

UNIVERSIDAD DE GRANADA

**E.T.S. DE INGENIERÍA
INFORMÁTICA**



**Departamento de Ciencias de la Computación
e Inteligencia Artificial**

**BASES DE DATOS RELACIONALES DIFUSAS:
MODELO TEÓRICO Y ASPECTOS DE SU
IMPLEMENTACIÓN**

TESIS DOCTORAL

Juan Miguel Medina Rodríguez

Granada, 25 de Mayo de 1994

Índice

1	Introducción General	11
2	Modelo Relacional de Bases de Datos	17
2.1	Introducción	17
2.2	Modelo Relacional de Codd. Elementos del Modelo	17
2.2.1	Estructura de Datos	18
2.2.2	Integridad de los Datos	20
2.2.3	Manipulación de los Datos	21
2.3	Evolución del Modelo Relacional	25
2.4	El Problema del Tratamiento de la Información Incompleta	28
2.4.1	La Aproximación de Codd	29
2.4.2	El Esquema de los “Valores por Defecto”	36
2.4.3	Críticas a las Aproximaciones Clásicas	38
2.5	Sistemas Relacionales Comerciales	42
2.5.1	El SBDR Oracle	43
3	Modelos de Bases de Datos Relacionales Difusas	47
3.1	Introducción	47
3.2	Teoría de Conjuntos Difusos	47
3.2.1	Conjuntos Difusos	48
3.2.2	Operaciones Sobre Conjuntos Difusos	51
3.2.3	Números Difusos	52
3.3	El Tratamiento de la Información Incompleta por medio de Conjuntos Difusos	56
3.3.1	El Enfoque Posibilístico del Modelo de Prade-Testemale	57

3.3.2	El Modelo de Umano et al.	59
3.3.3	Comentarios	60
3.4	Características de un Modelo de Base de Datos Relacional Difuso . . .	60
3.5	Principales Modelos de Bases de Datos Difusas	63
3.5.1	Modelo Relacional Difuso	64
3.5.2	Unificación Mediante Relaciones de Similitud. Modelo de Buckles-Petry	65
3.5.3	Modelos Relacionales Sobre Distribuciones de Posibilidad	70
3.5.3.1	Modelo de Umano y Fukami	71
3.5.3.2	Modelo de Prade-Testemale	75
3.5.3.3	Modelo de Zemankova-Kaendel	79
4	GEFRED. Modelo Generalizado de Bases de Datos Relacionales Difusas	83
4.1	Introducción	83
4.2	Estructura de Datos	84
4.2.1	Dominio Difuso Generalizado, D_G	84
4.2.2	Relación Difusa Generalizada, R_{FG}	88
4.2.2.1	Significado de la Relación Difusa Generalizada	89
4.2.3	Clave Primaria Generalizada K_G	92
4.3	Manipulación de los Datos	95
4.3.1	Operadores de Comparación Difusa	95
4.3.1.1	Comparador Extendido, θ_e	95
4.3.1.2	Comparador Difuso Generalizado, $\Theta_G^{\theta_e}$	96
4.3.2	Álgebra Relacional Difusa Generalizada	98
4.3.2.1	Unión Difusa Generalizada, \cup_G	98
4.3.2.2	Intersección Difusa Generalizada, \cap_G	103
4.3.2.3	Diferencia Difusa Generalizada, $-_G$	106
4.3.2.4	Producto Difuso Generalizado, \times_G	107
4.3.2.5	Proyección Difusa Generalizada, \mathcal{P}_G	108
4.3.2.6	Selección Difusa Generalizada, \mathcal{S}_G	109
4.3.2.7	Reunión Difusa Generalizada, \bowtie_G	111
4.3.3	Cardinalidad Difusa Generalizada	113
4.4	Ejemplo de Representación y Manipulación sobre GEFRED	115

4.5	Estudio de GEFRED frente a los Modelos Teóricos Precedentes	123
4.5.1	GEFRED vs. Modelo de Buckles-Petry	123
4.5.1.1	Estructura de Datos	123
4.5.1.2	Operaciones Algebraicas	124
4.5.1.3	Resolución de Consultas	125
4.5.2	GEFRED vs. Modelo de Umano-Fukami	128
4.5.2.1	Estructura de Datos	128
4.5.2.2	Operaciones Algebraicas	129
4.5.2.3	Resolución de Consultas	131
4.5.3	GEFRED vs. Modelo de Prade-Testemale	135
4.5.3.1	Estructura de Datos	136
4.5.3.2	Operaciones Algebraicas	137
4.5.3.3	Resolución de Consultas	139
4.5.4	GEFRED vs. Modelo de Zemankova-Kandel	144
4.5.4.1	Estructura de Datos	144
4.5.4.2	Operaciones Algebraicas	145
4.5.4.3	Resolución de Consultas	148
4.6	Conclusiones	151
5	Un Sistema Gestor de BD Relacionales Difusas Basado en GEFRED	153
5.1	Introducción	153
5.2	FIRST: Un Interface Difuso para Sistemas Relacionales	154
5.2.1	Esquema General de FIRST	155
5.2.2	Representación del Conocimiento Impreciso	158
5.2.2.1	Representación de los Datos	159
5.2.2.2	Comparadores Difusos Generalizados	163
5.2.2.3	Cualificadores del Umbral de Consulta	166
5.2.2.4	Cuantificadores difusos de la Consulta	168
5.3	Implementación de FIRST	176
5.3.1	Implementación del Conocimiento Impreciso en la Base de Datos	178
5.3.2	Implementación del Conocimiento Impreciso en el Catálogo de la BD. Base de Metaconocimiento Difuso (FMB)	180
5.3.2.1	Tablas de la Base de Metaconocimiento Difuso	181
5.3.3	Ejemplo de Implementación en FIRST de la BD y la FMB	188

5.3.4	Implementación del Conocimiento Impreciso en los Procedimientos del Gestor. El Procesador de Llamadas Difusas (FCP)	193
5.3.4.1	Procesado de las Operaciones de Manipulación Difusas	195
5.3.4.2	Implementación de la Selección Difusa Generalizada .	196
5.3.4.3	Cardinalidad Absoluta y Relativa en FIRST	224
5.3.4.4	La Cuantificación en FIRST	228
5.3.5	Ejemplo de Procesamiento de Consulta en FIRST	233

Conclusiones y Direcciones Futuras **237**

A	Implementación de FIRST sobre un RDBMS comercial(ORACLE)	241
A.1	Introducción	241
A.2	Base de Datos Ejemplo	243
A.3	Estructura de Datos de la FMB y de la BD	256
A.4	El Procesador de Llamadas Difusas (FCP)	265
A.4.1	Funciones del FCP Implementadas	267
A.5	Un Procesador Elemental de FSQL	276
A.5.1	Funcionamiento del Procesador de FSQL	278
A.5.2	Rutinas del Procesador de FSQL	281
A.6	FIRST en Acción.	285
A.6.1	Ejemplos de Consultas	286

Tablas

2.1	Tablas de verdad para la lógica trivaluada	31
2.2	Tablas de verdad para la lógica tetraevaluada	32
2.3	Relaciones de Proveedores P, de Componentes C, y equireunión externa de P y C sobre los atributos PCIUDAD y CCIUDAD	33
2.4	Reunión natural externa sobre P y C	34
2.5	Relación de Proveedores P	41
3.1	Conjunto difuso representando el concepto “joven”	49
3.2	Representación de información incompleta mod. posibilísticos	61
3.3	Relación de Similitud sobre el dominio “color de pelo”	66
3.4	Relación Persona	67
3.5	Relación intermedia	68
3.6	Resultado final de la consulta	68
3.7	Ejemplo de relación para el modelo de Umano et al.	71
3.8	Ejemplo de relación para el modelo de Prade-Testemale	75
4.1	Tipos de datos	85
4.2	Tabla de Empleados, Emp	116
4.3	Relación de semejanza sobre el dominio RENDIMIENTO.	116
4.4	Cálculo de \mathcal{S}_1	119
4.5	Cálculo de \mathcal{S}_2	120
4.6	Cálculo de \mathcal{S}_3	120
4.7	Cálculo de \mathcal{S}_4	121
4.8	Cálculo de $\mathcal{S}_1 \wedge_G \mathcal{S}_2$	121
4.9	Cálculo de $\mathcal{S}_3 \wedge_G \mathcal{S}_4$	121
4.10	Cálculo de $((\mathcal{S}_1 \wedge_G \mathcal{S}_2) \vee_G (\mathcal{S}_3 \wedge_G \mathcal{S}_4))$	122

4.11	$\mathcal{P}_G((\mathcal{S}_1 \wedge_G \mathcal{S}_2) \vee_G (\mathcal{S}_3 \wedge_G \mathcal{S}_4); X)$	122
4.12	Relación Persona	126
4.13	Relación de Similitud sobre el dominio “color de pelo”	126
4.14	Cálculo de la Selección \mathcal{S}_{color}	127
4.15	Resultado final de la Consulta	127
4.16	Relación Persona	131
4.17	Cálculo de $\mathcal{S}_T^{es-joven}$	133
4.18	Cálculo de $\mathcal{P}(\mathcal{S}_T^{es-joven}; X)$	133
4.19	Cálculo de $\mathcal{S}_P^{es-joven}$	134
4.20	Cálculo de $\mathcal{S}_P^{es-joven} -_G \mathcal{S}_T^{es-joven}$	134
4.21	Cálculo de $\mathcal{P}(\mathcal{S}_P^{es-joven} -_G \mathcal{S}_T^{es-joven}; X)$	135
4.22	Relación Estudiantes	140
4.23	Cálculo de $\mathcal{S}_N^{\gg bueno}$	142
4.24	Cálculo de $\mathcal{P}_G(\mathcal{S}_N^{\gg bueno}; X)$	142
4.25	Cálculo de $\mathcal{S}_{\Pi}^{\gg bueno}$	143
4.26	Cálculo de $\mathcal{S}_{\Pi}^{\gg bueno}$	143
4.27	Relación Persona	149
4.28	Relación de Similitud sobre el dominio “color de pelo”	149
4.29	Cálculo de $\mathcal{P}_G(\mathcal{S}_P^{\approx casta\tilde{n}o}; X)$	150
4.30	Cálculo de $\mathcal{P}_G(\mathcal{S}_C^{\approx casta\tilde{n}o}; X)$	151
5.1	Representación de Atributos del tipo 2.	178
5.2	Representación para atributos de Tipo 3	180
5.3	EMPLEADOS	189
5.4	Relación de Semejanza sobre RENDIMIENTO.	189
5.5	EMPLEADOS	191
5.6	FUZZY_COL	191
5.7	FUZZY_OBJECT_LIST	192
5.8	FUZZY_LABEL_DEF	192
5.9	FUZZY_NEARNESS_DEF	193
5.10	FUZZY_APROX_DEF	193
5.11	Selección basada en FEQ sobre atributos Tipo 1	199
5.12	Selección usando FEQ(F, “crisp”, u) sobre atributos de Tipo 2	201
5.13	Selección usando FEQ(F, “difuso”, u) sobre atributos de Tipo 2	204

5.14	Continuación de la tabla 5.13	205
5.15	Selección usando FEQ(F, “intervalo[r,s]”, u) sobre atributos de Tipo 2	206
5.16	Continuación de la tabla 5.15	207
5.17	Selección usando FEQ(F, “aprox(d)”, u) sobre atributos de Tipo 2	208
5.18	Continuación de la tabla 5.17	209
5.19	Selección usando FEQ(F, “simple”, u) sobre atributos de Tipo 3	210
5.20	Selección usando FEQ(F, “distr. posib.”, u) sobre atributos de Tipo 3	211
5.21	Composición del comparador “mayor o igual” con los diferentes tipos	213
5.22	Composición del comparador “menor que” con los diferentes tipos	217
5.23	Resultado de la consulta ejemplo	236
A.1	Tabla <i>Personal</i>	247
A.2	Tabla <i>Aptitud</i>	248
A.3	Tabla <i>Dpto.</i>	249
A.4	Tabla <i>Puestos</i>	249
A.5	Tabla <i>Empleos</i>	250
A.6	Tabla <i>Ciudades</i>	251
A.7	Relación de semejanza sobre el dominio <i>Estudios</i>	251
A.8	Relación de semejanza sobre el dominio <i>Profesión</i>	252
A.9	Relación de semejanza sobre el dominio <i>Rendimiento</i>	252
A.10	Contenido de la tabla Fuzzy_Col_List relativo a la BD ejemplo	256
A.11	Contenido de la tabla Fuzzy_Object_List relativo a la BD ejemplo	257
A.12	Continuación de la tabla A.11	258
A.13	Contenido de la tabla Fuzzy_Label_Def relativo a la BD ejemplo	258
A.14	Contenido de la tabla Fuzzy_Nearness_def relativo a la BD ejemplo	259
A.15	Contenido de la tabla Fuzzy_Aprox_def relativo a la BD ejemplo	260
A.16	Estructura en Oracle de la tabla <i>Personal</i>	260
A.17	Estructura en Oracle de la tabla <i>Aptitud</i>	261
A.18	Estructura en Oracle de la tabla <i>Dpto.</i>	261
A.19	Estructura en Oracle de la tabla <i>Puestos</i>	262
A.20	Estructura en Oracle de la tabla <i>Empleos</i>	262
A.21	Estructura en Oracle de la tabla <i>Ciudades</i>	263
A.22	Representación en Oracle de los datos de la tabla <i>Empleos</i>	264
A.23	Formato de salida para datos difusos	270

