UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

SISTEMA RECUPERADOR PARA LA BASE DE DATOS GEOGRAFICA GEOBASE

T E S I S

QUE PARA OBTENER EL TITULO DE

INGENIERO EN COMPUTACION

P R E S E N T A :

LUIS RAFAEL SILVA MACIAS

DIR. DR. RENATO BARRERA RIVERA

MEXICO, D. F. 1984

P R E F A C I O

Los avances obtenidos en el área de la computación en los últimos 40 años son sorprendentes. Día con día aparecen en el mercado nuevos productos tanto de hardware como de software, haciendo las cosas cada vez más sencillas para el usuario experto en computación y para aquél que no lo es, pero que requiere del uso de una computadora.

Uno de los productos de software más importantes, es la tecnología de base de datos que ha sido descrita como "una de las áreas de computación y ciencias de la información que ha crecido más rápidamente". Los fabricantes de computadoras empezaron a ofrecer sistemas manejadores de bases de datos en los sesentas. Actualmente miles de organizaciones dependen del continuo y buen funcionamiento de sus sistemas de base de datos.

En el Instituto de Investigaciones en Matemáticas Aplicadas y Sistemas (IIMAS) de la Universidad Nacional Autónoma de México, el Dr. Renato Barrera ha desarrollado una base de datos geográfica llamada GEOBASE.

El presente trabajo es una ampliación a GEOBASE y está organizado de la siguiente forma: en el capítulo primero se definen un serie de conceptos básicos; en el segundo se hace una descripción detallada del sistema GEOBASE con sus características lógicas y físicas; en el tercer capítulo se muestran dos tipos diferentes de recuperadores de información que son los fundamentos del sistema recuperador elaborado, el cual es explicado detalladamente en el capítulo cuatro. Finalmente vienen las conclusiones y un anexo con las Referencias bibliográficas, las que son mencionadas a lo largo de todo el trabajo.

I N D I C E

Capítulo 1

Introducción a las Bases de Datos

1.1 DEFINICION.

Una base de datos es un sistema cuyo propósito fundamental es mantener información almacenada; dicho en otras palabras, no es más que un conjunto de archivos agrupados lógicamente. Por extensión, podemos decir que un archivo es un conjunto de registros y que un registro es un conjunto de campos con información relacionada.

El campo de acción de las bases de datos es un campo relativamente joven, pero no por eso poco desarrollado.

1.2 VENTAJAS DE USO.

La ventaja principal en el manejo de la tecnología de base de datos, es que mantiene el control centralizado de la información operacional de una empresa, puesto que en muchas ocasiones se halla ampliamente dispersa y es más fácil perder el control de ella en un momento dado. Inclusive es importante que exista una persona que tenga la responsabilidad total de la base y de su contenido. Esta persona es el administrador de la base de datos (database administrator ó DBA).

El hecho de tener el control centralizado de la base de datos tiene como consecuencia las siguientes ventajas (referencia 9):

- Reducción de redundancia.

Según el enfoque tradicional, cada sistema se elaboraba con sus propios archivos de datos, lo que trae como consecuencia en muchas ocasiones redundancia en la información y espacio de almacenamiento desperdiciado, puesto que muchas aplicaciones requieren de datos similares. En una base de datos se pueden integrar los archivos con la misma información y eliminar así la redundancia (por supuesto que no toda la redundancia se puede eliminar en ciertas aplicaciones, pero si se tiene controlada).

- Eliminación de inconsistencias.

Existen casos en los cuales la información entre archivos no concuerda o es inconsistente. El manejar la información importante en forma centralizada, permite superar el problema al verificar cualquier otro dato en la misma base.

- Información compartida entre los usuarios.

Esto significa que no sólo los propietarios de la información o de las aplicaciones usarán los datos contenidos, sino que también podrán usarlos otros usuarios.

- Definición de estándares de uso.

Teniendo centralizado el control de la información, el DBA puede asegurarse de que los estándares definidos sean seguidos.

- Aplicación de restricciones de seguridad.

Al tener el DBA la jurisdicción completa sobre los datos operacionales, se puede asegurar que los accesos a la base se hacen a través de los canales adecuados, y que la verificación de autorización puede revisarse cuando se intenten accesos no previstos.

- Conflicto entre requerimientos.

Cuando existe conflicto entre requerimientos de usuarios diferentes al manejar la base de datos, éstos se pueden balancear por el DBA, estructurando la base del mejor modo posible de acuerdo a los requerimientos de la empresa.

- Independencia de datos.

Esta independencia más que una ventaja es un objetivo, puesto que a menudo, sucede que la dependencia de datos para alguna aplicación, hace imposible cambiar la estructura de almacenamiento o la estrategia de acceso, sin afectar las aplicaciones drásticamente. Entonces una recomendación de diseño es que las aplicaciones no dependan en modo alguno de una estructura particular de almacenamiento y de una estrategia de acceso.

1.3 COMPONENTES.

Los componentes de una base de datos son el software, el hardware, los datos y los usuarios.

El software es aquél que permite manejar o actualizar la base de datos (Data base management system o DBMS). Todos los requerimientos de los usuarios para manejar la base son a través del DBMS.

El hardware consiste en los volúmenes de almacenamiento

en los cuales la base de datos reside (discos, cintas, tambores, etc.) así como dispositivos, unidades de control, canales, etc.

Los datos son la información contenida en una base de datos cumpliendo con las funciones de integración y cubrimiento. Por integración queremos decir que en una base de datos se pueden agrupar archivos con datos muy diversos, en los cuales la redundancia de información es muy baja o inclusive nula.

El cubrimiento significa que cada uno de los archivos o partes de las que se compone la base de datos, puede ser usado por uno o varios usuarios para cubrir sus necesidades de información, teniendo cada uno de ellos acceso a una o varias partes de la base, sin importar el tipo de aplicación que cada uno de ellos maneje, puesto que la misma base de datos será percibida de diferentes modos por cada usuario.

El usuario es la persona que tiene alguna relación con la base de datos y podemos considerar que hay 3 tipos de usuarios. Primero está el programador de aplicaciones que es el responsable de escribir programas que usen la base de datos operando sobre ellos en: creación, eliminación o modificaciones.

El segundo tipo de usuarios es el administrador de la base de datos (DBA) que es la persona responsable del control total del sistema de base de datos.

El tercer tipo de usuario es el usuario final que es la persona que emplea por terminal el "lenguaje de recuperación" al trabajar en línea, o los programas de aplicación que accesan la base al trabajar en batch. Ellos ejecutan todas las funciones de recuperación, creación, borrado o modificación de los datos contenidos en la base.

1.4 ARQUITECTURA.

La arquitectura de la base de datos es como un esqueleto que nos permite describir conceptos generales y así explicar la estructura de los sistemas individuales.

La arquitectura de una base de datos puede adecuarse a un esquema triple en donde cada elemento de él está perfectamente definido al igual que la interrelación que existe entre ellos.

El objetivo principal de dicho esquema es la total independencia de los datos contenidos en la base. Esto se logra a través de sus componentes que a groso modo son los siguientes:

- Esquema externo: representa el concepto que tienen los diferentes usuarios de los datos almacenados en la base.

- Esquema interno: es el modelo de almacenamiento utilizado para guardar la información.

- Esquema conceptual: es la visión general de la base de datos independientemente del modelo de almacenamiento y de las aplicaciones que requieran los usuarios.

La figura 1.1 nos muestra las interrelaciones de los esquemas y las interfases lógicas de ellos.

Los usuarios finales así como el programador de aplicaciones, son los que manejan con mayor frecuencia el esquema externo, que es donde residen los conceptos para el manejo adecuado de la base de datos; ya sea mediante interfases entre la base y los lenguajes de aplicación (COBOL, PASCAL, C, PL/I, etc.) o mediante sublenguajes de datos (DSL), que son subconjuntos de algún otro lenguaje pero que están diseñados para realizar operaciones específicas en una base de datos.

En realidad cualquier sublenguaje es realmente una combinación de dos lenguajes: un lenguaje de definición de datos (DDL) que provee un medio para definir o describir los objetos de la base de datos y un lenguaje manipulador de datos (DML) que soporta el manejo o proceso de tales objetos.

Adicionalmente podemos afirmar que tanto el DDL como el DML, son parte integral del sistema manejador de la base de datos (DBMS).

El creador de la base de datos (DBC), es el generador del esquema interno que es la representación a más bajo nivel de la base de datos; en este esquema interno no solo se definen los diferentes tipos de registros almacenados, sino también se especifica que índices existen, cómo se representan los campos almacenados, cuál es la secuencia física o lógica de los registros y cosas por el estilo. El esquema interno está escrito usualmente en el DDL correspondiente a la base de datos en cuestión.

Los usuarios finales de la base, el creador de la base

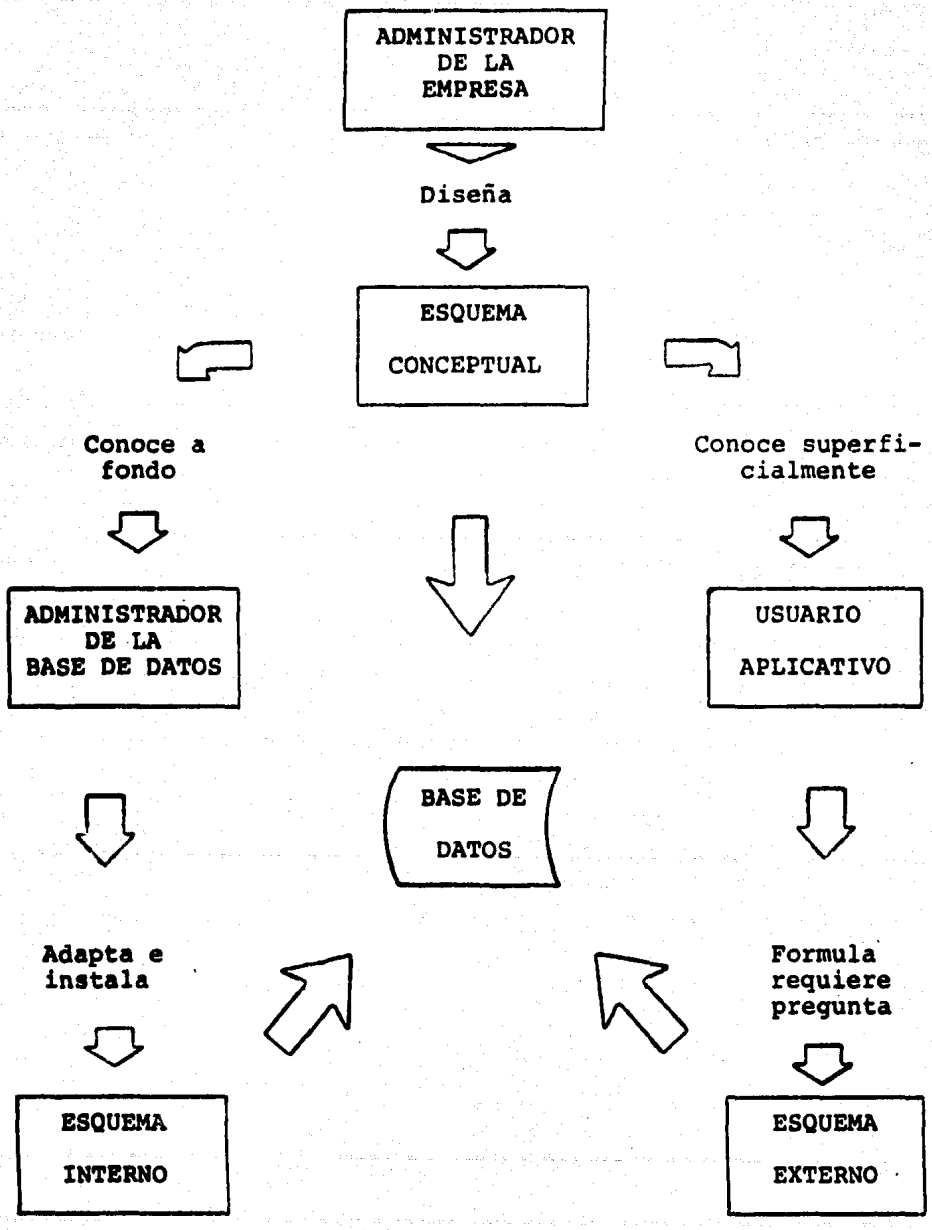


Figura 1.1
Interrelaciones en una base de datos.

y algunos directivos, son los encargados de generar el esquema conceptual de la base de datos, que es la representación de su contenido total en forma abstracta. En este esquema se definen los campos y elementos que debe contener un registro, el contenido real de los archivos, las relaciones entre campos de diferentes registros o archivos, etc. En general, en el esquema conceptual se especifican todos los detalles concernientes a la solución de los problemas de información que tienen los usuarios en sus áreas respectivas, además de que se establecen una serie de facilidades y restricciones para los diferentes usuarios de la base.

1.5 MODELOS

El lenguaje de manipulación utilizado en una base de datos, debe ser definido en términos del efecto que tenga en la estructura de la base. Las estructuras existentes se han agrupado en tres modelos: relacional, de red y jerárquico que se explican a continuación.

- Modelo Relacional.

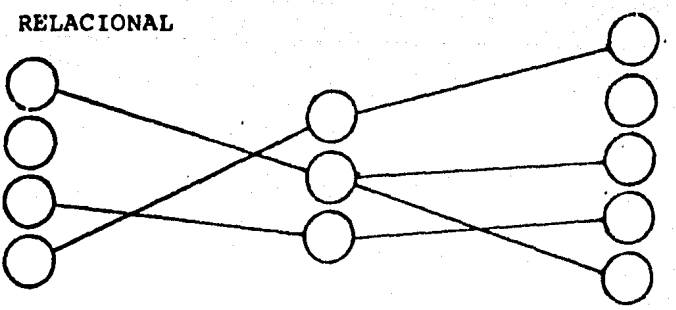
El modelo relacional está basado en el concepto matemático "relación", que es un subconjunto del producto cartesíano de una lista de dominios. En el contexto de las bases de datos se hacen relaciones considerando un orden, siendo éste, ya sea un orden definido por el sistema o su definición en términos de los valores que aparecen en la relación (fig. 1.2-a). Aquí definimos los atributos que son los valores comunes que pueden existir en determinada parte del archivo.

El modelo relacional en una base de datos, es aquél en donde cada uno de los archivos contenidos en la base es totalmente independiente de los demás, y las relaciones lógicas que pudiera haber entre ellos son establecidas en forma externa por el usuario o por los programas de aplicación.

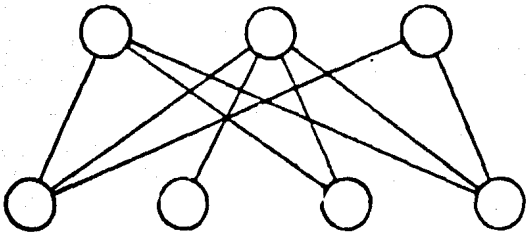
En este tipo de organización, cada uno de los archivos tiene sus llaves de acceso y "aparentemente" no hay interrelación entre un archivo y otro; sin embargo, ciertos campos de algún archivo sirven como llave en otro y se pueden hacer tantas relaciones como se desee sin afectar la estructura interna de los archivos.

La forma obvia de representar una relación, es como un archivo cuyo formato del registro consiste en una serie de campos que corresponden a atributos ordenados de una forma específica (cuando menos lógicamente). La estructura de sus

a) MODELO RELACIONAL



b) MODELO DE RED



c) MODELO JERARQUICO

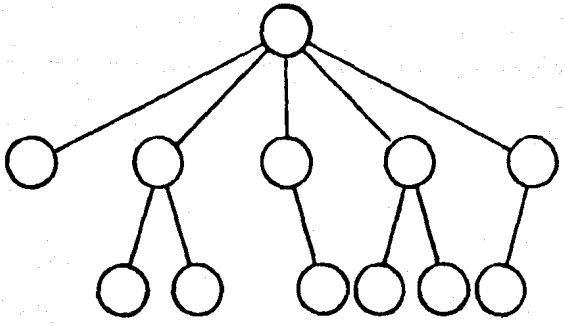


Figura 1.2

Modelos de bases de datos

archivos es la más sencilla de las presentadas al igual que el conjunto de operadores que la maneja.

El modelo relacional tiene características que los diferencian de los demás, pues cada archivo tiene un solo tipo de registro, cada registro tiene el mismo número de campos y un identificador único. Finalmente el orden del archivo no es necesariamente por la llave primaria.

Otra característica importante es que el manejo de llaves en el modelo relacional se lleva a efecto en una área adicional del archivo de datos, usualmente con el formato de lista invertida para recuperación rápida. En tales listas se incluye la dirección del dato en el archivo de datos.

Entre las ventajas de los sistemas relacionales tenemos: primero permite al diseñador de aplicaciones un amplio rango de acción entre el diseño del sistema y el problema en sí; y segundo provee una recuperación y actualización de información casi directa.

