INCORPORACION DE OPERADORES DIFUSOS EN MANEJADORES DE BASES DE DATOS RELACIONALES

CARLOS MARIO ZAPATA JARAMILLO

UNIVERSIDAD NACIONAL DE COLOMBIA
SECCIONAL MEDELLIN
FACULTAD NACIONAL DE MINAS
ESCUELA DE INGENIERIA DE SISTEMAS
MEDELLIN
2.002

INCORPORACION DE OPERADORES DIFUSOS EN MANEJADORES DE BASES DE DATOS RELACIONALES

CARLOS MARIO ZAPATA JARAMILLO

Trabajo de Investigación presentado como requisito parcial para optar al Título de "Magister en Ingeniería de Sistemas"

Directora: CLAUDIA JIMENEZ
Magister en Ingeniería de Sistemas U. Nal. De Colombia

Co-Director: JUAN DAVID VELASQUEZ
Magister en Ingeniería de Sistemas U. Nal. De Colombia

UNIVERSIDAD NACIONAL DE COLOMBIA
SECCIONAL MEDELLIN
FACULTAD NACIONAL DE MINAS
ESCUELA DE INGENIERIA DE SISTEMAS
MEDELLIN
2.002

A Vicky, Juan Sebastián y Andrés Felipe, los motivos de mi esfuerzo incesante por ser cada día mejor.

Sin su apoyo incondicional este trabajo no habría podido llegar hasta el final.

AGRADECIMIENTOS:

El autor desea expresar su gratitud a:

CLAUDIA JIMENEZ y JUAN DAVID VELASQUEZ, Codirectores del Trabajo, por sus aportes y orientaciones constantes.

MADERAS DE OCCIDENTE LTDA. Por permitir el uso de su licencia de Fox Pro 2.6 bajo Windows y por facilitar el tiempo requerido para la realización de las diferentes actividades del trabajo.

DEMETRIO ARTURO OVALLE CARRANZA, por el apoyo incondicional y la oportunidad que me dio para ingresar en la Maestría.

ESCUELA DE SISTEMAS DE LA FACULTAD DE MINAS, por la beca que me permitió la culminación de mis estudios.

Todas aquellas personas que hicieron posible la realización de este trabajo.

CONTENIDO

LISTA DE FIGURAS

RESUMEN:

Los Manejadores de Bases de Datos Relacionales han contribuido a dinamizar el manejo profesional de los datos "precisos". Sin embargo, cuando surge la imprecisión o la incertidumbre, las búsquedas a través de los manejadores convencionales fallan y provocan dificultades en la recuperación de la información. Esta propuesta aborda el tema de la incorporación de operadores difusos en las búsquedas SQL convencionales, de modo que se puedan manejar términos lingüísticos que no poseen una equivalencia numérica "precisa" sino una aproximación a ella. Los operadores difusos definidos para este trabajo son sencillos y de fácil comprensión, y permiten emplear información de la base de datos para inferir ciertas características de los datos que no han sido explícitas por el usuario.

Palabras Clave: Búsqueda difusa, Operador difuso, Lenguaje de Búsquedas estructuradas (SQL)

ABSTRACT:

The Relational Data Base Management Systems have contributed to energize the "accurate" data professional management. However, when either imprecision or uncertainty emerge, query process through conventional management systems fails and it provokes difficulties on information recovery. This work is approaching to the issue about incorporation of fuzzy operators in conventional SQL queries, so that it could be handled linguistic terms with no "accurate" numeric equivalence, but with an approximation to it. Fuzzy operators defined for this work are single and easy understandable, and they permits to use database's information to deduce certain characteristics from data, unexplicitly mentioned by the user.

Key words: Fuzzy Query, Fuzzy Operator, Structured Query Language (SQL)

1. INTRODUCCION

Es innegable el carácter versátil e imprescindible que han alcanzado las bases de datos en la actualidad. Casi desde que nacemos, cada uno de nosotros comienza a ser parte integrante de innumerables Bases de Datos que recopilan nuestra información y la distribuyen para múltiples usos: comercial, estadístico, educativo, etc. Ha llegado a tal grado la utilización de herramientas de este tipo, que se hace prácticamente inconcebible la idea de encontrar empresas, entidades estatales, establecimientos comerciales, centros asistenciales o cualquier tipo de institución u organismo que no emplee en mayor o menor grado las bondades que de ello se pueden generar.

Sin embargo, y pese a los grandes avances que se han logrado en corto lapso para los Manejadores de Bases de Datos en términos por ejemplo de seguridad, confiabilidad y accesibilidad de la información, continúa sin una solución adecuada a lo largo de su evolución un aspecto que incide en la recuperación de la información contenida en las bases de datos actuales y su relación con los usuarios de las mismas: el carácter vago o impreciso que puede tener una búsqueda de datos realizada por un usuario desprevenido y carente de los conocimientos necesarios para dominar los lenguajes empleados en la elaboración de dichas búsquedas. ¿Cuántas veces en las empresas no se ha acercado un Propietario o Gerente en busca de los datos "más altos" de la facturación de los últimos años? ¿O indagando en relación con los empleados "más viejos" de la compañía?. En tales casos la respuesta del confuso Administrador del Sistema se limita a un "por favor, sea más específico, Doctor", a lo cual el desesperado Gerente o Propietario responderá: "¿Cómo así que más

específico? ¿Y es que ese aparato en el que tanto dinero hemos invertido no es capaz de sacar una simple respuesta a mi necesidad?".

Lo que el angustiado Gerente o Propietario no comprende y que el ahora también angustiado Administrador del Sistema intenta explicarle, es el hecho de que los Manejadores de Bases de Datos Relacionales de la actualidad presentan limitaciones en presencia de solicitudes de búsqueda vagas, imprecisas o simplemente "difusas", y que por ello, si quiere obtener alguna respuesta válida del sistema, deberá "precisar" los términos, de forma que el lenguaje que lo soporta pueda ser capaz de interpretar la necesidad para convertirla en un cierto número de tuplas que cumplan con el requerimiento inicial; en síntesis, que el sistema no entiende la pregunta porque no soporta "lenguaje natural".

El lenguaje natural está plagado de términos imprecisos para calificar ciertas características de los objetos, que reflejan la subjetividad de quien los está emitiendo y que de una forma u otra tienen mucho más significado que los valores precisos a los cuales se pueden también referir. Cuando se dice que una persona es "vieja", se pueden generar imágenes mentales inmediatas en quienes escuchan, de una intensidad mayor que si sólo se expresa la edad de dicha persona; en cierta forma es más expresivo, pues involucra el juicio subjetivo de quien lo emite. Esta forma lingüística se contradice, empero, con el tradicional esquema de almacenamiento de información mediante bases de datos, en las cuales los diferentes atributos poseen tipos claramente definidos en los cuales la imprecisión poca cabida tiene; es así como el atributo "edad" al que se refieren estas líneas iniciales tiene como tipo, por ejemplo, en una base de datos, "numérico sin decimales", para expresarlo en términos fácilmente almacenables. En las bases de datos tradicionales no tiene cabida una búsqueda imprecisa que busque entre sus tuplas las personas "viejas", empleando el término impreciso aludido, a menos que de alguna forma se pudiera definir el sentido de lo que el usuario quiere decir en términos numéricos o cuantitativos entendibles para la base de datos. Muchas han sido tradicionalmente las formas de abordar el tema

y, si bien ha habido avances, siguen existiendo dificultades aún insolubles, de las cuales este trabajo pretende entregar nuevas propuestas de solución.

1.1 JUSTIFICACION

1.1.1 Análisis de los trabajos previos.

Desde hace algunos años, los investigadores han formulado posibles soluciones al problema de la realización de Búsquedas Difusas sobre Manejadores de Bases de Datos Relacionales, y se han propuesto algunos trabajos. He aquí una revisión de la literatura disponible al respecto.

Existen básicamente dos tendencias en lo que respecta a la resolución del problema en mención: mediante bases de datos difusas (las cuales pueden poseer atributos de tipo intervalo o cualidad que permiten en cierta forma manejar la imprecisión del lenguaje natural) y mediante la incorporación de características difusas en los Manejadores de Bases de Datos Relacionales convencionales.

En lo que respecta al primer enfoque, en Chiang et al [8] se entrega un trabajo teórico con las ecuaciones fundamentales para el estudio de las Bases de datos Relacionales Difusas, en tanto que en Medina et al [9], [10] y [11] y Galindo [6] se da la definición de nuevos atributos para extender las bases de datos a bases de datos relacionales difusas. Estos últimos investigadores pertenecen al grupo ARAI (Aproximate Reasoning and Artificial Intelligence) de la Universidad de Granada y sus trabajos han pasado por el planteamiento teórico, llegando incluso a instancias de definición de conceptos y modelamiento en este campo. En Damiani et al [15] se analiza la programación orientada a Objetos en bases de datos relacionales

difusas y en Hale et al [16] la Minería de Datos y Seguridad en este tipo de bases, desde el punto de vista de la realización de búsquedas que puedan vulnerar la seguridad de las mismas. Finalmente, en Kumar et al [18] se realiza una caracterización teórica de los modelos de Manejadores de Bases de Datos Relacionales Difusos de otros autores.

Una de las críticas que se han realizado a esta solución en particular obedece a que existe en el momento en el mundo una gran cantidad de Bases de Datos que incluyen información "precisa", y con un planteamiento tal se tendría que adicionar unos tipos "difusos" a la información preexistente en la Base de Datos, con lo cual deberían coexistir dos tipos de datos para hacer la recuperación de la información contenida en ella, que podrían tergiversar de una u otra manera la información. Particularmente, en Medina et al [11] en su modelo GEFRED se definen otros tipos de datos diferentes a los datos "precisos" que se presentan en los manejadores de bases de datos convencionales, tales como: valores semánticos, distribuciones de posibilidad, valores aproximados e intervalos. Todos estos tipos de datos pueden coexistir en la base de datos difusa bajo el mismo atributo. Por ejemplo, se puede decir para el atributo "edad" de diferentes individuos de la base de datos lo siguiente: que es ocho veces posible que tenga 30 años y una vez posible que tenga 31 años (distribución de posibilidades), que es joven, intermedio o viejo (valores semánticos), que tiene aproximadamente 28 años (valores aproximados) o que esté entre los 30 y los 35 años (intervalo). Según lo plantea el modelo, es el usuario quien define qué tipo de dato tiene una tupla particular de la tabla para ese atributo, lo cual puede descalificar de paso una serie de operaciones que se podrían hacer en la base de datos, como por ejemplo ¿cuál es la diferencia de edades entre dos integrantes cualquiera de esa base de datos? Esta pregunta podría ser fácilmente contestada en caso de poseer los valores precisos, como en las bases de datos relacionales convencionales, pero en este caso, sólo se obtendrían respuestas aproximadas, perdiendo con ello precisión en la información misma. Así pues, se sacrifica en este modelo la precisión de la información con miras a poderla utilizar luego en forma imprecisa o incierta.

Igualmente, otra opción sería la actualización de la información para que tuviera características "difusas" para efectos de recuperación de la información mediante búsquedas, con la principal dificultad en las bases de datos de gran tamaño, en las cuales la información recopilada es de tipo preciso y requeriría un esfuerzo adicional considerable la conversión de la información a las características "difusas" requeridas. Esto significaría, volviendo al mismo ejemplo anterior, que se volvieran "difusas" las características del atributo para cada tupla, pasándola de un valor numérico preciso a alguno de los tipos difusos definidos, con el fin de realizar la búsqueda con base en estos nuevos tipos de datos. Una conversión tal no tendría sentido y más bien debería ser claro como lo es luego para Galindo [6], que más bien se debe trabajar en la realización de un lenguaje que amplíe el SQL incorporándole características difusas y no tratar de hacer una conversión como la planteada. El sistema desarrollado por Galindo puede trabajar con bases de datos relacionales convencionales o difusas y se convierte por ello en una herramienta versátil cuando no existe manera de traducir los datos precisos a difusos.

En lo relativo al segundo enfoque, Bosc y Pivert [13] y [14] han trabajado ampliamente en la extensión del lenguaje SQL para incorporar características difusas que permitan el manejo de la imprecisión. Además, Kraft y Petry [17] han realizado una síntesis de las dos tendencias en recuperación de información imprecisa.

Estas dos tendencias se basan siempre en la preexistencia de una función de pertenencia que permite la realización de las búsquedas en uno u otro modelo. Respecto de la función de pertenencia, Cox [30] explica lo siguiente: "Un conjunto difuso μ definido en un universo de discurso X se expresa por medio de su función de pertenencia: $\mu : X \rightarrow [0,1]$, donde el grado de pertenencia $\mu(x)$ expresa el alcance para el cual x llena la categoría descrita por μ. La condición $\mu(x) = 1$ denota todos los elementos que son completamente compatibles con μ. La condición $\mu(x)=0$ identifica todos los elementos que definitivamente no pertenecen

a μ". Expresada en estos términos, la función de pertenencia al conjunto difuso de los "jóvenes" tiene, por ejemplo, un valor de uno para personas con edades inferiores a 25 años y un valor de cero para personas con edades superiores a 35 años. Entre 25 y 35 años se tienen diferentes grados de pertenencia al conjunto de los jóvenes, variando entre 1 y 0. Nótese que en este caso particular se han escogido en forma subjetiva los valores máximo y mínimo de la función de pertenencia y esos valores pueden variar sustancialmente si se pregunta consecutivamente a varias personas por sus valores particulares para esa función; eso quiere decir que una persona bien podría decir que los valores que se han aludido para la función de pertenencia al conjunto difuso de los jóvenes podrían ser 20 y 50 respectivamente. Este es precisamente uno de los métodos empíricos que se han empleado para la determinación de la función de pertenencia, descrito en [32] y que consiste en la realización de una encuesta en relación con el término difuso buscado, por ejemplo el concepto "joven"; luego de realizada la encuesta, se divide el universo de las respuestas entregadas por los encuestados en intervalos iguales y finalmente se calcula la frecuencia con que cada uno de los intervalos del universo es invocado dentro de la muestra. Finalmente, se calcula la frecuencia relativa de cada intervalo dividiendo la frecuencia obtenida entre el número de encuestados.

Es allí donde surge la principal crítica a los modelos conceptuales y/o físicos realizados hasta el momento con este enfoque, puesto que la subjetividad de quien ejecuta la búsqueda se encuentra sesgada por el punto de vista de quien programa el modelo, quien de antemano ha debido suponer la función de pertenencia correspondiente. Por ejemplo, en [14] Bosc y Pivert suponen repetidamente en su modelo las funciones de pertenencia correspondientes a las soluciones de sus búsquedas difusas, sin mostrar claramente un método objetivo para el cálculo de dichas funciones. El modelo de Galindo [6], que también puede enmarcarse dentro de este tipo de soluciones, emplea los tipos de datos definidos para Medina et al [11] en GEFRED, pero el uso de los mismos debe ser alimentado en su estructura por el usuario, el cual debe suministrar los parámetros

correspondientes en el momento de realizar la búsqueda, sin emplear la subjetividad, pero dejando a consideración del usuario la definición de los parámetros fundamentales para la realización de la búsqueda. En caso de que quien realice la búsqueda no tenga los conocimientos técnicos suficientes en lógica difusa, la búsqueda podría devolver unos valores diferentes a los que el usuario efectivamente está buscando.

En cuanto a la complejidad de los operadores involucrados, los modelos mencionados poseen algunas características, como sigue: Bosc y Pivert [14] no definen operadores sino calificadores de los diferentes atributos, como el ejemplo de la búsqueda siguiente planteada en su trabajo:

Encuentre los empleados (número y nombre) de Chicago, que son bien pagados y relativamente jóvenes. La expresión en su lenguaje, llamado SQLf, tiene la siguiente forma:

Select #emp, e-name *from* EMP where city="Chicago" *and* salary="well-paid" *and* age="rather-young"

Si bien es un modelo simple de expresión de las características difusas, supone, como se mencionó previamente, la forma y valores de las funciones de pertenencia para cada búsqueda; además, no delimita el alcance ni la cantidad de los calificadores para los atributos. Por ejemplo, como se definió "rather-young" (más bien joven), el usuario podría estar interesado en las personas "relativamente viejas" o en aquellas "algo maduras". Esa falta de limitación en los términos puede agregar una complejidad grande al modelo, por cuanto se deberían tener internamente gran cantidad de calificadores con sus funciones de pertenencia definidas, con el fin de atender las posibles preguntas de los usuarios.

En el otro extremo, Galindo [6] posee un modelo denominado Fuzzy Query, con una serie de operadores que añaden gran complejidad a la sintaxis de la

búsqueda misma. Por ejemplo, devuélvame todas las personas con cabello justo (en grado mínimo 0.5) que son posiblemente más altas que la etiqueta $Alta (en grado mínimo 0.8), se expresaría así:

```
SELECT * FROM Person
WHERE  Hair FEQ $Fair THOLD 0.5 AND
 Height FGT $Tall THOLD 0.8
```

Una búsqueda con estas características no parecería emitida por un usuario convencional, puesto que se requiere por lo menos un conocimiento previo de la teoría de posibilidades y de la lógica difusa para entender términos como "grado mínimo" y "posiblemente más altas". Además, las abreviaturas con las cuales define sus operadores le introducen dificultades semánticas que nuevamente reafirman el hecho de que es necesario conocer del tema ampliamente para efectuar la búsqueda, pues no es completamente claro que FEQ es el operador difuso para igualdad posible y THOLD es el grado mínimo en el cual el atributo cumple con la etiqueta con la cual se está comparando mediante el operador difuso.

Un ejemplo práctico interesante lo constituye el SmartRanker [19], un trabajo sencillo programado en Java, que estuvo disponible en Internet para realizar simulaciones de ordenamiento sobre bases de datos sencillas con el empleo de operadores difusos. De nuevo, el autor presupone la función de pertenencia de las variables involucradas, las cuales nunca son preguntadas al usuario. En este trabajo el tratamiento de los operadores difusos es un poco más interesante, pues define en forma sencilla y esquemática los operadores que empleará para los cálculos con los cuales realiza su ordenamiento. Dichos operadores son los siguientes: * (no importa), >~ (un valor grande), <~ (un valor pequeño), =~ x (un valor cercano a x), ~~ y (una clasificación cercana a y); en todos los casos x es un número real e y es un número real en el intervalo [0, 1]. Para el último operador ~~ 0 significa que se prefiere un valor pequeño (por lo que es idéntico a <~), ~~ 1

significa que se prefiere un valor grande (por lo que es idéntico a >~), y ~~ 0.5 significa que se prefiere un valor del atributo cercano al valor medio de ese atributo. Llama la atención en este modelo la sencillez en la definición de los operadores y la pequeña cantidad de ellos, de modo que quien quiera usar dicho sistema tiene que adaptarse a los operadores disponibles.

En lo relativo al aspecto comercial, se debe resaltar el hecho de que Oracle [20] ha incorporado una serie de operadores para manejar las aproximaciones en las búsquedas, tales como CONTAINS, ACCUMULATE, STEM y SOUNDEX; además de los mencionados, ha incluido un comodín que genera expansiones difusas en las palabras, representado por el símbolo "?". Esto, sin embargo, constituye tan sólo una aproximación de menor cuantía a lo que se puede hacer con la incorporación de búsquedas aproximadas empleando operadores difusos más consolidados. También comercialmente se han realizado lanzamientos como el de Sonalysts Inc. [21] y [22], que promete solucionar el problema de las búsquedas aproximadas mediante un paquete de software que presenta un enfoque diferente para la determinación de la función de pertenencia y los calificadores de los atributos. En su software Fuzzy Query, cada término lingüístico debe ser definido por el usuario en forma cuidadosa, no sólo en los límites superior e inferior del término buscado, sino también en la forma de la función de pertenencia asociada con el término. Sin embargo, ¿qué pasa si el usuario, como es de suponer en la generalidad de los casos, desconoce los principios fundamentales de la lógica difusa? ¿Si el usuario no conoce cómo debe ser la función de pertenencia para un caso particular (recta, curva, trapezoidal, triangular, etc.) qué tan confiable puede ser la selección que haga de dicha función en el término difuso buscado?.

En la figura No. 1, se puede apreciar cómo la búsqueda correspondiente a los mejores alumnos de una institución se debe calcular mediante una variable GPA (promedio general del alumno), en la cual el conjunto difuso "bueno" (Fuzzy set name en la gráfica) para el promedio debe ser definido mediante una ardua

selección de la función de pertenencia (Fuzzy set shape) más ajustada a la pregunta subjetiva que se está realizando y los valores máximo y mínimo del dominio de esa función, que corresponden a los valores que el usuario cataloga como el máximo a partir del cual un promedio es siempre bueno y el mínimo antes del cual el promedio es definitivamente malo. Entre máximo y mínimo, no se puede dar un calificativo específico al promedio, sino que más bien se dice que tiene un cierto grado de pertenencia al conjunto de los buenos. Con los parámetros definidos, el programa realiza la gráfica que se muestra en la figura No. 1, en la cual el eje horizontal corresponde al promedio general del alumno y el eje vertical corresponde al grado de pertenencia del alumno al conjunto difuso de los "buenos" promedios generales del alumno. Esas características en el software de Sonalysts deben ser definidas por el usuario y de su elección dependerá cuáles tuplas van a ser devueltas por el sistema en la búsqueda que se plantea de los estudiantes con los mejores promedios.

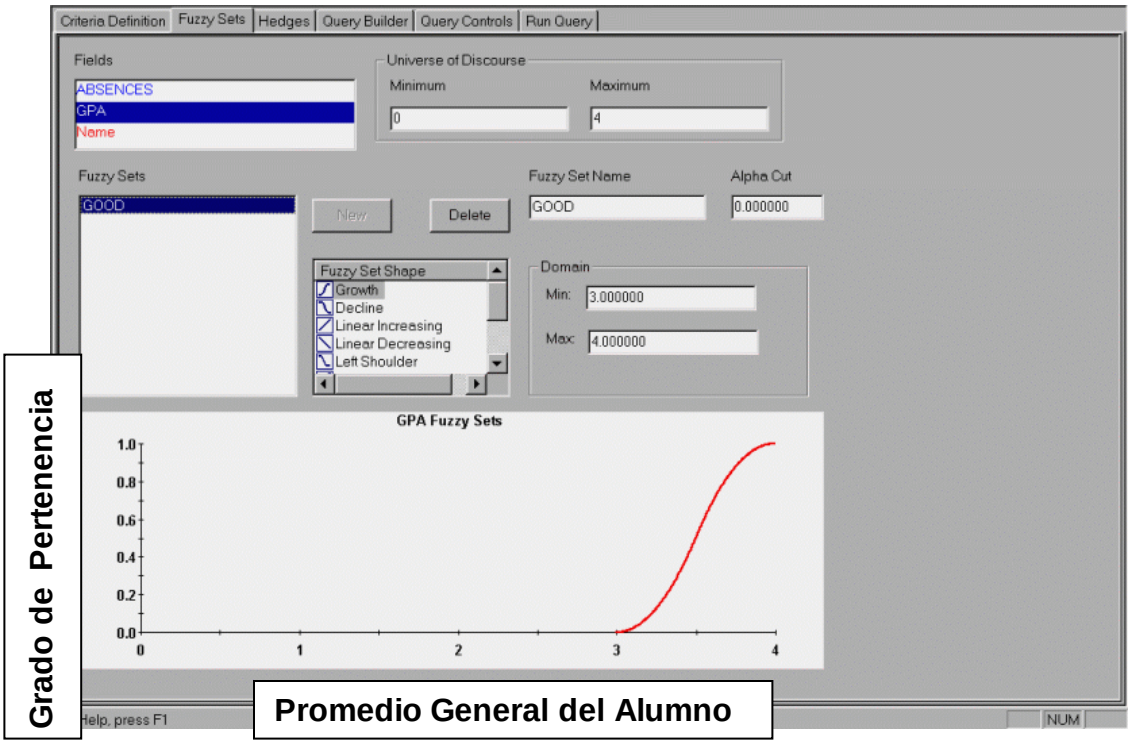

Figura No. 1. Imagen del Programa Fuzzy Query, de Sonalysts

1.1.2 Motivos para realizar este trabajo:

En la sección anterior se pudo apreciar una recopilación de los trabajos que se han realizado en esta materia y una discusión simultánea de los principales problemas que adolecen. Dichos problemas se pueden sintetizar así:

- La recuperación de la información se está realizando en algunos de los trabajos con el uso de funciones de pertenencia que deben ser definidas completamente por los usuarios en su forma y valores. Ello trae como consecuencia directa que los usuarios de este tipo de sistemas requieren una alta competencia técnica en lógica difusa para conocer las formas típicas de las funciones de pertenencia que se pueden ligar con los diferentes atributos de una base de datos, pues de ello depende que haga la selección adecuada a las necesidades de la búsqueda particular que está realizando. Otros modelos, en cambio, liberan completamente al usuario de la definición de la función de pertenencia asociada con un determinado atributo, pero tampoco realizan un cálculo objetivo de ella, suponiendo formas y valores que definen las funciones de pertenencia que se ajustan a la subjetividad de quien está programando los modelos.