A.24 Resultado de la Consulta A.1	288
A.25 Resultado de la Consulta A.2	289
A.26 Resultado de la Consulta A.3	290
A.27 Resultado de la Consulta A.4	293

Figuras

3.1	Aspecto general de un número difuso	53
3.2	Número difuso trapezoidal normalizado	54
4.1	Definición de etiquetas sobre los atributos EDAD y SALARIO.	117
4.2	Etiquetas lingüísticas definidas sobre Calificación	141
5.1	Esquema General de FIRST	156
5.2	Distribución Trapezoidal	158
5.3	Etiqueta Lingüística "alto"	159
5.4	"Aproximadamente n"	160
5.5	Distribución Intervalar	161
5.6	Tipos UNKNOWN y UNDEFINED	162
5.7	Operadores "igual" y "aproximadamente igual"	164
5.8	Operadores Relacionales Difusos	167
5.9	Etiqueta lingüística para el cuantificador "muchos"	172
5.10	Etiqueta lingüística para el cuantificador relativo "mayoría"	174
5.11	Esquema de la Base de Metaconocimiento FMB.	182
5.12	Definiciones de FUZZY_QUALIF_DEF y FUZZY_QUANT_DEF	187
5.13	Definición de etiquetas sobre los atributos EDAD y SALARIO.	190
5.14	Etiqueta lingüística para el cuantificador "muchos"	230
5.15	Etiqueta lingüística para el cuantificador relativo "mayoría"	232
A.1	Etiquetas sobre el dominio <i>Edad</i> , (en años)	253
A.2	Etiquetas sobre el dominio <i>Experiencia</i> , (en años)	253
A.3	Etiquetas sobre el dominio <i>Beneficios</i>	254
A.4	Etiquetas sobre el dominio <i>Sueldo</i>	254

A.5	Etiquetas sobre el dominio <i>Comisión</i>	255
A.6	Etiquetas sobre el dominio <i>Distancia</i>	255
A.7	Alfacortes para un umbral u	268
A.8	Esquema del Procesado de una Sentencia FSQ	279

Capítulo 1

Introducción General

Hoy en día la información es uno de los factores que mas peso específico tiene en el desarrollo de un país. Por este motivo, cualquier sociedad que pretenda no quedar descolgada en su desarrollo debe estar al tanto de las técnicas que van surgiendo en el almacenamiento, transmisión y análisis de la información. Historicamente, las Bases de Datos han sido las herramientas diseñadas para llevar a cabo las tareas de almacenamiento y para proporcionar algunos de los mecanismos necesarios para el análisis de la información.

El objetivo de una Base de Datos es el de almacenar la información en forma conveniente, el permitir su modificación de forma segura y el de facilitar el proceso de recuperación de aquella información que en un momento dado nos resulte necesaria, todo ello en un formato adecuado a nuestras necesidades.

A lo largo de las últimas décadas han sido múltiples las aproximaciones surgidas para atender estos requisitos. Los criterios a los que responden las diferentes clasificaciones que se pueden realizar sobre las mismas son: la organización de los datos y el tipo de los mismos. Con respecto a la organización de los datos, los principales enfoques aparecidos han sido el de Redes, el Jerárquico, (en [Date 90I] puede encontrarse una descripción de los principales desarrollos construidos en torno a estos dos enfoques), el Relacional [Codd 70] (actualmente en un grado de implantación creciente) y, ultimamente, el Orientado a Objetos [Kim 90] y el de Bases de Datos Lógicas, cuyas bases teóricas pueden encontrarse en: [Gallaire, Minker, Nicolas 84], [Reiter 84]. En cuanto al tipo de los datos que pueden soportar las diferentes aproximaciones, tradicionalmente han sido datos de tipo “texto” (datos numéricos, fechas, alfanuméricos, documentales).

Actualmente los esfuerzos se encaminan hacia el tratamiento de información audiovisual surgiendo las Bases de Datos Multimedia.

Es interesante hacer notar que, si bien la organización de los datos no tiene que estar en relación, necesariamente, con el tipo de datos que se pueden soportar, obviamente, ciertas organizaciones presentan mayor flexibilidad que otras, en cuanto a la diversidad de tipos que pueden tratar. En particular, el enfoque de Bases de Datos Orientadas a Objeto es el que se muestra más versátil en este aspecto.

Como hemos apuntado al principio, el acceso a la información es un elemento estratégico en la sociedad actual. Sin embargo, lo más importante no es la cantidad de información a la que se pueda acceder, si no la calidad de los mecanismos de que se dispone para acceder a aquella información que nos interesa en un momento dado. En este último aspecto, los modelos de Bases de Datos vigentes en la actualidad representan un gran avance con respecto a los mecanismos de que se disponía hace no mucho tiempo. No obstante, a pesar de los esfuerzos realizados, la representación de la información y el tratamiento de la misma, se encuentra todavía lejos de los mecanismos de expresión utilizados habitualmente por el ser humano. En este sentido, se está realizando un gran esfuerzo en el ámbito de la Inteligencia Artificial para resolver los problemas teóricos y prácticos relacionados con la elaboración de Bases de Datos Inteligentes, (en [Parsaye et al 89] se recogen algunas propuestas al respecto). La idea que subyace a este tipo de bases de datos es la de facilitar mecanismos para almacenar y recuperar información siguiendo un esquema más próximo al empleado por el ser humano. Algunos de los principales aspectos que caracterizan a este tipo de aproximaciones son:

- La posibilidad de representar y manipular información cuya semántica se encuentra más próxima al esquema humano de representación de la realidad.
- La introducción de mecanismos que doten al sistema de capacidad para inferir información a partir de la que se encuentra previamente almacenada en el sistema.

El primer aspecto conlleva la incorporación en las bases de datos de capacidad para representar y manipular información imprecisa y subjetiva en todas sus vertientes. El segundo aspecto implica la integración de las bases de datos y un amplio abanico de disciplinas relacionadas con la Lógica.

Las propuestas surgidas con relación al primer aspecto se distinguen de los sistemas de BD tradicionales en que, en estos primeros sistemas, la información que se

posee sobre un atributo, existe o no existe, no permitiéndose sobre su conocimiento ningún grado de incertidumbre ni de imprecisión, es decir no se pueden representar ni tratar informaciones del tipo *"Jorge tiene un color de pelo mas castaño que moreno tendiendo a rubio"*. Tampoco se contempla la obtención de información en términos imprecisos a partir de aquella que se encuentra expresada en forma precisa, así, no se arbitran mecanismos para, p.e., obtener, de un atributo que almacene información sobre las alturas de una población, aquellos individuos que sean *"altos"*. El estudio que contempla esta memoria, está inscrito en el ámbito de las propuestas que tratan el tipo de información analizada arriba y va a centrarse en el análisis de aquellas nuevas técnicas, aparecidas en el campo de las bases de datos y de la teoría de conjuntos difusos de Zadeh [Zadeh 65], que permiten la representación y el tratamiento de este tipo de información.

En este sentido, las diferentes aproximaciones aparecidas en la literatura presentan su propia versión de como combinar los modelos de representación de conocimiento anteriores con la teoría de conjuntos difusos de Zadeh [Zadeh 65]. La mayoría de los autores centran sus propuestas teóricas en una extensión del modelo relacional que contemple, en mayor o menor grado, un tratamiento para este tipo de datos. Esto no quiere decir que no existan otras organizaciones que presenten un marco adecuado para la inclusión en las mismas del tratamiento de información difusa o imprecisa. En particular el enfoque orientado a objetos presenta buenas perspectivas para representar información de estas características. No obstante, el Modelo Relacional, [Codd 70], se encuentra en un estado de definición e implementación mas avanzado, por lo que vamos a centrar nuestro esfuerzo en el estudio de la diferentes propuestas aparecidas en torno al Modelo de Bases de Datos Relacional Difuso.

Objetivos

Con esta memoria se pretende contribuir al desarrollo del modelo relacional con el objeto de dotarlo de una mayor capacidad para representar y manipular información imprecisa en el sentido más amplio de la acepción. Como propósito adicional, y no por ello menos importante, se ha intentado sentar las bases teórico-prácticas para la implementación de desarrollos específicos de Sistemas de Bases de Datos Relacionales Difusos. Estos objetivos generales se desglosan en los siguientes propósitos parciales:

- Definir un modelo de Base de Datos Relacionales Difusas que combine e integre

en un marco común los aspectos más relevantes de anteriores contribuciones. Este modelo, que hemos denominado GEFRED, (A Generalized model for Fuzzy Relational Databases) [Medina et al, 93(b)], contempla, además, un esquema de representación y manipulación de la información, difusa o no, que permite considerar a las anteriores propuestas, incluyendo al propio modelo relacional clásico, como casos particulares del mismo.

- La definición del modelo propuesta tiene como uno de sus principales objetivos el de proporcionar un marco teórico adecuado para la construcción de implementaciones de SBDRD. En este sentido, una de las ideas más importantes que orientaron su especificación es la de servir de soporte para el desarrollo de implementaciones clasificadas en dos categorías:
 - Sistemas de Bases de Datos Relacionales Difusas construidos “ad-hoc” de acuerdo con las especificaciones expresadas en GEFRED. Los sistemas pertenecientes a esta categoría presentan la ventaja de que las características de representación y manipulación se pueden implementar a más bajo nivel, con lo que la eficiencia puede verse incrementada. Por contra, presentan el inconveniente de que requieren de un mayor esfuerzo su desarrollo. Sin embargo, puesto que GEFRED incluye al modelo relacional clásico como un caso particular, es factible, (y deseable), que los desarrollos construidos de acuerdo con esta filosofía sean compatibles con el extenso parque de sistemas relacionales instalado, potenciando sus capacidades.
 - SBDRD construidos sobre plataformas relacionales existentes. Los sistemas construidos de acuerdo con estos planteamientos presentan la ventaja de que pueden aprovechar los recursos proporcionados por el RDBMS anfitrión para llevar a cabo una extensión de sus capacidades a fin de que estos puedan incorporar tratamiento para información imprecisa. Esto hace que su desarrollo sea más fácil, por lo que representan una buena alternativa para la construcción de prototipos que evalúen la viabilidad de los postulados de GEFRED.
- De acuerdo con los objetivos planteados anteriormente, nos hemos propuesto introducir el esquema de un sistema que opere según lo establecido en la segunda categoría expuesta arriba. Dicho sistema llamado FIRST, (Fuzzy Interface for

Relational Systems) [Medina et al, 93(a)], constituye una pequeña muestra de como pueden llevarse a cabo desarrollos basados en GEFRED que conduzcan a la implementación efectiva de SBDRD sobre RDBMS convencionales.

- Por último, para ilustrar las posibilidades reales que encierran los contenidos de esta memoria, se ha llevado a cabo la construcción de un prototipo experimental de FIRST sobre un RDBMS comercial.

Descripción por Capítulos

El capítulo segundo nos presentará las características del “Modelo Relacional de Bases de Datos” ([Codd 70], [Codd 85], [Codd 86b], etc.). Se realizará un recorrido elemental por los elementos que configuran el modelo relacional, haciendo hincapié en los aspectos en los que los modelos de bases de datos relacionales difusas pueden aportar nuevas perspectivas. El capítulo tercero mostrará, de acuerdo con la definición del “Modelo Relacional” precedente, cuales son las buenas propiedades que ha de reunir una extensión “Difusa” del mismo. Se analizarán los objetivos que ha de cumplir dicho modelo, así como la forma en que ha de satisfacer los requisitos específicos del tratamiento de información difusa. A continuación se estudiarán los diferentes modelos aparecidos en la literatura ([Umano 82], [Buckles, Petry, 82], [Zemankova 84], [Prade, Testemale, 87], etc.). En el capítulo cuarto se dará una descripción detallada de GEFRED, el modelo teórico propuesto en esta memoria. Se enunciarán sus principales elementos, tanto estructurales, como de manipulación de información de naturaleza “difusa”. Una vez introducidos los aspectos que lo configuran, este modelo será contrastado con las propuestas más significativas aparecidas en el ámbito de las bases de datos relacionales difusas.

