35

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

Facultad de Ingeniería

Sistema de consulta vía Internet a la base de datos de vientos y contaminantes de la ciudad de México

T E S I S

que para obtener el título de

INGENIERO EN COMPUTACION

presenta

FRANCISCO LORENZANA ZARCO

Director: Ing. German Santos Jaimes

México, D. F., 2001

Este trabajo esta dedicado a mi familia y a todas las personas que me han ayudado en mi formación personal, académica y profesional.

Agradecimientos

Ing. German Santos

Director de tesis

Ing. Sergio Noble

Gracias por todos los conocimientos y enseñanzas transmitidas, y por su colaboración en la realización de esta tesis.

Ing. Daniel Sol

Por la coordinación en la realización de este proyecto.

Al jurado compuesto por:

Ing. Laura Sandoval,

Ing. Sergio Noble,

Ing. Luis Alberto Aguilar,

Ing. Santiago Igor Valiente

gracias por sus comentarios, correcciones y sugerencias en el desarrollo de ésta tesis.

A mis amigos y compañeros de la Coordinación de Prospección e Innovación(CPI) de la DGSCA.

Índice

Capítulo 4 Apéndices

Introducción

En el presente documento se describirán los pasos que se siguieron para realizar el sistema de consulta a la base de datos de vientos y contaminantes de la ciudad de México a través de internet, los problemas que se presentaron en la realización y los logros que se obtuvieron de él.

Se iniciará con una breve introducción referente al problema de la contaminación en México y la importancia que tiene el disponer de información específica de la cantidad de contaminantes en la ciudad de México, ya que el conocimiento del comportamiento de dichos contaminantes es de gran ayuda para poder hacer predicciones de las variaciones que puedan tener en un periodo de tiempo determinado. Por lo tanto surge la inquietud de tener un sistema que sea capaz de brindar información detallada de los vientos y contaminantes de la ciudad de México a la población en general, pero principalmente a un conjunto de investigadores que serán los encargados de realizar la investigación pertinente.

En las páginas siguientes se menciona el problema de la contaminación, el Ciclo de Vida de los Sistemas de Información, el diseño de la base de datos y se describirá la tecnología cliente/servidor ya que en este modelo se basa la realización de este sistema; también se mencionará la importancia que tiene el uso de software de código abierto (open source) y las características del mismo, esto con la finalidad de mostrar al lector las herramientas utilizadas en la realización de este sistema.

Una vez hecha la descripción del problema y mencionar la tecnología a utilizar se analiza a detalle el Ciclo de Vida de los Sistemas de Información (C.V.S.I.) y las etapas que lo conforman aplicándolo a este sistema.

Después de documentado el CVSI, se agrega un apartado dedicado a las conclusiones personales y por último los apéndices que servirán de consulta para entender a detalle el sistema de consulta a la base de datos de vientos y contaminantes de la Cd. de México.

Capítulo

Antecedentes

En el primer tema de este capítulo, se menciona el problema de la contaminación en la Cd. de México como primer aproximación para entender el motivo de la realización de este documento y del sistema mismo. En los siguientes temas se menciona brevemente lo siguiente: el ciclo de vida de los sistemas de información, sugerencias para diseñar bases de datos, simbología de las metodologías más utilizadas en los diagramas entidad relación, la forma de obtener información de bases de datos por Internet, la tecnología Cliente – Servidor y los principales tipos de software que existen actualmente. Este primer capítulo pretende brindar al lector las herramientas necesarias para entender los capítulos posteriores.

Antecedentes

En los siguientes apartados se mencionará una breve descripción de los conceptos teóricos en que se basa este sistema, los cuales son necesarios para poder ubicar el problema que se tiene y la forma de resolverlo. Además se pretende que estos apartados sirvan como material de consulta para las personas que lean este documento, ya que se mencionan brevemente la metodología y las herramientas utilizadas.

El problema de la contaminación.

La contaminación del aire es uno de los problemas ambientales más importantes, y es resultado de las actividades del hombre. Las causas que originan esta contaminación son diversas, pero el mayor índice es provocado por las actividades industriales, comerciales, domésticas y agropecuarias.

La combustión empleada para obtener calor, generar energía eléctrica o movimiento, es el proceso de emisión de contaminantes más significativo. Existen otras actividades, tales como la fundición y la producción de sustancias químicas, que pueden provocar el deterioro de la calidad del aire si se realizan sin control alguno.

El aire puro es una mezcla gaseosa compuesta por un 78% de nitrógeno, un 21% de oxígeno y un 1% de diferentes compuestos tales como el argón, el dióxido de carbono y el ozono. Entendemos pues por contaminación atmosférica cualquier cambio en el equilibrio de estos componentes, lo cual altera las propiedades físicas y químicas del aire.

Los principales contaminantes del aire se clasifican en:

PRIMARIOS :

Son los que permanecen en la atmósfera tal y como fueron emitidos por la fuente. Para fines de evaluación de la calidad del aire se consideran: óxidos de azufre, monóxido de carbono, óxido de nitrógeno, hidrocarburos y partículas.

SECUNDARIOS :

Son los que han estado sujetos a cambios químicos, o bien, son el producto de la reacción de dos o más contaminantes primarios en la atmósfera. Entre ellos destacan los oxidantes fotoquímicos y algunos radicales de corta existencia como el ozono.

A nivel nacional, la contaminación atmosférica se limita a las zonas de alta densidad demográfica o industrial. Las emisiones anuales de contaminantes en el país son superiores a 16 millones de toneladas, de las cuales el 65 % es de origen vehicular.

En la Ciudad de México se genera 23.6% de dichas emisiones, en Guadalajara el 3.5%, y en Monterrey el 3%. Los otros centros industriales del país generan el 70% restante.

Ciclo de vida de los sistemas de información.

En las actividades cotidianas de las empresas se realizan un gran número de operaciones rutinarias que son controladas de manera automatizada con sistemas de información.

Los *sistemas de información* (SI) están conformados por todos los recursos materiales y humanos que participan en la recolección, administración, uso y distribución de la información en una empresa. Estos recursos incluyen los datos que son manejados por la empresa, las aplicaciones, los medios de almacenamiento, el equipo de cómputo, el personal que usa y mantiene el sistema, los programadores de aplicaciones, los medios de comunicación, etc.

Los SI se dividen principalmente en tres categorías: sistemas para el procesamiento de transacciones (TPS), sistemas de información administrativa (MIS) y sistemas para el soporte de decisiones (DSS).

Los TPS se caracterizan por brindar al usuario una gran velocidad y precisión en las transacciones, un claro ejemplo de este tipo de sistemas son los cajeros automáticos; los MIS presentan reportes periódicos del comportamiento de

la empresa, lo cual permite a los administradores, gerentes o dueños de la misma, ver el comportamiento que tiene en un periodo de tiempo determinado, estos reportes tienen un formato establecido; los DSS son una extensión de los MIS pero son más flexibles ya que es posible cambiar el formato de los reportes y contienen módulos en los que es posible generar simulaciones del comportamiento de la empresa.

Para poder desarrollar un SI existen diversos métodos con ventajas y desventajas, pero todos tienen características en común. Estas características, que básicamente son las etapas por las que atraviesa el desarrollo de un SI son: Análisis, Diseño, Desarrollo, Instalación y Mantenimiento.

El método del *Ciclo de Vida de los Sistemas de Información* (**CVSI**) es un conjunto de actividades que realizan los diseñadores, analistas, administradores y programadores en la creación de un sistema de información. Las etapas por las que atraviesa el desarrollo de los SI pueden variar dependiendo el tipo de organización o empresa, ya que existen grandes diferencias entre una organización pública y una privada.

Análisis	Estudio preliminar	Primer acercamiento (reuniones o entrevistas) entre el analista y el usuario del sistema, en donde se pretende obtener una visión general del SI. Se establecen requerimientos.
	Planeación	Se determinan las actividades a realizar, personas participantes, asignación de tareas, tiempo estimado, recursos con los que se cuenta (Hardware y Software). Sirve para determinar la duración y el costo de un proyecto
	Propuesta	Documento formal en el que se especifican costos, tiempos estimados, formas de pago, garantías, cláusulas del servicio, etc.
	Análisis detallado	Mediante el uso de herramientas de Ingeniería de Sofware Asistida por Computadora (**CASE**), se muestran las características en detalle de todos los procesos que habrán de realizarse, los cuales se esquematizan mediante Diagramas de Flujo de Datos (**DFD**) de la metodología Gane & Sarson.

Diseño	Modelado de datos	Después de hacer un análisis de los datos manejados, se crean entidades (tablas) con sus atributos (campos). El objetivo del modelado de datos es crear un diseño conceptual de la base de datos, es decir se hace un diseño que sea independiente del Sistema Manejador de Bases de Datos Relacionales (**RDBMS**) a utilizar, generalmente se utiliza un modelado de datos de alto nivel como el Entidad-Relación.
	Normalización	Es el proceso en el que se analiza cada entidad, los atributos que la conforman y las relaciones que existen entre ellas. La normalización se hace utilizando las formas normales, procurando evitar redundancia de los datos y sobretodo el ahorro de espacio en disco. Se establecen llaves candidatas, primarias y foráneas.
	Diagrama estructurado	Es un apoyo basado en diagramas en los que se muestran los módulos a programar. Sirve como material de apoyo al programador ya que muestra la conexión entre los módulos, las variables y los datos. Este diagrama es independiente del lenguaje utilizado para desarrollar los programas. Es un modelo lógico de los programas a realizar y se conocen como cartas de estructura.
	Prototipo	Es la simulación de la ejecución de un SI. Sirve para mostrar al usuario la forma en que funciona el SI. Es de gran ayuda pues permite atender a las sugerencias del usuario final referentes a cuestiones de diseño, presentación, funcionamiento y facilidad o dificultad en cuanto al uso del SI. El prototipo esta sujeto a correcciones hechas por el programador o por el usuario final.
Desarrollo	Programación de funciones	Es la creación de los programas utilizando un lenguaje de programación específico. Es la implementación física de las cartas de estructura.
	Pruebas	A cada módulo se le realizan pruebas individuales y pruebas de acoplamiento con los otros módulos para ver que cumplan con el objetivo fijado.
	Liberación del SI	Si todos los módulos funcionan bien en conjunto y el SI cumple con el objetivo el sistema se libera. Generalmente se dice que se libera el SI cuando cumple con los requerimientos en el equipo de pruebas.

Instalación	Instalación	El SI se instala en el equipo de cómputo en el que estará funcionando permanentemente. Se copian todos los programas y las bases de datos.
	Puesta a punto	Es la configuración del SI y del Hardware para que funcione correctamente.
	Capacitación	Se crea el manual para el usuario y para el administrador. Se hacen sesiones con los usuarios para atender sus dudas y capacitarlos en el uso del SI con la ayuda de los manuales.
Mantenimiento	Preventivo	Se refiere al realizar revisiones periódicas al SI y verificar la pureza de la información mediante cifras de control y reportes.
	Correctivo	Se realiza cuando se detecta alguna falla en alguno de los programas, por lo que se tendrá que rehacer completamente o corregir el punto en el que se presente el error.
	Adaptativo	Costa de la creación de nuevos módulos o modificación de los ya existentes para que realicen de manera diferente alguna tarea.

Cuadro 1. Etapas del CVSI

En todas las etapas mencionadas en el cuadro 1 se debe hacer la documentación pertinente. Ya que servirán de apoyo a las personas que estén interesadas (administradores y usuarios) en el funcionamiento del SI. También es de gran ayuda para cuestiones legales, ya que existen un gran número de documentos en los que se firman acuerdos entre los participantes del proyecto.

En el capítulo siguiente se podrá ver la aplicación de estas etapas en el sistema de consulta a la base de datos de vientos y contaminantes de la Cd. de México.

Diseño de bases de datos

El éxito de cualquier SI depende en gran medida del diseño de la base de datos. Ya que al diseñar una base de datos se espera satisfacer los requerimientos de *contenido* de información de los usuarios, desarrollar aplicaciones, proveer una *estructuración* de la información, atender cuestiones de *rendimiento*, como el tiempo de respuesta, el tiempo de procesamiento y el espacio de almacenamiento. Estos puntos son difíciles de lograr debido a que el

diseño de las bases de datos a menudo comienza con requerimientos muy informales y muy mal definidos. Por tal motivo es de gran importancia identificar seis fases principales en el proceso de diseño de bases de datos.

1. **Recolección y análisis de requerimientos**
2. **Diseño conceptual de la base de datos**
3. **Elección de un Sistema Manejador de Bases de Datos (DBMS)**
4. **Diseño lógico de la base de datos**
5. **Diseño físico de la base de datos**
6. **Implementación del sistema de base de datos**

La *fase 1* implica recabar información sobre el uso que se pretende dar a la base de datos. En esta fase se tienen que realizar las siguientes actividades: identificar las principales áreas de aplicación y grupos de usuarios de la base de datos; estudiar y analizar la documentación existente relativa a las aplicaciones; estudiar el entorno de operación actual y de los planes de aprovechamiento de la información y recolectar respuestas a preguntas realizadas a los posibles usuarios.

En la *fase 2* se realiza un esquema conceptual de la base de datos que sea independiente del DBMS y que sea entendible por los usuarios no expertos. Este modelo generalmente se hace atendiendo a un modelo de datos de alto nivel como el de Entidad-Relación (**E/R**) en donde exista un entendimiento completo de la estructura, el significado (semántica), los vínculos y las restricciones de la base de datos.

El modelado de datos conceptual requiere que tenga las siguientes características:

1. Expresividad: El modelado de datos debe ser lo bastante expresivo para distinguir los diferentes tipos de datos, vínculos y restricciones.
2. Sencillez: El modelo debe ser lo bastante simple para que la mayoría de los usuarios no expertos comprendan y usen los conceptos.

3. Representación diagramática. El modelo debe contar con una representación diagramática para mostrar un esquema conceptual que sea fácil de interpretar.
4. Formalidad: El modelo debe representar una especificación formal, sin ambigüedad de los datos. Por lo que los conceptos del modelo deben definirse con exactitud para evitar confusiones.

En esta fase también se diseñan las características de las transacciones conocidas a la base de datos con independencia del DBMS, a esto se le conoce como diseño de transacciones.

La *fase 3* se refiere a la elección de un DBMS el cual depende de varios factores, algunos de ellos técnicos, otros económicos y otros mas relativos a políticas de la empresa. Al escoger un DBMS se requiere considerar: el costo de la adquisición del software, el costo de mantenimiento, costo de adquisición de hardware, costo de creación y conversión de la base de datos, costo de personal, costo de capacitación, entre otros.

En la *fase 4* también denominada diseño lógico de la base de datos se hace una transformación del esquema conceptual de la *fase 2*, a un modelado de datos del DBMS elegido en la *fase 3*.

En la *fase 5* se diseñan las especificaciones para la base de datos en términos de estructuras de almacenamiento físicas, colocación de registros y caminos de acceso dependiendo del DBMS que se haya seleccionado. Con herramientas CASE puede realizarse la fase 2 por ejemplo con un diagrama E/R y a partir de este diagrama hacer la fases 3, 4 y 5. Ya que estas herramientas generan el modelo conceptual hasta llegar al diseño físico, de algunos DBMS comerciales.

En la *fase 6* se hace la implementación de la base de datos para un DBMS en particular, generalmente se compilan los enunciados escritos en el DDL (Lenguaje de Definición de Datos) y en el SDL (Lenguaje de Definición de Almacenamiento) del DBMS elegido. Con esto se crea la estructura de la base de datos vacía. Después se tienen que **cargar** (alimentar) los datos a la base de datos y si los datos tienen que convertirse de un sistema computarizado antiguo,

tal vez se requieran **rutinas de conversión** para modificar el formato de los datos y así poderlos almacenar en la nueva base de datos. Una vez almacenados los datos en la base de datos, los programadores de aplicaciones deben implementar las transacciones a la base de datos generalmente con programas que contengan ordenes de DML(Lenguaje de Manipulación de Datos). Una vez que las transacciones estén listas y los datos se hayan almacenado en la base de datos, la fase de diseño e implementación habrá terminado y se inicia la fase de operación del sistema.

Diagramas Entidad-Relación

Para realizar el modelo conceptual de la base de datos generalmente nos apoyamos en un modelado de alto nivel como el E/R, para esto es necesario entender las diferentes formas que existen para representar estos diagramas. A continuación se verá la simbología utilizada para realizar estos diagramas.

IDEF1X

IDEF1X (Integration Definition for Information Modeling) es un lenguaje de especificación que utiliza figuras para representar entidades y relaciones de una base de datos relacional.

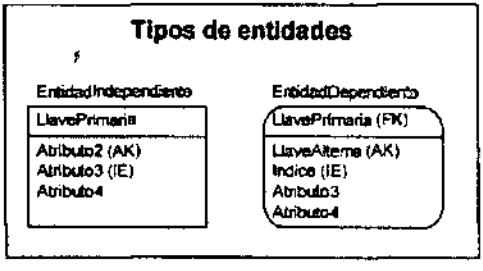


Figura 1

Hay dos tipos de entidades: entidades independientes (padre) y dependientes (hijo). Las entidades independientes tienen atributos o características propias mientras que las dependientes heredan atributos de otras entidades. En la figura 1 se muestra la representación de las entidades con las

llaves primarias(PK), las llaves foráneas (FK) y las llaves alternas(AK). También se muestran los índices (IE) que son utilizados para agilizar las búsquedas.

En este tipo de diagramas también existe simbología para representar la cardinalidad en las relaciones, la cual se muestra en la figura 2.

Figura 2

Las relaciones fuertes se dan cuando la llave primaria de la entidad dependiente es la misma que la llave primaria de la entidad independiente, y resulta de una migración de llaves. Las relaciones débiles se dan cuando se migra la llave primaria de una entidad a otra y esta no es llave primaria en la entidad a la que fue migrada, solamente forma parte de los atributos de la entidad. La figura 3 ejemplifica estas situaciones.

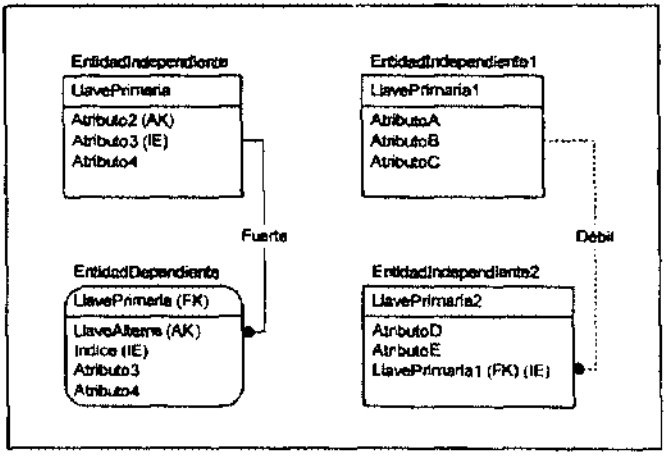


Figura 3

IDEF1X es utilizado por el software ER-Win/ERX al generar el esquema de la base de datos para algún RDBMS comercial. Lo que permite el ahorro de tiempo en el diseño lógico y físico de la base de datos.

Crow's Foot

Es un lenguaje de especificación que sirve para ejemplificar con figuras modelos de bases de datos relacionales.

La simbología utilizada es similar a IDEF1X, ya que existen los mismos elementos pero se representan de diferentes formas las entidades y el tipo de relaciones.

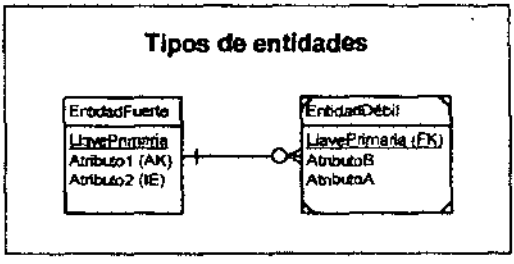


Figura 4

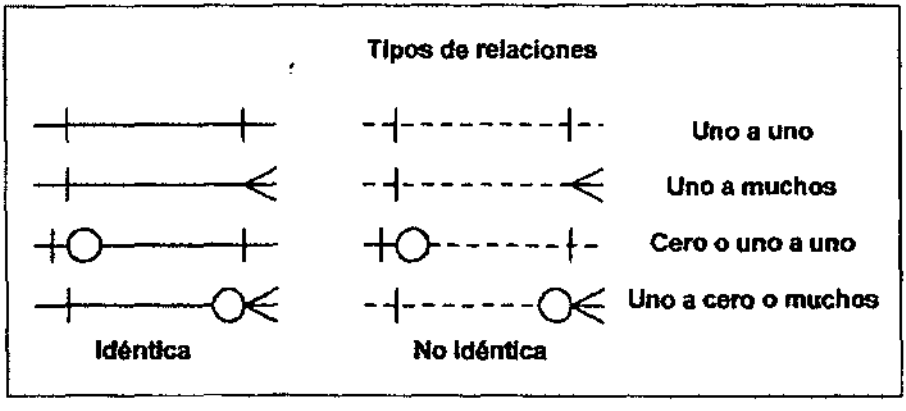


Figura 5

Estas son las simbologías más utilizadas en las herramientas CASE actuales, ya que a partir del modelo conceptual que se haga es posible generar el modelo lógico y físico para un RDBMS en particular.

Recuperación de información de bases de datos

Al realizar transacciones por Internet con acceso a bases de datos, se realizan una serie de actividades que pueden variar de acuerdo al SI que se este utilizando, pero principalmente son las siguientes:

1. El usuario llama a un programa gateway que utiliza CGI generalmente haciendo clic sobre un hipervínculo u oprimiendo un botón en el navegador.
2. El navegador reúne toda la información insertada por el usuario para enviarla al programa CGI
3. El navegador establece contacto con el servidor HTTP en la máquina donde reside el programa CGI y le pide que localice a este último y le transfiera la información.
4. El servidor HTTP corrobora si la máquina solicitante tiene autorización de acceso al programa CGI.
5. Si el usuario tiene acceso, el servidor HTTP localiza el programa gateway y transfiere la información del navegador Web al mismo. La figura nuestra los pasos del 1 al 5.

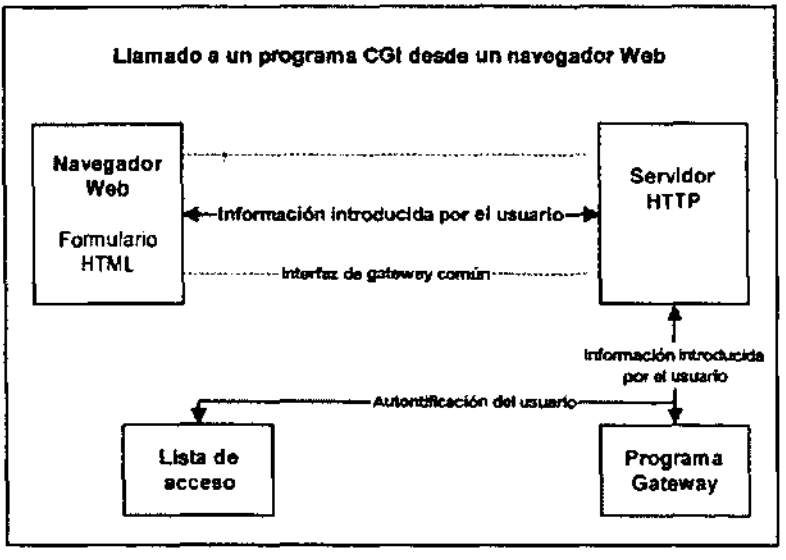


Figura 6

6. Se ejecuta el programa gateway

7. El gateway convierte la información recibida a un formato que la base de datos es capaz de entender.
8. El gateway usa el módulo de la base de datos para transferir la consulta a la interfaz de ésta.
9. La interfaz de la base de datos analiza la sintaxis de la consulta para asegurar su exactitud.
10. Si la interfaz encuentra un error de sintaxis en la consulta, se envía un mensaje de error al programa gateway.
11. El mensaje de error se envía al servidor HTTP, el cual lo transfiere al navegador Web para que éste lo despliegue al usuario. El proceso se detiene aquí.
12. Si no hay error, la interfaz envía la consulta a la base de datos.
13. El DBMS efectúa la consulta y devuelve los resultados al programa gateway a través de la interfaz.
14. El programa gateway formatea los resultados y los envía al servidor, a través del CGI, para su envío al navegador Web.
15. El navegador Web despliega los resultados.