- En algunos modelos, los operadores seleccionados por los diferentes sistemas y teorías que abordan el tema son complejos y requieren definiciones cuidadosas que faciliten el empleo de los mismos en búsquedas imprecisas simples que los usuarios puedan ejecutar. En otros, si bien los operadores obedecen a un parámetro de sencillez que hace entendible el lenguaje por parte de un usuario normal del mismo, la solución carece de límites definidos, dejando abierta para el usuario la posibilidad de emplear casi cualquier

calificativo que se le ocurra, generando una complejidad grande en la implementación del modelo mismo.

- Una de las vertientes de solución trabaja con bases de datos difusas, en las cuales se incorporan tipos de datos especiales que no tienen en general los manejadores de bases de datos relacionales convencionales. Ello dificulta el empleo de tales herramientas en manejadores convencionales, pues pueden poseer gran cantidad de información que, en general, no está almacenada en la forma que requieren dichos sistemas. Se plantea allí la discrepancia entre la conversión de los datos "precisos" a los nuevos datos "difusos" perdiendo la precisión de los tipos de datos tradicionales, o la coexistencia de los datos "precisos" con los datos "difusos" con las consecuentes inconsistencias que se puede generar en la información y la dificultad de realización de operaciones que combinen los dos tipos de datos.

Estos tres problemas hacen necesaria la implementación de un lenguaje más simple que permita la realización de búsquedas imprecisas por parte de los usuarios sin que requieran alta competencia técnica en el manejo de la lógica difusa o en herramientas similares en las que se puedan expresar búsquedas de este tipo y que a su vez solucione las dificultades que implica el cálculo de las funciones de pertenencia que sirven de base para la recuperación de la información solicitada por el usuario. Tales funciones de pertenencia deben partir de la información consignada en la base de datos para lograr que el cálculo sea adecuado a las características del atributo que se estudia. Además, y con miras a minimizar la complejidad del modelo, debe existir un número limitado de operadores que permitan atender gran cantidad de búsquedas que puedan ser solicitadas por los usuarios.

1.2 OBJETIVOS

1.2.1 Objetivo General:

Incorporar algunos operadores difusos en el lenguaje SQL estándar de un sistema manejador de bases de datos relacionales convencional, de modo que se pueda manejar algún tipo de información imprecisa para la realización de búsquedas e incrementar la capacidad del lenguaje, acercándolo más al lenguaje natural. Ello implica la complementación del SQL con unos operadores nuevos que enriquezcan el lenguaje y posibiliten las búsquedas con información imprecisa.

1.2.2. Objetivos específicos:

1.2.2.1 Elaborar un Modelo de Interpretación que permita la representación y comprensión de los operadores difusos y tipos de datos requeridos para mejorar la calidad del SQL en términos de su capacidad expresiva y simplicidad.

1.2.2.2 Convertir el Modelo de Interpretación en los Modelos Físicos apropiados.

1.2.2.3 Proponer un caso de estudio que permita la aplicación de los Modelos Físicos así desarrollados en la ejecución de consultas.

1.2.2.4 Validar el modelo usando un ejemplo práctico de una base de datos.

2. EXPOSICION DEL PROBLEMA

En la revisión bibliográfica se han hallado algunos modelos e implementaciones que responden al interrogante que plantea esta Tesis de Maestría en relación con los acercamientos a manejadores de bases de datos convencionales para extraer de ellos información que puede ser considerada como vaga, imprecisa o difusa. Sin embargo, en los modelos teóricos y prácticos hallados, e incluso en los comerciales, un asunto continúa sin hallar una respuesta satisfactoria: Si bien los conceptos difusos están en el pensamiento de la persona que va a ejecutar la búsqueda y en esta medida pertenecen a su subjetividad particular, ¿cómo podemos tener certeza de que los registros que devuelve una determinada consulta coinciden realmente con lo que el usuario quiere de ella?. El principal inconveniente en el uso de términos difusos es precisamente que la información no es lo suficientemente específica como para que se pueda expresar fácilmente en términos conocidos para los manejadores de bases de datos convencionales, lo que dificulta la recuperación de la información que puede ser valiosa para el usuario pues, al verse abocado a precisar los términos de la búsqueda, debe hacer simplificaciones y suposiciones en relación con los datos que pueden sesgar la información y conducirlo a conclusiones equivocadas. Si, por ejemplo, el asesor de una empresa necesitase preguntar a la base de datos de empleados por aquéllos que tengan salarios altos, en un manejador de bases de datos convencional debería precisar la búsqueda para definir, por ejemplo, que los salarios altos son aquellos superiores a dos millones de pesos; sin embargo, si la empresa es de mensajería, el asesor encontraría que hay muy pocos empleados que cumplan esa condición. Sin embargo, pueden existir salarios altos entre los mensajeros (por categoría o antigüedad, por ejemplo) que no alcancen los dos

millones de pesos y ello no se vería reflejado en la búsqueda por el sesgo que debió introducir el usuario.

Adicionalmente, los modelos que plantean una redefinición de los tipos de datos que pueden ser empleados en los manejadores de bases de datos convencionales con el fin de ampliar su cobertura al almacenamiento de datos difusos adolecen de una carencia de universalidad, puesto que existen actualmente grandes cantidades de bases de datos en el mundo administradas mediante manejadores relacionales convencionales y por ello cabría también la pregunta ¿qué hacer con los datos? ¿Se deben modificar registro por registro para incluir la información difusa que se está añadiendo? ¿Se debe hacer un corte en una determinada fecha y a partir de allí continuar llenando dicha información? ¿Qué tan confiables pueden ser las búsquedas en esas condiciones? En general los usuarios no están acostumbrados a manejar datos del tipo difuso, pues los diferentes atributos de las entidades tienen características que son definidas para los usuarios. En general se puede conocer con precisión la edad, la estatura o el salario de una persona; lo que se dificulta es la identificación de esa persona dentro de un grupo con los límites vagamente establecidos. En otras palabras, no es difícil saber que una persona mide 1.65 cm o que tiene 30 años; lo verdaderamente difícil es establecer si esa persona es alta o vieja y en qué grado. Es por ello que el almacenamiento de la información difusa, si bien puede ser un eslabón posterior en la cadena, no es una característica determinante para lidiar con la imprecisión del lenguaje común. Es más, la información almacenada en forma precisa entrega información importante para la elaboración de búsquedas difusas, puesto que sólo tendría que lidiar con la imprecisión del término buscado, en tanto que con datos imprecisos también se introduciría imprecisión por el tipo de almacenamiento de los datos. Siguiendo con el ejemplo, si en lugar de almacenar la estatura de la persona como 1.65 cm se almacena como "bajo" y resulta que esa persona está involucrada en la base de datos de los jinetes de un determinado hipódromo, lo más probable es que esa persona sea considerada como "alta" y que una búsqueda de los "altos" no lo devuelva como una instancia de dicha búsqueda.

Como se menciona en el análisis de los trabajos previos, trabajos teóricos como los de Bosc y Pivert [13] y [14] presuponen unas funciones de pertenencia para los diferentes atributos de las bases de datos que son conocidas por quienes programan el lenguaje de búsqueda, pero que no hacen partícipe al usuario de dicha definición. Además, no es claro cómo se obtienen dichas funciones de pertenencia y cómo se puede hacer para que sus valores tengan la objetividad requerida basada en el contexto sobre el cual ellos se mueven. Por ejemplo, si estamos hablando de la base de datos de los empleados de una empresa en Colombia o de una base de datos de los jugadores adscritos a la asociación nacional de baloncesto de Estados Unidos, el concepto de "alto" varía sustancialmente. La respuesta, en definitiva, no puede ser una función de pertenencia única para la determinación de los registros que cumplen con la condición solicitada en ambas bases de datos. Tiene que existir algún elemento que permita al usuario traducir sus necesidades al tipo de búsqueda que se está realizando, o que por lo menos le facilite la búsqueda sugiriéndole los valores que puedan ser más acertados a los términos que está buscando, sin que por ello deba hacer suposiciones o sesgar la información como se mostró antes.

La solución planteada por Sonalysts [21] y [22] es facilista e incómoda para el usuario normal que trata de aproximarse a una base de datos con la esperanza de no tener dificultades en la recuperación de la información que requiere. En general la teoría de la lógica difusa es poco conocida por usuarios normales de los sistemas y menos se conoce aún el tema de las funciones de pertenencia asociadas con un determinado atributo de una base de datos; ¿qué tipo de función escogería un usuario normal en relación con una pregunta como la planteada por Sonalysts? ¿Sería acertada esa selección? Por ejemplo, un usuario podría elegir una función de pertenencia decreciente para el GPA de la figura No. 1, afectando decisivamente la información a recuperar con la búsqueda y perfectamente podría hacerlo porque el modelo no se lo impide. ¿Quedaría conforme con esa elección? ¿Se daría cuenta de que algo extraño está ocurriendo porque lo que está recuperando de información no se adapta con lo que

intuitivamente está pensando recuperar? Definitivamente, una información tal podría conducir a equívocos y lo que se pretende con las búsquedas difusas es facilitar la aproximación del usuario convencional a la información mediante un lenguaje cercano al lenguaje natural y no complicar a tal grado la definición de la búsqueda que termine por confundirse y recuperar información que definitivamente no pertenece a lo que está buscando.

Dadas estas características para los modelos actuales, la inquietud se centra ahora en la definición de una solución que pueda aportar cierto grado de universalidad al asunto de las búsquedas difusas, pero sin perder la flexibilidad que requiere el usuario para la satisfacción de sus necesidades. No se pretende la incorporación de nuevos tipos de datos en los manejadores de bases de datos relacionales convencionales, por la incomodidad que ello puede representar para quienes ya poseen grandes cantidades de datos almacenados durante algún tiempo. Tampoco se pretende la definición en forma unívoca de las funciones de pertenencia de los términos difusos que se requieren para ejecutar una búsqueda o su simplificación a tal grado que no se puedan adaptar a las condiciones particulares de una base de datos específica, o que no puedan ser calculadas a partir de la información que posea la base de datos. Es por ello que se considera que la solución a implementar debe tomar en cuenta la información que está contenida en la base de datos, y que no requiere de tipos de datos especiales para la determinación de la información que está buscando el usuario; también, debe emplear un lenguaje accesible al usuario final y que permita el uso de interfaces amigables que contribuyan a facilitar la búsqueda. En el capítulo siguiente se puede apreciar el modelo como tal y las características que rodean su implementación.

3. SOLUCION PROPUESTA

La exposición del problema realizada en el capítulo anterior, permite abordar el objetivo general de esta tesis que consiste en la realización de un modelo de incorporación de operadores difusos en un manejador de bases de datos relacionales, el cual permita la recuperación de la información mediante la adición de operadores difusos al SQL, que enriquezcan su sintaxis y aproximen al usuario a la realización de consultas. No se pretende cambiar las estructuras existentes de la base de datos, ni tener que suponer de antemano otros datos que precisen la consulta, o que hagan que se requiera de alta competencia técnica para la realización de las búsquedas para recuperar piezas importantes de información.

3.1 PLANTEAMIENTO DEL MODELO

El modelo parte de la forma general de formulación de una consulta en SQL, la cual se puede consultar en [31]:

SELECT [DISTINCT] <atributos>
FROM <tabla o vista>
[WHERE <condiciones>]

En [31] se incluye un análisis completo de la sintaxis del SQL convencional y las diferentes condiciones que se pueden presentar en dichas búsquedas.

Para el modelo de búsquedas difusas, en <condiciones> se pueden incluir:

- Expresiones del tipo "preciso" o convencional de SQL o del tipo "difuso".

- Conectores difusos (~Y~, ~O~), cuya función principal es la separación de expresiones de tipo difuso, para realizar el cálculo de la función de pertenencia total de las expresiones unidas mediante el conector difuso. En [30] se incluye una definición de dicho cálculo, tal como lo estableció Zadeh al enunciar la teoría de la lógica difusa. Los símbolos $\wedge, \vee$, de las ecuaciones siguientes, corresponden respectivamente a ~Y~, ~O~.

  $\mu_{A\cup B}(x)=\text{máx}(\mu_A(x), \mu_B(x))= \mu_A(x) \vee \mu_B(x)$
  $\mu_{A\cap B}(x)=\text{mín}(\mu_A(x), \mu_B(x))= \mu_A(x) \wedge \mu_B(x)$

- Operadores difusos (~MUY GRANDE~, ~MUY PEQUEÑO~, ~GRANDE~, ~PEQUEÑO~, ~CERCANO A~, ~ALTO~, ~ALTA~, ~RAPIDO~, ~RAPIDA~, ~MUCHO~, ~MUCHA~, ~PEQUEÑA~, ~BAJO~, ~BAJA~, ~LENTO~, ~LENTA~, ~POCO~, ~POCA~, ~MUY ALTO~, ~MUY ALTA~, ~MUY RAPIDO~, ~MUY RAPIDA~, ~MUY PEQUEÑA~, ~MUY BAJO~, ~MUY BAJA~, ~MUY LENTO~, ~MUY LENTA~, ~MUY POCO~, ~MUY POCA~, ~CERCANA A~, ~ALREDEDOR DE~).

Para todos los operadores difusos se usa la notación posfijo, es decir, van después del nombre del atributo. Las condiciones son las siguientes para dichos operadores excepto ~CERCANO A~, ~CERCANA A~ y ~ALREDEDOR DE~:

<atributo> <operador difuso> [(<mínimo>,<máximo>)] [<número de tuplas a retornar>, <grado mínimo pertenencia>]

<mínimo> y <máximo> son los valores sugeridos por el usuario para el establecimiento de la función de pertenencia del operador difuso, que

corresponden a los valores entre los cuales es difusa la definición de los límites del concepto (por ejemplo, para el atributo "edad" <mínimo> sería el valor por debajo del cual definitivamente la persona es joven y <máximo> sería el valor por encima del cual la persona se considera definitivamente vieja). <número de tuplas a retornar> y <grado mínimo pertenencia> permiten restringir la cantidad de tuplas que se recuperan mediante el uso del modelo y son valores opcionales también.

Como los valores entre corchetes son opcionales, el usuario tiene la opción de incluirlos o no dentro de la sintaxis de la búsqueda. En general, un usuario convencional no querrá explicitar dichos valores y dejará a consideración del modelo la determinación de los grados de pertenencia correspondientes a cada búsqueda y finalmente el grado de pertenencia total de la búsqueda. Bajo esta consideración, si los valores <mínimo> y <máximo> son omitidos por el usuario al plantear la búsqueda, se calcula la distribución de frecuencias relativas de posibilidades a partir de la información consignada en la base de datos.

Surge entonces el interrogante ¿Por qué usar la distribución de frecuencias relativas de posibilidades para la determinación de la función de pertenencia de un determinado atributo?

Cooman [38] expone lo siguiente: "Muchos científicos prominentes han defendido la idea de que la incertidumbre y la aleatoreidad no son únicas ni la misma cosa. Más sucintamente, la incertidumbre puede ser causada por más que el azar en sí mismo. En 1978, Zadeh observó en su artículo semanal de teoría de la posibilidad, que la gente a veces transmite información usando oraciones afirmativas simples en lenguaje natural del tipo sujeto - verbo - predicado, donde el predicado nos dice algo sobre el sujeto, o para formularlo en una manera matemática, impone una restricción sobre los valores que el sujeto puede asumir en un apropiado universo del discurso. La información transmitida o representada de esta manera podría llamarse 'información lingüística'. En la mayoría de los casos en lenguaje natural, sin embargo, la información contenida en tales

oraciones no es suficiente para determinar el sujeto inequívocamente, simplemente porque el predicado involucrado es impreciso o vago. Para dar un ejemplo, la proposición 'la edad de John está entre 20 y 30 años' nos da alguna información sobre qué tan viejo es John, pero no determina completamente su edad, dado que el predicado 'entre 20 y 30 años' es impreciso. Similarmente, cuando decimos 'Mary es alta' entregamos alguna información sobre la estatura de Mary, pero puesto que 'alta' es un predicado vago, esta información no es suficiente para determinarlo completamente".

"En ambos casos, si bien la sentencia envuelta contiene información, nos deja alguna incertidumbre mientras que el valor preciso que el sujeto de la oración asume en su universo del discurso. Hemos también tropezado con una clase de información, en adelante llamada incertidumbre lingüística, que no puede estar directamente atribuida a la aleatoreidad o a la oportunidad, y la cual está siempre presente en nuestra comunicación normal".

"Zadeh ha avanzado la tesis de que información lingüística, o dualmente, incertidumbre lingüística, no tiene nada que hacer con las probabilidades y no puede ser representada por medidas probabilísticas. Para proveer una representación matemática para esta incertidumbre, introdujo las medidas posibles, variables difusas y sus distribuciones de posibilidad, produciendo medidas posibles y la noción de interactividad Además, coloca las bases de una teoría de la posibilidad, una colección de nociones y resultados concernientes a las medidas de posibilidad, y, según Zadeh, negociando con la descripción matemática de incertidumbre lingüística. El trabajo de Zadeh ha sido además refinado y extendido por un número de científicos. Sin pretender ser lo más completo, quiero mencionar a Dubois y Prade y varios colaboradores, quienes entre otras cosas introdujeron la noción dual de una medida de necesidad, estudiaron las relaciones entre medidas de necesidad y otras representaciones de incertidumbre y más recientemente presentaron un estudio de la independencia posibilística, de acondicionamiento en una estructura posibilística, estudiaron

formas de convertir medidas de posibilidad en medidas de posibilidad y viceversa, estudiaron las relaciones entre teoría de posibilidad y la teoría Dempster - Shafer de la evidencia y exploraron la relación entre teoría de posibilidades y lógica modal y también contribuyeron al sujeto de la posibilidad condicional".

"Se debe señalar que hay otras interpretaciones de medidas de posibilidad diferentes a la dada por Zadeh. Por ejemplo, se ha notado que las medidas de posibilidad son tipos especiales de medidas de creencias y probabilidades superiores, y que pueden ser interpretadas además dentro de la estructura de la teoría de Dempster - Shafer de la evidencia y de las probabilidades imprecisas, respectivamente. Giles hizo un intento de interpretar las medidas de posibilidad como alguna clase de probabilidades superiores. Dubois y Prade también estudiaron interpretaciones de la teoría de posibilidad en relación con funciones de probabilidad y con estados epistémicos".

En [25] se establece lo siguiente: "El análisis de intervalos, comúnmente aplicado en física, meramente sirve para tirar las imprecisiones de los instrumentos de medida, en la forma de intervalos, de vuelta a las magnitudes estimadas por las medidas. En términos matemáticos, uno evalúa la imagen de una función cuyos argumentos son subconjuntos. El análisis de intervalos no tiene gradaciones: mientras uno no conozca el valor exacto de un parámetro, no conoce los límites exactos de su dominio de variación. Nótese que, dada una medida imprecisa M de la magnitud X, las proposiciones del tipo 'X pertenece al intervalo I' pueden ser calificadas naturalmente por las modalidades de 'posible' y 'necesario', así":

"1. Si $M \cap I$ no es vacío, entonces '$X \in I$' es posiblemente cierto".

"2. Si $M \subseteq I$ entonces '$X \in I$' es necesariamente cierto".

"Aquí enfatizamos las relaciones entre estas modalidades y la teoría de conjuntos: lo posible se evalúa respecto de la intersección de la teoría de conjuntos entre los

contenidos M e I de las dos proposiciones 'X $\in$ M' y 'X $\in$ I'; lo necesario se evalúa respecto de la inclusión de la teoría de conjuntos".

Ahora, respecto de las medidas de confianza, Dubois y Prade [25] continúan así: "Consideremos un conjunto de eventos asociados con un cuerpo de conocimiento incierto e impreciso y considerado como subconjuntos de referencia del conjunto Ω, llamado el 'evento (siempre) seguro'. El conjunto vacío es identificado como el 'evento (siempre) imposible'. Se supone que con un evento A $\subseteq \Omega$ se asocia un número real g(A) proporcionado por un individuo que posee el cuerpo (o por un procedimiento de procesamiento de datos aplicado a la información en memoria de un sistema de cómputo). g(A) mide la confianza que uno puede tener en la ocurrencia del evento A, tomando el estado del conocimiento en cuenta. Por convención, g(A) aumenta cuando aumenta la confianza. Además, si A es un evento seguro, entonces se toma g(A)=1, y si A es un evento imposible, entonces g(A)=0, en particular: g($\varnothing$)=0 y g(Ω)=1".

"No obstante, g(A)=1 (o 0) no necesariamente significa que A es seguro (o imposible)".

"El axioma más débil que uno podría concebir para asegurar que el conjunto función g tiene un mínimo de coherencia es aquél que sería monótono respecto de la inclusión: A $\subseteq$ B $\Rightarrow$ g(A) $\leq$ g(B) (Ecuación 1)".

"Este axioma significa que si el evento A implica otro evento B, entonces uno siempre tiene al menos tanta confianza en la ocurrencia de B como en la ocurrencia de A".

"Tal conjunto de funciones fue propuesto por Sugeno para la evaluación de la incertidumbre, bajo el nombre de 'medidas difusas'. A. Kaufman sugirió el nombre de 'valoraciones'. Aquí adoptaremos el término 'medidas de confianza'. Es

apropiado anotar que no hay medidas en el sentido aditivo usual, a menos que se indique expresamente".

"Cuando Ω es un conjunto de referencia infinito, uno puede introducir axiomas de continuidad escritos como sigue: para cada secuencia anidada $(A_n)_n$ de conjuntos $Ao \subseteq A1 \subseteq \ldots An \subseteq \ldots$, o $Ao \supseteq A1 \supseteq \ldots An \supseteq \ldots$, tenemos":

"$\lim_{n\to+\infty} g(A_n) = g(\lim_{n\to+\infty} A_n)$"

"Se supone que una medida de confianza satisfará esta ecuación para al menos uno de los dos tipos de secuencia (incremental o decremental)".

Siguiendo esta línea de análisis, Dubois y Prade llegan a las siguientes ecuaciones que muestran la relación existente entre probabilidad (P), necesidad (N) y posibilidad (Π):

$$P(A)+P(\tilde{A})=1$$
$$N(A)+N(\tilde{A})\leq 1$$
$$\Pi(A)+\Pi(\tilde{A})\geq 1$$

Y llegan a una clase de medidas de probabilidad P, tal que:

$$P=\{P|\forall A,\ N(A) \leq P(A) \leq \Pi(A)\}$$

Igualmente, demuestran que a partir de las distribuciones de probabilidad se pueden calcular las correspondientes distribuciones de posibilidades, de la siguiente manera:

$$\pi(\omega_i)=\sum_{j=1}^{n} \text{mín}(p(\omega_i),\ p(\omega_j))$$

Además afirman que: "Cuando una medida de posibilidad tiene valores en el intervalo unitario, uno puede por lo tanto interpretar su distribución π como la función de pertenencia de un conjunto difuso F". Pedrycz [30] coincide con esta afirmación cuando establece que "el asunto de la estimación de las funciones de pertenencia puede ser dirigido construyendo algunos mapas biyectivos entre probabilidades discretas y grados de pertenencia (transformaciones probabilidad - posibilidad). Consideramos el desarrollo de funciones de pertenencia haciendo uso de las funciones de distribución de probabilidad fundamentadas en datos experimentales disponibles". Y continúan diciendo "por inspección, la función de pertenencia y la correspondiente distribución de probabilidad tienen la misma forma". Terano, Asai y Sugeno [29] coinciden también con esta apreciación cuando afirman: "Una explicación para los conjuntos difusos es que la función de pertenencia puede ser vista como una distribución de posibilidad".