Con los temas abordados en los cuatro primeros capítulos tendremos sentadas las bases teóricas sobre las que se fundamenta la segunda parte de este trabajo. En esta parte vamos a introducir los elementos básicos sobre los que construir FIRST, la implementación de un Gestor de Bases de Datos Relacionales Difusas basado en el modelo GEFRED.

El capítulo quinto presenta el esquema general del gestor propuesto, así como los criterios adoptados para la representación del conocimiento impreciso en que sustenta su implementación. A continuación, se detallarán los elementos básicos de la imple-

mentación y se mostrarán las decisiones tomadas acerca de las variables que presenta la misma. En el apéndice incluido en esta memoria se abordará la descripción de un desarrollo experimental de FIRST sobre el gestor Oracle. Se emplearán algunos ejemplos para ilustrar el funcionamiento de este prototipo.

Por último, el capítulo seis recoge una síntesis de los resultados alcanzados, así como de los trabajos a realizar y las direcciones a seguir.

Capítulo 2

Modelo Relacional de Bases de Datos

2.1 Introducción

Este capítulo pretende dar una visión clara de lo que es un Sistema de Base de Datos Relacional (SBDR en lo sucesivo). La tarea no es fácil puesto que se ha escrito y estudiado mucho sobre este tema desde que E.F. Codd [Codd 70] publicó en 1970 el artículo en el que definía el “Modelo Relacional” por primera vez, apareciendo diversas extensiones y modificaciones del mismo. Trataremos de apuntar los elementos básicos del modelo, su evolución, así como el nivel de implantación en el que se encuentra en la actualidad. El estudio de las características del modelo nos permitirá analizar los aspectos del mismo que precisan de mayor elaboración con objeto de adaptarlo a las nuevas expectativas surgidas en el campo de las bases de datos.

2.2 Modelo Relacional de Codd. Elementos del Modelo

Partiendo de la propuesta original descrita en [Codd 70] vamos a introducir una definición marco, la cual se irá refinando mediante la inclusión de nuevos elementos en el modelo. En principio un SBDR responde a dos características básicas:

- a) Los datos son percibidos por el usuario como tablas o relaciones.

- b) Los operadores de los que dispone el usuario, (p.e. para consulta), generan como resultado de su aplicación nuevas tablas a partir de las antiguas. Existirán, por ejemplo, un operador para extraer un subconjunto de filas de una tabla dada y otro para extraer un subconjunto de columnas. Esta claro que la combinación de ambos subconjuntos puede ser contemplada como una tabla.

La definición del “Modelo Relacional” se desglosa en tres ámbitos:

1. *Estructura de Datos:*

dominios (valores atómicos), relaciones n-arias:
atributos, tuplas, claves primarias.

2. *Integridad de los Datos*

- (a) las claves primarias no pueden tomar valores nulos
- (b) las claves externas tienen que tener valores adoptados por las claves primarias a las que están ligadas (o tomar valor nulo)

3. *Manipulación de los Datos*

operaciones del álgebra relacional (o sus equivalentes en el cálculo):

unión, intersección, diferencia, producto, restricción, proyección, reunión, división

asignación relacional

A continuación detallaremos los anteriores elementos.

2.2.1 Estructura de Datos

La estructura de los datos viene caracterizada por el concepto de relación. Para precisar dicho concepto es necesaria la introducción de algunos términos que también forman parte de la definición de la sección estructural del “Modelo Relacional”:

- *DOMINIO*: Es el conjunto de todos los valores posibles que puede tomar un atributo. Una característica inicial que se le exigía a los valores del dominio era la atomicidad, en el sentido de que no exista una descomposición de los mismos que aporte significado. No obstante esta premisa se ha relajado con la evolución del modelo, y actualmente se encuentran en la literatura muchas matizaciones de la misma. Un dominio puede llevar asociado un conjunto de operadores específicos del mismo.
- *ATRIBUTO*: Es la instanciación de un *DOMINIO* en una *RELACIÓN*. Se corresponde con lo que habitualmente denominamos campo o columna.
- *TUPLA*: Es cada una de las filas o registros de la *RELACIÓN*.
- *VALOR DE DOMINIO*: Es la instanciación de uno de los posibles valores que constituyen el *DOMINIO*. Su localización viene determinada por la intersección de una *TUPLA* y un *ATRIBUTO*.
- *RELACIÓN*: O tabla, es el elemento principal del modelo. Una relación R sobre una colección de dominios D_1, D_2, \dots, D_n , (no necesariamente distintos), consta de dos partes, *cabecera* y *cuerpo*.

- La *cabecera* consiste en un conjunto fijo de pares atributo-dominio,

$$\{(A_1 : D_1), (A_2 : D_2), \dots, (A_n : D_n)\}$$

donde cada atributo se corresponde exactamente con el dominio subyacente $D_j (j = 1, 2, \dots, n)$

- El *cuerpo* consta de un conjunto de *tuplas*, donde cada tupla consiste en un conjunto de pares atributo-valor,

$$\{(A_1 : v_{i1}), (A_2 : v_{i2}), \dots, (A_n : v_{in})\}$$

con $(i = 1, 2, \dots, m)$, siendo m el número de tuplas que contiene el conjunto). Para cada atributo A_j existe un par atributo-valor $(A_j : v_{ij})$. Para un par atributo-valor dado $(A_j : v_{ij})$, v_{ij} es un valor concreto perteneciente al dominio D_j asociado al atributo A_j (es lo que habíamos denominado *valor*

de dominio). Los valores m y n se denominan *cardinalidad* y *grado* de la relación R , respectivamente. Mientras que el *grado* se mantiene constante en el tiempo, la *cardinalidad* varía con este.

- **CLAVE PRIMARIA:** Es un conjunto de atributos que identifican unívocamente cada tupla de una relación. Pueden existir varios de estos conjuntos para una relación dada, pero solamente se seleccionará uno de estos como **CLAVE PRIMARIA**, quedando los restantes como **CLAVES ALTERNATIVAS**.

Propiedades de las Relaciones

1. No hay tuplas duplicadas. Siempre deberá existir una clave primaria.
2. Los atributos no están ordenados. Esta propiedad proviene del hecho de que la *cabecera* de una relación es un conjunto matemático.
3. Las tuplas, así mismo, no se encuentran ordenadas. También esta propiedad se apoya en que la definición del *cuerpo* de una relación se corresponde con un conjunto matemático.
4. Los valores de los atributos son atómicos, en el sentido en el que apuntábamos cuando introducíamos el concepto de dominio. Una relación que cumpla esta propiedad se dice

que esta **NORMALIZADA**. Aunque a primera vista esta propiedad pueda suponer una fuerte restricción a la representación de imprecisión, se pueden hacer lecturas de la misma que posibiliten la representación de este tipo de información sin ocasionar una pérdida de validez del modelo. En los capítulos 3 y 4 se prestará una atención especial al

estudio del tratamiento que los diferentes modelos proporcionan a esta propiedad.

2.2.2 Integridad de los Datos

Una base de datos consiste en una configuración de datos que se supone que representan una porción del mundo real. Ningún modelo de base de datos puede garantizar que esa representación se corresponda con la realidad en todo momento, esto supondría, entre otras cosas, que la base de datos tendría que poseer un conocimiento, no solo

sobre los datos, sino también sobre su significado. Lo que si puede y debe hacer un SBDR es no permitir que se introduzca información que no pueda ser identificada, ni que se haga referencia en un lugar de la base de datos a información que no exista en la misma. Estos son los enunciados informales de las reglas de *Identidad* y *Referencial* respectivamente. En forma más precisa pueden ser formulados como sigue:

- *Regla de Identidad.* Ningún componente de la clave primaria de una relación base puede aceptar un valor nulo.

Donde nulo significa que la información no se encuentra por alguna razón (p.e., que la propiedad no sea aplicable o porque el valor sea desconocido).

- *Regla Referencial.* La base de datos no puede contener valores para la clave externa que no hallen correspondencia con los adoptados por la clave primaria a que hacen referencia.

La *clave externa* es un atributo o conjunto de atributos de una relación que, para cada valor adoptado por los mismos, identifican unívocamente una tupla en otra relación. Esto implica que los atributos que componen la clave externa en la relación de partida han de tener correspondencia con los que conforman la clave primaria en la relación referenciada.

En el punto anterior hemos introducido la definición de *clave primaria*. La necesidad de que exista una *clave primaria* para cada relación estriba en que es la única forma de acceder de forma unívoca a cada tupla de la misma.

2.2.3 Manipulación de los Datos

La manipulación de los datos contenidos en un SBDR se realiza a través de dos lenguajes formales que son equivalentes entre sí. *El Álgebra Relacional* y *El Cálculo Relacional*. El primero proporciona un conjunto de operadores mediante los cuales especificar la operación a realizar sobre las tablas, mientras que el segundo proporciona una sintaxis con la que expresar aquello que se desea obtener de las relaciones sin tener que especificar el mecanismo para obtenerlo.

a) Álgebra Relacional.

Dispone de tres tipos de operadores:

- *Operador de asignación.* Asigna el resultado de otras operaciones sobre relaciones a una nueva relación. A través de este operador es posible conservar el resultado de otras operaciones.
- *Operadores tradicionales sobre conjuntos.* Son Unión, Intersección, Diferencia y Producto Cartesiano. Salvo para el Producto Cartesiano, es necesario que las relaciones sobre las que operan, sean compatibles en el sentido de que posean el mismo grado y de que estén definidas sobre el mismo producto cartesiano de dominios.
 - UNIÓN: La unión de dos relaciones A y B es el conjunto de aquellas tuplas que pertenecen a A ó a B ó a ambas. Lo notaremos como $A \cup B$.
 - INTERSECCIÓN: La intersección de dos relaciones A y B da como resultado el conjunto de aquellas tuplas que pertenecen a A y a B. Lo notaremos como $A \cap B$.
 - DIFERENCIA: La diferencia de dos relaciones A y B es el conjunto de tuplas que pertenecen a A y no pertenecen a B. Lo notaremos como $A - B$.
 - PRODUCTO CARTESIANO EXTENDIDO: El producto cartesiano extendido de dos relaciones A y B consiste en todas las posibles combinaciones de pares de tuplas, una de cada relación. Lo notaremos como $A \times B$.
- *Operaciones especiales.* Son Selección, Proyección, Reunión y División:
 - SELECCIÓN: Este operador extrae de una relación A, un subconjunto de tuplas que cumple una condición. Esta condición puede ser: atómica o compuesta. En el primer caso está formada por una sola comparación, mientras que en el segundo caso se trata de un conjunto de condiciones combinadas mediante el uso de operadores lógicos (and, or y not). Lo notaremos como $\sigma_X(A)$.
 - PROYECCIÓN: Produce un subconjunto “vertical” de una relación A dada. Lo notaremos como $\pi_{atributo_1, atributo_2, \dots, atributo_n}(A)$.
 - REUNIÓN: Mediante este operador, a partir de dos relaciones A y B, se construye una relación compuesta de todas las posibles combinaciones de pares de tuplas concatenadas, pertenecientes, una a la relación A y la

otra a la relación B, tales que las tuplas que componen el par satisfacen una condición común. En el caso de que la condición sea la igualdad “=”, se hablará de EQUI-REUNIÓN. Se notará como $A \bowtie B$.

- DIVISIÓN: Dada una relación A con la cabecera (X,Y) y una relación B con cabecera (Y,Z), la división de A entre B dará como resultado una relación C con cabecera (X,Z) y cuyo cuerpo consiste en todas las tuplas (X:x,Z:z), tal que aparece una tupla (X:x,Y:y) en A por todas las tuplas (Y:y,Z:z) que aparecen en B. Se notará como $A \div B$

b) El Cálculo Relacional.

