

Conjuntos y Sistemas Difusos

(Lógica Difusa y Aplicaciones)

11. FSQL (Fuzzy SQL), un Lenguaje para Consultas Difusas: Definición e Implementación de una BDRD



E.T.S.I. Informática

J. Galindo Gómez

Introducción: Consultas Flexibles

- **Consultas Flexibles o Difusas a Bases de Datos:**
 - Muchos científicos se han planteado la forma de efectuar este tipo de consultas (a través del álgebra y cálculo relacional difuso, por ejemplo).
 - Las **ventajas** más importantes son:
 - **Condiciones Difusas:** Proveer (más) respuestas cuando las consultas clásicas no lo hacen por ser demasiado restrictivas.
 - **Resultado Difuso:** Conseguir, para cada uno de los elementos recuperados, un grado que indique en qué medida cumple las condiciones establecidas.
 - Esto nos permite, por ejemplo, ordenar por orden de importancia los elementos recuperados para elegir sólo los más apropiados.
 - Básicamente, **el problema tiene dos vertientes:**
 - **Consultas Flexibles a Bases de Datos Difusas** (con imprecisión).
 - **Consultas Flexibles a Bases de Datos Clásicas** (sin imprecisión).
 - Un sistema de Consultas flexibles debería considerar esta segunda opción, ya que en la actualidad la inmensa mayoría de BD existentes son Clásicas (*crisp*) y puede ser útil implantar un sistema de consulta difusa, sin necesidad de convertir la BD a Difusa.

Introducción: Consultas Flexibles

- **Consulta Flexible**: Consulta que tiene Condiciones Flexibles.
 - Para aparecer en el resultado **no es necesario cumplir estrictamente las condiciones** impuestas en la consulta.
 - Los elementos del resultado **pueden cumplir dichas condiciones de forma parcial**.
- **Condiciones Flexibles**: Hay muchos aspectos que pueden tenerse en cuenta:
 - **Conceptos Imprecisos** (etiquetas lingüísticas).
 - **Ejemplo**: “Dame los alumnos **JÓVENES**”.
 - Este tipo de condiciones **necesitan un umbral para indicar a partir de que grado de cumplimiento los elementos serán recuperados**.
 - Pueden permitirse **Modificadores Lingüísticos**: “muy”, “poco”, “no muy”, “más o menos”: “Dame los alumnos **MUY Jóvenes**”.
 - **Conectivos Lógicos**, para unir condiciones simples: **NOT, AND, OR...**
 - Suelen usarse los operadores típicos para la **negación** ($1-x$), **conjunción** (t-norma del mínimo) y **disyunción** (s-norma del máximo), pero pueden usarse otros (Yager, 1991; Dubois, Prade, 1995).

3

Introducción: Consultas Flexibles

- **Comparadores Difusos**: Pueden utilizarse distintos comparadores difusos: aproximadamente igual, mayor difuso, mayor o igual difuso, mucho mayor que... (Galindo et al., 1998b, 2000b).
- **Establecer un nivel de importancia en las condiciones simples** (Dubois, Prade, 1986; Sanchez, 1989).
- **Cuantificadores Difusos en las Consultas** (Yager, 1983; Zadeh, 1983; Vila et al., 1995; Galindo et al., 2000a):
 - **Los cuantificadores pueden ser absolutos o relativos**.
 - **Pueden definirse de forma invariante o depender de un argumento**: Por ejemplo \$Mucho_Mayor_que[3].
 - **En síntesis hay dos tipos de consultas**:
 - “ **Q elementos de X cumplen A** ”
 - » **Ejemplo**: “Dame los equipos en los que **CASI TODOS** sus jugadores son Buenos”.
 - “ **Q elementos de X que cumplen B también cumplen A** ”
 - » **Ejemplo**: “Dame los equipos en los que **CASI TODOS** sus jugadores Buenos son también Altos”.
- **Funciones de Grupo o de Agregación** (Dubois, Prade, 1990): Tanto sobre un resultado difuso como sobre atributos difusos.

4

El Lenguaje SQLf de Bosc-Pivert

- **Lenguaje de Consulta Difusa SQLf de Bosc y Pivert (1995):**
 - **Representa una síntesis de otros trabajos anteriores sobre consultas difusas a bases de datos clásicas.** Refs: (Bosc et al., 1988; Li, Liu, 1990; Nakajima et al., 1993; Tahani, 1977; Wong, 1990).
 - **SQLf sólo considera el comando SELECT de SQL.**
 - **Formato:**
SELECT [N | T | N,T] <lista_se_selección>
FROM <lista_de_tablas>
WHERE <condición_difusa>
 - La relación resultante se obtiene de aplicar la **<condición_difusa>** al producto cartesiano de todas las tablas de la **<lista_de_tablas>**.
 - Puede indicarse que **se desean recuperar:**
 - Sólo las **N tuplas** que cumplan la condición con mayor grado (las **N mejores** tuplas).
 - Aquellas en las que su grado de cumplimiento es mayor que el **umbral T** (threshold).
 - Pueden especificarse **N** y **T** a la vez.
 - La **<condición difusa>** sólo permite la comparación de:
 - Columnas *crisp* con etiquetas lingüísticas
 - Valores *crisp* usando el comparador “aproximadamente igual”.
 - Permite el uso de **cuantificadores difusos** de manera similar a como veremos que lo hace **FSQL**.

5

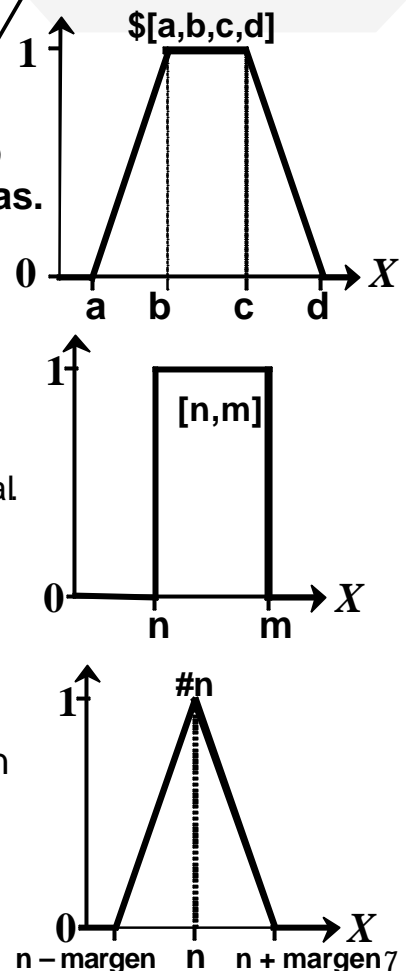
Formato de una BDRD: FIRST

- **FIRST:** Resumen, **FIRST** es la definición del formato interno de la **BDRD**, y su esquema global de implementación.
 - La necesidad inicial de FIRST es debida a que se pretende construir un Sistema de BDRD sobre un SGBD ya existente.
 - En particular, se ha usado el SGBD Oracle®, por su potencia, flexibilidad, popularidad y robustez.
 - **Fundamentos de FIRST**, que se explicarán más adelante:
 - **Atributos Difusos:** Los tipos de atributos difusos permitidos se han sintentizado en tres: Tipo 1 (*crisp*), Tipo 2 (posibilístico) y Tipo 3 (escalares).
 - **FMB (Fuzzy Metaknowledge Base, Base de Metaconocimiento Difuso):** Incluida dentro del catálogo o diccionario del sistema, la FMB almacena información relacionada con la extensión difusa de la base de datos.
 - **Servidor FSQL:** Oracle® sólo permite el uso de SQL y no de **FSQL**. Por eso, ha sido necesario construir un programa que permita el uso de **FSQL** en la BDRD definida por FIRST.

