las palabras adecuadamente. Calcula la frecuencia de cada palabra por siglo	Se proceso el archivo adecuadamente y se obtuvieron todos los offsets y las frecuencias.	Ninguna

		Calcula la frecuencia de cada palabra por documento		
Se procesa un archivo	Siglo XVIII.txt	Se obtienen todas las palabras del archivo Identifica como palabra completa los números con puntos decimal Respeta las palabras formadas con algún carácter que no sea alfabético Obtiene los offsets de cada palabra Inserta las palabras adecuadamente. Calcula la frecuencia de cada palabra por siglo Calcula la frecuencia de cada palabra por documento	Se proceso el archivo adecuadamente y se obtuvieron todos los offsets y las frecuencias.	Ninguna
Se procesa un archivo	Siglo XIX	Se obtienen todas las palabras del archivo Identifica como palabra completa los números con puntos decimal Respeta las palabras formadas con algún carácter que no sea alfabético Obtiene los offsets de cada palabra Inserta las palabras adecuadamente. Calcula la frecuencia de cada palabra por siglo Calcula la frecuencia de cada palabra por documento.	Se proceso el archivo adecuadamente y se obtuvieron todos los offsets y las frecuencias	Ninguna
Se procesan archivos	Prueba1.txt	Se obtienen todas las palabras del	Se proceso el archivo	Ninguna

		archivo Identifica como palabra completa los números con puntos decimal Respeta las palabras formadas con algún carácter que no sea alfabético Obtiene los offsets de cada palabra Inserta las palabras adecuadamente. Calcula la frecuencia de cada palabra por siglo Calcula la frecuencia de cada palabra por documento	adecuadamente y se obtuvieron todos los offsets y las frecuencias	
Se procesa un archivo en Inglés	Pruebal.txt	Se obtienen todas las palabras del archivo Identifica como palabra completa los números con puntos decimal Respeta las palabras formadas con algún carácter que no sea alfabético Obtiene los offsets de cada palabra Inserta las palabras adecuadamente. Calcula la frecuencia de cada palabra por siglo Calcula la frecuencia de cada palabra por documento	Se proceso el archivo adecuadamente y se obtuvieron todos los offsets y las frecuencias	Se insertaron las contraciones de las palabras con sus respectivos apostrofes como l´m, don´t.

II.4.2 Casos de Prueba: Buscar Concordancias

Caso de Prueba	Datos de Entrada	Resultado Esperado	Resultado Real	Notas
Se busca la palabra “a” en la referencia SigloXVI.txt y se comparan los resultados con los archivos originales	Siglo:XVI Ventana: 100 Palabra: a Referencia: SigloXVI.txt	Concordancias correctas en las referencias del siglo XVI con la palabra “a”	Se obtuvo el resultado correcto.	La respuesta se vuelve un poco lenta con esta palabra. La verificación se realizó con los 10 primeras, 10 intermedias y 10 últimas concordancias que la búsqueda proporciono.
Se busca la palabra “en”, en la referencia SigloXVII.txt y se comparan los resultados con los archivos originales.	Siglo:XVII Ventana: 150 Palabra: en Referencia: Siglo XVII.txt	Concordancias correctas en las referencias del siglo XVII con la palabra “en”	Correcto	
Se busca la palabra “la” en la referencia SigloXIX.txt y se comparan los resultados con los archivos originales.	Siglo: XIX. Ventana:54 Palabra: la Referencia: SigloXIX,txt	Concordancias correctas en las referencias del siglo XIS con la palabra “la”	Correcto	
Se busca la palabra “está” en la referencia SigloXVIII.txt y se comparan los resultados con los archivos originales,	Siglo: XVIII Ventana:92 Palabra: está Referencia: SigloXVIII	Concordancias correctas en las referencias del siglo XVIII con la palabra “está”	Correcto	
Se busca la palabra “a” en el siglo XVI siglos. Y se comparan los resultados con los archivos originales.	Siglo: XVI Ventana:100 Palabra: a Referencia: Todas	Obtener todas las concordancias con la palabra “a” de todos los archivos del siglo XV	Correcto	
Se buscan diferentes	Ingresar diferentes	Obtener las concordancias	Se obtuvo el resultado	