El Cálculo Relacional está basado en una rama de la Lógica Matemática llamada Cálculo de Predicados. En [Kuhns] se encuentra una primera propuesta para utilizar el Cálculo de Predicados como base para formular un lenguaje de bases de datos. Sin embargo es en [Codd 71] donde es propuesto por primera vez el Cálculo Relacional como una forma específica del Cálculo de Predicados. El propio Codd presentó en [Codd 71b] un lenguaje basado en dicho cálculo denominado ALPHA. Este lenguaje, que no llegó a ser implementado, dio origen a una realización comercial del RDBMS INGRES denominada QUEL. Ambos lenguajes están basados en el concepto de variable de tupla estando clasificados por tanto en la categoría del Cálculo Relacional Orientado a Tuplas. En [Lacroix, Pirotte, 77(a)] se propone un cálculo relacional alternativo denominado Cálculo Relacional Orientado a Dominios, en el cual las variables de tuplas son reemplazadas por variables de dominio. En [Lacroix, Pirotte 77(b)] presentan un lenguaje basado en el Cálculo Orientado a Dominios (ILL), otros ejemplos de lenguajes son FQL [Pirotte, Wodom], DEDUCE [Chang 76], y Query-By-Example (QBE) [Zloof]. Este último es más conocido y de él existen varias realizaciones comerciales. En lo que sigue, daremos un breve repaso a las características principales del Cálculo Relacional Orientado a Tuplas.

A través del Cálculo Relacional Orientado a Tuplas podríamos formular una operación en los siguientes términos: “Mostrar

los atributos NOMBRE y CIUDAD para aquellos proveedores que vivan en Londres y que vendan la pieza P='P2'”. Nótese que en esta sentencia hemos indicado al sistema lo que ha de hacer pero no como ha de hacerlo, esta es la diferencia

conceptual entre el Álgebra y el Cálculo Relacional.

Los elementos sobre los que se construye el Cálculo Relacional Orientado a Tuplas son los siguientes:

- *VARIABLE TUPLA*. Consiste en una variable a la que se pueden asignar los valores específicos asumidos por una tupla. Cada variable de tupla viene restringida por un dominio que es preciso especificar al declararla. Al igual que ocurre con el Álgebra Relacional, el Cálculo Relacional lleva asociado un operador de *ASIGNACIÓN*, para almacenar los resultados de las operaciones en las *variables tupla*.
- *CUANTIFICADORES*. Se emplean el cuantificador existencial *EXISTS* y el cuantificador universal *FORALL*, con el primero se recoge el significado de la existencia o no de un valor y el segundo aplica el concepto de la globalidad del universo considerado.
- *EXPRESIONES*. Presentan la sintáxis:

T.A, U.B,...,V.C DONDE f

donde T,U,V... son *variables tupla*, A,B,C... son atributos y f es una fórmula bien formada (WFF). Por ejemplo, dada la *variable tupla* S, y los nombres de atributos de una relación, *NOMBRE*, *CIUDAD*, las siguientes cadenas representarían expresiones correctas para el Cálculo

Relacional:

S.NOMBRE

S.NOMBRE DONDE S.CIUDAD='LONDRES'

S.NOMBRE,S.CIUDAD DONDE EXISTS P(P.NOMBRE=S.NOMBRE AND P.PZ='P2')

El carácter descriptivo del Cálculo Relacional Orientado a Tuplas se pone de manifiesto a través de los ejemplos mostrados. Dicho Cálculo encarna toda la potencialidad manipulativa del Modelo Relacional por lo que cualquier lenguaje que presente elementos manipulativos para los que exista un equivalente en el Cálculo será tan potente como él. Se puede demostrar que el Álgebra y el Cálculo Relacional son equivalentes, o sea, que para cualquier expresión del Álgebra se puede

encontrar una equivalente en el Cálculo, y viceversa. En [Codd 71] puede encontrarse un algoritmo por medio del cual es posible reducir cualquier expresión arbitraria del Cálculo Relacional Orientado a tuplas a una expresión en términos del Álgebra Relacional, así mismo en [Ullman] puede encontrarse una demostración de la otra implicación¹. De acuerdo con esto, cualquier lenguaje para el que se puedan encontrar equivalencias con las cinco primitivas del Álgebra Relacional, (Selección, Proyección, Producto Cartesiano Extendido, Unión y Diferencia), se considerará “*relacionalmente completo*”, es decir, gozará de toda la potencia de expresión que proporciona el Cálculo Relacional.

2.3 Evolución del Modelo Relacional

El Modelo Relacional Clásico ha sufrido múltiples revisiones y actualizaciones para adecuarlo a la evolución de los sistemas de tratamiento de la información. Además, algunas de las carencias o inconvenientes presentes en la definición original han ido siendo subsanadas en sucesivas revisiones. Los esfuerzos realizados en este sentido, tanto por su creador, (E. F. Codd), como por otros autores, (C. J. Date y otros), han dado como resultado la aparición de varias versiones del modelo que el propio Codd clasifica como sigue:

- **RM/V1.** Las siglas constituyen el acrónimo de *Relational Model Version 1*. Esta versión engloba la definición inicial del Modelo [Codd 70], así como el contenido de todos los artículos publicados sobre el Modelo hasta 1979, fecha en que publica un artículo [Codd 79], que contiene lo que se ha dado en llamar la versión extendida del Modelo **RM/T**. Las principales características de esta versión han sido descritas en el apartado anterior.
- **RM/T.** En esta extensión las siglas están extraídas de la denominación en inglés de la revisión, donde la T proviene de Tasmania lugar en el que presentó por primera vez dicha extensión. Con esta revisión el autor inicia una serie de publicaciones en las que actualiza el Modelo y con las que abre una etapa de supervisión

¹Para que esta implicación sea cierta, es preciso asumir la condición de “seguridad” para las expresiones del Cálculo Relacional, esta condición permite restringir la cardinalidad obtenida mediante dichas expresiones, con el objeto de que ninguna de ellas permita obtener una cardinalidad infinita de una relación (que siempre es finita)

sobre las implementaciones comerciales denominadas a sí mismas Relacionales. Además, en esta etapa se producen intensos cruces de opiniones con otros autores en torno a qué aspectos del Modelo deben ser alterados y cómo deben ser llevadas a cabo dichas modificaciones. Algunos de los aspectos que suscitan mayor polémica son los referentes a la extensión de los operadores de manipulación de datos, principalmente en lo que se refiere al tratamiento de información “incompleta” o “ausente”. Relacionado con este problema está el de la alteración del concepto de “Reunión” para dar un soporte adecuado a este tipo de información. El concepto de atomicidad, la posibilidad de actualizar las vistas, la forma de mantener la integridad de la Base de Datos, etc., han sido también objeto de revisión.

Algunos de los artículos más importantes publicados en esta época, aparte del anteriormente mencionado, son:

- [Codd 82] en el que Codd define una vez más el Modelo, propone imponer reglas de integridad adicionales (específicas para la base de datos) con las que se puedan restringir el conjunto de estados o cambio de estados válidos para una base de datos de forma que resulte consistente.
- En [Codd 85], que consta de dos partes, proporcionó una nueva definición del Modelo Relacional constituida por nueve características estructurales, tres características de integridad y dieciocho características para la manipulación de datos. En este trabajo introduce también las famosas Doce Reglas de Codd, con las que, junto con las características anteriores, se pretende evaluar el carácter relacional de un DBMS (ver también [Codd 86b]).
- El artículo [Codd 86a] propone un enfoque basado en la lógica de tres valores para tratar de resolver el problema del tratamiento de la información “incompleta” o “ausente”. Este artículo responde a ciertas críticas vertidas por Date en [Date 86] y es contestado a su vez en [Date 89]. Como veremos en la próxima sección el tratamiento de la información “incompleta” o “ausente” es uno de los puntos del Modelo que más controversia ha suscitado. En la literatura sobre el Modelo Relacional Clásico se encuentran diversas propuestas y contrapropuestas en torno a como abordar la solución del problema. Ejemplos de ello son los trabajos y tesis sostenidas en [Codd 87] donde Codd

amplía su enfoque de lógica de tres valores a uno de lógica de cuatro valores para soportar el concepto de “propiedad no aplicable”. En [Date 86II] se introduce una aproximación denominada de “valores por defecto” para tratar el problema de la información por aportar al sistema, así mismo, apunta la posibilidad de emplear tipos de “valores nulos por defecto” para soportar la semántica que rodea a aquellos datos “ausentes” que representan una “propiedad no aplicable”.

Todas estas y otras aportaciones diseminadas en la literatura han forzado al creador del modelo a realizar un esfuerzo de recopilación para ofrecer lo que ha dado en llamar la Revisión dos del Modelo Relacional.

- **RM/V2.** [Codd 90] cubre la descripción de las dos versiones del Modelo Relacional (**RM/V1** y **RM/V2**). Las características adicionales más importantes soportadas por la revisión 2 son:
 - Un nuevo tratamiento para la información “ausente”. En esta revisión describe su propuesta basada en la lógica de cuatro valores para tratar los datos “ausentes-pero-aplicables” y los datos “ausentes-y-no-aplicables”.
 - Nuevas características para el soporte de todo tipo de restricciones para mantener la integridad del sistema. Mención especial reciben los tipos definidos por el usuario, esta característica incorpora al Modelo Relacional un concepto de creciente implantación en los RDBMS comerciales como es el del “trigger” o disparador el cual proporciona un mecanismo bastante potente para mantener la integridad en una Base de Datos.
 - Un repaso al tema de la actualización de las vistas.
 - Nuevas directrices en torno a las pautas a seguir para el desarrollo de productos RDBMS y de lenguajes relacionales.
 - Un detallado repaso al que debería ser el contenido del catálogo del RDBMS.
 - Algunos aspectos en torno a los sistemas de bases de datos distribuidas.
 - Un acopio de nuevas leyes que regirán los destinos de los que hayan de denominarse Sistemas Relacionales.

En [Codd 90] se enfatiza además, algunos de los aspectos semánticos del Modelo Relacional tales como:

- Los dominios, las claves primarias, y las claves externas.
- Los valores redundantes se permiten en las columnas de una relación pero insiste en el tema de que las tuplas redundantes estén prohibidas.
- El tratamiento sistemático de la información “ausente” independientemente del tipo de dato a que “pertenezca” esa ausencia.

A lo largo del apartado 2.2 se han descrito los elementos básicos del Modelo Relacional (la mayoría de ellos recogidos del primer enunciado del mismo). En este apartado hemos hecho un recorrido por los aspectos abordados con mayor o menor éxito por las sucesivas revisiones del Modelo. Cae fuera del ámbito de esta introducción al Modelo Relacional una descripción exhaustiva de dichas revisiones. Esto es así, debido por un lado, a la extensión de las mismas (la revisión dos consta de nada menos que 333 características) y, por otro, a la falta de asentamiento y de consenso sobre alguno de sus aspectos (tratamiento de la información “ausente”, extensión del conjunto de operadores, posibilidad de actualizar las vistas, etc.).

2.4 El Problema del Tratamiento de la Información Incompleta

El tratamiento de la información “incompleta” responde a la necesidad de abordar situaciones reales que se presentan en el diseño, mantenimiento y explotación de las bases de datos. Con el término información “incompleta” aglutinamos todos aquellos casos en los que no podemos asumir un valor para un dato (valor “ausente”) o, no podemos asumirlo en forma precisa, sea cual fuere el motivo por el que no poseemos dicho valor. El problema que plantea el tratamiento de este tipo de información radica en que altera sustancialmente algunos de los aspectos básicos sobre los que se construye el Modelo Relacional. Dichos aspectos conciernen principalmente a la representación de este tipo de información en la estructura de una relación y, sobre todo, a los aspectos relacionados con su manipulación. El último aspecto es el que representa un mayor problema, puesto que todas las propuestas que pretendan proporcionar mecanismos para manipular este tipo de información, se verá abocada inevitablemente a modificar de forma sustancial los operadores algebraicos del Modelo Relacional. En particular, la Selección y la Reunión son conceptos fuertemente afectados por tal revisión. Por otro

lado, como veremos a continuación, cuando se incorpora el tratamiento de información incompleta, ninguna propuesta garantiza la continuidad de las buenas propiedades que presentaba el Modelo Relacional sin este tipo de información.

Estudiaremos en este apartado cuales son las principales propuestas “clásicas” para abordar el problema de la representación y tratamiento de la información “ausente”. Como alternativa mostraremos en el próximo capítulo algunos de los recursos empleados usando la Teoría de Conjuntos Difusos y la Teoría de la Posibilidad para dar una respuesta elegante a este problema. Dicha aproximación trata la información “ausente” como un caso particular del problema de representación y manipulación de información “imprecisa” en Sistemas de Bases de Datos Relacionales.

2.4.1 La Aproximación de Codd

Una descripción detallada de esta aproximación puede encontrarse en [Codd 86a], en [Codd 87] y en [Codd 90]. Codd aborda el tratamiento de la información “ausente” desde la siguiente perspectiva semántica. Empieza estableciendo una distinción entre dos aspectos concernientes a la naturaleza de la información “ausente”. Estos aspectos son:

1. Que tipo de información que se encuentra “ausente”.
2. Cual es el motivo por el que se encuentra “ausente”.

El primer aspecto trata de si la información “ausente” es un registro, un atributo, o un conjunto de atributos. Dicha “ausencia” debe ser interpretada, por tanto, dentro de un contexto estructural.