6

Tres Tipos de Atributos Difusos

- **Tipo 1 (*crisp*):** Son atributos clásicos (*crisp*) pero sobre los que podremos efectuar **consultas difusas**.
 - Permiten la utilización en las consultas de:
 - **Comparadores difusos** (14 tipos de comparadores que veremos más adelante).
 - **Constantes Difusas:**
 - **UNKNOWN, UNDEFINED y NULL**, en el sentido de Umano-Fukami.
 - **$\$[a,b,c,d]$** : Distrib. de posibilidad Trapezoidal
 - **$\$label$** : Etiqueta lingüística definida como un trapecio en la FMB.
 - **$[n,m]$** : Intervalo “Entre a y b ” ($a=b=n$ y $c=d=m$).
 - **$\#n$** : Aproximadamente n ($b=c=n$ y $n-a=d-n=margen$ definido en la FMB).
 - **FMB**: Almacena las etiquetas (con su definición asociada a cada una), el **margen** y la distancia mínima M para considerar dos valores como “muy separados” (usada en los comparadores del tipo “mucho mayor” y “mucho menor”).



Tres Tipos de Atributos Difusos

- **Tipo 2 (*posibilísticos*):** Atributos similares a los de Tipo 1, pero que **además** permiten el **almacenamiento** de datos difusos como distribuciones de posibilidad (sobre referencial o dominio subyacente **ordenado**).
 - Permiten **almacenar** en la BD valores de **todos los tipos de constantes** definidas anteriormente para las consultas con Tipo 1.
 - **FMB**: Almacena los mismos valores que los Atributos Difusos Tipo 1.
 - Internamente, un atributo difuso **Tipo 2** son 5 atributos, que toman los siguientes valores para cada tipo de constante difusa:
 - Suponemos que el atributo difuso se llama **F**, el margen para valores aproximados es **mg** y **F_ID** es un identificador que referencia una etiqueta concreta de entre todas las definidas para ese atributo:

Tipo de Valor	FT	F1	F2	F3	F4
UNKNOWN	0	NULL	NULL	NULL	NULL
UNDEFINED	1	NULL	NULL	NULL	NULL
NULL	2	NULL	NULL	NULL	NULL
Crisp C	3	C	NULL	NULL	NULL
Label	4	F_ID	NULL	NULL	NULL
Intervalo $[n,m]$	5	n	NULL	NULL	m
Aprox(n)	6	d	d-mg	d+m	mg
Trapecio $[a,b,c,d]$	7	a	b-a	c-d	d

Tres Tipos de Atributos Difusos

- **Tipo 3 (escalares):** Son atributos sobre referencial **no ordenado**.
 - Su dominio es un **conjunto de escalares** sobre los que no está definida una relación de orden.
 - **Ej.:** Color del pelo = {rubio, moreno, pelirrojo, castaño, canoso}.
 - Sobre los escalares definidos debe definirse una **relación de “proximidad”** m que indique para cada par de valores en qué medida se parecen. **Ej.:** $m(\text{rubio}, \text{pelirrojo})=0.8$, $m(\text{rubio}, \text{moreno})=0...$
 - **Valores especiales:**
 - **SIMPLE:** Una etiqueta con su grado de posibilidad que debe ser 1 para estar normalizado (label es el identificador de la etiqueta en la FMB): {1/label}
 - **DISTRIBUCIÓN de POSIBILIDAD:** Sobre las etiquetas definidas, con un máximo de n parejas posibilidad/etiqueta.
 - **FMB:** Almacena las etiquetas, relación de proximidad m y el valor n .

Tipo de Valor	FT	FP1	F1	FP2	F2	...	FPn	Fn
UNKNOWN	0	NULL	NULL	NULL	NULL	...	NULL	NULL
UNDEFINED	1	NULL	NULL	NULL	NULL	...	NULL	NULL
NULL	2	NULL	NULL	NULL	NULL	...	NULL	NULL
Simple	3	p	label	NULL	NULL	...	NULL	NULL
Distr. Posibilidad	4	p1	label1	p2	label2	...	Pn	labeln

9

Tablas de la FMB

- La **FMB** almacena información sobre los datos difusos en forma relacional.
 - **OBJ#, COL#:** Son dos atributos que almacenan sendos números que identifican una columna concreta dentro de una tabla concreta.
 - Todas las tablas de la FMB tienen esos dos atributos como llave primaria (o como parte de ella).
 - Oracle© identifica cada tabla del sistema con un número **OBJ#**, que se puede obtener consultando la tabla **USER_OBJECTS**.
 - Dentro de una tabla se identifica cada columna con un número **COL#** que puede consultarse en **USER_TAB_COLUMNS**.
- **Tablas de la FMB:** Veamos las 6 más importantes.
 - **FUZZY_COL_LIST:** Descripción de los atributos difusos:
 - **OBJ#,COL#:** Atributo difuso.
 - **F_TYPE:** Tipo de atributo difuso (1, 2 ó 3).
 - **LEN:** Longitud máxima de una distribución de posibilidad en atributos Tipo 3 (máximo 10).
 - **COM:** Comentario (usualmente se pone el nombre del atributo).