palabras en los diferentes siglos y se comparan con los archivos originales.	datos en las opciones de entrada.	correctas de las diferentes búsquedas.	correcto.	
Se busca la palabra dixo con una ventana vacía	Siglo: XVIII Ventana:92 Palabra: está Referencia: SigloXVIII	Mensaje de error indicando que no se puede realizar una búsqueda con una ventana vacía.	Se obtuvo el resultado correcto.	
Se busca la palabra yndio con una ventana de 3	Siglo: XVI Ventana:3 Palabra: yndio Referencia: SigloXVIII	Mensaje de error indicando que la ventana no puede ser menor a la longitud de la palabra	Se obtuvo el resultado correcto.	
Se realiza una búsqueda sin palabra	Siglo: XVIII Ventana:30 Palabra: Referencia: Todas	Mensaje de error indicando que se debe ingresar una palabra	Se obtuvo el resultado correcto.	

III. Conclusiones

Como resultado de este trabajo podemos concluir que es factible el procesamiento de corpus mediante el uso de base de datos relacionales a nivel de almacenamiento de n-gramas y su posición en un texto(offset), y la búsqueda y recuperación de sus concordancias, resaltando que para esto es importante el uso de la indexación de columnas. considerando que aunque el diseño de la base de datos es sencillo, no así el tamaño de la misma, llegando a generar más de 23998 registros con tan solo 4 documento, la recuperación de registros es rápida, no obstante, se muestra lentitud al buscar palabras tan comunes como la conjunción “Y” con la que se recuperaron 4465 registros, del mismo número de documentos.

El alcance de este proyecto sólo incluye la recuperación de concordancias de unigramas en un texto, es decir, la búsqueda de una sola palabra, quedando por hacer la obtención de bigramas (dos palabras) y trigramas (tres palabras) pero sobre todo y lo más importante ya que implica conocer un la estructura del lenguaje, el etiquetado del corpus, para lo cual también existen diversas propuestas y quizá algún proyecto futuro inicie con la hipótesis a favor de otra tecnología.

Cabe incluir en estas conclusiones que además del resultado práctico y profesional, la lectura de algunos de estos textos que en su mayoría son demandas hacía el grupo clérigo de la época, me dejó un poco de conocimiento acerca de la evolución de un lenguaje que poco conocemos.

IV. Anexo

Este anexo contiene el análisis de los ciclos de vida con el que se desarrollaría el Corpus.

Ciclo de vida	Actividades	Ventajas	Desventajas
En Cascada	1. **Análisis de requerimientos:** Se analizan las necesidades de los usuarios finales del software para determinar qué objetivos debe cubrir. De esta fase surge una memoria llamada SRD (Documento de Especificación de Requisitos), que contiene la especificación completa de lo que debe hacer el sistema sin entrar en detalles internos. 2. **Diseño del Sistema:** Se descompone y organiza el sistema en elementos que puedan elaborarse por separado, aprovechando las ventajas del desarrollo en equipo. Como resultado surge el SDD (Documento de Diseño del Software), que contiene la descripción de la estructura global del sistema y la especificación de lo que debe hacer cada una de sus partes, así como la manera en que se combinan unas con otras. 3. **Diseño del Programa:** Es la fase en donde se realizan los algoritmos necesarios para el cumplimiento de los requerimientos del usuario así como también los análisis necesarios para saber que herramientas usar en la etapa de	• Es ideal si las especificaciones del sistema son completas y sin ambigüedades. • Si las especificaciones del sistema no son claras o son informales, el Modelo del ciclo de vida también puede ser usado. Sin embargo, la estimación inicial y los compromisos pueden ser usados sólo para la etapa de especificaciones. El plan detallado puede ser comprometido sólo después de que las especificaciones estén terminadas. En caso de que se inicie un plan de proyecto tiene que incluir las fechas de inicio y fin esperadas que se le darán al cliente. En la implementación de protocolos estándar de comunicación, esté modelo puede ser aplicado exitosamente.	• Este modelo no tiene un buen manejo de riesgos dado que es muy costoso y es difícil volver a revisar una etapa anterior. • No hay un producto ejecutable hasta el final del ciclo para que el cliente lo utilice.