Es claro, entonces, que calculando la distribución de posibilidades se consigue la función de pertenencia a una determinada búsqueda difusa.

Basados en este análisis, Roux y Desachy [12], estudiando el proceso de reconocimiento de imágenes por satélite, propusieron un método práctico para el cálculo de la distribución de posibilidades de un conjunto difuso a partir del histograma de frecuencias, con el fin de fortalecer el peso de las frecuencias más representativas de los intervalos del histograma, permitiéndoles ser más distinguibles unos de otros.

Un ejemplo práctico de dicho método de cálculo de la distribución de posibilidades es el siguiente:

Se tiene el Histograma de frecuencias que se muestra en la figura No. 2. Se tienen en total 2+7+14+17+17+8+3 = 68 pixeles.

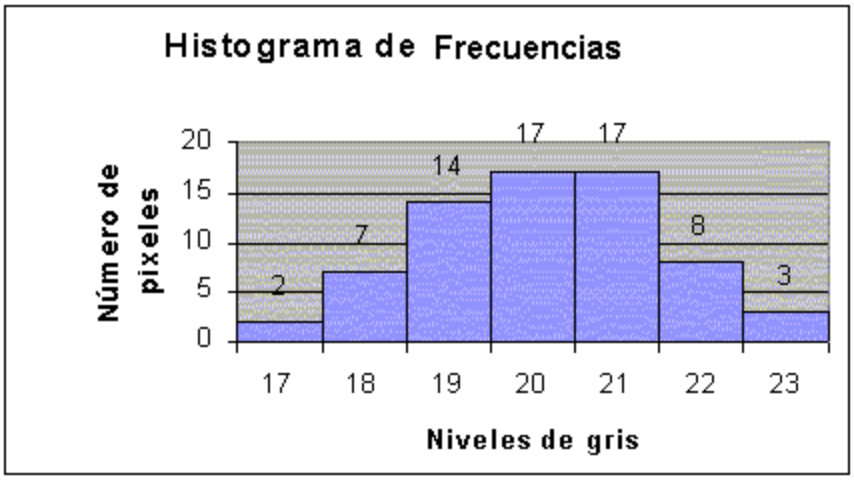

Figura No. 2: Histograma de frecuencias de un ejemplo particular.

La distribución de posibilidades en cada caso se calcula así:

$$\pi_s(Ci/17) = \frac{2+2+2+2+2+2+2}{68} = \frac{14}{68} = 0.21$$

$$\pi_s(Ci/18) = \frac{2+7+7+7+7+7+3}{68} = \frac{40}{68} = 0.59$$

$$\pi_s(Ci/19) = \frac{2+7+14+14+14+8+3}{68} = \frac{62}{68} = 0.91$$

$$\pi_s(Ci/20) = \frac{2+7+14+17+17+8+3}{68} = \frac{68}{68} = 1.00$$

$$\pi_s(Ci/21) = \frac{2+7+14+17+17+8+3}{68} = \frac{68}{68} = 1.00$$

$$\pi_s(Ci/22) = \frac{2+7+8+8+8+8+3}{68} = \frac{44}{68} = 0.65$$

$$\pi_s(Ci/23) = \frac{2+3+3+3+3+3+3}{68} = \frac{20}{68} = 0.29$$

La gráfica resultante para la distribución de posibilidades se muestra en la figura No. 3.

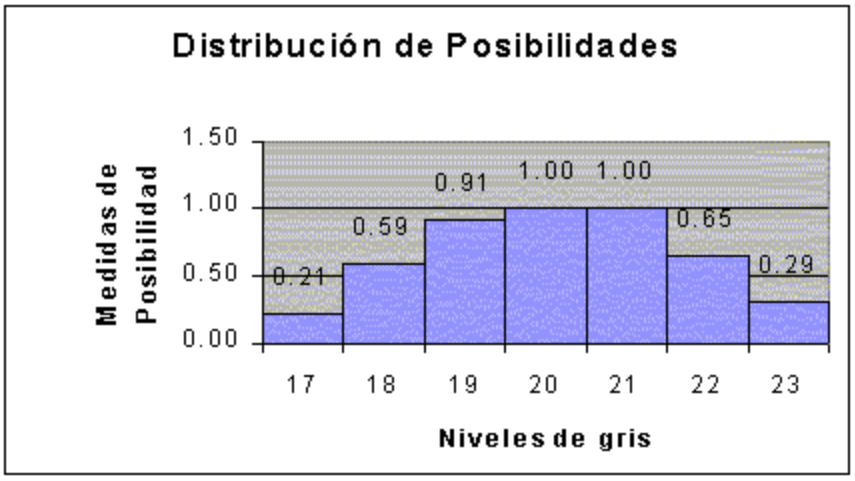

Figura No. 3: Distribución de Posibilidades para el mismo ejemplo.

Finalmente, se realiza el cálculo de la Distribución de posibilidades relativas acumuladas, que es la que permite determinar la forma de la función de pertenencia al conjunto difuso definido, como se muestra en la figura No. 4.

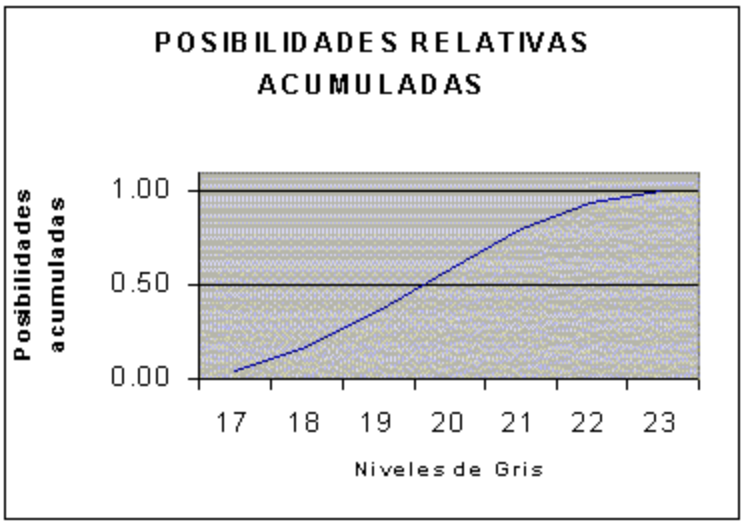

Figura No. 4: Distribución de Posibilidades Relativas Acumuladas del ejemplo

Existen igualmente en la literatura otras técnicas para manejar este tipo de datos, particularmente KDD (Knowledge Discovery of Databases), la cual, con su principal técnica de adquisición de conocimientos, Data Mining, puede realizar

ciertas búsquedas que identifiquen las relaciones internas de los datos de una base de datos. Dichas técnicas, entre las cuales se cuenta el clustering, se describen en [36] y [37]. El problema es que entre mayor sea la cantidad de atributos de la base de datos, mayores posibilidades existirán de que una explosión combinatoria de los datos pueda presentarse, lo cual implica que en bases de datos que tengan muchos atributos, existe gran cantidad de combinaciones posibles de los atributos que hacen casi imposible la agrupación de los datos y la consecuente relación entre ellos. Por ejemplo, en [36] se demuestra cómo una tabla con ocho atributos puede agruparse en 4140 formas, y no estamos hablando de un número grande de atributos. Además, las técnicas de clustering se ocupan de las relaciones entre los atributos para formar clases que, como vemos, pueden ser muy numerosas y tendrían que ser definidas en su totalidad para compararlas en un momento dado con una búsqueda de tipo difuso que plantee el usuario; por ejemplo, si en una búsqueda de los empleados de una compañía nos interesan los que tienen salarios altos y son jóvenes, tendría que existir una clase definida en esos términos para agrupar los diferentes datos y determinar hasta qué punto una tupla particular cumple los requerimientos de la búsqueda. El método de cálculo de la distribución de posibilidades acumuladas presenta mayor consistencia en este caso, puesto que, si el tamaño de la base de datos es suficientemente representativo, dicha distribución reflejará adecuadamente las relaciones entre los datos para un atributo particular, sin tener que recurrir a la definición de clases o grupos que manejen la información.

Volviendo al planteamiento del modelo, cuando no se suministran los datos correspondientes a <mínimo> y <máximo>, el modelo realiza el siguiente método para el cálculo de la función de pertenencia correspondiente:

- De la tabla o vista se toma el atributo que está siendo consultado mediante el operador difuso.
- Se busca el valor máximo y el valor mínimo entre todas las tuplas.
- Se divide el intervalo en diez partes iguales.

- Se calculan las frecuencias correspondientes a cada intervalo.
- Se construye la distribución de posibilidades como se describió según el trabajo de Roux y Desachy [12].
- Se calcula la distribución de posibilidades relativa y se acumula (ello con el fin de garantizar que los valores de la función de pertenencia nunca sean superiores a uno). Esta es la función de pertenencia definitiva, pues, como se justificó en las líneas previas, toda distribución de posibilidades hace las veces de función de pertenencia para un atributo particular de la tabla o vista.

Con el método descrito se construye la distribución de frecuencias relativas de posibilidades acumuladas para los operadores difusos ~GRANDE~, ~ALTO~, ~ALTA~, ~RAPIDO~, ~RAPIDA~, ~MUCHO~, ~MUCHA~. Para los operadores ~PEQUEÑO~, ~PEQUEÑA~, ~BAJO~, ~BAJA~, ~LENTO~, ~LENTA~, ~POCO~, ~POCA~ se usa la definición del operador ¬ definido por Zadeh, que se incluye en [30]:

$\neg\mu_A(x)=1-\mu_A(x)$

Igualmente, el calificador ~MUY~ aplicable a los operadores difusos anteriores, se calcula con el concepto de concentración, también incluido en [30]:

CON $\mu(x) = \mu^2(x)$

Es posible encontrar usuarios del modelo que prefieran de todos modos definir los valores máximo y mínimo de la función de pertenencia, con el fin de validar lo que internamente piensan del atributo que se está evaluando. En este caso, el cálculo del grado de pertenencia μ_a de las tuplas a la búsqueda difusa planteada se realiza con base en una función de pertenencia que conserva la misma forma calculada mediante la distribución de frecuencias relativas de posibilidades acumuladas como se planteó en el método, pero adaptando los valores máximo y mínimo a los

valores sugeridos por el usuario. Con ello se busca que el comportamiento de la tabla o vista impacte de todos modos sobre la subjetividad del usuario y devuelva las tuplas con el modelo mental del usuario pero con las tendencias que lleva consigo el atributo correspondiente.

Cuando el usuario no establece un <grado mínimo pertenencia> se supone 0.5, puesto que valores más bajos perfectamente podrían ser interpretados como más cercanos al límite inferior de la búsqueda. Si el usuario no establece un <número de tuplas a retornar>, el modelo se limita a presentar todas las tuplas que cumplan como mínimo con un grado de pertenencia a la búsqueda de 0.5.

Para los operadores ~CERCANO A~, ~CERCANA A~ y ~ALREDEDOR DE~, la notación es igualmente posfijo y las condiciones son las siguientes:

<campo> ~CERCANO A~ <b: valor> [<t: tolerancia>] [<número de tuplas a retornar>]

<b: valor> siempre debe ser establecido por el usuario y <t: tolerancia> puede ser omitida, en cuyo caso se trabaja con una desviación estándar de los datos correspondientes.

En este caso, la función de pertenencia a usar es la siguiente:

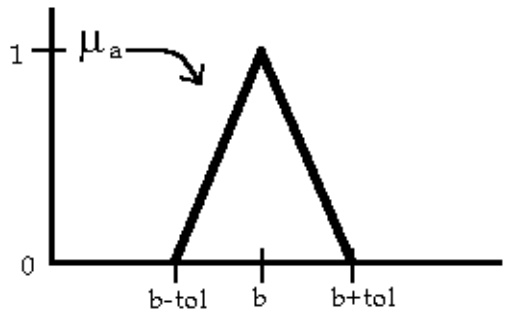

Figura No. 5: Función de Pertenencia para el operador ~CERCANO A~

Las búsquedas precisas se separan de las difusas con los conectores clásicos AND y OR, los cuales permiten añadir el valor total de la función de pertenencia para la búsqueda realizada a las características "precisas" que se requieran para terminar de filtrar la información.

Nótese que el modelo definido realiza una partición de cada Universo del Discurso en unas cuantas etiquetas lingüísticas que se pueden resumir así: lo alto y lo muy alto, lo bajo y lo muy bajo y lo cercano a un valor. Otros modelos se ocupan de la labor de etiquetar otras características del universo del discurso (lo medianamente alto, lo razonablemente bajo, lo suficientemente cercano, etc.), que finalmente introducen unas características definidas en forma subjetiva, atentando contra la objetividad de la información consignada en la base de datos.

En relación con este tema, Teska [3] presenta algunos trabajos de otros autores como Shyi Ming Chen donde se establecen otros modificadores a los operadores difusos, así:

Cuantificadores difusos	Intervalos numéricos
Siempre	1.00 a 1.00
Muy fuerte	0.95 a 0.99
Fuerte	0.80 a 0.94
Más o menos fuerte	0.65 a 0.79
Medio	0.45 a 0.64
Más o menos débil	0.30 a 0.44
Débil	0.10 a 0.29
Muy débil	0.01 a 0.09
No	0.00 a 0.00

Nótese que en esta tabla los límites bien podrían haber sido definidos de forma diferente por otras personas, de acuerdo con su subjetividad particular, y por ello

es una materia en la cual no habría un consenso general para la determinación de las etiquetas lingüísticas apropiadas y los límites numéricos de aplicación de los mismos. Además, siempre existe la posibilidad de que se puedan definir otras etiquetas lingüísticas no consignadas en la tabla anterior.

3.2 CARACTERISTICAS GENERALES DEL MODELO

El sistema debe cumplir con las siguientes características:

3.2.1 Debe satisfacer al máximo los requerimientos del Modelo Relacional. En [31] se pueden encontrar algunas de las características más relevantes del Modelo relacional y del SQL que deben ser respetadas por el modelo a implementar. Existen algunos problemas asociados con la adición de tipos difusos como se plantea en otros modelos (ver [10], [11], [23] en la página 62, [24] en la página 82): se puede violar la integridad de entidad requerida en dicho modelo al incluir información difusa, por ejemplo, en la clave primaria de una entidad, pues, al no ser completamente definidos los límites de la información, se podría perder la característica de unicidad requerida para que el atributo sea identificado como clave primaria. Por ejemplo, si la clave primaria de la base de datos de los alumnos de una universidad fuera el carné y pudiera admitir datos como "un valor cercano a 1000", ello podría representar a los estudiantes identificados con los carnés 999, 1000 y 1001. Otro problema asociado con la adición de tipos difusos lo constituye la posibilidad de recuperar todas las tuplas en una búsqueda, incluso aquellas que por seguridad no deben ser conocidas por todos los usuarios; este problema también está vigente para la adición de operadores difusos en el SQL convencional. Pese a lo anterior, y tomando como restricciones las

características anotadas, la adición de tipos difusos a los manejadores de bases de datos relacionales puede constituir un paso adelante en la generalización de la información, aunque para el presente modelo no será tomada en consideración, y se seguirá trabajando para el modelo con bases de datos convencionales donde los datos son precisos y la información difusa se manejará mediante el lenguaje de búsqueda, de la manera presentada para el modelo.

3.2.2 No incluirá todas las vertientes posibles de manejo de la información difusa, por la gran amplitud que ello podría tener, si bien los conceptos difusos provienen de aproximaciones al lenguaje natural y pueden existir muchas variaciones lingüísticas para el manejo de la información difusa. En el anexo 1 se puede encontrar una discusión en torno del manejo de la incertidumbre en la información. Este modelo se concentrará en los atributos de tipo numérico, los cuales son más fáciles de usar en la implementación de ciertos operadores en términos difusos. Se escapa al alcance de este modelo el tratamiento de tipos de datos alfanuméricos o de fechas que se encuentren coexistiendo en la base de datos, por ejemplo problemas tales como "Busque en la base de empleados aquellos empleados cuyo color de pelo es castaño", puesto que castaño es una denominación para el cabello de las personas que no son completamente rubias ni completamente pelinegras; la implementación de un modelo que incluya esos tipos de datos requiere la definición particular de cada una de las características difusas de tales atributos por parte del usuario, lo cual a la postre puede complicar de forma tal el modelo que sería mucho más simple para el usuario realizar sus propias suposiciones para realizar la búsqueda (por ejemplo "busque los empleados que tengan el cabello negro o rubio").

3.2.3 Debe entregar un marco adecuado para representar la información difusa que puede manejar, lo cual se traduce en unas condiciones propicias para realizar operaciones y recuperar información a partir de ella. En este sentido, el lenguaje de búsqueda debe soportar la inclusión de un conjunto mínimo de operadores, cuya función consiste en la recuperación y el tratamiento de la información, tomando en consideración que ésta puede ser difusa en el planteamiento de las búsquedas. Igualmente, Las búsquedas difusas deben y pueden ser combinadas con búsquedas precisas expresadas en SQL convencional, y no se deben presentar problemas de compatibilidad por este tipo de combinaciones.

3.2.4 Debe aceptar ciertas suposiciones con base en la información que se pueda recuperar de la tabla o vista, con el fin de dar un manejo inicial adecuado a las funciones de pertenencia internas que permiten la recuperación de la información. Ello se logra por ejemplo con el cálculo de las funciones de pertenencia a partir de la construcción de histogramas de probabilidades, como se estableció en la sección anterior. Igualmente, debe permitir la retroalimentación del usuario, en el caso de que la información que está recuperando con el planteamiento de la búsqueda difiera con lo que espera obtener el usuario de ella. Por ejemplo, si al realizar una búsqueda con el modelo el usuario detecta información incongruente, puede emplear los valores opcionales de máximo y mínimo de las funciones de pertenencia, para simular distintos escenarios que le puedan entregar nueva información sobre la tabla o vista particular que se está analizando. Además, es recomendable que el modelo se aplique a bases de datos grandes, puesto que, por el hecho de extraer información consignada en la tabla o vista, dicha información será más confiable si el tamaño de la misma es representativo.

3.2.5 El modelo debe ser un "organizador" que realiza un cálculo interno para definir el grado de pertenencia de cada una de las tuplas de la relación para la búsqueda difusa que se realiza. Es por ello deseable que el modelo pueda incluir el "umbral" para el cual es interesante para nosotros ver los resultados (expresado como un grado de pertenencia mínimo y/o una cantidad máxima de tuplas a devolver en el proceso).

3.3 EJEMPLOS

3.3.1 Ejemplo 1:

Si en la relación "Empleados" de la base de datos de una empresa nos interesa buscar el nombre y cédula de los empleados "Altos", el planteamiento de la búsqueda principia por la elección del atributo que servirá para realizar el filtro, en este caso "estatura" y continúa con la elección del operador difuso apropiado, en este caso ~ALTA~. Si el usuario así lo prefiere podría anotar los valores que considera pueden ser de utilidad para el cálculo (por ejemplo mínimo 1.4 m y máximo 1.8 m) y finalmente anotar que le interesan únicamente los tres más altos. Esta consulta se escribiría así en el modelo:

```
SELECT nombre, cedula
FROM Empleados
WHERE estatura ~ALTA~ (1.4,1.8) 3
```

3.3.2 Ejemplo 2

Si en la misma relación anterior nos interesa conocer el nombre y cédula de los empleados de Medellín bien pagados y jóvenes, la consulta se expresaría así:

```
SELECT nombre, cedula
FROM Empleado
WHERE ciudad = 'Medellín' AND ((sueldo ~GRANDE~ ) ~Y~ (edad ~POCA~ (25,35)))
```

Nótese que en este caso el usuario no definió los parámetros para "sueldo", por lo cual la consulta primero deberá calcular los valores máximo y mínimo con base en la información de la tabla Empleado en los atributos involucrados en la búsqueda, y además realizar el cálculo de la distribución de frecuencias relativas de posibilidades acumuladas, que es el equivalente de la función de pertenencia. Nótese también que se pueden combinar búsquedas precisas con búsquedas difusas para obtener un número más reducido de tuplas en el momento de la generación de la consulta.

3.4 CARACTERISTICAS DEL MODELO FISICO

3.4.1 El modelo se realizó en Fox Pro 2.6 bajo Windows, el cual, si bien no es un manejador de bases de datos relacionales propiamente dicho, sí tiene características de facilidad de programación y de manejo de tablas que permite el modelamiento de los principales requisitos definidos. El código fuente del programa se presenta en el anexo 2.

3.4.2 Se emplea la teoría de compiladores, lo cual implica que se descomponen los diferentes elementos del enunciado del problema y se les asigna un significado que ayuda a determinar la solución particular en cada caso. La descomposición de los términos se almacena en una tabla adicional llamada interpre.dbf que se actualiza con la información de cada búsqueda particular.

3.4.3 Se realiza una interpretación sintáctica completa para evitar que ciertas inconsistencias de la información se involucren en la búsqueda. Por ejemplo, el modelo chequea la existencia de las tablas o vistas, los atributos correspondientes y los tipos de dato de dichos atributos.

3.4.4 Se emplean algunos operadores difusos definidos para este modelo en particular, y con unos símbolos especiales definidos para este lenguaje propio.

3.4.5 Cuando el usuario no entrega los datos suficientes para el cálculo de las funciones de pertenencia correspondientes, el programa los calcula con base en la teoría de posibilidades enunciada cuando se discutió el método de cálculo de las distribuciones de frecuencias relativas de posibilidades acumuladas.

3.4.6 Añade a la información que retorna por consulta el valor evaluado de la función de pertenencia para cada tupla. Para ello, añade a la tabla o vista

una serie de atributos para el cálculo de valores parciales de la función de pertenencia. Este paso se puede obviar luego cuando se use un manejador de bases de datos relacionales propiamente dicho que emplee vistas.

3.5 PROGRAMA DE BUSQUEDAS DIFUSAS: MANUAL DEL USUARIO

3.5.1 Requerimientos de Hardware:

Procesador 486 o pentium.
8 MB de Memoria Ram.
Espacio mínimo en disco duro: 50 MB.
Unidad de disco flexible de 3.5".

3.5.2 Requerimientos de Software:

Windows 95, 98 o 2000. No funciona bien con Windows NT Server.
Internet Explorer 4.0 o superior.

3.5.3 Estructura interna del Programa:

Tablas

La mayoría de las siguientes tablas tiene asociado un archivo de índices, con el mismo nombre, pero con extensión cdx:

Empleado.dbf: Ejemplo de una tabla que trae el software.

Foxuser.dbf: tabla que requiere el fox pro 2.6 bajo windows para funcionar.

Frecuenc.dbf: tabla que se emplea en el cálculo de las distribuciones de probabilidades y posibilidades.
Interpre.dbf: tabla que se emplea para atomizar las diferentes búsquedas ya sea escritas en el lenguaje SQL extendido o mediante la interfaz de búsquedas difusas.

Partes.dbf: Ejemplo de una tabla que trae el software.

Programas:

Programa.prg: Programa principal de búsquedas difusas, escrito en Fox Pro 2.6 bajo Windows.