10

Tablas de la FMB

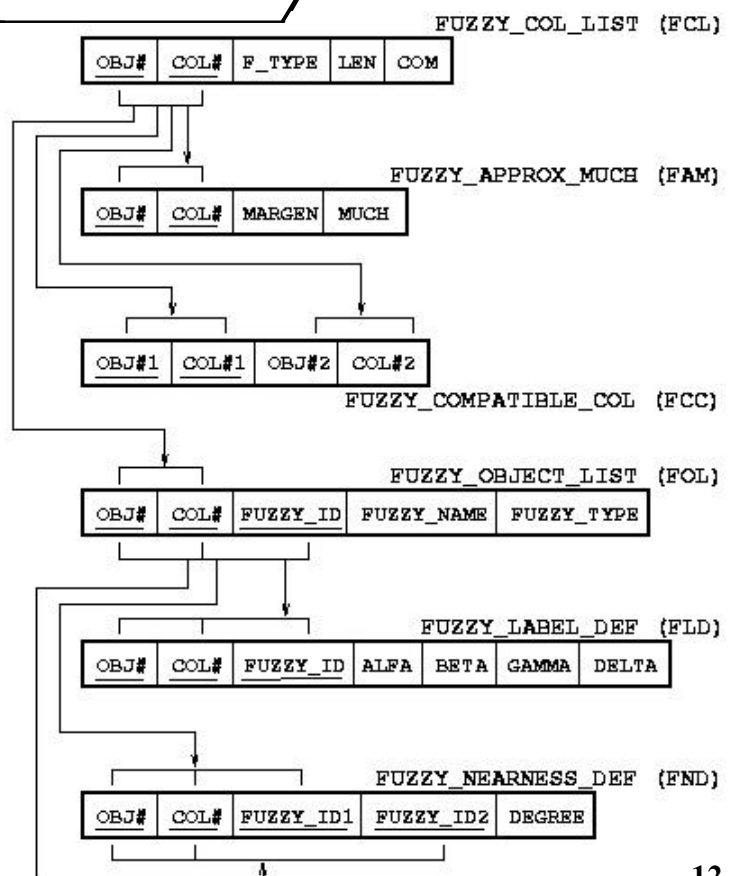
- **FUZZY_OBJECT_LIST:** Descripción de los objetos difusos definidos para cada atributo (OBJ#,COL#):
 - FUZZY_ID: Identificador del objeto difuso (llave externa a FLD y FND).
 - FUZZY_NAME: Nombre del objeto (sin espacios).
 - FUZZY_TYPE: Tipo del objeto: etiquetas trapezoidales (0), etiquetas de tipo 3 (1), cualificadores (2), cuantificadores relativos y absolutos (3,4).
- **FUZZY_LABEL_DEF:** Definición de etiquetas trapezoidales.
 - FUZZY_ID: Identificador del objeto difuso.
 - ALFA, BETA, GAMMA, DELTA: Los cuatro valores del trapecio.
- **FUZZY_APPROX_MUCH:** Valores para el margen (MARGEN) y el valor mínimo o distancia mínima M para considerar dos valores muy separados (MUCH) para atributos difusos Tipo 1 ó 2.
- **FUZZY_NEARNESS_DEF:** Medidas de proximidad, semejanza o similitud (DEGREE) entre cada dos etiquetas de un atributo difuso Tipo 3 (FUZZY_ID1, FUZZY_ID2).
- **FUZZY_COMPATIBLE_COL:** Indica las parejas de atributos difusos Tipo 3 que son compatibles entre sí: Con el mismo dominio.
 - Esto no sólo evita tener que redefinir las etiquetas y la relación de similitud sino que además permite comparar dos atributos difusos compatibles, ya que el sistema sabe que tienen el mismo dominio.

11

Tablas de la FMB: Esquema Gráfico

• Esquema de las Tablas de la FMB:

- **Llaves o Claves:**
 - **Primarias:** Subrayadas.
 - **Externas:** Con flechas.
- **OBJ#:** Identifica un objeto concreto de la BD (como una tabla).
- **COL#:** Identifica un atributo concreto dentro del objeto OBJ#.
- **FUZZY_ID:** Identifica distintos objetos definidos para un atributo particular (como una etiqueta).



12

Resumen de SQL (Structured Query Language)

- **SQL está en continua evolución:** Es una evolución del lenguaje SEQUEL de D.D. Chamberlin y R.F. Boyce (1974) y fue implementado por primera vez por IBM en su BDR llamado SYSTEM R.
 - **ISO** (*International Standards Organization*) y **ANSI** (*American National Standards Institute*) desarrollaron una versión estándar en 1986, llamada SQL-86 o **SQL1**. Posteriormente, se desarrolló **SQL2** o SQL-92. Actualmente se desarrolla **SQL3**, que incluye conceptos de BD orientadas a objetos.
- **SQL es un lenguaje estándar para GESTIÓN de BDR:**
 - **Está incluido en muchos SGBD (DBMS)**, como DB2 (de IBM), Oracle, Ingres, Informix, Sybase, Access, SQL Server...
 - Las mismas sentencias sirven en distintos SGBD.
 - Si se usan sólo las características estándares facilita la tarea de migrar de SGBD.
 - Hay funciones no estándar en algunos SGBD.
 - **Fácil de usar y aprender:** Tiene similitudes con el Álgebra Relacional, aunque se parece más al Cálculo y es más fácil e intuitivo que ambos lenguajes formales.

13

SQL: Resumen del Comando SELECT

- **SELECT** <Select_list>
 - Expresiones a recuperar (atributos, funciones, operaciones...).
- **FROM** <Table_list>
 - Tablas necesarias para la consulta (incluyen tablas de reunión, subconsultas...).
- **[WHERE <condición>]**
 - Condición de selección de tuplas, incluyendo condiciones de reunión.
- **[GROUP BY <atributos_para_agrupar>]**
 - Atributos por los que agrupar el resultado (cada grupo tendrá los mismos valores en estos atributos). Las funciones de grupo o de agregación se aplicarán sobre cada uno de estos grupos. Se aplicarán sobre todas las tuplas si no existe esta cláusula.
- **[HAVING <condición_de_grupo>]**
 - Condición sobre los grupos (no sobre las tuplas).
- **[ORDER BY <atributos_para_ordenar>]**
 - Atributos por los que ordenar. El orden de estos atributos influye.
 - Puede ponerse el nombre de los atributos por los que se quiere ordenar o un número indicando la posición del atributo en la <Select_list>.

14

SQL: Insertar, Borrar y Actualizar

- **INSERTAR: INSERT INTO <tabla> VALUES (a1, a2, ..., an);**
 - **Lista de valores:** En el mismo orden en el que se creó con **CREATE TABLE**
 - **Ejemplo:** INSERT INTO pieza **VALUES** (4,'Teja',50,1000);
 - Insertar el resultado de una **subconsulta**: INSERT INTO <tabla> <subconsulta>;
- **BORRAR: DELETE[FROM] <tabla> [<alias>] [WHERE <condición>];**
 - **Borra** las tuplas que cumplan la condición.
 - **Puede implicar el borrado de otras tuplas en otras tablas**, para que se cumplan la restricción de integridad referencial.
 - **Si se omite la cláusula WHERE** (y su condición), se borran todas las tuplas de la tabla: La tabla quedará vacía (pero sigue existiendo).
 - **Ejemplo:** DELETE Suministrador **WHERE** S# **IN** (2,4,8);
- **ACTUALIZAR: UPDATE <tabla> [<alias>] SET <actualizaciones> [WHERE <cond>];**
 - <actualizaciones> puede ser:
 - 1. Una lista de asignaciones separadas por coma del tipo: <atrib> = <expr>.
 - 2. Una lista de atributos entre paréntesis a los que se le asigna una subconsulta: (<A1>, ..., <Am>) = (<subconsulta>)
 - La cláusula **WHERE** selecciona las tuplas a modificar.
 - **Ejemplo:** UPDATE Pieza **SET** Cantidad=Cantidad+100 **WHERE** P# **IN** (SELECT P# **FROM** Suministros **WHERE** S# **IN** (3,7));

15

DML del LENGUAJE FSQL: SELECT



- **SELECT:** Novedades que incorpora **FSQL** (Galindo et al 1998a).
 - **Etiquetas Lingüísticas:** En las sentencias **FSQL** las etiquetas van precedidas del símbolo \$, para poder distinguirlas fácilmente.
 - **Comparadores Difusos** (Galindo et al., 2000b; Carrasco et al., 2001): Tiene definidos 14 comparadores difusos divididos en dos familias (de Posibilidad y de Necesidad).