	4. **Codificación.** Es la fase de programación propiamente dicha. Aquí se desarrolla el código fuente, haciendo uso de prototipos así como pruebas y ensayos para corregir errores. Dependiendo del lenguaje de programación y su versión se crean las librerías y componentes reutilizables dentro del mismo proyecto para hacer que la programación sea un proceso mucho más rápido. 5. **Pruebas:** Los elementos, ya programados, se ensamblan para componer el sistema y se comprueba que funciona correctamente antes de ser puesto en explotación. 6. **Implantación:** El software obtenido se pone en producción. 7. **Mantenimiento:** Durante la explotación del sistema software pueden surgir cambios, bien para corregir errores o bien para introducir mejoras. Todo ello se recoge en los Documentos de Cambios.	• Aplicación y administración sencilla. Los criterios de entrada y salida para la etapa de desarrollo pueden ser claramente definidos junto con las métricas para el control efectivo del proyecto. La administración de la configuración también llega a ser fácil porque sólo hay una versión de la línea base de la documentación y código en algún punto en el desarrollo del proceso. • Énfasis en la Verificación y Validación a través del ciclo de vida; al finalizar cada etapa, puede ser verificada para evitar errores al final del proyecto.	
En Espiral	Las actividades de este modelo son un espiral, cada bucle es una actividad y estas no se fijan inicialmente, sino que las siguientes se eligen en función de un análisis de riesgo. **1. Determinar o Fijar objetivos:** • Fijar también los productos definidos a obtener: requerimientos, especificación, manual de usuario.	1. El análisis del riesgo se hace de forma explícita y clara. Une los mejores elementos de los restantes modelos.	Genera mucho trabajo adicional. Cuando un sistema falla se pierde tiempo y coste dentro de la empresa Exige una cierta habilidad en los analistas (es bastante difícil).

	• Fijar las restricciones. • Identificación de riesgos del proyecto y estrategias alternativas para evitarlos. • Hay una cosa que solo se hace una vez: planificación inicial o previa. **2. Análisis de Riesgos:** • Se estudian todos los riesgos potenciales y se seleccionan una o varias alternativas propuestas para reducir o eliminar los riesgos. **3. Desarrollar, Verificar y Validar:** • Tareas de la actividad propia y se prueba. • Análisis de alternativas e identificación resolución de riesgos. **4. Planificar** • Revisamos todo lo hecho, evaluándolo, y con ello decidimos si continuamos con las fases siguientes y planificamos la próxima actividad.		
De Prototipo	1. Inicia con la definición de los objetivos globales para el software. 2. Se Identifican los requisitos conocidos y las áreas del esquema en donde es necesaria más definición. 3. Se plantea con rapidez una iteración de construcción de prototipos y se presenta el modelado o un diseño rápido.	1. Este modelo es útil cuando el cliente conoce los objetivos generales para el software, pero no identifica los requisitos detallados de entrada, procesamiento o salida. 2. También ofrece un mejor enfoque	1. El cliente ve funcionando lo que para el es la primera versión del prototipo que ha sido construido con “plastilina y alambres”, y puede desilusionarse al decirle que el sistema aun no ha sido

		cuando el responsable del desarrollo del software está inseguro de la eficacia de un algoritmo, de la adaptabilidad de un sistema operativo o de la forma que debería tomar la interacción humano-máquina.	construido. 2. El desarrollador puede caer en la tentación de ampliar el prototipo para construir el sistema final sin tener en cuenta los compromisos de calidad y de mantenimiento que tiene con el cliente.