Entre sus desventajas, está el que no se puede evitar la inconsistencia de la información en forma automática, sino que tiene que ser manual. Además la redundancia de información nunca podrá ser nula.

Un ejemplo de este tipo de sistemas es el SYSTEM R.

Algunos autores afirman, que la mayoría de los sistemas comerciales están casi exclusivamente basados en uno de los otros dos modelos, una situación que se espera cambie paulatinamente (referencias 9 y 13).

- Modelo de red.

El modelo de red es una estructura que permite niveles jerárquicos entre la información en una rango muy general (fig. 1.2-b). Un registro dado puede tener cualquier número de inmediatos superiores (así como cualquier número de inmediatos dependientes).

En este tipo de estructura se manejan las listas ligadas y las doblemente ligadas, llamando a las primeras trayectorias o rutas, y a las segundas, cadenas. De este modo se establecen las relaciones de uno a varios o viceversa.

En un modelo de red, sí existen cambios en las estrategias de acceso, es probable que cambie la estructura de la base, puesto que normalmente hay más de un simple apuntador a ciertos tipos de información.

Un tipo de organización en el que se pueden almacenar los archivos es el de "multilistas", donde cada registro tiene apuntadores a los tipos de registro que forman su trayectoria, los cuales apuntan a campos específicos de longitud fija. Podemos seguir la cadena de una trayectoria en particular sin encontrarnos con otra trayectoria accidentalmente. Es notorio que si quisiéramos representar relaciones de varios a varios, caeríamos en serios problemas, puesto que cada registro podría estar en varias cadenas para una misma relación.

La estructura interna de los archivos de red es la más compleja de los modelos presentados, por lo cual los diseñadores de aplicaciones deben tener bien claro el concepto del manejo de su información, para disminuir los problemas de acceso y actualización.

Entre las ventajas de los sistemas de red está la consistencia total de la información, su simetría y la facilidad de lecturas encadenadas para registros que tienen un valor común.

Sus desventajas incluyen la significativa complicación para actualizar los apuntadores entre registros, la redundancia de la información y la imposibilidad de accesar los registros por dos trayectorias a la vez.

Ejemplos de este tipo de sistemas, son los manejadores de base de datos IMAGE y TOTAL (referencias 9 y 12).

- Modelo Jerárquico.

El modelo de base de datos jerárquico es muy similar al de red (fig. 1.2-c), sólo que en éste se permite que haya un registro inmediato anterior y no varios. La representación jerárquica es similar a una estructura de árbol, donde a partir de la raíz (parte superior del árbol), se llega a cualquier parte de la estructura por medio de apuntadores.

En general, la raíz puede tener cualquier número de dependientes, cada uno de los cuales puede, a su vez, tener también cualquier número de dependientes en el siguiente nivel y así sucesivamente hasta llegar al dependiente final al que se le llama "hoja".

En una estructura jerárquica podemos tener agrupados todos los registros en un solo archivo, pero obviamente éste contendrá varios tipos de registros (no sólo uno) y ocurrencias encadenadas. Esto indica que cada nivel tiene una especificación diferente a su superior y a su inferior.

Un detalle importante en el modelo, es que normalmente algún registro apunta a sus registros hijos y que éstos a su vez apuntan al padre, de tal modo que se puede "manejar" en el árbol según se necesite, accesos hacia atrás o hacia adelante.

La estructura interna de los archivos de este modelo, está en general totalmente desbalanceada y es asimétrica entre superiores y dependientes, lo cual puede acarrear complicaciones a los usuarios.

Por otro lado, la jerarquía es una forma natural de representar las estructuras de la vida real, aunque éste sea desbalanceado y asímetrico.

Entre las ventajas del modelo jerárquico, está la eliminación total de redundancia, la facilidad de acceso para registros del mismo nivel y la sencilla navegación a través de todo el árbol.

Sus desventajas se pueden concentrar en los problemas de complejidad al actualizarla, pues se requiere más tiempo de programación de lo que pudiera esperarse.

Ejemplos de este modelo son los manejadores SYSTEM-2000, IMS y GEOBASE del cual hablaremos en el siguiente capítulo.

1.6 MANTENIMIENTO.

Volviendo un poco a hablar de los diferentes usuarios y una vez definidos los esquemas de una base de datos, podemos definir las responsabilidades de un administrador de la base de datos (DBA) que entre otras tenemos las siguientes:

- Decide el contenido de información de la base de datos, o dicho de otro modo, identifica los elementos de interés de la empresa y la información que se debe almacenar.

- Decide la estructura de almacenamiento y la estrategia de acceso. EL DBA decide cómo presentar los datos en la base y debe especificar la estructura de almacenamiento para optimizar el espacio de máquina y los tiempos de acceso.

- Define procesos para validación y verificación de autorización.

- Define las estrategias para recuperación y respaldo. Como uno de los elementos más importantes de la empresa es

su información, el DBA debe estar preparado para cualquier contingencia de pérdida y recuperación de información.

- Monitorea el funcionamiento de la base y propone los cambios de estrategias.

Es claro que el DBA requiere de un conjunto de programas de utilería para realizar sus funciones. En algunas ocasiones esta utilería es parte del DBMS.

Una de las herramientas más importes del DBA es el diccionario de datos, que es una base de datos en sí que contiene datos acerca de los datos, o sea las descripciones de los objetos que existen en la base de datos propiamente dicha. En algunas ocasiones el diccionario de datos tiene una tabla de referencias cruzadas entre los datos contenidos.

Capítulo 2

Descripción de GEOBASE

2.1 PARTICULARIDADES DEL SISTEMA.

Los sistemas de información geográfica existen desde hace algún tiempo. Muchas de sus características son comunes a los sistemas de información pictográfica, como son: las necesidades de almacenar tanto información no posicional como información posicional; la de ejecutar operaciones geométricas (intersecciones, distancias, etc.) y la de disponer de gran almacenamiento secundario y enormes cantidades de tiempo de procesador central. La diferencia básica entre los sistemas de información pictográfica y geográfica es que los primeros están orientados a la extracción e interpretación de información "virgen", mientras que los segundos están dirigidos hacia la explotación de grandes volúmenes de información previamente digerida. Así, los segundos están primordialmente orientados hacia la consulta de datos ya analizados tal como se encuentra en los mapas y por lo tanto no es necesario que tengan la capacidad de responder preguntas del tipo de similitud (p.e. dame las figuras que tengan forma de pipa), o de extracción de patrones (dime que regiones pueden tener carreteras en una imagen multiespectral).

Los sistemas de información geográfica deben poder almacenar tanto información "estadística" (p.e. la población de un municipio o el número de camas de sus hospitales), como información "geométrica" (p.e. la imagen de un municipio).

El problema de representación geométrica consiste en codificar la partición de una región bidimensional en subregiones mutuamente exclusivas, tales como estados, municipios, etc., pudiendo considerarse como casos degenerados las características geográficas de menos de 2 dimensiones como caminos, pozos, etc.

Para codificar una partición pueden seguirse dos enfoques: representar el interior de las regiones (métodos de pixels y de parches), o bien las fronteras de las mismas (métodos de contorno). Cada uno de los métodos tiene una gran cantidad de variantes.

Uno de los métodos más convenientes es la representación por medio de pixeles, en el cual se guarda la imagen como un arreglo rectangular de elementos de resolución o pixeles (pixel=picture element), cada uno de los cuales tiene un color asignado. Esta representación tiene algunas variantes, siendo la principal la del árbol cuaternario (QUAD-TREE).

Esta representación por pixeles, ocupa mucho espacio y

consume mucho tiempo de procesador, pero en cambio tiene muchas facilidades para despliegue y operaciones geométricas.

El costo de alimentar una base de datos con información relevante es muy alto y es antieconómico tener información definida redundantemente en sistemas incompatibles, debido a las restricciones en sus diferentes representaciones internas. Como cada una de estas representaciones tiene su razón de ser, parece apropiado que una base de datos geográfica acepte diferentes tipos de representación interna y que haya independencia entre su representación y el software de aplicaciones, para poder reestructurar su representación interna y sus métodos de acceso en una forma transparente al usuario.

GEOBASE tiene las características descritas arriba. El esquema de la base tiene un solo tipo de datos definidos recursivamente permitiendo un alto grado de imposición de consistencia.

El sistema está orientado hacia la consulta y aplicaciones de planeación.

La carga y modificación de la base serán hechas por un usuario privilegiado (el administrador de la base de datos o DBA). Un usuario no puede definir nuevos entes y por economía (para no sobrecargar el sistema con archivos inútiles), no se podrá tener archivos de datos externos.

Para la implantación del sistema, se consideró el modelo jerárquico de base de datos aplicado en una microcomputadora ONYX bajo el sistema operativo UNIX. El sistema manejador está escrito en "C" en código reentrante y requiere 72K bytes para correr. Actualmente consta de 3 paquetes:

- Compilador del esquema: Este paquete toma una descripción de la base de datos, verifica su consistencia, y de estar correcta, genera los archivos necesarios.

- Módulo de carga y edición: Paquete interactivo que mediante opciones del tipo "menú", permite al usuario (DBA) tanto la carga como la modificación de la base de datos.

- Utilerías: Compactación de la base, carga y despliegue de mapas.

VENTAJAS.

GEOBASE tiene algunas ventajas sobre otros sistemas, como son:

- Almacena la información geográfica en forma diferente de la estadística logrando mayor eficiencia en la manipulación de imágenes y mejor uso de la memoria.

- Primitivos desarrollados específicamente para manipular imágenes.

- Podrá responder (en línea) consultas de información estadística, o geográfica, que involucren posición con respecto a otras imágenes o especifiquen un lugar determinado. Preguntas con restricciones lógicas en imágenes y preguntas en que se mezclen atributos estadísticos con restricciones geográficas.

- Corre en una computadora de 16 bits bajo un sistema operativo ampliamente disponible (UNIX). Su versión actual requiere de un mínimo de 36K bytes para código reentrante y 36K bytes para datos.

Puede manejar imágenes de hasta 2**232 pixeles y 2**32 instancias de un mismo tipo.

LIMITANTES.

Por otro lado, entre sus limitantes tenemos:

- Un solo método de almacenamiento de imágenes. Actualmente se tiene sólo árboles cuaternarios pero existe la posibilidad de incluir otros.

- No se tiene un lenguaje de consulta que interactúe con algún superlenguaje como Pascal o C, pues no se juzga conveniente dejar esto abierto al usuario mientras no se incluyan mecanismos de control de concurrencia y de recuperación de transacciones perdidas por fallas del sistema.

FUTURAS MODIFICACIONES.

Además del lenguaje recuperador en forma interactiva que es el motivo de este trabajo, se tienen las siguientes:

- Adquirir experiencia con los usuarios a fin de desarrollar un lenguaje de consulta que ellos (más que nosotros), consideren amistoso.

- Incluir más tipos de representación interna de mapas y un esquema interno.

- Incluir una opción de manejo desde un lenguaje de programación (por ejemplo C), con opciones de esquema externo y control de concurrencia.

- Enriquecer los tipos de datos disponibles para los atributos estadísticos de una imagen.

2.2. TIPOS DE DATOS.

IMAGENES.

El esquema de GEOBASE está definido en términos de un tipo abstracto de datos llamado imagen. Las imágenes pueden ser bidimensionales (IM2), unidimesionales (IM1) o de dimensión cero (IMØ). Las IM2 nos permiten definir áreas sobre el terreno (p.e. estados municipios, etc.), las IM1 líneas (p.e. carreteras, ríos, etc.) y las IMØ puntos específicos (p.e. refinerias, montañas, etc.).

Las imágenes pueden construirse recursivamente a partir de otras imágenes, sujeto, claro está, a cietras restricciones. Así pues, se construye una jerarquía de incluisión entre imágenes.

En este sistema existe una sola instancia de imagen llamada MAPSET que por constituir la raíz del árbol, no tiene ancestro.

RESTRICCIONES A IMAGENES.

Para el manejo de jerarquías en las imágenes, se definieron una serie de restricciones sintácticas y semánticas que son: dimensionalidad, contención y unicidad.

* Dimensionalidad: Un tipo de imagen no puede contener tipos de imágenes de mayor dimensión.
  Por ejemplo: una IM1 no puden contener un IM2.

* Contención: Imágenes contenidas en otra imagen deben estar dentro de su frontera geométrica.
  p.e. el mapa de un estado debe estar contenido dentro de un país.

* Unicidad: Las imágenes contienen un atributo llamado nombre. Este nombre debe ser único dentro de la instancia de imagen padre que la contenga.

ATRIBUTOS ESTADISTICOS DE UN IMAGEN.

Un atributo representa las cualidades o propiedades de una imagen. Todas las imágenes, incluyendo el MAPSET pueden tener atributos. Los atributos pueden clasificarse en dos formas: según su granularidad y según su tipo de dato.

Por granularidad de un atributo, definimos el espacio geométrico en el que está contenido. Por ejemplo para un municipio en particular, se puede tener como datos su población, su número de camas de hospital y la altura de su relieve.

La población y la cantidad de camas, son atributos de todo el municipio, mientras que la altura del terreno esta definida en cada punto del municipio, así pues por su granulidad los atributos se dividen en:

a) Globales (tienen sentido para la imagen completa).

b) Locales (tienen definición para cada punto de la imagen).

Huelga decir que a su vez, a los atributos locales van ligados mapas (lo que está sembrado de trigo, lo que tiene una altura mayor de 600 m., etc.).

Por su tipo de dato los atributos los clasificamos en numéricos, alfabéticos o taxonomía.

Un atributo numérico es un entero de 16 bits, un alfabético es una cadena de caracteres de hasta 12 caracteres (de 8 bits cada uno) y una taxonomía es un campo alfabético cuyas posibles ocurrencias son determinados en el esquema. Por ejemplo los días de la semana (lunes, martes, etc.) pueden declararse como taxonomía y el sistema se quejará al tratar de introducir un día de la semana llamado "robinson" en lugar de "viernes".

No hay ningún problema en introducir otro tipo de dato y lo más sencillo es incluir el resto de tipo de datos elementales del lenguaje (short, long, float, double, struct).

2.3 ESTRUCTURA DE ARCHIVOS.

La base de datos consta de cinco tipos de archivos que a saber son:

- de esquema,
- de índices,
- de imágenes,
- de valores locales y
- de mapas.

cuya definición se explica en los párrafos siguientes.

Para describir el formato de los diferentes archivos de **GEOBASE** usaremos la notación y tamaño de las variables del lenguaje "C", de la ONYX que son las siguientes:

Mnemónico	S i g n i f i c a d o	Tamaño en bytes
char	caracter	1
int	variable entera	2
short	variable entera corta	2
long	variable entera larga	4

Además el caso de arreglos alfabéticos (usualmente de 12 posiciones), lo representamos como:

char arreglo [12]

2.3.1 Archivo de esquema (AE).

Este archivo contiene la descripción de los componentes de la base. Su nombre completo es:

GEOBASE/AE/XXXXXX

donde "XXXXXX" denomina a la base en cuestión.

Contiene 6 tipos distintos de registros de longitud fija almacenados en forma secuencial, todos ellos de 48 bytes de longitud.

Los tipos de registro son: encabezado, libres, de definición de imágenes, de atributos, de taxonomía y de valores taxonómicos.

a) Registro de encabezado. Es un registro único que ocupa los bytes 0 a 47 del archivo y sirve para apuntar a la lista de registros libres y llevar estadísticas.

Su formato está en un struct de C y es el siguiente:

Tipo	N o m b r e	D e s c r i p c i ó n
long	apprimlibre	apuntador al primer registro libre del archivo.
int	numtot	número total de registros del archivo.
int	numlibres	número de registros libres.
int	numtdi	número de registros en TDI.

int	numtda	número de registros en TDA.
int	numtdt	número de registros en TDT.
int	numtvt	número de registros en TVT.
char	blancos[32]	relleno para 48.

b) Registro de definición de imágenes. Este conjunto de registros constituye la Tabla de Definición de Imágenes (TDI) y nos da el nombre y algunas características de las imágenes residentes. Está descrito por:

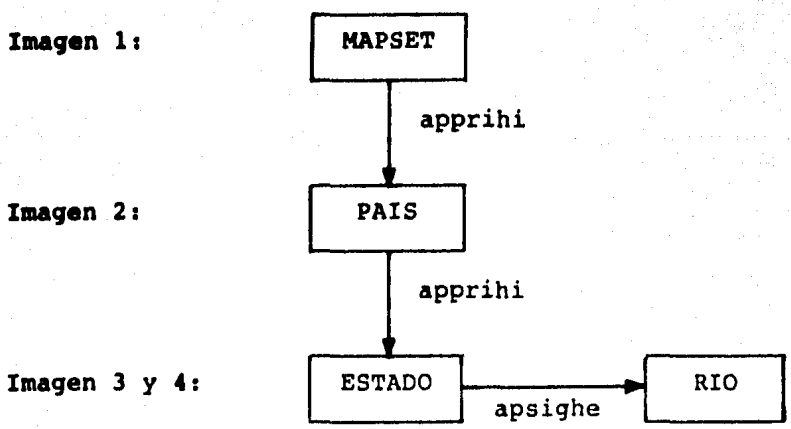
Tipo	Nombre	Descripción
char	tipoimagen[12]	nombre del tipo de imagen.
short	dim	dimensión de la imagen. (si es IM2=2, IM1=1 e IM0=0).
char	tipopadre[12]	nombre del tipo de imagen padre.
long	apprihi	apuntador al primer tipo de imagen hija (en el mismo archivo.)
long	apsighe	apuntador al siguiente tipo de imagen hermana (en el mismo archivo).
long	apprimatr	apuntador al primer atributo que le pertenece en la tabla de definición de atributos (TDA).
int	longitud	longitud del registro en la tabla de contención de imágenes (TCI) correspondiente a una instancia del tipo de imagen. Depende del número y tipo de sus atributos.
char	blancos[9]	relleno para completar 48.