Tecnología Cliente/Servidor

Como su nombre lo dice esta tecnología se caracteriza por tener dos elementos principales *el cliente* y *el servidor*. El cliente es la parte en la que el usuario define la información que requiere y el servidor es el que atiende la petición del cliente y contiene a la base de datos.

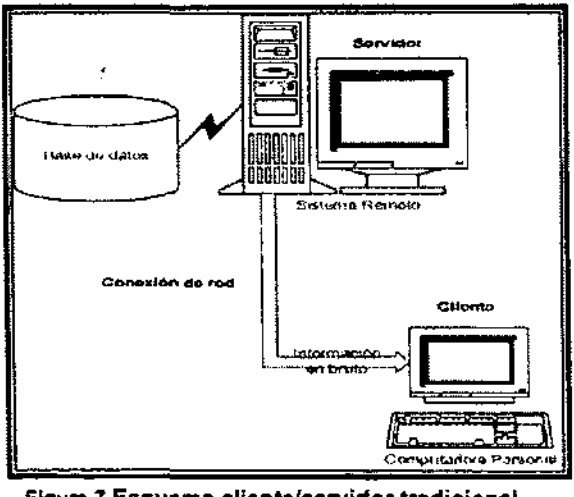


Figura 7 Esquema cliente/servidor tradicional

Tradicionalmente la información que se presentaba a un usuario parecía seguir un camino recto. El usuario inicia el proceso pidiendo información desde un programa que se ejecuta en su computadora. Este programa ejecutándose en el equipo local, tiene acceso entonces a una base de datos remota y le solicita información. La información de la base de datos se transfiere al equipo local, donde se procesa; la respuesta se le proporciona finalmente al usuario. En este modelo (figura 7), los datos en bruto se transfieren a lo largo de la red hasta el equipo local.

Mucha gente consideró la transmisión de datos en bruto era un gasto innecesario de los recursos de la computadora, pues una preciada mercancía (el tiempo necesario para transmitir la información a través de la red) se consume al transmitirlos en grandes cantidades. Para superar este inconveniente se desarrolló un modelo diferente de procesamiento en red, llamado ***cliente/servidor.***

En este modelo, el equipo local (el cliente) no solicita información en bruto sino una respuesta final. El procesamiento de la información se lleva a cabo en el sistema remoto (el servidor). De esta manera, solo se transmite la respuesta a través de la red; por lo tanto, se requiere menos tiempo y recursos como lo muestra la figura 8:

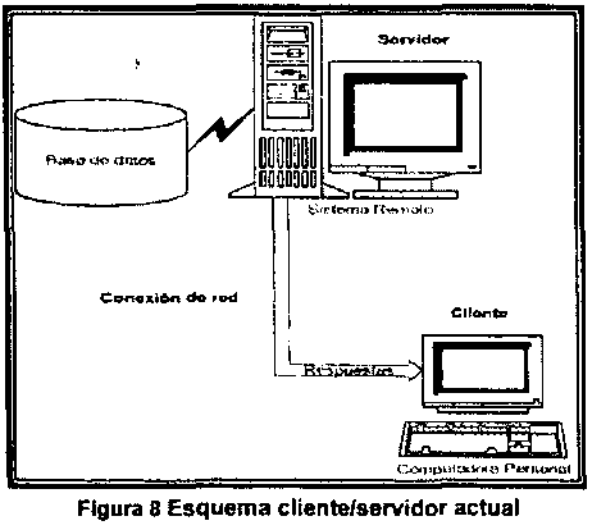


Figura 8 Esquema cliente/servidor actual

Los beneficios que brinda el modelo de procesamiento cliente/servidor son:

- Menos tiempo para la transmisión de datos.
- Los datos están más seguros, pues la fuente de datos en sí no se transmite necesariamente.
- El software cliente se puede optimizar para proporcionar la mejor interfase de usuario posible.
- El software cliente, como sólo se utiliza para hacer la solicitud y no el procesamiento principal, puede ser menos complejo y, por lo mismo, puede adaptarse mejor a computadoras pequeñas.
- El software servidor puede optimizarse para proporcionar acceso rápido de datos y un procesamiento eficiente de solicitudes.

Tipos de Software

Software de código abierto (Open Source Software)

El software libre (Free software) es aquel que viene con permiso para ser usado, copiado y distribuido por cualquier persona que lo desee. Esto significa que el código fuente debe ser disponible. Bajo esta filosofía "*si el código fuente no esta disponible no es considerado software*".

Ventajas

El usar software libre en lugar de software comercial, presenta grandes ventajas debido a que un gran número de usuarios y desarrolladores al tener acceso al código fuente pueden corregir errores, desarrollar aplicaciones y compartirlos a la comunidad de usuarios de manera inmediata. Esta situación no se presenta al usar software comercial, ya que si encontramos un error en el software lo que generalmente se tiene que hacer es esperar hasta que liberen una versión estable en donde el error sea corregido; ésto provoca un costo adicional y retardo en el desarrollo; ya que la corrección de los errores esta a cargo de un reducido grupo de desarrolladores en comparación con la comunidad de usuarios de algún software libre.

Ejemplos de proyectos de código abierto.

Entre los principales ejemplos de estos proyectos destacan:

- **Apache (http://www.apache.org)**

Apache es el más exitoso servidor de web usado en más del 60% de los servidores. Existe una gran cantidad de desarrolladores encargados de implementar y modificar nuevos módulos, estas partes son creadas de acuerdo a ciertos estándares de tal manera que sea fácil conjuntar todos los módulos en uno solo.

- **Linux (http://www.linux.org)**

Este es el mejor ejemplo de un proyecto de código abierto. El kernel fue inicialmente creado por Linus Torvalds al desarrollar un tipo sistema operativo UNIX para computadoras personales, pero mas tarde al poner a disposición el código fuente a la comunidad éste fue mejorando hasta llegar a las versiones actuales de Linux.

- **Perl (http://www.perl.com)**

Es un lenguaje de programación interpretado de alto nivel desarrollado por Larry Wall. Perl incluye una gran cantidad de características de otros lenguajes de programación, que lo hace ser uno de los lenguajes mas populares del Web por su rapidez y portabilidad debido a que es un lenguaje interpretado.

- **PostgreSQL (http://www.postgresql.org)**

El antecesor de PostgreSQL es Ingres, desarrollado en la Universidad de California en Berkeley de 1977 a 1985. Con el fin de mejorar Ingres, Michael Stonebraker generó en 1986 un nuevo servidor de bases de datos y lo llamó Postgres, es decir posterior a Ingres. Ya para 1994, Jolly Chen y Andrew Yu le agregaron la funcionalidad del lenguaje de consulta estructurado (SQL, por sus siglas en inglés), una norma mundial establecida varios lustros antes y lo llamaron Postgres95 (1994-1995). Durante 1996 se dieron dos cambios importantes: se cambió el nombre a PostgreSQL y se formó un grupo especial de desarrollo, comandado por Marc. G. Fournier en Toronto, Canadá.

- **PHP (http://www.php.net)**

Este proyecto surge bajo la iniciativa de Rasmus Lerdorf . Rasmus es la persona que tiene la última decisión en cuanto a modificaciones que se hagan al

código y coordina al equipo de desarrolladores de dicho proyecto. PHP (acrónimo de "Hypertext Preprocessor") es un lenguaje interpretado de alto nivel embebido en páginas HTML y ejecutado en el servidor.

Otros proyectos de código abierto son: los lenguajes Python, Netscape Mozilla, GLX , SAMBA, etc.

Definición de software abierto (OSD)

El código abierto no significa solamente tener acceso al código fuente. Los términos de distribución del software de código abierto deben ser complementados con los siguientes criterios:

1. Redistribución gratuita.
2. Código fuente.
3. Trabajos derivados
4. Integridad del código fuente del autor.
5. No debe haber discriminación contra personas o grupos.
6. Distribución de la licencia.
7. La licencia no debe ser especifica a un producto.
8. La licencia no debe contaminar a otro software.

La siguiente lista solo muestra algunos tipos de software; para ver una lista mas detallada consultar el sitio de la fundación del software libre FSF (http://www.fsf.org).

- **Dominio Público**

Es el software que ha sido liberado sin ninguna restricción para su uso, no existen derechos de autor.

- **Freeware**

Usualmente se refiere al tipo de software que se puede redistribuir generalmente en forma **compilada**, pero que algunas veces no incluye el código fuente, lo cual impide al usuario hacer modificaciones. El autor no pide dinero por

el programa, solamente retiene los derechos de autor y que el usuario registre el programa. No se debe confundir Freeware con software libre.

- **Semi-Freeware**

Este tipo de software es un termino medio entre freeware y shareware, ya que no se pide dinero por su uso, el autor en cambio pide que se le mande por ejemplo un email con las opiniones del software , una tarjeta postal, dulces, etc., o que se realice una labor de caridad.

- **Shareware**

Este tipo de software es el que se distribuye en forma compilada para evaluación y se rige bajo la filosofía de "úsalo antes de comprar", por lo que no se distribuye el código fuente. Generalmente después de un periodo de tiempo el programa pide que compres la licencia de uso. Se distribuye la versión completa del software.

- **Demoware/Crippleware**

A diferencia del shareware en este software solamente se ponen a evaluación algunas herramientas de la versión completa. Lo que imposibilita el uso total del software.

- **Software propietario**

Es un software diseñado especificamente para una empresa, solamente el autor tiene derecho a modificar el código fuente y se tiene que pagar el uso de la licencia. Además no se permite la redistribución a menos que se llegue a un acuerdo con el autor. Ejemplo de este tipo de software es el que usan las líneas aéreas, bancos, sistemas de seguridad, etc en donde la confidencialidad en las transacciones es importante,

- **Software comercial**

Es el que desarrolla una empresa para venderlo a cualquier persona que desee utilizarlo, por ejemplo Microsoft Office.

Capítulo

Ciclo de Vida del Sistema de Información

En este capítulo se describen las etapas del ciclo de vida del *Sistema de consulta vía Internet a la base de datos de vientos y contaminantes de la ciudad de México*. Como podrá observarse a lo largo de este capítulo, en cada etapa se hace una descripción de las actividades realizadas, auxiliándonos con diagramas formales como son los diagramas de flujo de datos, diagramas E/R, diagramas de tiempo, cartas de estructuras, esquemas de la base de datos, entre otros; todo esto se hace para entender de una manera fácil el sistema desarrollado.

Etapa 1: **Análisis**

1.1 Estudio preliminar

1.1.1 Objetivo.

Crear una base de datos con la que se pueda consultar información de las mediciones hechas por diversas estaciones de monitoreo ambiental en la ciudad de México. Esta información será consultada a través de Internet.

Debido a la naturaleza del sistema, se debe atender en especial al modelo cliente / servidor, mencionado en un apartado anterior. Ya que es necesario que el sistema tome la petición del usuario y le entregue los resultados procesados, es decir el sistema debe procesar la información recibida de la base de datos que radica en el servidor y mandarle al usuario la información que él requiera en el formato especificado, logrando así un incremento en la velocidad de respuesta a las consultas.

1.1.2 Planteamiento del problema.

De acuerdo a las sesiones celebradas entre los usuarios del sistema, se ha establecido que éste estará dividido en las siguientes partes:

1. Programación del simulador de transporte.
2. **Base de datos de vientos y contaminantes de la Cd. de México.**
3. Inventario de fuentes contaminantes.
4. Tener un simulador de cinética química.
5. Visualización de resultados con Vis5d
6. Calibración del modelo
7. Optimización del programa.
8. Reporte de resultados.

En la lista anterior se puede observar que el propósito de este trabajo es desarrollar el punto número 2, referente al desarrollo de la base de datos de

vientos y contaminantes de la Cd. de México. En donde será posible aplicar los conocimientos adquiridos a lo largo de la carrera de ingeniería en computación, principalmente de las materias relacionadas con el desarrollo de sistemas.

De las reuniones celebradas con los usuarios del sistema, se pueden enunciar los requerimientos que debe cumplir el subsistema para poderlo acoplar con los otros módulos que conforman al sistema principal, estos requerimientos son:

a) Analizar el formato la información recibida por Red Automática de Monitoreo Ambiental (**R.A.M.A**).

b) Diseñar la base de datos de vientos y contaminantes para que se apegue al modelo entidad relación.

c) Crear la base de datos con un sistema manejador de bases de datos relacionales (**R.D.B.M.S.**) como ORACLE o PostgreSQL.

d) Desarrollar un programa que adapte el formato recibido de R.A.M.A. a un formato que sea utilizado por ORACLE o PostgreSQL.

e) Verificar que la información suministrada a la base de datos sea correcta.

f) Desarrollar la interfase para que la base de datos pueda ser consultada por la Internet.

g) Realizar programas que elaboren consultas especificas para los investigadores que lo requieran.

Para realizar lo anterior es necesario hacer un análisis de la información recibida de las estaciones de monitoreo. Este análisis se hará principalmente a los archivos que son recibidos de la R.A.M.A. , ya que el formato que manejan es de Dbase 3.0 . Esta información se encuentra almacenada en un disco compacto con aproximadamente 10 años de mediciones, los cuales deben ser analizados y

descartar las mediciones de algunos parámetros que no son considerados en la realización de este sistema.

Analizando la información otorgada por la RAMA se puede observar que para el mes de noviembre de 1998 los archivos que se reciben son:

```
C:\Mis documentos\paco\simca\9811>dir *.dbf
9811COP.DBF                94,221
9811H2SP.DBF               17,564
9811NO2P.DBF               98,438
9811NOXP.DBF               98,438
9811O3P.DBF               103,429
9811PM1P.DBF               58,613
9811RHP.DBF                58,240
9811SO2P.DBF              138,812
9811TMPP.DBF               57,906
9811UVBP.DBF               12,512
9811WDRP.DBF               62,422
9811WSPP.DBF               51,298
```

Del cuadro anterior se puede apreciar que el nombre de cada archivo nos da información de su contenido. Por ejemplo, el archivo **9811COP.DBF** nos dice que se tienen las mediciones de **CO** en el mes **11** del año **98**, la letra **P** al final del nombre del archivo nos indica que las mediciones están en partes por millón. Y así para todos los archivos que se encuentran en el directorio.

Para entender mejor el significado de estas abreviaturas se muestra la ***tabla A.1***, en donde es posible observar la clave y el nombre de la magnitud monitoreada. Esto es de gran importancia debido a que el diseño de la base de datos se debe adecuar a esta información. La clave -en el caso de los contaminantes- no representa su símbolo químico, ya que es una representación del valor que será utilizado para almacenar dicho datos en la base de datos.

Para mayor información de las características de los contaminantes y el efecto que presentan en la salud del hombre se sugiere consultar el **Apéndice 1**. En este apéndice se podrán ver las unidades manejadas y los rangos considerados para determinar la calidad del aire.

Clave	Nombre
O3	Ozono
NOX	Óxido de Nitrógeno
NO2	Óxido Nitroso
SO2	Anhídrido Sulfuroso
CO	Monóxido de Carbono
H2S	Acido Sulfúrico
PM1	Partículas Suspendidas
WSP	Velocidad de los vientos
WDR	Dirección de los vientos
RH	Humedad relativa
TMP	Temperatura
UVB	Indice ultravioleta

Tabla A.1 Magnitudes monitoreadas por la R.A.M.A.

Al abrir uno de los archivos de Dbase (*.dbf) se puede observar que el formato en que R.A.M.A. maneja la información es el siguiente:

FECHA	HORA	MCOLAG	LAG	MCOTAC	TAC	MCOEAC	EAC	MCOTLA	TLA	MCOXAL	XAL	MCOMER	MER	MCOPED	PED
1/11/98	1	3,1		1,4		2,8		2,3		2,7		4,1		0,7	
1/11/98	2	2,4		0,9		2,6		2,2		2,5		3,1		0,7	
1/11/98	3	2,4		1,0		2,1		2,2		1,8		3,0		0,7	
1/11/98	4	2,2		1,1		1,8		1,4		1,7		2,2		0,7	
1/11/98	5	2,9		1,0		1,6		1,6		2,6		2,5		1,0	
1/11/98	6	2,2		1,1		1,8		1,6		1,9		3,5		0,8	

Formato de los archivos *.DBF manejados por la RAMA.

Una muestra de este archivo puede ser revisado en el **Apéndice 2**. Se habla de una muestra debido a que cada archivo tiene 720 o 744 filas (cantidad que sería difícil mostrar en una hoja de este documento); esto es porque las mediciones se toman cada hora y los archivos vienen con datos mensuales; es decir si el mes tiene 30 días el archivo tendrá 720 filas, si tiene 31 tendrá 744.

En estos archivos se puede apreciar (después de analizar la información) todos inician con el siguiente patrón:

FECHA	HORA	MCOLAG	LAG	MCOTAC	TAC	MCOEAC	EAC	MCOTLA	TLA	MCOXAL	XAL	MCOMER	MER	MCOPED	PED

En donde:

FECHA. Es la fecha de la medición. Y está dada en día/mes/año, el problema es que el año es manejado solo con 2 dígitos. Por tal motivo se tiene que considerar que los actuales R.D.B.M.S. manejan 4 dígitos en el año por el tan mencionado problema del año 2000.

HORA. Es la hora de la medición. El rango de horas es de 1 a 24, pero ORACLE y PostgreSQL maneja de 0 a 23, por lo tanto se tiene que hacer la conversión de horas.

La cabecera de cada columna de las mediciones vienen con el siguiente formato **M***Magnitud**Estacion*; por ejemplo para **MCOLAG** la **M** indica que es una ***Medición***, **CO** es el ***nombre de contaminante o magnitud*** y **LAG** es la ***clave de la estación*** que toma medidas de la magnitud.

Este patrón se repite para todos los archivos *.DBF que maneja la RAMA. Lo único que varía son los nombres de las mediciones y de las estaciones que realizan dichas mediciones. Cabe recalcar que sólo algunas mediciones se realizan en cada estación de monitoreo.

Por lo tanto se tiene que crear un programa que transforme el formato anterior en un formato que sea entendido por el RDBMS de ORACLE o PostgreSQL.

Para la creación de scripts para ORACLE o PostgreSQL es necesario tener en consideración la manera en que se va a manejar y la forma en que es recibida la información, por tal motivo se pretende utilizar PERL por la facilidad que presenta para manejar archivos y en especial el manejo de cadenas de texto.

Para entender el significado de las claves se debe tener en cuenta que en la Cd. de México las estaciones de monitoreo ambiental se encuentran distribuidas geográficamente de la siguiente manera:

Zona	Clave	Nombre
NO	VAL	Vallejo
	TAC	Tacuba
	EAC	ENEP Acatlán
	AZC	Azcapotzalco
	TLA	Tlanepantla
	IMP	IMP(Instituto Mexicano del Petroleo)
	CUI	Cuitláhuac
	TLI	Tultitlán
	ATI	Atizapan
NE	LLA	Los Laureles
	LPR	La Presa
	LVI	La Villa
	SAG	San Agustín
	XAL	Xalostoc
	ARA	Aragón
	NET	Netzahualcoyotl
	CHA	Chapingo
	VIF	Villa de las Flores
CE	LAG	Lagunilla
	MER	Merced
	HAN	Hangares
	BJU	Benito Juárez
	MIN	Insurgentes
	PL	Palacio Legislativo
SO	SUR	Santa Ursula
	PED	Pedregal
	PLA	Plateros
	CUA	Cuajimalpa
	TPN	Tlalpan
SE	CES	Santa Ursula
	UIZ	Pedregal
	TAX	Plateros
	TAH	Cuajimalpa

Tabla A.2 Claves de las estaciones de monitoreo de la Ciudad de México.

Para mostrar las zonas en que se ubican estas estaciones se presenta el siguiente mapa del Distrito Federal dividido por zonas geográficas.

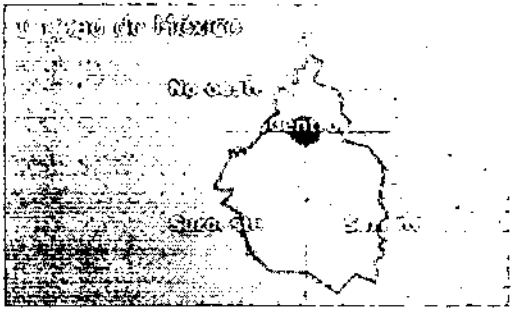


Figura A.1 Distribución geográfica de la ciudad de México

1.1.3 Alcance del sistema.

El objetivo de este sistema es poder brindar información específica, obtenida a partir de las mediciones que realizan diversas estaciones de monitoreo ambiental, a los investigadores del problema de la contaminación en la Cd. de México.

Además se pretende que este sistema sirva para hacer pruebas referentes al desempeño de diversos R.D.B.M.S., pero principalmente a aquellos que se distribuyen gratuitamente bajo licencia pública general (**GPL**), como es el caso de PostgreSQL. Ya que esto servirá para ahorrar costos de licencias a la UNAM.

1.2 Planeación

1.2.1 Planeación de las actividades a realizar.

Para la elaboración de este sistema se tiene planeado utilizar un R.D.B.M.S. de ORACLE, pero de acuerdo a las pruebas realizadas a PostgreSQL, tal vez en un futuro se haga una migración a este R.D.B.M.S.

Al momento de escribir este documento ORACLE cuenta con una gran experiencia en el ramo de las bases de datos, mientras que PostgreSQL se encuentra en constante evolución debido a que existe una gran cantidad de personas que constantemente hacen mejoras al código fuente para mejorar el rendimiento y acercarse cada vez más a los niveles de ORACLE.

En un futuro se debe tener especial consideración en el hardware y software de los servidores en los que se pretende instalar el sistema. Las diferencias principales entre estos servidores son:

a) La arquitectura (hardware). El servidor con sistema operativo Solaris es una estación de trabajo de la marca SUN. El servidor con sistema operativo Linux es una computadora personal genérica.

b) El costo. El costo de una estación de trabajo es mucho mayor al de una computadora personal.

c) Las aplicaciones (software). Generalmente el software de una estación de trabajo es costoso ya que se tienen que pagar licencias para poder utilizarlo, en cambio las aplicaciones que corren bajo Linux generalmente son de distribución gratuita por la naturaleza misma del sistema operativo.

Estos puntos deben ser considerados, ya que el manejo de una arquitectura u otra tiene que ver directamente con el precio, y lo que se pretende es precisamente disminuir costos en la realización de éste y otros sistemas principalmente por tratarse de una institución educativa.

Las principales actividades a realizar son:

- **Analizar la información recibida por R.A.M.A.**

En este punto se debe analizar detenidamente el formato que es utilizado en los archivos *.DBF para poder identificar el patrón que siguen y así realizar el programa que transforme dicho formato a uno manejado por el R.D.B.M.S.

- **Diseñar la base de datos de vientos y contaminantes de tal manera que se apegue al modelo entidad relación.**

Una vez analizados los datos e identificadas las posibles entidades se prosigue al diseño de la base de datos de tal manera que se apegue al modelo entidad relación. Para esto se cuenta con ER-Win que es una herramienta (CASE) para diseñar bases de datos relacionales. Con esta herramienta es posible ver el modelo lógico y físico de la base de datos.

- **Crear la base de datos con R.D.B.M.S. como es ORACLE o PostgreSQL.**

Aprovechando las ventajas que presenta ER-Win en cuanto a la generación de scripts para determinados R.D.B.M.S. se creará la base de datos a partir del diseño que se tiene; para ésto se debe tener especial consideración en los tipos de datos que tendrán cada atributo en las entidades manejadas.