Posibilidad	Necesidad	Significado
FEQ	NFEQ	Posiblemente/Necesariamente Igual que
FGT (FGEQ)	NFGT (NFGEQ)	Pos./Nec. Mayor (o igual) que
FLT (FLEQ)	NFLT (NFLEQ)	Pos./Nec. Menor (o igual) que
MGT (MLT)	NMGT (NMLT)	Pos./Nec. Mucho Mayor (Menor) que

- Permiten comparar dos atributos o un atributo con una constante.
- Para atributos difusos Tipo 3 sólo puede usarse **FEQ**.
- Para usar el comparador de “distinto” poner delante de la comparación la negación NOT.
- **Conectivos Lógicos:** Pueden usarse NOT, AND y OR, para enlazar condiciones difusas simples.

16

DML del LENGUAJE FSQ: SELECT

- **Umbral de Cumplimiento** (threshold γ): Tras cada condición simple puede imponerse un umbral de cumplimiento mínimo (por defecto es 1), con el siguiente **formato**: **<condición_simple> THOLD g**
 - La palabra reservada THOLD es opcional y puede sustituirse por un comparador *crisp* (=, <, <=...) modificando el sentido de la consulta. Por defecto es equivalente al comparador >=.
- **Constantes Difusas**: Pueden usarse en el **SELECT** todas las constantes difusas ya definidas: **UNKNOWN**, **UNDEFINED** y **NULL**, **\$[a,b,c,d]** (Distrib. de posibilidad Trapezoidal), **\$label** (Etiquetas), **[n,m]** (Intervalo) y **#n** (valores aproximados).
- **Ejemplos**:
 - “Dame todas las personas cuya edad es aproximadamente 20 años” (con grado mínimo 0.6):
SELECT * FROM Personas
WHERE Edad FEQ #20 THOLD 0.6;
 - “Dame todas las personas más o menos Rubias (con grado mínimo 0.5) cuya edad es posiblemente superior a Joven (con grado mínimo 0.8):
SELECT * FROM Personas
WHERE Pelo FEQ \$Rubio THOLD 0.5
AND Edad FGT \$Joven THOLD 0.8; 17

DML del LENGUAJE FSQ: SELECT

- **Función CDEG(<atributo>)**: Usada en la lista de selección, la función **CDEG** calcula, para cada tupla, el grado de cumplimiento del atributo del argumento en la condición de la cláusula **WHERE**.
- **Función CDEG(*)**: Calcula el grado de cumplimiento de cada tupla en la condición de forma global, para todos sus atributos y no sólo para uno de ellos en particular
 - La función CDEG usa, por defecto, los operadores típicos para la **negación** (1-x), **conjunción** (t-norma del mínimo) y **disyunción** (s-norma del máximo), pero pueden usarse otros (si se definen).
- **Carácter Comodín %**: Similar al carácter comodín * de SQL, pero este incluye además la función CDEG aplicada a todos los atributos de la condición. No incluye CDEG(*).
- **Condición con IS**: También admite condiciones del tipo:
<atributo_difuso> IS [NOT] {UNKNOWN | UNDEFINED | NULL}
- **Comentarios**: FSQ permite incluir 3 tipos de comentarios en sus sentencias:
 - **Hasta fin de línea**: Con un doble guión --
 - **Hasta fin de sentencia**: Con un /* (sin cerrar).
 - **Bloque comentario**: Empieza con un /* y termina con */

DML del LENGUAJE FSQ: SELECT

- **Cuantificadores Difusos:** Tiene dos modalidades que se aplican como condición en la cláusula **HAVING** que sigue a una cláusula **GROUP BY**:

1. “ Q elementos de X cumplen A ”:

\$Cuantificador FUZZY[r] (condición_difusa) THOLD g

2. “ Q elementos de X que cumplen B también cumplen A ”:

\$Cuantificador FUZZY[r]

(condición_difusa1) ARE (condición_difusa2) THOLD g

- **\$Cuantificador:** Es un cuantificador difuso absoluto o relativo que puede ser con o sin argumento, definido sobre alguno de los atributos que aparecen en la cláusula **GROUP BY**.
- **THOLD g :** Umbral de cumplimiento mínimo (**THOLD** es opcional).
- **FUZZY:** Es opcional y si aparece indica que la evaluación del cuantificador se efectuará sumando no los elementos que cumplen la condición, sino el grado de cumplimiento de los mismos.
 - Si aparece el argumento r se indica que se tendrán en cuenta el resultado de ambas sumatorias, con y sin la palabra **FUZZY**, con los valores $r-1$ y r , respectivamente.
- **Ejemplo:** “Equipos que tienen **muchos más de 3** (con grado mínimo 0.5) jugadores Altos” (con grado mínimo 0.75):

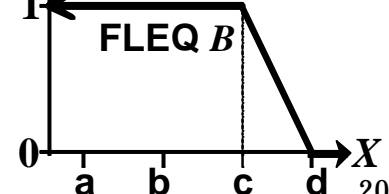
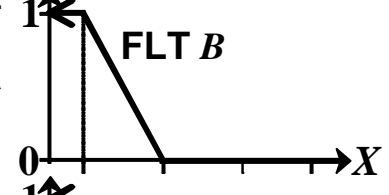
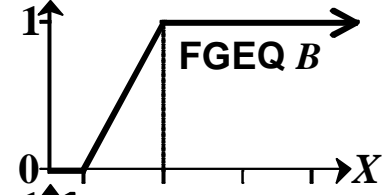
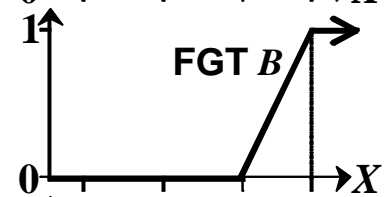
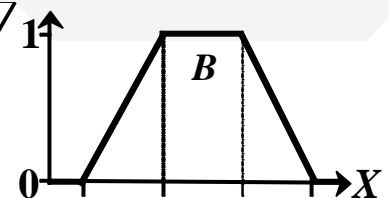
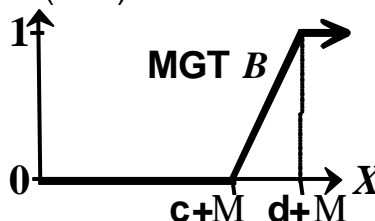
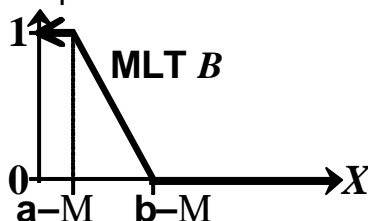
SELECT Equipo, CDEG(*) FROM Personas
GROUP BY Equipo

HAVING \$Muchos_Mas_Que[3] (Altura FEQ \$Alto 0.75) 0.5;

19

Comparadores Difusos (Tipo 1 y 2)

- Sean dos distribuciones de posibilidad A y B que deseamos **comparar**:
 - A **FEQ** B : $\text{Poss}(A, B) = \sup_{x \in X} \{\min(A(x), B(x))\}$
 - A **NFEQ** B : $\text{Nec}(A, B) = \inf_{x \in X} \{\max(1 - A(x), B(x))\}$
 - **Resto de Comparadores de POSIBILIDAD:**
 Se utiliza **FEQ** pero modificando la distribución de posibilidad de la **derecha** (B) según indique el comparador.
 - Si $A=[a,b,c,d]$, entonces se modifica como: ®
 - **Comparadores de NECESIDAD:** El proceso es similar al anterior **pero negando** además la distrib. de posibilidad de la **izquierda** de la comparación (A).
 - **MGT/NMGT (MLT/NMLT):** Se desplaza B hacia la derecha (izquierda) el valor M (de la FMB) y luego se aplica la modificación **FGT** (**FLT**).