Los tipos de imágenes en TDI se encuentran ligados según su jerarquía. Además apuntan a su cadena de atributos. La figura 2.1a ilustra las ligas usadas en un ejemplo.

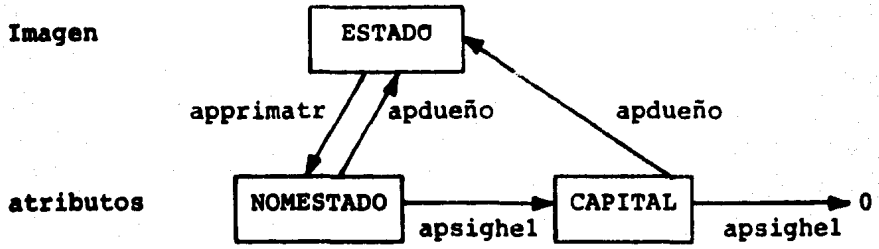
c) Registros de definición de atributos. Este conjunto de registros constituye la Tabla de Definición de Atributos (TDA). Aquí se define el nombre, granularidad y tipo de cada uno de los atributos. Tienen ligas entre sí, así como con el registro del tipo TDI del que dependen. Su descripción es la siguiente:

Tipo	Nombre	Descripción
char	nombreatr[12]	nombre del tipo de atributo.
char	localglob[1]	es un caracter "L" si es un atributo local y un "G" si es global.
char	claatrib[1]	es un caracter "N" si es un atributo numérico, una "A" si es alfabético o una "T" si es taxonomía.
long	apdueño	apuntador al tipo de imagen en TDI al que pertenece el atributo.
long	apsighel	apuntador al siguiente atributo en TDA que pertenezca a la misma imagen (hermano).
int	bytecomienza	indica en que byte de su registro del archivo de imágenes (TCI) comienza el valor del atributo.
int	bytefin	indica en que byte de su registro del TCI termina la información del atributo.
long	aptax	si el atributo es una taxonomía apunta a la definición de ésta (TDT).
char	blancos[18]	espacio para completar 48.

a) Imágenes en forma jerárquica.



b) Atributos de una imagen cualquiera (apuntadores).



c) Taxonomías (TDT) y valores taxonómicos (TVT).

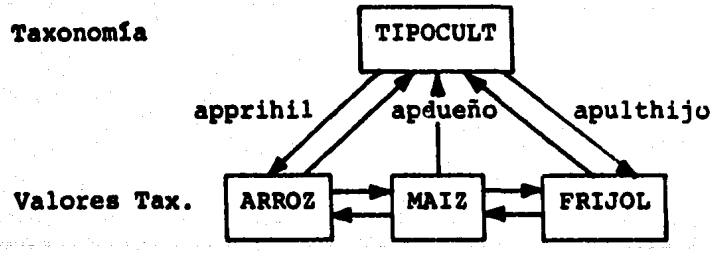


Figura 2.1
Estructura del archivo de esquema.

Los apuntadores de atributos para la imagen estado (por ejemplo) suponiendo que sean nombre del estado (NOMESTADO) y CAPITAL están representados en la figura 2.1b.

d) Registros de definición de Taxonomías. Este grupo de registros forman la tabla de definición de taxonomías (TDT) y nos da el nombre de la taxonomía y la liga a una cadena de posibles valores. Su definición es la siguiente:

Tipo	Nombre	Descripción
char	nomtax[12]	nombre del tipo de taxonomía.
long	apprihil	apuntador al primer registro de la cadena asociada de valores en la tabla de valores taxonómicos (TVT).
long	apulthijo	apuntador al último registro de la cadena asociada de valores en TVT.
int	numhijos	número de valores de la taxonomía.
char	blancos[26]	relleno para completar 48.

e) Registros de valores taxonómicos. A estos se les llama tabla de valores taxonómicos (TVT) y almacena todos los valores de cada una de las taxonomías. Tiene el formato:

Tipo	Nombre	Descripción
char	nomvalor[12]	nombre del valor de la taxonomía.
long	apdueño1	apuntador a la definición de taxonomía al cual pertenece.
long	apsighe2	apuntador al siguiente valor que pertenezca a la misma definición de taxonomía.

Tipo	Nombre	Descripción
long	aphermant	apuntador al valor anterior que pertenezca a la misma definición de taxonomía.
int	rango	número de orden que tiene tal valor dentro de la cadena de registros.
char	blancos[20]	relleno para 48 caracteres.

Los registros de TVT forman una lista doblemente ligada, cuya cabeza es el registro TDT correspondiente a la taxonomía a quien pertenecen. La figura 2.1c nos muestra la taxonomía tipo de cultivo(TIPOCULT) con los valores ARROZ, MAIZ y FRIJOL.

f) Registros libres (RLE). Forman una lista doblemente ligada cuya cabeza y cola es el registro de encabezado. Su formato es el siguiente:

Tipo	Nombre	Descripción
long	apant	apuntador al registro libre anterior.
long	apsig	apuntador al siguiente registro libre.
char	blancos[40]	espacios en blanco.

2.3.2 Archivo de índices (AI).

Contiene la información para el acceso de todos los registros de la base, tanto para su localización única (índice primario), como múltiple (índices secundarios). Los índices se encuentran en la forma de un árbol de Bayer o árbol balanceado pero conocido como B-tree o árbol B.

Este archivo se denomina GEOBASE/AI/XXXXX, donde "XXXXX" es el nombre de la base de datos en cuestión. Contiene tres tipos de registros de 512 bytes cada uno y son:

- de encabezado,
- libres y
- ocupados.

a) Registro de encabezado. Es único en el archivo y ocupa los bytes 0 a 511 del mismo. Tiene la dirección de la raíz del árbol B y del primero de la lista ligada de registros libres, así como estadísticas de ocupación. Su estructura es:

Tipo	Nombre	Descripción
long	dirraíz	apuntador de la raíz del árbol B.
long	dirprimerlibre	apuntador al primer registro libre.
long	numregtree	número de registros ocupados en el árbol.
long	numregistros	número total de registros en el archivo.
char	blancos[496]	relleno para 512.

b) Registros libres. Los registros libres se encadenan en una lista doblemente ligada de la cual se toman al momento de hacer crecer el árbol. Su formato es:

Tipo	Nombre	Descripción
long	dirant	dirección del registro libre anterior.
long	dirsig	dirección del siguiente registro libre.
char	blancos[504]	espacios de relleno para 512.

c) Registro ocupado. En este tipo de registro reside el árbol B. En cada uno de ellos puede haber de 8 a 16 llaves.

La "llave" se encuentra constituida por su tipo (puede haber 10 tipos de índices) y la clave por la que propiamente se busca (clavechica). Su formato es:

Tipo	Nombre	Descripción
int	tipoin	tipo de dato que se trata. Puede valer de 1 a 10 y nos define el dato al que apunta (imagen,

Tipo	Nombre	Descripción
		atributo, taxonomía, valor taxonómico, etc.).
char	clavechica[20]	el valor por el cual se busca en el índice.

A su vez cada "llave" tiene su correspondiente apuntador al esquema formando un "nodo" del B-tree, en formato long.

Cada uno de los nodos tiene un apuntador y con él forma un átomo. Dicho apuntador es un long y se dirige a la "hoja" sucesora del árbol.

El "átomo" repetido 16 veces y algunos datos adicionales forman la "hoja" según el siguiente formato.

Tipo	Nombre	Descripción
long	po	primer apuntador a un descendiente de la hoja.
struct	átomo[16]	descrito en párrafos anteriores.
int	número	número de átomos almacenados en la hoja.
char	sobra[26]	relleno para 512 caracteres.

2.3.3. Archivo de imágenes(TCI).

Este archivo constituye la Tabla de Contención de Imágenes(TCI) y contiene las instancias de los tipos de imágenes y sus atributos globales. Su nombre está formado por GEOBASE/TCI/XXX/YYY donde "XXX" es el nombre de la base de datos y "YYY" es el tipo de imagen. Se crea un archivo de esta naturaleza por cada imagen definida en el esquema. El tamaño de sus registros varía según el tipo de imagen, y puede tener una longitud máxima de 512 bytes. Sus instancias no están ligadas entre sí y únicamente apuntan a la instancia de la imagen de la cual dependen(imagen padre).

Contiene 3 tipos diferentes de registros: de encabezados, ocupados y libres.

a) Registro de encabezado. Es único y ocupa los primeros bytes del archivo. Apunta a la cadena de registros libres. Como el formato de cada archivo de TCI es variable, si n es el número de bytes en un registro, su formato completo es:

Tipo	Nombre	Descripción
long	apprimlibre	apuntador al primer registro libre del archivo.
int	numreglibre	número de registros libres del archivo.
int	numtotreg	número total de registros del archivo.
char	blancos[n-8]	relleno para la longitud del registro.

b) Registro ocupado. La composición de un registro ocupado varía de acuerdo al número y tipo de atributos que según la imagen lo integren. Sin embargo, todos los registros constan de tres secciones principales:

1) Sección fija (primeros 25 caracteres):

i) Nombre de la instancia de la imagen (12 caracteres).

ii) Apuntador a la instancia de la imagen padre(long).

iii) Apuntador a su tipo de imagen en el esquema del archivo AE (long).

iv) Su dimensionalidad, puede ser 2, 1 ó 0 (short).

v) Apuntador a su mapa correspondiente en el archivo QT (long).

2) Sección atributos globales (siguientes caracteres). Se deja espacio suficiente para almacenar el valor del atributo: 12 caracteres si es alfabético o taxonomía y un entero si es numérico. La posición de comienzo y fin debe concordar con lo prescrito por los campos byte-comienza y byte-fin del registro de atributos correspondiente que se encuentra en el archivo de esquema(AE).

3) Sección de atributos locales (últimas posiciones). Para cada atributo local se reserva un long (o sea cuatro carac-

teres), con el fin de poner allí un apuntador a su cadena de registro de valores que están en el archivo RVL.

Para ilustrar la composición de un registro de TCI supongamos que tenemos una imagen del tipo ESTADO, cuyo padre es del tipo PAIS. La imagen ESTADO posee los siguientes atributos: NOMESTADO(atributo global alfabético), POBLACION (global numérico) y TIPOCULTIVO(local alfabético). Las instancias de este tipo de imagen se graban en el archivo.

GEOBASE/TCI/XXXXX/ESTADO

siendo "XXXXX" el nombre de la base de datos.

Los registros de ese archivo tienen una longitud fija de 43 bytes y su estructura está dada por la figura 2.2.

c) Registros libres. Tienen la estructura:

Tipo	Nombre	Descripción
long	apsiglibre	apuntador al siguiente registro libre.
char	blancos[n-4]	relleno para ocupar los n caracteres del registro.

2.3.4 Archivo de valores locales(RVL).

Su nombre está formado por GEOBASE/RVL/XXXX, donde "XXXX" es el nombre de la base en cuestión.

Este archivo contiene registros que guardan los valores de los atributos locales pertenecientes a las diversas instancias de imágenes. Forman con su correspondiente imagen una lista doblemente ligada.

Contiene tres tipos de registros: de encabezado, ocupados y libres, todos ellos con una longitud de 32 bytes.

a) Registro de encabezado. Es único y contiene la lista de los registros libres. Su formato es:

Tipo	Nombre	Descripción
long	apprimlibre	es el apuntador al primer registro libre del archivo.
int	numreglibres	indica cuántos registros libres tiene el archivo.

43 BYTES

12	4	4	1	4	12	2	4
Nombre de la instancia	Apuntador a la instancia padre	Apuntador al tipo de imagen	Dimensionalidad	Apuntador al árbol cuaternario.	Valor de NOMESTADO	Valor de POBLACION	Apuntador a su cadena TIPOCULTIVO

Figura 2.2

Estructura del registro de ESTADO en TCI.

Tipo	Nombre	Descripción
int	numtotreg	nos da el total de registros contenidos.

b) Registro ocupado. Contiene el valor del atributo y forma una lista doblemente ligada con atributos del mismo tipo correspondientes a la misma imagen. La cabeza y cola de esa lista es la instancia del registro del TCI que corresponde a la imagen dueña de la cadena de atributos. A su vez cada registro de valor local apunta a su imagen propietaria y al mapa del valor local en el archivo QT. Su estructura es:

Tipo	Nombre	Descripción
long	aptci	apuntador al registro de TCI (instancia del tipo de imagen que lo contiene).
long	aptda	apuntador al tipo de atributo al que pertenece dentro del archivo AE.
long	apant	apuntador al valor anterior (en el mismo archivo).
long	apsig	apuntador al valor siguiente (en el mismo archivo).
long	apqt	apuntador a su árbol cuaternario correspondiente en el archivo QT.
char	valor[12]	valor del atributo. En caso de ser numérico se usan sólo sus primeros dos bytes.

c) Registros libres. Su estructura es:

Tipo	Nombre	Descripción
long	apsiglibre	apuntador al siguiente registro libre (el último apuntador es nulo -1L).
char	blancos[28]	relleno para 32 posiciones.

2.3.5 Archivo de mapas o de árboles cuaternarios..

Su nombre está formado por GEOBASE/QT/XXXXX donde "XXXXX" es el nombre de la base de datos.

Este archivo contiene mapas de la imágenes. Está compuesto de bloques irregulares entreverando áreas libres con árboles cuaternarios compactados. El formato es un método de almacenamiento de figuras en forma binaria(referencia 3 de la bibliografía). Las áreas libres están encadenadas y se asignan siguiendo la política de "al primero que ajuste" (first fit).

Se tienen 3 tipos de bloques en el archivo: de encabezado, árboles cuaternarios compactados y libres.

a) Bloque de encabezado. Es único con una longitud de 512 bytes y contiene datos estadísticos sobre el archivo, así como un apuntador al primer bloque libre. Su estructura es:

Tipo	Nombre	Descripción
long	aprilibre	apuntador a la primera área libre del archivo. Si no hay áreas libres tiene un apuntador nulo (-1L).
long	longi	longitud del archivo en bytes.
long	asiglibre	no se usa.
char	área[500]	no se usa.

b) Bloque de árbol cuaternario compactado. Los bloques del árbol cuaternario compactado tienen 2 secciones: un encabezado de 14 bytes y el árbol compactado propiamente dicho. Su formato es:

Tipo	Nombre	Descripción
long	apsigimagen	apuntador al tipo de imagen en el esquema. En caso de tratarse de un atributo local, su valor es nulo (-1L).
long	apregco	apuntador al registro correspondiente en TCI o RVL.

Tipo	Nombre	Descripción
int	numniv	número de niveles en el árbol cuaternario.
long	lonarcom	longitud del árbol cuaternario compactado (en bytes).
char	arcuacom[X]	árbol cuaternario compactado en sí, donde X es la longitud del árbol.

c) **Bloque libre.** Tiene la siguiente estructura:

Tipo	Nombre	Descripción
long	lonareal	Longitud del área libre.
long	apsigarel	Apuntador a la siguiente área libre.
int	vacio	Indicador de bloque libre (-1).
char	espirre	Espacio irrelevante.

Capítulo 3

Referencia de sistemas recuperadores de información

3.1 INTRODUCCION.

Contrastando un poco en la discusión de los lenguajes de manipulación o sublenguajes de recuperación, y los lenguajes de programación con archivos convencionales, concluimos que estos últimos, deben ejecutarse para manejar un registro a la vez. Sin embargo, muchos problemas en las bases de datos se expresan más que en registros individuales, en conjuntos de ellos. Esto nos lleva a la necesidad de usar lenguajes más poderosos, los cuales tengan operadores capaces de manipular conjuntos completos de elementos (o registros), en lugar de hacerlo con uno a la vez.

En este capítulo mostraremos dos lenguajes de recuperación de singular importancia para este trabajo, ellos son el Algebra relacional y la Consulta según ejemplo.