- **Hacer un programa que adapte el formato recibido de RAMA a un formato que sea utilizado por el R.D.B.M.S. y alimente a la base de datos relacional.**

A partir del análisis de la información recibida por RAMA se debe generar un programa que transforme los archivos *.DBF a scripts que sean manejados por el R.D.B.M.S., generalmente estos scripts son archivos de texto plano con extensión *.sql. Se pretende transformar los archivos *.DBF a archivos de texto *.txt separado por tabuladores, utilizando Excel. Una vez transformados los archivos a texto separado por tabuladores, es más fácil procesarlos con PERL debido a que existen funciones especificas para el manejo de cadenas en archivos de texto. Una vez creados los scripts, los datos que éstos contienen será suministrada a la base de datos relacional.

- **Verificar que la información suministrada a la base de datos sea la correcta.**

Mediante cifras de control se verificará que los datos suministrados a la base de datos sean los correctos.

Representando al archivo *.DBF en una matriz cuadrada, las cifras de control serán seleccionadas de la siguiente manera:

FECHA	HORA	MCOLAG	LAG	MCOTAC	TAC	MCOEAC	EAC	MCOTLA	TLA	MCOXAL	XAL	MCOMER	MER	MCOPED	PED
1/11/98	1	3,1		1,4		2,8		2,3		2,7		4,1		0,7	
1/11/98	2	2,4		0,9		2,6		2,2		2,5		3,1		0,7	
1/11/98	3	2,4		1,0		2,1		2,2		1,8		3,0		0,7	

En donde las celdas resaltadas son las mediciones seleccionadas como cifras de control. Obteniendo con esto que al verificarse el valor de las cifras en la base datos se garantice que todos los datos han sido agregados correctamente.

- **Realizar la interfaz para que la base de datos pueda ser consultada por la Internet.**

Por otra parte se pretende realizar programas CGI's (Common Gateway Interface) escritos en PERL que procesen las peticiones de las usuarios del sistema que realicen consultas a través de formularios en páginas web. Estos CGI´s generarán la consulta para hacer la petición a la base de datos a partir de las opciones seleccionadas en dichos formularios. Una vez obtenido el resultado de la consulta éste será procesado de tal manera que el usuario pueda obtener alguno de los dos formatos de salida, ya sea formato de texto o formato HTML.

El sitio deberá estar estructurado de tal manera que el usuario encuentre un ambiente amigable y fácil de usar para realizar las consultas.

- **Elaborar consultas específicas para los investigadores que lo requieran.**

Se tiene que preguntar a los investigadores y personas involucradas en el proyecto cuáles son las consultas que requieren para realizar su investigación. De acuerdo a la respuesta que ellos den se formarán las consultas para que estén disponibles a través del sitio. Cabe mencionar que por motivos de seguridad el sitio solo servirá para hacer consultas de selección; no se permitirá actualizar,

insertar o borrar datos, mucho menos modificar el diseño de la base de datos, esto solo lo podrá hacer el administrador del sistema cuando lo crea conveniente.

1.2.2 Recursos con los que se cuenta.

En los siguientes puntos se mencionan los recursos con los que se cuenta para realizar este sistema. Principalmente se enumeran los recursos de hardware y software con los que se pretende trabajar y la bibliografía que será utilizada como medio de apoyo técnico.

1. Se tienen cuentas de propietario de la base de datos (DBO); una de ORACLE y otra de PostgreSQL.
2. Ambos servidores en los que se encuentran los R.D.B.M.S. tienen instalado PERL y cuentan con servicios de web (Apache Web Server); además, se tienen permisos para la creación de CGI's y páginas web.
3. Se cuenta con el software necesario (Excel) para procesar la información recibida de R.A.M.A., diseñar la base de datos (ER-Win) y realizar los diagramas necesarios (Visio 5) para documentar.
4. Se cuenta con bibliografía especializada de PERL, HTML, ORACLE y PostgreSQL.

1.2.3 Tiempo estimado.

El siguiente diagrama muestra el tiempo estimado para realizar las actividades relacionadas con cada etapa del Ciclo de Vida de los Sistemas de Información. Cabe mencionar que este diagrama fue hecho considerando los tiempos de holgura para cada actividad.

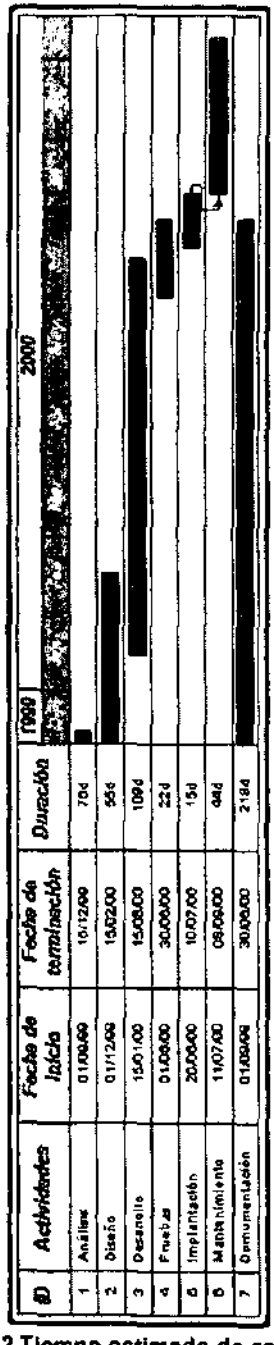


Figura A.2 Tiempo estimado de actividades

1.3 Análisis detallado

1.3.1 Diagrama de Flujo de Datos (DFD)

Para mostrar de manera esquemática los procesos que se deben realizar en el sistema de consulta a la base de datos de vientos y contaminantes de la Cd. de México, se han realizado dos DFD's uno para el usuario y otro para el administrador del sistema.

El primer DFD nos muestra los procesos asociados al **usuario** del sistema.

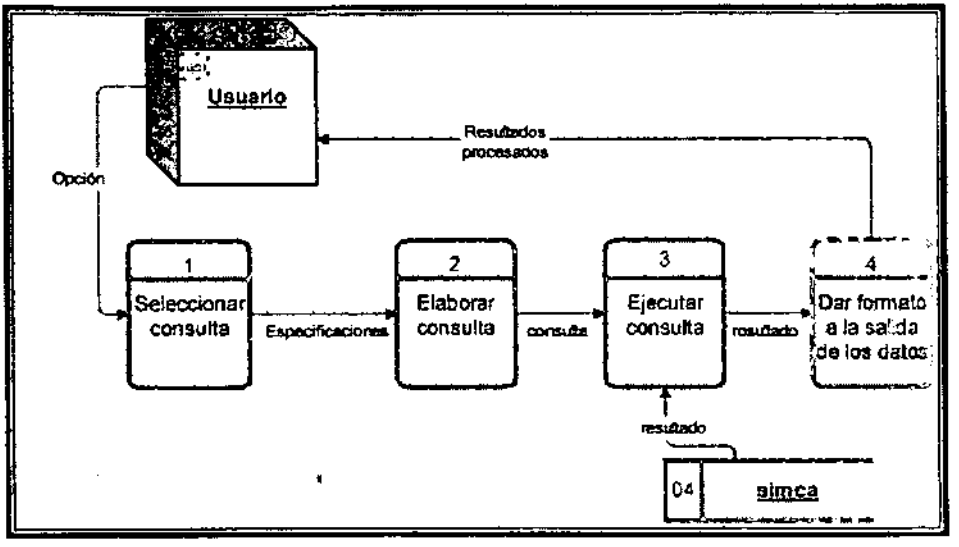


Figura A.3 DFD referente al usuario (nivel 0)

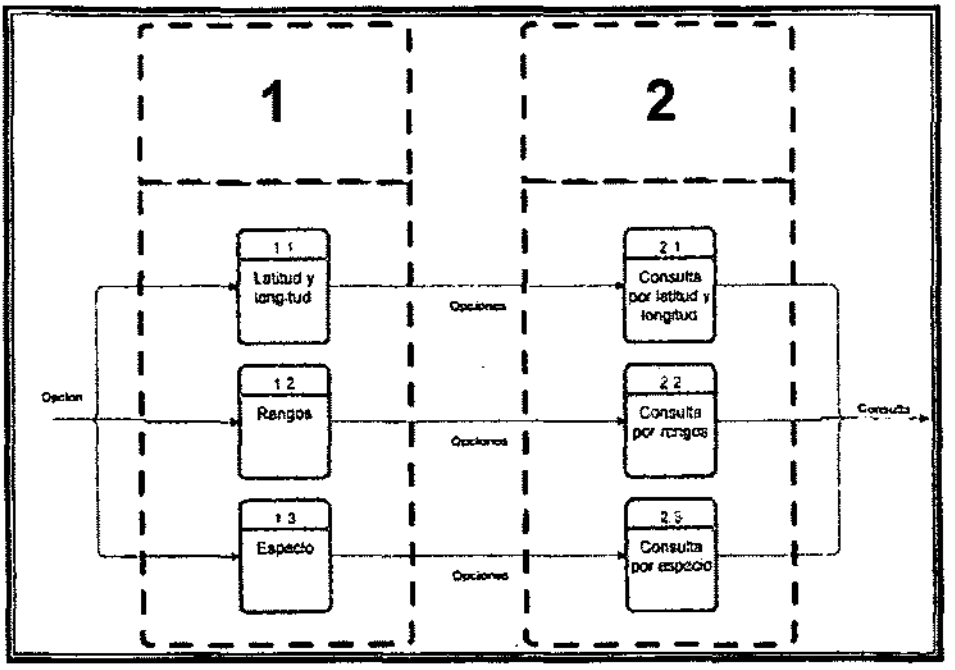


Figura A.4 DFD referente al usuario procesos 1 y 2 (nivel 1)

El anterior DFD (nivel 0 y 1) ha sido desarrollado considerando que el usuario del sistema tendrá la posibilidad se seleccionar de un menú de opciones la consulta que desee (opción), una vez seleccionada la opción el usuario definirá los valores que quiera consultar (especificaciones), con las especificaciones el programa (CGI) elaborará la consulta (comandos SQL) a la base de datos simca y por último a los registros seleccionados se les dará un formato de salida ya sea texto plano separado por tabuladores o HTML.

El segundo DFD nos muestra los procesos asociados al **administrador** del sistema. El cual transformará los archivos *.DBF a archivos.txt y los enviará al servidor en donde se encuentre RDBMS para que un programa transforme los archivos.txt en scripts SQL que "alimenten" de datos a la base de datos simca, por último verificará que los datos suministrados sean los correctos.

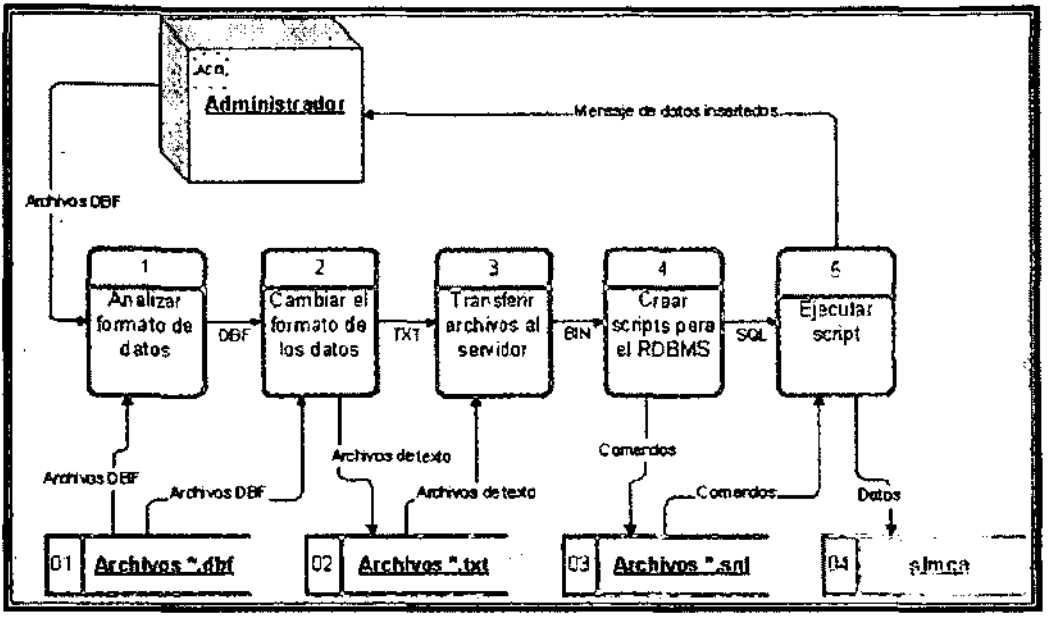


Figura A.5 DFD referente al administrador

Para ver la documentación de los procesos, entidades externas y flujos de datos de los DFD's antes mencionados se sugiere consultar el **Apéndice 4**.

Etapa 2: **Diseño**

2.1 Diseño estructurado

El método del análisis estructurado nos permite comprender la interacción que existe entre los módulos que conforman un sistema de información, representando a cada módulo con diagramas que sean fáciles de entender por el programador y por el usuario final.

Como se verá en los siguientes diagramas, el sistema se divide en componentes que permiten la construcción de un modelo del sistema.

El análisis estructurado se concentra en especificar lo que se requiere que haga el sistema o la aplicación. No se establece cómo se realizarán las tareas, ni que lenguajes de programación se utilizarán o la forma en que se implantará la aplicación.

Este método permite que las personas observen los elementos lógicos (lo que hará el sistema) separados de los componentes físicos (equipo de cómputo, terminales, servidores, medios de almacenamiento, etc.)

El diseño estructurado representa gráficamente la conexión que existe entre los distintos módulos que conforman al sistema de información, las variables y el flujo de datos de un módulo a otro. Este diseño nos permite ver la jerarquía y secuencia de los módulos y es una clara ayuda para saber cuántos y cuáles son los programas a realizar, por lo que este diseño se debe hacer antes de que se empiece a generar el código para cada módulo.

Al igual que los DFD's se harán las cartas de estructura para el usuario y para el administrador. Estas cartas de estructura no son mas que la representación gráfica de los programas a realizar.

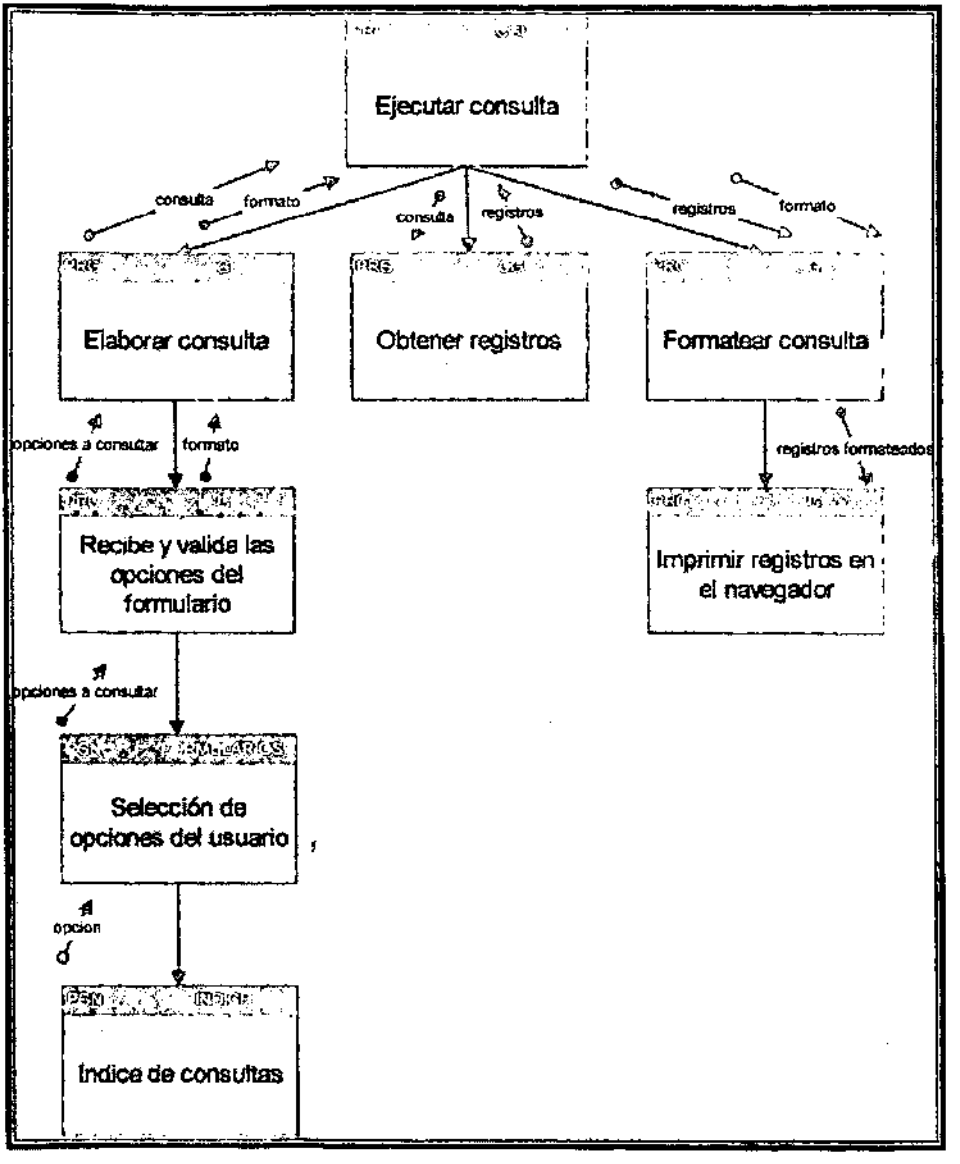


Figura D.1 Carta de estructura para el usuario

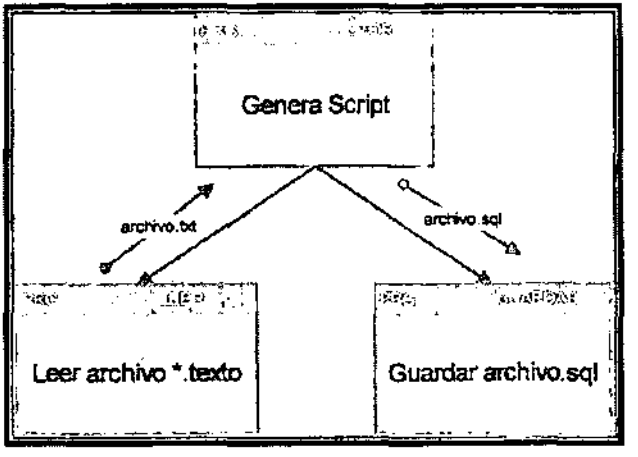


Figura D.2 Carta de estructura para el administrador

Gracias a las cartas de estructura antes mencionadas es posible identificar fácilmente los módulos que hay que programar, a estas cartas muchas veces se les conoce como mapa del programador.

En las figuras D.1 y D.2 se puede observar que los programas se representan con rectángulos, las variables y los datos con flechas; por ejemplo PRG indica que es un programa y PGN se refiere a una página web

2.2 Modelado de datos

El modelado de datos es una etapa del CVSI en la que básicamente se identifican las entidades con los atributos que las conforman, con el fin de diseñar la base de datos. Es una primera aproximación a la estructura que tendrá la base de datos.

A partir del análisis de la información recibida de la RAMA es posible enumerar los siguientes puntos:

- Cada archivo trae el nombre de cada una de las 12 magnitudes que son monitoreadas por la RAMA, además de mencionar la fecha de las mediciones (año y fecha).

- Dentro de cada archivo vienen las mediciones de las magnitudes realizadas por las estaciones de monitoreo, las cuales inician desde la primer hora del primer dia, hasta la última hora del último día del mes. Cabe recalcar que no todas las mediciones de magnitudes son realizadas por cada estación de monitoreo ambiental. Por ejemplo la estación ubicada en Chapingo (CHA) monitorea las cantidades de Ozono (O3) pero no monitorea la velocidad del viento (WSP). El formato de las fechas es importante ya que en los archivos el formato utilizado es: Día/Mes/Año Hora. Además se debe considerar que el Año es manejado con **2 dígitos** y el rango de horas va de **1 a 24**.

- Se tiene la ubicación de las estaciones y las claves de las magnitudes (contaminantes, UVB, WSP, etc.) a medir.

- Nos interesa almacenar en la base de datos
  - Nombre de la magnitud
  - Clave de la estación que monitorea dicha magnitud
  - Fecha y hora de la medición
  - La medición.

Atendiendo a los puntos anteriores es posible identificar las entidades **Estación** y **Magnitud** así como la relación **Monitorea** con los atributos que las conforman en la figura D.3.

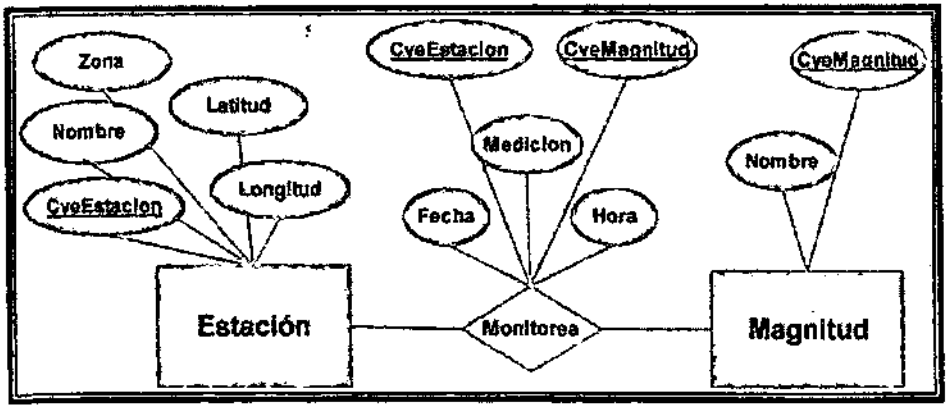


Figura D.3 Diagrama de esquema E-R para la base de datos SIMCA utilizando la simbología de Chen.

2.3 Normalización

La normalización es un proceso de identificación y migración de llaves, es decir establecer las llaves primarias y foráneas de cada entidad. Este proceso se realiza con la finalidad de evitar redundancia de datos, para agilizar las búsquedas y ahorrar espacio en disco.

En esta etapa no sólo se atiende al modelado lógico de los datos, sino también se hace énfasis en el modelado físico para definir cuales van a ser los tipos de datos a utilizar.

Utilizando el software ER-Win/ERX 2.5.01 for Power Builder (el cual se puede instalar en cualquier sistema operativo windows), se desarrolló el esquema de la base de datos utilizando el generador de esquema para ORACLE v.7. Este software es de gran ayuda pues a partir del diseño gráfico de la base de datos permite generar un script que contiene una serie de comandos SQL correspondientes al lenguaje de definición de datos (**DDL**), entre los que destacan CREATE TABLE, CREATE INDEX, CREATE TRIGGER, entre otros. Este script contiene la descripción física de la base de datos pues cada comando SQL define como serán creadas las tablas y las relaciones que existen entre ellas así como las acciones que se sucederán al tratar de ingresar datos que no correspondan a los definidos previamente en el diseño de la base de datos.

Una vez que se haya entrado a trabajar con el RDBMS, es posible ejecutar todos los comandos del script generado con ER-Win y así tener rápidamente la base de datos apegándose al diseño previo.

La versión de ORACLE que se tiene instalada en el servidor , al momento de realizar este documento, corresponde a la 8. No obstante, existe una completa compatibilidad con la versión 7.

El diseño de la base de datos se hizo considerando puntos específicos de la información recibida de la RAMA, entre los que destacan la manera en la que almacenan la información en los archivos DBF y principalmente el formato de las

fechas que manejan, ya que éste es un punto importante para el manejo de los tipos de fechas en la base de datos.

