Publicaciones en Ciencias y Tecnología. Vol.12, N°2, Julio-Diciembre (2018) 83–95

Artículo de investigación

Gestión de Datos Difusos: atributos Tipo 2 y Tipo 3 en bases de datos relacionales

Fuzzy Data Management: Type 2 and Type 3 attributes in relational databases

Soraya Carrasquel[a], David Coronado[a], Rosseline Rodríguez[a], Leonid Tineo[a]

[a] *Universidad Simón Bolívar, Caracas, Venezuela.*

Recibido: 15-03-2018 Aceptado: 12-10-2018

Resumen

En el marco del Proyecto *Desafíos del Modelo Relacional Difuso* cuyo objetivo principal es dar solución a problemas abiertos al Modelo Relacional Difuso vinculados a relaciones de similitud, distribuciones de posibilidad y operaciones con atributos difusos, se presentan, entre otros, resultados sobre el tratamiento difuso de datos Tipo 2 y Tipo 3 y un prototipo de un Sistema Gestor de Bases de Datos (SGBD) extendido, *fuzzy*doDB extensión de PostgreSQL, que soporta consultas con ordenamiento y agrupamiento sobre estos tipos de datos. Además, se muestra un portal web llamado *fuzzy*doDB, cuya finalidad es dar a conocer los resultados de los proyectos del grupo de investigación.

Palabras clave: modelo relacional difuso, SGBD, *fuzzy*doDB, datos difusos, consulta difusa.
Código UNESCO: 120304- Inteligencia Artificial.

Abstract

The Project *Challenges of the Fuzzy Relational Model*, aim to solve open problems about the treatment of fuzzy data in relational databases. In the present work several results of this project are reported. Some concern with Type 2 fuzzy attributes, which are based on probability distributions. Others are on Type 3 fuzzy data, based on similarity relationships. One of the most emblematic results is a prototype of Database Management System for fuzzy data management. This prototype is called *fuzzy*doDB and consists of an extension of PostgreSQL, wich supports queries with ordering and grouping on Type 2 and Type 3 fuzzy data. Additionally, a web portal also called *fuzzy*doDB is shown, whose purpose is to publicize the results of this project.

Key words: fuzzy relational model, DBMS, *fuzzy*doDB, fuzzy data, fuzzy query.
UNESCO Code: 120304- Artificial intelligence.

1. Introducción

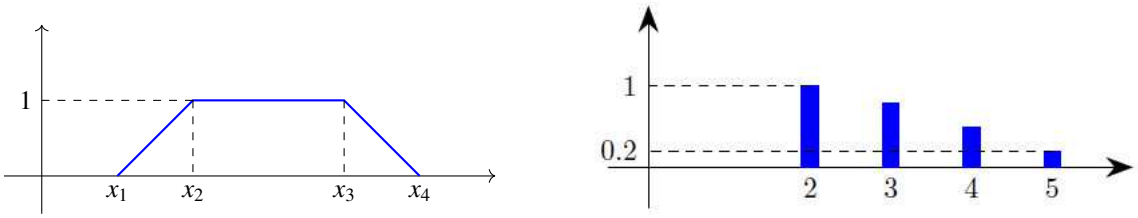
Las bases de datos tradicionales sólo manejan datos y consultas precisas y no permiten manipular información imprecisa o vaga. La teoría de Conjuntos Difusos propuesta por Zadeh [1] permite representar la imprecisión y la incertidumbre de

Impresa: ISSN 1856-8890. Dep. Legal pp200702LA2730. Digital: ISSN 2477-9660 Dep. Legal ppi201402LA4590. Licencia CC BY-NC-SA
Email addresses: scarrasquel@usb.ve (Soraya Carrasquel), dcoronado@usb.ve (David Coronado), crodrig@usb.ve (Rosseline Rodríguez), leonid@usb.ve (Leonid Tineo)

manera que se pueden extender las bases de datos para que no sólo manejen datos y consultas precisos sino que también permitan datos y consultas difusas.

Los conjuntos difusos se caracterizan por una función de membresía μ cuyo rango está en el intervalo real $[0, 1]$. Cuando el grado de membresía de un elemento es cercano a 1, se dice que está más posiblemente (o certeramente) incluido en el conjunto. Así 0 es la medida de completa exclusión y 1 la de completa inclusión. En bases de datos, este concepto permite dar semántica a criterios vagos (o condiciones difusas) que expresan preferencias del usuario y/o particularidades del contexto de los datos o dominio de aplicación. Otra aplicación en bases de datos es la representación de datos imperfectos (difusos).

En caso que el conjunto difuso sea definido sobre un universo numérico ordenado, la representación más sencilla y usual de la función de membresía es la forma trapezoidal, como en la Figura 1(a), la cual se especifica simplemente con una cuádrupla (x_1, x_2, x_3, x_4) de elementos ordenados del dominio $(x_1 \leq x_2 \leq x_3 \leq x_4)$ que definen los vértices del trapecio $\{(x_1, 0), (x_2, 1), (x_3, 1), (x_4, 0)\}$. Un conjunto difuso también se puede representar por extensión, esto es, dando explícitamente el grado de pertenencia de cada valor al conjunto difuso, por ejemplo, $A = \{1/2, 0{,}8/3, 0{,}6/4, 0{,}2/5\}$ cuya representación gráfica se muestra en la Figura 1(b).



(a) Trapezoidalmente. (b) Por extensión.

Figura 1: Representación de un conjunto difuso

Un conjunto difuso puede ser usado para representar números difusos. En este caso se le colocan las siguientes restricciones a la función de membresía: debe ser normalizada y convexa. Se dice que la función es normalizada cuando al menos un valor del rango es 1.

En [2], Zadeh introduce el concepto de relación de similitud como una extensión de una relación de equivalencia. Según esta definición, una relación de similitud S en un universo X, es un subconjunto difuso de $X \times X$, cuya función de membresía es reflexiva $\mu_S(x, x) = 1, \forall x \in X$, simétrica $(\forall x, y \in X)(\mu_S(x, y) = \mu_S(y, x))$, y transitiva $(\forall x, y, z \in X)(\mu_S(x, z) = \vee_y(\wedge(\mu_S(x, y), \mu_S(y, z)))$. La propiedad de transitividad ha sido objeto de varias modificaciones por diferentes autores, por ser inadecuada para algunos dominios de aplicación, en particular para el área de las bases de datos.

En un marco de incertidumbre, para una variable A, en lugar de tener un valor preciso, pudiera tenerse un conjunto difuso de sus posibles valores. Este conjunto define la distribución de posibilidad de A, denotada por $\pi_A = \{\pi_A(x)/x, x \in X\}$. Donde $\pi_A : X \rightarrow [0, 1]$ para un elemento d (dato) en el universo X, indica la posibilidad que el valor de la variable A sea d. Así $\pi_A(d) = 1$ significa que d es un valor completamente posible para A, mas no significa que el valor de A es d. Sin embargo si $\pi_A(d) = 1$ y $\forall d' \in X,\ d' \neq d,\ \pi_A(d) = 0$ entonces el valor de A es d. La distribución de posibilidad π_A debe ser *normalizada* sobre X, esto es, por lo menos hay un valor completamente posible, $\exists d \in X,\ \pi_A(d) = 1$.