Auxili.prg: Programa de la opción "auxiliares" del software, que sirve para las labores de actualización y borrado de tuplas de las diferentes tablas

Otros archivos:

Ayuda.htm: Archivo en HTML que describe la ayuda y puede visualizarse mediante Internet Explorer 4.0 o superior.

Autor.bmp y Un.bmp: Archivos de imágenes que se usan en el software.

Busqdifu.exe: Archivo ejecutable que inicia el software.

3.5.4 Instalación del Software:

Para la instalación se debe crear una carpeta llamada BUSQDIFU en la unidad C: del equipo y allí se debe copiar la totalidad de los archivos del CD en la carpeta BUSQDIFU. Los archivos contenidos en la carpeta LIBRERIAS deben ser instalados en la carpeta C:\WINDOWS.

3.5.5 Funcionamiento del programa:

Una vez instalado, el programa se activa mediante el archivo BUSQDIFU.EXE que se encuentra en la carpeta BUSQDIFU de la unidad C:. Al ingresar al sistema, el software presenta un mensaje de bienvenida en el cual se debe dar click en INGRESAR, y luego el programa presenta cinco opciones: Versión Texto, Versión Gráfica, Auxiliares, Ayuda y Terminar. Cada una de ellas tiene una letra subrayada que sirve para el acceso rápido a la opción correspondiente. El funcionamiento de cada opción es el siguiente:

3.5.5.1 Versión Texto:

Se accede con la letra V o dando click en la opción correspondiente. Al acceder a esta opción aparece la ventana que se muestra en la figura No. 5.

En este caso, el ingreso de la información es similar a como se haría en SQL convencional, sólo que se escribe al frente de cada una de las

partes de la instrucción (SELECT, FROM, WHERE) y desplazándose con el mouse o con el tabulador a la parte siguiente. Una vez se ha diligenciado la información necesaria, se termina con Grabar. Existe también la opción de Cancelar para regresar al menú principal. Las opciones de grabar y cancelar se pueden acceder también desde cualquier posición en la ventana oprimiendo Alt simultáneamente con la letra subrayada (G para Grabar y C para Cancelar).

En esta opción, se debe conocer específicamente la sintaxis del lenguaje, la cual incluye los campos que se van a mostrar en la consulta en SELECT, la tabla de la cual se va a extraer la información en FROM, y la combinación de cláusulas de SQL convencional ampliadas con búsquedas difusas que emplean los operadores ~MUY GRANDE~, ~MUY PEQUEÑO~, ~GRANDE~, ~PEQUEÑO~, ~CERCANO A~, ~ALTO~, ~ALTA~, ~RAPIDO~, ~RAPIDA~, ~MUCHO~, ~MUCHA~, ~PEQUEÑA~, ~BAJO~, ~BAJA~, ~LENTO~, ~LENTA~, ~POCO~, ~POCA~, ~MUY ALTO~, ~MUY ALTA~, ~MUY RAPIDO~, ~MUY RAPIDA~, ~MUY PEQUEÑA~, ~MUY BAJO~, ~MUY BAJA~, ~MUY LENTO~, ~MUY LENTA~, ~MUY POCO~, ~MUY POCA~, ~CERCANA A~, ~ALREDEDOR DE~ y los conectores difusos ~Y~ y ~O~. Para más información, remitirse al numeral 3.1.

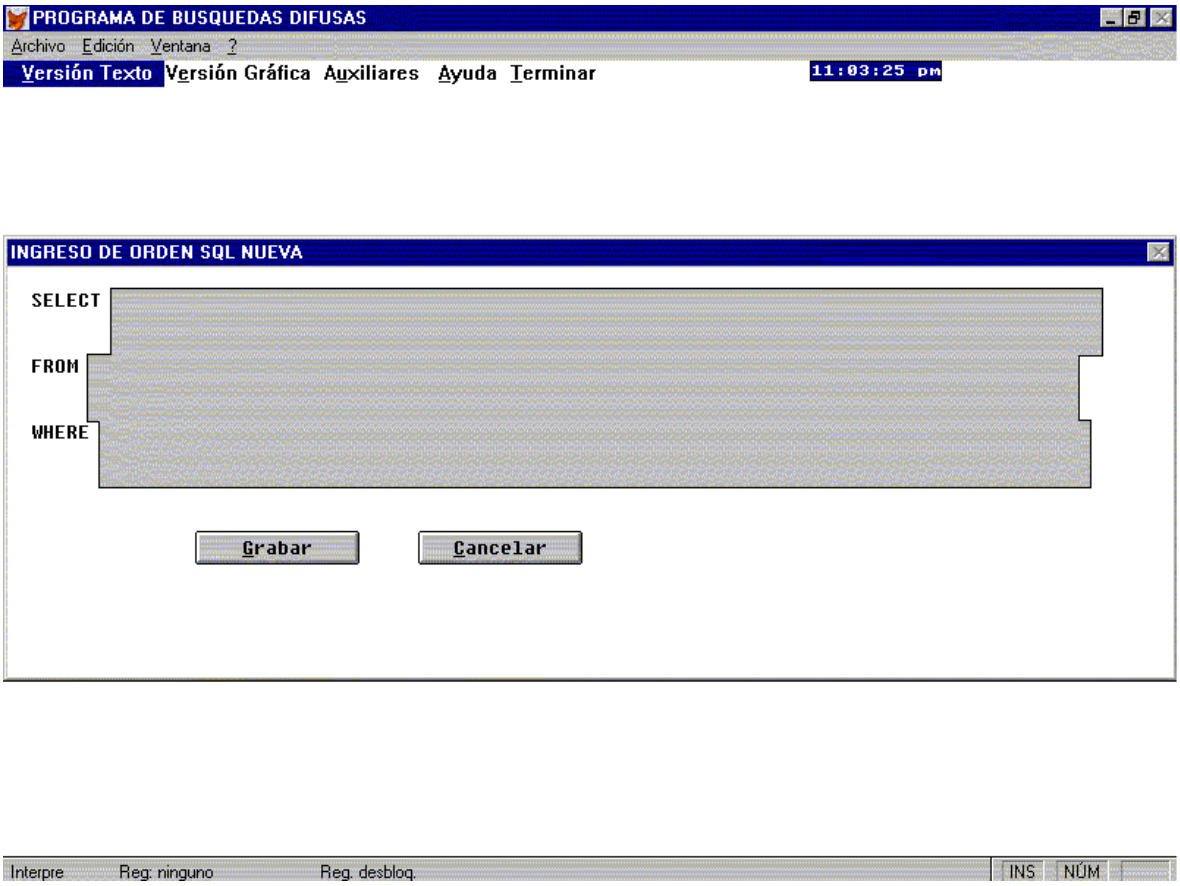

Figura No. 6: Versión texto del programa.

El ejemplo del numeral 3.3.2 puede ilustrar un poco el uso. Las pantallas de ingreso de la información y respuesta, se muestran a continuación:

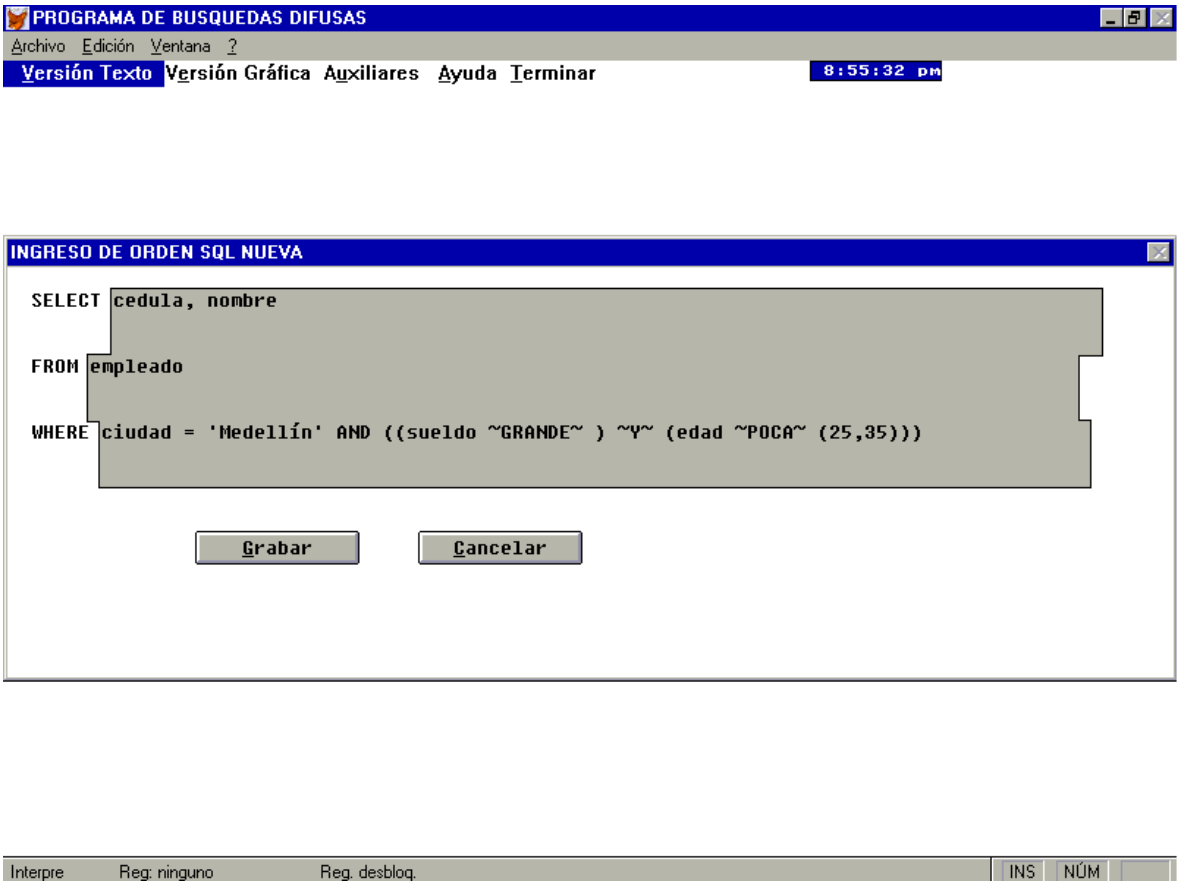

Figura No. 7: Ejemplo del numeral 3.3.2 realizado en la versión texto del programa.

Nótese que el valor "pertotal" que se incluye en la respuesta a la búsqueda es el grado de pertenencia de cada tupla de la relación "partes" a la búsqueda difusa realizada. Para salir de la tabla de respuestas se oprime la tecla escape.

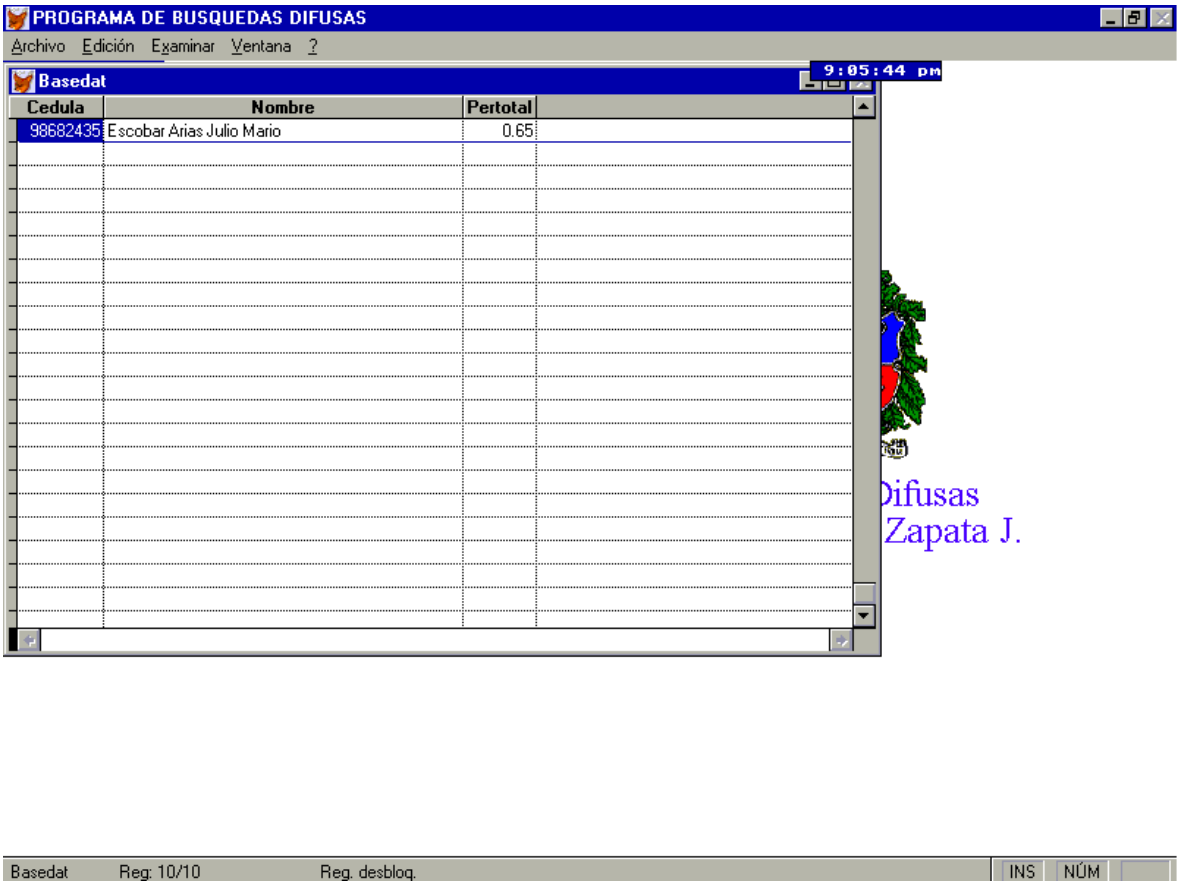

Figura No. 8: Respuesta del programa a la búsqueda del ejemplo del numeral 3.3.2

3.5.5.2 Versión Gráfica:

Se accede a esta opción digitando la letra e o dando click con el mouse en la opción correspondiente. Al acceder a esta opción, el programa solicita una base a usar, con una ventana como la siguiente:

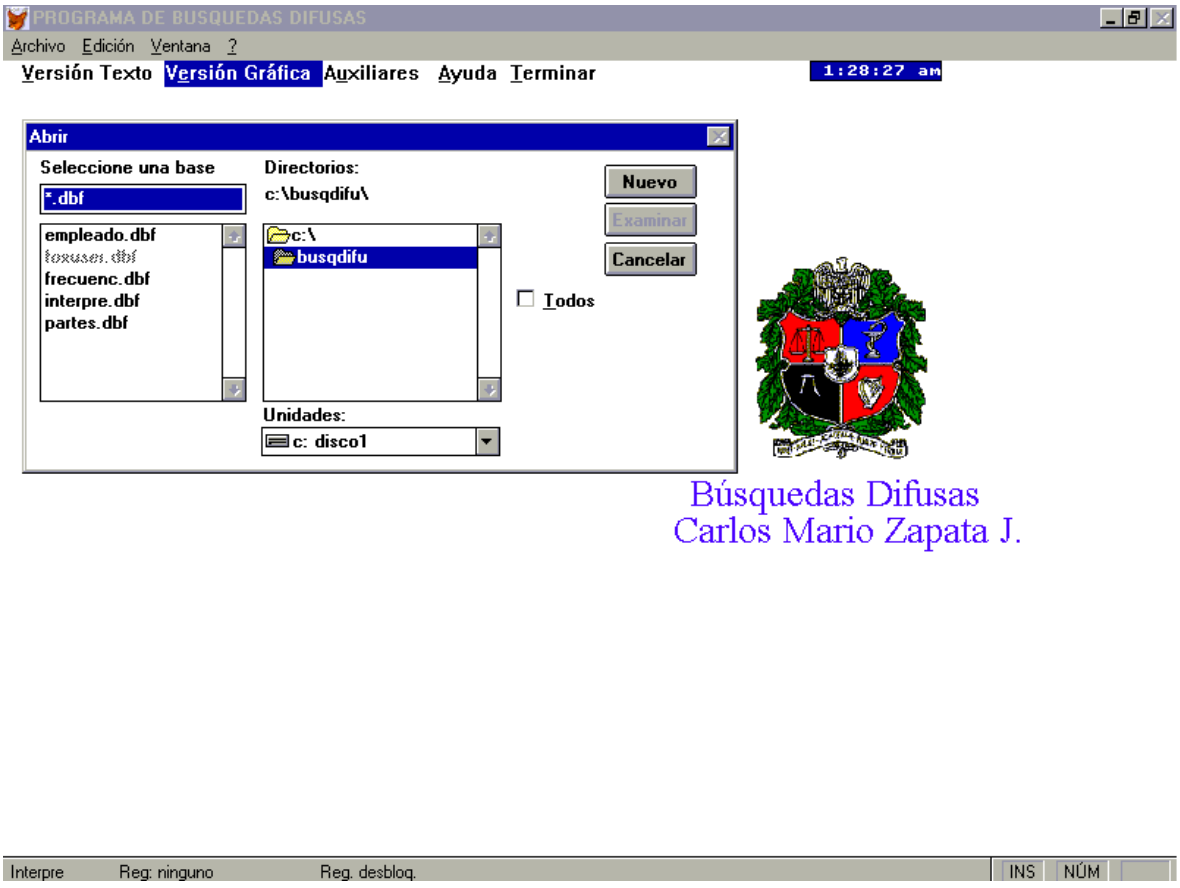

Figura No. 9: Ventana de selección de base en la versión gráfica del programa.

Al dar doble click sobre la base correspondiente, el programa ingresa en otra ventana que permite definir la búsqueda en forma gráfica seleccionando los campos que van a hacer parte de la visualización final y las diferentes combinaciones de búsquedas precisas con búsquedas difusas. El ejemplo 4 del numeral 4.3, expresado en versión gráfica y después de dar doble click en la base "partes", quedaría como se muestra en la figura siguiente. Se debe notar que en la sección donde se establecen las búsquedas difusas o precisas (dependiendo del tipo de operador), el sistema saca ya sea el campo correspondiente a valor, para llenarlo en caso de tratarse de una búsqueda precisa, o los campos correspondientes a máximo, mínimo, número de tuplas y tolerancia, en el caso de tratarse de una búsqueda difusa. En ambos casos, se debe

presionar Shift+tab hasta que se ilumine la opción correspondiente a "campos" para continuar con la definición de la búsqueda siguiente.

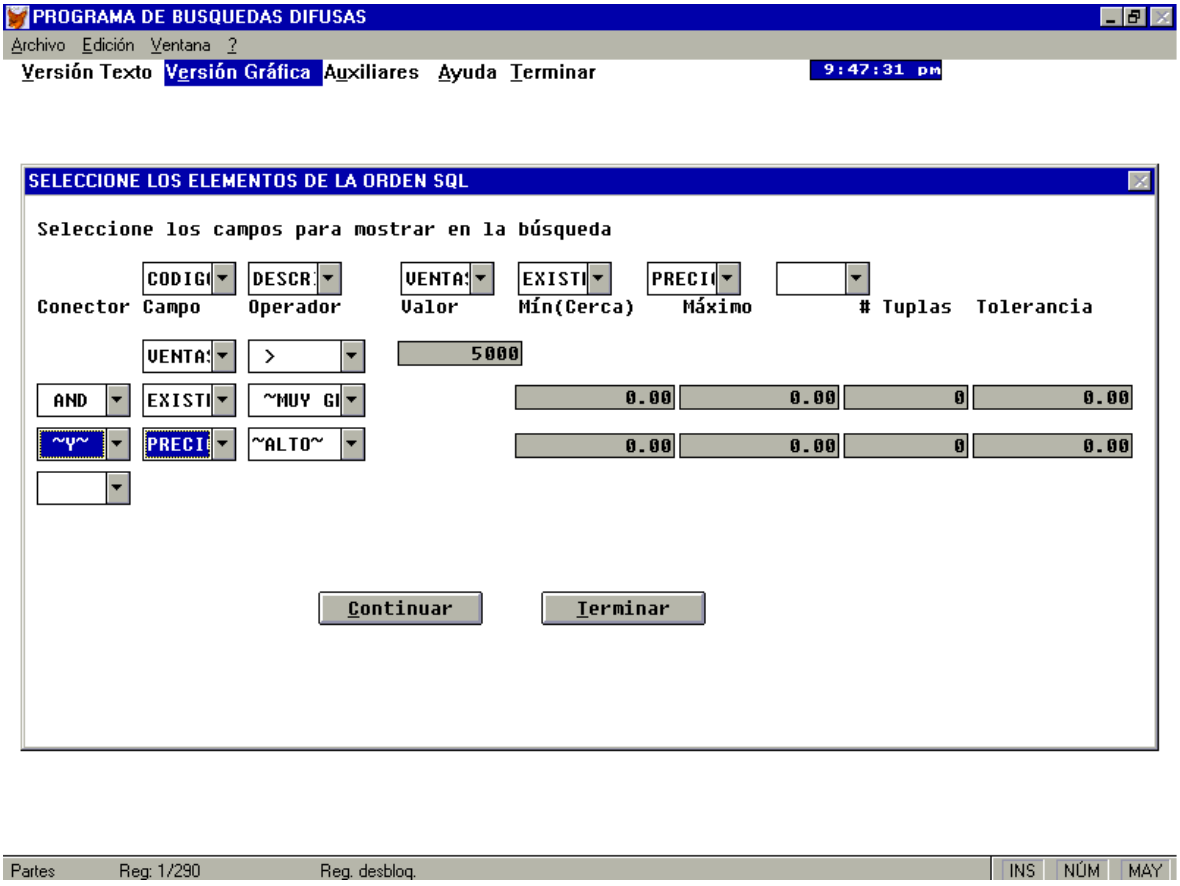

Figura No. 10: Ejemplo 4 del numeral 4.3 aplicado a la versión gráfica del programa.

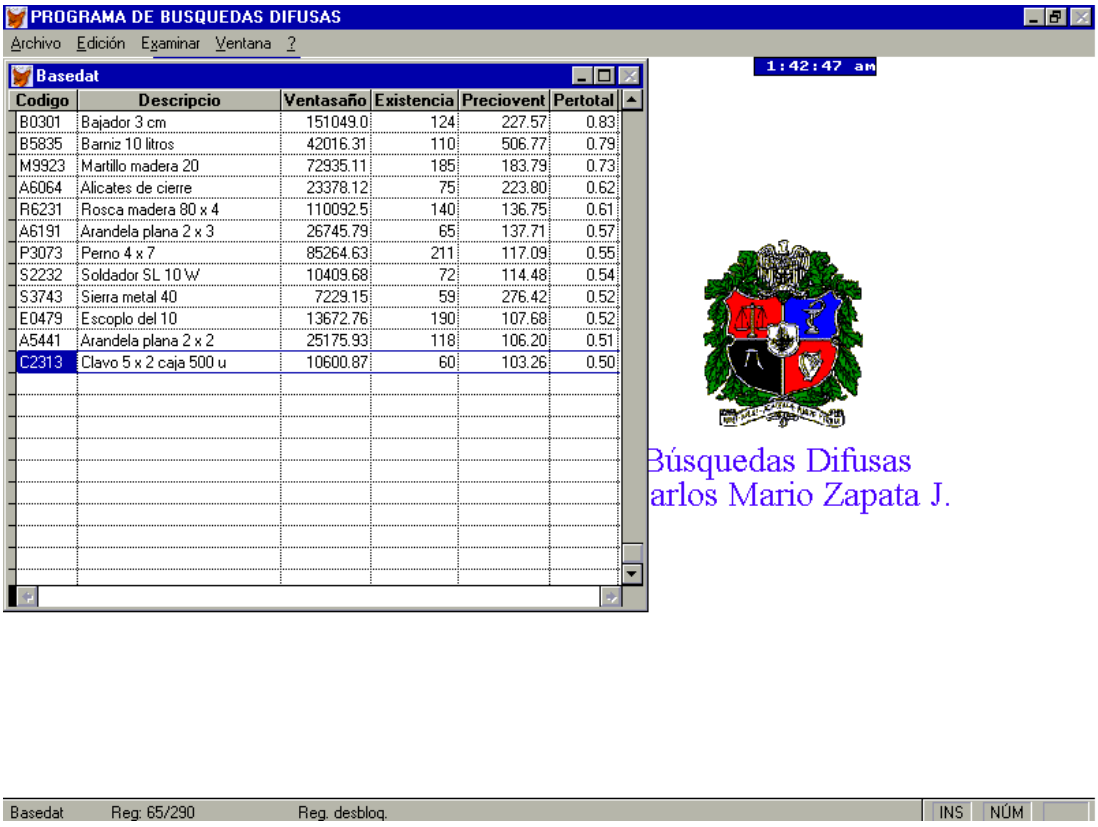

Figura No. 11: Tabla de resultados de la búsqueda difusa No. 4 del numeral 4.3.