Como se puede ver en el diagrama existen dos entidades independientes (estación y medida) y doce entidades dependientes que corresponden a las magnitudes monitoreadas. Se realizaron estas doce entidades independientes debido a que mensualmente se reciben doce archivos con las mediciones realizadas durante el mes; por lo que la relación monitorea fue dividida en doce entidades. Con esta división se agiliza la consulta a cada tabla y se mantiene la integridad referencial ya que se migran las llaves de las entidades padre a las entidades hijas y se forman así las relaciones, garantizando con esto que antes de ingresar los datos a las entidades hijas existan los valores correspondientes en las entidades padres. Por ejemplo si se quieren insertar datos a la tabla **O3** de alguna estación que no este definida en la tabla padre **Estación**, los datos serán descartados hasta que no se agregue la estación a la tabla Estación. Una situación similar sucede con la tabla **Medida**.

En las figuras D.4 y D.5 se muestran los diseños de la base de datos atendiendo al modelo E-R. El primer esquema muestra la estructura de la base de datos para ORACLE v. 8 y el segundo para PostgreSQL v. 6.25. En ambos diagramas se utilizó la simbología IDEF1X (Integration Definition for Information Modeling), la cual nos permite representar con figuras los elementos de una base de datos.

El motivo por el que se hicieron dos diseños se revisa en detalle en la etapa de desarrollo. Pero se puede observar que el esquema es similar, las variaciones que se dan son referentes al manejo de los tipos de datos para fechas, ya que ambos DBMS tienen su forma propia para almacenar este tipo de información. En el caso de ORACLE v.8 es posible almacenar la fecha (DD/MM/YYYY) y la hora (HH24) en un solo atributo, en cambio PostgreSQL v. 6.25 maneja la fecha y la hora por separado ya que en esta versión todavía no se implementa un tipo de dato en el que se incluyan fecha y hora juntos.

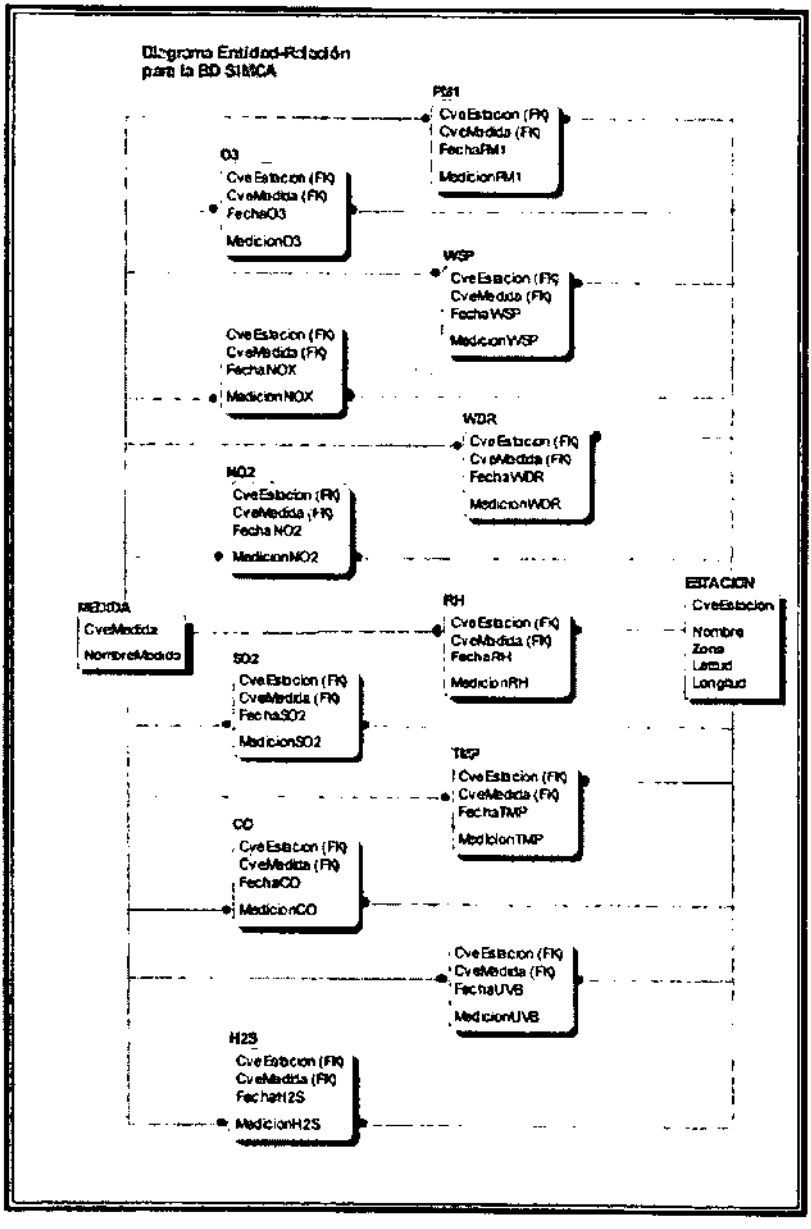


Figura D.4 Esquema E-R de la base de datos SIMCA para ORACLE v.8

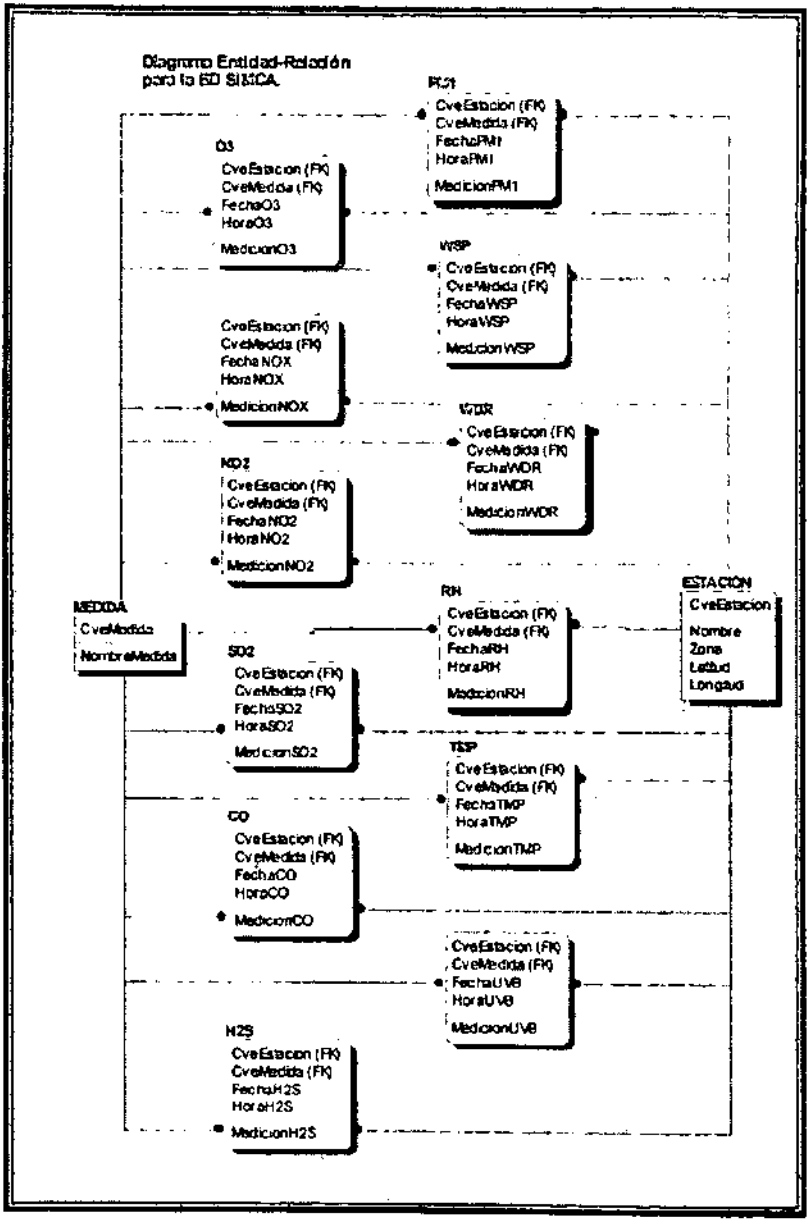


Figura D.5 Esquema E-R de la base de datos SIMCA para PostgreSQL v.6.25 y v.7.0.1

Para la descripción de cada entidad y de sus atributos se sugiere consultar el **Apéndice 4** en el apartado dedicado al diagrama E-R. Aquí se podrán observar aspectos tales como el tipo de datos de los atributos, las llaves primarias y foráneas, etc. Además se puede consultar el contenido de los scripts utilizados para la generación de la parte física de la base de datos en los dos RDBMS.

Como se puede observar en los esquemas anteriores las entidades padres corresponden a la entidad ESTACION y a MEDIDA, las cuales migran su llave primaria a la relación MONITOREA la cual esta divida en las magnitudes que son monitoreadas por cada estación.

La relación MONITOREA tiene la llave primaria compuesta por la clave de estación, clave de la magnitud y para hacerla llave primaria se agregó el atributo fecha, con esto es posible identificar a cada registro de forma única.

2.4 Prototipo

El método del prototipo es de gran ayuda en el desarrollo de sistemas de información debido a que muestra una primera aproximación del sistema funcionando. Con el prototipo es posible analizar puntos referentes al diseño y funcionamiento del sistema, en donde los usuarios finales pueden llegar a hacer sugerencias para hacer más amigable el uso del mismo.

El prototipo contempla la interfaz web para realizar consultas a la base de datos simca mediante páginas web. Estas páginas tendrán formularios con botones para entrada de datos, entradas tipo checkbox, radio y selección.

Esto debido a que los programas CGI´s procesan principalmente la información recibida de los formularios, por eso es muy importante poner especial atención al tipo de entradas que se pondrán en los formularios y el nombre de las variables que se utilizarán.

La siguiente figura muestra el prototipo del sitio en donde podrá ser consultada la base de datos. En esta figura se puede apreciar un Menú del lado izquierdo, un formulario del lado derecho y en la parte superior el nombre del

proyecto. El prototipo solamente muestra la forma en que se pretende se hará la navegación por el sitio.

Como lo muestra la siguiente figura, el sistema tendrá la interfase vía web en donde sea posible seleccionar el tipo de consultas y los rangos para delimitar los resultados obtenidos de dicha consulta, así como el formato en que serán presentados.

El prototipo de cualquier sistema es de gran ayuda pues es creado con la finalidad de probar ideas y suposiciones, es la primera versión y es un sistema de prueba que funciona, en el que los usuarios evalúan tanto el diseño como la efectividad de la información proporcionada por el mismo.

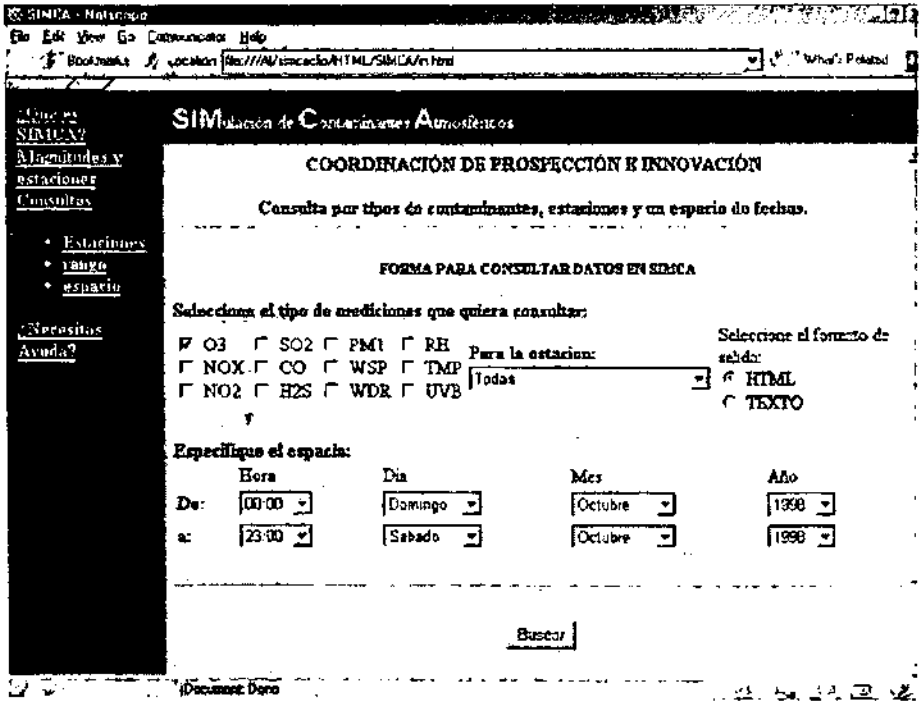


Figura D.6 Prototipo del sistema de consulta a la base de datos SIMCA

Etapa 3: **Desarrollo**

3.1 Creación de la base de datos

La base de datos en ORACLE se creo de la siguiente manera:

- Se generó el script con ER-Win y se guardó con el nombre BDSIMCA.sql. Este script contiene una serie de comandos SQL del lenguaje de definición de datos (DDL) con los que es posible la creación de la base de datos.
- Debido a que el software ER-WIN se encuentra instalado en sistemas operativos windows, es necesario transferir el archivo BDSIMCA.sql (mediante FTP) al servidor en el que se encuentre instalado el RDBMS de ORACLE ya que no cuenta con soporte para conectarse como cliente de ORACLE v.8. El nombre del servidor es **alpha.dgsca.unam.mx**.
- Una vez que el archivo este en el servidor, iniciamos una sesión (mediante TELNET) en éste y ejecutamos el programa SQLPLUS, lo cual desplegará el siguiente mensaje:

```
alpha% sqlplus

SQL*Plus: Release 8.0.4.0.0 - Production on Fri Jun 16 17:42:9 2000

(c) Copyright 1997 Oracle Corporation.  All rights reserved.

Enter user-name: system
Enter password:

Connected to:
Oracle8 Enterprise Edition Release 8.0.4.0.0 - Production
PL/SQL Release 8.0.4.0.0 - Production

SQL>
```

- Una vez que introducimos el login y password entramos al ambiente de SQLPlus, el cual es la versión de SQL para ORACLE. Para crear la base de

datos simca ejecutamos los comandos que vienen en el script BDSIMCA.sql de la siguiente manera:

```
SQL>@BDSIMCA.sql
```

- Se debe tener en cuenta que el script BDSIMCA.sql se encuentre en el mismo directorio desde donde se llamo al programa SQLPlus, si se ejecuta el programa desde otro directorio se tiene que especificar la ruta en donde se encuentra el script. Si la ubicación del script es la correcta ya se encuentra creada la base de datos SIMCA.
- El siguiente paso es crear los programas que transformen los archivos *.txt en scripts.sql, los cuales servirán para alimentar de datos a la base de datos **SIMCA**.

La base de datos en PostgreSQL fue creada de manera similar.

- Se generó el script con una serie de instrucciones SQL y se guardó con el nombre simcapostgres.sql.
- Se transfiere el archivo por ftp al servidor.
- Iniciamos una sesión en el servidor **clio.dgsca.unam.mx** (ya sea en la terminal o por Telnet).
- Creamos la base de datos SIMCA.
- Una vez en el shell del servidor ejecutamos el script que contiene los comandos SQL para crear las tablas de la base de datos de la siguiente manera:

```
[pacolo@clio ~]$ psql -f  simcapostgres.sql simca
```

- Después de ejecutar el comando anterior la estructura de la base de datos ha sido creada, el siguiente paso es agregar datos a la misma.

Los datos son agregados con archivos generados a partir de los programas que generan los scripts de alimentación a la base de datos

Para ver el contenido de los scripts BDSIMCA.sql y simcapostgres.sql se sugiere consultar el **Apéndice 4.**

3.2 Programación de la Interfaz de alimentación y consultas por web

Una vez creada la base de datos es necesario realizar los programas que creen los scripts que nutran de datos a ésta. Para llevar a cabo esta tarea se utiliza el diseño estructurado el cual nos indica de forma clara cuáles son los programas a realizar.

El diseño estructurado sirve para representar en forma gráfica cuántos y cuáles son los módulos a realizar. Estos diagramas se conocen como mapa del programador.

Utilizando la carta de estructura del administrador fue posible generar el programa **lector.pl** escrito en PERL tiene como argumento de entrada el nombre de un archivo *.txt (archivo de texto plano separado por tabuladores) y genera un script *.sql (archivo de texto plano con comandos SQL) que alimenta a la base de datos SIMCA. Los archivos *.sql generados contienen comandos SQL referentes al DML.

La figura DE.1 muestra el funcionamiento de **lector.pl**.

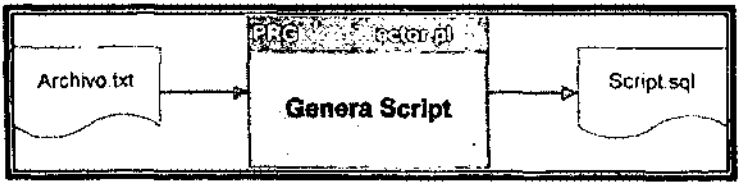


Figura DE.1 Funcionamiento de lector.pl

Pero como se tienen que procesar varios archivos de texto sobre un mismo directorio se realizó el programa **genssql.pl**, el cual llama a **lector.pl** y genera los scripts.sql de todos los archivos.txt que estén en el mismo directorio que **lector.pl** y **genssql.pl**, para ver una muestra de los archivos utilizados y generados por **genssql.pl** se sugiere consultar el **Apéndice 2** en donde se puede observar una muestra del archivo.txt de entrada y una muestra del archivo.sql de salida. La figura DE.2 muestra este proceso:

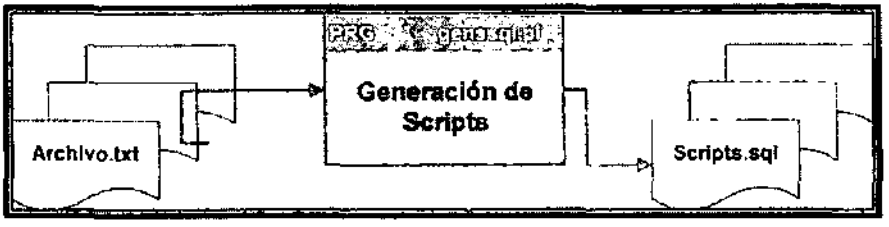


Figura DE.2 Funcionamiento de genssql.pl

Las operaciones anteriores se realizan en el servidor en donde se encuentre la base de datos simca y se resumen de la siguiente forma:

lector.pl: genera scripts.sql para cada archivo.txt. Al leer cada archivo.txt se genera un archivo temporal llamado ASF (Archivo Salida de Fechas) el cual contiene todas las fechas(DD/MM/YY:HH24) de cada archivo.txt

genssql.pl: utiliza a lector.pl para generar los scripts de todos los archivos.txt que se encuentran en el directorio, también utiliza al archivo de texto arch_txt el cual contiene la salida del comando ls *.txt; este archivo es necesario para saber cuáles son los archivos de texto que se encuentran en el directorio, a los cuales se les generará

El código fuente de los programa **lector.pl** y **genssql.pl** se encuentran en el **Apéndice 3**.

Gracias a los programas **lector.pl** y **genssql.pl** es posible generar los scripts que servirán para "alimentar" de datos a la base de datos. Este proceso es similar al realizado para la creación de la base de datos SIMCA, ya que se tienen que ejecutar los comandos SQL que vienen en los scripts generados por los programas. Para ver el proceso de alimentación a la base de datos SIMCA se sugiere consultar el **Apéndice 5** en donde se pone a disposición el manual del administrador .

A partir de la carta de estructura del usuario (ver etapa de diseño) se puede ver que se tienen que programar inicialmente tres CGI's referentes al tipo de consultas a realizar y hacer tres formularios para capturar las opciones de los usuarios del sistema.

La función principal de los CGI´s para este sistema es elaborar una consulta con las opciones recibidas de los formularios, una vez elaborada la consulta ésta se ejecuta, si se obtienen registros se formatean de acuerdo a la opción de tipo de formato seleccionado, si no se obtienen registros se manda un mensaje para informar el resultado.

El código fuente de un CGI para este sistema se encuentran en el **Apéndice 3.**

3.3 Pruebas

Para el mes de noviembre de 1998 se tiene el archivo 9811.zip con todas las mediciones de las magnitudes para este periodo. Utilizando los programas de generación de scripts antes mencionados se generó un script con 120,000 lineas lo que equivale a insertar 120,000 registros a la base de datos SIMCA en ORACLE.

Al realizar esta tarea no se presentaron problemas, pero al querer almacenar otra cantidad aproximadamente igual de registros para otro mes, el RDBMS empezó a tener problemas con sus índices y segmentos de rollbacks.

Se revisó que hubiera espacio disponible para seguir almacenando los datos, y lo que se pudo observar fue que existía el espacio de memoria necesario, pero el espacio de tablas (*Table space*) destinado para la cuenta de ORACLE que estaba utilizando se estaba saturando debido a que casi todos los usuarios utilizaban dicho espacio.

En lo que se solucionaba este problema, debido tal vez a una mala configuración, se realizó el diseño e implementación de la base de datos pero ahora para PostgreSQL y hasta el momento de escribir este documento no se han presentado problemas con dicho RDBMS.

En algunas pruebas de rendimiento se ha podido observar un desempeño mejor de PostgreSQL con respecto a ORACLE en cuanto a velocidad de acceso a la información. La ventaja que se tiene es que un RDBMS cuesta y el otro no.

Otro punto a considerar es que el usuario **nobody** del servicio de web en el servidor *alpha.dgsca.unam.mx* tiene los permisos para ejecutar por web cualquier tipo de comandos SQL ya sea del DDL o DML de ORACLE. En cambio el usuario **nobody** para el servicio de web del servidor *clio.dgsca.unam.mx* necesita la autorización para realizar las consultas que éste requiera; estos permisos solo pueden ser otorgados por el DBA de PostgreSQL.

Por estos motivos se ha estado migrando la información de ORACLE a PostgreSQL mientras esta serie de detalles de configuración sea corregida.

El procedimiento que se lleva a cabo para la alimentación de la base de datos es similar al mencionado para ORACLE. Este procedimiento se puede observar en el **Apéndice 5 Manual del administrador** en donde se explica a detalle el proceso de alimentación a la base de datos SIMCA, pero ahora para PostgreSQL.

El conjunto de pruebas que se hizo para verificar que los programas de generación de scripts (versión PostgreSQL) consistió en lo siguiente:

Creación de archivos (*.txt) de prueba para simular las mediciones realizadas en el año 2000. Suponiendo que se sigan manejando los formatos del nombre de los archivos, se creó el siguiente archivo:

```
/home/pacolo/SIMCA/PRUEBA/001003P.txt
```

Una muestra del archivo de prueba creado se muestra a continuación:

FECHA	HORA	MO3LAG	LAG	MO3TAC	TAC
1/10/00 1	0.006	0.009	0.009	0.009	0.024
1/10/00 2	0.009	0.006	0.009	0.009	0.026
1/10/00 3	0.007	0.005	0.009	0.009	0.025
1/10/00 4	0.006	0.010	0.009	0.009	0.024
1/10/00 5	0.009	0.010	0.009	0.009	0.026
1/10/00 6	0.016	0.009	0.009	0.009	0.017
1/10/00 7	0.012	0.005	0.009	0.009	0.009

La siguiente tabla es una muestra del archivo generado (001003 sql)

```
insert into O3  values ('LAG', 'O3', '1/Oct/2000','1:00', 0.006);
insert into O3  values ('TAC', 'O3', '1/Oct/2000','1:00', 0.009);
insert into O3  values ('EAC', 'O3', '1/Oct/2000','1:00', 0.009);
insert into O3  values ('SAC', 'O3', '1/Oct/2000','1:00', 0.024);
insert into O3  values ('AZC', 'O3', '1/Oct/2000','1:00', 0.016);
insert into O3  values ('TLA', 'O3', '1/Oct/2000','1:00', 0.008);
insert into O3  values ('XAL', 'O3', '1/Oct/2000','1:00', 0.010);
insert into O3  values ('MER', 'O3', '1/Oct/2000','1:00', 0.010);
```

Además de estas pruebas para identificar posibles errores con futuras fechas, se verificó que los datos fueran ingresados correctamente mediante las cifras de control mencionadas desde la etapa de análisis.