20

Comparadores Difusos (Tipo 1 y 2)

- **Restrictividad:** Un comparador C1 es más restrictivo que otro C2 si se cumple que:
 - El resultado de C1 está incluido en C2.
 - Dicho de otra forma: Si el comparador C2 recupera siempre mayor o igual cantidad de tuplas y el grado de cumplimiento de cada tupla es mayor o igual con C2 que con C1.

Familia	De mayor a menor restrictividad
Igual Difuso	NFEQ > FEQ
Mayor Difuso	NFGT > FGT > NFGEQ > FGEQ
Menor Difuso	NFLT > FLT > NFLEQ > FLEQ
Mucho Mayor Difuso	NMGT > MGT
Mucho Menor Difuso	NMLT > MLT

- Los comparadores de Necesidad son más restrictivos que sus correspondientes de Posibilidad.
- Los comparadores del tipo mayor (menor) estricto son más restrictivos que los del tipo mayor o igual que (menor o igual que).

21

Comparadores Difusos (Tipo 1)

- **Comparación de un Atributo Difuso Tipo 1 con un valor *crisp*:**
 - Como valor *crisp* puede ser:
 - Otro atributo difuso Tipo 1.
 - Un atributo no difuso.
 - Una constante no difusa o tipo intervalo.
 - Este tipo de comparaciones puede efectuarse utilizando **comparadores clásicos**.
 - Si se emplean **comparadores difusos**, el Lenguaje FSQL define que el valor *crisp* de la derecha será **difuminado** empleando el valor del **margen (mg)** del atributo difuso Tipo 1 que haya a la izquierda:
 - Si es un valor crisp **n** se convertirá en un valor aproximado **#n**.
 - Si es un intervalo **[n,m]** se convertirá en un trapecio **\$(n-mg,n,m,m+mg)**.
 - **Ejemplo:** Son equivalentes AtribT1 FEQ 6 \equiv AtribT1 FEQ #6
 - Con los **comparadores difusos MGT/NMGT y MLT/NMLT** se difumina también el valor crisp del atributo de la izquierda.
 - Esto consigue una **comparación más gradual**, lo cual parece lógico debido a que estos comparadores son difusos “per se” (no tienen su correspondiente comparador clásico).

22

Comparador FEQ para Tipo 3

- **Comparar dos valores de atributos difusos Tipo 3:** Es necesario utilizar la relación de proximidad m entre cada par de etiquetas:
 - **Comparación de valores SIMPLES, F y X:** El resultado de la comparación es el grado de similitud entre ambas etiquetas $m(F, X)$.
 - **Comparación de DISTRIBUCIONES de POSIBILIDAD, F y X,** siendo $F = \{FP_i / \text{label}F_i\}$ con $i \in [1, \text{LEN}_F]$
 $X = \{XP_j / \text{label}X_j\}$ con $j \in [1, \text{LEN}_X]$
entonces el resultado de la comparación $F \text{ FEQ } X$ viene dado por:
$$\max_{i \in [1, \text{LEN}_F], j \in [1, \text{LEN}_X]} \{m(\text{label}F_i, \text{label}X_j) * FP_i * XP_j\}$$
 - **Observaciones:**
 - Si los valores SIMPLES no están normalizados serán comparados como distribuciones de posibilidad de longitud 1.
 - Si uno de los valores es SIMPLE y otro es una distribución de posibilidad, la obtención del grado de similitud se simplifica considerablemente.

23

Otros Comandos del DML de FSQL

- Aparte del comando **SELECT**, otros comandos fundamentales del **DML** son **INSERT**, **DELETE** y **UPDATE**.
- Su sintaxis es muy similar a la del SQL, modificando:
 - **Expresiones:** En FSQL admiten constantes y atributos difusos.
 - **Subconsultas:** En FSQL pueden ser subconsultas difusas.
 - **Condiciones:** En FSQL las condiciones siempre pueden ser condiciones difusas.
- **INSERT:**
 - Pueden insertarse expresiones difusas o subconsultas difusas.
- **DELETE:**
 - Puede establecerse una condición difusa en la cláusula **WHERE** para borrar aquellas tuplas que la cumplan. Puede ser peligroso: Usar Rollback.
- **UPDATE:**
 - Los nuevos valores pueden ser expresiones difusas o subconsultas difusas.
 - Puede establecerse una condición difusa en la cláusula **WHERE** para actualizar sólo aquellas tuplas que la cumplan.

24

DDL del LENGUAJE FSQ

- Básicamente, los **Comandos del DDL** sirven para:
 - **Crear (CREATE),**
 - **Modificar (ALTER) y**
 - **Borrar (DROP)**las estructuras u **objetos** donde se guardan los datos (a parte de sentencias para control de seguridad, índices y control del almacenamiento físico de los datos).
- En **FSQ** estas sentencias se han modificado para incluir las características de los datos difusos. Así, afectan a 4 objetos:
 - **TABLE:** Se amplía el comando de SQL para expresar las necesidades de una BDRD.
 - **VIEW:** Básicamente, con **FSQ** una vista puede ser también una subconsulta difusa
 - **LABEL:** Objeto exclusivo de **FSQ** que referencia etiquetas para atributos difusos Tipo 1 y 2 que están asociadas a trapecios.
 - **NEARNESS:** Objeto exclusivo de **FSQ** que obliga tanto a la definición de las etiquetas de un atributo difuso Tipo 3 como de la relación de similitud o proximidad existente entre ellas.
- A continuación veremos sólo el comando **CREATE**, por ser el que más novedades incluye y el más importante.