El segundo aspecto plantea la cuestión de si la información no aparece porque se desconoce su valor en este momento o, porque dicho valor no es aplicable. Para modelar este aspecto aboga por una representación basada en el empleo de dos tipos de “marcas” sobre los atributos susceptibles de admitir tratamiento de información “ausente”. Para indicar que un *db-value*, (notación abreviada que emplea para referirse a un valor elemental de una base de datos), se encuentra “ausente”, en el sentido de que se desconoce su valor pero que es aplicable, emplea la “marca de valor ausente pero aplicable”, notada por brevedad como “A-marca”. Para representar que el motivo por el que se encuentra “ausente” un *db-value* es el de que “no es aplicable”, propone el

empleo de “I-marcas” que significa que la información se encuentra “ausente” y que además lo está por que “no es aplicable”.

Un ejemplo del primer tipo de información puede ser un atributo que recoja la fecha de nacimiento de cada empleado en una relación que almacene información sobre los mismos. Si, para un empleado dado, desconocemos su fecha de nacimiento, habremos de señalar esta circunstancia mediante el empleo de una “A-marca” para el atributo en la tupla correspondiente. El empleo del segundo tipo de marcas se puede ilustrar mediante un ejemplo de atributo que recoja información sobre la comisión por ventas que percibe cada empleado, si poseemos información en la base de datos sobre cual es el tipo de actividad que desarrolla cada empleado, está claro que no podremos proporcionar un valor para el atributo comisión por ventas para un empleado que no se dedique a ventas, esta “ausencia” de información la marcaremos mediante una “I-marca” en el atributo comisión por ventas en la tupla correspondiente.

Codd justifica el empleo del término “marca” en vez de los términos “valor”, “nulos” o “valores nulos” por el hecho de que considera las “marcas” como “marcas” para expresar una circunstancia y no como valores. Además, existen dos tipos de marcas frente a uno de “nulo”. Por otro lado sugiere una posible confusión en cuanto a cual es el significado de la palabra “nulo” en los lenguajes anfitriones.

Las “marcas” modelan la circunstancia por la que la información se encuentra “ausente”, por tanto no dependen del tipo de información que estemos considerando y por ello no son consideradas como valores de dominio.

Una vez que ha establecido la forma en que representa la información “ausente” precisa establecer de que forma han de operar sobre las “marcas” los principales conceptos relativos a la manipulación de datos, es decir, los operadores relacionales ($=, \neq, <, \leq, >, y \geq$), los operadores aritméticos y los operadores lógicos. Un análisis exhaustivo de como se comporta el Álgebra Relacional con el esquema de las “marcas” escapa al propósito de este estudio. Haremos hincapié sobre los conceptos básicos que soportan esta aproximación.

• Aplicación de la Igualdad

Se plantea el problema de si dos “marcas” son iguales o no. Distingue entre dos tipos de igualdad: *semántica* y *simbólica* o *formal*. Para el primer tipo de igualdad ha de tener en cuenta si el tipo de las “marcas” es el mismo o no, caso este último que

conduce a afirmar que la expresión “A-marca” = “I-marca” se evalúa como FALSO. Otro problema es evaluar igualdades en las que las “marcas” son del mismo tipo. Por ejemplo, si tenemos la expresión “A-marca” = “A-marca”, simbólicamente son iguales no obstante, semánticamente, la interpretación por la que se opta es la de que no sabemos si son iguales o no, por tanto, no podemos evaluar una expresión de igualdad entre dos “A-marcas” como VERDADERO ni como FALSO, dicha expresión se evalúa mediante un tercer valor lógico denominado MAYBE (“puede ser”) cuyo significado es el de que “puede que sea cierta la igualdad o puede que sea falsa”.

Esto conduce al empleo de una lógica trivaluada, que ha de extenderse a una lógica de cuatro valores para soportar el tratamiento de las “I-marcas”.

• Lógica Trivaluada y Tetravaluada

Al principio Codd proporcionó tratamiento para la información “ausente” sin considerar si era “aplicable” o “no aplicable”. Para este desarrollo inicial hizo uso de la lógica trivaluada cuya expresión se muestra en la tabla 2.1 en la que el estado m significa “quizá” (MAYBE originariamente). Para manipular las situaciones representadas por las “A-marcas” y las “I-marcas” hubo de emplear la lógica tetravaluada que recoge la tabla 2.2.

AND	T	m	F	OR	T	m	F	NOT	
T	T	m	F	T	T	T	T	T	F
m	m	m	F	m	T	m	m	m	m
F	F	F	F	F	T	m	F	F	T

Tabla 2.1: Tablas de verdad para la lógica trivaluada

• Efectos en el Álgebra Relacional para la Lógica Trivaluada

Los operadores tradicionales del Álgebra Relacional (Unión, Diferencia, Producto Cartesiano, Proyección y Theta-selección) alteran su comportamiento cuando se aplica el concepto de valor “nulo”.

AND	T	A	I	F	OR	T	A	I	F	NOT	
T	T	A	I	F	T	T	T	T	T	T	F
A	A	A	I	F	A	T	A	A	A	A	A
I	I	I	I	F	I	T	A	I	F	I	I
F	F	F	F	F	F	T	A	F	F	F	T

Tabla 2.2: Tablas de verdad para la lógica tetraevaluada

Para el tratamiento específico de “nulos” introduce la función `IS_NULL` la cual se aplica sobre un escalar arbitrario y devuelve *verdadero* si éste es nulo y *falso* en cualquier otro caso. Mediante el empleo de esta función construye dos nuevos operadores algebraicos: `MAYBE_SELECT` y `MAYBE_JOIN`. El primero es una modalidad del operador `THETA_SELECT`, con el que unicamente se recuperan las tuplas que presentan un valor nulo para el atributo consultado. El operador `MAYBE_JOIN` reúne aquellas tuplas en las que existe un valor nulo para alguno de los atributos implicados en la reunión. Los mencionados operadores, mediante el empleo de `IS_NULL` y de `THETA_SELECT`, quedan expresados como sigue:

$$\begin{aligned}
 \text{MAYBE_SELECT (R, Ai)} &= \text{THETA_SELECT (R, IS_NULL(Ai))} \\
 \text{MAYBE_JOIN (R, S, Ai, Bj)} \\
 &= \text{THETA_SELECT (PRODUCT (R, S),} \\
 &\quad \text{(IS_NULL (RAi) OR IS_NULL (SBj)))}
 \end{aligned}$$

Con estos operadores pueden definirse las siguientes variedades de REUNIÓN:

- La `THETA_REUNIÓN` normal. Esta variedad de REUNIÓN devuelve unicamente aquellas tuplas cuya condición de REUNIÓN se evalúa como *cierta*, es decir, se eliminan tanto las que no cumplen la condición, como aquellas que se evalúan con “nulo” para alguno de los atributos implicados (ya que en este caso la condición de REUNIÓN se evalúa como *“MAYBE”*).
- La `THETA_REUNIÓN` externa. Con esta modalidad se trata de conservar alguna información adicional tras la operación de REUNIÓN. Para ello, añade al resultado de aplicar una `THETA_REUNIÓN` normal, una tupla adicional por

cada tupla en cada una de las relaciones implicadas sobre las que no ha habido concordancia. Dicha tupla es extendida con valores “nulos” para cada uno de los atributos para los que no poseía valor en la relación original.

Ejemplo 2.1 La tabla 2.3 ilustra cual es el resultado de aplicar un “*equireunión externa*” sobre dos relaciones que contienen una lista de proveedores P(S#, PNOMB, ESTADO, CIUDAD) y la otra de componentes C(P#, CNOMB, COLOR, PESO, CIUDAD). Los atributos sobre los que se realiza el “*equireunión externa*” son P.CIUDAD y C.CIUDAD. (El símbolo “?” representa la presencia de un valor “nulo”).

P#	CIUDAD	C#	CIUDAD	PP#	PCIUDAD	CC#	CCIUDAD
P7	Londres	C7	Londres	P7	Londres	C7	Londres
P8	Paris	C8	Madrid	P8	Paris	?	?
P9	?	C9	?	P9	?	?	?
				?	?	C8	Madrid
				?	?	C9	?

Tabla 2.3: Relaciones de Proveedores P, de Componentes C, y equireunión externa de P y C sobre los atributos PCIUDAD y CCIUDAD

- THETA_REUNIÓN externa izquierda y THETA_REUNIÓN externa derecha. Se obtiene la primera en la forma en que se obtiene la THETA_REUNIÓN externa, en la que unicamente se extienden la tuplas sin concordar de la relación a la izquierda de la THETA_REUNIÓN. La segunda realiza esa extensión para las tuplas sin concordancia de la relación situada a la derecha.
- REUNIÓN natural externa. Este tipo de REUNIÓN se obtiene de la equireunión externa en la que se sustituyen los atributos implicados en dicha REUNIÓN, por un atributo genérico cuyos valores se calculan aplicando los siguientes criterios sobre cada par de valores provenientes de los mencionados atributos:

1. Si los dos valores son “no-nulos” e iguales, la tupla en cuestión se recupera mostrando uno de dichos valores para el atributo genérico.
2. Si los dos valores confrontados en una tupla dada son nulos (“?”), entonces ese atributo toma el valor “?” para dicha tupla.
3. Si uno de los valores es “nulo” (“?”) y el otro no, entonces el atributo genérico almacena el valor “no-nulo” para la tupla considerada.

La tabla 2.4 muestra como se realiza la REUNIÓN natural externa sobre el ejemplo 2.1.

P#	CIUDAD	C#
P7	Londres	C7
P8	Paris	?
P9	?	?
?	Madrid	C8
?	?	C9

Tabla 2.4: Reunión natural externa sobre P y C

• El Cualificador MAYBE en la Lógica Tetraevaluada

Hemos visto que para poder tratar la información “ausente”, Codd acude al recurso del cualificador MAYBE el cual puede ser aplicado a cualquier expresión lógica, (a la totalidad o a una parte de la misma), para seleccionar unicamente aquellos valores, (tuplas), que satisfacen la expresión, (o la fracción de expresión sujeta a tal cualificación), con un valor de verdad igual a m . Esto es así en el caso de que estemos considerando la lógica trivaluada vista anteriormente (ver tabla 2.1).

Para la aproximación basada en la lógica de cuatro valores, el modelo proporciona una colección de tres cualificadores:

- El cualificador MAYBE, cuya aplicación en los términos expuestos anteriormente, proporciona aquellas tuplas que satisfacen la expresión, (o la fracción de expresión sujeta a tal cualificación), con un valor de verdad igual a “A” o igual a “I”.

- El cualificador `MAYBE_A`, que opera en igual forma sólo que las únicas tuplas proporcionadas son las que satisfacen la expresión con un valor de verdad igual a “A”.
- El cualificador `MAYBE_I`, que proporciona las que evalúan la expresión con un valor de verdad “I”.

• Otros Aspectos de la Aproximación

Codd adapta el empleo de la aritmética al uso de las “marcas”. En este sentido establece una jerarquía para los datos en la que sitúa las “I-marcas” en el nivel superior, seguidas en un segundo nivel por las “A-marcas”, para terminar a nivel más bajo con los “*db-values*”. Según esto, la combinación, aritmética o de otra índole, de dos elementos dará como resultado otro elemento cuyo nivel será el de aquel elemento que posea mayor nivel de los dos.

Existen otra serie de consideraciones en torno a la aproximación, que pasamos a resumir:

- No se permiten “marcas” en los atributos que constituyan la “clave primaria”. Dicha característica debe ser forzada por el sistema.
- Prohíbe las “I-marcas” en los atributos que constituyan una “clave externa”. Lo contrario contradice el concepto de “clave externa”.
- Desaconseja que los atributos que componen una “clave externa” puedan contener “A-marcas”.
- El DBMS debe rechazar aquellas tuplas de una relación que no contengan otra cosa que “A-marcas” y/o “I-marcas”.
- Para las operaciones de manipulación de datos ha de tenerse en cuenta la forma en que actúan los operadores relacionales con las “marcas”. De manera que, salvo que aparezca algún cualificador “MAYBE” en la operación, las expresiones ignorarán aquellas tuplas que intervengan con “marcas” en la evaluación de la expresión.