Algunas características de espacio ocupado por cada registro son mencionadas en el manual del administrador.

3.4 Liberación del sistema

Después de realizar las pruebas referentes a la parte de alimentación de la base de datos y programación de CGI's, en los directorios establecidos para tal propósito, se liberó el sistema, esto quiere decir que a nivel de desarrollo el sistema cumple con las expectativas marcadas.

El siguiente paso es cambiar todos los programas a los directorios en donde estarán funcionando permanentemente y puedan ser utilizados por todas las personas que lo requieran.

Etapa 4: **Instalación y mantenimiento**

4.1 Instalación

En esta etapa se cambiaron los programas y CGI´s de los directorios de prueba a los directorios en donde se tiene el servicio de web y cgi. Estos directorios se llaman comúnmente htdocs para el servicio de web y el cgi-bin para los CGI´s. A los programas CGI's se le dio el permiso de ejecución para que pudieran ser ejecutados desde el web y así poder conectarse a la base de datos **simca** y ejecutar la consulta.

Para poder conectarse a la base de datos desde web, fue necesario dar permiso para realizar consultas de selección al usuario **nobody.** Este usuario es el que se configura por default al instalar el servicio de web con apache.

Como se menciona en un apartado anterior cuando se estuvo trabajando con el RDBMS de Oracle, el usuario nobody podía hacer lo que quisiera por web a las bases de datos; caso contrario de lo que sucede con PosgreSQL ya que aquí al tener cuentas de administrador del RDBMS se restringió al usuario nobody a solamente poder realizar consultas de selección.

4.2 Puesta a punto

Después de hacer las actividades anteriores y las pruebas pertinentes, el sistema puede ser consultado en la dirección:

http://clio.dgsca.unam.mx/pacolo/SIMCA

Además se puso a disposición de los usuarios un formulario de contacto en donde las personas puedan sugerir el tipo de consultas que les gustaría estuvieran

en el sitio. Estas peticiones serán atendidas por el administrador del sistema y él se encargará de realizar los nuevos CGI's.

4.3 Capacitación

Para administrar el sistema se diseñó el manual del administrador, en donde se explica paso a paso los procesos que se deben llevar a cabo para agregar información a la base de datos SIMCA y mantener el sitio web. Este manual puede ser consultado el **Apéndice 5.** No existe un manual de usuario debido a que el sitio contiene ayuda para explicar como utilizar el sistema.

4.4 Mantenimiento

Esta parte del CVSI es la de mayor duración pues se tienen que hacer revisiones periódicas al sistema para verificar la pureza de la información. Además se deben agregar nuevos módulos relacionados con las consultas que los investigadores requieran y corregir posibles errores que se presenten.

4.4.1 Mantenimiento correctivo:

El **mantenimiento correctivo** se realizará cuando se detecte alguna falla en alguno de los programas, por lo que se tendrá que rehacer completamente o corregir el punto en el que se presente la falla en el programa.

Por ejemplo el programa lector.pl es el encargado de leer los archivos de texto separado por tabuladores y transformarlos en scripts con instrucciones SQL. Este programa originalmente funcionaba muy bien para el RDBMS de ORACLE pero para el de PostgreSQL, presento el error que se describe a continuación:

Para el RDBMS de ORACLE los scripts generados por lector.pl contienen instrucciones del siguiente tipo:

```
insert into O3  values ('LAG', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.005);
insert into O3  values ('TAC', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.012);
```

Como se utiliza la función TO_DATE, este formato de fecha es entendido por el RDBMS de ORACLE, pero en PostgreSQL no esta implementada esta

función, por lo que se tuvo que hacer una versión del programa lector.pl para PostgreSQL, dando por resultado que los scripts tuvieran el siguiente formato:

```
insert into O3  values ('LAG', 'O3', '1/11/1998','1:00', 0.005);
insert into O3  values ('TAC', 'O3', '1/11/1998','1:00', 0.012);
```

El problema que se presento fue que este formato de fecha hace que el RDBMS de PostgreSQL entienda que la fecha que se le esta pasando equivale al 11 de enero de 1998, pero en realidad se trata del 1° de Noviembre de 1998.

Para corregir este error fue necesario manejar los meses con letras y no con números; lo que da por resultado que el script generado tenga el siguiente formato:

```
insert into O3  values ('LAG', 'O3', '1/Nov/1998','1:00', 0.005);
insert into O3  values ('TAC', 'O3', '1/Nov/1998','1:00', 0.012);
```

Con la corrección anterior fue posible alimentar a la base de datos SIMCA creada en el RDBMS de PostgreSQL. Una muestra de estos archivos se encuentra en el **Apéndice 2**.

4.4.2 Mantenimiento adaptativo

El **mantenimiento adaptativo** consta de la creación de nuevos módulos, generalmente programas CGI's que elaboren un nuevo tipo de consultas a la base de datos SIMCA. Generalmente para realizar estos nuevos programas se modifica alguno de los CGI's ya creados, ya que todos tienen la misma estructura. La principal modificación que se hace es al tipo de consulta a la base de datos y esto equivale a modificar pocas lineas de código.

Este tipo de mantenimiento se dio al migrar de un RDBMS a otro, ya que se tuvo que adaptar la base de datos SIMCA (originalmente diseñada para trabajar en ORACLE) al RDBMS de PostgreSQL. Los cambios son significativos, para poderlo observar basta con revisar el Apéndice 4, y ver los scripts con que fue creada la base de datos **SIMCA** para ORACLE y PostgreSQL.

El script utilizado en el RDMS fue generado utilizando ER-WIN, mientras que el de PostgreSQL, fue creado escribiendo instrucción por instrucción, lo cual permitió tener un mayor control del mismo.

4.4.3 Mantenimiento preventivo

El **mantenimiento preventivo** se refiere al realizar revisiones periódicas al sistema de información mediante cifras de control y reportes. Generalmente este tipo de revisiones se hará cada vez que se agreguen nuevos datos a la base de datos SIMCA. La elección de las cifras de control fueron mencionadas desde la etapa de análisis y los reportes se obtendrán consultando la base de datos por Internet.

Capítulo

Análisis de resultados

Al finalizar el desarrollo del sistema de información se hace una evaluación de los resultados obtenidos y se compara con las metas fijadas inicialmente. En este capítulo se mencionan las conclusiones personales y se analizan los problemas que se presentaron en el desarrollo del sistema, así como también los beneficios que se obtuvieron del mismo.

Conclusiones

En el presente documento se ha podido observar la metodología que se siguió para desarrollar el sistema de consulta a la base de datos de vientos y contaminantes de la Cd. de México por Internet. Esta metodología corresponde al Ciclo de Vida de los Sistemas de Información (CVSI) y es útil -como su nombre lo indica- en el desarrollo de sistemas de información (SI), pues nos permite analizar, diseñar y desarrollar las etapas por las que atraviesa la creación de un SI, auxiliándonos con diagramas formales, como son DFD's, cartas de estructura, diagramas de tiempo, diagramas de bases de datos, etc.

El sistema desarrollado muestra la manera de obtener rápidamente información de la base de datos SIMCA por Internet. Tal vez parezca sencillo el desarrollar este tipo de aplicaciones, pero la parte fundamental y que fue la más difícil de desarrollar fue la creación de los programas que transformaran los datos manejados por RAMA (archivos DBase) a un formato manejado por los RDBMS de ORACLE y PostgreSQL, ya que se presentaron principalmente problemas relacionados con el manejo de fechas. Los programas elaborados sirven para transformar archivos de texto separado por tabuladores a scripts con instrucciones SQL correspondientes al DML. Estos programas con pequeñas modificaciones pueden servir para hacer migraciones de un RDBMS a otro, ya que la mayoría de éstos entienden el SQL estándar. Además de utilizar estos programas para migrar de DBase a ORACLE y PostgreSQL, se ha podido hacer migraciones de datos de la hoja de cálculo Excel al DBMS de MySQL; esto se logra convirtiendo a cada archivo en texto plano separado por tabuladores, lo cual permite procesarlos para ser utilizados por un DBMS en particular. Esta forma de hacer la migración de los dato resulta más rápida en comparación al uso de programas "cliente" o aplicaciones ODBC.

A grandes rasgos este sistema puede dividirse en tres partes fundamentales : la base de datos SIMCA que se apega el modelo Entidad-

Relación, los programas que transforman los archivos DBase a un formato que sea entendido por el RDBMS y los programas CGI´s que elaboran las consultas a la base de datos y dan formato a los resultados obtenidos dependiendo de las opciones seleccionadas por los usuarios del SI.

Lo anterior, como puede verse a lo largo de este documento, ha sido documentado para servir como medio de apoyo para las personas interesadas en el funcionamiento del sistema, en donde se encontrarán conceptos técnicos que se pretenden que sean entendidos por cualquier persona.

Problemas

Este proyecto atravesó por una serie de contratiempos, pero el que mayor afectó al desarrollo del mismo fue el paro en la UNAM, lo que provocó entre otras muchas cosas, retardos en la realización de actividades. Aunado a ésto, hubieron cambios de directivos en la DGSCA, lo cual trajo consigo que algunos proyectos no tuvieran la continuidad proyectada.

Pero a pesar de todo ésto, el sistema ha sido creado para que en el momento en que sea retomado el proyecto, las personas encargadas del mismo encuentren la documentación que les facilite y agilice el trabajo. Principalmente este documento ha sido elaborado para que sirva de apoyo y material de consulta para la realización de otros SI.

Logros

El principal logro fue cumplir con el objetivo planteado al inicio del proyecto, el cual fue poder brindar a los investigadores y usuarios del sistema, información específica de las mediciones hechas por las estaciones de monitoreo ambiental en un periodo de tiempo determinado.

Debido a la naturaleza del proyecto se pudo realizar una serie de comparaciones entre los RDBMS de ORACLE y PostgreSQL y analizar cuáles eran las ventajas que presentaban cada uno de ellos. Principalmente se midieron tiempos de recuperación e ingreso de datos, se investigó la forma de configurar el

acceso a la base de datos por Internet y la asignación de permisos a los usuarios. además de analizar las funciones incorporadas para el manejo de fechas.

Se aprendió a utilizar diversas herramientas CASE como ER-Win y Visio, sobre todo para la creación de diagramas (DFD, cartas de estructura, diagramas de tiempo, etc) y el modelado de los datos. Todo esto para aplicar metodologías formales de análisis, diseño y desarrollo para la creación de sistemas de información.

En lo personal este fue el primer sistema que realice para resolver un problema real, en donde pude aplicar los conocimientos adquiridos en materias relacionadas con la creación de sistemas. El mérito del desarrollo de este sistema fue que me brindó la experiencia para desarrollar otros sistemas con calidad, que es lo que principalmente se busca en la iniciativa pública y privada.

A pesar de todos los problemas y obstáculos por los que atravesó el desarrollo de este sistema han sido mucho los puntos rescatables del mismo, ya que me ha servido no sólo como experiencia profesional, sino que también, me ha brindado los elementos necesarios para participar y dirigir otros proyectos relacionados con SI.

Actualmente el auge que esta teniendo Internet hace que la mayoría de las empresas realicen una gran cantidad de transacciones por este medio, que van desde el *elearning* hasta el *ebusiness*. Para llevar a cabo esta tarea se requiere la creación de SI con calidad, en donde se apliquen las tecnologías actuales y se evalúen tanto herramientas de software comerciales y de libre distribución, lo que permitirá crear sistemas de información que sean fáciles de utilizar por el usuario final y fáciles de mantener por los administradores del mismo.

Capítulo

Apéndices

En este capítulo se concentra la información necesaria para entender a detalle el funcionamiento del sistema de información creado. En los apéndices se puede encontrar la documentación necesaria para saber cuáles son las magnitudes monitoreadas por la Red Automática de Monitoreo Ambiental y las características de cada una, también es posible ver cómo se hace la transformación de archivos mediante los programas que adecuan los datos recibidos a un formato utilizado por el RDBMS seleccionado. En el diccionario de datos se puede encontrar la documentación referente a los diagramas utilizados a lo largo de la tesis y principalmente en este capítulo se encuentra el manual del administrador del SI .

Apéndice 1

Magnitudes monitoreadas

Contaminación Atmosférica

En el presente apartado se hace un breve descripción de las magnitudes monitoreadas por la RAMA, de las unidades empleadas y del efecto que presentan en la salud del hombre

Unidades Empleadas para el Monitoreo de la Calidad del Aire.

De acuerdo a la información recibida por la R.A.M.A los parámetros que se monitorean constantemente, y de los cuales se tiene información almacenada son:

PARÁMETRO	CLAVE	UNIDAD	RED
Monóxido de Carbono	CO	PPM	MONITOREO AUTOMÁTICO
Dióxido de Azufre	SO2	PPM	
Dióxido de Nitrógeno	NO2	PPM	
Ozono	O3	PPM	
Oxido de Nitrógeno	NOX	PPM	
Ácido sulfhídrico	H_2S	PPM	
Particulas menores a 10 micras	PM-10	$\mu g/m^3$	
Temperatura	TMP	°C	MONITOREO METEOROLÓGICO
Humedad Relativa	RH	% de Hum. Rel.	
Velocidad del Viento	WSP	m/s	
Índice Ultravioleta	UVB		
Dirección del Viento	WDR	grados	

Índice Metropolitano de la Calidad del Aire (IMECA)

El índice de la calidad del aire, se define como un valor representativo de los niveles de contaminación atmosférica y sus efectos en la salud, dentro de una región determinada.

El IMECA consta de dos algoritmos de cálculo fundamentales: ***el primero, para la obtención de subíndices correspondientes a diferentes indicadores de la calidad del aire; y el segundo, para la combinación de éstos en un índice global.***

El primero involucra la utilización de funciones segmentadas basadas en dos puntos de quiebra principales. Estos puntos fueron obtenidos a partir de los criterios mexicanos de la calidad del aire, así como de niveles para los que ocurren daños significativos a la salud. Al primero se le asignó el valor de 100 y al segundo el de 500; entre estos dos puntos se definieron tres más, cuyo objetivo es clasificar el intervalo en diferentes términos descriptivos de la calidad del aire.

La función principal del IMECA es mantener informada a la población sobre la calidad del aire en la Ciudad de México, así como observar el comportamiento de los distintos contaminantes y comparar la calidad del aire entre zonas que utilicen índices similares.

IMECA	CALIDAD DEL AIRE	EFECTOS
0-100	**Satisfactoria**	Situación favorable para la realización de todo tipo de actividades
101-200	**No Satisfactoria**	Aumento de molestias menores en personas sensibles
201-300	**Mala**	Aumento de molestias e intolerancia relativa al ejercicio en personas con padecimientos respiratorios
301-500	**Muy mala**	Aparición de diversos síntomas e intolerancia al ejercicio en la población

Partes Por Millón (PPM)

Para determinar la concentración de una sustancia química en un volumen se utilizan las partes por millón. Se divide el volumen en un millón de partes iguales. Cada millonésima parte de este volumen, correspondiente a la sustancia de nuestro interés, se considera una parte por millón de la sustancia.

Las PPM se utilizan para determinar concentraciones muy pequeñas de gases en la atmósfera.

Partes Por Billón (PPB)

Para determinar la concentración de una substancia química en un volumen se utilizan las partes por billón. Se divide el volumen en un billón de partes iguales. Cada billonésima parte de este volumen, correspondiente a la substancia de nuestro interés, se considera una parte por billón de la substancia.

Las PPB se utilizan para determinar concentraciones muy pequeñas de gases en la atmósfera.

Partículas Suspendidas en su Fracción Respirable (PM-10)

Criterios para evaluar la calidad del aire

150 µg/m3 (microgramos sobre metro cúbico) en un promedio de 24 horas.

Características del contaminante

Partículas sólidas o líquidas dispersas en la atmósfera (su diámetro va de 0.3 a 10 µm) como polvo, cenizas, hollín, partículas metálicas, cemento o polen. La fracción respirable de PST, conocida como PM-10, está constituida por aquellas partículas de diámetro inferior a 10 micras, que tienen la particularidad de penetrar en el aparato respiratorio hasta los alvéolos pulmonares.

Fuentes principales

Combustión industrial y doméstica del carbón, combustóleo y diesel; procesos industriales; incendios, erosión eólica y erupciones volcánicas.

Efectos principales

- **Salud**.- Irritación en las vías respiratorias; su acumulación en los pulmones origina enfermedades como la silicosis y la asbestosis. Agravan el asma y las enfermedades cardiovasculares.
- **Materiales**.- Deterioro en materiales de construcción y otras superficies.
- **Vegetación**.- Interfieren en la fotosíntesis.
- **Otros**.- Disminuyen la visibilidad y provocan la formación de nubes.

Monóxido de Carbono

Criterios para evaluar la calidad del aire

Un promedio de 11 PPM en 8 horas.

Características del contaminante

Gas incoloro e inodoro que se combina con la hemoglobina para formar la carboxihemoglobina y puede llegar a concentraciones letales.

Fuentes principales

Combustión incompleta de hidrocarburos y sustancias que contienen carbono, tales como la gasolina, el diesel, etc.. Otra importante fuente de formación del monóxido de carbono son los incendios.

Efectos principales

Salud.- La carboxihemoglobina afecta al sistema nervioso central provocando cambios funcionales cardiacos y pulmonares, dolor de cabeza, fatiga, somnolencia, fallos respiratorios y hasta la muerte.

Ozono

Criterios para evaluar la calidad del aire

Ozono: Un promedio horario máximo de 216 µg/m3 (0.11 PPM).

Características del contaminante

Compuesto gaseoso incoloro producido en presencia de luz solar. Oxida materiales no inmediatamente oxidables por el oxigeno gaseoso.

Fuentes principales

Reacciones atmosféricas de hidrocarburos y óxidos de nitrógeno bajo la influencia de la luz solar.

Efectos principales

Salud.- Irritación de los ojos y del tracto respiratorio. Agravan las enfermedades respiratorias y cardiovasculares.

Materiales.- Deterioran el hule, los textiles y la pintura.

Vegetación.- Provocan lesiones en las hojas y limitan su crecimiento.

Otros.- Disminución de la visibilidad.

Dióxido de Nitrógeno

Criterios para evaluar la calidad del aire

Un promedio horario máximo de 395 µg/m3 (0.21 PPM).

Características del contaminante

Gas café rojizo de olor picante.

Fuentes principales

Combustión a alta temperatura en industrias y vehículos. Tormentas eléctricas.

Efectos principales

Salud.- Irrita los pulmones; agrava las enfermedades respiratorias y cardiovasculares.

Materiales.- Desteñimiento de pinturas.

Vegetación.- Caída prematura de las hojas e inhibición del crecimiento.

Otros.- Disminución la visibilidad.

Dióxido de Azufre

Criterios para evaluar la calidad del aire

Un promedio móvil de 0.13 PPM en 24 hrs.

Características del contaminante

Gas incoloro con olor picante que al oxidarse y combinarse con agua forma ácido sulfúrico, principal componente de la lluvia ácida.

Fuentes principales

Combustión de carbón, diesel, combustóleo y gasolina con azufre. Fundición de betas metálicas ricas en azufre, procesos industriales y erupciones volcánicas.

Efectos principales

- **Salud**.- Irrita los ojos y el tracto respiratorio. Reduce las funciones pulmonares y agrava las enfermedades respiratorias como el asma, la bronquitis crónica y el enfisema.
- **Materiales**.- Corroe los metales; deteriora los contactos eléctricos, el papel, los textiles, las pinturas, los materiales de construcción y los monumentos históricos.
- **Vegetación**.- Provoca lesiones en las hojas y reducción en la fotosíntesis.

Hidrocarburos

Sólo en una estación de la R.A.M.A. del Valle de México se miden hidrocarburos tales como Benceno, Tolueno y Formaldehído, para los que aún no existe una norma de calidad del aire.

Características del contaminante

Compuestos orgánicos que contienen carbono e hidrógeno en estado gaseoso. Se pueden combinar en presencia de la luz solar con óxidos de nitrógeno y participan en la formación del smog fotoquímico.

Fuentes principales

Combustión incompleta de combustibles y otras sustancias que contienen carbono. Procesamiento, distribución y uso de compuestos derivados del petróleo,

tales como la gasolina y los solventes orgánicos. Incendios, reacciones químicas en la atmósfera, y descomposición bacteriana de la materia orgánica en ausencia del oxigeno.

Efectos principales

Salud.- Trastornos en el sistema respiratorio; algunos hidrocarburos provocan el cáncer.

Apéndice 2
Transformación de archivos

Muestra del archivo 9811O3P.DBF

La siguiente tabla es una muestra obtenida del archivo 9811O3P.DBF, este archivo es guardado como texto separado por tabuladores para hacer más fácil su procesamiento.

FECHA	HORA	MO3LAG	LAG	MO3TAC	TAC	MO3EAC	EAC	MO3SAG	SAG	MO3AZC	AZC	MO3TLA	TLA	MO3XAL	XAL
01/11/98	1	0,005		0,012		0,006		0,005		0,006		0,008		0,005	
01/11/98	2	0,005		0,017		0,004		0,005		0,006		0,008		0,005	
01/11/98	3	0,005		0,013		0,007		0,004		0,006		0,008		0,004	
01/11/98	4	0,005		0,007		0,010		0,005		0,007		0,008		0,006	
01/11/98	5	0,005		0,009		0,008		0,005		0,006		0,008		0,007	
01/11/98	6	0,004		0,007		0,008		0,006		0,006		0,008		0,005	
01/11/98	7	0,004		0,006		0,008		0,005		0,006		0,009		0,008	
01/11/98	8	0,005		0,007		0,004		0,007		0,007		0,011		0,017	
01/11/98	9	0,011		0,015		0,013		0,010		0,020		0,016		0,022	
01/11/98	10	0,044		0,054		0,056		0,042		0,063		0,049		0,067	
01/11/98	11	0,109		0,126		0,116		0,091		0,114		0,075		0,093	
01/11/98	12	0,144		0,137		0,116		0,120		0,112		0,091		0,087	
01/11/98	13	0,208		0,108		0,102		0,101		0,094		0,084		0,068	
01/11/98	14	0,203		0,128		0,101		0,103		0,105		0,076		0,052	
01/11/98	15	0,107		0,113		0,084		0,091		0,085		0,063		0,056	
01/11/98	16	0,086		0,085		0,077		0,077		0,077		0,064		0,041	
01/11/98	17	0,077		0,074		0,066		0,075		0,071		0,056		0,038	
01/11/98	18	0,051		0,057		0,062		0,067		0,053		0,052		0,030	
01/11/98	19	0,013		0,039		0,058		0,040		0,030		0,026		0,018	
01/11/98	20	0,009		0,020		0,041		0,007		0,020		0,022		0,014	
01/11/98	21	0,015		0,016		0,027		0,009		0,014		0,017		0,010	
01/11/98	22	0,008		0,022		0,032		0,015		0,026		0,026		0,011	
01/11/98	23	0,011		0,009		0,018		0,026		0,013		0,016		0,012	
01/11/98	24	0,007		0,005		0,012		0,006		0,015		0,019		0,006	
02/11/98	1	0,005		0,005		0,018		0,019		0,012		0,017		0,007	
02/11/98	2	0,006		0,007		0,010		0,016		0,016		0,011		0,007	
02/11/98	3	0,005		0,012		0,009		0,013		0,013		0,015		0,005	
02/11/98	4	0,005		0,011		0,016		0,017		0,012		0,011		0,009	
02/11/98	5	0,009		0,011		0,015		0,014		0,013		0,012		0,006	
02/11/98	6	0,005		0,009		0,014		0,010		0,014		0,015		0,010	
02/11/98	7	0,009		0,006		0,008		0,005		0,009		0,008		0,006	
02/11/98	8	0,004		0,006		0,004		0,005		0,008		0,010		0,008	
02/11/98	9	0,006		0,013		0,007		0,011		0,012		0,012		0,017	
02/11/98	10	0,017		0,032		0,026		0,026		0,029		0,021		0,029	
02/11/98	11	0,049		0,058		0,067		0,066		0,057		0,041		0,041	
02/11/98	12	0,098		0,115		0,101		0,116		0,097		0,066		0,087	
02/11/98	13	0,118		0,110		0,095		0,103		0,101		0,092		0,073	
02/11/98	14	0,136		0,130		0,100		0,097		0,127		0,088		0,053	
02/11/98	15	0,166		0,139		0,074		0,093		0,122		0,065		0,050	
02/11/98	16	0,161		0,097		0,066		0,086		0,085		0,066		0,047	
02/11/98	17	0,118		0,063		0,069		0,078		0,056		0,049		0,041	
02/11/98	18	0,049		0,040		0,044		0,059		0,037		0,041		0,021	
02/11/98	19	0,009		0,014		0,011		0,025		0,015		0,021		0,011	
02/11/98	20	0,006		0,013		0,041		0,011		0,012		0,009		0,006	
02/11/98	21	0,004		0,008		0,029		0,008		0,009		0,009		0,005	
02/11/98	22	0,005		0,007		0,022		0,004		0,008		0,009		0,005	
02/11/98	23	0,005		0,007		0,020		0,005		0,007		0,009		0,009	
02/11/98	24	0,004		0,004		0,013		0,004		0,005		0,008		0,008	
03/11/98	1	0,003		0,006		0,014		0,003		0,007		0,008		0,010	
03/11/98	2	0,004		0,011		0,013		0,005		0,012		0,010		0,008	

El archivo de texto 9811O3P.txt tiene un aspecto similar al archivo anterior, solamente varía en que las divisiones para cada columna es un tabulador en lugar de las líneas que se muestran en la tabla.