3.2 ALGEBRA RELACIONAL

Al obtener información de la base de datos es posible que más de un registro califique entre las restricciones que el usuario definió. Por tal motivo, se recupera un conjunto de registros y se forma una "tabla" que contiene todos los campos o elementos de los registros en cuestión, sobre la cual pueden a su vez aplicarse más restricciones, hasta llegar a una tabla menor o igual que responde a la información solicitada.

En otras palabras, el proceso de recuperación de información se convierte en un proceso de construcción de tablas. Reconociendo este hecho, podemos definir un conjunto de operadores constructores de tablas para usarlos en la recuperación. Ese conjunto de operadores es el álgebra relacional.

3.2.1 Descripción.

El álgebra relacional es una colección de operadores de relación, cada uno de los cuales toma una o más relaciones como sus operandos y produce una nueva relación como resultado. Como el resultado de la operación es otra relación, esta última está sujeta a nuevas operaciones algebráicas.

En la siguiente sección definimos las operaciones utilizadas en este lenguaje y algunos ejemplos ilustrativos.

3.2.2 Operación.

Si analizamos las operaciones del álgebra relacional

desde el punto de vista operativo, encontramos que son cinco básicas de las cuales derivan las demás. Ellas son:

- unión
- diferencia
- producto cartesino
- proyección y
- selección.

Las otras son expresadas en términos de las primeras, pero en algunas ocasiones son usadas como operaciones primitivas (como es nuestro caso), y son las siguientes:

- intersección,
- división y
- enlace.

Realmente la agrupación de los operadores disponibles nos facilita el manejo interno, pues podemos definir funciones a partir de otras ya existentes. La ventaja fundamental es que se trata de un conjunto completo de operaciones para el manejo de la base de datos (referencia 10).

Las operaciones usadas tradicionalmente en esta álgebra son: unión (OR lógico) intersección (AND lógico), diferencia y producto cartesiano.

- Unión.

La unión de dos relaciones compatibles A y B representada por A unión B es el conjunto de elementos "t", tal que "t" pertenezca a A ó "t" pertenezca a B ó a ambos. (figura 3.1-a).

- Intersección.

La intersección de dos relaciones compatibles A y B representada por A intersección B, es el conjunto de elementos "t", tales que, "t" pertenezcan a A y "t" pertenzcan a "B" (figura 3.1-b).

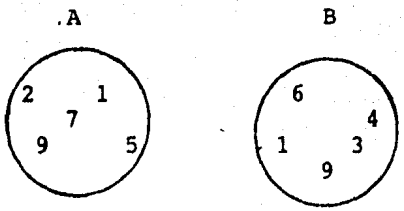
- Diferencia.

La diferencia de dos relaciones compatibles A y B representada por A diferencia B, es el conjunto de elementos "t" tales que "t" pertenezcan a A pero no pertenezcan a B (figura 3.1-c).

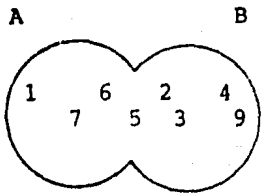
- Producto Cartesiano.

El producto cartesiano de dos relaciones A y B, repre-

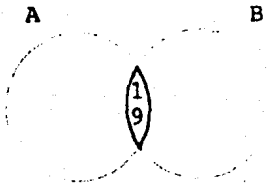
Sean las relaciones:



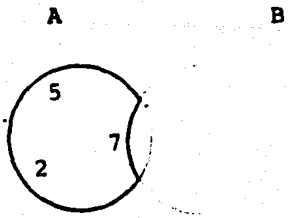
a) A unión B (A or B)



b) A Intersección B (A AND B)



c) A diferencia B (A MINUS B)



d) A veces B (A TIMES B)

1,1	2,1	5,1	7,1	. .
1,3	2,3	5,3	.	
1,4	2,4	5,4	.	
1,6	2,6	.		
1,9	2,9	.		

Figura 3.1 Ejemplo con operadores tradicionales del álgebra relacional.

sentado por A veces B, es el conjunto de todos los elementos "t", tales que "t" es la concatenación de un elemento "a" que pertenece a A y un elemento "b" que pertecece a B (figura 3.1-d).

La unión, intersección y producto cartesiano son asociativas, pero la diferencia no lo es.

El resto de operaciones son llamadas especiales y las explicaremos con más detalle. Ellas son:

SELECT (Selección).

El operador algebráico selección nos da un subconjunto "horizontal" de una relación dada; esto es, un subconjunto de elementos en una relación que cumplen con una condición determinada. El predicado es expresado como una combinación booleana de términos, donde cada uno de ellos es comparado con los elementos de la relación, estableciendo en cada caso la verdad o falsedad de la expresión. Así se eligen a todos los elementos cuya expresión sea cierta (verdadero).

En la figura 3.2-a se definen 2 relaciones cualquiera (ESTADOS y CIUDADES), que contienen ciertos atributos (columnas), cada una de las cuales tiene determinados valores. Se define la selección con la siguiente instrucción:

SELECT ESTADOS WITH ESTADO = MICHOACAN GIVING RESUL1

y se obtiene así el subconjunto de la figura 3.2-b, que es el resultado de una selección "horizontal" del conjunto original.

PROJECT (Proyección).

Con el operador proyección, se obtiene un subconjunto vertical de una relación determinada, que es un subconjunto seleccionando atributos específicos. Ningún atributo puede solicitarse más de una vez en una operación proyección. El omitir la lista de atributos en una proyección, indica que deseamos todos los atributos de la relación.

Un ejemplo de proyección podemos aplicarlo sobre el resultado de la selección del inciso anterior (RESUL1), del siguiente modo:

PROJECT RESUL1 OVER ESTADO CIUDAD GIVING RESUL2.

Obteniendo así la talba de la figura 3.2-c que es realmente una selección de atributos.

a) Ejemplo de relación:

E S T A D O S

ESTADO	CIUDAD	HAB.
.	.	.
MEXICO	TOLUCA	472,000
MICHOACAN	ZITACUARO	80,000
MICHOACAN	MORELIA	304,009
MICHOACAN	URUAPAN	125,441
MORELOS	CUERNAVACA	250,008
MORELOS	CUAUTLA	125,000
.	.	.
.	.	.

C I U D A D E S

CIUDAD1	HOSPITALES	AEROPUERTOS
.	.	.
MORELIA	15	2
JUCHITAN	3	0
CUERNAVACA	22	1
MONTERREY	74	3
ZITACUARO	12	0
CUAUTLA	8	0
URUAPAN	9	1
.	.	.

b) Ejemplo de selección (SELECT):

RESUL1

ESTADO	CIUDAD	HABITANTES
MICHOACAN	ZITACUARO	80,000
MICHOACAN	MORELIA	304,009
MICHOACAN	URUAPAN	125,441

c) Ejemplo de proyección (PROJECT):

RESUL2

ESTADO	CIUDAD
MICHOACAN	ZITACUARO
MICHOACAN	MORELIA
MICHOACAN	URUAPAN
.	.

d) Ejemplo de enlace (JOIN):

RESUL3

ESTADO	CIUDAD	HOSPITALES	AEROPUERTOS
MICHOACAN	ZITACUARO	12	0
MICHOACAN	MORELIA	15	2
MICHOACAN	URUAPAN	9	1

Figura 3.2

Ejemplo con Operadores especiales del Algebra relacional.

JOIN (enlace).

Cuando dos tablas tienen una columna(atributo) definida sobre un dominio común, éstas pueden ser enlazadas sobre esas dos columnas; el resultado de ese enlace es una nueva tabla más ancha que las dos enlazadas y que contiene en cada renglón la concatenación de los renglones de las tablas originales, siempre y cuando el valor del renglón en las columnas de enlace sea el mismo. Esta definición de JOIN corresponde sólo a un tipo de posibles enlaces, basado en la igualdad de valores en columnas comúnes y llamado equi-join (enlace igualitario).

Para evitar redundancia de recuperación, se elimina una de las dos columnas que son iguales, definiendo esta operación como natural-join(enlace natural). Para ilustrar el uso del natural-join, tomaremos el resultado obtenido en el ejemplo del PROJECT, haciendo un enlace entre las relaciones RESUL2 y CIUDADES de la siguiente forma:

```
JOIN RESUL2, CIUDADES THROUGH CIUDAD, CIUDAD1 GIVING RESUL3.
```

o sea efectuando un enlace natural entre relaciones como se muestra en la figura 3.2-d.

Existen otros tipos de enlaces que están relacionados con operadores de desigualdad (mayor que y menor que), siendo un caso especial del producto cartesiano de dos relaciones.

DIVISION (división).

El operador división divide un dividendo "relación A" de grado m + n por un divisor "relación B" de grado n y produce una "relación resultado" de grado m. Consideremos los dos primeros atributos de la relación ESTADOS y la relación MUESTRA (Figura 3.3-a); ESTADOS tiene un par de atributos y MUESTRA tiene un atributo, pero CIUDAD es comun en los dos casos, por lo tanto m + n=2 y como n=1, m=1 también. O sea, que el resultado es un solo atributo. Su codificación es:

```
ESTADOS DIVIDEBY MUESTRA GIVING RESUL4
```

y su resultado el mostrado en la figura 3.3-b.

a) **Relaciones iniciales:**

ESTADOS

ESTADO	CIUDAD
MEXICO	TOLUCA
MICHOACAN	ZITACUARO
MICHOACAN	MORELIA
MICHOACAN	URUAPAN
MORELOS	CUERNAVACA
MORELOS	CUAUTLA

MUESTRA

CIUDAD
ZITACUARO
MORELIA
URUAPAN

b) **Ejemplo de división** (DIVIDEBY):

RESUL4

ESTADO
MICHOACAN

Figura 3.3

Operadores especiales del Algebra relacional (cont.)

3.3 CONSULTA SEGUN EJEMPLO (QBE).

3.3.1 Descripción.

Es un lenguaje de manejo de bases de datos que permite un estilo homogéneo para la consulta, definición y control de una base de datos. Las operaciones del lenguaje se asemejan a la manipulación manual de tablas, con la misma simplicidad, simetría y neutralidad del modelo relacional. La especificación de una transacción permite expresar el proceso del pensamiento del usuario con la libertad suficiente en su formulación.

El sistema permite al usuario crear y suprimir tablas en la base de datos dinámicamente, dando al usuario, la capacidad de definir instrucciones de control y mecanismo de seguridad.

A continuación se mencionan las facilidades de la Consulta según ejemplo para recuperación, manipulación y definición.

3.3.2 Operación.

Dos conceptos básicos fundamentales para la consulta según ejemplo son:

- La programación se hace en formatos de tablas bidimensionales, llenando en los espacios de los arreglos un ejemplo de la solución.
- Los elementos para definir las consultas, son constantes (valores) y ejemplos (variables). Los elementos ejemplo se subrayan para distinguirlos de los elementos constantes.

Recuperación simple. Es listar todos los elementos sin repeticiones, indicando con el operador P. (print) los campos a imprimir.

Recuperación simple con ordenamiento. Similar al anterior, incluyendo cuál orden se desea AO (ascendente) y DO (descendente) en la columna correspondiente.

Selección simple con impresión de varios datos. Se indican los encabezados de las columnas de los datos requeridos y se aplica para todo el renglón el operador P. en la primera columna.

Recuperación de los nombres de los arreglos. Por medio del encabezado de la primera columna (nombre de la tabla) y de-

jando en blanco los demás, el sistema proporciona los encabezados correspondientes.

Al manejar el encabezado de la primera columna como variable y no indicar columnas, se obtiene el directorio de la base de datos, esto es, los nombres de todos los arreglos y los nombres de sus columnas correspondientes.

Recuperación Calificada. Se pueden emplear operadores relacionales como aplicados a los elementos que componen el ejemplo.

Recuperación calificada subrayada parcialmente. El usuario puede subrayar parcialmente el inicio, la mitad, al final de una palabra, frase o párrafo como elemento variable para definir la consulta. Resulta particularmente útil el manejar textos para localizar palabras, raíces o desinencias.

Recuperación calificada usando encadenamientos. Cuando un elemento ejemplo se usa simultáneamente en dos o más tablas, indicará que deben cumplirse simultáneamente las condiciones de las tablas. Dado que no se indica cómo debe de procesarse la recuperación, o por dónde principiar, esta formulación es neutral y simétrica.

Encadenamiento del elemento ejemplo en la misma tabla. El orden de los renglones no es significativo, el usuario no está obligado a estructurar la consulta en una forma específica. Al contrario, tiene libertad para formularla añadiendo especificaciones mas restrictivas, o con más elementos.

Recuperación usado una negación. La negación puede emplearse para componer las especificaciones en una tabla y a la vez, para negar todo el renglón de ella al emplearla en la primera columna.

Recuperación de salidas colectivas desde varias tablas. Debido a que el resultado compone de hecho una nueva tabla, el usuario debe generar el esqueleto de esa tercera tabla y llenarla con ejemplos mapeados de las dos tablas existentes que satisfacen el planteamiento de la consulta. Puesto que es una tabla creada por el usuario, éste puede optar por dar los encabezados o dejarlos en blanco.

Operaciones Aritméticas. Para hacerlas, se define la tabla de salida y se indica en ella la operación aritmética deseada para cada campo, que incluye los elementos y los operadores encerrados entre paréntesis.

Recuperación usando una caja de condición. En la consulta según ejemplo, existen dos entes bidimensionales. El primero es el esqueleto de la tabla, que ha sido descrita y el

segundo es la caja de condición, que tiene el encabezado "CONDICIONES". Puede desplegarse cuando lo desee el usuario.

Se usa para plantear condiciones difíciles de expresar en las propias tablas. Se manejan siempre relaciones lógicas en ellas y pueden emplearse varios renglones incluyendo todas esas condiciones simultáneamente.

Recuperación usando AND y OR. Se manejan implícitamente estas operaciones, cuando se dan varios renglones con condiciones en las tablas para que se cumplan a la vez, similar a la recuperación con encadenamiento. Por medio de la caja de condiciones se pueden establecer las expresiones lógicas que deben cumplirse, conectándolas por medio de símbolos para AND y OR.

Debe notarse que con los conceptos de elemento ejemplo, elemento constante, operadores y empleo en los esqueletos de las tablas y caja de condiciones, el usuario puede expresar consultas bastante complicadas. Con ellos se cubre una amplia variedad de acciones en las operaciones de las bases de datos, tales como proyecciones, selecciones, uniones, proyección de uniones, diferencia, intersección y operaciones aritméticas.

Recuperación usando funciones interconstruídas y el operador ALL. Existen seis operaciones en el lenguaje de consulta, según ejemplo, que son:

CNT. (cuenta)
SUM. (suma)
AVG. (promedio)
MAX. (máximo)
MIN. (mínimo)
UN. (único) - Puede combinarse con CNT., SUM., AVG.

Las funciones interconstruídas operan únicamente sobre expresiones de conjuntos y deben por ello estar acompañadas del operador ALL., o una expresión de conjuntos entre paréntesis.

Recuperación con agrupamiento. El elemento en base al cual se hace el agrupamiento, se indica con subrayado doble y los operadores actúan para cada grupo individualmente, al determinarlos según el elemento de agrupamiento.

Recuperación involucrando cadenas de conjuntos usando ALL. Cuando al elemento encadenante se le aplica el operador ALL para definir un conjunto múltiple, se están encadenando las tablas según ese conjunto.

Aunque un encadenamiento simple se consigue usando el mismo elemento de ejemplo en dos o más tablas, en un encadenamiento por conjuntos se debe distinguir entre igualdad de conjuntos y contención de conjuntos. Lo primero se obtiene con encadenamiento por conjunto idénticos y lo último por medio de un asterisco.

Finalmente diremos que existe también un conjunto de operaciones para actualización y creación de tablas (esqueletos) y datos contenidos en la base (referencia 4).

3.3.3 Consideraciones para bases de datos jerárquicas.

Las operaciones establecidas para la consulta según ejemplo son idependientes de la estructura de la base de datos; es decir, que son aplicables a cualquier modelo de base de datos, ya sea relacional, jerárquica, de red o cualquier combinación de las tres.

Es ventajoso reemplazar los lenguajes de manejo de datos de procedimiento, por lenguajes sin procedimiento de alto nivel, tal como consulta según ejemplo, pues, mientras que el programador de un lenguaje de procedimiento debe conocer la estructura jerárquica y las trayectorias de acceso de la base de datos para navegar en los árboles de trayectorias, el usuario de consulta según ejemplo requiere solamente saber la estructura de la base de datos. La transformación de una operación sin procedimiento a un procedimiento en un lenguaje de bajo nivel, lo hace el sistema por medio de un intérprete adecuado.

En un prototipo, se presenta al usuario una pantalla con el esqueleto de una tabla, en el que introduce el nombre de la tabla y los nombres de los campos. Por medio de teclas de función, el sistema puede llenar los nombres de las columnas. Si se requiere otra función, creará esqueletos de tablas adicionales, etc. Luego de tener en la pantalla todas las tablas y sus columnas requeridas, el usuario las llena con elementos para el planteamiento de la consulta.