Con base en la teoría de Conjuntos Difusos, surgen las propuestas del modelo relacional difuso [3],[4]. Posteriormente éstas son generalizadas en el modelo extendido para bases de datos relacionales difusas llamado GEFRED (modelo relacional difuso generalizado) [5]. Con base en GEFRED, se concibió una versión del modelo Entidad Relación Extendido para bases de datos difusas, llamado FuzzyEER [6]. Por otra parte, se extiende el lenguaje de consulta estructurado SQL (*Structured Query Language*), para incluir el manejo de información imprecisa. En un primer trabajo [7] se extiende el DDL (*Data Description Language*) con nuevos tipos sobre una extensión del concepto de dominio. Luego se completa la extensión de SQL a fin de manipular datos y consultas difusas en una base de datos relacional, surgiendo el lenguaje FSQL, el cual se especifica con detalle en [6].

Para representar datos difusos, en FuzzyEER se definen cuatro tipos de atributos difusos: **Tipo** 1, atributos con valores de datos precisos provistos de etiquetas lingüísticas, interpretadas como números difusos, con el propósito de ser usadas en condiciones difusas. **Tipo** 2, atributos con valores de datos difusos representados como números difusos, son distribuciones de posibilidad en un dominio ordenado. **Tipo** 3, atributos con valores en un dominio formado por etiquetas provisto de una relación de similitud entre las etiquetas, adicionalmente permite distribuciones de posibilidad

sobre las etiquetas. **Tipo** 4, similar a los atributos del Tipo 3, pero sin las relaciones de similitud.

Aunque los datos difusos Tipo 3, de la clasificación en *Fuzzy*EER, admiten distribuciones de posibilidad, en este trabajo se llamará datos **Tipo** 3 a atributos con valores en un dominio formado por etiquetas provisto de una relación de similitud entre las etiquetas, y datos **Tipo** 5 a distribuciones de posibilidad sobre un conjunto de etiquetas provisto de una relación de similitud.

Las implicaciones de tener bases de datos con atributos difusos basados en relaciones de similitud y en distribuciones de posibilidad no han sido suficientemente exploradas hasta la fecha. Particularmente en lo que concierne a consultas basadas en ordenamiento y particionamiento de datos. Por ello surgió el Proyecto Desafíos del Modelo Relacional Difuso. El objetivo del presente artículo es reportar algunos de los avances del mencionado proyecto.

El resto de este trabajo está estructurado de la siguiente forma: la Sección 2 está dedicada al tratamiento de atributos difusos Tipo 2; la Sección 3 versa sobre el tratamiento de atributos Tipo 3; la Sección 4 presenta algunos aspectos importantes de un sistema gestor de bases de datos con soporte a atributos Tipos 2 y Tipo 3, llamado *fuzzy*doDB; finalmente se muestran las conclusiones y trabajos futuros.

2. Tratamiento de atributos Tipo 2

Los datos Tipo 2 son atributos cuyo valor viene dado mediante un número difuso. El lenguaje FSQL no permite el uso de atributos de este tipo en las cláusulas de ordenamiento (ORDER BY) ni particionamiento (GROUP BY). De modo que en la práctica se ve limitado su uso. Sin embargo, sobre números difusos se han definido muchos ordenamientos, un resumen se puede encontrar en [8][9], donde también se establecen criterios para considerar un ordenamiento *mejor* que otro. Esto permite concebir una extensión de SQL que incorpore ordenamiento y particionamiento basado en atributos Tipo 2.

2.1. Ordenamiento de números difusos

Una de las razones por la que existen diversos ordenamientos para números difusos, según Yuan [10], es que ningún método da resultados satisfactorios para todos los problemas, algunos métodos se adaptan mejor que otros, o en ocasiones, hay que crear un método que se adapte a las expectativas del problema. En otros casos, los ordenamientos no son totales, no pueden comparar a todos los números difusos, o no pueden comparar números difusos con números precisos.

Es así como en [11] se propuso un ordenamiento de números difusos basado en el Principio de Extensión de Zadeh y en [12] se presentaron dos propuestas usando las integrales difusas de Sugeno y Choquet, respectivamente. Estos tres ordenamientos, además de ser totales, permiten realizar comparaciones de números difusos con números precisos.

Dados dos números difusos A y B, los tres ordenamientos se basan en establecer una distribución de posibilidad, para el ordenamiento,

$$\pi_{A<B} = \{\pi_{A<B}(\text{true})/\text{true}, \pi_{A<B}(\text{false})/\text{false}\}$$

y luego establecer el ordenamiento $<_P$ como

$$A <_P B \iff \pi_{A<B}(\text{true}) > \pi_{B<A}(\text{true}).$$

Se define además el comparador $=_P$ como $a =_P b \iff \pi_{a<b}(\text{true}) = \pi_{b<a}(\text{true})$

La diferencia de estos tres ordenamientos es la manera de cómo calcular $\pi_{A<B}$:

1. Usando el Principio de Extensión de Zadeh:

$$\pi_{A<B} = \left\{ \bigvee_{\substack{x_1<x_2 \\ x_1,x_2\in X}} (\mu_A(x_1) \wedge \mu_B(x_2))/\text{true}, \bigvee_{\substack{x_1\geq x_2 \\ x_1,x_2\in X}} (\mu_A(x_1) \wedge \mu_B(x_2))/\text{false} \right\}$$

   Donde $\bigvee$ y $\bigwedge$ son el máximo y el mínimo, respectivamente.

2. Usando la integral difusa de Sugeno se obtiene la misma distribución de posibilidad anterior.

3. Usando la integral difusa de Choquet, se obtiene

$$\pi_{A<B} = \left\{ \sum_{\substack{x_1<x_2 \\ x_1,x_2 \in X}} (\mu_A(x_1) \cdot \mu_B(x_2))\,/\text{true}, \sum_{\substack{x_1 \geq x_2 \\ x_1,x_2 \in X}} (\mu_A(x_1) \cdot \mu_B(x_2))\,/\text{false} \right\}$$

2.2. Extensión al sublenguaje SQL-DDL

En [13], se extiende el SQL-DDL (SQL Data Definition Language) para poder definir dominios de datos Tipo 2. La sintaxis propuesta es: CREATE FUZZY DOMAIN ⟨*name*⟩ AS POSSIBILITY DISTRIBUTION ON ⟨*ordered domain*⟩. Se incluyen además las sentencias:

DROP FUZZY DOMAIN ⟨*name*⟩

CREATE FUZZY CONSTANT ⟨*name*⟩ ⟨*domain name*⟩ := ⟨*value*⟩

DROP FUZZY CONSTANT ⟨*name*⟩ ⟨*domain name*⟩

Las cuales permiten, en orden de aparición, eliminar un dominio difuso, crear una constante difusa y eliminar una constante difusa. En el caso que la distribución de posibilidad es representada en forma trapezoidal con vértices $\langle x_1\rangle, \langle x_2\rangle, \langle x_3\rangle, \langle x_4\rangle$, su sintaxis es: $\{(\langle x_1\rangle, \langle x_2\rangle, \langle x_3\rangle, \langle x_4\rangle)\}$, para representar trapezoides degenerados se permite que $\langle x_1\rangle$ y $\langle x_2\rangle$ (o $\langle x_3\rangle$ y $\langle x_4\rangle$) sean iguales al valor $NULL$ y los valores menores o iguales a $\langle x_3\rangle$ (o mayores o iguales a $\langle x_2\rangle$) tengan grado 1. Cuando la distribución de posibilidad es definida por extensión su sintaxis es $\{\langle m_1\rangle/\langle x_1\rangle, \ldots, \langle m_n\rangle/\langle x_n\rangle\}$, donde $\langle m_i\rangle$ es el valor de la función de membresía para $\langle x_i\rangle$.