Muestra del archivo 9811O3.sql

La siguiente muestra fue tomada del archivo de salida 981103.sql que es el script generado a partir del archivo de texto 9811O3P.txt con el programa lector.sql (versión ORACLE).

```
insert into O3  values ('LAG', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.005);
insert into O3  values ('TAC', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.012);
insert into O3  values ('EAC', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.006);
insert into O3  values ('SAG', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.005);
insert into O3  values ('AZC', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.006);
insert into O3  values ('TLA', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.008);
insert into O3  values ('XAL', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.005);
insert into O3  values ('MER', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.007);
insert into O3  values ('PED', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.024);
insert into O3  values ('CES', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.004);
insert into O3  values ('PLA', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.017);
insert into O3  values ('HAN', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.008);
insert into O3  values ('UIZ', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.004);
insert into O3  values ('BJU', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.006);
insert into O3  values ('TAX', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.007);
insert into O3  values ('CUA', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.022);
insert into O3  values ('TPN', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.030);
insert into O3  values ('CHA', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.005);
insert into O3  values ('TAH', 'O3', TO_DATE('1/11/1998/1','DD/MM/YYYY/HH24'), 0.021);
insert into O3  values ('LAG', 'O3', TO_DATE('1/11/1998/2','DD/MM/YYYY/HH24'), 0.005);
insert into O3  values ('TAC', 'O3', TO_DATE('1/11/1998/2','DD/MM/YYYY/HH24'), 0.017);
insert into O3  values ('EAC', 'O3', TO_DATE('1/11/1998/2','DD/MM/YYYY/HH24'), 0.004);
insert into O3  values ('SAG', 'O3', TO_DATE('1/11/1998/2','DD/MM/YYYY/HH24'), 0.005);
insert into O3  values ('AZC', 'O3', TO_DATE('1/11/1998/2','DD/MM/YYYY/HH24'), 0.006);
insert into O3  values ('TLA', 'O3', TO_DATE('1/11/1998/2','DD/MM/YYYY/HH24'), 0.008);
insert into O3  values ('XAL', 'O3', TO_DATE('1/11/1998/2','DD/MM/YYYY/HH24'), 0.005);
```

Muestra del archivo 9811O3.sql

La siguiente muestra fue tomada del archivo de salida 981103.sql que es el script generado a partir del archivo de texto 9811O3P.txt con el programa lector.sql (versión para Postgresql).

```
insert into O3  values ('LAG', 'O3', '1/Nov/1998','1:00', 0.005);
insert into O3  values ('TAC', 'O3', '1/Nov/1998','1:00', 0.012);
insert into O3  values ('EAC', 'O3', '1/Nov/1998','1:00', 0.006);
insert into O3  values ('SAG', 'O3', '1/Nov/1998','1:00', 0.005);
insert into O3  values ('AZC', 'O3', '1/Nov/1998','1:00', 0.006);
insert into O3  values ('TLA', 'O3', '1/Nov/1998','1:00', 0.008);
insert into O3  values ('XAL', 'O3', '1/Nov/1998','1:00', 0.005);
insert into O3  values ('MER', 'O3', '1/Nov/1998','1:00', 0.007);
insert into O3  values ('PED', 'O3', '1/Nov/1998','1:00', 0.024);
insert into O3  values ('CES', 'O3', '1/Nov/1998','1:00', 0.004);
insert into O3  values ('PLA', 'O3', '1/Nov/1998','1:00', 0.017);
insert into O3  values ('HAN', 'O3', '1/Nov/1998','1:00', 0.008);
insert into O3  values ('UIZ', 'O3', '1/Nov/1998','1:00', 0.004);
insert into O3  values ('BJU', 'O3', '1/Nov/1998','1:00', 0.006);
insert into O3  values ('TAX', 'O3', '1/Nov/1998','1:00', 0.007);
insert into O3  values ('CUA', 'O3', '1/Nov/1998','1:00', 0.022);
insert into O3  values ('TPN', 'O3', '1/Nov/1998','1:00', 0.030);
insert into O3  values ('CHA', 'O3', '1/Nov/1998','1:00', 0.005);
insert into O3  values ('TAH', 'O3', '1/Nov/1998','1:00', 0.021);
insert into O3  values ('LAG', 'O3', '1/Nov/1998','2:00', 0.005);
insert into O3  values ('TAC', 'O3', '1/Nov/1998','2:00', 0.017);
insert into O3  values ('EAC', 'O3', '1/Nov/1998','2:00', 0.004);
*
*
insert into O3  values ('TAX', 'O3', '30/Nov/1998','23:00', 0.004);
insert into O3  values ('CUA', 'O3', '30/Nov/1998','23:00', 0.021);
insert into O3  values ('TPN', 'O3', '30/Nov/1998','23:00', 0.011);
insert into O3  values ('CHA', 'O3', '30/Nov/1998','23:00', 0.007);
insert into O3  values ('TAH', 'O3', '30/Nov/1998','23:00', 0.005);
insert into O3  values ('LAG', 'O3', '1/Dec/1998','0:00', 0.012);
insert into O3  values ('TAC', 'O3', '1/Dec/1998','0:00', 0.014);
insert into O3  values ('EAC', 'O3', '1/Dec/1998','0:00', 0.021);
insert into O3  values ('SAG', 'O3', '1/Dec/1998','0:00', 0.023);
insert into O3  values ('AZC', 'O3', '1/Dec/1998','0:00', 0.015);
insert into O3  values ('TLA', 'O3', '1/Dec/1998','0:00', 0.011);
insert into O3  values ('XAL', 'O3', '1/Dec/1998','0:00', 0.013);
insert into O3  values ('MER', 'O3', '1/Dec/1998','0:00', 0.007);
insert into O3  values ('PED', 'O3', '1/Dec/1998','0:00', 0.017);
insert into O3  values ('CES', 'O3', '1/Dec/1998','0:00', 0.007);
insert into O3  values ('PLA', 'O3', '1/Dec/1998','0:00', 0.012);
insert into O3  values ('HAN', 'O3', '1/Dec/1998','0:00', 0.007);
insert into O3  values ('UIZ', 'O3', '1/Dec/1998','0:00', 0.007);
insert into O3  values ('BJU', 'O3', '1/Dec/1998','0:00', 0.012);
insert into O3  values ('TAX', 'O3', '1/Dec/1998','0:00', 0.005);
insert into O3  values ('CUA', 'O3', '1/Dec/1998','0:00', 0.022);
insert into O3  values ('TPN', 'O3', '1/Dec/1998','0:00', 0.012);
insert into O3  values ('CHA', 'O3', '1/Dec/1998','0:00', 0.006);
insert into O3  values ('TAH', 'O3', '1/Dec/1998','0:00', 0.014);
```

Apéndice 3

Código fuente y archivos generados

Código del programa genssql.pl

```
#!/usr/bin/perl


#Programa elaborado por:
#                       Francisco Lorenzana Zarco
#                       pacolo2@yahoo.com    pacolo@servidor.unam.mx
# Este programa ejecuta el programa lector.pl dándole como argumentos
# los archivos de texto que se encuentren en el directorio
# 28 agosto 1999

$comando="ls *.txt > arch_txt";
system($comando);
open(AE,"<arch_txt");
while($linea=<AE>)
  {
   system("lector $linea");
  }
close AE;
```

Código del programa lector.pl (Versión para ORACLE)

```
#!/usr/bin/perl

#Programa elaborado por:
#                       Francisco Lorenzana Zarco
#                       pacolo2@yahoo.com    pacolo@servidor.unam.mx
# Este programa genera scripts sql a partir de archivos de texto .txt
# v 1.0 01 agosto 1999

sub nombre_medida($)
{#Esta rutina lo que hace es obtener WSP del nombre del archivo 9811WSPP.txt
  $fname = shift @_;
  # Nombre del archivo (shift @_ tiene el valor del parametro que se le
manda a la rutina
  @fname = split('',$fname);
  #$fname es una cadena que mediante la funcion split la convierte en @fname
(arreglo de caracteres)
  $longitud=@fname;
  if($longitud eq '11')
    {$medida=$fname[4].$fname[5]; # $medida tiene el nombre de la medida ejm
WSP
     $long_medida=2;}
  if($longitud eq '12')
    {$medida=$fname[4].$fname[5].$fname[6];
```

```
    $long_medida=3;}
}

sub claves($)
{
  $fila1 = shift @_;
  #fila 1 nombre de las claves
  #Ejemplo de la la fila del archivo FECHA      HORA  MWSPTAC     TAC
     MWSPEAC     EAC
  @columnas=split(/\t/,$fila1);
  #En el arreglo columnas se tiene cada valor de cada columna
  $total=@columnas;

  $indice=1;          #Para formar las claves
  $columnas2[0]=$columnas[0].$medida;
  #Se concatena FECHA con el nombre de la medida ejm FECHAWSP

  if($medida eq 'PM1')
    {$long_medida=4;}
    # Porque en las columnas del archivo.txt aparece MPM10estacion ejm
MPM10TLA Nota el problema es el 0
  for($i=0;$i<$total;$i++)
     {
      if( $columnas[$i] ne "MER" && $columnas[$i] ne "MIN" &&
$columnas[$i]=~m/^\M/)
        {
         if($long_medida=='2')
         {$columnas[$i]=~s/^\M..//g;}
       if($long_medida=='3')
         {$columnas[$i]=~s/^\M...//g;}
         if($long_medida=='4')
         {$columnas[$i]=~s/^\M....//g;}
         # Ejemplo La columna MPM10MER . Esta instruccion sustituye
         # MPM10 y deja MER, MER lo almacena en $columnas[$i]
         $columnas2[$indice]=$columnas[$i];
       # $columnas2[$indice] va a tener FECHA y el nombre de cada estacion
ejm MER, TAC, etc
         $indice++;
         }
       }
  #$lm =@columnas2;
  #print "$lm $indice \n";
}

sub leefecha (@)
{
 open(ASF,"+>ASF");
# ASF (Archivo de Salida de Fechas) Este archivo tiene todas las fechas de
cada archivo.txt
## Para agregarle 19 al año ya que ORACLE maneja 4 digitos en fecha y en los
arch.txt el año es de 2 digitos
  $bandera=0;
  while($reng=shift @_)
   {
    @medidas=split(/\t/,$reng);
```

```
    #Ejm $reng = "1/11/98     1     64,00" entonces $medidas[0]="1/11/98"
$medidas[1]="1" $medidas[2]="64,00"
    @fecha_parcial=split('/',$medidas[0]);
    # Como la fecha es de la forma 1/11/98  $fecha_parcial[0]="1"
$fecha_parcial[1]="11" $fecha_parcial[2]="98"
    if($fecha_parcial[2]<=99)

{$medidas[0]=$fecha_parcial[0].'/'.$fecha_parcial[1].'/19'.$fecha_parcial[2]
;}
      # Aqui $medidas[0]="1/11/1998"
    else{

$medidas[0]=$fecha_parcial[0].'/'.$fecha_parcial[1].'/20'.$fecha_parcial[2];
} #Para Y2K
##Aqui termina lo del año

## Para sustituir el 24 por 0 ya que las horas en ORACLE van  de 0 a 23
    if($bandera=='1')
      {$fecha=$medidas[0].'/0';
       print ASF "$fecha\n";
       $bandera=0;}
    if($medidas[1]=='24')
       {$bandera=1;}
    if($bandera=='0')
       {$fecha=$medidas[0].'/'.$medidas[1];
        print ASF "$fecha\n";}
   }

## aqui terminan las horas

### Para el ultimo renglon en el que ya no se corrige la fecha porque se
enciende la bandera=1
@date=split('/',$fecha);
#$fecha es el ultimo renglon del archivo.txt y ya no se corrige porque
termina el while y  bandera=1
$month=$date[1]+1;

  if($month>12)
   {$month=1;
    $date[2]=$date[2]+1;}
$fecha='1/'.$month.'/'.$date[2].'/0';
print ASF "$fecha\n";
###
close ASF;
}

############  Programa principal ##################
#Se abre una vez el archivo AE  para poder generar el archivo de fechas ASF
y generar las claves
open(AE,"<$ARGV[0]");

nombre_medida ($ARGV[0]); # Obtiene nombre de la medida del archivo.txt ejm
WSP de (9811WSPP.txt)
claves (<AE>); # Obtiene la clave de las estaciones de monitoreo ejm MER,
TAC, PLA, etc.
```

```
leefecha  (<AE>); # En el archivo ASF se tienen las fechas de todo el
archivo ejm.   1/11/98  1 lo convierte en 1/11/1998/1
close AE; #Cuando se llama a la funcion claves se lee la la fila y el
apuntador de archivo
       #se queda en la 2a, por eso cuando se llama a leefecha empieza a
leer desde la 2a fila


open(AE,"<$fname"); #$fname tiene el nombre del archivo
open(ASF,"<ASF");
open(AS,">$medida.sql");


print "Procesando el archivo $medida para las estaciones:\n @columnas2\n";

$renglon=<AE>;

while($renglon=<AE>)
{
 $fecha=<ASF>;
 chop($fecha);
 @valores=split(/\t/,$renglon);
 $num_col=@valores;
   for($i=1;$i<$num_col/2;$i++)
      {
       $ind=2*$i;
       if($valores[$ind] ne "")
         {
         $valores[$ind]=~s/\,/\./g;
       $valor=$valores[$ind];
       if($valor < 0) #Si hay valores negativos se convierten a espacios en
blanco
       {$valores[$ind]="NULL";}#Al parecer aqui voy a agregar en lugar de ""
NULL
         #chop($valores[$ind]); #Para quitarle el retorno de carro de
algunos valores.
         #Lo habia dejado como comentario porque si no encuentra el retorno
de carro quita el
         #ultimo caracter.
         print AS "insert into $medida  values ('$columnas2[$i]', '$medida',
TO_DATE('$fecha','DD/MM/YYYY/HH24'), $valores[$ind]); \n";
         }
      }
}

close AE;
close ASF;
close AS.sql;
```

Código del programa lector.pl (Versión para Postgresql)

```
#!/usr/bin/perl
```

```
#Programa elaborado por:
#                    Francisco Lorenzana Zarco
#                    pacolo2@yahoo.com    pacolo@servidor.unam.mx
# Este programa genera scripts sql a partir de archivos de texto .txt
# v 2.2 para Postgresql 06 Julio 2000

sub nombre_medida($)
{#Esta rutina lo que hace es obtener WSP del nombre del archivo 9811WSPP.txt
  $fname = shift @_;
  # Nombre del archivo (shift @_ tiene el valor del parametro que se le
manda a la rutina
  @fname = split('',$fname);
  #$fname es una cadena, mediante la funcion split la convierte en @fname
(arreglo de caracteres)
  $longitud=@fname;
  $fecha_medida=$fname[0].$fname[1].$fname[2].$fname[3]; # $fecha_medida
tiene el valor 9811
  if($longitud eq '11')
    {$medida=$fname[4].$fname[5];
     $long_medida=2;}
  if($longitud eq '12')
    {$medida=$fname[4].$fname[5].$fname[6]; # $medida tiene el nombre de la
medida ejm WSP
     $long_medida=3;}
}

sub claves($)
{
  $fila1 = shift @_;
  #fila 1 nombre de las claves
  #Ejemplo de la la fila del archivo FECHA     HORA  MWSPTAC     TAC
     MWSPEAC     EAC
  @columnas=split(/\t/,$fila1);
  #En el arreglo columnas se tiene cada valor de cada columna
  $total=@columnas;

  $indice=1;          #Para formar las claves
  $columnas2[0]=$columnas[0].$medida;
  # Se concatena FECHA con el nombre de la medida ejm FECHAWSP
  # $columnas2[1]=$columnas[1].$medida;
  # Se concatena HORA con el nombre de la medida ejm FECHAWSP

  if($medida eq 'PM1')
    {$long_medida=4;}
    # Porque en las columnas del archivo.txt aparece MPM10estacion ejm
MPM10TLA Nota el problema es el 0
  for($i=0;$i<$total;$i++)
     {
      if( $columnas[$i] ne "MER" && $columnas[$i] ne "MIN" &&
$columnas[$i]=~m/^\M/)
        {
         if($long_medida=='2')
         {$columnas[$i]=~s/^\M..//g;}
       if($long_medida=='3')
```

```
          {$columnas[$i]=~s/^\M.../g;}
          if($long_medida=='4')
          {$columnas[$i]=~s/^\M..../g;}
          # Ejemplo La columna MPM10MER . Esta instruccion sustituye
          # MPM10 y deja MER, MER lo almacena en $columnas[$i]
          $columnas2[$indice]=$columnas[$i];
        # $columnas2{$indice} va a tener FECHA y el nombre de cada estacion
ejm MER, TAC, etc
          $indice++;
          }
        }
  #$lm =@columnas2;
  #print "$lm $indice \n";
}

sub leefecha {@}
{
 open(ASF,"+>ASF");
# ASF (Archivo de Salida de Fechas) Este archivo tiene todas las fechas de
cada archivo.txt
## Para agregarle 19 al año ya que ORACLE maneja 4 digitos en fecha y en los
arch.txt el año es de 2 digitos
  $bandera=0;
  while($reng=shift @_)
   {
    @medidas=split(/\t/,$reng);
    # Ejm $reng = "1/11/98    1     64,00"
    # entonces $medidas[0]="1/11/98" $medidas[1]="1" $medidas[2]="64,00"
    @fecha_parcial=split('/',$medidas[0]);
    # Como la fecha es de la forma 1/11/98
    # $fecha_parcial[0]="1" $fecha_parcial[1]="11" $fecha_parcial[2]="98"

    # Postgresql se confunde al insertar la fecha 1/11/98 ya que no sabe si
es
    # el 1/Nov/98 o Jan/11/98 por eso es mejor convertir la fecha al formato
dd/Month/yyyy
    if($fecha_parcial[1]=='1')
       {$fecha_parcial[1]='Jan';
        $sig_mes='Feb';}
    elsif($fecha_parcial[1]=='2')
       {$fecha_parcial[1]='Feb';
        $sig_mes='Mar';}
    elsif($fecha_parcial[1]=='3')
       {$fecha_parcial[1]='Mar';
        $sig_mes='Apr';}
    elsif($fecha_parcial[1]=='4')
       {$fecha_parcial[1]='Apr';
        $sig_mes='May';}
    elsif($fecha_parcial[1]=='5')
       {$fecha_parcial[1]='May';
        $sig_mes='Jun';}
    elsif($fecha_parcial[1]=='6')
       {$fecha_parcial[1]='Jun';
        $sig_mes='Jul';}
    elsif($fecha_parcial[1]=='7')
```

```
       {$fecha_parcial[1]='Jul';
        $sig_mes='Aug';}
    elsif($fecha_parcial[1]=='8')
       {$fecha_parcial[1]='Aug';
        $sig_mes='Sep';}
    elsif($fecha_parcial[1]=='9')
       {$fecha_parcial[1]='Sep';
        $sig_mes='Oct';}
    elsif($fecha_parcial[1]=='10')
       {$fecha_parcial[1]='Oct';
        $sig_mes='Nov';}
    elsif($fecha_parcial[1]=='11')
       {$fecha_parcial[1]='Nov';
        $sig_mes='Dec';}
    elsif($fecha_parcial[1]=='12')
       {$fecha_parcial[1]='Dec';
        $sig_mes='Jan';}

    if($fecha_parcial[2]<=99 and $fecha_parcial[2]>0)

{$medidas[0]=$fecha_parcial[0].'/'.$fecha_parcial[1].'/19'.$fecha_parcial[2]
;}
      # Aqui $medidas[0]="1/11/1998" se le agrega 19 para manejar 4 digitos
en la fecha
    else{

$medidas[0]=$fecha_parcial[0].'/'.$fecha_parcial[1].'/20'.$fecha_parcial[2];
} #Para Y2K
##Aqui termina lo del año

## Para sustituir el 24 por 0 ya que las horas en ORACLE y Postgresql van
de 0 a 23
    if($bandera=='1')
      {$fecha=$medidas[0]."','0:00 ";
       print ASF "$fecha\n";
       $bandera=0;}
    if($medidas[1]=='24')
       {$bandera=1;}
    if($bandera=='0')
       {$fecha=$medidas[0]."','".$medidas[1].":00";
        print ASF "$fecha\n";}
   }

## aqui terminan las horas

### Para el ultimo renglon en el que ya no se corrige la fecha porque se
enciende la bandera=1
@date=split('/',$fecha);
#$fecha es el ultimo renglon del archivo.txt y ya no se corrige porque
termina el while y  bandera=1
$month=$sig_mes;
@year=split("'",$date[2]); #$fecha tiene la forma 31/10/1998','23:00 y
$date[2] tiene la forma 1998','23:00
  if($month eq 'Jan')
   {
```

```
    $year[0]=$year[0]+1;}
$fecha='1/'.$month.'/'.$year[0]."','0:00";
print ASF "$fecha\n";
###
close ASF;
}

############  Programa principal ##################
#Se abre una vez el archivo AE  para poder generar el archivo de fechas ASF
y generar las claves
open(AE,"<$ARGV[0]");

nombre_medida ($ARGV[0]); # Obtiene nombre de la medida del archivo.txt ejm
WSP de (9811WSPP.txt)
claves (<AE>); # Obtiene la clave de las estaciones de monitoreo ejm MER,
TAC, PLA, etc.
leefecha  (<AE>); # En el archivo ASF se tienen las fechas de todo el
archivo ejm.   1/11/98  1 lo convierte en 1/11/1998/1
close AE; #Cuando se llama a la funcion claves se lee la la fila y el
apuntador de archivo
       #se queda en la 2a, por eso cuando se llama a leefecha empieza a
leer desde la 2a fila

$salida=$fecha_medida.$medida;

open(AE,"<$fname"); #$fname tiene el nombre del archivo
open(ASF,"<ASF");
open(AS,">$salida.sql");


print "Procesando el archivo $medida para las estaciones:\n @columnas2\n";

$renglon=<AE>;

while($renglon=<AE>)
{
 $fecha=<ASF>;
 chop($fecha);
 @valores=split(/\t/,$renglon);
 $num_col=@valores;
   for($i=1;$i<$num_col/2;$i++)
      {
       $ind=2*$i;
       if($valores[$ind] ne "")
         {
         $valores[$ind]=~s/\,/\./g;
       $valor=$valores[$ind];
       if($valor < 0) #Si hay valores negativos se convierten a espacios en
blanco
       {$valores[$ind]="NULL";}#Al parecer aqui voy a agregar en lugar de ""
NULL
         #chop($valores[$ind]); #Para quitarle el retorno de carro de
algunos valores.
         #Lo habia dejado como comentario porque si no encuentra el retorno
de carro quita el
```

```
            #ultimo caracter.
            print AS "insert into $medida  values ('$columnas2[$i]', '$medida',
'$fecha', $valores[$ind]); \n";
            }
      }
}

close AE;
close ASF;
close AS.sql;
```

Contenido del archivo ASF

```
1/Oct/1999','1:00
1/Oct/1999','2:00
1/Oct/1999','3:00
1/Oct/1999','4:00
1/Oct/1999','5:00
1/Oct/1999','6:00
1/Oct/1999','7:00
1/Oct/1999','8:00
1/Oct/1999','9:00
1/Oct/1999','10:00
1/Oct/1999','11:00
1/Oct/1999','12:00
1/Oct/1999','13:00
1/Oct/1999','14:00
1/Oct/1999','15:00
1/Oct/1999','16:00
1/Oct/1999','17:00
1/Oct/1999','18:00
1/Oct/1999','19:00
1/Oct/1999','20:00
1/Oct/1999','21:00
1/Oct/1999','22:00
1/Oct/1999','23:00
2/Oct/1999','0:00
2/Oct/1999','1:00
2/Oct/1999','2:00
2/Oct/1999','3:00
2/Oct/1999','4:00
      *
      *
      *
      *
      *
      *
31/Oct/1999','20:00
31/Oct/1999','21:00
31/Oct/1999','22:00
31/Oct/1999','23:00
```

```
1/Nov/1999','0:00
```

*Nota: Este archivo es generado para poder manejar las fechas en un formato que sea entendido por los DBMS, ya que el formato en el que es recibido no se adapta a las DBMS modernos.