Para casos en los que no se dispone de una pantalla, existe una versión lineal de la consulta según ejemplo, la cual es un equivalente de la versión tabular. Se describe principiando con el nombre de la tabla siguiendo entre paréntesis, tantos nombres de columna y sus elementos ejemplo o constantes asociados con dos puntos (:) y separados por comas. La asociación entre tablas se indica anteponiendo el nombre de la tabla a la que se asocia la que se describe en la línea, separada por punto y coma (;).

En este modo de trabajo, se cumple también que el orden

entre los renglones no tiene influencia en el planteamiento de la consulta (referencia 5).

3.4 CONCLUSIONES.

El Algebra relacional originalmente fue diseñada para un modelo de base de datos relacional, pero los operadores que usa pueden manejarse en forma general. En nuestro caso, tomamos lo que consideramos más conveniente de ella y readaptamos algunas operaciones.

En primer término usamos todos los operadores del álgebra excepto el producto cartesiano y la división. En segundo lugar el manejo de los operadores especiales nos puede incrementar de manera considerable el tamaño de memoria requerido. Por ello en lugar de tablas usamos archivos; y para evitar grandes archivos, usamos el select y project en una sola instrucción que hace la discriminación tanto horizontal como verticalmente.

Por último diremos que en lugar de ejecutar el equijoin como tal, lo dividimos en 2 joins con un project cada uno. A esta operación se le llama usualmente semi-junta (semi-join).

De la consulta según ejemplo tomamos la versión lineal para base de datos jerárquicas y diseñamos un sistema propio con características similares. Es decir que nuestro sistema provee las capacidades de unión, intersección, diferencia, selección, proyección y enlace.

Capítulo 4

Sistema recuperador elaborado

4.1 DEFINICION.

La implantación del sistema se hizo en una microcomputadora ONYX C8000, perteneciente al Instituto de Investigaciones en Matemáticas Avanzadas y Sistemas (IIMAS) de la U.N.A.M. Dicha máquina funciona bajo el sistema operativo UNIX (de ahí el motivo de que todo lo referente a programación en este trabajo esté escrito en "C").

El sistema recuperador elaborado está enfocado a la facilidad de manejo de la base de datos geográfica (GEOBASE) por parte del usuario, combinando algunas facilidades de las mencionadas en el QBE para manejo de información jerárquica (a nivel externo), y aquellas referidas en el álgebra relacional (a nivel interno).

Como ya vimos en el capítulo anterior, el QBE es un lenguaje que provee facilidades que no existen en otros recuperadores, pero, el QBE fue diseñado para ser usado desde una terminal que usa un editor de pantalla especial para la elaboración de preguntas o requisiciones (el programa recuperador corre bajo el sistema operativo VM/CMS). Además para su manejo se vale de una serie de teclas para ejecutar funciones especiales como "menús", nombres de archivos, esqueletos de archivo, etc. En nuestro caso, incluimos sólo algunos de los principales conceptos del QBE, como son la "P." para impresión, el manejo de "ejemplos" para enlace entre imágenes y los operadores típicos de selección (menor que, menor o igual que, etc.).

De este modo se definió un "lenguaje" específico para la recuperación que incluye las características antes mencionadas, permitiendo al usuario una recuperación similar a la que podría efectuar en el sistema QBE.

Por otro lado el álgebra relacional juega un papel muy importante pero a nivel interno, es decir, en todas las operaciones relacionadas al manejo directo de la información existente en cada archivo. De ahí tomamos las cinco operaciones básicas: unión, diferencia, producto cartesiano, proyección y selección, y las tres adicionales: intersección, división y enlace para incluirlas en nuestro recuperador.

Los casos de unión, intersección y diferencia se definen explícitamente (a nivel externo) en el lenguaje. La selección, proyección y enlace, los aplicamos a nivel interno y están relacionados directamente con la recuperación de la base propiamente dicha. El producto cartesiano y la división no fueron usadas en el recuperador.

Puede verse entonces que tomamos lo que más se adecuaba a nuestras necesidades de cada uno de los sistemas, para generar un nuevo recuperador que cumpla los objetivos perseguidos que, entre otros, tenemos los siguientes:

- Facilidad de manejo a cualquier usuario.

Inclusive a aquellos que no están familiarizados con la utilización de sistemas de cómputo. En este punto se considera que el recuperador debe proveer al usuario información de apoyo para el manejo del sistema como tal y de los datos contenidos en la base. Es importante también que el método a utilizar en la formulación de las requisiciones sea sencillo y fácil de entender para evitar problemas de conceptualización. En otras palabras, debe ser un sistema "amistoso al usuario".

- Recuperación eficiente de información.

La recuperación de información de la base, debe efectuarse en forma inmediata, puesto que los usuarios que tengan acceso al sistema, estarán trabajando en tiempo real y deben obtener contestación rápida a las requisiciones o preguntas que formulen.

- Flexibilidad de recuperación.

Es obvio que los usuarios que tengan acceso al sistema recuperador, deben tener visiones muy diferentes de la información contenida en la base de datos, o sea que a pesar de tener objetivos comunes para el manejo de la base, cada uno de ellos podrá solicitar información según sus necesidades y conveniencias.

Esto nos lleva a que el recuperador permita cualquier tipo de pregunta que se introduzca, siempre y cuando cumpla con ciertos requisitos sintácticos y semánticos (definidos más adelante), y exista la información solicitada.

Se contemplan en el sistema recuperador 4 partes fundamentales, que están relacionadas directamente con los resultados esperados. Estas partes en general, son las funciones que debe realizar el recuperador y son las siguientes (figura 4.1):

* Supervisor.
* Análisis sintáctico.
* Análisis semántico.
* Funciones auxiliares.

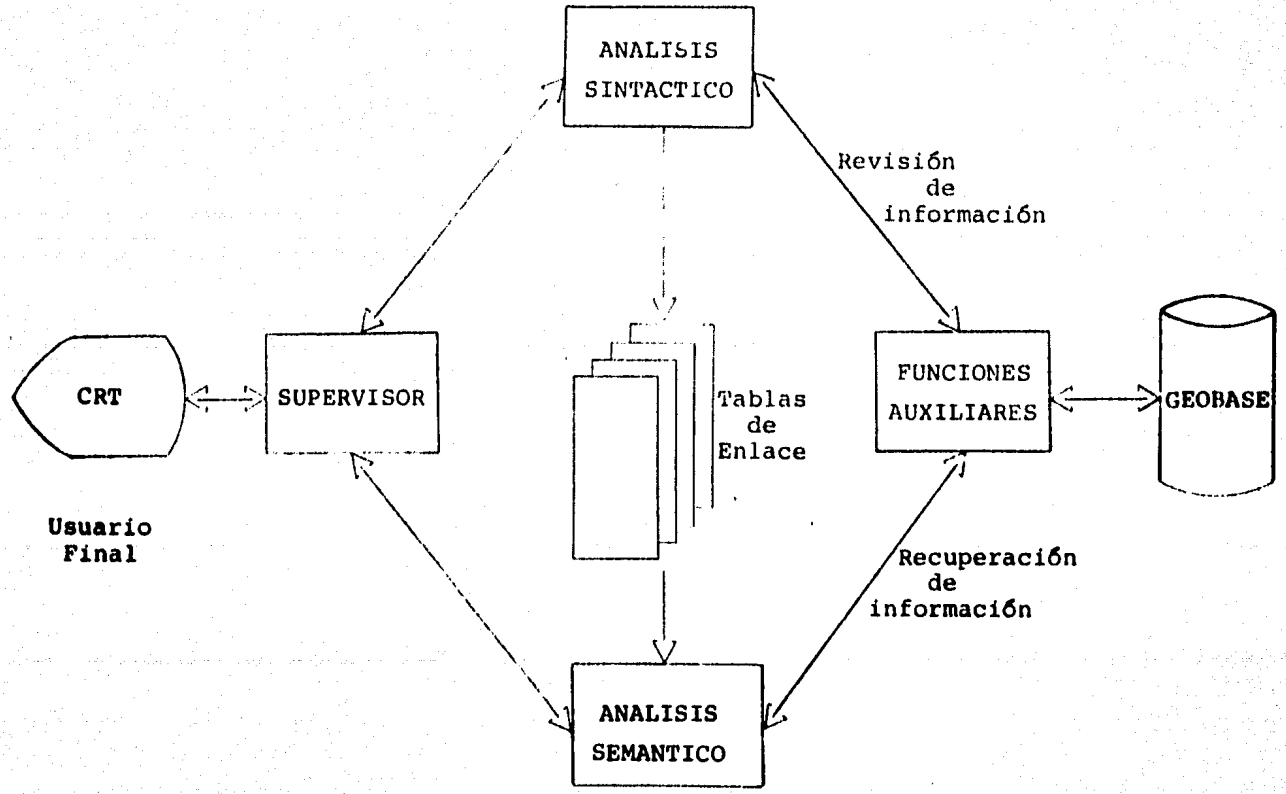


Figura 4.1 Diagrama general del recuperador.

Cada una de estas partes o funciones, esta constituída por programas o subprogramas, los que a su vez constan de una o varias subrutinas escritas en un lenguaje de la microcomputadora ONYX (en este sistema usamos "C").

En la parte de supervisión (como su nombre lo indica), deben coordinarse todas las acciones que se lleven a efecto al ejecutar o manejar el sistema recuperador. En ella se contempla el manejo de opciones y restricciones que el usuario puede conocer en una sesión. Además permite el manejo de las variables globales y el control del flujo general de la información.

La función de análisis sintáctico, tiene por objetivo revisar la correcta construcción de las preguntas y requisiciones que se formulen al recuperador, así como la verificación de fórmulas de relación, atributos e imágenes contenidas en la base de datos.

En análisis semántico, se ejecuta la recuperación de información de la base propiamente dicha, revisando la existencia de atributos en las imágenes, el ordenamiento correcto de cada imagen, la existencia de atributos para selección, enlace, impresión, y en fin todo aquello relacionado directamente con los resultados que se buscan.

La parte de funciones auxiliares se refiere a todos aquellos subprogramas o subrutinas en los cuales se sustentan las 3 etapas anteriores. En esta etapa se definen todas aquellas funciones necesarias para la correcta operacion del sistema.

En las secciones siguientes se detalla cada una de las 4 partes que constituyen el sistema recuperador, así como las relaciones externas e internas de cada función.

4.2 SUPERVISOR.

Esta fase la constituye básicamente el programa supervisor cuyo objetivo es (como ya lo mencionamos), coordinar todas las acciones que se llevan a efecto durante la ejecución del recuperador. A continuación se hace una descripción detallada de su funcionamiento y de las opciones que debe manejar.

4.2.1. Programa Principal.

Al ser invocado el sistema recuperador, el programa supervisor toma el control y verifica si el llamado se hizo en

forma accidental o intencional. Para ello se despliegan los mensajes:

Sistema recuperador de información para GEOBASE
Versión 1.0

Deme el nombre de la base que desea accesar:_

que son de identificación y solicitan el nombre de la base de datos que se quiere usar.

Se hace entonces una llamada para abrir el archivo que fue solicitado, cuyo nombre completo es:

GEOBASE/AI/XXXXXX

donde "XXXXXX" es una palabra de hasta 12 caracteres, que debe ser leída a continuación del último mensaje desplegado. La apertura del archivo debe ser con la opción 2.

En caso de que el nombre del archivo tecleado por el usuario sea mayor que doce (12), se envía un mensaje alusivo y se continúa de acuerdo a lo especificado en el inciso 4.2.7.

Si es adecuado el número de caracteres y el archivo se abrió correctamente (es decir que el valor que regresa la función "open" es mayor de dos (2)), se envía un mensaje de bienvenida al sistema, mencionando las funciones de "apoyo" y "comando" definidas en los incisos 4.2.3 y 4.2.5 y se ejecuta la subrutina "proceso" en forma iterativa mientras la bandera de "fin" esté apagada.

4.2.2. Proceso.

La subrutina proceso es la que realmente controla el flujo general de la información y como va a ser ejecutada repetitivamente, lo único que debe contener es el despliegue del mensaje:

Teclee su "pregunta" o comando de control.

y el indicador de la posición de tecleo inicial que es ">".

Dentro de la subrutina debe haber una instrucción de "switch" que contiene todas las opciones que el programa maneja, y que estarán definidas por la primera letra que teclee el usuario después del indicador mencionado. Esto significa que se lee el primer caracter que el usuario teclee, para determinar la acción a seguir, ignorando si hay más de un caracter tecleado (excepto en los casos que se especifican más adelante).

Los comandos válidos y su significado son los siguientes:

Comando	Significado
A	Apoyo al usuario
B	Borrado de cadena en memoria
C	Comandos disponibles
E	Ejecución semántica
F	Fin del proceso
M	Muestra la cadena en memoria
O	Obtiene información
R	Reemplaza
S	Ejecución sintáctica
T	Tablas de enlace
(	Inicio de pregunta

En caso de que el primer caracter que el usuario teclee después del indicador inicial sea alguno de los anteriores, se considera correcto y se procede a ejecutar la función correspondiente en forma de subrutina dentro del mismo programa, o bien, el programa externo que corresponda (las funciones de SINTAXIS y SEMANTICA se pueden incluir como subrutinas del supervisor o como programas independientes).

Por otro lado, si el primer caracter no es ninguno de los especificados, se envía el mensaje "Comando erróneo" y se regresa al programa principal (recuérdese que la subrutina proceso es repetitiva mientras la bandera "fin" sea igual a cero).

En los párrafos siguientes se especifica la función y parámetros de cada opción así como su modo de operación.

4.2.3 Comando "A" (Apoyo al usuario).

Con esta subrutina se despliega la sintaxis de recuperación para la obtención de información en GEOBASE. En ella debe incluirse -opcionalmente-, una breve descripción del método para representar sintaxis de lenguajes "Backus Naur Form", seguida de un ejemplo corto con una representación significativa.

Después de la explicación, viene la "Sintaxis completa" haciendo preguntas al llenarse una pantalla para continuar o suspender lo que haga falta (la sintaxis se especifica en la figura 4.2).

4.2.4 Comando "B" (Borrado de cadena en memoria).

Como se especifica en el inciso 4.2.13 se aparta en la

memoria un arreglo de 400 posiciones para almacenar la(s) requisición(es) del usuario. La rutina que se ejecuta con este comando, borra la cadena almacenada a partir del valor contenido en la variable entera que viene como parámetro, en este caso ese parámetro debe ser cero (0) para borrarla totalmente. Además se debe ejecutar una rutina que limpie todos los contadores y arreglos usados, (borrar es una forma simbólica de decir que debe incluirse en cada posición del arreglo el caracter '\0').

Una vez que la pregunta fue borrada se despliega un mensaje alusivo.

4.2.5 Comando "C" (Comandos disponibles).

En esta opción, se hace un desplegado de todos los comandos disponibles de acuerdo a como se especificaron en el inciso 4.2.2, con una breve descripción de cada uno de ellos, excepción hecha para el comando "T" que servirá únicamente como referencia al administrador de la base o al programador de aplicaciones. En el despliegue mencionado se deben incluir todos los caracteres tecleados en la "pregunta" con mayúsculas pues como se verá en el inciso 4.2.13, sólo se manejarán este tipo de letras.

4.2.6 Comando "E" (Ejecución Semántica).

La ejecución semántica se refiere a la recuperación "per se" de la pregunta del usuario y está condicionada a que la bandera de la sintaxis (verificada previamente) es correcta (esta bandera se enciende cuando la sintaxis ya fue revisada y no tuvo errores). El programa SEMANTICA describe en forma detallada su funcionamiento (sección 4 de este capítulo).

4.2.7. Comando "F" (Fin del proceso).

La ejecución del proceso "fin", consiste solamente en el encendido de la bandera que lleva el mismo nombre para terminar la ejecución del programa.

4.2.8 Comando "M" (Muestra la cadena en memoria).

Esta función se encarga de mostrar la "pregunta" contenida en la memoria para que el usuario pueda revisar si es exactamente lo que quiere.

Su funcionamiento será simplemente desplegar los 400 caracteres en los que está contenida.

4.2.9 Comando "O" (Obtención de información).

Con esta opción se ejecuta tanto la sintaxis como la semántica de la "pregunta" del usuario, y como resultado, se obtiene de la base la información solicitada. En caso de que la sintaxis no sea correcta, la semántica no es ejecutada.

4.2.10 Comando "R" (Reemplaza).

Este comando nos permite hacer ciertos reemplazos en la cadena de caracteres que tenemos en memoria. O sea que con el comando "R", podemos hacer pequeñas ediciones de la "pregunta" que originalmente cargamos en memoria.