2.3. Extensión a la cláusula ORDER BY

En [13], para atributos Tipo 2, se preserva la sintaxis de la cláusula ORDER BY.

ORDER BY $k_1 d_1, \ldots, k_o d_o$, donde los k_i son los atributos que componen la clave de ordenamiento, y los d_i son descriptores de ordenamiento para los respectivos atributos, que pueden ser ASC (ascendente) o $DESC$ (descendente), también se puede omitir un d_i, lo cual equivale a ASC que es la opción por defecto.

La semántica se extiende usando el comparador posiblemente menor ($<_P$) en lugar del menor clásico ($<$) para el $d_i = ASC$, en caso de ser k_i un atributo difuso Tipo 2. Análogamente para el $DESC$.

3. Extensión a la cláusula GROUP BY

El establecimiento de un orden para números difusos permite definir un comparador posiblemente igual $=_P$. Con este comparador se le da semántica al particionamiento basado en atributos Tipo 2.

Se propone que la cláusula GROUP BY preserve la misma sintaxis que en el caso clásico GROUP BY $k_1, k_2, \ldots, k_o$. El cambio en la semántica es que para todo atributo k_i de la clave de agrupamiento que sea un atributo difuso Tipo 2, en lugar de usar la igualdad clásica, al momento de generar la partición, se usa el comparador posiblemente igual $=_P$.

4. Tratamiento de atributos Tipo 3

Los atributos difusos Tipo 3 se basan en el uso de relaciones de similitud entre etiquetas. Al igual que en el caso de los atributos Tipo 2, FSQL no permite que atributos Tipo 3 sean usados en las cláusulas de ordenamiento (ORDER BY) y particionamiento (GROUP BY) de una consulta. De modo que se ha asumido el desafío de superar esta limitación de FSQL.

4.1. Nueva relación de similitud

Como se había mencionado, varios autores han propuesto definiciones alternativas a la propiedad de transitividad original de la relación de similitud propuesta por Zadeh [2], por considerar esta última inadecuada para algunos contextos. En

particular, el área de las bases de datos, donde las respuestas obtenidas no son satisfactorias a lo esperado por el usuario. Un estudio comparativo de estas propiedades se muestra en [14].

En [14], se define una nueva propuesta de relación de similitud con una propiedad transitiva más satisfactoria a las preferencias del usuario. Esta nueva definición permite extender el ordenamiento y agrupamiento en consultas a bases de datos relacionales con dominios de atributos Tipo 3, los cuales son atributos con valores en un dominio formado por etiquetas provisto de una relación de similitud entre las etiquetas. Sea X un conjunto de valores, S un subconjunto difuso de $X \times X$ con función de membresía $\mu_S : X \times X \to [0,1]$.

En [14], Carrasquel, Rodríguez y Tineo definen una *relación de similitud difusa* S como una relación que es Reflexiva: $(\forall x \in X)(\mu_S(x,x) = 1)$, Simétrica: $(\forall x,y \in X)(\mu_S(x,y) = \mu_S(y,x)$ y Transitiva: $((\forall x,z \in X, x \neq z,\ \forall y \in X)(\mu_S(x,y) = 1 \wedge \mu_S(y,z) = \beta \Rightarrow \mu_S(x,z) = \beta) \wedge ((\forall x,z \in X,\ x \neq z,\ \forall y \in X)(\mu_S(x,y) = \beta \wedge \mu_S(y,z) = 1 \Rightarrow \mu_S(x,z) = \beta))$. La relación de similitud S induce una partición difusa P_S sobre el conjunto de valores de X, $P_S = \{S[x]/x \in X\}$ cada elemento de esta partición difusa se llama *clase difusa*, que es el conjunto difuso descrito por la función de membresía $\mu_{S[x]}(y) = \mu_S(x,y)$.

4.2. Extensión al sublenguaje SQL-DDL

Para crear consultas difusas con datos Tipo 3, se extiende el SQL-DDL (SQL *Data Definition Language*) [14], la creación de dominios de datos difusos que tengan asociada una relación de similitud. Se presenta la sintaxis de definición de un dominio difuso de datos:

CREATE FUZZY DOMAIN ⟨*name*⟩ AS VALUES (⟨*label*⟩, [⟨*label*⟩, ..., ⟨*label*⟩])
[SIMILARITY {(⟨*label*⟩, ⟨*label*⟩)/⟨*value*⟩,
[(⟨*label*⟩, ⟨*label*⟩)/⟨*value*⟩, ..., (⟨*label*⟩, ⟨*label*⟩)/ ⟨*value*⟩]}].

Las siguientes cláusulas permiten, en orden de aparición, crear un dominio difuso, agregar datos al dominio difuso, agregar pares a la relación de similitud, eliminar datos del dominio difuso, eliminar pares de la relación de similitud y eliminar un dominio difuso. En [14] se presenta la formalización de estas operaciones y mayor detalle sobre su uso.

ALTER FUZZY DOMAIN ⟨*name*⟩ ADD VALUES (⟨*label*⟩, [⟨*label*⟩, ..., ⟨*label*⟩])

ALTER FUZZY DOMAIN ⟨*name*⟩ ADD SIMILARITY
{(⟨*label*⟩, ⟨*label*⟩)/⟨*value*⟩), [(⟨*label*⟩, ⟨*label* >⟩/⟨*value*⟩), ..., (⟨*label*⟩, ⟨*label*⟩)/⟨*value*⟩)]}

ALTER FUZZY DOMAIN ⟨*name*⟩ DROP VALUES (⟨*label*⟩, [⟨*label*⟩, ..., ⟨*label*⟩])

ALTER FUZZY DOMAIN ⟨*name*⟩ DROP SIMILARITY
{(⟨*label*⟩, ⟨*label*⟩), [(⟨*label*⟩, ⟨*label*⟩), ..., (⟨*label*⟩, ⟨*label*⟩)]}

DROP FUZZY DOMAIN ⟨*name*⟩

4.3. Extensión a la cláusula ORDER BY

El concepto de relación de similitud [14], permite extender la cláusula ORDER BY con una nueva semántica para los atributos con dominio difuso Tipo 3. En esta propuesta [14] se ordenan los elementos similares a un elemento dado de acuerdo al grado de membresía en la relación de similitud. La nueva sintaxis para la cláusula ORDER BY del SELECT sería:

SELECT $c_1, c_2, \ldots, c_n$ FROM T ORDER BY $k_1 \cdot d_1, \ldots, k_o \cdot d_o$

donde v_i es un valor del dominio difuso correspondiente a k_i y $c_1 \ldots c_n$ son atributos de T y T es una expresión SQL que denota una tabla, $k_i \in \{c_1, c_2, \ldots, c_n\}$ y $d_i \in \{ASC, DESC, STARTv\}$.

En [14], se definieron las formas verbosas alternativas para k_i START v_i como: (1) k_i STARTING FROM v_i; (2) SIMILARITY ON k_i START v_i; y (3) SIMILARITY ON k_i STARTING FROM v_i.

El uso de un criterio de ordenamiento de la forma k_i START v_i produce que el conjunto de tuplas resultado sean ordenadas descendentemente por el grado de membresía $\mu_{S[v_i]}$, con $S[v_i]$ la clase difusa de v_i. En [14] se da la definición

formal de esta nueva semántica.