3.5.5.3 Auxiliares:

A esta opción se ingresa presionando la letra u o dando click en la opción correspondiente. Al hacerlo, se habilitan cinco opciones adicionales, que son:

- Examinar, adicionar y marcar para borrado.
- Empaquetar tablas.
- Crear tablas.
- Modificar estructura.
- Restauración de índices.

Para todas las opciones, al activar la opción correspondiente se habilita una ventana como la de la figura No. 9 con el fin de seleccionar la tabla

que se va a procesar. Una vez se selecciona la tabla, en examinar, adicionar y marcar para borrado aparece la base de datos en una ventana y a través del menú superior se pueden realizar las diferentes opciones, oprimiendo control + la letra D se pueden adicionar tuplas a la base y oprimiendo control + la letra L se genera una marca en la tupla sobre la cual se está posicionado con el fin de que ese elemento pueda ser borrado de la tabla posteriormente con la opción empaquetar tablas. Si se quiere borrar la marca, basta con presionar nuevamente control + la letra L.

En restauración de índices, simplemente se selecciona la base y el programa hace un proceso interno para restaurar un índice que se requiere para el ordenamiento de los resultados de las búsquedas.

En modificar estructura, al seleccionar la tabla se ingresa en una ventana como la que se muestra en la figura No. 11, en la cual se pueden modificar las características de los campos de la tabla seleccionada, cambiando los tipos (caracter, numérico, fecha, lógico, etc.) y agregando o eliminando campos.

En el caso de crear tabla, no se selecciona una tabla existente sino que se escribe un nombre para la nueva tabla y luego se ingresa en una ventana como la de la figura No. 11 para hacer la creación de los campos correspondientes a esa tabla. Es de notar que las tablas nuevas que se quieran emplear para el sistema de búsquedas difusas deberán contener, además de los campos pertinentes a la información, otros diez campos definidos de la siguiente manera: pertene1, pertene2, ..., pertene9 y pertotal, todos con un tipo numérico de ancho 6 y dos decimales. Estos campos son necesarios porque el Fox Pro no maneja vistas como otros manejadores de bases de datos y con esos campos adicionales se

realizan los cálculos correspondientes a los grados de pertenencia de cada tupla a las búsquedas difusas planteadas.

Figura No. 12: Visualización de la ventana para modificar estructura de una tabla seleccionada.

3.5.5.4 Ayuda:

Al digitar la letra A o dar click en la opción correspondiente, el programa abre una ventana en html donde se encuentran las ayudas del mismo.

3.5.5.5 Terminar:

Con esta opción se termina la sesión y se cierra el programa.

4. ANALISIS DE RESULTADOS

4.1 LIMITACIONES

4.1.1 Unicamente se manejan conceptos difusos para campos de tipo numérico. Ello implica que con este modelo no podemos aproximarnos a bases de datos en busca de mujeres "BELLAS" o personas de cabello "CASTAÑO", como sugiere Medina en [23, página 13]. Igualmente, otros tipos de datos diferentes a los numéricos no se tratan mediante este modelo; por ejemplo, no se tratan campos que tengan un tipo de dato objeto, como en el caso de fotos, ilustraciones, páginas web, etc. Un modelo tal tiene una complejidad mucho mayor y lo que se pretende con este modelo es sentar las bases para trabajos futuros que se enfoquen en este tema de investigación.

4.1.2 Las consultas en este modelo aún no pueden ejecutarse en lenguaje natural; se requieren algunas abstracciones de todos modos. Si bien el sistema prototipo desarrollado tiene una versión gráfica que en cierta forma facilita la interacción del usuario con la base de datos, la realización de las consultas requiere tener un mínimo conocimiento de la estructura de las bases de datos. Ello se hace más evidente en la versión texto del sistema, la cual trabaja con una sintaxis parecida al SQL convencional, con la adición de los operadores difusos y los conectores correspondientes, lo cual en el fondo es la característica principal que se buscaba en el objetivo

general del presente trabajo de investigación. Esto implica que de alguna forma quien se aproxime a usar el sistema, especialmente en la versión texto, debe tener algunos conocimientos de SQL, los cuales, sin embargo, se pueden obtener de la lectura de la referencia [31].

4.1.3 En relación con el sistema mismo, por el hecho de haber sido realizado en Fox Pro 2.6 bajo Windows, que no es en sí mismo un manejador de bases de datos relacionales, muchas de las características del SQL convencional no van a tener aplicabilidad en el mismo. Por ejemplo, los comodines que se usan en SQL o incluso los join (enlaces o reuniones) sobre dos o más tablas para recuperar información simultáneamente de ellas no se van a poder realizar sobre el sistema desarrollado. Sin embargo, esto era previsible desde el principio, porque lo más importante era demostrar la aplicabilidad del SQL incrementado a un problema convencional de bases de datos. Además, los resultados de la investigación no se ven afectados por esta limitación, pues el prototipo se constituye tan sólo en una demostración académica de que es posible realizar la incorporación de operadores que se plantea en el ámbito de manejadores de bases de datos relacionales.

4.2 AMBITO DE APLICACION

La incorporación de operadores difusos al SQL convencional puede ser un tema de interés para los manejadores convencionales de bases de datos relacionales, pues puede facilitar la interacción de los usuarios comunes con ese lenguaje sin tener que hacer suposiciones que restrinjan las posibilidades del lenguaje mismo y

la recuperación de la información. A este respecto, se pueden encontrar en la literatura algunos ejemplos de manejos especiales para la realización de búsquedas difusas al interior del SQL convencional:

- En 1990, Bruce Lindsay y Laura Haas [33] en las memorias del Simposio Internacional de Müggelsee, Berlín, mencionan un ejemplo de realización de ciertas búsquedas imprecisas en SQL convencional, usando lo que ellos denominan "expresiones de tabla", las cuales "son esencialmente definiciones de vistas en la cláusula FROM de una expresión de búsqueda. Definen tablas que pueden ser referenciadas en cualquier parte de la búsqueda. Por ejemplo, dado el siguiente esquema:"

  Emp(e_name, e_sal, e_dno, ...)
  Dept(d_dno,...)
  "la siguiente búsqueda entrega el nombre, salario y relación entre el salario y el salario promedio de los empleados del departamento para los empleados 'bien-pagados'. Note cómo una expresión de tabla ('dsal') se usa para definir el salario promedio del departamento."

```
SELECT emp.e_name, emp.e_sal, emp.e_sal/dsal.ds_avg
FROM emp,
        dsal(ds_dno, ds_avg) AS
                (/* salario promedio por departamento */
                        SELECT d2.d_dno, AVG(e2.e_sal)
                        FROM dept d2, emp e2
                        WHERE d2.d_dno = e2.e_dno
                        GROUP BY (d2.d_dno)
                )
WHERE emp.e_dno = dsal.ds_dno
    AND emp.e_sal > dsal.ds_avg; /* empleado es "bien pagado" */
```

En este ejemplo se puede apreciar cómo la parte difusa de la búsqueda se soluciona mediante un artificio subjetivo de quien la realiza, al decir que los empleados "bien pagados" son aquellos que poseen un salario mayor que el promedio de los salarios de su departamento, como se aprecia en el AND de la cláusula WHERE, es decir, se está realizando una suposición para solucionar la parte difusa, sin que medie para ello ningún tipo de consulta a la base de datos y su información para determinar cuáles pueden ser los empleados "bien pagados".

- En Septiembre de 1999, Suad Alagic [34] explicó algunas modificaciones al modelo OQL (Object Query Language), el cual se basa en parte en SQL embebido y procura mejorar sus características para el manejo con objetos. En dicho paper, Alagic se vale de un ejemplo que, en la sintaxis OQL, procura devolver en una búsqueda los empleados bien pagados de una empresa. La sintaxis de ese ejemplo es la siguiente:

  ```
  select well_paid(emp: x.name, sal: x.salary())
  from employees as x
  where x.salary() . 50,000
  ```

  El tratamiento de la consulta difusa se realiza en este caso suponiendo que los empleados por encima de un valor de 50,000 son aquellos que son bien pagados en la compañía, como se muestra en la cláusula WHERE de la consulta. Optó quien desarrolló la consulta por definir previamente cuál es el límite para el cual se presentan los empleados bien pagados.

- En noviembre de 2000, Kazimiers Subieta [35] presentando una semántica especial para realizar búsquedas en bases de datos, llamada "loqis" define también el siguiente ejemplo: "La función WellPaid tiene un parámetro Jobpar y una variable local a:"

```
function WellPaid( JobPar )
    begin
        local a: real;
        a := avg(EMP.salary);
        return EMP where job = JobPar and salary > 2 * a;
    end;
```

Ya en este caso el parámetro para los empleados bien pagados es también subjetivo y es dos veces el promedio del salario de los empleados, como se puede notar en la cláusula RETURN de la función definida.

¿Cuál de los tres esquemas tiene completamente la razón para definir los empleados bien pagados? Subjetivamente hablando, los tres esquemas pueden tener la razón, pero aún así continúa siendo un término relativo que en lenguajes convencionales debe ser definido mediante suposiciones al interior de cada uno de los esquemas que se manejan. Es decir, no existe una herramienta al interior de los lenguajes tipo consulta convencionales que permita resolver adecuadamente el asunto de la subjetividad de las búsquedas cuando se presentan términos difusos.

El sistema desarrollado muestra que sí se puede elaborar un lenguaje tal y que puede hacerse tan refinado como sea posible, y además que la sencillez de aplicación y definición de los diferentes requisitos que tienen que ver con los operadores difusos puede ser fácilmente implementable en cualquier lenguaje que trabaje con SQL convencional. No existe, empero, una equivalencia entre las búsquedas que plantea el modelo definido en esta tesis y una búsqueda en SQL convencional, es decir, no existe una forma en SQL convencional de realizar las búsquedas que se plantean con los nuevos operadores difusos definidos (sin recurrir, obviamente a artificios como los definidos en los tres ejemplos anteriores), y que devuelvan exactamente las mismas tuplas que se han conseguido con el uso del sistema desarrollado.

Ahora bien, por el hecho de no requerir la definición de nuevos tipos de datos al interior de la base de datos, un modelo tal puede tener aplicación para la realización de búsquedas en casi cualquier base de datos existente, siempre y cuando se maneje para los campos de tipo numérico y con las limitaciones anotadas en el numeral anterior.

4.3 EJEMPLO DE APLICACION

El dueño de una ferretería cuenta con una base de datos de los productos que maneja (para el sistema se denomina partes.dbf; ver sección 3.5.3), la cual contiene la siguiente información: código del producto (codigo), descripción (descripcio), existencia actual (existencia), pedidos pendientes (pedidoptes), precio de venta (preciovent), costo de compra (costocompr), unidades compradas al año (unidadesañ) y ventas al año (ventasaño). En su afán por analizar la gestión comercial de su negocio, este señor se acerca a su administrador de sistemas y le hace algunas preguntas básicas, en su propio lenguaje:

1. Necesito revisar las necesidades de mis clientes en relación con las existencias de almacén. ¿Cuáles son los productos con pedidos pendientes más grandes y de los cuales poseemos pocas existencias actualmente?

2. Quiero saber cuáles de los productos que mayores ventas al año me producen tienen en este momento los pedidos más bajos, con el fin de hacer gestión entre mis clientes y averiguar por qué esos pedidos están bajos.

3. No cuento con dinero suficiente para comprar el inventario que se necesita para atender los pedidos pendientes. ¿Me podría decir cuáles son los productos que tienen los pedidos más grandes y a su vez los costos de compra más bajos?

4. Para fomentar las ventas necesito saber cuáles son los productos con ventas anuales superiores a US$5.000, de los cuales tenemos mayores existencias unitarias y que tienen un precio de venta alto.

El Administrador de sistemas tiene una copia del software de búsquedas difusas y procede entonces a plantear las búsquedas correspondientes en el SQL expandido que se explicó en el capítulo anterior, de la siguiente manera:

1. SELECT codigo, descripcio, pedidoptes, existencia
   FROM partes
   WHERE ((pedidoptes ~MUY GRANDE~) ~Y~ (existencia ~BAJA~))

2. SELECT codigo, descripcio, ventasaño, pedidoptes
   FROM partes
   WHERE ((ventasaño ~MUY ALTO~) ~Y~ (pedidoptes ~MUY BAJO~))

3. SELECT codigo, descripcio, pedidoptes, costocompr
   FROM partes
   WHERE ((pedidoptes ~MUY GRANDE~) ~Y~ (costocompr ~MUY BAJO~))

4. SELECT codigo, descripcio, ventasaño, existencia, preciovent
   FROM partes
   WHERE ventasaño > 5000 AND ((existencia ~MUY GRANDE~) ~Y~ (preciovent ~ALTO~))

Los Resultados de algunos de los ejemplos de aplicación y la forma de realización de las búsquedas con la interfaz con el usuario se pueden consultar en el numeral 3.5.5.

5. CONCLUSIONES Y FUTURAS LINEAS DE INVESTIGACION

5.1 CONCLUSIONES:

5.1.1 El modelo realizado para este trabajo de investigación incorpora los siguientes operadores difusos en el ámbito de manejadores de bases de datos relacionales convencionales: ~MUY GRANDE~, ~MUY PEQUEÑO~, ~GRANDE~, ~PEQUEÑO~, ~CERCANO A~, ~ALTO~, ~ALTA~, ~RAPIDO~, ~RAPIDA~, ~MUCHO~, ~MUCHA~, ~PEQUEÑA~, ~BAJO~, ~BAJA~, ~LENTO~, ~LENTA~, ~POCO~, ~POCA~, ~MUY ALTO~, ~MUY ALTA~, ~MUY RAPIDO~, ~MUY RAPIDA~, ~MUY PEQUEÑA~, ~MUY BAJO~, ~MUY BAJA~, ~MUY LENTO~, ~MUY LENTA~, ~MUY POCO~, ~MUY POCA~, ~CERCANA A~, ~ALREDEDOR DE~.

5.1.2 El modelo permite también la incorporación de los conectores difusos ~Y~ y ~O~, los cuales se basan en las definiciones entregadas por Zadeh en los inicios de la lógica difusa.

5.1.3 El modelo parte del SQL convencional que está contenido dentro de los principales manejadores de bases de datos relacionales e incorpora los operadores y conectores difusos mencionados. Para la realización de los cálculos de las funciones de pertenencia de las diferentes tuplas o

instancias de las relaciones, el modelo permite dos opciones: la entrega por parte del usuario de los valores más relevantes que le sirven al sistema para conformar la función de pertenencia y el cálculo interno a partir de la información contenida en la base de datos. En ambos casos se calcula la distribución de frecuencias relativas de posibilidades acumuladas y en el primer caso se adapta a los valores mínimo y máximo entregados por el usuario, con la opción de definir un número de tuplas a retornar y un determinado umbral para la función de pertenencia (es decir, un grado de pertenencia mínimo aceptable para la presentación de los datos). Es de notar que el modelo plantea la definición de unas pocas etiquetas lingüísticas, lo que a la postre se constituye en una fortaleza del mismo, puesto que no dirime con el usuario la implementación de otras etiquetas lingüísticas cuyos límites de aplicación se presentan como altamente subjetivos.

5.1.4 El modelo presenta algunas limitaciones básicas: se definen conceptos difusos únicamente para atributos de tipo numérico y no se emplean consultas en lenguaje natural. En el capítulo 4 se discuten más ampliamente estas limitaciones.

5.1.5 Se construyó un Software prototipo en Fox Pro 2.6 bajo Windows, cuyo código fuente se puede consultar en el anexo 2 y cuyo manual del usuario se puede apreciar en el numeral 3.5. Por constituir únicamente un prototipo de incorporación de operadores difusos en manejadores de bases de datos relacionales, dicho sistema no cumple completamente con las características del Modelo Relacional (que se pueden consultar en [31]), e incluso tiene limitaciones en el ámbito de aplicación del SQL convencional, tales como el manejo de tablas únicas para las búsquedas (no maneja el

concepto de JOIN del SQL) y el hecho de que no maneja los comodines de búsqueda, por ejemplo. Estas limitaciones, sin embargo, no vician el resultado final, puesto que sólo se trataba de demostrar en forma académica que era posible la incorporación de operadores difusos en el ámbito de un manejador de bases de datos relacionales convencional. Se tomó ventaja de la facilidad de programación y la habilidad para el manejo de bases de datos sencillas en la selección de dicho lenguaje para la realización del prototipo.

5.1.6 No existe una equivalencia entre el SQL convencional y el modelo de SQL que incorpora operadores y conectores difusos que se plantea en este trabajo. En SQL convencional sólo se podrían realizar búsquedas que involucraran operadores y conectores difusos realizando algunas suposiciones en relación con la información difusa que se está solicitando, buscando hacerla menos difusa y más "precisa". Ello es inconveniente por cuanto se limitan las búsquedas a la subjetividad de quien las está ejecutando y no se tienen en cuenta las condiciones y valores que están contenidos en la base de datos para ello.

5.2 FUTURAS LINEAS DE INVESTIGACION

Este tema se ha venido trabajando ampliamente en otros países y existe gran cantidad de literatura al respecto. En nuestro país, empero, no se han realizado trabajos que puedan dar cuerpo a una teoría como la que se establece en este trabajo de investigación. En adelante pueden existir algunas líneas de complementación para este trabajo en los siguientes aspectos:

5.2.1 Definición de otros operadores y modificadores que sean susceptibles de definir e incorporar en el ámbito de manejadores de bases de datos relacionales. La metodología es similar a la aplicada en desarrollo de este trabajo, pero teniendo especial cuidado en la forma de definición de los operadores y modificadores, pues es éste un tema bastante subjetivo y la literatura cuenta con muchas propuestas al respecto que deben ser examinadas con detenimiento para que el modelo no pierda la objetividad que se le intentó imprimir.

5.2.2 Definición de operadores difusos que permitan la realización de búsquedas sobre datos con tipos diferentes al numérico: alfanumérico, fechas, objetos, etc. Este tema es de muchísima amplitud y nuevamente el especial cuidado que se debe tener en el manejo de estos tipos de información hace que sea un tema de investigación exhaustiva y meticulosa.

5.2.3 El prototipo en Fox Pro 2.6 bajo Windows tiene limitaciones esbozadas para el comportamiento frente a manejadores de bases de datos relacionales convencionales, razón por la cual un tema de trabajo final para alguna de las especializaciones es la programación del modelo enunciado en un manejador de bases de datos relacionales. Puede ser posible, por ejemplo, la realización de un nuevo prototipo con programación en PL/SQL de Oracle, que pueda incluso usar SQL dinámico. Se reitera que dichas limitaciones en nada limitan o cuestionan los resultados de esta investigación, pues el prototipo sólo es un ejemplo académico de cómo se realizaría el modelo en un lenguaje más cercano a los manejadores de bases de datos relacionales.

5.2.4 Un trabajo futuro puede constituirse también en la elaboración de nuevos modelos basados en lenguaje natural y su reconocimiento sintáctico para empalmar con el modelo que se presenta en este trabajo de investigación. En este trabajo, como se ha mencionado a lo largo del mismo, el usuario debe de todos modos hacer ciertas selecciones que aún no se vinculan con el lenguaje natural, en parte porque el objetivo fundamental era ampliar la riqueza sintáctica del SQL de modo que se pudieran realizar búsquedas imprecisas de información y ello implica un público con conocimientos mínimos de SQL, que se aleja un poco del perfil de los usuarios no especialistas de las bases de datos.

ANEXO 1: Manejo de la Incertidumbre en la Información.

En [25] Dubois y Prade comienzan su discusión en relación con la imprecisión y la incertidumbre así: "La imprecisión y la incertidumbre pueden ser consideradas como dos aspectos complementarios de una realidad simple, la de la información imperfecta. Asumiremos que un ítem de información puede ser expresado como una proposición lógica con predicados y cuantificadores posibles. Un cuerpo de conocimiento será una colección de ítemes de información poseída por un individuo (o un sistema de cómputo, o un grupo de individuos), relativa a un problema simple. Los predicados que ocurren en las expresiones de esta información pueden ser interpretadas entonces como subconjuntos de un dominio simple de referencia. Una proposición puede también ser considerada como una afirmación concerniente a la ocurrencia de un evento. Tales eventos pueden en sí mismos representarse como subconjuntos de este dominio de referencia, el cual puede a su vez llamarse 'el evento seguro'. Así que tenemos tres formas equivalentes de contemplar una colección de información, dependiendo de si enfatizamos en su estructura (aspecto lógico), su contenido (aspecto de la teoría de conjuntos) o la relación entre los ítemes con los eventos reales (aspecto fáctico)".

"Desde el punto de vista práctico, un ítem de información será definido como un cuarteto (atributo, objeto, valor, confianza). 'Atributo' se refiere a una función que tiene adjunto un valor (o un conjunto de valores) para el objeto cuyo nombre figura en el ítem de información. Este valor corresponde al predicado, es decir, para un subconjunto del dominio de referencia asociado con el atributo. La confianza es un indicativo de la confiabilidad de la información. Claramente las cuatro entidades que conforman el ítem de información pueden ser compuestas (algunos

objetos, muchos atributos, enésimos predicados, diferentes grados de confianza). Además, se pueden introducir variables, especialmente para los objetos, si el ítem de información envuelve cuantificadores".

"En este contexto claramente podemos distinguir los conceptos de imprecisión e incertidumbre: la imprecisión se relaciona con el contenido de un ítem de información (el componente 'valor' del cuarteto) mientras que la incertidumbre se relaciona con su verdad, entendimiento y conformidad con una realidad (el componente 'confianza' del cuarteto)".

"La incertidumbre de un ítem de información puede ser evaluada por medio de cualificadores como 'probable', 'posible', 'necesario', 'plausible' o 'creíble', de los cuales intentaremos dar un significado preciso. La modalidad de ser probable ha sido extensamente estudiada por dos siglos. Tiene dos connotaciones distintas, una de las cuales es física, atada a experimentos estadísticos, y concerniente con la frecuencia de ocurrencia de un evento. La otra es epistemológica: aquí 'probable' se refiere a un juicio subjetivo. Las modalidades de Posible y necesario nos llevan a Aristóteles, quien acentuó su dualidad (si un evento es necesario, entonces su contrario es imposible). Singularmente, y en contraste con el concepto de 'probable', lo 'posible' y lo 'necesario' son considerados a menudo como categorías de todo o nada. Pero, como 'probable', 'posible' tiene dos interpretaciones: física (como una medida de la dificultad material de ejecutar una acción) y epistemológica (como un juicio subjetivo que no compromete a quien lo emite). 'Necesario', de otro lado, es una noción más fuerte, tanto en el sentido físico como en el epistemológico (la necesidad subjetiva suma para la certeza). Es natural admitir grados de posibilidad y necesidad, tanto como de probabilidad (un matiz que ya se presenta en el lenguaje natural, en el cual uno dice 'muy posible', por ejemplo). Las connotaciones de 'plausible' y 'creíble' son epistemológicas específicamente y se refieren respectivamente a 'posible' y 'necesario'. Cada uno corresponde a un modo de inferencia basado en un cuerpo dado de conocimiento:

algo que se puede deducir del cuerpo es 'creíble'; algo que no contradice el cuerpo es 'plausible' (aspecto inductivo) (... y) parece cercano a 'concebible'".