Para efectuar el reemplazo, debemos leer la cadena o secuencia de caracteres que pretendamos reemplazar y aquélla que se quiera colocar en lugar de la original. El formato de la instrucción "R" es el siguiente:

R/cadena-1/cadena-2/

donde la "cadena-1" es la secuencia de caracteres que se quiere reemplazar, y la "cadena-2" es la secuencia de caracteres por la que se va a reemplazar la primera.

En la ejecución de la rutina vamos a tener tres casos: que la "cadena-1" sea del mismo tamaño que la "cadena-2"; que la primera sea mayor que la segunda, y que la segunda sea mayor que la primera. Además la primera acción es revisar si la "cadena-1" existe en la secuencia almacenada en memoria. En caso de existir se sigue adelante, pero de no encontrarla se despliega un mensaje alusivo y se termina el proceso de esta rutina.

Para el caso de que el tamaño de las cadenas sea igual, simplemente se sustituye una por la otra. Si la segunda es mayor que la primera, se insertan los caracteres necesarios, y si la primera es mayor que la segunda, se borran los caracteres sobrantes.

En caso de que se pretenda eliminar una serie de caracteres, se proveen éstos en la "cadena-1" y se omite la "cadena-2" poniendo dos diagonales consecutivas.

Al teclear la instrucción "R", se cede el control a la rutina de reemplazo que debe leer los caracteres que vienen en las dos cadenas posteriores y las diagonales de separación. En caso de que no se cumpla cualquiera de las condiciones para efectuar un reemplazo adecuado, se termina la rutina desplegando un mensaje alusivo.

Finalmente, antes de regresar el control de esta rutina, se hace una comparación entre el número de caracteres que tenía la cadena original, menos el número de caracteres de la "cadena-1", más el número de caracteres de la "cadena-2", contra el tamaño de la nueva cadena de caracteres que tiene la memoria. En caso de que no coincidan, se despliega un mensaje indicando que hubo algún error, y las cantidades involucradas. Si por el contrario las cantidades coincidieron, se envía el mensaje "Reemplazo efectuado correctamente" y se termina con la ejecución de la rutina.

4.2.11 Comando "S" (Ejecución Sintáctica).

Este comando se refiere a la verificación sintáctica de la "pregunta" que el usuario elabora. Para ello es necesario que antes de revisar tal sintaxis, se haya tecleado la "pregunta", se encuentre almacenada en memoria, y se limpien las áreas auxiliares mediante un llamado a una rutina de limpieza como la mencionada en el inciso 4.2.13.

Si el resultado de la sintaxis es aceptable, se enciende la bandera de sintaxis correcta y es posible ya ejecutar la "función semántica" tecleando "E".

El diseño detallado de SINTAXIS se encuentra en la sección 3 de este mismo capítulo.

4.2.12 Comando "T" (Tablas de enlace).

Este comando nos va a desplegar la información que contengan las tablas de enlace entre las rutinas de SINTAXIS y SEMANTICA, de acuerdo al número de restricciones para SELECT o selección, ejemplos para JOIN o enlace y atributos de impresión ó PRINT.

Cada vez que la función SINTAXIS es ejecutada, llena una serie de tablas (3) que contienen toda la información de la pregunta desglosada en: tabla de "elementos constantes", "elementos ejemplo" y "elementos de impresión". El contenido de tales tablas sirven como parámetros para ejecutar la función SEMANTICA.

El formato de impresión es irrelevante y lo único que se requiere realmente, es que toda la tabla (en cada caso) sea desplegada.

4.2.13 Comando "{".

El caracter llave "{" más que definir un comando, nos indica que lo que viene a continuación es la misma "pregunta" que quiere ejecutarse. Esto quiere decir que, al detec-

tar el caracter llave, el programa debe leer todos los que vienen a continuación hasta hallar el "line feed" ('\n'), indicándonos que en esa posición el usuario tecleó "RETURN" ó "ENTER". Entonces se ejecuta la rutina que leerá la pregunta completamente, cambiando todas las minúsculas en mayúsculas y almacenándola en memoria en un arreglo de 400 posiciones. A continuación se ejecuta una rutina que limpia las tablas y variables auxiliares, e inmediatamente después se revisa la sintaxis de la pregunta como si el comando de control hubiera sido una "S" (igual a lo especificado en el inciso 4.2.11), finalmente, se ejecuta la función semántica (igual a lo dicho en el inciso 4.2.6), sólo si la sintaxis fue correcta.

4.2.14 Generales

Además de las funciones especificadas en los puntos anteriores, necesitamos definir las variables necesarias tanto globales como locales para el buen manejo del supervisor.

4.3 ANALISIS SINTACTICO.

Uno de los puntos fundamentales para el proceso de recuperación de información es el "lenguaje de recuperación". Dicho lenguaje debe permitir al usuario obtener información de la base mediante cláusulas sencillas, sin importar el grado de complicación interno que se necesite, ya que la simplicidad hacia los usuarios es uno de los objetivos perseguidos.

El "lenguaje de recuperación" además de simple, debe ser poderoso para recuperar tanta información como se necesite. Para ello se diseñó un lenguaje inspirado en la "consulta según ejemplos" que permite de un modo rápido y fácil recuperar tanta información como se desee. Las normas o reglas sintácticas que rigen este lenguaje las llamamos "Sintaxis de recuperación" y al programa que revisa esas normas simplemente SINTAXIS.

En las páginas siguientes se detalla el programa SINTAXIS y las reglas "ortográficas" que lo rigen.

4.3.1 Generalidades.

Objetivo: Verificar la sintaxis de la "pregunta" de recuperación del usuario.

Entradas: Pregunta elaborada por el usuario grabada en memoria por el programa supervisor.

Sintaxis completa

<pregunta>::=<conJunto requisiciones>"."

<conJunto requisiciones>::=<requisicion>|<conJunto requisiciones><requisicion>

<requisicion>::="("<lista imagenes>")"
|"(MAPA= "<nombre mapa><lista imagenes>")"

<lista imagenes>::=<imagen> | <lista imagenes><imagen>

<imagen>::=<nombre imagen>" ["<lista argumentos>"] "

<lista argumentos>::=<argumento> | <lista argumentos><logico><argumento>

<argumento>::=<argumento1> | " P. "<argumento0>

<argumento0>::=<argumento1> | <atributo>

<argumento1>::=<atributo><argumento2> | <atributo><formula>
| <atributo><formula><argumento2>

<argumento2>::=" _ "<ejemplo>

<logico>::=" AND " | " OR " | " MINUS " | " "

<formula>::=<operador><constante>

<operador>::=" < " | " <= " | " > " | " >= " | " = "

<constante>::=cualquier valor numerico | cualquier valor alfanumerico

<atributo>::=valor definido en la base de datos como tal

<nombre imagen>::=valor definido en la base de datos como tal

<nombre mapa>::=cualquier valor alfanumerico

* Dentro de una imagen no se puede meter atributos locales y globales

* Solo la imagen inferior en una requisicion puede tener atributos locales

* Las imagenes deben ir en orden Jerarquico

* Si los campos alfanumericos son mayores a 12 posiciones o se incluyen en el espacios en blanco, se debe encerrar todo el campo entre apostrofos (').

* Los nombres <ejemplo> deben ser unicos en una requisicion

* Un atributo local no puede tener <ejemplo>

Figura 4.2

Sintaxis Completa

Salidas: Tablas de enlace para la función semántica y banderas para el supervisor.

4.3.2 Descripción del Proceso.

El programa SINTAXIS debe ser un programa recursivo(que permita llamarse a sí mismo), para verificar cada requisición del usuario, puesto que una "pregunta" está compuesta por un conjunto de "requisiciones" según se aprecia en la "sintaxis completa" definida en la figura 4.2.

Por tanto el programa debe recibir un parámetro, que le indica a partir de qué posición empieza a revisar la sintaxis de la "requisición" contenida en la "pregunta".

Como la sintaxis está totalmente definida, nos limitaremos a mencionar los aspectos que consideramos más importantes a realizar por el programa.

Según se aprecia, una "pregunta" está compuesta de una o varias "requisición" encerradas entre llaves "{", "}".

Cada una de ellas, puede empezar de dos formas: con las palabras "MAPA= nombre del mapa" o bien con el nombre de una imagen. El primer caso será tecleado si al usuario le interesa la impresión de un mapa que cumpla los requisitos que solicitó, y el "nombre del mapa" puede ser cualquiera que no contenga espacios en blanco intermedios, que mida hasta 12 caracteres de longitud y que contenga un espacio antes y otro después como límites de él.

Si la requisición empieza con el "nombre imagen" o bien, dicho nombre viene a continuación del "nombre del mapa", se debe comprobar que exista en el archivo de índices llamado GEOBASE/AI/XXXXX (donde "XXXXX" es el nombre de la base usada), mediante un llamado a la función "traepri", indicándole que se trata de una imagen. Si el resultado de la función es menos uno (-1) significa que ese nombre no existe en la base de datos, por lo cual se manda un mensaje alusivo, se enciende la bandera de error y se termina el proceso de SINTAXIS. Si el valor de la función es mayor a cero, significa que sí existe y se guarda la dirección de esa imagen como válida, continuando con el siguiente punto.

Después de la imagen debe haber un juego de paréntesis cuadrados([ ]), dentro de los cuales se pueden incluir de uno a 10 "argumentos" separados por un "lógico", según sea el caso.

Cada argumento contenido dentro de los paréntesis nos da las condiciones que deben cumplirse en una imagen para

recuperar cierta información. Nos indica también los resultados que se quiere imprimir o los posibles enlaces con otras requisiciones.

En general los argumentos están constituídos básicamente por "atributos", los cuales deben existir en el "archivo de índices" -mencionado anteriormente- mediante una llamada a la subrutina "traepri", pero indicándole que es un atributo el que le estamos enviando. Si el resultado es erróneo, se envía un mensaje alusivo terminando con el proceso.

El momento para verificar el atributo depende del uso que se le esté dando, pues podrá ser cualquiera de los tres siguientes:

- como argumento de selección,
- de impresión, o bien
- de enlace con otra "requisición".

Si es usado como argumento de selección significa que precede a una "fórmula", la cual está formada por un "operador" y una "constante", donde a su vez el "operador" puede ser cualquiera de los siguientes:

Operador	S i g n i f i c a d o
<	menor que
<=	menor o igual a
>	mayor que
>=	mayor o igual a
=	igual a

y "constante" es el valor contra el que se está comparando. Dicho valor puede ser numérico, alfabético o alfanumérico y puede medir hasta 12 caracteres. En caso de que ese valor incluya espacios en blanco, podrá venir encerrado entre apóstrofos (').

Si hay más de un argumento de selección, se separan éstos por un "lógico" que puede ser cualquiera de los siguientes:

Lógico	S i g n i f i c a d o
AND	Intersección de conjuntos
OR	Unión de conjuntos
MINUS	Diferencia de conjuntos

En caso de haber más de un argumento de selección y no haber "lógico" se asume "AND".

El número de argumentos de selección que existan en una "pregunta" se almacena en una variable entera, y las características de cada argumento en una tabla llamada de elementos "constantes", que contendrá el nombre de la imagen a la que pertenece, el nombre del "atributo", el operador, la constante, el lógico correspondiente, el número de requisición en el que se encontró, y la dirección que tiene para el esquema (valor obtenido de la rutina "traepri").

En cualquier caso de error, ya sea en el "atributo", en el "operador" usado, en la "constante" o en el "lógico", se reporta con un mensaje descriptivo y se termina el proceso.

Si un "atributo" es usado como elemento de impresión, debe ser precedido por los caracteres "P.", y a su vez podrá preceder algún "ejemplo". No hay que olvidar que para indicar un "ejemplo", se usa un guión bajo "_". Los argumentos de impresión también se cuentan, y se almacena en una tabla la imagen a la que pertenecen, el nombre del atributo, el número de requisición en el que se encontró, y la dirección del esquema.

En cualquier caso de error se reporta éste y se termina el proceso.

El último formato de los argumentos, es el de enlace. En este caso el formato es únicamente el nombre del "atributo", un guión bajo "_" y el ejemplo que nos sirve como enlace. Debe existir también una variable que especifique cuántos argumentos de éstos encontró, y una tabla para guardar la imagen a la que pertenece, el nombre del atributo, el número de requisición, y el ejemplo que tiene asociado.

Cualquier caso de error se reporta y termina con el proceso.

Una vez que se terminaron los argumentos de una requisición, se buscan los caracteres de terminación que son "]" y "}". A continuación puede venir un punto "." que termina con la "pregunta" u otra llave "{", lo cual nos indica que a continuación hay otra requisición y en este caso se debe llamar nuevamente a SINTAXIS en forma recursiva, pero dándole como parámetro la siguiente posición de donde se encontró el último caracter, o sea, donde empieza la nueva "requisición". Si alguna de las condiciones arriba mencionadas no se cumple, se despliega un mensaje explicativo, se enciende la bandera de error y se termina con el proceso.

4.3.3 Observaciones importantes.

La frase de "termina con el proceso", indica que se

suspende la fase sintáctica, pero que el control es cedido al SUPERVISOR para que el usuario pueda corregir su error y solicite nuevamente la verificación de la sintaxis.

El análisis sintáctico puede ser elaborado como un programa independiente o bien como una función o subrutina dependiente del programa SUPERVISOR, en cuyo caso es conveniente incluir las tablas y variables de control en modo global para que puedan ser accesados por cualquier otra rutina en cualquier momento.

Cuando se ha terminado la revisión de una "requisición" se despliega el mensaje "Requisición n correcta", siendo n el número secuencial de la requisición. En caso de que la "pregunta" sea sintácticamente correcta, el mensaje será "**SINTAXIS CORRECTA**". Una vez desplegado tal mensaje, regresamos el control al SUPERVISOR después de encender la bandera de sintaxis correcta.

Al decir que apagamos o encendemos alguna bandera o indicador, queremos decir que le pondremos el valor de cero (0) ó uno (1) respectivamente. Cuando preguntamos si alguna bandera está encendida, es porque comparamos la variable en cuestión con el valor numérico 1.

Para cualquier acción que sea ejecutada más de una vez en diferentes lugares del programa, conviene hacer una función o subrutina adicional, con el objeto de ahorrar espacio en memoria. Además cada subrutina debe traer comentarios con su nombre, objetivo, fecha de elaboración y autor.

La función "traepri" es una de las funciones auxiliares que obtiene datos de GEOBASE. Para más detalles, se puede consultar la sección 5 de este capítulo.

4.4 ANALISIS SEMANTICO

Una vez que la "pregunta" de un usuario es revisada sintácticamente y es correcta, es posible revisarla semánticamente, o sea, interpretar su contenido. Para ello se definió la función SEMANTICA, la cual se encarga de efectuar todas las operaciones requeridas directamente con la base de datos.

Para tal efecto, esta función debe tomar el control cuando el SUPERVISOR se lo ceda, después de revisar si la bandera de la "sintaxis correcta" está encendida.

Las actividades por realizar dependen de las tablas de

comunicación que fueron llenadas cuando se ejecutó la sintaxis, y el método en el que se basa el desarrollo del programa tiene sus fundamentos en el "Algebra relacional" descrita en el capítulo anterior. Los principios del "Algebra relacional" no son aplicados al pie de la letra; sin embargo, se desarrollaron rutinas y funciones similares para obtener la información solicitada de la base de datos.

A continuación se hace una descripción detallada del programa SEMANTICA, de los archivos que utiliza y de la forma en que debe hacer el llamado a las rutinas o funciones auxiliares.

4.4.1. Generalidades.

Objetivo: Comprobar la validez interpretativa del usuario al elaborar y ejecutar su "pregunta" para la obtención de resultados.

Entradas: Tablas de "elementos ejemplo", "elementos de impresión", "elementos de enlace", banderas del proceso y variables auxiliares.

Archivos: Esquema, Indices, Arbol cuaternario, Valores Locales, Taxonomías y Valores taxonómicos de la base de datos a la que el usuario se está refiriendo.

Salidas: Desplegado de la información que fue solicitada por el usuario, de acuerdo al contenido de la base de datos.

4.4.2 Descripción del proceso.

Al inicio del programa y para evitar accesos continuos a la base de datos (manifiestos en el tiempo de respuesta), se carga todo el esquema de la base a memoria.

La carga del esquema se efectúa a partir del archivo "GEOBASE/AE/nombre de la base". Dicho archivo contiene registros fijos de 48 caracteres.

El primer registro almacena información estadística del mismo archivo, indicando cuandos registros hay de cada tipo.

Una vez leído este registro (que es el registro "cero") vienen a continuación "n" registros de imágenes, "m" de atributos, "l" de taxonomías y "k de valores taxonómicos. Por lo tanto se leen los "n" registros de imágenes, después los "m" atributos, etc. La idea es tener todo el esquema cargado en memoria para evitar los accesos repetidos al disco.

Es conveniente guardarlos en un "estruct" global para que los pueda accesar cualquier rutina en cualquier momento, así como también es importante crear una subrutina que abra el archivo, cargue el esquema y lo cierre.

El formato de los registros en cada caso, está definido en el capítulo 2.