4.4. Extensión a la cláusula GROUP BY

Se modifica la sintaxis de la cláusula GROUP BY permitiendo que cada columna en la lista aparezca sola, como es usual, o aparezca precedida de la nueva palabra clave SIMILAR [15]:

GROUP BY [SIMILAR] ⟨column⟩ ... [SIMILAR] ⟨column⟩

La semántica propuesta en [15] es la siguiente: Sea C una consulta de agrupamiento difuso de la forma

SELECT $c_1, c_2, \ldots, c_n$
FROM T GROUP BY $m_1k_1, \ldots, m_ok_oHC$.

donde T es una expresión SQL que denota una tabla; $k_1, \ldots, k_o$ son atributos de T; cada c_j es tal que $c_j \in \{k_1, \ldots, k_o\}$ o $c_j = agg_j(a_j)$ con agg_j una función de agregación y a_j atributo de T; HC es la cláusula HAVING, $HC = \varepsilon$ (el string vacío) o HC es de la forma HAVING *cond*, siendo *cond* una expresión lógica llamada la condición de grupo; cada m_i =SIMILAR o $m_i = \varepsilon$ (el string vacío), en el caso que m_i =SIMILAR, $dom(k_i)$ debe ser un dominio difuso fd_i dotado de una relación de similitud S_i.

La relación de similitud S_i asociada al dominio difuso de k_i permite establecer una relación de similitud sobre las tuplas de T, la cual induce una partición difusa. El agrupamiento se hace sobre las tuplas que pertenecen a la misma clase y que cumplen con la condición especificada en la cláusula HAVING. En [15] se puede observar la formalización de esta semántica.

5. Implementación de un SGBD extendido

Los resultados mencionados en las secciones anteriores constituyeron el sustento teórico para la creación de un prototipo de un SGBD relacional extendido llamado *fuzzy*doDB [16] (*fuzzy domains database*), base datos con dominios difusos, para la gestión de datos Tipo 2 y Tipo 3, así como consultas con ordenamiento y agrupamiento sobre atributos difusos. El desarrollo del prototipo se ha realizado con la colaboración de varios grupos de estudiantes de Ingeniería de la Computación en la Universidad Simón Bolívar cursando una asignatura electiva denominada Miniproyecto de Desarrollo de Software.

5.1. SGBD extendido para atributos Tipo 3

En [17] se reporta la creación de una librería que implementa consultas difusas que usan atributos Tipo 3, para las cláusulas de Ordenamiento y Agrupamiento. Para ello, se extiende el Sistema Gestor de Bases de Datos (SGBD), MariaDB, usando una arquitectura de acoplamiento medio, la cual consiste en una implementación intermedia donde la lógica de la extensión fue programada en el lenguaje nativo del SGBD. Esta librería se desarrolló como una capa externa programada en Java que implementa los esquemas de traducción [18] entre el lenguaje extendido y el lenguaje nativo del SGBD.

En el catálogo difuso [19], los metadatos correspondientes a dominios Tipo 3, se incluyen en el catálogo de la base de datos mediante esquemas relacionales.

Como se puede observar en la Figura 2, resaltan las claves primarias con subrayado simple, las alternas con subrayado doble y las foráneas con el nombre de la tabla referenciada como un superíndice.

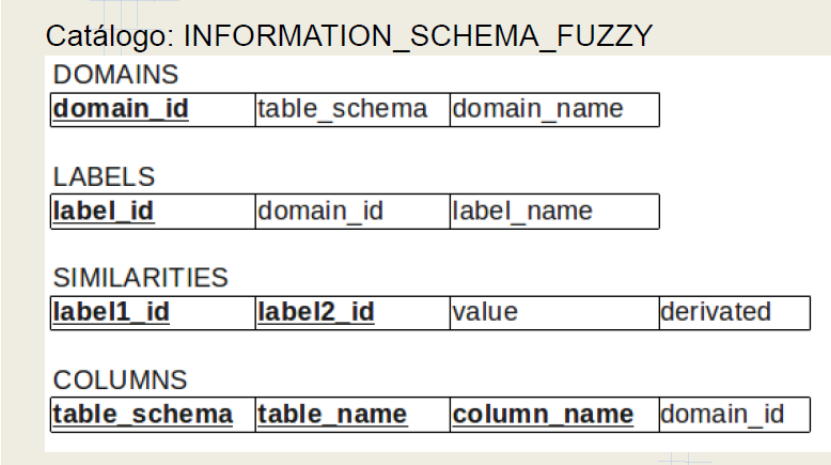


Figura 2: Catálogo extendido.

En DOMAIN(domainId, tableSchema, domainName) se encuentran los diferentes dominios de atributos Tipo 3, domainId es autogenerada para facilitar su manipulación.

LABEL(labelId, domainId DOMAIN , labelName) almacena las etiquetas (labelName) de los dominios difusos, referenciados por domainId y labelId es autogenerada.

SIMILARITY(label1Id LABEL , label2Id LABEL, value, derived) contiene los pares de etiquetas de las relaciones de similitud, con su grado de membresía (value) y derived indica si el par es o no derivado por reflexividad, simetría y transitividad.

COLUMN(tableSchema, tableName, columnName, domainIdDOMAIN) contiene la definición de columnas cuyo tipo es un dominio difuso (domainId) [19].

En [18] se extiende el SGBD MariaDB a fin de incorporar los dominios difusos correspondientes a atributos Tipo 3 y la posibilidad de realizar consultas con ordenamiento y agrupamiento difuso sobre estos datos. Para ello, fue necesario concebir y formalizar esquemas de traducción para atributos Tipo 3, que permiten convertir las instrucciones del SQL extendido a instrucciones de SQL nativo, utilizando el catálogo difuso. La definición se especifica usando lógica de primer orden y notación de conjuntos para representar las tablas y propiedades que éstas cumplen, las mismas son resultados de diferentes operaciones de SQL extendido. En [18] se detallan los esquemas de traducción implementados para cada una de las operaciones, así como ejemplos de su uso.

5.2. Integración del SGBD

Con la finalidad de integrar el manejo de atributos difusos Tipo 2 y Tipo 3 en un mismo SGBD, se migra a PostgreSQL la librería originalmente implementada sobre el SGBD MariaDB que da soporte a atributos Tipos 3, sin afectar la funcionalidad. Se integraron en un solo *parser* las extensiones correspondientes a ambos tipos de datos. De esta forma se llegó a un prototipo integrado, el cual se le dio el nombre de *fuzzy*doDB.

6. SGBD para atributos Tipo 2

El SGBD PostgreSQL permite la creación de tipos abstractos de datos definidos por el usuario (UDT), así como operadores de comparación para estos datos. Tomando ventaja de esta funcionalidad de PostgreSQL, la extensión para el manejo de atributos Tipo 2 se hizo en este SGBD, dado que se requiere de implementar los comparadores basados en el ordenamiento de números difusos. La implementación es con una arquitectura de acoplamiento medio. Una capa en Java se encarga del análisis sintáctico de la cláusulas del SQL extendido.

Se crea una tabla para el manejo de los metadatos correspondientes a la definición de dominios difusos Tipo 2 y otra para las constantes. Se define un UDT en PostgreSQL para dar soporte a valores de atributos difusos Tipo 2. Este UDT

permite dos formas de representación de números difusos: la trapezoidal y la de extensión. Para este tipo abstracto se definen los operadores <, =, y > pero con la semántica correspondiente al $<_P$, $=_P$ y $>_P$. PostgreSQL usará estos operadores en cualquier lugar en que se requiera una comparación de este tipo de datos. Particularmente, al ejecutar una cláusula ORDER BY o una cláusula GROUP BY.