"Algunos ejemplos de proposiciones inciertas son":

"Es probable que John es al menos 1.70 m de alto, que equivale a (Altura, John, >=1.70 m, probable)"

"La probabilidad de 10 mm de lluvia mañana es 0.5, que equivale a (cantidad, lluvia mañana, 10 mm, probabilidad = 0.5)"

"Un ítem de información se llamará preciso cuando el subconjunto correspondiente a su componente 'valor' no puede ser subdividido. Dependiendo en qué aspecto de la información se hace énfasis, hablaremos de proposiciones elementales (i. e., implicadas por otras proposiciones que guardan la proposición necesariamente falsa), de un evento elemental (aspecto de la teoría de conjuntos). La propiedad de ser 'preciso' depende por supuesto de la definición del dominio de referencia (en su 'granularidad', por ejemplo, la elección de una unidad de medida)".

"Hay otros cualificadores que se refieren a imprecisión, por ejemplo 'vago', 'difuso', 'general' o 'ambiguo'. Ambigüedad es una clase de imprecisión ligada al lenguaje, algunas veces debida a la homonimia: Un ítem de información es ambiguo en el evento de que se refiera a algunos posibles contextos o conjuntos de referencia. (...) 'Generalidad' es una forma (benéfica) de imprecisión ligada al proceso de abstracción; un ítem de información es general si designa una clase de objetos que expresan una propiedad común. Pero 'vaguedad' o 'difusión' en un ítem de información reside en la ausencia de un límite claro para el conjunto de valores adjunto a los objetos a los cuales se refieren. Muchos calificadores en el lenguaje cotidiano son difusos y reflejan las características de generalidad. Los siguientes son ejemplos":

"Una proposición imprecisa, no difusa: (x=y dentro de un e) equivale a (igualdad, (x, y), dentro de un e, 1)"

"Una proposición imprecisa difusa: (x aproximadamente igual a y) equivale a (igualdad, (x, y), aproximadamente, 1)"

"El término vago 'aproximadamente' designa algunos valores de e más o menos adecuados".

"Sobra decir que un término de información puede ser tanto vago como incierto al mismo tiempo, como lo testifica lo siguiente":

"Es probable que llueva un poco mañana, que equivale a (cantidad, lluvia mañana, un poco, probable)"

"Dado un conjunto de ítemes de información, la oposición entre precisión e incertidumbre se expresa en el hecho de que haciendo más preciso el contenido de una proposición, uno tiene a incrementar su incertidumbre. Y viceversa, el carácter incierto de un ítem de información conferirá en general una cierta imprecisión a las conclusiones que de él se deriven".

Podemos continuar la discusión diciendo que algo es "posible" cuando puede suceder y que es "probable" cuando le asignamos un "grado de confianza" a esa posibilidad. La forma tradicionalmente más sencilla de ilustrar esta diferencia se ejemplifica con el experimento de lanzar una moneda al aire; es posible que la moneda caiga como cara o como sello, pero es probable en un 50% que caiga cara o un 50% probable que caiga sello. En este caso no sabemos con certeza cuál será el resultado del experimento, pero sí conocemos el conjunto de resultados posible y probable del mismo. Ahora bien, como no conocemos con antelación cuál será el resultado de la realización del experimento, decimos que tenemos algún grado de "incertidumbre", que no es otra cosa que la falta de

"certidumbre" o "certeza" en su realización. Otros términos como "vaguedad", "inexactitud" o "imprecisión" se asocian con estados desconocidos de la naturaleza, como en [26]: "la imprecisión se refiere a la ausencia de conocimiento sobre el valor de un parámetro, y, por lo tanto, se expresa como un intervalo de tolerancia, que es el conjunto de valores posibles de los parámetros".

Los términos mencionados nos permiten llegar a la definición de conjuntos difusos y a diferenciarla de otros términos a primera vista similares. En [26] se dice "El carácter difuso no es un concepto elemental intuitivo u obvio y por tanto requiere alguna explicación. Este no debe confundirse con 'generalidad' o 'ambigüedad'. La generalización se refiere a la aplicación de un símbolo a múltiples objetos en un campo dado, ambigüedad a la asociación de un número finito de alternativas de significados posibles que tiene la misma forma fonética. Pero el carácter difuso de un símbolo está en la ausencia de límites bien definidos del conjunto de objetos al cual se aplica el símbolo". En los conjuntos difusos las verdades no son abruptas como en la lógica Booleana, sino que pueden ser graduales y varían según el grado de pertenencia que tenga el objeto comparado con el conjunto difuso. En [27] se establece lo siguiente: "Además de modelar la naturaleza gradual de las propiedades, los conjuntos difusos se pueden usar para representar estados incompletos del conocimiento. En este segundo uso, los conjuntos difusos interpretan el papel de una distribución de posibilidades que provee un ordenamiento completo de estados mutuamente exclusivos del mundo de acuerdo con sus respectivos niveles de posibilidad o plausibilidad. Por ejemplo, si sólo sabemos que 'John es alto' (pero no su altura precisa), donde el significado de 'alto' se describe, en contexto, por la función de pertenencia de un conjunto difuso (que es, μ_{alto}), entonces entre más grande es $\mu_{alto}(x)$, más grande es la posibilidad de que altura(John)=x; entre más pequeña es $\mu_{alto}(x)$, más pequeña es esta posibilidad.". En [28] se afirma que "en el mundo real existe mucho conocimiento difuso, id est, conocimiento que es vago, impreciso, incierto, ambiguo, inexacto o probabilístico en su naturaleza. El pensamiento y el razonamiento humanos envuelven frecuentemente información difusa, posiblemente originada de los

conceptos humanos inherentemente inexactos y que corresponde a experiencias similares, sino idénticas. En sistemas basados en la teoría de los conjuntos lógicos y la lógica bivalente, es muy difícil contestar algunas preguntas porque no tienen respuestas completamente ciertas. Los humanos, sin embargo, pueden dar respuestas satisfactorias, que son probablemente ciertas (...)", y continúa luego "la difusión ocurre cuando el límite de una pieza de información no tiene un límite definido. Por ejemplo, conceptos como *joven, alto, bueno* o *grande* son difusos. No hay un valor cuantitativo simple que defina el término *joven.* Para algunos, a los 25 años de edad se es *joven* y, para otros, a los 35 se es *joven.* De hecho, el concepto *joven* no tiene un límite claro y usualmente depende del contexto en el cual se considera. (...) En la teoría de los conjuntos difusos la pertenencia de un elemento a un conjunto puede ser parcial, id est, un elemento pertenece a un conjunto con un cierto grado (posibilidad) de pertenencia".