2.4.2 El Esquema de los “Valores por Defecto”

Date presentó en 1982 una aproximación para el tratamiento de valores nulos (dicha aproximación se recoge en [Date 86]). Parte de la opinión de que el problema del tratamiento de valores nulos no se encuentra bien definido en la actualidad, y de que, por tanto, de momento no debería incorporarse esta característica al Modelo Relacional. Partiendo de esta premisa, presenta una alternativa basada en el concepto que denomina de “valores por defecto”. Los aspectos básicos que sustentan esta aproximación pueden esquematizarse como sigue:

- En la declaración de cada dominio se designa un valor de dicho dominio como “valor por defecto”. Por ejemplo:

```
DCL LOCATION DOMAIN CHARACTER(15) DEFAULT ('???');
```

- En la declaración de cada atributo de una relación real existe una cláusula opcional `NODEFAULT`. Esta cláusula establece si dicho atributo, tiene o no, un “valor por defecto”. Es evidente que dicha especificación es asumida para cada uno de los atributos que componen la clave primaria. Para los atributos que constituyen las claves candidatas y para el resto de los atributos, esta cláusula establece si heredan, o no, el “valor por defecto” asumido por el correspondiente dominio subyacente.
- Cuando se inserta una nueva tupla en una relación,
 - a) el usuario tiene que proporcionar un valor para cada atributo que no tenga un “valor por defecto”;
 - b) para el resto de los atributos, si el usuario no proporciona ningún valor, el sistema habrá de suministrar el “valor por defecto” correspondiente a dicho atributo.
- El sistema deberá contar con una función `DEFAULT(R.A)`, donde `R` es una relación y `A` es un atributo de esa relación. Dicha función devuelve el “valor por defecto” asociado a `R.A`, devolverá un error si tal valor no existe.
- Las reglas de integridad de identidad y referencial han de ser revisadas en el siguiente sentido: *Reg. Identidad*: Ningún componente de la clave primaria puede

aceptar “valores por defecto”. *Reg. Referencial:* Todo dato que no sea “valor por defecto” de una clave externa tiene que hallar correspondencia con algún valor existente en la clave primaria.

- En las funciones de agregación tales como `AVG`, el usuario, si así lo desea, tendrá que especificar explícitamente un predicado para excluir los “valores por defecto”. Por ejemplo:

```
SELECT AVG (SP.QTY)
FROM SP
WHERE QTY != DEFAULT (SP.QTY)
```

- Además las funciones de agregación habrán de ser extendidas mediante un argumento opcional adicional para especificar el valor que devolverán si el primer argumento se evalúa como conjunto vacío. Por ejemplo:

```
SELECT DEPT#
FROM DEPT
WHERE 50000 <
      (SELECT AVG(SALARIO,0)
      FROM EMP
      WHERE EMP.DEPT# = DEPT.DEPT#)
```

Si el segundo argumento se omite y el conjunto es vacío, se asume el “valor por defecto” asociado al primer argumento, ocurrirá un error si no existe “valor por defecto” para dicho argumento y se obtiene un conjunto vacío.

- Los operadores `MAYBE_SELECT` y `MAYBE_JOIN` no son necesarios. Las operaciones de reunión externa son necesarias, pero son redefinidas para generar “valores por defecto”. Por desgracia la equireunión externa no puede ser generalizada adecuadamente al caso en intervienen n relaciones con $n > 2$. Tampoco resuelve el hecho de que la reunión natural externa es una proyección de la equireunión externa.

- Para ciertos atributos puede darse el caso de que cualquier valor del dominio subyacente sea un posible valor de dominio, con lo que se presenta la dificultad a la hora de determinar que valor emplear como “valor por defecto”. Dichos atributos deben ser manejados explícitamente mediante campos controlados por el usuario.

2.4.3 Críticas a las Aproximaciones Clásicas

En este subapartado vamos a examinar algunas de las críticas suscitadas las dos aproximaciones descritas. Algunas de estas críticas son autocríticas, otras son fruto del cruce de opiniones entre los autores de las propuestas y, por último, plantearemos algunas objeciones particulares.

Para el análisis de las objeciones suscitadas por las aproximaciones estudiadas, realizaremos una clasificación de las mismas atendiendo al aspecto sometido a debate.

• Representación de la Información “Ausente”

En este punto los modelos estudiados recogen tres esquemas de representación (uno en la aproximación de Date y dos en la de Codd). La aproximación de “valores por defecto”, (DV en lo sucesivo), representa información “ausente” mediante el empleo de unos valores específicos de cada dominio que son asumidos como “valores por defecto” en tanto que no se proporcionen valores concretos. La aproximación de Codd trata la “ausencia” de información por no proporcionada o por no aplicable. La diferencia en la representación radica en el hecho de que, en la aproximación DV el concepto de “nulo” se incluye en una extensión del dominio, (con un significado especial), con el que se construyen las relaciones. Por contra, en el esquema de Codd la “ausencia” no es un valor del dominio considerado, si no un estado de la información, que representa mediante el empleo de “marcas”.

El esquema de DV presenta el problema de que modela la información “ausente” mediante una extensión del dominio que luego integra en un concepto del Modelo Relacional modificado al efecto sin una perspectiva global. Esta concepción genera problemas a la hora de implementar efectivamente el concepto de “nulo” propuesto. No está muy claro que representación adoptar para un atributo con objeto de distinguir los “valores reales” del valor “nulo”.

Por otra parte Codd basa su esquema en una solución puntual que más bien parece ir dirigida a resolver los problemas de implementación que a reconsiderar formalmente el Modelo Relacional para tratar de forma elegante el problema.

Gran parte de los problemas que suscitan ambas aproximaciones proceden de la concepción semántica del problema adoptada, la cual conduce a una representación incorrecta de la información “ausente”. Como veremos en el próximo capítulo, un enfoque adecuado para solucionar el problema, pasaría por considerar una representación de la información en la que se integrara el valor (o valores) asumidos por la misma, con el grado de conocimiento que podemos aportar al sistema sobre dicha información. Cualquier aproximación que ignore esta doble concepción de la información estará abocada a un tratamiento parcial del problema.

Hecha esta precisión, la aproximación de Codd parece más adecuada para alcanzar una representación parcial del problema. Esto es así porque al menos distingue los dos aspectos que confluyen en el tratamiento de la información “ausente”. Por un lado está la información representada en forma de valor y, por otro, el estado de “conocimiento” en que esa información se encuentra en la base de datos: “presente en forma de valor”, “ausente pero aplicable” y “ausente pero no aplicable”. El primer estado queda representado por la ausencia de “marcas” relativas al atributo, el segundo por la presencia de una “A-marca” en el lugar que cada implementación habilite para ello y, el último, será notado en el mismo lugar por una “I-marca”. En cualquier caso, no mezcla los valores que pueda adoptar la información con los estados de “conocimiento” en que se pueda encontrar.

• El Problema de la Normalización

Un análisis del comportamiento del concepto de la “normalización” con respecto al empleo de “nulos”, revela ciertas dificultades para dar una solución adecuada al problema. Date plantea el problema sobre el siguiente ejemplo:

Ejemplo 2.2 *Sea $R(A, B, C)$ una relación que satisface la dependencia funcional $A \rightarrow B$ y en la que A no es una “clave primaria”.*

Reflexiona sobre que debe ocurrir si tenemos una tupla cuyo contenido es $(?, b1, c1)$ y tratamos de insertar otra cuyo contenido es $(?, b2, c2)$, donde “?” representa un valor

desconocido y b_1 y b_2 son distintos. Date opina que, para satisfacer la dependencia funcional, debe ocurrir que, o consideramos que los dos nulos son distintos, caso que va en contra de su esquema de DV, o rechazamos la segunda tupla por violar la dependencia funcional, o redefinimos el concepto de dependencia funcional.

Por su parte Codd alega que con su aproximación el problema está resuelto. En dicha aproximación adopta la tercera posibilidad sugerida por Date en el sentido de proporcionar una redefinición del concepto de dependencia funcional. Dicha definición sostiene que si un componente de una tupla se encuentra “ausente”, la dependencia funcional ($A \rightarrow B$) no es forzada por el DBMS hasta que se produzca un intento de reemplazar dicho “nulo” (o “marca”) por un valor concreto. Por otra parte, argumenta que, como según su esquema las “marcas” no son “valores” de la base de datos, las reglas de las dependencias funcional y multivaluada no son aplicables a las mismas, y por tanto, no se producen a su juicio problemas en el proceso de normalización. Como el esquema de DV considera las marcas como valores de la base de datos, el proceso de normalización no funciona bien.

• Anomalías Semánticas y Lógicas

En este apartado analizamos algunas de las objeciones vertidas en torno al carácter poco intuitivo que presentan las soluciones a algunos de los problemas propuestos.

Considérese la relación de proveedores mostrada en la tabla 2.5. Pensemos en el resultado de las dos consultas:

Q1 Encontrar los proveedores que están en Londres.

Q2 Encontrar los proveedores que no están en Londres.

Intuitivamente podríamos pensar que el resultado de esas dos consultas debería devolvernos todos los proveedores. No obstante, puesto que la aproximación de Codd solo considera las tuplas que satisfagan la condición con un valor de verdad VERDADERO, el resultado de ambas consultas no recupera a los proveedores Dupont ni a Grid. Para que estén todos habría que incluir la consulta:

Q3 Encontrar los proveedores que “quizá” estén en Londres.

Otro problema que se plantea por el uso de las lógicas multivaluadas es el tratamiento que proporcionan a las tautologías. Supongamos el ejemplo de consulta:

P#	NOMBRE	CIUDAD	...
P1	Jones	Londres	...
P2	Smith	Bristol	...
P3	Dupont	?	...
P4	Eiffel	Paris	...
P5	Grid	?	...

Tabla 2.5: Relación de Proveedores P

Ejemplo 2.3 *Encuentrense aquellos empleados cuyo año de nacimiento sea 1940, anterior a 1940 o posterior a 1940.*

Obviamente la respuesta a esta consulta debería estar constituida por todos los empleados. Sin embargo, si para algún empleado tenemos un “nulo” en el atributo año de nacimiento el resultado de la comparación correspondiente será “desconocido” y por tanto no se recuperará esa tupla. Codd encuentra dos soluciones al problema:

1. Aconsejar a los usuarios que no empleen tautologías en sus sentencias, lo cual no siempre es realizable, o
2. Dejar al DBMS que examine las condiciones para determinar si una condición es una tautología y actuar en consecuencia.

Otros problemas de tipo lógico son:

- La definición de duplicados. Si la comparación $x = x$ se evalúa como desconocido cuando x es “nulo”, no se sostiene que se considere x como duplicado de sí mismo cuando se realiza, por ejemplo, una proyección. Codd responde a la objeción alegando que en el concepto de supresión de duplicados emplea la igualdad simbólica.
- No se garantiza que la Reunión natural consigo mismo sea igual a la relación original.

Conclusiones

Como resultado del estudio anterior podemos adherirnos a la opinión manifestada

por algunos autores en la que se recoge la impresión de que el tratamiento de información “ausente” no se encuentra lo suficientemente maduro como para incluirlo en el Modelo Relacional. La postura del autor en este sentido es que el enfoque teórico adoptado para tratar el problema constituye un callejón sin salida y que hay que abordar el cuestión desde la perspectiva general del problema del tratamiento de la información “incompleta” y que el marco adecuado para ello es el de la Teoría de Conjuntos Difusos y la Teoría de la Posibilidad. En el próximo capítulo estudiaremos algunas de las líneas desarrolladas en este sentido.

2.5 Sistemas Relacionales Comerciales

En este apartado vamos a dar un breve repaso a algunos de los SBDR comerciales más importantes, surgidos con la intención de resolver el problema del tratamiento de los datos en el ámbito del modelo relacional. Este repaso se hace con objeto de mostrar cual es la implantación efectiva y los mecanismos adoptados para hacerla. Los sistemas “relacionales” comerciales aparecidos a los largo de los últimos años caen, según el carácter relacional que presenten, dentro de alguna de las categorías siguientes:

- Sistemas que soportan una estructura de datos tabular, pero donde los operadores a nivel de conjuntos no pueden ser llamados tabulares. Estos sistemas no son sistemas relacionales. En algunos textos aparecen como “*semirrelacionales*”. Los sistemas denominados de “listas invertidas” caen dentro de esta categoría. Un ejemplo comercial de los mismos lo constituye el DATACOM/DB de IBM que corre en “mainframes” MVS, VM o VSE.
- Sistemas que soportan tablas (sólo), y los operadores de selección, de proyección y la reunión, pero no otros operadores relacionales. Estos sistemas corresponden a la categoría denominada “*minimamente relacional*”. Algunos ejemplos de los mismos lo constituyen gran parte de los sistemas desarrollados para microcomputadores, tales como DBASE III.
- Sistemas que soportan tablas (sólo), y todos los operadores del Álgebra Relacional. Estos sistemas se denominan “*relacionalmente completos*”. La mayoría de los sistemas actuales, pretendidamente relacionales, caen dentro de esta categoría. Tal es el caso de DB2, SQL/DS, INGRES, ORACLE, Rdb/VMS, etc.