25

DDL del LENGUAJE FSQ: CREATE

- **CREATE TABLE:**
 - **Tipos de datos Difusos:**
 - **FTYPE1 o CRISP (m,M) <tipo_base>:** Para atributos difusos Tipo 1.
 - Entre paréntesis se indican los valores para los atributos MARGEN (**m**) y MUCH (**M**) de la tabla FUZZY_APPROX_MUCH.
 - <tipo_base> es el dominio crisp de este atributo. Por defecto es NUMBER.
 - **FTYPE2 o POSSIBILISTIC (m,M) <tipo_base>:** Para atributos difusos Tipo 2, similar a los de Tipo 1.
 - **FTYPE3 o SCALAR (L) :** Para atributos difusos Tipo 1.
 - Entre paréntesis se indica el máximo número de elementos de sus distribuciones de posibilidad (atributo LEN de FUZZY_COL_LIST).
 - Puede añadirse la palabra DOMAIN seguida de un atributo ya existente: Esto hace que el atributo que se esté definiendo sea compatible con el que ya existe, tomando ambos el mismo dominio (etiquetas y relación μ).
 - **Subconsultas, constantes, condiciones y expresiones:** Donde el comando **CREATE TABLE** de **SQL** permite utilizarlos, **FSQ** permite utilizarlos en su versión difusa. Por ejemplo, la cláusula **DEFAULT** permite poner una constante difusa como valor por defecto.
 - **Restricciones de Columna:** Además de las clásicas, se pueden añadir otras como **NOT UNDEFINED, NOT UNKNOWN, NOT LABEL, NOT TRAPEZOID, NOT INTERVAL, NOT APPROX, CHECK (condición_difusa)...**

26

DDL del LENGUAJE FSQ: CREATE

- **Ejemplo de comando CREATE TABLE:**

- Supongamos que queremos crear una BDRD para una **agencia inmobiliaria** con atributos difusos para los PISOS (Galindo et al., 1999a):

CREATE TABLE PISOS (

PISO# NUMBER(9) NOT NULL PRIMARY KEY,

DUENNO# NUMBER(9) NOT NULL,

DIRECCION VARCHAR2(60),

SUPERFICIE FTYPE1 (15,25) NUMBER(4)

CONSTRAINT NULL_INVALIDO_SUPERFICIE NOT NULL,

PRECIO FTYPE2 (500,3000) NUMBER(6)

DEFAULT UNKNOWN

CONSTRAINT NULL_INVALIDO_PRECIO NOT NULL

CONSTRAINT UNDEFINED_INVALIDO_PRECIO NOT UNDEFINED,

ZONA FTYPE3 (3) DEFAULT UNKNOWN

CONSTRAINT NULL_INVALIDO_ZONA NOT NULL

CONSTRAINT UNDEFINED_INVALIDO_ZONA NOT UNDEFINED);

- Ej. Tupla:

(2, 34, "C/ Gandhi", 90, #10, {1/Centro, 0.8/Norte, 0.2/Este})

27

DDL del LENGUAJE FSQ: CREATE

- **CREATE LABEL:** Creación y definición de etiquetas asociadas a un atributo difuso determinado. Dos formatos:

- 1. **Crear nuevas etiquetas** para atributos difusos Tipo 1 ó 2:

CREATE LABEL <nombre_label> ON [esquema.]tabla.atributo
VALUES alfa, beta, gamma, delta;

- 2. **Copiar las etiquetas definidas en un atributo dentro del dominio de otro atributo**, para atributos difusos de todo tipo:

CREATE LABEL * ON [esquema.]tabla.atributo1
FROM [esquema.]tabla.atributo2

- Si los atributos difusos son de Tipo 3, las etiquetas no se copian, sino que simplemente se establecen ambos atributos como **compatibles** pudiendo usar las mismas etiquetas con su relación de similitud y pudiendo así compararse entre esos dos atributos.

- **Ejemplo:** Crear una etiqueta para el atributo Superficie definido anteriormente:

CREATE LABEL PEQUENNO ON PISOS.SUPERFICIE
VALUES 30, 45, 65, 80;

28

DDL del LENGUAJE FSQL: CREATE

- **CREATE NEARNESS:** Creación de etiquetas asociadas a atributos difusos **Tipo 3** y definición de su relación de proximidad (m):

```
CREATE NEARNESS ON [esquema.]tabla.atributo
    LABEL <lista_de_n_etiquetas>
    VALUES m(E1,E2), m(E1,E3), . . . , m(E1,En),
           m(E2,E3), . . . , m(E2,En),
           . . .
           m(En-1,En);
```

- Para **n** etiquetas es necesario dar $(n^2/2 - n/2)$ valores de m, donde / es la división entera (truncando).

- **Ejemplo:** Crear las etiquetas para el atributo Zona (Barrio) anterior:

```
CREATE NEARNESS ON PISOS.ZONA
    LABEL Centro, Norte, Sur, Este, Oeste
    VALUES 0.8, 0.8, 0.8, 0.8,
           0, 0.5, 0.5,
           0.5, 0.5,
           0;
```

29

Arquitectura de la BDRD

- La **Arquitectura de la BDRD con el Servidor FSQL** es una **arquitectura Cliente/Servidor**, en la que destacan los siguientes **Elementos**:
 - **1. Base de datos:**
 - **Tradicional:** Relaciones con los datos (con el formato especial estudiado para los atributos difusos).
 - **FMB:** Información (en formato relacional) sobre la BDRD: Etiquetas lingüísticas, Relaciones de semejanza, margen de valores aproximados y valor mínimo para considerar dos valores como muy separados.
 - **2. Servidor FSQL:** Hace posible el uso del lenguaje **FSQL** en el SGBD tradicional.
 - Está programado en lenguaje PL/SQL®, un lenguaje inmerso del SGBD Oracle® que permite programar aplicaciones eficientes.
 - **3. Cliente FSQL:** Es un programa independiente que sirve de interfaz entre el usuario y el **Servidor FSQL**.
 - Actualmente existe un cliente llamado **FQ** (Fuzzy Queries) para Windows.

30

Arquitectura de la BDRD

- **Funcionamiento del Servidor FSQL:** Cinco Pasos a nivel interno:

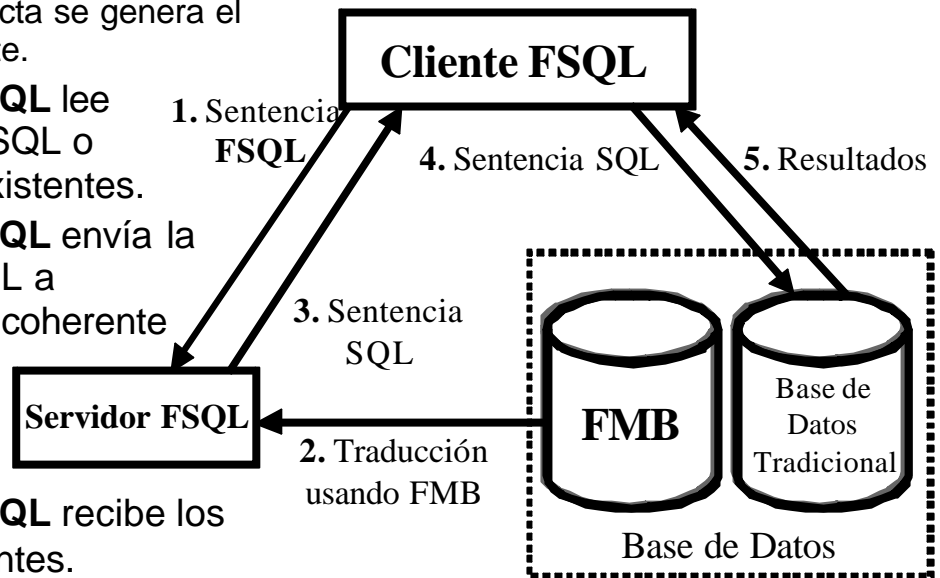
- 1. El **Cliente FSQL** envía la sentencia **FSQL** al **Servidor FSQL**.
- 2. El **Servidor FSQL** analiza la sentencia y si es correcta genera una sentencia en SQL que resuelve dicha sentencia.