ANEXO 2: Programa de Búsquedas Difusas: Código Fuente

- Código de programa.prg

```
* Preparación del entorno
* Limpia memoria y escritorio
CLOSE ALL
CLEAR ALL
CLEAR

SET CENTURY ON
SET TALK OFF
SET ECHO OFF
SET NOTIFY OFF
SET SHADOWS ON
SET READBORDER ON
SET STATUS BAR ON
SET BELL ON
SET CONFIRM ON
SET POINT TO .
SET REPROCESS TO AUTOMATIC
SET EXCLUSIVE OFF

tiempo=SYS(2)
DO CASE
CASE tiempo>='0' .AND. tiempo<'43200'
	DO presenta WITH 'Buenos Días.  Bienvenido al Programa de Búsquedas difusas.'
CASE tiempo>'43200' .AND. tiempo<'64800'
	DO presenta WITH 'Buenas Tardes.  Bienvenido al Programa de Búsquedas difusas.'
CASE tiempo>'64800'
		DO presenta WITH 'Buenas Noches.  Bienvenido al Programa de Búsquedas difusas.'
ENDCASE


* Esconde menú del FoxPro
	MODI WINDOW SCREEN TITLE 'PROGRAMA DE BUSQUEDAS DIFUSAS' FONT 'Times',10
* Pantalla de presentación
	@ 8,90 SAY 'autor.bmp' BITMAP
	SET CLOCK TO 0,110
* Define menu principal del programa
DEFINE MENU principal NOMARGIN
DEFINE PAD texto	OF principal		PROMPT '    \<Versión Texto   ' MESSAGE 'Ingresar una búsqueda empleando la sintaxis SQL extendida'
DEFINE PAD grafico	OF principal		PROMPT 'V\<ersión Gráfica   'MESSAGE 'Ingresar una búsqueda seleccionando campos y operadores'
DEFINE PAD auxilia	OF principal		PROMPT 'A\<uxiliares   ' MESSAGE 'Permitir modificaciones en la base de datos'
DEFINE PAD ayuda	OF principal		PROMPT '\<Ayuda  '  MESSAGE 'Invocar la ayuda del programa'
DEFINE PAD terminar	OF principal		PROMPT '\<Terminar  '  MESSAGE 'Salir del programa'

ON SELECTION PAD texto		OF principal DO evalua
ON SELECTION PAD grafico	OF principal DO grafico
ON SELECTION PAD auxilia	OF principal DO auxilia
ON SELECTION PAD ayuda		OF principal DO ayudame
ON SELECTION PAD terminar	OF principal DO terminar

* Rutina ilimitada para mantener el menú activado
DO WHILE .T.

  ACTIVATE MENU principal
ENDDO
```

```
*FIN DEL PROGRAMA
                                                    *******  ******        ***  **
                                                    *******  ******   **** **
                                                    **                        **     **** **
                                                    *****                **     ** ****
                                                    *****                **     **  ***
                                                    **                     ******   **  ***
                                                    **                     ******   **   **


* * * * * * * * * * *
PROCEDURE terminar
* Interroga sobre el deseo de terminar la sesión de trabajo
        PRIVATE opcion,sino
        sino=1
        DO sino WITH sino,'Esto terminará su sesión de trabajo'
* Opción cancelar
        IF sino=2
          RETURN
        ENDIF
        MODI WINDOW SCREEN
* Cierra todo y limpia variables
        SET BELL TO
        CLOSE ALL
        CLEAR
        SET EXCLUSIVE OFF
        SET SYSMENU TO DEFAULT
        ON ERROR
        sino=1
        DO sino WITH sino,'¿Desea salir de FoxPro 2.6?'
        IF sino=1
                QUIT
        ELSE
                CANCEL
        ENDIF
RETURN



******************
PROCEDURE evalua
HIDE POPUP ALL
SET EXACT ON
USE interpre EXCL
SET SAFETY OFF
        ZAP
SET SAFETY ON
USE
SELE 0
USE interpre
        DEFINE WINDOW ingcodi FROM 0,0 TO 19,160 SYSTEM TITLE "INGRESO DE ORDEN SQL NUEVA" FONT 'fixedsys',9
        MOVE WINDOW ingcodi CENTER
        ACTIVATE WINDOW ingcodi
* Ciclo para registrar codigos
        DO WHILE .T.
                @ 1,2 SAY 'SELECT' GET mcodigo DEFA SPACE(254) SIZE 3,84
                @ 4,2 SAY 'FROM' GET mcodig1 DEFA SPACE(254) SIZE 3,84
                @ 7,2 SAY 'WHERE' GET mcodig2 DEFA SPACE(254) SIZE 3,84
                @ 12,16 GET mop  FUNC '*HT \<Grabar;\?\<Cancelar" DEFA 1 SIZE 1.5,14,5
                READ CYCLE
        * Cuando la opción es cancelar se sale del ciclo
                IF mop = 2
                        RELEASE WINDOW ingcodi
                        RETURN
                ENDIF
                mcodigo=ALLTRIM(mcodigo)
                mcamposp=mcodigo
                mcodig1=ALLTRIM(mcodig1)
                mcodig2=ALLTRIM(mcodig2)
                DIMENSION SINON(26)
                SINON(1)="~ALTO~"
                SINON(2)="~ALTA~"
                SINON(3)="~RAPIDO~"
                SINON(4)="~RAPIDA~"
                SINON(5)="~MUCHO~"
                SINON(6)="~MUCHA~"
                SINON(7)="~PEQUEÑA~"
                SINON(8)="~BAJO~"
```

```
SINON(9)="~BAJA~"
SINON(10)="~LENTO~"
SINON(11)="~LENTA~"
SINON(12)="~POCO~"
SINON(13)="~POCA~"
SINON(14)="~MUY  ALTO~"
SINON(15)="~MUY  ALTA~"
SINON(16)="~MUY  RAPIDO~"
SINON(17)="~MUY  RAPIDA~"
SINON(18)="~MUY  PEQUEÑA~"
SINON(19)="~MUY  BAJO~"
SINON(20)="~MUY  BAJA~"
SINON(21)="~MUY  LENTO~"
SINON(22)="~MUY  LENTA~"
SINON(23)="~MUY  POCO~"
SINON(24)="~MUY  POCA~"
SINON(25)="~CERCANA  A~"
SINON(26)="~ALREDEDOR  DE~"
FOR i=1 TO 6
        mocur=OCCURS(SINON(i),mcodig2)
        IF mocur<>0
                FOR j=1 TO mocur
                        mubica=AT(SINON(i),mcodig2)
                        primero=LEFT(mcodig2,mubica-1)
                        ultimo=RIGHT(mcodig2,LEN(mcodig2)-LEN(SINON(i))-mubica+1)
                        mcodig2=primero+"~GRANDE~"+ultimo
                ENDFOR
        ENDIF
ENDFOR
FOR i=7 TO 13
        mocur=OCCURS(SINON(i),mcodig2)
        IF mocur<>0
                FOR j=1 TO mocur
                        mubica=AT(SINON(i),mcodig2)
                        primero=LEFT(mcodig2,mubica-1)
                        ultimo=RIGHT(mcodig2,LEN(mcodig2)-LEN(SINON(i))-mubica+1)
                        mcodig2=primero+"~PEQUEÑO~"+ultimo
                ENDFOR
        ENDIF
ENDFOR
FOR i=14 TO 17
        mocur=OCCURS(SINON(i),mcodig2)
        IF mocur<>0
                FOR j=1 TO mocur
                        mubica=AT(SINON(i),mcodig2)
                        primero=LEFT(mcodig2,mubica-1)
                        ultimo=RIGHT(mcodig2,LEN(mcodig2)-LEN(SINON(i))-mubica+1)
                        mcodig2=primero+"~MUY  GRANDE~"+ultimo
                ENDFOR
        ENDIF
ENDFOR
FOR i=18 TO 24
        mocur=OCCURS(SINON(i),mcodig2)
        IF mocur<>0
                FOR j=1 TO mocur
                        mubica=AT(SINON(i),mcodig2)
                        primero=LEFT(mcodig2,mubica-1)
                        ultimo=RIGHT(mcodig2,LEN(mcodig2)-LEN(SINON(i))-mubica+1)
                        mcodig2=primero+"~MUY  PEQUEÑO~"+ultimo
                ENDFOR
        ENDIF
ENDFOR
FOR i=25 TO 26
        mocur=OCCURS(SINON(i),mcodig2)
        IF mocur<>0
                FOR j=1 TO mocur
                        mubica=AT(SINON(i),mcodig2)
                        primero=LEFT(mcodig2,mubica-1)
                        ultimo=RIGHT(mcodig2,LEN(mcodig2)-LEN(SINON(i))-mubica+1)
                        mcodig2=primero+"~CERCANO  A~"+ultimo
                ENDFOR
        ENDIF
ENDFOR
busca1=","
pos1=1
pos2=1
i=1
DO WHILE pos2<>0
        pos2=AT(busca1,mcodigo)
        IF pos2<>0
```

```
                                        pos3=LEN(mcodigo)
                                        DIMENSION atributo(i)
                                        atributo(i)=ALLTRIM(SUBSTR(mcodigo,pos1,pos2-1))
                                        i=i+1
                                        pos1=pos2+1
                                        mcodigo=ALLTRIM(SUBSTR(mcodigo,pos2+1,pos3))
                                        pos1=1
                        ENDIF
                ENDDO
                DIMENSION atributo(i)
                atributo(i)=mcodigo
                buscado=mcodig1+'.dbf'
                IF FILE(buscado)=.F.
                        DO mensaje WITH "El Archivo "+buscado+" No existe.  Intente con otro"
                        RELEASE WINDOW ingcodi
                        CLOSE ALL
                        RETURN
                ENDIF
                SELE 0
                USE &mcodig1
                REPLACE pertotal WITH 0 ALL
                FOR xx=1 TO 9
                        borrado='pertene'+ALLTRIM(STR(xx))
                        REPLACE &borrado WITH 0 ALL
                ENDFOR
                a=fcount()
                for l=1 to i
                        encontrado="no"
                        for k=1 to a
                                if UPPER(atributo(l))=UPPER(field(k))
                                        encontrado="si"
                                endif
                        endfor
                        if encontrado="no"
                                DO mensaje WITH "El campo "+atributo(l)+" no pertenece a la tabla "+mcodig1+".  Cámbielo y
reintente"
                                RELEASE WINDOW ingcodi
                                CLOSE ALL
                                RETURN
                        endif
                endfor
                DIMENSION operadif(21)
                operadif(1)=" ~MUY GRANDE~ "
                operadif(2)=" ~MUY PEQUEÑO~ "
                operadif(3)=" ~GRANDE~ "
                operadif(4)=" ~PEQUEÑO~ "
                operadif(5)=" ~CERCANO A~ "
                operadif(6)=" <> "
                operadif(7)=" >= "
                operadif(8)=" <= "
                operadif(9)=" = "
                operadif(10)=" > "
                operadif(11)=" < "
                operadif(12)=" ~Y~ "
                operadif(13)=" ~O~ "
                operadif(14)=" ~NO~ "
                operadif(15)="("
                operadif(16)=")"
                operadif(17)=","
                operadif(18)=""""
                operadif(19)=" AND "
                operadif(20)=" OR "
                operadif(21)=" NOT "
                DIMENSION tipo(21)
                tipo(1)='O'
                tipo(2)='O'
                tipo(3)='O'
                tipo(4)='O'
                tipo(5)='O'
                tipo(6)='P'
                tipo(7)='P'
                tipo(8)='P'
                tipo(9)='P'
                tipo(10)='P'
                tipo(11)='P'
                tipo(12)='D'
                tipo(13)='D'
                tipo(14)='D'
                tipo(15)='R'
                tipo(16)='R'
```

```
tipo(17)='R'
tipo(18)='R'
tipo(19)='U'
tipo(20)='U'
tipo(21)='U'
DIMENSION cantidad(21)
I=0
total=0
mayor=1
FOR I=1 TO 21
	cantidad(I)=OCCURS(operadif(I),mcodig2)
	total=total+cantidad(I)
	IF cantidad(I)>mayor
		mayor=cantidad(I)
	ENDIF
ENDFOR
DIMENSION posicion(21,mayor)
I=0
J=0
FOR I=1 TO 21
	FOR J=1 TO mayor
		posicion(I,J)=0
	ENDFOR
ENDFOR
I=0
J=0
K=1
L=1
FOR I=1 TO 21
	pos3=0
	valor=mcodig2
	FOR J=1 TO cantidad(I)
		pos4=len(valor)
		pos2=AT(operadif(I),valor)
		pos3=pos2+pos3
		posicion(I,J)=pos3
		DIMENSION ordenam(k)
		DIMENSION ordconj(l)
		IF (i=12 OR i=13 OR i=14 OR i=19 OR i=20 OR i=21) AND pos3<>0
			ordconj(l)=pos3
			l=l+1
		ENDIF
		DIMENSION campo(k)
		DIMENSION valorcam(k)
		ordenam(K)=pos3
		SELE interpre
		APPEND BLANK
		REPLACE elemento WITH operadif(i), posicion WITH pos3, tipo WITH tipo(i), longitud WITH
len(operadif(i))
		SELE &mcodig1
		K=K+1
		pos3=pos3-1+len(operadif(i))
		IF pos2+len(operadif(I))<pos4
			valor=SUBSTR(valor,pos2+len(operadif(I)),pos4)
		ENDIF
	ENDFOR
ENDFOR
DIMENSION ordenam(k)
DIMENSION campo(k)
DIMENSION valorcam(k)
ordenam(k)=0
DIMENSION ordenam(k+1)
ordenam(k+1)=len(mcodig2)+1
M=K-1
=ASORT(ordenam)
cant=0
FOR J=1 TO mayor
	IF posicion(18,j)<>0
		cant=cant+1
	ENDIF
ENDFOR
IF cant/2-INT(cant/2)<>0
	DO mensaje WITH "Falta '"
	RELEASE WINDOW ingcodi
	CLOSE ALL
	RETURN
ENDIF
cant=0
FOR J=1 TO mayor
	IF posicion(15,j)<>0
```

```
                        cant=cant+1
                ENDIF
        ENDFOR
        cant1=0
        FOR J=1 TO mayor
                IF posicion(16,j)<>0
                        cant1=cant1+1
                ENDIF
        ENDFOR
        IF cant<>cant1
                IF cant>cant1
                        DO mensaje WITH "Falta )"
                        RELEASE WINDOW ingcodi
                        CLOSE ALL
                        RETURN
                ELSE
                        IF cant<cant1
                                DO mensaje WITH "Falta ("
                                RELEASE WINDOW ingcodi
                                CLOSE ALL
                                RETURN
                        ENDIF
                ENDIF
        ENDIF
        FOR I=1 TO 11
                FOR J=1 TO mayor
                        IF posicion(I,J)<>0
                                pvec=ASCAN(ordenam,posicion(I,J))
                                campo(pvec)=SUBSTR(mcodig2,ordenam(pvec-1)+1,ordenam(pvec)-ordenam(pvec-
1)-1)
                                a=fcount()
                                encontrado="no"
                                for k=1 to a
                                        if UPPER(campo(pvec))=UPPER(field(k))
                                                encontrado="si"
                                                mtipo=TYPE(field(k))
                                                SELE interpre
                                                APPEND BLANK
                                                REPLACE elemento WITH campo(pvec), posicion WITH
ordenam(pvec-1)+1, tipo WITH 'C', longitud WITH ordenam(pvec)-ordenam(pvec-1)-1
                                                SELE &mcodig1
                                                IF mtipo="C" OR mtipo="D"
                                                        IF i<=5
                                                                DO mensaje WITH "No se pueden usar
campos alfanuméricos con operadores difusos"
                                                                RELEASE WINDOW ingcodi
                                                                CLOSE ALL
                                                                RETURN
                                                        ELSE
                                                                IF
SUBSTR(mcodig2,ordenam(pvec+1),1)<>"'" AND i>=6 AND i<=11
                                                                        DO mensaje WITH "El
campo "+campo(pvec)+" es alfanumérico y le falta el apóstrofe"
                                                                        RELEASE WINDOW
ingcodi
                                                                        CLOSE ALL
                                                                        RETURN
                                                                ELSE

        valorcam(k)=SUBSTR(mcodig2,ordenam(pvec+1)+1,ordenam(pvec+2)-ordenam(pvec+1)-1)
                                                                        SELE interpre
                                                                        APPEND BLANK
                                                                        REPLACE elemento WITH
valorcam(k), posicion WITH ordenam(pvec+1)+1, tipo WITH 'V', longitud WITH ordenam(pvec+2)-ordenam(pvec+1)-1
                                                                        SELE &mcodig1
                                                                ENDIF
                                                        ENDIF
                                                ELSE
                                                        IF i<=4

        mnpos=ASCAN(posicion,ordenam(pvec+1))
                                                                mnele=INT((mnpos-1)/mayor)+1
                                                                IF mnele=15

        mnpos1=ASCAN(posicion,ordenam(pvec+2))
                                                                        mnele1=INT((mnpos1-
1)/mayor)+1

        mnpos2=ASCAN(posicion,ordenam(pvec+3))
```

```
                                                                    mnele2=INT((mnpos2-
1)/mayor)+1
                                                                    IF mnele1=17 AND
mnele2=16
        minimo=VAL(ALLTRIM(SUBSTR(mcodig2,ordenam(pvec+1)+1,ordenam(pvec+2)-ordenam(pvec+1)-1)))
                                                                        SELE interpre
                                                                        APPEND
BLANK
                                                                        REPLACE
elemento WITH ALLTRIM(SUBSTR(mcodig2,ordenam(pvec+1)+1,ordenam(pvec+2)-ordenam(pvec+1)-1)), posicion WITH
ordenam(pvec+1)+1, tipo WITH 'I', longitud WITH ordenam(pvec+2)-ordenam(pvec+1)-1
                                                                        SELE &mcodig1
        maximo=VAL(ALLTRIM(SUBSTR(mcodig2,ordenam(pvec+2)+1,ordenam(pvec+3)-ordenam(pvec+2)-1)))
                                                                        SELE interpre
                                                                        APPEND
BLANK
                                                                        REPLACE
elemento WITH ALLTRIM(SUBSTR(mcodig2,ordenam(pvec+2)+1,ordenam(pvec+3)-ordenam(pvec+2)-1)), posicion WITH
ordenam(pvec+2)+1, tipo WITH 'M', longitud WITH ordenam(pvec+3)-ordenam(pvec+2)-1
                                                                        SELE &mcodig1
        mnpos3=ASCAN(POSICION,ordenam(pvec+3)+1)
                                                                        IF mnpos3=0
        mnpos4=ASCAN(posicion,ordenam(pvec+4))
        mnele4=INT((mnpos4-1)/mayor)+1
        tuplas=VAL(ALLTRIM(SUBSTR(mcodig2,ordenam(pvec+3)+1,ordenam(pvec+4)-ordenam(pvec+3)-1)))
        SELE interpre
        APPEND BLANK
        REPLACE elemento WITH ALLTRIM(SUBSTR(mcodig2,ordenam(pvec+3)+1,ordenam(pvec+4)-ordenam(pvec+3)-1)), posicion
WITH ordenam(pvec+3)+1, tipo WITH 'N', longitud WITH ordenam(pvec+4)-ordenam(pvec+3)-1
        SELE &mcodig1
                                                                        IF
mnele4=17
        mnpos5=ASCAN(posicion,ordenam(pvec+5))
        mnele5=INT((mnpos5-1)/mayor)+1
        umbral=VAL(ALLTRIM(SUBSTR(mcodig2,ordenam(pvec+4)+1,ordenam(pvec+5)-ordenam(pvec+4)-1)))
        SELE interpre
        APPEND BLANK
        REPLACE elemento WITH ALLTRIM(SUBSTR(mcodig2,ordenam(pvec+4)+1,ordenam(pvec+5)-ordenam(pvec+4)-1)), posicion
WITH ordenam(pvec+4)+1, tipo WITH 'T', longitud WITH ordenam(pvec+5)-ordenam(pvec+4)-1
        SELE &mcodig1
        ENDIF
                                                                        ENDIF
                                                                    ELSE
                                                                        DO mensaje
WITH 'Error en la definición de los valores de la función de pertenencia'
                                                                        RELEASE
WINDOW ingcodi
                                                                        CLOSE ALL
                                                                        RETURN
                                                                    ENDIF
                                                                ELSE
        sig=posicion(i,j)+len(operadif(i))
        posig=ASCAN(ordenam,sig)
                                                                    IF posig=0
                                                                        DO mensaje
WITH 'no puede seguir un número después de un operador difuso'
                                                                        RELEASE
WINDOW ingcodi
                                                                        CLOSE ALL
                                                                        RETURN
```

```
                                                ENDIF
                                            ENDIF
                                        ELSE
                                            IF i>=6 AND i<=11
        mnpos=ASCAN(posicion,ordenam(pvec+1))
                                                sig=pvec+len(operadif(i))+1
        posig=ASCAN(ordenam,sig)
                                                IF posig<>0
                                                    DO      mensaje
WITH 'No puede seguir un operador después de '+operadif(i)
                                                    RELEASE
WINDOW ingcodi
                                                    CLOSE ALL
                                                    RETURN
                                                ELSE
        mvalor=VAL(SUBSTR(mcodig2,sig,mnpos-sig))
        valorcam(k)=SUBSTR(mcodig2,ordenam(pvec)+2,ordenam(pvec+1)-ordenam(pvec)-1)
                                                    SELE interpre
                                                    APPEND
BLANK
                                                    REPLACE
elemento WITH valorcam(k), posicion WITH ordenam(pvec)+1, tipo WITH 'V', longitud WITH ordenam(pvec+1)-ordenam(pvec)-1
                                                    SELE &mcodig1
                                                ENDIF
                                            ENDIF
                                        ENDIF
                                    ENDIF
                                ENDIF
                            ENDFOR
                            if encontrado="no"
                                DO mensaje WITH "El campo "+campo(pvec)+" no pertenece a la tabla
"+mcodig1+".  Cámbielo y reintente"
                                RELEASE WINDOW ingcodi
                                CLOSE ALL
                                RETURN
                            endif
                        ENDIF
                    ENDFOR
                ENDFOR
                EXIT
        ENDDO
* Cierra la rutina
        RELEASE WINDOW ingcodi
        CLOSE ALL
        DO interpr
RETURN

***********************************
PROCEDURE interpr

CLOSE ALL
SELE 0
USE frecuencia ALIAS cosa SHARED
SELE 0
USE &mcodig1 SHARED ALIAS basedat
REPLACE pertene1 WITH 0, pertene2 WITH 0, pertene3 WITH     0, pertene4 WITH 0, pertene5 WITH 0 ALL
REPLACE pertene6 WITH 0, pertene7 WITH 0, pertene8 WITH     0, pertene9 WITH 0, pertotal WITH 0 ALL
SELE 0
USE interpre SHARED
SET ORDER TO posicion
SET FILTER TO tipo='N'
CALCULATE MIN(VAL(elemento)) TO n
SET FILTER TO tipo='U' OR tipo='D'
melemento='pertotal>=0.5 '
mtipo='D'
anterior=0
valor=1
GO TOP
SCAN
        msepara=elemento
        mtipo=tipo
        mposicion=posicion
        mreg=RECNO()
        IF mtipo='U'
                SET FILTER TO posicion>anterior AND posicion<=mposicion AND tipo='P'
```

```
		COUNT TO a
		IF a<>0
			SET FILTER TO posicion>anterior AND posicion<mposicion
			melemento=melemento+" AND "
			SCAN
				melemento=melemento+ALLTRIM(elemento)
			ENDSCAN
		ELSE
			SET FILTER TO posicion>anterior AND posicion<mposicion

		ENDIF
ELSE
		DIMENSION separado(valor)
		IF msepara=' ~Y~ '
			separado(valor)='min'
		ELSE
			separado(valor)='max'
		ENDIF
		SET FILTER TO posicion>anterior AND posicion<=mposicion AND tipo='C'
		GO TOP
		campo=elemento
		SET FILTER TO posicion>anterior AND posicion<=mposicion AND tipo='O'
		GO TOP
		mopera=ALLTRIM(elemento)
		SET FILTER TO posicion>anterior AND posicion<=mposicion AND (tipo='I' OR tipo='M')
		COUNT TO a
		SELE basedat
		IF a=0
			SELE basedat
			cosa="pertene"+alltrim(str(valor))
			CALCULATE MAX(&campo) TO valmax
			CALCULATE MIN(&campo) TO valmin
			COUNT TO numele
			SELE cosa
			GO TOP
			i=0
			SCAN
				REPLACE valor WITH valmin+(valmax-valmin)/10*i
				i=i+1
			ENDSCAN
			GO TOP
			mfant=0
			REPLACE frecuencia WITH 0
			SKIP
			SCAN
				mvalfrec=valor
				SELE basedat
				COUNT FOR &campo<=mvalfrec TO Y
				SELE cosa
				REPLACE frecuencia WITH Y
				REPLACE frecnorm WITH Y/numele
				REPLACE frecind WITH Y-mfant
				mfant=y
			ENDSCAN
			GO TOP
			SCAN
				mregno=RECNO()
				mpinic=frecind
				mposib=0
				GO TOP
				FOR I=1 TO 11
					IF RECNO()=mregno OR frecind<mpinic
						mposib=mposib+frecind
					ELSE
						mposib=mposib+mpinic
					ENDIF
					SKIP
				ENDFOR
				LOCATE FOR RECNO()=mregno
				REPLACE posibilida WITH mposib/numele
			ENDSCAN
			SUM posibilida TO totposib
			GO TOP
			SCAN
				mregno=RECNO()
				CALCULATE sum(posibilida) FOR RECNO()<=mregno to msumpos
				LOCATE FOR RECNO()=mregno
				REPLACE posibilnor WITH msumpos/totposib
			ENDSCAN
			SELE basedat
```

```
                                GO TOP
                                SCAN
                                        mvalcamp=&campo
                                        SELE cosa
                                        LOCATE FOR valor=mvalcamp
                                        IF FOUND()=.T.
                                                mfrec=posibilnor
                                        ELSE
                                                SET FILTER TO valor<mvalcamp+(valmax-valmin)/10 AND valor>mvalcamp-
(valmax-valmin)/10
                                                CALCULATE max(valor) TO C
                                                CALCULATE min(valor) TO D
                                                CALCULATE max(posibilnor) TO E
                                                CALCULATE min(posibilnor) TO F
                                                mfrec=f+(mvalcamp-d)*(e-f)/(c-d)
                                                SET FILTER TO
                                        ENDIF
                                        SELE basedat
                                        IF mopera='~PEQUEÑO~' OR mopera='~MUY PEQUEÑO~'
                                                mfrec=1-mfrec
                                        ENDIF
                                        IF mopera='~MUY PEQUEÑO~' OR mopera='~MUY GRANDE~'
                                                mfrec=mfrec*mfrec
                                        ENDIF
                                        REPLACE &cosa WITH mfrec
                                ENDSCAN
                        ELSE
                                SELE interpre
                                SET FILTER TO posicion>anterior AND posicion<=mposicion AND tipo='I'
                                GO TOP
                                minimo=VAL(elemento)
                                SET FILTER TO posicion>anterior AND posicion<=mposicion AND tipo='M'
                                GO TOP
                                maximo=VAL(elemento)
                                SELE basedat
                                cosa="pertene"+alltrim(str(valor))
                                SCAN
                                        mvalor=&campo
                                        IF mvalor>=0 AND mvalor<=minimo
                                                mcalculo=0
                                        ELSE
                                                IF mvalor>minimo AND mvalor<maximo
                                                        mcalculo=(mvalor-minimo)/(maximo-minimo)
                                                ELSE
                                                        mcalculo=1
                                                ENDIF
                                        ENDIF
                                        IF mopera='~PEQUEÑO~' OR mopera='~MUY PEQUEÑO~'
                                                mcalculo=1-mcalculo
                                        ENDIF
                                        IF mopera='~MUY PEQUEÑO~' OR mopera='~MUY GRANDE~'
                                                mcalculo=mcalculo*mcalculo
                                        ENDIF
                                        REPLACE &cosa WITH mcalculo
                                ENDSCAN
                        ENDIF
                        valor=valor+1
                ENDIF
                SELE interpre
                SET FILTER TO tipo='U' OR tipo='D'
                LOCATE FOR RECNO()=mreg
                anterior=mposicion
ENDSCAN
IF mtipo='D'
        SET FILTER TO posicion>anterior AND tipo='C'
        GO TOP
        campo=elemento
        SET FILTER TO posicion>anterior AND tipo='O'
        GO TOP
        mopera=ALLTRIM(elemento)
        SET FILTER TO posicion>anterior AND (tipo='I' OR tipo='M')
        COUNT TO a
        SELE basedat
        IF a=0
                SELE basedat
                cosa="pertene"+alltrim(str(valor))
                CALCULATE MAX(&campo) TO valmax
                CALCULATE MIN(&campo) TO valmin
                COUNT TO numele
                SELE cosa
```

```
		GO TOP
		i=0
		SCAN
			REPLACE valor WITH valmin+(valmax-valmin)/10*i
			i=i+1
		ENDSCAN
		GO TOP
		mfant=0
		REPLACE frecuencia WITH 0
		SKIP
		SCAN
			mvalfrec=valor
			SELE basedat
			COUNT FOR &campo<=mvalfrec TO Y
			SELE cosa
			REPLACE frecuencia WITH Y
			REPLACE frecnorm WITH Y/numele
			REPLACE frecind WITH Y-mfant
			mfant=y
		ENDSCAN
		GO TOP
		SCAN
			mregno=RECNO()
			mpinic=frecind
			mposib=0
			GO TOP
			FOR I=1 TO 11
				IF RECNO()=mregno OR frecind<mpinic
					mposib=mposib+frecind
				ELSE
					mposib=mposib+mpinic
				ENDIF
				SKIP
			ENDFOR
			LOCATE FOR RECNO()=mregno
			REPLACE posibilida WITH mposib/numele
		ENDSCAN
		SUM posibilida TO totposib
		GO TOP
		SCAN
			mregno=RECNO()
			CALCULATE sum(posibilida) FOR RECNO()<=mregno to msumpos
			LOCATE FOR RECNO()=mregno
			REPLACE posibilnor WITH msumpos/totposib
		ENDSCAN
		SELE basedat
		GO TOP
		SCAN
			mvalcamp=&campo
			SELE cosa
			LOCATE FOR valor=mvalcamp
			IF FOUND()=.T.
				mfrec=posibilnor
			ELSE
				SET FILTER TO valor<mvalcamp+(valmax-valmin)/10 AND valor>mvalcamp-(valmax-valmin)/10
				CALCULATE max(valor) TO C
				CALCULATE min(valor) TO D
				CALCULATE max(posibilnor) TO E
				CALCULATE min(posibilnor) TO F
				mfrec=f+(mvalcamp-d)*(e-f)/(c-d)
				SET FILTER TO
			ENDIF
			SELE basedat
			IF mopera='~PEQUEÑO~' OR mopera='~MUY PEQUEÑO~'
				mfrec=1-mfrec
			ENDIF
			IF mopera='~MUY PEQUEÑO~' OR mopera='~MUY GRANDE~'
				mfrec=mfrec*mfrec
			ENDIF
			REPLACE &cosa WITH mfrec
		ENDSCAN
ELSE
		SELE interpre
		SET FILTER TO posicion>anterior AND tipo='I'
		GO TOP
		minimo=VAL(elemento)
		SET FILTER TO posicion>anterior AND tipo='M'
		GO TOP
		maximo=VAL(elemento)
		SELE basedat
```

```
                cosa="pertene"+alltrim(str(valor))
                SCAN
                        mvalor=&campo
                        IF mvalor>=0 AND mvalor<=minimo
                                mcalculo=0
                        ELSE
                                IF mvalor>minimo AND mvalor<maximo
                                        mcalculo=(mvalor-minimo)/(maximo-minimo)
                                ELSE
                                        mcalculo=1
                                ENDIF
                        ENDIF
                        IF mopera='~PEQUEÑO~' OR mopera='~MUY PEQUEÑO~'
                                mcalculo=1-mcalculo
                        ENDIF
                        IF mopera='~MUY PEQUEÑO~' OR mopera='~MUY GRANDE~'
                                mcalculo=mcalculo*mcalculo
                        ENDIF
                        REPLACE &cosa WITH mcalculo
                ENDSCAN
        ENDIF
ELSE
        SET FILTER TO posicion>anterior AND posicion<=mposicion AND tipo='P'
        COUNT TO a
        IF a<>0
                SET FILTER TO posicion>anterior
                SCAN
                        melemento=melemento+ALLTRIM(elemento)
                ENDSCAN
        ENDIF
ENDIF
evalor=valor-1
IF evalor=0
        REPLACE pertotal WITH pertene1 ALL
ELSE
        FOR i=1 TO valor-1
                cosa1='pertene'+ALLTRIM(STR(i))
                cosa2='pertene'+ALLTRIM(STR(i+1))
                REPLACE pertotal WITH &separado(i) (&cosa1,&cosa2) ALL
        ENDFOR
ENDIF
SET ORDER TO indicar
IF n=0
        IF LEN(melemento)<>0
                SET FILTER TO &melemento
        ENDIF
ELSE
        IF LEN(melemento)<>0
                SET FILTER TO &melemento
        ENDIF
        MNUM=n
        GO TOP
        canti=ALLTRIM(STR(RECNO()))
        canti1="RECNO()="+canti
        FOR I=1 TO mnum-1
                SKIP
                canti=ALLTRIM(STR(RECNO()))
                canti1=canti1+" OR RECNO()="+canti
        ENDFOR
        IF LEN(melemento)<>0
                SET FILTER TO &melemento AND &canti1
        ELSE
                SET FILTER TO &canti1
        ENDIF
ENDIF
BROWSE FIELDS &mcamposp, pertotal
CLOSE ALL
RETURN



* * * * * * * * *
PROCEDURE mensaje
* Presenta mensajes de advertencia o de aviso
        PARAMETER  mensaje
        PRIVATE longitud,opcion
        longitud=LEN(ALLTRIM(mensaje))+4
* Longitud mínima de 14 caracteres
        IF longitud<14
```

```
                longitud=14
        ENDIF
        DEFINE WINDOW mensaje FROM 0,0 TO 6,longitud+10 TITLE ' ¡ATENCIÓN! ' SYSTEM
        MOVE WINDOW mensaje TO 10,1
        ACTIVATE WINDOW mensaje
        @ 1,9 SAY mensaje COLOR R+/W*
        @ 3,longitud/2+2 GET opcion FUNCTION '*HT Aceptar' DEFAULT 1 SIZE 1.5,10
        READ CYCLE
        RELEASE WINDOW mensaje
RETURN


* * * * * * * *
PROCEDURE sino
* Procedimiento para opciones de si o no
        PARAMETER  sino,mensaje
        PRIVATE longitud
        longitud=LEN(ALLTRIM(mensaje))+10
* Longitud mínima de 14 caracteres
        IF longitud<14
                longitud=14
        ENDIF
        DEFINE WINDOW sino FROM 0,0 TO 6,longitud+8        TITLE ' ¡ATENCIÓN! ' SYSTEM FLOAT
        MOVE WINDOW sino TO 10,1
        ACTIVATE WINDOW sino
        @ 1,9 SAY mensaje COLOR B+/W*
        @ 3,longitud/2-5 GET sino FUNCTION '*HT \!\<Si;\?\<No' DEFAULT sino SIZE 1.5,10
        READ CYCLE
        RELEASE WINDOW sino
* Regresa 1 para Si-Aceptar, 0 para No-Cancelar
RETURN sino

* * * * * * * * *
PROCEDURE presenta
* Presenta mensajes de ingreso al sistema
        PARAMETER  presenta
        PRIVATE longitud,opcion
        longitud=LEN(ALLTRIM(presenta))+4
* Longitud mínima de 14 caracteres
        IF longitud<14
                longitud=14
        ENDIF
        @ 1,45 SAY 'un.bmp' BITMAP
        @ 15,10 SAY presenta COLOR B+/W* FONT 'Times',18
        @ 20,45 GET opcion FUNCTION '*HT Ingresar' DEFAULT 1 SIZE 1.5,10
        READ CYCLE
        CLEAR
RETURN


******************************
PROCEDURE ayudame
HIDE POPUP ALL
CLOSE ALL
RUN c:\archiv~1\intern~1\iexplore.exe c:\busqdifu\ayuda.htm
RETURN


******************************
PROCEDURE grafico
HIDE POPUP ALL
PUBLIC mtipoc1, mtipoc2, mtipoc3, mtipoc4
PUBLIC val, val2, val3, val4
PUBLIC min, min2, min3, min4
PUBLIC max, max2, max3, max4
PUBLIC tup, tup2, tup3, tup4
PUBLIC tol, tol2, tol3, tol4
PUBLIC O4, O5, O7, O8, OA, OB
mtipoc1=" "
mtipoc2=" "
mtipoc3=" "
mtipoc4=" "
min=0.00
min2=0.00
min3=0.00
min4=0.00
max=0.00
max2=0.00
max3=0.00
max4=0.00
```

```
tup=0
tup2=0
tup3=0
tup4=0
tol=0.00
tol2=0.00
tol3=0.00
tol4=0.00



SET EXACT ON
USE interpre EXCL
SET SAFETY OFF
        ZAP
SET SAFETY ON
USE
SELE 0
USE interpre
marchivo=GETFILE("DBF", "Seleccione una base", "Examinar", 1)
mcodig1=ALLTRIM(marchivo)
SELE 0
USE (marchivo)
FOR I=1 TO FCOUNT()-10
        DIMENSION campos(I)
        campos(I)=ALLTRIM(FIELD(I))
ENDFOR
=ASORT(campos)
FOR I=1 TO FCOUNT()-10
        DIMENSION campos1(I)
        campos1(I)=ALLTRIM(FIELD(I))
ENDFOR
DIMENSION campos1(FCOUNT()-9)
campos1(FCOUNT()-9)=" "
=ASORT(campos1)
DIMENSION conector(7)
conector(1)=" ~Y~ "
conector(2)=" ~O~ "
conector(3)=" ~NO~ "
conector(4)=" AND "
conector(5)=" OR "
conector(6)=" NOT "
conector(7)=" "
DIMENSION tipo(6)
tipo(1)='D'
tipo(2)='D'
tipo(3)='D'
tipo(4)='U'
tipo(5)='U'
tipo(6)='U'
DIMENSION operador(38)
operador(1)="  ~MUY GRANDE~ "
operador(2)="  ~MUY PEQUEÑO~ "
operador(3)="  ~GRANDE~ "
operador(4)="  ~PEQUEÑO~  "
operador(5)="  ~CERCANO A~ "
operador(6)="  <> "
operador(7)="  >= "
operador(8)="  <= "
operador(9)="  = "
operador(10)="  > "
operador(11)="  < "
operador(12)="  "
operador(13)="~ALTO~"
operador(14)="~ALTA~"
operador(15)="~RAPIDO~"
operador(16)="~RAPIDA~"
operador(17)="~MUCHO~"
operador(18)="~MUCHA~"
operador(19)="~PEQUEÑA~"
operador(20)="~BAJO~"
operador(21)="~BAJA~"
operador(22)="~LENTO~"
operador(23)="~LENTA~"
operador(24)="~POCO~"
operador(25)="~POCA~"
operador(26)="~MUY  ALTO~"
operador(27)="~MUY  ALTA~"
operador(28)="~MUY  RAPIDO~"
operador(29)="~MUY  RAPIDA~"
```

```
operador(30)="~MUY  PEQUEÑA~"
operador(31)="~MUY  BAJO~"
operador(32)="~MUY  BAJA~"
operador(33)="~MUY  LENTO~"
operador(34)="~MUY  LENTA~"
operador(35)="~MUY  POCO~"
operador(36)="~MUY  POCA~"
operador(37)="~CERCANA  A~"
operador(38)="~ALREDEDOR  DE~"
DIMENSION tipop(11)
tipop(1)='O'
tipop(2)='O'
tipop(3)='O'
tipop(4)='O'
tipop(5)='O'
tipop(6)='P'
tipop(7)='P'
tipop(8)='P'
tipop(9)='P'
tipop(10)='P'
tipop(11)='P'
* Definición de la ventana de entrada de datos
        DEFINE WINDOW ingsql FROM 0,0 TO 25,155 SYSTEM TITLE "SELECCIONE LOS ELEMENTOS DE LA ORDEN SQL" FONT
'fixedsys',9
        MOVE WINDOW ingsql CENTER
        ACTIVATE WINDOW ingsql
        @ 1,1 SAY "Seleccione los campos para mostrar en la búsqueda"
        @ 3,10 GET cam1 PICT '@^' FROM campos1 DEFA 1 SIZE 4,8
        @ 3,19 GET cam2 PICT '@^' FROM campos1 DEFA 1 SIZE 4,8
        @ 3,32 GET cam3 PICT '@^' FROM campos1 DEFA 1 SIZE 4,8
        @ 3,42 GET cam4 PICT '@^' FROM campos1 DEFA 1 SIZE 4,8
        @ 3,53 GET cam5 PICT '@^' FROM campos1 DEFA 1 SIZE 4,8
        @ 3,64 GET cam6 PICT '@^' FROM campos1 DEFA 1 SIZE 4,8
        @ 4.5,1 SAY "Conector Campo  Operador  Valor  Mín(Cerca)  Máximo  # Tuplas  Tolerancia"
        @ 6.5,10 GET O1 PICT '@^' FROM campos SIZE 4,8 DEFA 1
        @ 6.5,19 GET O2 PICT '@^' FROM operador SIZE 4,10 DEFA 12 VALID valida1()
        @ 8.5,1 GET O3 PICT '@^' FROM conector SIZE 4,8 DEFA 7 VALID valida2()
        @ 10.5,1 GET O6 PICT '@^' FROM conector SIZE 4,8 DEFA 7 VALID valida4()
        @ 12.5,1 GET O9 PICT '@^' FROM conector SIZE 4,8 DEFA 7 VALID valida6()
        @ 18,25 GET mop  FUNC "*HT \<Continuar;\?\<Terminar" DEFA 1 SIZE 1.5,14,5
        READ CYCLE
* Cuando la opción es cancelar se sale del ciclo
        IF mop = 2
                RELEASE WINDOW ingsql
                CLOSE ALL
                RETURN
        ENDIF
        mcamposp=" "
        IF cam1<>1
                mcamposp=mcamposp+ALLTRIM(campos1(cam1))
        ENDIF
        IF cam2<>1
                mcamposp=mcamposp+", "+ALLTRIM(campos1(cam2))
        ENDIF
        IF cam3<>1
                mcamposp=mcamposp+", "+ALLTRIM(campos1(cam3))
        ENDIF
        IF cam4<>1
                mcamposp=mcamposp+", "+ALLTRIM(campos1(cam4))
        ENDIF
        IF cam5<>1
                mcamposp=mcamposp+", "+ALLTRIM(campos1(cam5))
        ENDIF
        IF cam6<>1
                mcamposp=mcamposp+", "+ALLTRIM(campos1(cam6))
        ENDIF
        mcamposp=ALLTRIM(mcamposp)
        SELE interpre
        APPEND BLANK
        REPLACE elemento WITH campos(o1), posicion WITH 1, tipo WITH "C"
        APPEND BLANK
        DO CASE
                CASE O2>=13 AND O2<=18
                        O2=3
                CASE O2>=19 AND O2<=25
                        O2=4
                CASE O2>=26 AND O2<=29
                        O2=1
                CASE O2>=30 AND O2<=36
                        O2=2
```

```
        CASE O2>=37 AND O2<=38
                O2=5
ENDCASE
REPLACE elemento WITH operador(o2), posicion WITH 2, tipo WITH tipop(o2)
IF O2>5 AND O2<13
        IF mtipoc1="N"
                APPEND BLANK
                REPLACE elemento WITH ALLTRIM(STR(val)), posicion WITH 3, tipo WITH "V"
        ELSE
                APPEND BLANK
                REPLACE elemento WITH """+ALLTRIM(val)+""", posicion WITH 3, tipo WITH "V"
        ENDIF
ELSE
        IF min<>0
                APPEND BLANK
                REPLACE elemento WITH ALLTRIM(STR(min,7,2)), posicion WITH 4, tipo WITH "I"
        ENDIF
        IF max<>0
                APPEND BLANK
                REPLACE elemento WITH ALLTRIM(STR(max,7,2)), posicion WITH 5, tipo WITH "M"
        ENDIF
        IF tup<>0
                APPEND BLANK
                REPLACE elemento WITH ALLTRIM(STR(tup)), posicion WITH 6, tipo WITH "N"
        ENDIF
        IF tol<>0
                APPEND BLANK
                REPLACE elemento WITH ALLTRIM(STR(tol,7,2)), posicion WITH 7, tipo WITH "T"
        ENDIF
ENDIF
IF O3<>7
        APPEND BLANK
        REPLACE elemento WITH conector(O3), posicion WITH 8, tipo WITH tipo(O3)
        APPEND BLANK
        REPLACE elemento WITH campos(o4), posicion WITH 9, tipo WITH "C"
        APPEND BLANK
        DO CASE
                CASE O5>=13 AND O5<=18
                        O5=3
                CASE O5>=19 AND O5<=25
                        O5=4
                CASE O5>=26 AND O5<=29
                        O5=1
                CASE O5>=30 AND O5<=36
                        O5=2
                CASE O5>=37 AND O5<=38
                        O5=5
        ENDCASE
        REPLACE elemento WITH operador(o5), posicion WITH 10, tipo WITH tipop(o5)
        IF O5>5 AND O5<13
                IF mtipoc2="N"
                        APPEND BLANK
                        REPLACE elemento WITH ALLTRIM(STR(val)), posicion WITH 11, tipo WITH "V"
                ELSE
                        APPEND BLANK
                        REPLACE elemento WITH """+ALLTRIM(val)+""", posicion WITH 11, tipo WITH "V"
                ENDIF
        ELSE
                IF min2<>0
                        APPEND BLANK
                        REPLACE elemento WITH ALLTRIM(STR(min2,7,2)), posicion WITH 12, tipo WITH "I"
                ENDIF
                IF max2<>0
                        APPEND BLANK
                        REPLACE elemento WITH ALLTRIM(STR(max2,7,2)), posicion WITH 13, tipo WITH "M"
                ENDIF
                IF tup2<>0
                        APPEND BLANK
                        REPLACE elemento WITH ALLTRIM(STR(tup2)), posicion WITH 14, tipo WITH "N"
                ENDIF
                IF tol2<>0
                        APPEND BLANK
                        REPLACE elemento WITH ALLTRIM(STR(tol2,7,2)), posicion WITH 15, tipo WITH "T"
                ENDIF
        ENDIF
ENDIF
IF O6<>7
        APPEND BLANK
        REPLACE elemento WITH conector(O6), posicion WITH 16, tipo WITH tipo(O6)
        APPEND BLANK
```

```
		REPLACE elemento WITH campos(o7), posicion WITH 17, tipo WITH "C"
		APPEND BLANK
		DO CASE
			CASE O8>=13 AND O8<=18
				O8=3
			CASE O8>=19 AND O8<=25
				O8=4
			CASE O8>=26 AND O8<=29
				O8=1
			CASE O8>=30 AND O8<=36
				O8=2
			CASE O8>=37 AND O8<=38
				O8=5
		ENDCASE
		REPLACE elemento WITH operador(o8), posicion WITH 18, tipo WITH tipop(o8)
		IF O8>5 AND O8<13
			IF mtipoc3="N"
				APPEND BLANK
				REPLACE elemento WITH ALLTRIM(STR(val)), posicion WITH 19, tipo WITH "V"
			ELSE
				APPEND BLANK
				REPLACE elemento WITH """+ALLTRIM(val)+""", posicion WITH 19, tipo WITH "V"
			ENDIF
		ELSE
			IF min3<>0
				APPEND BLANK
				REPLACE elemento WITH ALLTRIM(STR(min3,7,2)), posicion WITH 20, tipo WITH "I"
			ENDIF
			IF max3<>0
				APPEND BLANK
				REPLACE elemento WITH ALLTRIM(STR(max3,7,2)), posicion WITH 21, tipo WITH "M"
			ENDIF
			IF tup3<>0
				APPEND BLANK
				REPLACE elemento WITH ALLTRIM(STR(tup3)), posicion WITH 22, tipo WITH "N"
			ENDIF
			IF tol3<>0
				APPEND BLANK
				REPLACE elemento WITH ALLTRIM(STR(tol3,7,2)), posicion WITH 23, tipo WITH "T"
			ENDIF
		ENDIF
ENDIF
IF O9<>7
		APPEND BLANK
		REPLACE elemento WITH conector(O9), posicion WITH 24, tipo WITH tipo(O9)
		APPEND BLANK
		REPLACE elemento WITH campos(oA), posicion WITH 25, tipo WITH "C"
		APPEND BLANK
		DO CASE
			CASE OB>=13 AND OB<=18
				OB=3
			CASE OB>=19 AND OB<=25
				OB=4
			CASE OB>=26 AND OB<=29
				OB=1
			CASE OB>=30 AND OB<=36
				OB=2
			CASE OB>=37 AND OB<=38
				OB=5
		ENDCASE
		REPLACE elemento WITH operador(oB), posicion WITH 26, tipo WITH tipop(oB)
		IF OB>5 AND OB<13
			IF mtipoc4="N"
				APPEND BLANK
				REPLACE elemento WITH ALLTRIM(STR(val)), posicion WITH 27, tipo WITH "V"
			ELSE
				APPEND BLANK
				REPLACE elemento WITH """+ALLTRIM(val)+""", posicion WITH 27, tipo WITH "V"
			ENDIF
		ELSE
			IF min4<>0
				APPEND BLANK
				REPLACE elemento WITH ALLTRIM(STR(min4,7,2)), posicion WITH 28, tipo WITH "I"
			ENDIF
			IF max4<>0
				APPEND BLANK
				REPLACE elemento WITH ALLTRIM(STR(max4,7,2)), posicion WITH 29, tipo WITH "M"
			ENDIF
			IF tup4<>0
				APPEND BLANK
```

```
                                        REPLACE elemento WITH ALLTRIM(STR(tup4)),  posicion WITH 30, tipo WITH "N"
                                ENDIF
                                IF tol4<>0
                                        APPEND BLANK
                                        REPLACE elemento WITH ALLTRIM(STR(tol4,7,2)),  posicion WITH 31, tipo WITH "T"
                                ENDIF
                        ENDIF
                ENDIF
                CLOSE ALL
                RELEASE WINDOW ingsql
                DO interpr
                RETURN


*************************
PROCEDURE valida1
@ 6.5,31 clear to 8,89
mtipoc1=TYPE(FIELD(O1))
IF O2>5 AND O2<13
                IF mtipoc1="N"
                        val=0
                        @ 6.7,32 GET val
                ELSE
                        val=SPACE(15)
                        @ 6.7,32 GET val
                ENDIF
ELSE
                @ 6.7,42 GET min DEFA 0.00
                @ 6.7,56 GET max DEFA 0.00
                @ 6.7,70 GET tup DEFA 0
                @ 6.7,81 GET tol DEFA 0.00
ENDIF
READ
SHOW GETS
RETURN

****************************
PROCEDURE valida2
IF O3<>7
                O4=7
                O5=1
                @ 8.5,10 GET O4 PICT '@^' FROM campos SIZE 4,8
                @ 8.5,19 GET O5 PICT '@^' FROM operador SIZE 4,10 VALID valida3()
ENDIF
READ
RETURN


*******************************
PROCEDURE valida3
@ 8.5,31 clear to 10,89
mtipoc2=TYPE(FIELD(O4))
IF O5>5 AND O5<13
                IF mtipoc2="N"
                        val2=0
                        @ 8.7,32 GET val2
                ELSE
                        val2=SPACE(15)
                        @ 8.7,32 GET val2
                ENDIF
ELSE
                @ 8.7,42 GET min2 DEFA 0.00
                @ 8.7,56 GET max2 DEFA 0.00
                @ 8.7,70 GET tup2 DEFA 0
                @ 8.7,81 GET tol2 DEFA 0.00
ENDIF
READ
SHOW GETS
RETURN

****************************
PROCEDURE valida4
IF O6<>7
                O7=7
                O8=1
                @ 10.5,10 GET O7 PICT '@^' FROM campos SIZE 4,8
                @ 10.5,19 GET O8 PICT '@^' FROM operador SIZE 4,10 VALID valida5()
ENDIF
READ
SHOW GETS
```

```
RETURN

*****************************
PROCEDURE valida5
@ 10.5,31 clear to 12,89
mtipoc3=TYPE(FIELD(O7))
IF O8>5 AND O8<13
        IF mtipoc3="N"
                val3=0
                @ 10.9,32 GET val3
        ELSE
                val3=SPACE(15)
                @ 10.9,32 GET val3
        ENDIF
ELSE
        @ 10.9,42 GET min3 DEFA 0.00
        @ 10.9,56 GET max3 DEFA 0.00
        @ 10.9,70 GET tup3 DEFA 0
        @ 10.9,81 GET tol3 DEFA 0.00
ENDIF
READ
SHOW GETS
RETURN

****************************
PROCEDURE valida6
IF O9<>7
        OA=7
        OB=1
        @ 12.5,10 GET OA PICT '@^' FROM campos SIZE 4,8
        @ 12.5,19 GET OB PICT '@^' FROM operador SIZE 4,10 VALID valida7()
ENDIF
READ
RETURN

*****************************
PROCEDURE valida7
@ 13.5,31 clear to 15,89
mtipoc4=TYPE(FIELD(OA))
IF OB>5 AND OB<13
        IF mtipoc4="N"
                val4=0
                @ 13.7,32 GET val4
        ELSE
                val4=SPACE(15)
                @ 13.7,32 GET val4
        ENDIF
ELSE
        @ 13.7,42 GET min4 DEFA 0.00
        @ 13.7,56 GET max4 DEFA 0.00
        @ 13.7,70 GET tup4 DEFA 0
        @ 13.7,81 GET tol4 DEFA 0.00
ENDIF
READ
SHOW GETS
RETURN
```