Como se propusieron distintos ordenamientos de números difusos, en la implementación se crea un UDT para cada propuesta de ordenamiento. La representación estructural de estos UDT es la misma, lo que cambia es la definición de los comparadores. Al momento de querer usar una propuesta de ordenamiento específica en una cláusula ORDER BY o GROUP BY, lo que se hace es una conversión (*casting*)al UDT respectivo.

7. Aplicaciones y experimentación

Con los prototipos de SGBD extendidos para atributos difusos Tipo 3 y Tipo 2, se hicieron algunas aplicaciones experimentales y estudios de desempeño.

7.1. Aplicación sobre una guía comercial

En [20] se presenta una aplicación web construida con la librería desarrollada para el manejo de atributos difusos Tipo 3 en MariaDB. Esta aplicación usa datos reales provenientes de una guía comercial de servicios. Para ello, se definieron dominios provistos de relaciones de similitud sobre los atributos ciudad y categoría de las empresas promocionadas en esta guía. La aplicación extiende las búsquedas por estos atributos, permitiendo distintas combinaciones de agrupamiento y ordenamiento difuso. Con este trabajo se muestra la utilidad práctica de este tipo de consultas y atributos difusos.

La aplicación usa las definiciones formales de ordenamiento y agrupamiento difuso para atributos Tipo 3. Como ejemplos de las consultas tenemos: Dada una ciudad obtener la lista de categorías “similares.ª una categoría provista, listar las ciudades “cercanas.ª una ciudad con categoría fija.

A manera de ejemplo, se presenta en la Figura 3 cómo se realiza una consulta con atributos Tipo 3 dentro de esta aplicación. En la columna de la izquierda se observa una empresa seleccionada, en una consulta precisa previa. En la columna de la derecha se observan ocho listas donde el usuario puede escoger el tipo de consulta que desee, siendo trasparente si ésta es difusa o precisa. Las consultas difusas se identifican por las palabras “cercanasz “similares"las cuales indican que para calcular el resultado se está usando la relación de similitud definida para el dominio difuso Tipo 3 “Categoríasz la relación de similitud “Cercanas"definida para el dominio difuso Tipo 3 denominado “Ciudades".

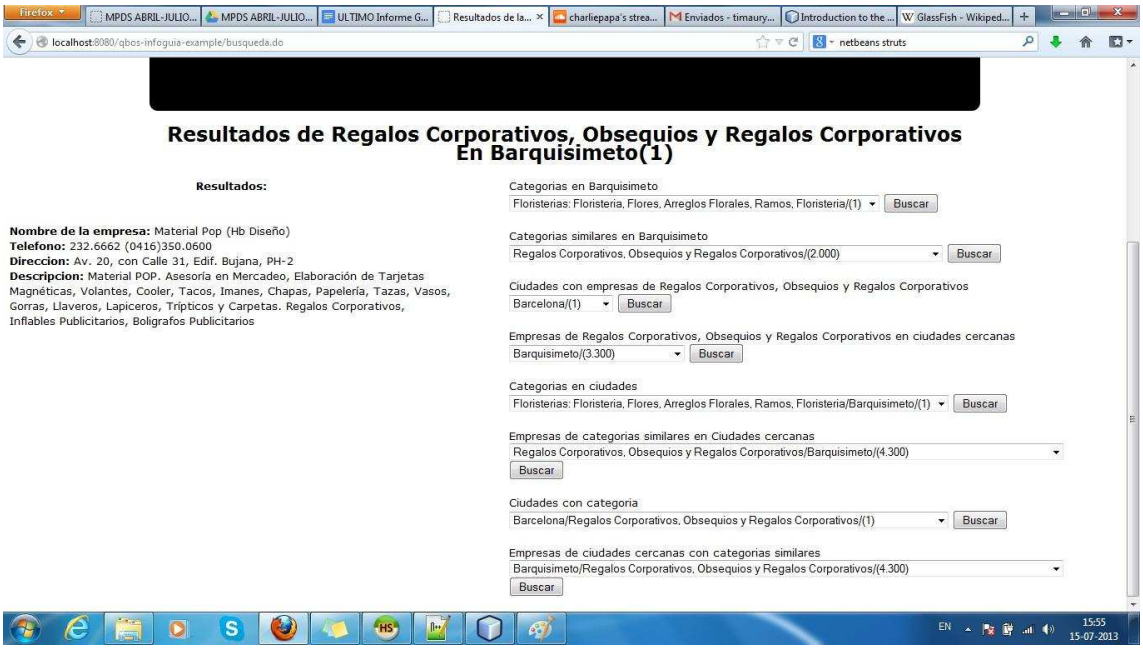


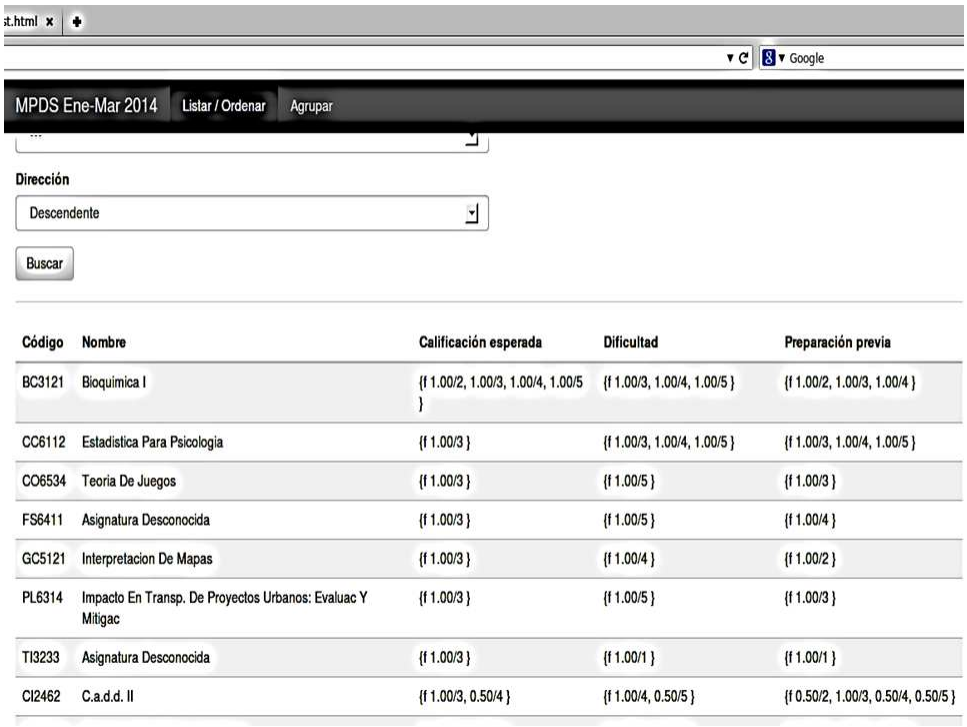
Figura 3: Consulta de atributos Tipo 3 en *fuzzy*doDB

7.2. Aplicación de encuesta de opinión estudiantil

Se desarrolló una aplicación web que admite consulta con datos difusos Tipo 2, tomando la base de datos de la "Encuesta de Opinión Estudiantil"de la Universidad Simón Bolívar.