Contenido del archivo arch_txt

```
001003P.txt
991003P.txt
```

*Nota: Este archivo contiene el listado de todos los archivos de texto en el directorio en el que se encuentren los programas gennssql.pl y lector.pl

Código del CGI que realiza la consulta "Espacio" a la base de datos SIMCA

```
#!/usr/bin/perl
#CGI para buscar datos en simca
#Francisco Lorenzana Zarco
#3-Noviembre-1999
#Ver 1


###Declaracion de funciones

sub formato_html()
{

    print "Content-type: text/html\n\n";
    print "<html> <head><title>Consulta7</title></head><body> \n";


for($n=1;$n<=12;$n++)
{
$clave="cvmed".$n;
$cvmed=$vid->param($clave);
  if ($cvmed ne "")
    {
    $fmed="FECHA".$cvmed;
    $nmed="MEDICION".$cvmed;

    $consulta="SELECT CVEESTACION, TO_CHAR ($fmed,'DAY/MM/YYYY:HH24'), $nmed
             FROM $cvmed
             WHERE TO_CHAR ($fmed,'D') >= '$dias_semana{$d1}'
               AND TO_CHAR ($fmed,'D') <= '$dias_semana{$d2}'
             AND  TO_CHAR  ( $fmed,'HH24' ) >= '$h'
             AND  TO_CHAR  ( $fmed,'HH24' ) <= '$h2'
             $est ";


    print "<center><table BORDER CELLSPACING=0 CELLPADDING=0 >\n";

    #Realizacion del select
```

```
    $dbh = DBI->connect("dbi:Oracle:","system","manager")
           || die("No me puedo conectar a la BD SIMCA</BODY></HTML>\n");
    $sth= $dbh->prepare($consulta)
           || die("<h1>Error de sintaxis</h1> </BODY></HTML>\n");
    $sth->execute
           || die("<h1>Error de ejecucion del comando</h1>
</BODY></HTML>\n");

    print "La consulta se hizo de  <b> $d a $d2 </b>, en la hora  <b>$h a
$h2</b> para la medicion <b>$cvmed</b> de acuerdo a su seleccion son:";
    print "<tr ALIGN=CENTER BGCOLOR=#7CC4F2> <td> Clave </td> <td> <b> Fecha
</td> <td> Medicion</td> <td> </td> </tr> \n";
    $cr=0;
    while(@res  = $sth->fetchrow_array)
    {
      $cr++;
      print "<tr ALIGN=CENTER BGCOLOR=#C6CEEC> <td> ",$res[0],"</td> <td>
<b>",$res[1],"</td><td>",$res[2],"</td> <td>",$res[3],"</td> </tr> \n";
    }
    $sth->finish;

    if($cr==0)
    {
      print "<tr><td><h1>No hubo renglones que concuerden con el criterio
especificado</h1></td></tr>\n";
    }


    print "</table></center>\n";


    }
}

# print " <br>$consulta ::";
print "</body></html> \n";

}

sub formato_txt {}
{

  print $vid->header;

for($n=1;$n<=12;$n++)
{
$clave="cvmed".$n;
$cvmed=$vid->param($clave);
  if ($cvmed ne "")
    {
    $fmed="FECHA".$cvmed;
    $nmed="MEDICION".$cvmed;

    $consulta="SELECT CVEESTACION, TO_CHAR ($fmed,'DAY/MM/YYYY:HH24'), $nmed
            FROM $cvmed
```

```
            WHERE TO_CHAR ($fmed,'D') >= '$dias_semana{$d}'
              AND TO_CHAR ($fmed,'D') <= '$dias_semana{$d2}'
            AND  TO_CHAR  ( $fmed,'HH24' ) >= '$h'
            AND  TO_CHAR  ( $fmed,'HH24' ) <= '$h2'";


    print "\n";

    #Realizacion del select
    $dbh = DBI->connect("dbi:Oracle:","system","manager")
           || die("No me puedo conectar a la BD SIMCA \n");
    $sth= $dbh->prepare($consulta)
           || die(" Error de sintaxis</h1> \n");
    $sth->execute
           || die(" Error de ejecucion del comando \n");

    print "La consulta se hizo de $d a $d2 , en la hora  $h a $h2  para la
medicion $cvmed de acuerdo a su seleccion son: \n";

    $cr=0;
    while(@res  = $sth->fetchrow_array)
    {
      $cr++;
      print  $res[0]."\t".$res[1]."\t".$res[2]."\t".$res[3]."\n";
    }
    $sth->finish;

    if($cr==0)
    {
      print "No hubo renglones que concuerden con el criterio
especificado\n";
    }


    print "\n";


    }
}

# print " $consulta :: \n";


 }

### Fin de las funciones

use CGI;
use DBI;

$vid=new CGI;

%dias_semana=(
       "Domingo",   "1",
       "Lunes","2",
```

```
        "Martes",   "3",
        "Miercoles",   "4",
        "Jueves",    "5",
        "Viernes",   "6",
        "Sabado",   "7",
             );

 %meses=(
        "Enero",   "Jan",
        "Febrero","Feb",
        "Marzo",   "Mar",
        "Abril",   "Apr",
        "Mayo",    "May",
        "Junio",   "Jun",
        "Julio",   "Jul",
        "Agosto",  "Aug",
        "Septiembre","Sep",
        "Octubre",   "Oct",
        "Noviembre","Nov",
        "Diciembre","Dec"
             );
%estacion=(
      "Vallejo","VAL",
      "Tacuba","TAC",
      "ENEP Acatlan","EAC",
      "Azcapotzalco","AZC",
      "Tlanepantla", "TLA",
      "Instituto Mexicano del Petroleo", "IMP",
      "Cuitlahuac", "CUI",
      "Tultitlan", "TLI",
      "Atizapan", "ATI",
      "Los Laureles", "LLA",
      "La Presa", "LPR",
      "La Villa", "LVI",
      "San Agustin", "SAG",
      "Xalostoc", "XAL",
      "Aragon", "ARA",
      "Netzahualcoyotl", "NET",
      "Chapingo", "CHA",
      "Villa de las Flores", "VIF",
      "Lagunilla", "LAG",
      "Merced", "MER",
      "Hangares", "HAN",
      "Benito Juarez", "BJU",
      "Insurgentes", "MIN",
      "Palacio Legislativo", "PL",
      "Santa Ursula", "SUR",
      "Pedregal", "PED",
      "Plateros", "PLA",
      "Cuajimalpa", "CUA",
      "Tlalpan", "TPN",
      "Cerro de la Estrella", "CES",
      "UAM Iztapalapa", "UIZ",
      "Taxqueña", "TAX",
      "Tlahuac","TAH",
```

```
        );

#Variables de ambiente para Oracle:
$ENV{ORACLE_SID}="ora8";
$ENV{ORACLE_HOME}="/robot/oracle/oracle8";
$ENV{USER}="oracle8";

$d=$vid->param('dia');
$h=$vid->param('hora');
$m=$vid->param('mes');
$a=$vid->param('year');

$d2=$vid->param('dia2');
$h2=$vid->param('hora2');
$m2=$vid->param('mes2');
$a2=$vid->param('year2');

$cvest=$vid->param('cvest');
$est="";
if( $cvest ne 'Todas.')
 {
  $est="AND CveEstacion='$estacion{$cvest}'";
  }

$formato=$vid->param('formato');

if($formato eq 'HTML')
 {
  formato_html();
  }
elsif($formato eq 'TXT')
 {
  formato_txt();
  }
```

Apéndice 4
Diccionario de datos

Diagrama de flujo de datos USUARIO

Entidades externas

Identificador: **USU**
Nombre: **Usuario**
Descripción: *El usuario es la persona que va a realizar las consultas a la base de datos a través de la Internet. Mediante formas HTML seleccionará las opciones que él desee para elaborar su consulta, estas opciones comprenden rangos de fechas, selección de estaciones, valores específicos y formato de salida de la información.*

Procesos

Identificador: **1**
Nombre: **Seleccionar consulta**
Descripción: *En este proceso el usuario seleccionará de un menú el tipo de consulta que desee realizar. Este menú estará en una página web y al seleccionar el tipo de consulta a realizar le aparecerá otra página en donde especifique el criterio de la búsqueda. En este proceso se deberá realizar una serie de páginas web principalmente que contengan formas; en éstas se le presentará al usuario las opciones posibles para elaborar su consulta y una vez seleccionadas las opciones se enviarán a programas CGIs para ser procesada la información.*

Identificador: **2**
Nombre: **Elaborar consulta**
Descripción: *Mediante programas CGIs escritos en PERL o PHP3 se procesará la información recibida de las formas html. En este proceso se verificará que los datos vengan completos, es decir que no haya faltado seleccionar alguna casilla en la forma html , ya que de lo contrario no se podrá elaborar dicha consulta. La consulta a elaborar seguirá el formato estándar de SQL y principalmente se elaborará un comando SELECT con las opciones seleccionadas en la froma HTML.*

Identificador: **3**
Nombre: **Ejecutar consulta**
Descripción: *Una vez formada la consulta (SELECT) el CGI se conectará a la base de datos (SIMCA) para obtener los registros que cumplan con el criterio especificado.*

Identificador: **4**
Nombre: **Dar formato a los datos de salida**
Descripción: *De acuerdo al formato de salida de los datos seleccionado, los datos serán mostrados ya sea en formato texto separado por tabuladores o en formato HTML. Este proceso de dar formato a los datos se realiza en el servidor en donde se encuentra instalada la base de datos, lográndose así el ahorro de tiempo para procesar los datos por parte del usuario con algún programa específico para manejar archivos de texto.*

Almacenamiento de datos

Id : **04**
Nombre : **Simca**
Descripción: *Este almacenamiento se refiere a la base de datos y se llamará* ***simca*** *(**sim**ulación de **c**ontaminantes **a**tmosféricos), la cual contendrá todos los datos referentes a las mediciones realizadas por las estaciones de monitoreo de la RAMA.*

Flujo de datos

Id : **Opción**
Origen : **USU**
Destino : **1**
Tipo : Evento clic
Descripción : Mediante la selección de una opción el usuario del sistema seleccionará el tipo de consulta a realizar.

Id : **Especificaciones**
Origen : **1**
Destino : **2**
Tipo : Cadena de caracteres
Descripción : De acuerdo a las opciones que haya seleccionado el usuario en un formulario HTML se enviaran éstas a un programa CGI escrito en PERL para procesar la información.

Id : **Consulta**
Origen : **2**
Destino : **3**
Tipo : Cadena de caracteres
Descripción : Con las opciones seleccionadas en el formulario HTML, se elabora la consulta la cual asemeja al siguiente comando SQL: `SELECT CveMedida, MedicionO3 FROM O3`, una vez elaborada se ejecuta y se obtienen los datos para presentarlos al usuario en el formato que haya seleccionado ya sea HTML o texto plano.

Id : **resultado**
Origen : **04**
Destino : **4**
Tipo : Arreglo tipo char
Descripción : Al ejecutar la consulta se hace una operación de lectura a la base de datos SIMCA; los resultados obtenidos se almacenan en un arreglo tipo char, para después darle formato de salida ya sea de texto o HTML.

Id : **Resultados procesados**
Origen : **4**
Destino : **USU**
Tipo : HTML o texto plano
Descripción : De acuerdo a la selección de formato de salida, se presentarán los datos obtenidos de la consulta en formato texto o HTML.

Diagrama de flujo de datos Administrador

Entidades externas

Identificador: **ADM**
Nombre: **Administrador**
Descripción: *El administrador tiene la tarea de:*

- *Analizar la información recibida de la RAMA.*
- *Convertir el formato de los datos recibidos a formato de texto separado por tabuladores.*
- *Mandar los archivos de texto al servidor*
- *Convertir los archivos de texto a scripts con instrucciones SQL*
- *Agregar los datos a la base de datos*
- *Mediante cifras de control verificará que los datos contenidos en la BD sean los correctos*
- *Crear la interfaz para que la base de datos pueda ser consultada por Internet*
- *Estará pendiente del tipo de consultas que requieran los investigadores del fenómeno de los contaminantes atmosféricos*

Procesos

Identificador: **1**
Nombre: **Analizar formato de datos**
Descripción: *En este proceso el administrador analizará los datos recibidos de la RAMA (archivos *.DBF) y descartará todos los datos que no hayan sido considerados en el diseño de la base de datos SIMCA.*

Identificador: **2**
Nombre: **Cambiar el formato de los datos**
Descripción: *Utilizando Excel, los archivos *.DBF serán transformados a texto separado por tabuladores ya que así es más fácil procesarlos con los programas escritos en PERL.*

Identificador: **3**
Nombre: **Transferir archivos al servidor**
Descripción: *Los archivos de texto separado por tabuladores (*.txt) serán enviados vía FTP al servidor en donde serán transformados en scripts con instrucciones SQL.*

Identificador: **4**
Nombre: **Crear scripts para el RDBMS**
Descripción: *Con los programas escritos en PERL se transformarán los archivos de texto a scripts que contienen instrucciones SQL, los cuales servirán para alimentar a la base de datos SIMCA.*

Identificador: **5**
Nombre: **Ejecutar script**
Descripción: *Para alimentar a la base de datos SIMCA se debe ejecutar el comando específico para cada RDBMS de tal manera que reciba un script SQL, el cual ha sido generado previamente con los programas escritos en PERL.*

Almacenamiento de datos

Id : 01

Nombre : **Archivos *.dbf**
Descripción: *Este almacenamiento se refiere al conjunto de archivos con las mediciones de los contaminantes y vientos de la Cd. de México que son recibidos de la RAMA. El formato de estos archivos es de Dbase 3.0*

Id : **02**
Nombre : **Archivos *.txt**
Descripción: *Es el conjunto de archivos de texto separado por tabuladores que resultan de la transformación de los archivos DBase utilizando Excel.*

Id : **03**
Nombre : **Archivos *.sql**
Descripción: *Son el conjunto de scripts con instrucciones SQL que son generados a partir de los archivos de texto separado por tabuladores, utilizando los programas creados para tal propósito.*

Id : **04**
Nombre : **Simca**
Descripción: *Este almacenamiento se refiere a la base de datos y se llamará* ***simca*** *(**sim**ulación de **c**ontaminantes **a**tmosféricos), la cual contendrá todos los datos referentes a las mediciones realizadas por las estaciones de monitoreo de la RAMA.*

Flujo de datos

Id : **Archivos DBF**
Origen : **ADM**
Destino : **1**
Tipo : Archivos DBF
Descripción : Es el conjunto de archivos recibidos de la RAMA.

Id : **DBF**
Origen : **1**
Destino : **2**
Tipo : Archivos DBF
Descripción : Es el conjunto de archivos recibidos de la RAMA que ya han sido analizados y a los cuales se les han desechado los datos que no son utilizados en la base de datos, tal es el caso de magnitudes monitoreadas que no intervienen en la base de datos SIMCA .

Id : **TXT**
Origen : **2**
Destino : **3**
Tipo : Archivos TXT
Descripción : Conjunto de archivos de texto plano separado por tabuladores que resultan de la transformación de los archivos de Dbase utilizando el programa Excel.

Id : **BIN**
Origen : **3**
Destino : **4**
Tipo : Datos binarios

Descripción : Cuando los archivos de texto son enviados al servidor con un programa FTP se envían en forma binaria.

Id : **SQL**
Origen : **4**
Destino : **5**
Tipo : Scripts SQL
Descripción : Son archivos que contienen una serie de instrucciones SQL y resultan de la transformación de los archivos de texto.

Id : **Mensaje de datos insertados**
Origen : **5**
Destino : **ADM**
Tipo : Char
Descripción : Son mensajes que informan al administrador del sistema que los datos han sido agregados correctamente.

Id : **Comandos**
Origen : **4**
Destino : **03**
Tipo : Char
Descripción : Es el conjunto de comandos SQL que serán almacenados en los scripts SQL. La instrucción SQL que viene en estos archivos es:

```
insert into O3  values ('LAG', 'O3', '1/Nov/1998','1:00', 0.005);
```

Id : **Datos**
Origen : **5**
Destino : **04**
Tipo : Char
Descripción : Son los datos que serán insertados a la base de datos SIMCA, estos datos resultan de interpretar cada comando de los scripts SQL.

Diagrama Entidad-Relación de la base de datos SIMCA.

Entidades:

Nombre: **CO**

Definición: En esta entidad se almacenan las mediciones realizadas de CO (Monóxido de Carbono) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL

Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaCO DATE NOT NULL
Columnas: MedicionCO NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaCO
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **ESTACION**
Definición: En esta entidad se encuentran alamcenados los datos de todas las estaciones de monitoreo ambiental de la ciudad de México.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: Latitud CHAR(15) NULL
Columnas: Longitud CHAR(15) NULL
Columnas: Nombre CHAR(30) NULL
Columnas: Zona CHAR(5) NULL

Llave primaria: CveEstacion

Relación Padre en: ESTACION R/18 O3 *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/19 PM1 *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/20 NOX *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/21 WSP *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/22 NO2 *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/23 WDR *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/24 SO2 *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/25 RH *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/26 CO *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/27 TMP *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/28 H2S *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: ESTACION R/29 UVB *Cardinalidad: Uno a cero,uno o muchos*

Nombre: **H2S**
Definición: En esta entidad se almacenan las mediciones realizadas de H2S (Acido Sulfidrico) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaH2S DATE NOT NULL
Columnas: MedicionH2S NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaH2S
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **MEDIDA**
Definición: En esta entidad se encuentran almacenada la clave y el *Nombre* de la magnitud.

Columnas: CveMedida CHAR(5) NOT NULL
Columnas: *Nombre*Medida CHAR(30) NULL

Llave primaria: CveMedida

Relación Padre en: MEDIDA R/1 CO *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/10 O3 *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/11 PM1 *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/12 WSP *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/13 WDR *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/14 RH *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/16 UVB *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/17 H2S *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/6 TMP *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/7 SO2 *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/8 NO2 *Cardinalidad: Uno a cero,uno o muchos*
Relación Padre en: MEDIDA R/9 NOX *Cardinalidad: Uno a cero,uno o muchos*

Nombre: **NO2**
Definición: En esta entidad se almacenan las mediciones realizadas de NO2 (Oxido Nitroso) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaNO2 DATE NOT NULL
Columnas: MedicionNO2 NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaNO2
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **NOX**
Definición: En esta entidad se almacenan las mediciones realizadas de NOX (Oxido de Nitrogeno) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaNOX DATE NOT NULL
Columnas: MedicionNOX NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaNOX

Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **O3**
Definición: En esta entidad se almacenan las mediciones realizadas de O3 (Ozono) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaO3 DATE NOT NULL
Columnas: MedicionO3 NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaO3
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **PM1**
Definición: En esta entidad se almacenan las mediciones realizadas de PM1 (Parcitulas Suspendidas) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaPM1 DATE NOT NULL
Columnas: MedicionPM1 NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaPM1
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **RH**
Definición: En esta entidad se almacenan las mediciones realizadas de RH (Humedad Relativa) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaRH DATE NOT NULL
Columnas: MedicionRH NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaRH
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **SO2**

Definición: En esta entidad se almacenan las mediciones realizadas de SO2 (Anhidrido Sulfuroso) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaSO2 DATE NOT NULL
Columnas: MedicionSO2 NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaSO2
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **TMP**
Definición: En esta entidad se almacenan las mediciones realizadas de TMP (Temperatura) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaTMP DATE NOT NULL
Columnas: MedicionTMP NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaTMP
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **UVB**
Definición: En esta entidad se almacenan las mediciones realizadas de UVB (Indice Ultravioleta) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaUVB DATE NOT NULL
Columnas: MedicionUVB NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaUVB
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **WDR**
Definición: En esta entidad se almacenan las mediciones realizadas de WDR (Dirección de los vientos) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL

Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaWDR DATE NOT NULL
Columnas: MedicionWDR NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaWDR
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Nombre: **WSP**
Definición: En esta entidad se almacenan las mediciones realizadas de WSP (Velocidad de los vientos) por las estaciones de monitoreo.