En la próxima sección daremos un repaso a las características relacionales de un sistema particular como es ORACLE.

- Sistemas que soportan todos los aspectos del modelo, incluyendo los relativos al dominio y a las dos reglas de integridad, estos sistemas se dicen sistemas “*completamente relacionales*” según la definición de principios de los 80,s. En la actualidad empiezan a aparecer sistemas que satisfacen dichos requisitos como es el caso de la versión 7 de Oracle.

Las conclusiones que podemos obtener de esta clasificación y del grado de implantación comercial de las diferentes categorías son:

- La implantación comercial de los esquemas teóricos propuestos, presenta un retraso con respecto a los mismos por motivos de dificultad de implementación, de operatividad de los mismos, de dependencia con la evolución de la capacidad de proceso de los soportes hardware, ya que, por ejemplo, el soporte de las reglas de integridad supone un incremento en el procesado requerido por una operación realizada sobre una base de datos, o simplemente por motivos de índole puramente comercial.
- El contacto con la problemática real a la se que enfrentan los prototipos experimentales (que implementan los modelos teóricos), da como resultado, además de la revisión estructural de los mismos, la modificación y actualización de los modelos teóricos existentes tendente a proporcionar un marco adecuado para la solución de los problemas planteados por la experiencia real. Así el modelo relacional ha sufrido varias revisiones con respecto a la propuesta inicial de Codd.

2.5.1 El SBDR Oracle

Como ejemplo de SBDR comercial vamos a comentar algunos aspectos del gestor de bases de datos ORACLE de Oracle Corporation. Éste fue el primer fabricante en introducir un producto relacional en el mercado, adelantándose a la propia IBM. Actualmente es el mayor vendedor de SBDR independiente. En este subapartado daremos un breve repaso a la versión 6 de dicho RDBMS, el motivo de ello es que el desarrollo que se muestra en el Apéndice A está construido, por razones de disponibilidad, sobre dicha versión. En el momento de terminar esta memoria se encontraba en la calle la

versión 7 del gestor, sin embargo, a pesar de las mejoras introducidas, el aspecto más importante del modelo relacional introducido es el soporte para las reglas de integridad.

Los elementos relacionales que proporciona ORACLE:

1. Estructura de Datos

Estos se organizan a través de tablas y en general de acuerdo con el modelo relacional. Las cabeceras de atributo, que en Oracle se denominan columnas, no presentan un orden de acuerdo con el modelo relacional y respondiendo a la noción de conjunto que subyace a estas. Las columnas pueden estar definidas sobre los tipos de dominio que a continuación se relacionan:

- **Carácter.** Una columna definida sobre un dominio de este tipo, deberá especificar la longitud máxima de la cadena de caracteres que podrá contener. La longitud máxima para una columna definida sobre este dominio dependerá del tipo de la columna, oscilando entre los 255 caracteres de máximo para el tipo CHAR y los 65535 del tipo LONG.
- **Numérico.** Los diferentes dominios numéricos soportados por Oracle son subconjuntos de los enteros y los reales. Todos ellos pueden ser representados a través del tipo NUMBER(p,s), donde los parámetros p (precisión) y s (escala) indican, respectivamente, el número máximo de dígitos y el número de cifras decimales que puede contener el atributo. Existen otras expresiones sintácticas de tipo aceptadas pero todas ellas pueden encontrar una expresión equivalente mediante el tipo NUMBER.
- **Fecha.** Oracle permite representar fechas en diferentes formatos para un rango comprendido entre el 1 de Enero del año 4712 A.C. y el 31 de Diciembre del 4712 D.C. El tipo que opera sobre este dominio es el tipo DATE.
- **Binario.** A través de los datos definidos sobre este dominio se pueden representar cualquier tipo de información mediante una representación binaria, como una colección de bytes, sin que Oracle les atribuya significado alguno. Los tipos que sustentan este dominio son el RAW para una longitud máxima de 255 bytes y el LONG RAW para una longitud de 65535 bytes.
- **Identificador de fila.** Oracle identifica mediante un código cada uno de los registros de la base de datos. Este dominio contiene todos los valores

que puede adoptar la mencionada codificación. El tipo ROWID refiere un atributo a dicho dominio.

En cuanto a la naturaleza de las tuplas, estas presentan las características establecidas por el modelo para las mismas, es decir, no se encuentran ordenadas (aunque se pueden recuperar de la base de datos siguiendo algún criterio de ordenación mediante la cláusula ORDER BY del SQL). Con respecto a la *unicidad* de las tuplas, Oracle proporciona mecanismos para garantizar la misma mediante la creación de índices con la cláusula UNIQUE sobre las claves primarias.

Es posible la creación de *vistas* como representación lógica de una tabla o un conjunto de tablas. Sin embargo estas, aunque no presentan problemas en consulta, pueden plantearlos en la actualización de las mismas, al resultar imposible en algunas ocasiones reconstruir la información a actualizar en las tablas base sobre las que se construyen.

Para todos los dominios existe un valor denominado NULL que significa que, o no se conoce el valor o no es aplicable. Oracle proporciona un tratamiento específico para este tipo de valores.

Otros elementos estructurales proporcionados por Oracle son:

- Los *índices*, los cuales facilitan, en forma transparente al usuario, rápido acceso a las tuplas de una tabla y pueden forzar la unicidad de las mismas.
- Las *agrupaciones (clusters)* pueden ayudar al sistema a almacenar físicamente estructuras de datos a las que se accede conjuntamente con frecuencia, aunque pertenecen a tablas distintas. Al igual que los índices, su empleo es transparente al usuario y es el optimizador el que se encarga de usarlos para mejorar la eficiencia del sistema.

2. Integridad de los Datos

Oracle fuerza la *Regla de Identidad* mediante el uso de la cláusula NOT NULL en la creación y modificación de las tablas. Como hemos dicho, la *Regla Referencial* está actualmente soportada semánticamente por la versión 7 de Oracle. Además, dispone de un conjunto de restricciones para mantener la integridad de la base de datos que se pueden aplicar a las sentencias de creación y modificación de tablas

(CREATE TABLE y ALTER TABLE). Dichas restricciones se pueden imponer a una columna o a un grupo de ellas y determinan, que las operaciones que se realicen sobre las mismas, satisfagan algunas condiciones. Estas restricciones son:

- que una columna o grupo de ellas contengan valores no nulos. Esta restricción, como hemos apuntado anteriormente, supone el soporte para la *Regla de Identidad*.
- que una columna o grupo de ellas sean únicas dentro de la tabla.
- se puede identificar una columna o grupo de ellas como CLAVE PRIMARIA.
- se puede forzar a que existan en alguna tabla las columnas o grupos de ellas que constituyan la CLAVE EXTERNA en otra tabla. Osea que se cumpla en todo caso la *Regla de Integridad Referencial*.
- restringir los *valores de dominio* que pueden ser adoptados por una columna o grupo de ellas. Con esta restricción se pretende especificar, en forma precisa, el dominio que subyace a un atributo o columna.

3. Manipulación de los Datos

En este ámbito, a través de un lenguaje basado en el *Álgebra Relacional*, SQL, se soportan todas las operaciones relacionales consideradas en el “modelo relacional”. También es posible modelizar mediante este lenguaje el Cálculo Relacional Orientado a Tuplas. Con respecto al Cálculo Relacional Orientado a Dominios Oracle cuenta con un desarrollo basado en lenguajes de 4ª Generación denominado SQLFORMS que permite realizar consultas mediante un sistema parecido al QBE.

Capítulo 3

Modelos de Bases de Datos Relacionales Difusas

3.1 Introducción

En el capítulo anterior se han introducido las principales características del “modelo relacional”. Esto es básico puesto que si queremos construir un modelo de *bases de datos relacionales difusas*, (SBDRD en lo sucesivo), hemos de satisfacer con el mismo los requerimientos del “modelo relacional”. Por otra parte, un SBDRD tendrá que contemplar los mecanismos para proporcionar almacenamiento y tratamiento a la información de tipo difuso. En este capítulo introduciremos algunos elementos básicos de la teoría de conjuntos difusos sobre los que se fundamentan los desarrollos difusos del modelo relacional. Tras un repaso a las aproximaciones posibilísticas al problema del tratamiento de la información incompleta, enunciaremos algunas de las características que debe reunir una extensión del modelo relacional en el ámbito de la teoría de conjuntos difusos. A continuación, estudiaremos los principales modelos disponibles en la literatura, resaltando las características más importantes de cada uno de ellos y analizando las ventajas e inconvenientes que presentan.

3.2 Teoría de Conjuntos Difusos

Vamos a dedicar este apartado a introducir algunas nociones elementales sobre la teoría de conjuntos difusos, así como la notación utilizada al respecto a lo largo de

esta memoria. En este resumen nos detendremos en los aspectos semánticos y de representación relacionados con esta potente herramienta teórica. En la literatura podemos encontrar una gran cantidad de trabajos sobre esta teoría introducida por primera vez en [Zadeh 65]. En [Yager 87] podemos encontrar una recopilación de algunos de los artículos más interesantes publicados sobre el tema por L. A. Zadeh. En [Dubois, Prade, 79], [Dubois, Prade, 88] y [Zimmermann 91] es posible encontrar recopilados los aspectos más importantes que constituyen la teoría de conjuntos difusos así como la teoría de la posibilidad.

3.2.1 Conjuntos Difusos

La interpretación original de conjunto difuso proviene de una generalización del concepto clásico de subconjunto ampliado a la descripción de nociones “vagas” e “imprecisas”. Esta generalización se realiza como sigue:

1. La pertenencia de un elemento a un conjunto pasa a ser un concepto “difuso”.
2. Dicha pertenencia puede ser cuantificada por un grado. Dicho grado se denomina habitualmente como “grado de pertenencia” de dicho elemento al conjunto y toma un valor en el intervalo $[0, 1]$.

Mediante esta herramienta podemos representar de forma adecuada conceptos “imprecisos”. En la tabla 3.1 ilustramos la forma en que podemos modelar el concepto “joven”. Para ello consideramos los valores que puede tomar la edad de un individuo, (universo de discurso), y el grado de pertenencia de cada edad al concepto “joven”.

Es necesario hacer notar que el concepto “joven” como otros muchos de naturaleza “imprecisa” responde a criterios subjetivos. De forma más precisa podemos introducir la definición de conjunto difuso como sigue:

Definición 3.1 *Un conjunto difuso \tilde{A} sobre un universo de discurso Ω es un conjunto de pares*

$$\tilde{A} = \{x, \mu_A(x) : x \in \Omega, \mu_A(x) \in [0, 1]\} \quad (3.1)$$

donde $\mu_A(x)$ se denomina grado de pertenencia de x a \tilde{A}

Según esto, si la “edad” es el universo de discurso de “joven”, el conjunto difuso que representa dicho concepto quedaría expresado en la forma:

$$joven = \{(20, 1.00), (25, 1.00), \dots, (40, 0.10), (50, 0.04)\}$$

Edad	Grado de Pertenencia
20	1.00
25	1.00
26	0.96
28	0.74
30	0.50
35	0.20
40	0.10
50	0.04

Tabla 3.1: Conjunto difuso representando el concepto “joven”

El identificador “joven” con la connotación de que lleva asociado un conjunto difuso recibe la denominación de “etiqueta lingüística”. Existen varias notaciones para el concepto de conjunto difuso dependiendo de la naturaleza del universo de discurso sobre el que definamos un conjunto difuso. Algunas de las más importantes son:

- Dado un universo de discurso finito

$$\Omega = \{x_1, x_2, \dots, x_n\}$$

un conjunto difuso \tilde{A} se puede notar como

$$\tilde{A} = \mu_1/x_1 + \mu_2/x_2 + \dots + \mu_n/x_n \quad (3.2)$$

donde μ_i representa el grado de pertenencia de x_i , con $i = 1, 2, \dots, n$. Habitualmente los elementos con grado cero no se listan.

- Dado un universo de discurso infinito Ω , un conjunto difuso \tilde{A} sobre Ω se puede representar como

$$\tilde{A} = \int \mu_{\tilde{A}}(x)/x, \quad (3.3)$$

donde $\mu_{\tilde{A}}(x)$ es el grado de pertenencia de x .