Se extiende el campo asignatura con tres atributos difusos: Calificación Esperada, Preparación Previa y Dificultad del Curso. Los valores de estos atributos fueron calculados a partir de las respuestas dadas por los estudiantes a las respectivas preguntas. Para cada opción de respuesta posible se calculó su frecuencia midiendo la cantidad de ocurrencias y dividiendo entre el total de respuestas, luego se normaliza este conjunto de valores para obtener una distribución de posibilidad.

Se puede consultar la base de datos para ordenar o agrupar asignaturas de acuerdo a los campos difusos establecidos. Así se pueden hacer consultas del tipo ¿Qué materia(s) tienen la menor calificación esperada?. En la Figura 4 se muestra un fragmento de la interfaz de la aplicación, en la cual se pueden observar los tres atributos mencionados cuyos valores son distribuciones de posibilidad, es decir datos difusos Tipo 2.



Código	Nombre	Calificación esperada	Dificultad	Preparación previa
BC3121	Bioquimica I	{f 1.00/2, 1.00/3, 1.00/4, 1.00/5 }	{f 1.00/3, 1.00/4, 1.00/5 }	{f 1.00/2, 1.00/3, 1.00/4 }
CC6112	Estadistica Para Psicologia	{f 1.00/3 }	{f 1.00/3, 1.00/4, 1.00/5 }	{f 1.00/3, 1.00/4, 1.00/5 }
CO6534	Teoria De Juegos	{f 1.00/3 }	{f 1.00/5 }	{f 1.00/3 }
FS6411	Asignatura Desconocida	{f 1.00/3 }	{f 1.00/5 }	{f 1.00/4 }
GC5121	Interpretacion De Mapas	{f 1.00/3 }	{f 1.00/4 }	{f 1.00/2 }
PL6314	Impacto En Transp. De Proyectos Urbanos: Evaluac Y Mitigac	{f 1.00/3 }	{f 1.00/5 }	{f 1.00/3 }
TI3233	Asignatura Desconocida	{f 1.00/3 }	{f 1.00/1 }	{f 1.00/1 }
CI2462	C.a.d.d. II	{f 1.00/3, 0.50/4 }	{f 1.00/4, 0.50/5 }	{f 0.50/2, 1.00/3, 0.50/4, 0.50/5 }

Figura 4: Resultado de una consulta difusa en la aplicación para la Encuesta de Opinión Estudiantil con atributos Tipo 2 (Calificación esperada, Dificultad y Preparación Previa)

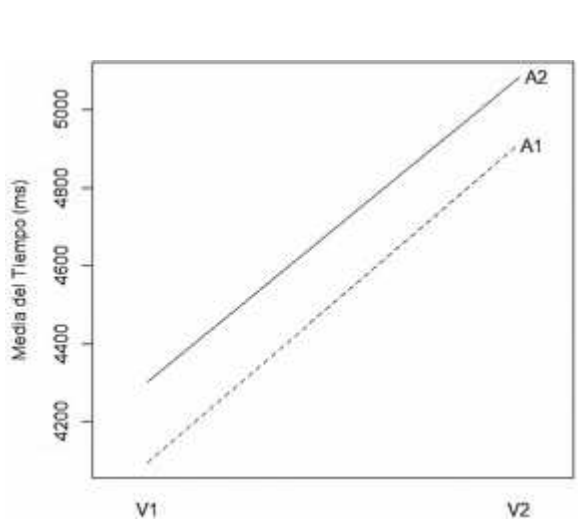
7.3. Análisis de desempeño con World y Pokémon

Con el fin de hacer un estudio del desempeño del prototipo SGBD extendido *fuzzy*doDB, se realiza la extensión difusa de dos bases de datos *World* [21] y Pokémon [19], para soportar el tratamiento difuso de datos Tipo 2 y Tipo 3. La base de datos World [22] es una base de datos experimental disponible en Internet, mientras que la base de datos Pokémon [23] fue creada con la información tomada de sitios web [22, 23] de la popular serie de videojuegos, sobre unos pequeños monstruos de bolsillo.

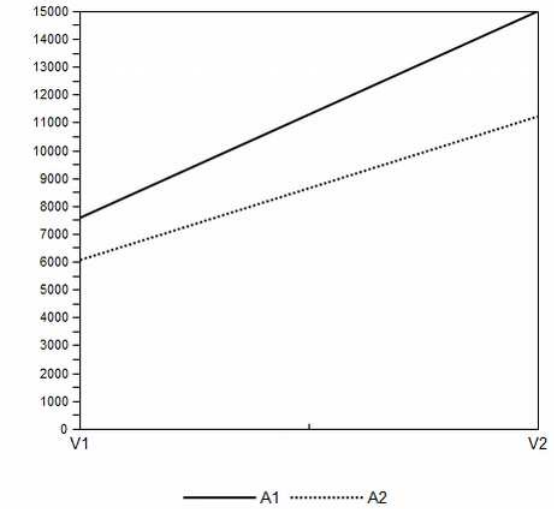
Asímismo, se hizo el análisis de desempeño de las consultas difusas con atributos difusos Tipo 3 realizadas a las bases de datos *World* [21] y Pokémon [19] sobre el SGBD extendido *fuzzy*doDB a fin de observar el impacto de dicha extensión sobre el SGBD original. Para el experimento se utilizó un modelo factorial de 2^{k+1}, donde $k = 2$ es el número de factores utilizados; volumen de datos (alto,bajo) y tipo de consulta (difusa o precisa).

Se usaron quince consultas creadas para cada una de las bases de datos propuestas, usando las sentencias SELECT, GROUP BY y ORDER BY. Las consultas fueron clásicas o con atributos difusos. El experimento se diseñó para medir los tiempos de respuestas de cada tipo de consulta con dos réplicas por cada una de las consultas.

A manera de ejemplo, en la Figura 5(a), se observa el promedio del tiempo de ejecución (ms), con la gráfica de interacción de los factores Volumen-Atributo en consulta ORDER BY que se obtuvo con la base de datos *World*. Mientras que en la Figura 5(b), se observa el promedio del tiempo de ejecución (ms), con la gráfica de interacción de los factores Volumen-Atributo en consulta GROUP BY que se obtuvo con la base de datos Pokémon. El factor A se refiere al atributo involucrado en la cláusula que se está experimentando, el cual puede ser preciso (A_1) o Tipo 3 (A_2). Mayor detalle de los resultados y análisis de estos experimentos se pueden ver en [18] y [20].



(a) Volumen-Atributo en consulta GROUP BY para la base de datos Pokémon.



(b) Volumen-Atributo en consulta ORDER BY con la base de datos *World*.

Figura 5: Tiempo de ejecución de las consultas

7.4. Portal *fuzzy*doDB

Para poner a la disposición el SGBD extendido, así como las experiencias con él, se comienza la creación de un portal web denominado *fuzzy*doDB [16], ver Figura 6, disponible en `www.fuzzydodb.usb.ve`, el cual permitirá a personas interesadas en formar parte de una comunidad de desarrolladores de SGBD extendidos con atributos difusos. Actualmente, el portal no está funcionando porque se encuentra en migración a otro servidor. Los lectores interesados en éste pueden escribir a la dirección de los autores de este trabajo y se les facilitará la información sobre el momento en que el portal sea activado.