V. Glosario

Término	Definición
Corpus	Los corpus lingüísticos son una recopilación de textos hablados y escritos con la finalidad de realizar cierto análisis lingüístico, como evolución del lenguaje, uso de las palabras, uso de las reglas sintácticas
Grama	Cada una de las palabras que forman un Corpus.
Lema	Forma canónica o estándar de un conjunto de palabras. En un diccionario o enciclopedia, cada una de las palabras o términos que se definen o traducen.
Partes del lenguaje (POS Part of speach)	Son los diferentes componentes que forman una frase u oración, por ejemplo: artículo, sustantivo, verbo.
Offset	Es el número de la posición de una palabra en un texto.
Concordancias	Es un recurso gramatical de las lenguas para marcar las relaciones gramaticales entre los diversos constituyentes mediante referencias cruzadas. Se lleva a cabo requiriendo que la palabra que ocupa una determinada posición sintáctica tome una u otra forma según algún rasgo determinado por otra palabra con la que "concuerda" en ese rasgo o accidente gramatical.
Frecuencia de onda	Se define por frecuencia, al número de ciclos de onda que ocurren en una unidad de tiempo
Intensidad de onda	La intensidad de un sonido depende de la mayor o menor amplitud de la onda, ya que la energía de la misma es función del cuadrado de la amplitud. La audición está unida a la intensidad de la onda sonora. Para cada frecuencia hay una intensidad mínima (umbral de audición) por debajo de la cual no se oye; y una intensidad máxima que produce sensación de dolor (umbral doloroso). El nivel de intensidad sonoro, se mide en decibelios (dB):

VI. Bibliogafía

Beaulier Alan, 2006, Aprende SQL, España, Anaya Multimedia.

Bobadilla Sancho Jesús, Sancho Hernández Adela, 2003, Comunicaciones y bases de datos con JAVA a través de ejemplos, México, Alfaomega Grupo Editor.

Castaño Adoración de Miguel, 1999, Fundamentos y Modelos de Bases de Datos, México, Alfaomega Grupo Editor.

Ceballos Sierra Francisco Javier, 2000, Java 2 Lenguaje y aplicaciones, México, Alfaomega Grupo Editor.

Cucezan Silviu and Yarowsky David, 2002, Bootstrapping a Multilingual Part of speech Tagger in One Person-day, Taiwan.

Date C.J, 1995, Introducción a los Sistemas de Bases de Datos. México. Addison Wesley Iberoamericana, SA de CV.

David Mark, 2003, Advances Research on syntactic and semantic change with the Corpus del Español, Illinois.

David Mark, 2003, Relational n-gram databases as a basis for unlimited annotation on large corpora, Illinois.

Davis Mark, 2003, Annotation without lexicons: an alternative to the standard bootstrapping approach, Illiniois.

Fagin, R, 1977, Multivalued Dependencies and a New Normal Form for Relational Databases, *ACM TODS* 2.

García Rincon Luis Francisco, 2007, Bases de Datos: Un Enfoque Práctico, México, Trillas.

Hjellmslev Lous, 1980, Sistema Lingüístico y Cambio Lingüístico, Gredos, Madrid.

Hockett Ch. F, 1974, El Estado Actual de la Lingüística. Akal Editor, Madrid.

Ken Arnold, 2003, El Lenguaje de Programación Java, España, Pearson Educación de México.

Leech G., 1993, Corpus Annotation Schemes. Literary and Linguistic Computing

Llisterri, J. & J. M. Garrido, 1998. Informe sobre los recursos lingüísticos para el español II. Corpus orales y escritos disponibles y en desarrollo en España. Documento disponible en la red: http://www.cervantes.es/internet/acad./oeil.

López Morales H, 1974, Introducción a la lingüística generativa. Romania, Madrid.

Luque Ruiz Irene, Gómez Miguel Angel, Gomez Espinosa Enrique, Corruela García Gonzalo, 2001, Bases de Datos: Desde Chen hata Cood con Oracle, México, RA-MA.

Martinet Andre, 1975, La Lingüística Guía Alfabética, Anagrama, Barcelona, 2ª Edición. Traducción Carlos Manzano

Mason Oliver, 2000, Programming for Corpus Linguistics How to Do Text Analysys with Java, Gran Bretaña, Endinburgh University Press.

Medina Urrea Alfonso, 2005, Protocolo CHEM.