Una vez con el esquema en memoria, se arma la rama completa del árbol tomando como base la imagen de la última requisición, y subiendo en la estructura del esquema (por medio del padre) hasta llegar a una imagen igual a la contenida en la primera requisición.

Esto debe cumplirse únicamente para las requisiciones que tienen algún "atributo" con restricción (P.E. NOMRIO = LERMA o bien HABITANTES > 10000).

Para ello es conveniente generar una función cuyos parámetros sean la imagen inicial y la final; entonces la función busca en el esquema la imagen final para tomar el padre, y buscar ese padre entre las demás imágenes hasta encontrarlo, repitiendo la operación sucesivamente hasta llegar a la imagen definida como inicial. Cada vez que al buscar un padre lo encuentre, guarda el valor del índice en el que está contenida la imagen en el "estruct" de imágenes.

En caso de no encontrar la imagen inicial al subir de nivel, se señala el error de que una imagen no está contenida en la otra, y se hace el regreso de la función con un valor igual a cero.

Por otro lado, si se encuentra la imagen inicial, el valor de regreso de la función es el número total de imágenes encontradas (incluyendo las 2 de los parámetros). De este modo, si el valor final de la función es 1, sabremos que la "pregunta" está elaborada realmente sobre una sola imagen.

En conclusión, esta función lo que hace realmente es armar las imágenes intermedias entre la imagen superior e inferior de una pregunta (armamos una "rama" del árbol).

A continuación se debe asignar a cada uno de los elementos del arreglo: "elementos de impresión", y el índice del esquema en el que se encuentra cada atributo; o sea que cada atributo de cada requisición, se busca en el esquema y se guarda el índice correspondiente.

Asímismo, se calcula el índice de la imagen a la que pertenece, dividiendo el campo "apuntador del dueño" entre

48L para obtener el índice del padre en las imágenes del esquema. Una vez obtenido tal índice se compara si la imagen a la que apuntó es efectivamente la misma que definió el usuario en su requisición. En caso de no serlo, se imprime un mensaje alusivo y se enciende la bandera de error para dar por terminado el proceso.

Como puede verse, es conveniente hacer una función que realice todo lo anterior, es decir, que asigne el apuntador al atributo en la "tabla de definición de atributos", al dueño en la "tabla de definición de imágenes" y las tablas de apuntadores que se encuentren en forma global.

Una vez ejecutadas las tres funciones auxiliares anteriores, se procesa lo que es la semántica propiamente dicha, que consta de 3 etapas fundamentales: selección, enlace e impresión (correspondientes a las definidas en el capítulo 3 como select-project, join y print del Algebra relacional), y una auxiliar que se refiere a la generación de mapas. A continuación se describe cada una de ellas.

4.4.2.1 Selección.

La selección es la recuperación de la base de todos los atributos que cumplan una restricción de acuerdo a lo especificado en alguna requisición y consta de dos fases:

a) La fase hacia abajo.
b) La fase hacia arriba.

a) En la fase hacia abajo se van generando llamadas a ciertas rutinas a partir de la imagen superior, que tenga alguna restricción y tomando como base la "rama" creada (anteriormente mencionada), hasta llegar a la imagen inferior. La secuencia es la siguiente:

i) A partir de la imagen superior se hace un llamado a la rutina "sacalistasinpadre" con los atributos restrictivos que tenga, asumiendo que tal imagen está en el nivel i de la pregunta. Al archivo de salida lo llamamos "corto.i." Si la imagen es la inferior, aquí termina la fase hacia abajo; si no lo es hacemos k=i.

ii) Añadimos 1 a k (K=K+1) y revisamos si la imagen en ese nivel tiene restricción(es). En caso de tenerlas, continuamos en el punto iii, pero si no tiene, lo hacemos a partir del iv.

iii) Llamamos a la rutina "sacalistaconpadre" usando como padre el archivo "corto.k -1" poniendo el resultado en "corto.k". Continuamos con el punto v.

iv) Hacemos una llamada a "daimagenconpadre" usando como archivo de entrada o padre "corto.k -1", y como salida "corto.k". Continuamos con el punto v.

v) Si el tipo de imagen en el nivel k es el inferior, terminamos con la fase hacia abajo; pero si no lo es, continuamos con el punto ii.

La descripción genérica de las rutinas usadas en esta fase se encuentra en la sección 5 de este capítulo.

Es conveniente hacer notar que el bajar de nivel significa, ya sea revisar y agrupar la información correspondiente a la imagen tal como fue definida en la requisición (si es que existe y tiene alguna restricción), o bien obtener los elementos en una imagen, mediante el llamado a una rutina que sólo requiere de la imagen padre para la recuperación de esos datos.

También es de tomarse en cuenta que las rutinas "sacalistasinpadre" y "sacalistaconpadre", requieren de un arreglo de hasta 512 caracteres (bytes), en el que se incluyen las restricciones en notación polaca post-fija, pudiendo tener hasta 19 restricciones diferentes (que son de 26 caracteres), ligados con 18 operadores lógicos (de 1 caracter cada uno).

Toda la información requerida en el llamado a las rutinas, está contenida ya sea en la tabla de elementos constantes o en el esquema que cargamos a memoria. Hacemos una excepción con los números de archivo que deben ser obtenidos de acuerdo a como se van necesitando. El número de archivo depende de si existe ya, o apenas lo estamos creando. Para ello se hacen las llamadas a las siguientes instrucciones en "C":

```
num-ar = open(archivo, 2);   *
num-ar = creat(archivo,0777);
```

dependiendo de si el "archivo" existe o no respectivamente (como puede deducirse, la función "open" abre un archivo que ya existe y la función "creat" crea uno nuevo).

---

* La variable "archivo" es un arreglo en el que está cargado el nombre completo del archivo por usar y "num-ar" es un entero que guardará el número de archivo que asigna la máquina (identificador del archivo).

b) Una vez que terminamos con la fase hacia abajo, supongamos que la imagen del nivel inferior estaba en el nivel "m" y asumiendo que es la enésima requisición, la fase hacia arriba es:

i) Si el atributo inferior no es local se hace una llamada a la rutina "sacaregistrolargo" usando como archivo de entrada "corto.m" y como salida "largo.n" continuando con lo especificado en el punto iii.

ii) En caso de que el atributo inferior sea local, se llama a la misma rutina del punto anterior, pero la salida se deja en el archivo "momentáneo". Se llama entonces a la rutina "sacalocal" usando "momentáneo" como entrada y "largo.n" como salida. Se sigue adelante con los niveles superiores (que sólo tendrán atributos globales) como se indica en el punto iii.

iii) Se comprueba si el nivel en el que estamos es el superior. En caso de serlo, ahí termina la fase hacia arriba; si no es, se continúa con el punto iv.

iv) Se calcula n=n-1 para revisar si el siguiente nivel hacia arriba tiene atributos de interés. En caso de tener, se llama a la rutina "sacaregistrolargo" usando como entrada "corto.n" y como salida "largo.n". Se continúa el proceso con el punto iii aunque no se hayan encontrado atributos de interés en el nivel revisado.

Como puede apreciarse en la fase hacia arriba, sólo se efectúan llamadas a la rutinas "sacaregistrolargo" para todos los niveles que tienen atributos de interés globales y "sacalocal" para todos aquellos que tienen atributos de interés locales. Además, debe considerarse que para la fase hacia arriba también se requiere de la "rama" que se armó entre las imágenes inferior y superior, pero en sentido opuesto a la fase hacia abajo.

En la llamada a la rutina "sacaregistrolargo" se requieren 5 parámetros, de los cuales los 3 primeros son números de archivo. El 4o. se llama "longrec" y se refiere a la longitud máxima de registro contenido en el archivo TCI de la imagen a la que nos referimos, o sea, a la contenida en la requisición. Esta longitud se obtiene de la tabla de definición de imágenes (TDI) del esquema.

El 5o. parámetro se refiere al número de atributos globales que tendremos en una imagen determinada, y el 6o. es un arreglo que contendrá hasta 39 tercetas en las que pondremos información referente a los atributos de interés (aquellos que están en la requisición asociados con "P." ó bien con algún "ejemplo").

Para hacer la llamada a la rutina "sacalocal" necesitamos abrir los archivos "GEOBASE/QT/nombre de la base", "GEOBASE/RVL/nombre de la base" y crear un archivo de "mapa" que lleve ese nombre. Por otro lado, se requiere un arreglo de 512 caracteres que contenga todos los atributos locales que tienen restricción, en notación polaca post-fija.

Como se puede ver, habrá dos tipos principales de archivos: los cortos y los largos. Los primeros contendrán solamente las direcciones de las imágenes que cumplen con una restricción en 4 bytes (un "long" de "C"), y apuntan al archivo de TCI de la imagen correspondiente; mientras que los largos contienen -entre otras cosas-, todos los atributos de interés de una requisición.

El archivo "largo.n" está compuesto de "n" subregistros, uno para cada imagen de la "pregunta" o bien, uno para cada "requisición" que tenga atributo(s) de interés. Cada subregistro está a su vez compuesto de la siguiente información:

Tipo de dato	D e s c r i p c i ó n
long	Clave de la imagen a la que nos estamos refiriendo (apuntador de la imagen en AE).
long	Clave de la imagen padre de la que nos referimos (apuntador de la imagen padre en AE).
long	Clave del mapa (apuntador al QT de la imagen en cuestión).
"k" char	Atributos de interés de la imagen. Su longitud será la suma de todos los atributos numéricos por dos (2) más todos los que sean alfanuméricos por doce (12) o sea:

$$k = 2 * \#\text{atributos numéricos} + 12 * \#\text{atributos alfanuméricos}.$$

Esto quiere decir que un registro largo siempre tendrá 2 partes, una que es fija de 12 bytes (compuesta por 3 long), y una variable (compuesta por "k" caracteres), que depende del número de atributos de interés que contenga la imagen en una "requisición".

A tiempo de revisión de "atributos de interés" en la fase hacia arriba, se buscan todos los atributos por impri-

mir y se llena una serie de tablas auxiliares (para más detalles véase incisos anteriores).

4.4.2.2 Enlace.

La etapa de "enlace", se refiere a la comparación e igualación de los valores que contienen algunos atributos en diferentes imágenes. En nuestro recuperador, el "enlace" está definido por los "ejemplo" que se encontraron en la "pregunta" del usuario, o sea, que el número de ejemplos nos dirá la cantidad de elementos que contiene la tabla de "elementos ejemplo". Es de esperarse que no todos los ejemplos sean para enlace, por lo tanto el número de ejemplos en una pregunta está solo limitado por el tamaño de la tabla de "elementos ejemplo". Obviamente si el número de ejemplos en la tabla es igual a cero, significa que no hay "enlaces" por resolver.

El método que usamos para realizar los enlaces correspondientes es la semi-junta (sección 3.4), en la cual tenemos que hacer el proceso en los dos sentidos, o sea, de la imagen primera a la segunda y viceversa para obtener la información completa. Para ello nos valemos de un conjunto de funciones especiales que inclusive nos permiten romper ciclos cerrados.

En el proceso "enlace", tomaremos como entrada los registros largos que obtuvimos en la etapa de selección para hacer las semi-juntas. Una semi-junta se identifica por dos ejemplos iguales que estén en diferentes requisiciones (en diferentes imágenes), pudiendo formar enlaces cerrados o abiertos. Un enlace abierto es por ejemplo, aquél en donde una imagen "A" se enlaza con "B" y ésta a su vez con "C"; y un enlace cerrado, sería si en este ejemplo, "C" se enlazara nuevamente con "A". A continuación se describe el proceso para cada uno de los enlaces mencionados.

a) Enlace abierto.

Supongamos que son tres imágenes enlazadas (A, B y C) cuyos atributos de enlace son: entre "A" y "B", "a" y "b1"; y entre "B" y "C", "b" y "c1". El modo de ejecución, es mediante un llamado a la función especial "semijoin", dando como parámetros la identificación de los 2 archivos largos de entrada, la identificación del archivo de salida, la longitud de los registros de los dos archivos de entrada, el número de argumentos por comparar, y los datos específicos de cada argumento. Como resultado del llamado, tendremos un archivo largo que es un subconjunto del primer archivo de entrada.

El método en general consiste en reducir a su mínima expresión a todos los archivos largos que tienen enlace hacia atrás y hacia adelante. En nuestro ejemplo, el archivo de la imagen "A" se reduce con respecto a aquél de la imagen "B" y éste a su vez con el de la imagen "C"; por último se reduce "C" con respecto al archivo reducido de "B". De este modo tenemos los archivos largos reducidos de acuerdo a los valores de los atributos correspondientes.

El número de semi-juntas necesario para esta reducción es igual al número de imágenes involucradas.

b) Enlace cerrado.

En un enlace cerrado, la última de las imágenes tiene un enlace con la inicial. Para mostrarlo usaremos el mismo ejemplo del inciso anterior, pero supondremos que existe un enlace más que une a las imágenes "C" y "A" a través de los atributos "c" y "a1" respectivamente. El tratamiento es: romper el enlace entre las imágenes "C" y "A" pero no definitivamente, sino como preparativo de un proceso diferente.

El siguiente paso, es ejecutar el enlace como si se tratara de un "enlace abierto" hasta obtener la mínima expresión; a continuación se añade un atributo al resultado de la primera imagen, que depende del valor de los atributos de enlace en el registro largo. Dicho atributo será un número consecutivo, que se asigna a cada valor diferente a ser comparado con el siguiente archivo largo y será agregado al final de él. Para ello usaremos una de las funciones especiales llamada "aumentacampoaregistro" con la opción uno(1). Dicha función requiere del identificador del archivo largo de entrada y de uno similar como salida.

A las imágenes restantes, también se les añade el campo entero al final de sus registros largos, pero dicho campo debe quedar en cero. También para ello usamos la rutina "aumentacampoaregistro" pero con la opción cero (Ø), (la descripción detallada de la rutina está en el inciso 5 de este capítulo).

Una vez que añadimos el campo entero a todos los registros de los archivos resultantes, y que al primero lo numeramos de acuerdo al valor de sus atributos, trasladamos dicho valor a todos los demás archivos sobre su campo entero vacío valiéndonos de la rutina "asignadiscriminante", la cual lee el primer archivo de entrada, busca el valor del atributo ahí contenido en el segundo archivo de entrada, y cuando lo encuentra, traslada el valor entero de las dos últimas posiciones del primer archivo a su correspondiente en el segundo, actuando similarmente con todos los registros encontrados.

Cuando todos los archivos del enlace cerrado (recíen abierto) tienen el número trasladado, enlazamos (nuevamente) la última imagen con la primera a través de una semi-junta doble, o más bien dicho una semi-junta en la cual se revisa la existencia de dos atributos: el atributo de enlace entre las imágenes "C" y "A", y el atributo numérico que nosotros le asignamos al final de cada registro, que como se recordará es un entero. Con éste último paso, además de terminar con los enlaces, aseguramos el resultado obtenido.

El método presentado para romper enlaces es similar al definido por Wong (referencia 10 pp 226-233), para descomposición de "gráficas conectadas".

Otra opción que puede presentarse en un enlace abierto o cerrado, es cuando entre dos imágenes cualquiera hay dos lazos definidos o sea un enlace doble. Esta situación se resuelve mendiante una semi-junta doble, o sea una semi-junta en donde se revisan dos atributos a la vez.

Un detalle que debe tomarse en cuenta, es que si se detecta en un momento dado que el resultado de un "semijoin" es nulo, entonces el resultado final de la operación también es nulo, es decir, que no hubo registros que cumplieran las restricciones solicitadas.

Como el resultado de la operación enlace es un conjunto de archivos con información y formatos diferentes, el formato y número de cada uno de los archivos debe almacenarse en una tabla, para después imprimirse los resultados necesarios.

4.4.2.3 Impresión.

La impresión de resultados, está relacionada directamente con los archivos largos generados en los puntos "selección" y "enlace" previamente definidos, y obviamente son los atributos precedidos por los caracteres "P." encontrados en cada una de las requisiciones del usuario.

Para la impresión de resultados usaremos un conjunto de tablas, que deben ser llenadas cuando en la etapa de selección nos encontramos en la fase hacia arriba, a tiempo de revisión de atributos de impresión, o bien en el proceso de enlace. En ellas se debe incluir: la identificación del archivo largo, la longitud del registro contenido en él, y un arreglo de 234 caracteres que contenga hasta 39 tercetas de 6 caracteres con la siguiente información:

- Byte inicial del registro largo del campo a imprimirse.

- Byte final del mismo campo del registro largo.

- Un caracter '1' si el campo es numérico o un '2' si es alfabético.

Además debe tenerse un arreglo que nos indique cuántos atributos de impresión se encontraron en cada nivel; en caso de no haber en cierto nivel, tal información se omite.

Se repite entonces la rutina de impresión dependiendo del número de niveles que tengan atributos por imprimir.