- Este paso utiliza la información de la **FMB**.
- Si no es correcta se genera el error pertinente.

- 3. El **Cliente FSQL** lee la sentencia SQL o los errores existentes.

- 4. El **Cliente FSQL** envía la sentencia SQL a cualquier BD coherente con la FMB.

- 5. El **Cliente FSQL** recibe los datos pertinentes.



31

Configuración del Servidor FSQL

- El **Servidor FSQL** debe ser instalado por un **DBA** (usualmente **SYS**) y tiene un conjunto de tablas para almacenar información de diversa índole:
 - **FSQL_ALL_INFO**: Tabla con información general sobre el **Servidor FSQL** (versión, fecha de instalación, instalador, fecha del último uso...).
 - Tiene dos vistas que son a las que accede cada usuario particular:
 - **FSQL_INFO**: Información del sistema.
 - **FSQL_OPTIONS**: Opciones de configuración. Por ejemplo, tiene el tiempo empleado en la última traducción, el número de errores producidos, permite configurar la función de negación, t-norma, s-norma para aplicar cuando se utilicen operadores lógicos en una cláusula **WHERE**.
 - **FSQL_STATS**: Tabla con información sobre el uso del **Servidor FSQL** (Accesos producidos, Accesos sin errores, número de errores global, tiempo total empleado, tiempo total empleado en los accesos sin errores...).

32

Algunas Aplicaciones de FSQL

1. FSQL en aplicaciones de Gestión (Galindo et al., 1999a, 1999b): Aplicación a la gestión de una Inmobiliaria.

– Ejemplos de **Atributos Difusos**:

- **Tipo 1**: Número habitaciones, precio comunidad...
- **Tipo 2**: Precio, superficie, antigüedad...
- **Tipo 3**: Zona, tipo de inmueble (piso, estudio, duplex, solar, trastero, cochera, chalet...), vistas, luz...

– Ejemplo:

```
SELECT CDEG(*), Inmuebles_Venta.*
FROM   Inmuebles_Venta
WHERE  Tipo          FEQ   $Chalet  0.5
       AND Superficie FGEQ $Grande 0.7
       AND Habitaciones FGEQ #6      0.7
       AND Precio     FEQ   #10000 0.7;
```

- Flexibilidad con distintos **comparadores difusos**, **constantes difusas**, **umbrales** y **operadores lógicos**, que nos permiten no sólo encontrar mejor lo que el cliente busca sino además obtener una lista ordenada según el grado de cumplimiento de la condición por parte de cada inmueble.

33

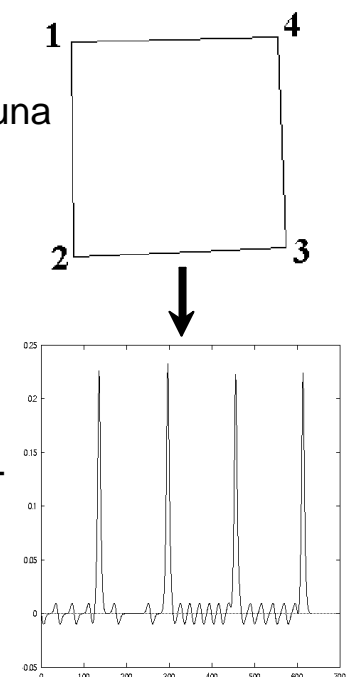
Algunas Aplicaciones de FSQL

2. Clasificación Difusa de Imágenes

(Aranda et al., 1998): A partir de datos extraídos de una imagen con los contornos de cierto objeto puede aplicarse un sistema para clasificar dicha figura:

- a) **Obtener el contorno**, a partir de la imagen.
- b) **Calcular la curva de curvatura**: Mide el nivel de curva/recta que tiene el contorno en cada punto.
- c) **Obtener Atributos**: Número de vértices, signo y valor de la curvatura en esos vértices, distancia entre vértices, tamaño de la curvatura...
- d) **Clasificación con consultas difusas sobre esos atributos**: Ejemplo, para un triángulo isósceles:

```
SELECT Imagen#, CDEG(*) FROM OBJETOS
WHERE Signo1='P' AND Signo2='P' AND Signo3='P'
       AND Dist1 FEQ   $Grande 0.7 AND GradoK1 FEQ   $Grande 0.7
       AND Dist2 FLEQ $Normal 0.7 AND GradoK2 FLEQ $Normal 0.7
       AND Dist3 FEQ   $Grande 0.7 AND GradoK3 FLEQ $Normal 0.7;
```



34

Algunas Aplicaciones de FSQL

3. FSQL como Herramienta de Data Mining:

- Se sospecha que un lenguaje de consulta difusa a bases de datos (*crisp* o difusas) puede ser de gran utilidad en procesos de Data Mining. Aún no ha sido muy estudiado en este aspecto.
- Una de las aplicaciones estudiadas consiste en aplicarlo en un entorno financiero (Carrasco et al., 1999):
 - **DAPHNE** (Carrasco, 1998): A partir de la BD de una entidad financiera este programa efectúa, en síntesis lo siguiente:
 - **Agrupamiento (*Clustering*)** de los elementos según sus atributos.
 - Obtiene **valor de los atributos para un elemento “típico”**: **Centroides** “ C_i ” para los atributos “ A_i ” del Grupo G.
 - Con esos **CENTROIDES**, pueden calcularse los elementos de ese grupo (G) y el grado de pertenencia de cada uno:

```
SELECT tabla.*, CDEG(*)
FROM tabla
WHERE A1 FEQ C1 THOLD g
AND A2 FEQ C2 THOLD g
...
AND An FEQ Cn THOLD g;
```

```
Ej.:SELECT cliente#, CDEG(*)
FROM clientes
WHERE Nomina='N'
AND UsoTarjeta FEQ $Medio 0.7
AND Saldo FEQ #30533 0.7
ORDER BY 2 DESC;
```

35

Algunas Aplicaciones de FSQL

4. Deducción en Bases de Datos Difusas (Blanco, 2001): La idea principal es la **Generalización de las Reglas de la Lógica Clásica**:

- **Una Regla está formada por una Cabeza y un Cuerpo**:
 - **Cabeza**: Es el predicado que se está definiendo y cuyos datos queremos averiguar.
 - **Cuerpo**: Es un conjunto de predicados que definen las condiciones que deben cumplir los datos que pertenezcan al predicado de la cabeza.
- **Ejemplo**: $R(X,Y) \rightarrow S(X,Z) \cup T(Z,Y)$
 - Las dos ocurrencias de la variable Z no tienen porqué ser la misma: Existe una igualdad implícita.
 - Ese tipo de comparaciones pueden hacerse de forma DIFUSA.
- **Ejemplo: Dadas dos relaciones**
 - **Historial (Paciente, Edad)**: Con datos como (1, 20), (2, #40), (3, Joven)...
 - **Riesgos (Edad, Enfermedad, Probabilidad)** : Con datos como (Joven, Cefalea, Alta), (Viejo, Parkinson, Normal), ([0,4], Gastroenteritis, [30,40])...