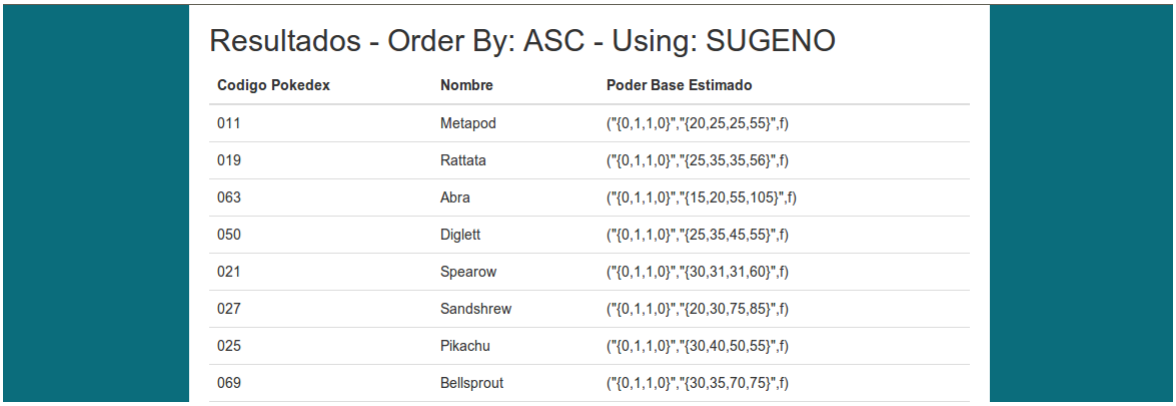
Figura 6: Portal *fuzzy*doDB

También, a través de este portal se podrá acceder a los resultados obtenidos en el proyecto o emplear este prototipo para su uso en investigación y desarrollo. Para la implementación de este portal se usó una metodología ágil con iteraciones y seguimiento semanales.

El portal distingue cinco roles o tipos de usuario: Rol, Visitante, Administrador, Desarrollador y Participante. Como se observa en la Figura 6, las secciones que presenta el portal son siete. Éstas aparecen en una barra horizontal, identificadas como: INICIO y ACERCA, donde se encuentra la información de los miembros del grupo de investigación; DESCARGA, donde los usuarios registrados pueden descargar las bases de datos extendidas; API, donde se encuentran las aplicaciones desarrolladas; DOCUMENTACIÓN, acá los usuarios pueden obtener un abstract y los usuarios registrados pueden acceder a los artículos desarrollados por el grupo de investigación; PRUEBAS EN LÍNEA, donde los usuarios registrados pueden realizar consultas difusas sobre las bases de datos difusas; DESEMPEÑO, donde un usuario registrado puede realizar análisis de desempeño; y COMUNIDAD, donde los usuarios pueden interactuar con los miembros del grupo de investigación y de esta manera ambos actores puedan enriquecer sus investigaciones.

A manera de ejemplo, se presenta en la Figura 7(a), una consulta con ordenamiento difuso disponible en la sección PRUEBAS EN LÍNEA, sobre la base de datos Pokémon. El atributo difuso Tipo 2 es “poder". El usuario puede decidir el sentido del ordenamiento (ASC: ascendente; DESC: descendente) en un menú desplegable. También el método de ordenamiento (Sugeno, Choquet, Centroide o Riemman) que desea usar. En la Figura 7(b), se observa parte del resultado de esta consulta para el sentido ascendente y método Sugeno. Para mayor detalle del uso de este portal, se invita a leer [16].

(a) Consulta con ordenamiento difuso sobre datos Tipo 2 realizada en el portal *fuzzy*doDB sobre la base de datos Pokémon.



Resultados - Order By: ASC - Using: SUGENO

Codigo Pokedex	Nombre	Poder Base Estimado
011	Metapod	("{0,1,1,0}","{20,25,25,55}",f)
019	Rattata	("{0,1,1,0}","{25,35,35,56}",f)
063	Abra	("{0,1,1,0}","{15,20,55,105}",f)
050	Diglett	("{0,1,1,0}","{25,35,45,55}",f)
021	Spearow	("{0,1,1,0}","{30,31,31,60}",f)
027	Sandshrew	("{0,1,1,0}","{20,30,75,85}",f)
025	Pikachu	("{0,1,1,0}","{30,40,50,55}",f)
069	Bellsprout	("{0,1,1,0}","{30,35,70,75}",f)

(b) Resultado parcial de la consulta difusa especificada en la Figura 7(a).

Figura 7: Consulta y Resultado parcial de una consulta difusa

8. Conclusiones y trabajos futuros

En este trabajo se muestran algunos de los aportes principales al Proyecto *Desafíos del Modelo Relacional Difuso* cuyo objetivo principal es resolver problemas abiertos del Modelo Relacional Difuso, vinculados a relaciones de similitud, distribuciones de posibilidad y operaciones con atributos difusos. Particularmente, este trabajo se enfoca en los llamados atributos difusos Tipo 2 y Tipo 3.

Los atributos Tipo 2 permiten datos cuyo valor viene dado por un número difuso. Los aportes aquí reflejados son: la definición de un nuevo ordenamiento para números difusos; la extensión del sublenguaje de definición de datos de SQL para dar soporte a atributos Tipo 2; la extensión de las cláusulas ORDER BY y GROUP BY cuando éstas involucran atributos Tipo 2.

Por otro lado, los atributos Tipo 3, según la definición que aquí se maneja, son etiquetas provistas de una relación difusa de similitud. Los aportes en este sentido son: una nueva definición para relaciones de similitud, donde la transitividad resulta más conveniente; la extensión del sublenguaje de definición de datos de SQL para manejar atributos Tipo 3; la extensión de las cláusulas ORDER BY y GROUP BY basadas en la partición inducida por una relación de similitud para atributos Tipo 3.

Las funcionalidades añadidas a SQL para el tratamiento de atributos Tipo 3 fueron implementadas como una extensión con acoplamiento medio sobre MariaDB. Esto requirió la definición de esquemas de traducción de SQL extendido al SQL clásico con el uso de un catálogo de datos difusos. Las funcionalidades correspondientes al tratamiento de atributos Tipo 2 se implementaron también con una arquitectura de acoplamiento medio, sólo que sobre PostgreSQL como base. Para esto se usó la capacidad de definir operadores de comparación para tipos abstractos definidos por el usuario en PostgreSQL. Ambas extensiones se integraron en una sola, migrando la de MariaDB a PostgreSQL, surgiendo así *fuzzy*doDB.

Haciendo uso de estos prototipos de SGBD difusos, se implementó una aplicación sobre una guía comercial, la cual muestra la utilidad de las funcionalidades propuestas para atributos Tipo 3. También se mostró la utilidad de las extensiones para Tipo 2, mediante una aplicación sobre la encuesta de opinión estudiantil.

Se creó la extensión difusa de dos bases de datos experimentales: Pokémon (creada) y *World* (disponible en Internet). Todos estos resultados se incorporaron en la versión final del prototipo de SGBD difuso, denominada *fuzzy*doDB, el cual permite consultas sobre atributos difusos Tipo 2 y Tipo 3, que permite ordenamiento y agrupamiento. Con estas dos bases de datos experimentales, se hizo un estudio de análisis de desempeño usando el método de experimentación de estadística formal. El estudio trató de mostrar el costo añadido que pudiera tener el proveer las nuevas funcionalidades.

Finalmente, se construyó un portal web que pretende servir como medio para difundir los resultados del proyecto y crear una comunidad de interés en el tratamiento de atributos difusos en bases de datos.

Todos estos resultados reportados aquí ponen en evidencia que sí es posible almacenar atributos difusos Tipo 2 y Tipo 3 en una base de datos relacional, sin tener que restringir su uso en operaciones de consultas, siempre que se le de la semántica adecuada, la cual, si no existe, se puede construir con los aportes matemáticos y/o informáticos que sean requeridos; más aún todo esto no sólo tiene una importancia teórica, sino una aplicabilidad práctica y una utilidad que permitirá finalmente el desarrollo de sistemas más adecuados a la realidad de los datos y usuarios.