Mukhar Kevin, 2001, Bases de Datos con Java, España, Anaya.

Pérez Cesar, 2007, Oracle 10 g: Administracion y Analisis de Bases de Datos, México, Alfaomega Grupo Editor.

Pérez López, Cesar, 2008, Oracle PL/SQL, Madrid, RA-MA.

Pfleeger, Shari Lawrence, 2002, Ingeniería del Software, Argentina, Prentice Hall Argentina.

Piattini Velthuis Marío Gerardo, 2007, Tecnología y Diseño de Bases de Datos, México, Alfaomega Grupo Editor.

Rivero Cornelio Enrique, 2002: Introducción al SQL para Usuarios y Programadores: a Nivel de IBM, DB2, UDB, Version 7.2 o Superior, España, Thomson Learning.

Sagan Carl, 1981, Los Dragones del Eden, México, Grijalbo.

Schach Stephen, 2006, Ingeniería del Software Orientado a Objetos, México, Mcgraw-Hill / Interamericana de México.

The PostgreSQL Global Development Group, 2005 PostgreSQL 8.1.3 Documentation

The PostgreSQL Global Development Group, 2008 PostgreSQL 8.3.5 Documentaition.

WB,01 http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=lengua. Diccionario de la lengua española- Vigésima Segunda Edición.

WB,02 http://www.revista.unam.mx/vol.2/num1/art1/. Problemas Actuales de la Lingüística Computacional, Grigory Sidorov, 31 de Marzo 2001, Vol. 2 No 1.

WB,03 http://es.wikipedia.org/wiki/C%C3%ADrculo_de_Praga. Escuela de Praga.

WB,04 http://es.wikipedia.org/wiki/Fonolog%C3%ADa. Fonología.

WB,05 http://es.wikipedia.org/wiki/Fon%C3%A9tica. Fonética.

WB,06 http://es.wikipedia.org/wiki/Base_de_datos. Wikipedia La enciclopedia Libre. Base de datos

WB,07 http://elies.rediris.es/elies17/ Annette Becker Estudios de Lingüística Española (ELiE´s) Volumen 18, 2002,

WB,08 http://iling.torreingenieria.unam.mx/CursoCorpus2005 Curso de Corpus, Instituto de Ingeniería.

WB,09 http://elies.rediris.es/elies18/index.html. Estudios de Lingüística Española (ELiE´s) Volumen 18, 2002,

WB,10 http://elies.rediris.es/elies9/4-2-3.htm Antonio Moreno Ortiz, 01/Enero/2006 “El enfoque relacional”.

WB,11 http://www.lania.mx/~ebenitez/CursoBD07.pdf Edgar I. Benítez Guerrero, Fundamentos de Bases de Datos.

WB,12 http://www.postgresonline.com/journal/index.php?/archives/10-How-does-CLUSTER-ON-improve-index-performance.html, Leo Hsu y Regina Obe, How does CLUSTER ON improve index performance

WB, 13 http://es.wikipedia.org/wiki/Sistema_administrador_de_bases_de_datos_relacionales Sistema Administrador de Bases de Datos Relacionales.

WB,14 http://www.atpsoftware.net/Docs/12ReglasCodd.htm las 12 reglas de codd que determinan la fidelidad de un sistema relacional al modelo relacional

WB,15 http://www.monografias.com/trabajos5/norbad/norbad2.shtml#, Manuel Torres Remon, Normalización de base de datos.

WB, 16 http://www.geocities.com/v.iniestra/apuntes/est_bd/ Victor Manual Iniestra Alvarez Estructuras y Bases de Datos 08/Febrero/2006

WB, 17 http://es.tldp.org/Postgresql-es/web/navegable/tutorial/sql.html, El equipo de desarrollo de PostgreSQL, Tutorial de PostgreSQL.

WB, 18 http://www.htmlpoint.com/sql/sql_03.htm, Lucio Benfante, Tutorial de PostgreSQL

WB, 19 http://www.monografias.com/trabajos13/trsqlinf/trsqlinf.shtml#, Manuel Torres Remon, Normalización de Bases de Datos.