La rutina de impresión debe hacer un llamado al subprograma "imprime", que es parte de las funciones auxiliares (definidas en 4.5).

4.4.2.4 Mapas.

La función de generación de mapas se ejecuta sólo cuando el usuario haya solicitado un mapa de los resultados. Para ello se hace un llamado al subprograma que se encarga del desplegado de los mapas cuyo nombre es precisamente "mapas".

Usualmente los mapas se encuentran en el archivo QT excepto en el caso de atributos locales que se guardan en "mapalocal". Por lo tanto para el desplegado de mapas debemos conocer la dirección de inicio del mapa y el identificador del archivo en el que está almacenado.

4.5 FUNCIONES AUXILIARES

Las funciones auxiliares están constituidas por un conjunto de subprogramas que efectúan el manejo directo de la base de datos -en algunos casos-, ó archivos de trabajo -en otros-.

4.5.1. Subprogramas.

Los subprogramas son enviados a procesar por un programa llamador que es el padre. A continuación damos una breve descripción de los subprogramas usados en el recuperador, su nombre de ejecución y su nombre largo (referencia 3).

Nombre largo: asignadiscriminante
Nombre de ejecución: asdiscr
Objetivo: traslada el campo agregado de enlaces cerrados a cualquier archivo largo. Como salida genera un nuevo archivo largo.

Nombre largo: aumentacampoaregistro
Nombre de ejecución: aumcmre
Objetivo: agrega un campo a un registro largo para romper enlaces cerrados.

Nombre largo: daimagenconpadre
Nombre de ejecución: dimgcpa
Objetivo: a partir de un archivo corto de padres, obtiene a todos sus hijos que sean imagen.

Nombre largo: daimagenes
Nombre de ejecución: daimagenes
Objetivo: Teniendo el apuntador a TDI, obtiene la dirección de la imagen en el archivo TCI.

Nombre largo: imprime
Nombre de ejecución: imprime
Objetivo: imprime los campos o instancias contenidos en un archivos largo en la terminal.

Nombre largo: mapa
Nombre de ejecución: mapa
Objetivo: A partir de una posición inicial y un archivo de mapas, despliega el mapa correspondiente.

Nombre largo: sacalistaconpadre
Nombre de ejecución: slistcpa
Objetivo: dado un criterio de selección y un archivo corto de padres, obtiene del archivo AI de la base utilizada la dirección de todos los hijos que cumplan con la restricción.

Nombre largo: sacalistasinpadre
Nombre de ejecución: slistspa
Objetivo: dado un criterio de selección sobre una imagen, obtiene la dirección en el archivo AI de la base utilizada y genera un archivo corto con tales direcciones.

Nombre largo: sacalocal
Nombre de ejecución: sacaloc
Objetivo: obtiene los registros largos del archivo RVL de la base utilizada, cuando los atributos de una imagen son locales solamente.

Nombre largo: sacaregistrolargo
Nombre de ejecución: sregl
Objetivo: obtiene el registro largo de todas las direcciones grabadas en un archivo corto, incluyendo todas las instancias encontradas (sólo atributos globales).

Nombre largo: semijoin
Nombre de ejecución: semijoin
Objetivo: realiza el enlace (semijoin), entre los atributos contenidos en dos archivos largos y genera un tercero con los resultados.

Nombre largo: sortea
Nombre de ejecución: sortea
Objetivo: dada una o varias longitud(es) y posición(es) inicial(es) de cierto(s) registro(s), clasifica un archivo en orden ascendente o descendente.

4.5.2 Llamadas externas.

El sistema operativo UNIX permite la ejecución de hijos a un proceso, o sea que un proceso en ejecución puede mandar ejecutar a otro mediante una relación de padre-hijo y el control del programa es cedido al hijo o devuelto al padre según sea necesario. Para ello efectuamos un llamado a un intrínseco de UNIX: "fork" y le debemos pasar todos los parámetros en forma de caracteres, puesto que UNIX inserta un blanco entre cada uno de ellos (referencia 3).

4.6 Operación del recuperador

Después que un usuario se da de alta en una terminal de la ONYX, teclea el nombre del recuperador y comienza la ejecución del SUPERVISOR, que es el que controla el manejo del sistema completo.

Se le pide al usuario el nombre de la base de datos que quiere usar, puesto que pueden existir varias versiones de la misma base. Una vez que el usuario tecleó el nombre solicitado, el SUPERVISOR abre el archivo AI de la base y verifica su existencia. En caso de no encontrarlo en ese número de cuenta, envía un mensaje de error y termina con el proceso, pero si el nombre de la base es correcto, da la bienvenida al usuario, le hace algunas indicaciones por si es la primera vez que accesa el recuperador, y le contesta con el indicador de la posición inicial de tecleo, que es " ".

En este momento se puede teclear alguna instrucción de control, o bien la pregunta que desea uno formular al sistema concretamente.

Las instrucciones de control son distinguidas por el primer caracter, sin importar lo que venga a continuación (excepción hecha con el comando "r" o "R" como se verá más adelante). El conjunto de instrucciones permitidas se ob-

tiene con la instrucción de control "c" ó "C".

En caso de hacer la pregunta inmediatamente que aparece el indicador inicial, ésta debe comenzar con una llave abierta "{" y su codificación será como se especifica en la sintaxis de recuperación (figura 4.2). Además, si el recuperador detecta que el primer caracter es una llave abierta, verifica automáticamente la sintaxis de la pregunta (mediante un llamado a la función SINTAXIS), y la almacena en memoria.

En caso de error, no es necesario teclear nuevamente la pregunta puesto que ya fue almacenada en memoria; podemos desplegarla con el comando de control "m" (muéstrala), editarla con el comando "r" (reemplaza), o borrarla con el comando "b" (bórrala). De estos comandos el primero y el último trabajan sin parámetros adicionales, pero en el reemplazo, se requiere la cadena de caracteres que se va a reemplazar y la cadena que los va a sustituir, separados por una diagnonal "/", la cual indica el principio o fin de cada una de las cadenas. En el reemplazo debe tenerse cuidado de no cambiar algo que no se desea. Esto se previene incluyendo en la "cadena 1", el número de caracteres necesarios (aún espacios en blanco), para discriminar adecuadamente la cadena que se desea cambiar. En cualquier caso, se puede usar como "cadena 2" la cadena nula, al incluir después de la "cadena 1" dos diagonales consecutivas, lo cual indica que la "cadena 1" se borra totalmente.

Una vez que la pregunta fue editada y corregida, se puede revisar su sintaxis nuevamente mediante el uso del comando "s", y cuando la sintaxis es correcta se puede "ejecutar" la pregunta con el comando "e", que hace la recuperación directamente de la base de datos.

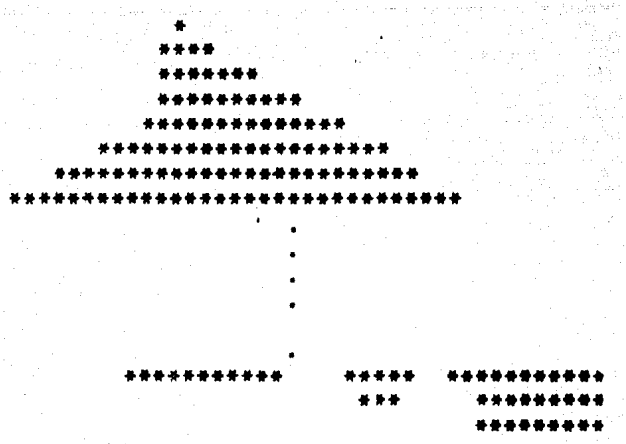
Al "ejecutar" una pregunta se hace un llamado a la función SEMANTICA, la cual puede detectar algunos errores de manejo de la base y contestar con un mensaje relacionado. Se puede entonces modificar la pregunta que no ha sido borrada de la memoria y revisar nuevamente su sintaxis, puesto que para ejecutar una "pregunta" es necesario verificar primero la sintaxis de ella. En caso de olvido de la "sintaxis de recuperación", se puede teclear el comando "a" el cual explica la notación en la que se definió la sintaxis, y después la sintaxis en sí misma.

Para terminar con la sesión de consulta, sólo se necesita teclear el comando "f", que finaliza con el sistema recuperador en cualquier momento.

En las páginas siguientes se muestran algunos ejemplos de recuperación con el sistema ya funcionando.

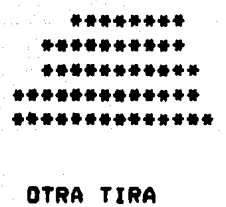
```
        Sistema recuperador de informacion para GEOBASE
                        Version 1.0

Deme el nombre de la base que desea accesar: prueba

   Si usted no ha utilizado este manejador con anterioridad,
 puede teclear la instruccion "(A)poyo" para desplegar la
 sintaxis de las preguntas que puede hacer o "(C)omandos" para
 revisar los comandos de control del programa. En caso contrario,

Teclee su <pregunta> o Comando de control.
>{pais [ p. nompais > ' ' ]}.
** SINTAXIS CORRECTA **

MEXICO
GUATEMALA

Teclee su <pregunta> o Comando de control.
>{pais [ p. nompais >  ' ' p. poblacion ]}.
** SINTAXIS CORRECTA **

  4587   MEXICO
   234   GUATEMALA

Teclee su <pregunta> o Comando de control.
>{pais [ nompais > ' ' ]}{estado [ p. nomestado > ' ' ]}.
Nivel correcto - 1
** SINTAXIS CORRECTA **

CAMPECHE
YUCATAN
TACNA
GUATEMALA
BELICE
CHIAPAS

Teclee su <pregunta> o Comando de control.
>{pais [ nompais = mexico ]}{estado [ p. nomestado > ' ' ]}.
Nivel correcto - 1
** SINTAXIS CORRECTA **

CAMPECHE
YUCATAN
CHIAPAS

Teclee su <pregunta> o Comando de control.
>{mapa = '* paises *' pais [ p. nompais > ' ' ]}.
** SINTAXIS CORRECTA **


MEXICO
GUATEMALA
```

MAPA - * PAISES *



OTRA TIRA

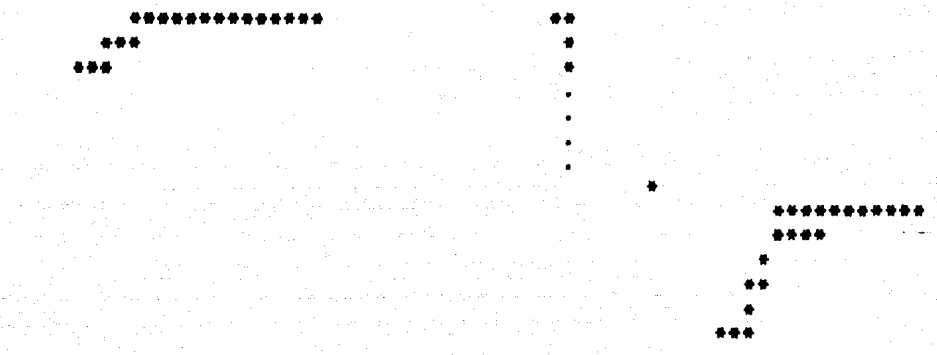
MAPA - * PAISES *



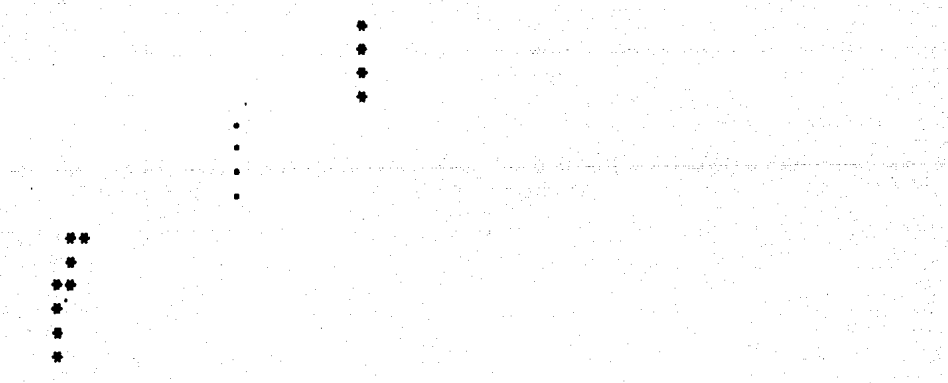
OTRA TIRA

```
Teclee su <presunta> o Comando de control.
>{pais [ nompais > '   ' ]}{rio [ p. nomrio > ' ' ]}.
Nivel correcto - 1
** SINTAXIS CORRECTA **

GRIJALVA
NUTU-TUN

Teclee su <presunta> o Comando de control.
>{pais [ nompais = mexico ]}{rio [ p. nomrio > ' ' ]}.
Nivel correcto - 1
** SINTAXIS CORRECTA **

GRIJALVA

Teclee su <presunta> o Comando de control.
>{rio [ p. nomrio > ' ' ]}.
** SINTAXIS CORRECTA **

GRIJALVA
NUTU-TUN
```

```
Teclee su <presunta> o Comando de control.
>{mapa= '* rios *' rio [ r. nomrio > ' ' ]}.
** SINTAXIS CORRECTA **

GRIJALVA
MUTU-TUN

MAPA - * RIOS *
```



```
 OTRA TIRA

MAPA - * RIOS *
```



```
 OTRA TIRA

Teclee su <presunta> o Comando de control.
>{pais [ nompais > ' ' _ ejemplo ]}{estado [ r. nomestado > ' ' _ ejemplo
Nivel correcto - 1
** SINTAXIS CORRECTA **

GUATEMALA

Teclee su <presunta> o Comando de control.
>f

Fin
```

C O N C L U S I O N E S

El sistema se elaboró en tres etapas sucesivas: en la primera se terminó el programa SUPERVISOR; en la segunda se le añadió una subrutina que se encarga de la revisión sintáctica (SINTAXIS) y finalmente se anexo la subrutina de recuperación propiamente dicha, llamada SEMANTICA. Actualmente se encuentra funcionando.

Debido à que sólo existe una palabra reservada en la "pregunta" del usuario, no hubo necesidad de un análisis lexicográfico en la revisión de ella; ya que todas las imágenes y atributos de la "pregunta" deben estar contenidas en el archivo de índices de la versión de la base que se este utilizando.

Dentro de las restricciones que tiene una microcomputadora, tenemos el reducido espacio de memoria para el alojamiento de programas de usuario, el cual fue resuelto gracias a la versatilidad del sistema operativo UNIX versión 7, en el que un proceso determinado puede ejecutar otro proceso en modo jefarquico (es decir en una relación padre-hijo), cediéndole el control de la máquina y recuperándolo una vez que el hijo terminó la tarea asignada. Esta asignación de tareas puede llevarse a efecto en varios niveles. De este modo los hijos pueden a su vez tener más hijos dando así mayor flexibilidad al sistema.

Dado que uno de los objetivos del recuperador es facilitar el manejo al usuario, se incluyen en el SUPERVISOR las funciones de "apoyo" y "comandos". En ellas se explica el funcionamiento del recuperador, evitandose así un voluminoso manual de operación, que generalmente el usuario no consulta a menos de que sea estrictamente necesario.

Por último, se espera que el recuperador sea usado por algunos usuarios, y que finalmente sean ellos los que decidan que tan sencillo o complicado es manejarlo y ayuden a mejorarlo.

REFERENCIAS BIBLIOGRAFICAS

1) An architecture for geographical data base systems.
R. Barrera, A. Buchmann, T. Rad Hakrishman
IIMAS-UNAM, Concordia University

2) Schema definition and query language for a geographical data base system.
R. Barrera A. Buchmann
IEEE Workshop on computer architectures for pattern analysis and image database management.
IIMAS-UNAM
1981

3) Definición del Sistema GEOBASE.
R. Barrera
IIMAS-UNAM
1982

4) Query by example a data base language.
M.M. Zloof
IBM Systems journal # 4 pp 324-343
1977

5) Query by example operations on hierarchical data bases.
M.M. Zloof
PROC. NCC pp 845-853
1976

6) Optimization of query evaluation.
S.B. Yao
Transactions on data base systems
Vol. 4 (ACM) pp 133-155
1979

7) Optimizing the performance of a Relational algebra data base interface.
M. Smith, Yeu T. Chang.
Communications of the ACM Vol. 18 No. 10
1975

8) Power of natural semi-joins.
A. Berstain, N. Goodman
SIAM Vol. 1o. No. 4
1981

9) An introduction to data base systems (3rd. Edition).
C.J. Date
Addison Wesley
1982

10) Data base systems.
J.D. Ullman
Computer science press
1980

11) Data base design.
G. Wiederhold
Mc. Graw-Hill
1977

12) IMAGE Data Base Management System.
Reference Manual
Hewlett-Packard Co.
1979

13) Computer Data Base Organization.
J. Martin
2nd. Edition
Prentice-Hall Inc.
1977

14) The C programming language
B. Kernighan
Bell Telephone Laboratories
1978