Como trabajos futuros se plantean: el tratamiento atributos difusos Tipo 4 y Tipo 5, la extensión difusa de la cláusula WINDOW BY, estudiar otras propuestas de ordenamiento de datos Tipo 2, propuestas de ordenamiento para datos Tipo 4 y Tipo 5, la extensión difusa de alguna base de datos (de las disponibles como referencia de comparación para SGBD relacionales) para ser utilizada en estudios experimentales, así como el uso de consultas más complejas en esos estudios.

Agradecimientos

Este trabajo es resultado del Proyecto de Grupo "*Desafíos del Modelo Relacional Difuso*" que cuenta con el apoyo de la Universidad Nacional Experimental Politécnica .Antonio José de Sucre"(UNEXPO), coordinado por los profesores Carlos Lameda y Leonid Tineo. "*Así será el conocimiento de la sabiduría para tu alma. Si la hallas, habrá un porvenir, y tu esperanza no será frustrada.*" (Proverbios 24:14).

Referencias

[1] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965. OnLine.

[2] L.A. Zadeh. Similarity relations and fuzzy orderings. *Information Sciences*, 3(2):177–200, 1971. OnLine.

[3] S. Fukami; M. Umano; M. Muzimoto; H. Tanaka. Fuzzy database retrieval and manipulation languaje. *IEICE Technical Reports*, 78(223):65–72, 1979.

[4] P. Bosc; O. Pivert. SQLf: a relational database languaje for fuzzy quering. *IEEE Transanctins on Fuzzy Systems*, 3(1):1–17, 1995. OnLine.

[5] J. Medina; O. Pons; M. Vila. GEFRED: A generalized model of fuzzy relational databases. *Information Sciences*, 76(1-2):87–109, 1994. OnLine.

[6] J. Galindo; A. Urrutia; M. Piattini. Fuzzy databases: Modeling, design and implementation. Idea Group Publishing Hershey, USA, 2006.

[7] I. Blanco; N. Marin; O. Pons; M.A. Vila. An extension of data description language (DDL) for fuzzy data handling. In Henrik L. Larsen, Troels Andreasen, Henning Christiansen, Janusz Kacprzyk, and Sławomir Zadrożny, editors, *Flexible Query Answering Systems*, pages 54–64, Heidelberg. Physica-Verlag HD, 2001. OnLine.

[8] G. Bortolan; R. Degani. A review of some methods for ranking fuzzy subsets. *Fuzzy Sets and Systems*, 15(1):1 – 19, 1985. OnLine.

[9] S.H. Chen. Ranking fuzzy numbers with maximizing set and minimizing set. *Fuzzy Sets and Systems*, 17(2):113–129, 1985. OnLine.

[10] Y. Yuan. Criteria for evaluating fuzzy ranking methods. *Fuzzy Sets and Systems*, 43(2):139–157, 1991. OnLine.

[11] D. Coronado; S. Carrasquel; R. Rodríguez; L. Tineo. Ordenamiento de números difusos. In *Memorias de la Segunda Conferencia Nacional de Computación, Informática y Sistemas.* Caracas, Venezuela, pages 205–208, October 2014. OnLine. ISBN: 978 - 980 - 7683 - 00 - 5.

[12] S. Carrasquel; D. Coronado; R. Monascal; R. Rodríguez; L. Tineo. Números difusos y algunos ordenamientos. In *Jornadas de Investigación de Matemáticas Puras y Aplicadas en la Universidad Simón Bolívar JIMPA-USB 2015. Libro de Resúmenes. Caracas*, pages 54–55, June 2015. OnLine.

[13] S. Carrasquel; D. Coronado; R. Rodríguez; L. Tineo. Consultas con ordenamiento de datos posibilísticos. En Revisión en la Revista *Telematique*.

[14] S. Carrasquel; R. Rodríguez; L. Tineo. Consultas con ordenamiento basado en similitud. *Telematique*, 12(1):24–45, 2013. OnLine.

[15] S. Carrasquel; R. Rodríguez; L. Tineo. Queries with grouping based on similarity. *Ingeniare*, 22(4):517–527, 2014. OnLine.

[16] S. Carrasquel; D. Coronado; R. Monascal; R. Rodríguez; L. Tineo. Portal de *fuzzy*doDB. In *Memorias de la Tercera Conferencia Nacional de Computación, Informática y Sistemas, Valencia, Carabobo. 28 al 30 de Octubre de 2015*, pages 328–332, 2015. ISBN: U978-980-7683-01-2UU.

[17] S. Carrasquel; A. Gyomrey; S. Moreau; R. Rodríguez; B. Stornelli; C. Timauiy; L. Tineo. Extensión de MariaDB para ordenamiento y agrupamiento difuso. *Novática*, 229:92–97, 2014.

[18] S. Carrasquel; R. Monascal; R. Rodríguez; L. Tineo. *Processing of Queries with Fuzzy Similarity Domains*. Hershey (USA). IGI Global, 2016. OnLine.

[19] S. Carrasquel; D. Coronado; R. Monascal; R. Rodríguez; L. Tineo. Análisis de desempeño de fuzzydoDB. *Ventana Informática, Manizales, Colombia*, 36:17–34, 2017. OnLine.

[20] S. Carrasquel; R. Rodríguez; L. Tineo. Una aplicación de consultas con ordenamiento y agrupamiento basado en atributos difusos tipo 3. In *III Simposio Científico y Tecnológico en Computación. Universidad Central de Venezuela, Caracas, 14 al 16 de mayo*, pages 135–143, 2014.

[21] S. Carrasquel; D. Coronado; R. Monascal; R. Rodríguez; L. Tineo. Benchmark de consultas con agrupamiento y ordenamiento difuso. *Novática*, 238:41–46, 2016. OnLine.

[22] PostgreSQL Development Group. Fusion forge, 2014. OnLine. Consultado en Octubre 2014.

[23] THE POKEMON COMPANY, 2011. Bellevue (USA): The Pokemon Company. OnLine. Consultado el 20/04/2016.

Sobre los autores

Soraya Carrasquel
Licenciada en Matemáticas, Magister Scientiarum en Matemáticas. Profesora e Investigadora en la Universidad Simón Bolívar, Caracas, Venezuela. Adscrita al Departamento de Computación y Tecnología de la Información.
Correo: scarrasquel@usb.ve - ORCID

David Coronado
Licenciado en Matemáticas, Magister Scientiarum en Matemáticas. PhD. en Matemáticas. Profesor e Investigador en la Universidad Simón Bolívar, Caracas, Venezuela. Adscrito al Departamento de Computación y Tecnología de la Información. Correo: dcoronado@usb.ve - ORCID

Rosseline Rodríguez
Ingeniero en Computación. Magister Scientiarum en Computación. Profesora e investigadora en la Universidad Simón Bolívar, Caracas, Venezuela. Adscrita al Departamento de Computación y Tecnología de la Información.
Correo: crodrig@usb.ve - ORCID

Leonid Tineo
Ingeniero en Computación. Magister Scientiarum en Computación. PhD. en Computación. Profesor e Investigador en la Universidad Simón Bolívar, Caracas, Venezuela. Adscrito al Departamento de Computación y Tecnología de la Información. Correo: leonid@usb.ve - ORCID