Columnas: CveEstacion CHAR(5) NOT NULL
Columnas: CveMedida CHAR(5) NOT NULL
Columnas: FechaWSP DATE NOT NULL
Columnas: MedicionWSP NUMBER NULL

Llave primaria: CveEstacion
Llave primaria: CveMedida
Llave primaria: FechaWSP
Llave foránea: CveEstacion Referencia: ESTACION
Llave foránea: CveMedida Referencia: MEDIDA

Contenido del script simcapostgres7.sql para PostgreSQL v. 7.0.1

```
CREATE TABLE ESTACION (
       CveEstacion          CHAR(5) NOT NULL,
       Nombre               CHAR(30) ,
       Zona                 CHAR(5) ,
       Latitud              CHAR(15) ,
       Longitud             CHAR(15) ,
        PRIMARY KEY (CveEstacion)
)
 ;

CREATE TABLE MAGNITUD (
       CveMagnitud             CHAR(5) NOT NULL,
       NombreMagnitud          CHAR(30) ,
        PRIMARY KEY (CveMagnitud)
)
;
```

```
CREATE TABLE O3 (
       CveEstacion          CHAR(5)  REFERENCES ESTACION,
       CveMagnitud          CHAR(5)  REFERENCES MAGNITUD,
       FechaO3              DATE ,
       HoraO3               TIME,
       MedicionO3           NUMERIC(10,3) ,
        PRIMARY KEY (CveEstacion, CveMagnitud, FechaO3, HoraO3)
);

CREATE TABLE PM1 (
       CveEstacion          CHAR(5)  REFERENCES ESTACION,
       CveMagnitud          CHAR(5)  REFERENCES MAGNITUD,
       FechaPM1             DATE ,
       HoraPM1              TIME,
       MedicionPM1          NUMERIC(10,3),
        PRIMARY KEY (CveEstacion, CveMagnitud, FechaPM1, HoraPM1)

);

CREATE TABLE NOX  (
       CveEstacion          CHAR(5)  REFERENCES ESTACION,
       CveMagnitud          CHAR(5)  REFERENCES MAGNITUD,
       FechaNOX             DATE ,
       HoraNOX              TIME,
       MedicionNOX          NUMERIC(10,3),
        PRIMARY KEY (CveEstacion, CveMagnitud, FechaNOX, HoraNOX)

);

CREATE TABLE NO2  (
       CveEstacion          CHAR(5)  REFERENCES ESTACION,
       CveMagnitud          CHAR(5)  REFERENCES MAGNITUD,
       FechaNO2             DATE ,
       HoraNO2              TIME,
       MedicionNO2          NUMERIC(10,3),
        PRIMARY KEY (CveEstacion, CveMagnitud, FechaNO2, HoraNO2)

);

CREATE TABLE SO2  (
       CveEstacion          CHAR(5)  REFERENCES ESTACION,
       CveMagnitud          CHAR(5)  REFERENCES MAGNITUD,
       FechaSO2             DATE ,
       HoraSO2              TIME,
       MedicionSO2          NUMERIC(10,3),
        PRIMARY KEY (CveEstacion, CveMagnitud, FechaSO2, HoraSO2)

);

CREATE TABLE CO  (
       CveEstacion          CHAR(5)  REFERENCES ESTACION,
       CveMagnitud          CHAR(5)  REFERENCES MAGNITUD,
       FechaCO              DATE ,
```

```
        HoraCO                TIME,
        MedicionCO            NUMERIC(10,3),
         PRIMARY KEY (CveEstacion, CveMagnitud, FechaCO, HoraCO)

);

CREATE TABLE H2S  (
        CveEstacion           CHAR(5)  REFERENCES ESTACION,
        CveMagnitud           CHAR(5)  REFERENCES MAGNITUD,
        FechaH2S              DATE ,
        HoraH2S               TIME,
        MedicionH2S           NUMERIC(10,3),
         PRIMARY KEY (CveEstacion, CveMagnitud, FechaH2S, HoraH2S)

);

CREATE TABLE WSP  (
        CveEstacion           CHAR(5)  REFERENCES ESTACION,
        CveMagnitud           CHAR(5)  REFERENCES MAGNITUD,
        FechaWSP              DATE ,
        HoraWSP               TIME,
        MedicionWSP           NUMERIC(10,3),
         PRIMARY KEY (CveEstacion, CveMagnitud, FechaWSP, HoraWSP)

);

CREATE TABLE WDR  (
        CveEstacion           CHAR(5)  REFERENCES ESTACION,
        CveMagnitud           CHAR(5)  REFERENCES MAGNITUD,
        FechaWDR              DATE ,
        HoraWDR               TIME,
        MedicionWDR           NUMERIC(10,3),
         PRIMARY KEY (CveEstacion, CveMagnitud, FechaWDR, HoraWDR)

);

CREATE TABLE RH  (
        CveEstacion           CHAR(5)  REFERENCES ESTACION,
        CveMagnitud           CHAR(5)  REFERENCES MAGNITUD,
        FechaRH               DATE ,
        HoraRH                TIME,
        MedicionRH            NUMERIC(10,3),
         PRIMARY KEY (CveEstacion, CveMagnitud, FechaRH, HoraRH)

);

CREATE TABLE TMP  (
        CveEstacion           CHAR(5)  REFERENCES ESTACION,
        CveMagnitud           CHAR(5)  REFERENCES MAGNITUD,
        FechaTMP              DATE ,
        HoraTMP               TIME,
        MedicionTMP           NUMERIC(10,3),
         PRIMARY KEY (CveEstacion, CveMagnitud, FechaTMP, HoraTMP)

);
```

```
CREATE TABLE UVB  (
       CveEstacion          CHAR(5)  REFERENCES ESTACION,
       CveMagnitud          CHAR(5)  REFERENCES MAGNITUD,
       FechaUVB             DATE ,
       HoraUVB              TIME,
       MedicionUVB          NUMERIC(10,3),
        PRIMARY KEY (CveEstacion, CveMagnitud, FechaUVB, HoraUVB)

);
```

Contenido del script BDSIMCA.sql para ORACLE v. 8.

```
CREATE TABLE ESTACION (
       CveEstacion          CHAR(5) NOT NULL,
       Nombre               CHAR(30) NULL,
       Zona                 CHAR(5) NULL,
       Latitud              CHAR(15) NULL,
       Longitud             CHAR(15) NULL,
       PRIMARY KEY (CveEstacion)
);

CREATE UNIQUE INDEX XPKESTACION ON ESTACION
(
       CveEstacion                    ASC
);


CREATE TABLE MEDIDA (
       CveMedida            CHAR(5) NOT NULL,
       NombreMedida         CHAR(30) NULL,
       PRIMARY KEY (CveMedida)
);

CREATE UNIQUE INDEX XPKMEDIDA ON MEDIDA
(
       CveMedida                      ASC
);


CREATE TABLE CO (
       CveEstacion          CHAR(5) NOT NULL,
       CveMedida            CHAR(5) NOT NULL,
       FechaCO              DATE NOT NULL,
       MedicionCO           NUMBER NULL,
       PRIMARY KEY (CveEstacion, CveMedida, FechaCO),
       FOREIGN KEY  (CveEstacion)
              REFERENCES ESTACION,
       FOREIGN KEY  (CveMedida)
```

```
                REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON CO
(
       CveEstacion                    ASC,
       CveMedida                      ASC,
       FechaCO                        ASC
);


CREATE TABLE NO2 (
       CveEstacion          CHAR(5) NOT NULL,
       CveMedida            CHAR(5) NOT NULL,
       FechaNO2             DATE NOT NULL,
       MedicionNO2          NUMBER NULL,
       PRIMARY KEY (CveEstacion, CveMedida, FechaNO2),
       FOREIGN KEY  (CveEstacion)
              REFERENCES ESTACION,
       FOREIGN KEY  (CveMedida)
              REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON NO2
(
       CveEstacion                    ASC,
       CveMedida                      ASC,
       FechaNO2                       ASC
);


CREATE TABLE NOX (
       CveEstacion          CHAR(5) NOT NULL,
       CveMedida            CHAR(5) NOT NULL,
       FechaNOX             DATE NOT NULL,
       MedicionNOX          NUMBER NULL,
       PRIMARY KEY (CveEstacion, CveMedida, FechaNOX),
       FOREIGN KEY  (CveEstacion)
              REFERENCES ESTACION,
       FOREIGN KEY  (CveMedida)
              REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON NOX
(
       CveEstacion                    ASC,
       CveMedida                      ASC,
       FechaNOX                       ASC
);


CREATE TABLE WDR (
       CveEstacion          CHAR(5) NOT NULL,
       CveMedida            CHAR(5) NOT NULL,
       FechaWDR             DATE NOT NULL,
```

```
        MedicionWDR             NUMBER NULL
                                       CHECK (MedicionWDR BETWEEN 0 AND 360),
        PRIMARY KEY (CveEstacion, CveMedida, FechaWDR),
        FOREIGN KEY  (CveEstacion)
                REFERENCES ESTACION,
        FOREIGN KEY  (CveMedida)
                REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON WDR
(
        CveEstacion                     ASC,
        CveMedida                       ASC,
        FechaWDR                        ASC
);


CREATE TABLE WSP (
        CveEstacion             CHAR(5) NOT NULL,
        CveMedida               CHAR(5) NOT NULL,
        FechaWSP                DATE NOT NULL,
        MedicionWSP             NUMBER NULL,
        PRIMARY KEY (CveEstacion, CveMedida, FechaWSP),
        FOREIGN KEY  (CveEstacion)
                REFERENCES ESTACION,
        FOREIGN KEY  (CveMedida)
                REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON WSP
(
        CveEstacion                     ASC,
        CveMedida                       ASC,
        FechaWSP                        ASC
);


CREATE TABLE PM1 (
        CveEstacion             CHAR(5) NOT NULL,
        CveMedida               CHAR(5) NOT NULL,
        FechaPM1                DATE NOT NULL,
        MedicionPM1             NUMBER NULL,
        PRIMARY KEY (CveEstacion, CveMedida, FechaPM1),
        FOREIGN KEY  (CveEstacion)
                REFERENCES ESTACION,
        FOREIGN KEY  (CveMedida)
                REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON PM1
(
        CveEstacion                     ASC,
        CveMedida                       ASC,
        FechaPM1                        ASC
);
```

```
CREATE TABLE H2S (
       CveEstacion          CHAR(5) NOT NULL,
       CveMedida            CHAR(5) NOT NULL,
       FechaH2S             DATE NOT NULL,
       MedicionH2S          NUMBER NULL,
       PRIMARY KEY (CveEstacion, CveMedida, FechaH2S),
       FOREIGN KEY  (CveEstacion)
              REFERENCES ESTACION,
       FOREIGN KEY  (CveMedida)
              REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON H2S
(
       CveEstacion                    ASC,
       CveMedida                      ASC,
       FechaH2S                       ASC
);


CREATE TABLE SO2 (
       CveEstacion          CHAR(5) NOT NULL,
       CveMedida            CHAR(5) NOT NULL,
       FechaSO2             DATE NOT NULL,
       MedicionSO2          NUMBER NULL,
       PRIMARY KEY (CveEstacion, CveMedida, FechaSO2),
       FOREIGN KEY  (CveEstacion)
              REFERENCES ESTACION,
       FOREIGN KEY  (CveMedida)
              REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON SO2
(
       CveEstacion                    ASC,
       CveMedida                      ASC,
       FechaSO2                       ASC
);


CREATE TABLE O3 (
       CveEstacion          CHAR(5) NOT NULL,
       CveMedida            CHAR(5) NOT NULL,
       FechaO3              DATE NOT NULL,
       MedicionO3           NUMBER NULL,
       PRIMARY KEY (CveEstacion, CveMedida, FechaO3),
       FOREIGN KEY  (CveEstacion)
              REFERENCES ESTACION,
       FOREIGN KEY  (CveMedida)
              REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON O3
```

```
(
        CveEstacion                     ASC,
        CveMedida                       ASC,
        Fecha03                         ASC
);


CREATE TABLE RH (
        CveEstacion             CHAR(5) NOT NULL,
        CveMedida               CHAR(5) NOT NULL,
        FechaRH                 DATE NOT NULL,
        MedicionRH              NUMBER NULL,
        PRIMARY KEY (CveEstacion, CveMedida, FechaRH),
        FOREIGN KEY  (CveEstacion)
                REFERENCES ESTACION,
        FOREIGN KEY  (CveMedida)
                REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON RH
(
        CveEstacion                     ASC,
        CveMedida                       ASC,
        FechaRH                         ASC
);


CREATE TABLE UVB (
        CveEstacion             CHAR(5) NOT NULL,
        CveMedida               CHAR(5) NOT NULL,
        FechaUVB                DATE NOT NULL,
        MedicionUVB             NUMBER NULL,
        PRIMARY KEY (CveEstacion, CveMedida, FechaUVB),
        FOREIGN KEY  (CveEstacion)
                REFERENCES ESTACION,
        FOREIGN KEY  (CveMedida)
                REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON UVB
(
        CveEstacion                     ASC,
        CveMedida                       ASC,
        FechaUVB                        ASC
);


CREATE TABLE TMP (
        CveEstacion             CHAR(5) NOT NULL,
        CveMedida               CHAR(5) NOT NULL,
        FechaTMP                DATE NOT NULL,
        MedicionTMP             NUMBER NULL,
        PRIMARY KEY (CveEstacion, CveMedida, FechaTMP),
        FOREIGN KEY  (CveEstacion)
                REFERENCES ESTACION,
```

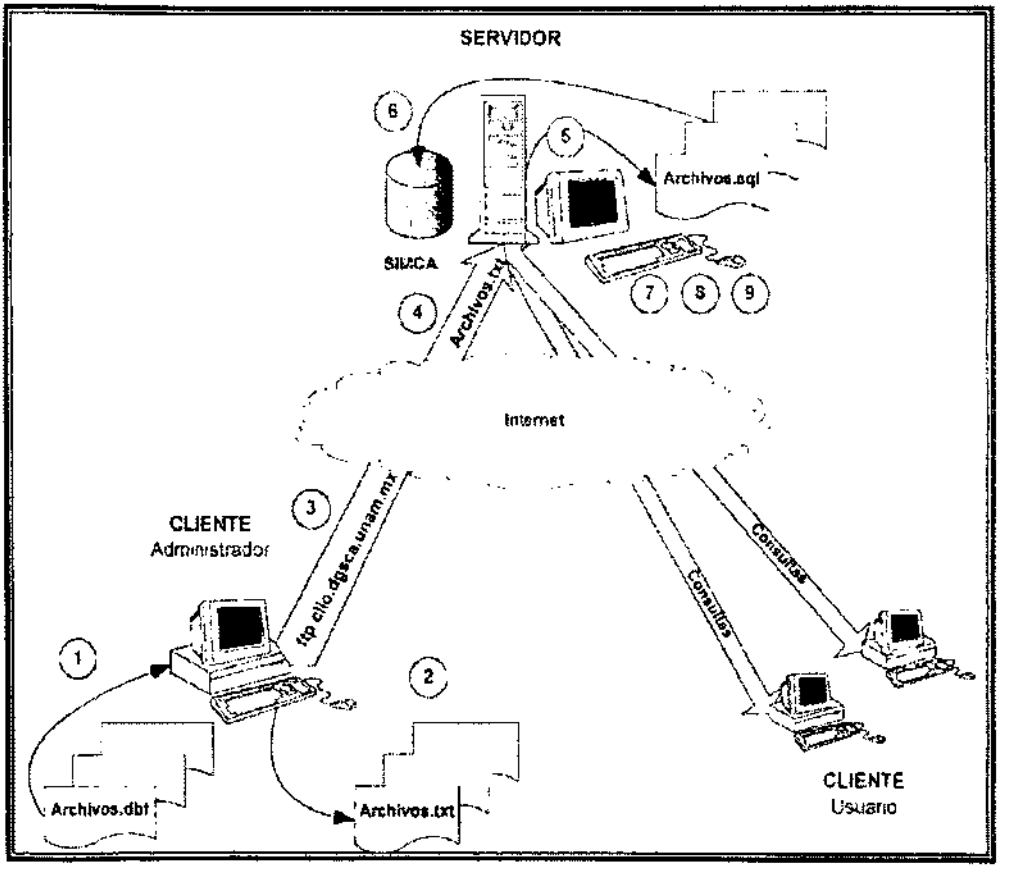
```
        FOREIGN KEY   (CveMedida)
                REFERENCES MEDIDA
);

CREATE UNIQUE INDEX XPKMEDICION ON TMP
(
        CveEstacion                     ASC,
        CveMedida                       ASC,
        FechaTMP                        ASC
);
```

Apéndice 5

Manual del administrador

Este manual comprende los procesos necesarios que debe llevar a cabo el administrador del sistema para poder alimentar la base de datos **SIMCA**.



Esquema de procesos para administrar el sistema

Los pasos a seguir en la administración del sistema de información son:

1. Recibir los archivos en Dbase (*.DBF).
2. Mediante un programa (Excel) transformar los archivos Dbase a texto separado por tabuladores y guardarlos en este formato (*.txt).
3. Con algún programa FTP mandar los archivos *.txt al servidor en donde se encuentren los programas lector.pl y genssql.pl. Supongamos que los programas *.pl se encuentran en el servidor **clio.dgsca.unam.mx**, en el shell podemos observar la ubicación y los permisos que deben tener los programas que transforman los archivos *.txt a archivos SQL (*.sql).

```
[pacolo@clio PRUEBA]$ pwd
/home/pacolo/SIMCA/PRUEBA
[pacolo@clio PRUEBA]$ ls -l
-rwxr-xr-x   1 pacolo   staff        161 Jul  6 17:27 genssql.pl
-rwxr-xr-x   1 pacolo   staff       6750 Jul  6 17:21 lector.pl
[pacolo@clio PRUEBA]$
```

*Nota: para ejecutar los programas es necesario estar trabajando en un shell del servidor. Los programas se pueden ejecutar de manera remota por telnet o trabajar directamente en el servidor.

4. Una vez que los archivos han sido transferidos al servidor (por ejemplo dos archivos *.txt) se puede ver el listado del directorio de la siguiente manera:

```
[pacolo@clio PRUEBA]$ ls -l
total 232
-rw-r--r--   1 pacolo   staff     102462 Jul  6 17:26 9810N02P.txt
-rw-r--r--   1 pacolo   staff     107621 Jul  6 17:22 9810O3P.txt
-rwxr-xr-x   1 pacolo   staff        161 Jul  6 17:27 genssql.pl
-rwxr-xr-x   1 pacolo   staff       6750 Jul  6 17:21 lector.pl
[pacolo@clio PRUEBA]$
```

*Nota: pueden ser más de dos archivos de texto. Para ver el contenido de los archivos de texto se sugiere consultar el **Apéndice 2**. El código fuente de los programas *.pl se encuentra en el **Apéndice 3**.

5. Para generar los scripts *.sql que alimentarán a la base de datos **SIMCA**. Se ejecuta el programa genssql.pl de la siguiente manera:

```
[pacolo@clio PRUEBA]$ perl genssql.pl
Procesando el archivo NO2 para las estaciones:
 FECHANO2 TAC EAC SAG TLA XAL MER PED CES HAN BJU TLI ATI VIF PLA LAG AZC UIZ TAX TAH
Procesando el archivo O3 para las estaciones:
 FECHAO3 LAG TAC EAC SAG AZC TLA XAL MER PED CES PLA HAN UIZ BJU TAX CUA TPU CHA TAH
[pacolo@clio PRUEBA]$
```

*Nota los mensajes de procesando el archivo ..., significan que el programa gennsql.pl procesa los archivos de texto (para este ejemplo 9810NO2P.txt y 9810O3P.txt) que se encuentran en el mismo directorio. Si observamos el contenido del directorio:

```
[pacolo@clio PRUEBA]$ ls -l
total 2132
-rw-r--r--   1 pacolo   staff      943326 Jul  6 17:56 9810NO2.sql
-rw-r--r--   1 pacolo   staff      102462 Jul  6 17:26 9810NO2P.txt
-rw-r--r--   1 pacolo   staff      967380 Jul  6 17:56 9810O3.sql
-rw-r--r--   1 pacolo   staff      107621 Jul  6 17:22 9810O3P.txt
-rw-r--r--   1 pacolo   staff       14382 Jul  6 17:56 ASF
-rw-r--r--   1 pacolo   staff          25 Jul  6 17:56 arch_txt
-rwxr-xr-x   1 pacolo   staff         161 Jul  6 17:27 genssql.pl
-rwxr-xr-x   1 pacolo   staff        6750 Jul  6 17:21 lector.pl
[pacolo@clio PRUEBA]$
```

podemos observar que se encuentran los archivos 9810NO2.sql y 9810O3.sql que son los scripts que son entendidos por el RDBMS y que contienen una serie de instrucciones SQL. Además son generados otros dos archivos ASF(Archivo de Salida de Fechas) y arch_txt(listado de los archivos de texto en el directorio).

Si se analiza el código fuente de los programas (ver **Apéndice 3**), ASF es un archivo generado para poder corregir el formato de fechas que es manejado por los archivos DBF recibidos de la RAMA y es modificado cada vez que se ejecuta el programa gennsql.pl. El contenido de arch_txt y ASF se puede observar en el **Apéndice 3**.

Si son generados muchos scripts *.sql es recomendable concatenarlos todos en un archivo de la siguiente manera:

```
[pacolo@clio PRUEBA]$ cat *.sql > mediciones.sql
```

Nota: Esto facilita la ejecución de un solo archivo (mediciones.sql) en el RDBMS. Un ejemplo de los archivos de entrada (.txt) y de los scripts SQL generados(*.sql) se puede ver en el Apéndice 2.

6. Para alimentar a la base de datos se utiliza la siguiente sintaxis:

   Bajo el ambiente de SQLPLUS de ORACLE:

```
SQL> @mediciones.sql
```

   En PostgreSQL podemos alimentar a la base de datos, sin estar en el ambiente de PostgreSQL, con un archivo (en este caso mediciones.sql) y especificando el nombre de la base de datos (en este caso **simca**).

```
[pacolo@clio PRUEBA]$ psql -f mediciones.sql simca
```

*Nota: Si el archivo mediciones.sql contiene la información de las 12 magnitudes monitoreadas en un mes, éste tiene aproximadamente 120K líneas que significan 120K registros a insertar; el espacio que ocupan estos registros en la base de datos es de 22.7 Mb y tarda aproximadamente 1 hora en llevarse a cabo este proceso.

7. Terminados los pasos anteriores, se pueden realizar consultas en la terminal para ver los resultados.

8. Se recomienda borrar o hacer un respaldo de todos los archivos *.txt y *.dbf del directorio en donde fueron generados, ésto con el motivo de tener espacio suficiente para procesamientos de archivos futuros.
9. El último paso es modificar los formularios de las páginas web para dar de alta los meses o años que pueden ser consultados por Internet.

Para administrar correctamente el sistema es necesario estar en contacto con el administrador del servidor ya que cualquier problema que se tenga con los servicios de web y la base de datos, deberá ser tratado y reportado con éste.

Las siguientes tablas muestran la ubicación de las páginas HTML y los programas (CGI´s) en el servidor **clio.dgsca.unam.mx**. Es necesario estar en contacto con el administrador por si existe algún cambio en cuanto a la ubicación de los directorios que se mencionan a continuación.

La ruta común para los archivos HTML es:

/usr/local/apache/htdocs/pacolo/SIMCA

Archivo	Función
index.html	**Página con frames que llama a las páginas indice.html, top.html y presenta.html**
indice.html	Es la pagina que muestra un índice del contenido del sitio.
top.html	Es la cabecera que muestra el nombre del proyecto
presenta.html	**Pagina de presentación**
simca.html	**Descripción del proyecto SIMCA**
consultas.html	Descripción de las consultas que se pueden hacer en este sistema.
consulta1.html	Pagina que contiene el formulario para consulta puntual
consulta2.html	Pagina que contiene el formulario para consulta lineal
consulta3.html	Pagina que contiene el formulario para consulta espacial.

contacto.html	Pagina en la que los usuarios del sistema ingresarán sus datos y sugerencias de consultas a la Base de Datos. Los datos serán enviados a la dirección de correo del administrador.

Tabla 1. Páginas web del sistema

La ruta común para los programas CGI´s es:

```
/usr/local/apache/cgi-bin/pacolo/SIMCA
```

Archivo	Función
consulta1.cgi	**Este CGI realiza una consulta puntual y presenta el resultado de la consulta en texto separado por tabuladores, especificando la latitud y longitud de las estaciones de monitoreo**
consulta2.cgi	Genera una consulta con la restricción de un rango de fechas.
consulta3.cgi	Elabora una consulta espacial, en donde el investigador es capaz de seleccionar horas, dias de la semana y meses.

Tabla 1. Programas CGI´s del sistema

El sitio puede ser consultada por Internet en la siguiente dirección:

http://clio.desca.unam.mx/pacolo/SIMCA

Bibliografía

MEDINETS, David. **Perl 5 a través de ejemplos**. Ed. Prentice-Hall Hispanoamericana, México 1997, 658 pag.

Wyatt, Allen L. **La magia de internet.** Ed. McGraw-Hill, México 1995, 457 pag.

Castagnetto, Jesus. **Professional PHP Programming**. Ed. Wrox. Estados Unidos 2000, 910 pag

Rowe, Jeff. **Webmaster's building Internet database servers with CGI.** Ed. New Riders, México1996, 395pag

Senn, James A. **Análisis y diseño de sistemas de información**. Ed. McGraw-Hill, México,1988 643 p

Elmasri/Navathe. **Sistemas de bases de datos**. Ed. Addison Wesley Iberoamericana, EUA,1997 887 p

Tutoriales de Oracle

http://misdb.bpa.arizona.edu/~mis531b/s99/tutorials/oracle8/oracle8.htm

Referencia Oracle 8

http://misdb.bpa.arizona.edu/~oracle/

Referencia Oracle 7

http://www.cs.ucl.ac.uk/staff/N.Zin/oracle/sql73/toc.htm

Información de contaminantes

http://www.uacj.mx/cema/Numeroesp/Default.htm

http://www.sima.com.mx/sima/df/index.html

Recursos

http://www.perl.com/pub

http://www.postgresql.org/index.html