Averiguar las probabilidades de que cada paciente tenga cada enfermedad según su edad:

```
CorreRiesgo(Paciente,Enfermedad,Probabilidad) →
Historial (Paciente, Edad1) ∪
Riesgos(Edad2,Enfermedad,Probabilidad) ∪
Edad1 FEQ Edad2 THOLD umbral
```

36

Bibliografía

- M.C. Aranda, J. Galindo, "Clasificación de Imágenes de una Base de Datos utilizando información de su forma". IV Jornadas Internacionales de Informática, pp 565-574, Las Palmas de Gran Canaria (Spain), July 1998.
- I.J. Blanco, "Deducción en Bases de Datos Relacionales Difusas". Ph. Doctoral Thesis, University of Granada (Spain), July 2001.
- P. Bosc, M. Galibourg G. Hamon, "Fuzzy Querying with SQL: Extensions and Implementation Aspects". Fuzzy Sets and Systems, 28, pp. 333-349, 1988.
- P. Bosc, O. Pivert, "SQLf: A Relational Database Language for Fuzzy Querying". IEEE Transactions on Fuzzy Systems, 3, pp. 1-17, 1995.
- R.A. Carrasco, "Data Mining: Un prototipo para Clustering Financiero". Trabajo de Investigación del programa de doctorado "Tratamiento de la Información en Inteligencia Artificial", Dpto. Ciencias de la Computación e I.A., Universidad de Granada (Spain), Septiembre 1998.
- R.A. Carrasco, J. Galindo, M.A. Vila, J.M. Medina, "Clustering and Fuzzy Classification in a Financial Data Mining Environment". 3rd International ICSC Symposium on Soft Computing, SOCO'99, pp. 713-720. Genova (Italy), June 1999.
- R. Carrasco, J. Galindo, A. Vila, "Using Artificial Neural Network to Define Fuzzy Comparators in FSQL with the Criterion of some Decision-Maker". In "Bio-inspired applications of connectionism.- 2001", eds. J. Mira and A. Prieto, Lecture Notes in Computer Science (LNCS) 2085, part II, pp. 587-594. Ed. Springer, 2001.
- D. Dubois, H. Prade, "Weighted Minimum and Maximum Operations in Fuzzy Set Theory". Information Sciences, 39, pp. 205-210, 1986.
- D. Dubois, H. Prade, "Measuring Properties of Fuzzy Sets: A General Technique and its use in Fuzzy Query Evaluation". Fuzzy Sets and Systems, 38, pp. 137-152, 1990.

37

Bibliografía

- D. Dubois, H. Prade, "A Review of Fuzzy Set Aggregation Connectives". Information Sciences, 36, pp. 85-121, 1995.
- J. Galindo, J.M. Medina, O. Pons, J.C. Cubero, "A Server for Fuzzy SQL Queries", in "Flexible Query Answering Systems", eds. T. Andreassen, H. Christiansen and H.L. Larsen, Lecture Notes in Artificial Intelligence (LNAI) 1495, pp. 164-174. Ed. Springer, 1998a.
- J. Galindo, J.M. Medina, A. Vila, O. Pons, "Fuzzy Comparators for Flexible Queries to Databases". Iberoamerican Conf. on Artificial Intelligence, IBERAMIA'98, Lisbon (Portugal), pp. 29-41, 1998b.
- J. Galindo, "Tratamiento de la Imprecisión en Bases de Datos Relacionales: Extensión del Modelo y Adaptación de los SGBD Actuales". Ph. Doctoral Thesis, University of Granada (Spain), March 1999 (www.lcc.uma.es).
- J. Galindo, J.M. Medina, J.C. Cubero, O. Pons, "Management of an Estate Agency Allowing Fuzzy Data and Flexible Queries". EUSFLAT-ESTYLF Joint Conference, pp. 485-488, Palma de Mallorca (Spain), September 1999a.
- J. Galindo, M.C. Aranda, "Gestión de una Agencia de Viajes usando Bases de Datos Difusas y FSQL". Turismo y tecnologías de la información y las comunicaciones: Nuevas tecnologías y calidad. TuriTec'99, pp. 65-77, Málaga (Spain), September 1999b.
- J. Galindo, J.M. Medina, J.C. Cubero, M.T. García, "Fuzzy Quantifiers in Fuzzy Domain Calculus". 8th Int. Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU'2000, pp. 1697-1704. Madrid (Spain), Julio 2000a (www.lcc.uma.es).
- J. Galindo, J.M. Medina, J.M. Rodríguez, "Comparadores para Bases de Datos Difusas: Definiciones, Clases y Relaciones". X Congreso Español sobre Tecnologías y Lógica Fuzzy (ESTYLF'2000), pp. 187-192. Sevilla (Spain), September 2000b (www.lcc.uma.es).

38

Bibliografía

- D. Li, D. Liu, "A Fuzzy Prolog Database System". Research Studies Press LTD. John Wiley & Sons Inc., 1990.
- J.M. Medina, "Bases de Datos Relacionales Difusas. Modelo Teórico y Aspectos de su Implementación". PhD. Thesis, Univ. of Granada (Spain), 1994 (www.decsai.ugr.es).
- J.M. Medina, O. Pons, M.A. Vila, "FIRST. A Fuzzy Interface for Relational SysTems". VI International Fuzzy Systems Association World Congress (IFSA'1995). Sao Paulo (Brasil), 1995.
- H. Nakajima, T. Sogoh, M. Arao, "Fuzzy Database Language and Library - Fuzzy Extension to SQL". Second International Conference on Fuzzy Systems, FUZZ-IEEE'93, pp. 477-482, 1993.
- E. Sanchez, "Importance in Knowledge Systems". Information Systems, 14, pp. 455-464, 1989.
- V. Tahani, "A Conceptual Framework for Fuzzy Query Processing--A Step toward Very Intelligent Database Systems". Information Processing and Management, 13, pp. 289-303, 1977.
- M.A. Vila, J.C. Cubero, J.M. Medina, O. Pons, "The Generalized Selection: An Alternative Way for the Quotient Operations in Fuzzy Relational Databases". In "Fuzzy Logic and Soft Computing". Eds. B. Bouchon-Meunier, R.R. Yager, L.A. Zadeh, pp. 241-250. World Scientific, Singapore, 1995.
- M. Wong, K Leung, "A Fuzzy Database-Query Language". Inform. Systems, 15, pp. 583-590, 1990.
- R.R. Yager, "Quantified Propositions of a Linguistic Logic". International Journal of Man-Machine Studies, 19, pp. 195-227, 1983.
- R.R. Yager, "Connectives and Quantifiers in Fuzzy Sets". Fuzzy Sets and Syst., 40, pp. 39-76, 1991.
- L.A. Zadeh, "A Computational Approach to Fuzzy Quantifiers in Natural Languages". Computer Mathematics with Applications, 9, pp. 149-183, 1983.