Código de Auxilia.prg

```
DEFINE POPUP auxilia MARGIN FROM 1.2,45
DEFINE BAR 1 OF auxilia PROMPT '\<Examinar, Adicionar y Marcar para Borrado' MESSAGE 'Permite examinar las bases existentes, adicionar registros y marcar registros para borrado'
DEFINE BAR 2 OF auxilia PROMPT 'Em\<paquetar tablas' MESSAGE 'Permite la eliminación definitiva de los registros marcados para borrado'
DEFINE BAR 3 OF auxilia PROMPT '\<Crear tablas' MESSAGE 'Permite la creación de tablas con estructuras definidas para usar en el programa'
DEFINE BAR 4 OF auxilia PROMPT '\<Modificar estructura' MESSAGE 'Permite la modificación de las estructuras de las tablas existentes'
DEFINE BAR 5 OF auxilia PROMPT '\<Restauración de Indices' MESSAGE 'Permite restaurar el índice que permite la visualización ordenada de las tablas'
```

```
ON SELECTION BAR 1 OF auxilia DO examina
ON SELECTION BAR 2 OF auxilia DO empaque
ON SELECTION BAR 3 OF auxilia DO creacio
ON SELECTION BAR 4 OF auxilia DO modific
ON SELECTION BAR 5 OF auxilia DO restaur


ACTIVATE POPUP auxilia
DEACTIVATE POPUP auxilia

*********************************
PROCEDURE examina
HIDE POPUP ALL
marchivo=GETFILE("DBF", "Seleccione una base", "Examinar", 1)
mcodig1=ALLTRIM(marchivo)
SELE 0
USE (marchivo)
BROW
CLOSE ALL
RETURN


*********************************
PROCEDURE empaque
HIDE POPUP ALL
marchivo=GETFILE("DBF", "Seleccione una base", "Examinar", 1)
mcodig1=ALLTRIM(marchivo)
SELE 0
USE (marchivo) EXCL
PACK
CLOSE ALL
RETURN


*********************************
PROCEDURE creacio
HIDE POPUP ALL
CREATE
CLOSE ALL
RETURN


*********************************
PROCEDURE modific
HIDE POPUP ALL
marchivo=GETFILE("DBF", "Seleccione una base", "Examinar", 1)
mcodig1=ALLTRIM(marchivo)
SELE 0
USE (marchivo) EXCL
MODI STRU
CLOSE ALL
RETURN


*********************************
PROCEDURE restaur
HIDE POPUP ALL
marchivo=GETFILE("DBF", "Seleccione una base", "Examinar", 1)
mcodig1=ALLTRIM(marchivo)
SELE 0
USE (marchivo) EXCL
INDEX ON pertotal TAG indicar
CLOSE ALL
RETURN
```

BIBLIOGRAFIA Y REFERENCIAS:

1. Conceptos de Lógica Borrosa.
http://cabezon.gsi.dit.upm.es/~anto/tesis/html/fuzzylog.html. Se desconocen fechas de creación y actualización. Último acceso Septiembre 12 de 2000. Esta página ya no se encuentra en la red, pero conservo copia en archivo de la misma.

2. D. Brubaker. Fuzzy Operators. EDN Access, Noviembre 9 de 1995.
http://www.ednmag.com/reg/1995/110995/23column.htm. Último acceso Enero 7 de 2002.

3. K. Teska. FUZZY LOGIC QUERY PROCESSING.
http://falcon.cc.ukans.edu/~teska/FUZZY.html. Se desconocen fechas de creación y actualización. Último acceso Enero 7 de 2002.

4. J. F. Brule. Fuzzy Systems - a tutorial. http://www.austinlinks.com/Fuzzy/tutorial.html. Fecha de creación 1.985. No se conocen fechas de actualización. Último acceso: Enero 7 de 2002.

5. D. Rasmussen, R. Yager. "SummarySQL - A Fuzzy Tool for Data Mining". Intelligent Data Analysis, Elsevier, 1.987.

6. J. Galindo. FSQL (Fuzzy SQL): A fuzzy query language. http://www.lcc.uma.es/~ppgg/FSQL.html . Último acceso: Enero 7 de 2002. Se desconocen fecha de creación y última fecha de actualización.

7. E. Cox. The Fuzzy Systems Handbook. Primera edición, AP Professional, Massachusetts, 1994. 623 p.

8. D. Chiang, L. R. Chow, N. Hsien. FUZZY INFORMATION IN EXTENDED FUZZY RELATIONAL DATABASES. Fuzzy Sets and Systems 92 (1997) 1-20

9. O. Pons, J. M. Medina, M. A. Vila. INFERENCE FROM A FUZZY DATABASE USING FUZZY RULES. Diciembre de 1.995. Publicado en la página del grupo de investigación ARAI. http://decsai.ugr.es/difuso/tre.html. Último acceso Enero 7 de 2002.

10. J. M. Medina, M. A. Vila, J. C. Cubero, O. Pons. FUZZY KNOWLEDGE REPRESENTATION IN RELATIONAL DATABASES. Noviembre de 1.994. Publicado en la página del grupo de investigación ARAI. http://decsai.ugr.es/difuso/tre.html. Último acceso Enero 7 de 2002.

11. J. M. Medina, O. Pons, M. A. Vila. "GEFRED. A Generalized Model of Fuzzy Relational Databases". Information Sciences, 76, 1-2, pp. 87 - 109. 1994.

12. L. Roux, J. Desachy. "Multisources Information - Fusion Application for Satellite Image Classification". Paper incluido en el libro "Fuzzy Information Engineering - A guided tour of Applications" de D. Dubois, H. Prade, R. Yager. Primera Edición, Wiley Computer Publishing, New York, 1.997. pp. 111-121.

13. P. Bosc, O. Pivert. SQLF: A RELATIONAL DATABASE LANGUAGE FOR FUZZY QUERYING. IEEE TFS, VOL 3, NO. 1 (1995) 1-17.

14. P. Bosc, O. Pivert. SQLF Query Functionality on Top of a Regular Relational Database Managemente System. 6th International Conference on Information Processing and Management of Uncertainty in Knowledge - Based Systems (IPMU'96), pp. 1-6, 1996.

15. E. Damiani, M. G. Fugini, E. Fusaschi. A DESCRIPTOR - BASED APROACH TO OO CODE REUSE. IEEE Computer, Vol. 30, No. 10, Octubre de 1.997

16. J. Hale, S. Shenoi. Analyzing FD Inference in Relational Databases. CORA. Artículo remitido a Data and Knowledge Engineering Journal.

17. D. H. Kraft, F. E. Petry: Fuzzy Information Systems: managing uncertainty in databases and information retrieval systems. Fuzzy Sets and Systems 90, 1997, 183-191.

18. S. Kumar, R. Biswas, A. R. Roy. On extended fuzzy relational database model with proximity relations. Fuzzy sets and Systems 117, 2000, 195-201.

19. P. Wang. SmartRanker 1.0. http://www.cogsci.indiana.edu/farg/peiwang/SmartRanker/ Ultima actualización Marzo 26 de 1998. Último acceso Enero 7 de 2002.

20. UNDERSTANDING QUERY EXPRESSIONS. Oracle8(TM) ConText(R) Cartridge Application Developer's Guide, Release 2.0. http://technet.oracle.com/doc/context206/A54630_01/ch03.htm . Oracle Corporation, 1.997. No se conocen fechas de actualización. Último acceso Mayo 29 de 2000. La página ya no existe pero conservo copia en archivo.

21. SONALYSTS RELEASES OF FUZZY QUERY VERSION 1.1. http://www.internetwire.com/technews/tn/tn981665.htx . Enero 19 de 1999. No se conocen fechas de actualización. Último acceso: Mayo 29 de 2000. La página ya no existe pero conservo copia en archivo.

22. FUZZY QUERY. http://www.sonalysts.com/fq.html . Sonalysts Inc., 2000. No se conocen fechas de actualización. Ultimo acceso: Mayo 29 de 2000. La página ya no existe pero conservo copia en archivo.

23. J. M. Medina. Bases de Datos Relacionales Difusas: Modelo Teórico y aspectos de su implementación. Universidad de Granada, 1994.

24. O. Pons. Representación Lógica de Bases de Datos Difusas. Fundamentos teóricos e implementación. Universidad de Granada, 1994.

25. D. Dubois, H. Prade. Possibility Theory: An approach to Computerized Processing of Uncertainty. Primera Edición, Plenum Press, Nueva York, 1988. 263 p.

26. R. Padrón. "Conjuntos difusos y su aplicación al control automático". Control, Cibernética y Automatización No. 3, Año XXII, 1988. p. 3-4.

27. D. Dubois, H. Prade, P. Smets. "Partial truth is not Uncertainty" IEEE Expert, Agosto de 1994. p. 16.

28. R. A. Orchard. "FuzzyCLIPS Version 6.04A User's Guide". Integrated Reasoning, Institute for Information Technology, National Research Council Canada, Octubre de 1998. p. 9.

29. T. Terano, K. Asai, M. Sugeno. "Fuzzy Systems Theory". Primera Edición, Academic Press Limited, San Diego, 1992. 268 p.

30. W. Pedrycz. "Fuzzy Control and Fuzzy Systems". Segunda Edición, Research Studies Press Limited, Somerset, 1993. 350 p.

31. Java Hispano. www.javahispano.com/download/bd/tema1.pdf . Documentación en bases de datos. Se desconocen fechas de creación y actualización. Último acceso: Diciembre 29 de 2001.

32. H. X. Li, V. C. Yen. FUZZY SETS AND FUZZY DECISION - MAKING. Nueva York, CRC Press, 1995. Pp. 93 - 121

33. "Lecture Notes in Computer Science: Database Systems of the 90s". Edición única, Berlín, Springer-Verlag, 1990. Pp. 217 a 243.

34. S. Alagic. "Type-Checking OQL Queries in the ODMG Type Systems". ACM Transactions on Database Systems, Vol. 24, No. 3, Septiembre de 1.999. Pp. 319 - 360.

35. K. Subieta. "Mapping Heterogeneous Ontologies Through Object Views". http://www.ipipan.waw.pl/~subieta/papers/EfisMapHeterOnt.ppt . Creación Junio de 2000. Se desconoce fecha de última actualización. Último acceso Enero 31 de 2002.

36. M. Holsheimer, A. Siebes. "Data Mining; The search for knowledge in databases". Amsterdam, NEC Research laboratory, 1994. 78 p.

37. H. Hyötyniemi, H. Koivo. Multimedia aplications in industrial automation. Helsinki, Control Engineering laboratory, 1997. 18 p.

38. G. de Cooman. Possibility Theory I: The measure and integral-theoretic groundwork. Zwijnaarde (Bélgica, Universidad Gent, 1996. 34 p.