Algunos conceptos sobre conjuntos difusos son:

- *Igualdad* de conjuntos difusos

Definición 3.2 Dos conjuntos difusos \tilde{A} y \tilde{B} sobre Ω se dicen **iguales**, (notado como $\tilde{A} = \tilde{B}$), sii

$$\forall x \in \Omega, \mu_{\tilde{A}}(x) = \mu_{\tilde{B}}(x) \quad (3.4)$$

- *Inclusión* de un conjunto en otro

Definición 3.3 Dados dos conjuntos difusos \tilde{A} y \tilde{B} sobre Ω , decimos que $\tilde{A} \subseteq \tilde{B}$ sii

$$\forall x \in \Omega, \mu_{\tilde{A}}(x) \leq \mu_{\tilde{B}}(x) \quad (3.5)$$

- *Soporte* de un conjunto difuso

Definición 3.4 El **soporte** de un conjunto difuso \tilde{A} definido sobre Ω es un subconjunto de dicho universo que satisface:

$$supp(\tilde{A}) = \{x \in \Omega, \mu_{\tilde{A}}(x) > 0\} \quad (3.6)$$

- *Núcleo* de un conjunto difuso

Definición 3.5 El **núcleo** de un conjunto difuso \tilde{A} definido sobre Ω es un subconjunto de dicho universo que satisface:

$$kern(\tilde{A}) = \{x \in \Omega, \mu_{\tilde{A}}(x) = 1\} \quad (3.7)$$

- *Altura* de un conjunto difuso

Definición 3.6 La **Altura** de un conjunto difuso \tilde{A} definido sobre Ω se define como:

$$hgt(\tilde{A}) = \sup_{x \in \Omega} \mu_{\tilde{A}}(x) \quad (3.8)$$

- Conjunto difuso *normalizado*

Definición 3.7 Un conjunto difuso \tilde{A} definido sobre Ω se dice **normalizado** sii

$$\exists x \in \Omega, \mu_{\tilde{A}}(x) = 1 \quad (3.9)$$

esta definición implica que $hgt(\tilde{A}) = 1$

3.2.2 Operaciones Sobre Conjuntos Difusos

Las operaciones que inmediatamente se sugieren de la definición de conjunto difuso son la unión, la intersección y el complemento.

- **Union**

Definición 3.8 Si \tilde{A} y \tilde{B} son dos conjuntos difusos sobre un universo de discurso Ω , la función de pertenencia de la **union** de ambos conjuntos, $\tilde{A} \cup \tilde{B}$, viene dada por

$$\mu_{\tilde{A} \cup \tilde{B}}(x) = f(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)), \quad x \in \Omega \quad (3.10)$$

donde f es una T -conorma [Schweizer, Sklar].

- **Intersección**

Definición 3.9 Si \tilde{A} y \tilde{B} son dos conjuntos difusos sobre un universo de discurso Ω , la función de pertenencia de la **intersección** de ambos conjuntos, $\tilde{A} \cap \tilde{B}$, viene dada por

$$\mu_{\tilde{A} \cap \tilde{B}}(x) = g(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)), \quad x \in \Omega \quad (3.11)$$

donde g es una T -norma [Schweizer, Sklar].

Las definiciones anteriores no son únicas ya que existen varios operadores que satisfacen el concepto de T -norma y de T -conorma. Los más importantes son:

- Los operadores *idempotentes*. El **máximo** y el **mínimo** para la unión y la intersección respectivamente. Satisfacen, además de la idempotencia, la propiedad distributiva aplicada sobre ambos y son estrictamente crecientes. Estos operadores son los más utilizados porque conservan gran cantidad de las propiedades de los operadores booleanos. En [Bellman, Giertz, 73] puede encontrarse una justificación a la elección de los operadores **max** y **min** para definir los anteriores conceptos.
- Los operadores *arquimedianos*. Emplean la **suma probabilística**, $(x + y - x * y)$, y el **producto**, $(x * y)$, para la unión y la intersección, respectivamente. Estos operadores no satisfacen la propiedad distributiva ni son idempotentes.

- Los operadores *acotados*. Los operadores dados por, $\min(1, x+y)$ y $\max(0, x+y-1)$, representan la unión y la intersección respectivamente. Estos operadores no satisfacen la idempotencia, la propiedad distributiva ni la propiedad de absorción. Por contra, satisfacen conmutatividad, asociatividad e identidad.

• Complemento

La noción de complemento se puede construir a partir del concepto de negación fuerte [Trillas 79].

Definición 3.10 *Una función C de $[0,1]$ en $[0,1]$ es una **negación fuerte** si satisface las siguientes condiciones:*

1. $C(0)=1$
2. $C(C(a))=a$ (involución)
3. C es estrictamente decreciente
4. C es continua.

Aunque existen varios tipos de operadores que satisfacen tales propiedades o versiones relajadas de las mismas, nosotros, para el complemento, emplearemos principalmente la versión proporcionada en [Zadeh 65], en la cual:

$$C(x) = 1 - x \quad (3.12)$$

Por tanto, para un conjunto difuso \tilde{A} sobre un universo de discurso Ω , la función de pertenencia del **complemento** de \tilde{A} , $\neg\tilde{A}$, viene dada por

$$\mu_{\neg\tilde{A}}(x) = 1 - \mu_{\tilde{A}}(x), \quad x \in \Omega \quad (3.13)$$

3.2.3 Números Difusos

El concepto de número difuso fue introducido por primera vez en [Zadeh 75] con el propósito de analizar y manipular valores numéricos aproximados. El concepto ha sido refinado sucesivamente, en esta memoria entenderemos por número difuso lo siguiente [Dubois, Prade, 88]:

Definición 3.11 Sea A un subconjunto difuso de \mathfrak{R} y μ_A su función de pertenencia cumpliendo:

1. $\forall x, y \in \mathfrak{R}, \forall \mu_A(t) \geq \min\{\mu_A(x), \mu_A(y)\}$, es decir que es convexo.
2. μ_A es semicontinua superiormente.
3. El soporte de A es un conjunto acotado.

entonces diremos que A es un **número difuso**

Algunos autores incluyen en la definición la necesidad de que el subconjunto difuso esté normalizado.

La forma general de la función de pertenencia de un número difuso M , es la siguiente:

$$\mu_M(x) = \begin{cases} r_M(x) & \text{si } x \in [m-a, m] \\ \alpha_M & \text{si } x \in [m, n] \\ s_M(x) & \text{si } x \in (n, n+b] \\ 0 & \text{en otro caso} \end{cases} \quad (3.14)$$

donde $r_M, s_M : \mathfrak{R} \rightarrow [0, 1]$, r_M no decreciente, s_M no creciente, $r_M(m) = \alpha_M = s_M(n)$, $\alpha_M \in (0, 1]$ y $a, b, m, n \in \mathfrak{R}$.

Al número α_M se le denomina *altura* del número difuso, al intervalo $[n, m]$ *intervalo modal* y a los números a y b *holguras izquierda y derecha* respectivamente. El número difuso de la fig. 3.1 es una representación de “aproximadamente entre m y n ”.

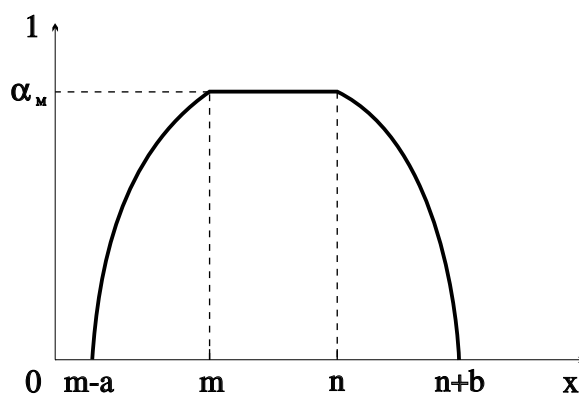


Figura 3.1: Aspecto general de un número difuso

A lo largo de esta memoria utilizaremos a menudo un caso particular de números difusos que se obtienen cuando consideramos a r_M y s_M como funciones lineales. En este caso la función de pertenencia adopta la forma:

$$\mu_M(x) = \begin{cases} \alpha_M + (x - m)\alpha_M/a & \text{si } x \in [m - a, m] \\ \alpha_M & \text{si } x \in [m, n] \\ \alpha_M - (x - n)\alpha_M/b & \text{si } x \in (n, n + b] \\ 0 & \text{en otro caso} \end{cases} \quad (3.15)$$

A un número difuso de este tipo lo llamaremos *triangular o trapezoidal*. Usualmente trabajaremos con números difusos normalizados por lo que $\alpha_M = 1$, en este caso podremos caracterizar un número difuso trapezoidal normalizado M , mediante el empleo de los parámetros m, n, a, b como sigue:

$$M \equiv (m, n, a, b)$$

La fig. 3.2 muestra una representación gráfica de dicho número.

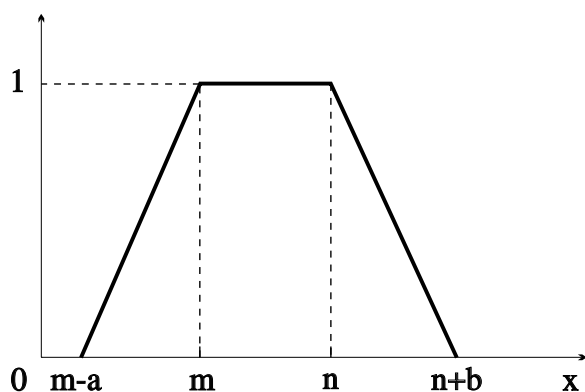


Figura 3.2: Número difuso trapezoidal normalizado

El Principio de Extensión

Este principio, propuesto en [Zadeh 75], proporciona un método general que permite extender conceptos matemáticos no difusos para el tratamiento de cantidades difusas. Se define como sigue:

Definición 3.12 Sea Ω un producto cartesiano de universos $\Omega = \Omega_1 \times \Omega_2 \times \dots \times \Omega_n$, y $\widetilde{A}_1, \widetilde{A}_2, \dots, \widetilde{A}_n$, n conjuntos difusos de $\Omega_1, \Omega_2, \dots, \Omega_n$ respectivamente, f una función desde Ω al universo Ω' , $y = f(x_1, x_2, \dots, x_n)$, entonces un conjunto difuso \widetilde{B} de Ω' viene definido por:

$$\widetilde{B} = \int_{\Omega'} \mu_{\widetilde{B}}(y)/y$$

donde,

$$\mu_{\widetilde{B}}(y) = \begin{cases} \max \min_{(x_1, x_2, \dots, x_n) \in f^{-1}(y)} \{\mu_{\widetilde{A}_1}(x_1), \dots, \mu_{\widetilde{A}_n}(x_n)\}, & f^{-1}(y) \neq \emptyset \\ 0, & \text{en otro caso} \end{cases}$$

Aritmética Difusa

Gracias al principio de extensión es fácil extender las operaciones aritméticas clásicas al tratamiento de números difusos. De esta forma las cuatro operaciones principales quedan extendidas como sigue:

- *Suma Extendida*

Dadas dos cantidades difusas A_1 y A_2 , la función de pertenencia de la suma viene dada por la expresión:

$$\mu_{A_1 \oplus A_2}(y) = \sup \{ \min(\mu_{A_1}(y-x), \mu_{A_2}(x)) / x \in \mathfrak{R} \} \quad (3.16)$$

- *Diferencia Extendida*

Dadas dos cantidades difusas A_1 y A_2 , la función de pertenencia de la suma viene dada por la expresión:

$$\mu_{A_1 \ominus A_2}(y) = \sup \{ \min(\mu_{A_1}(y+x), \mu_{A_2}(x)) / x \in \mathfrak{R} \} \quad (3.17)$$

- *Producto Extendido*

El producto de dos cantidades difusas $A_1 \odot A_2$ se obtiene:

$$\mu_{A_1 \odot A_2}(z) = \begin{cases} \sup \{ \min(\mu_{A_1}(z/y), \mu_{A_2}(y)) / y \in \mathfrak{R} - \{0\} \} & \text{si } z \neq 0 \\ \max(\mu_{A_1}(0), \mu_{A_2}(0)) & \text{si } z = 0 \end{cases} \quad (3.18)$$

- *División Extendida*

La división de dos cantidades difusas se define mediante:

$$\mu_{A_1 \div A_2}(z) = \sup\{\min(\mu_{A_1}(y.z), \mu_{A_2}(y)) / y \in \mathfrak{R}\} \quad (3.19)$$

Basándose en una expresión particular del principio de incertidumbre adaptada al empleo de α -cortes y en un tipo de número difuso similar al descrito anteriormente, denominado número $L - R$, en [Dubois, Prade, 79b] se describen fórmulas de cálculo rápido para las anteriores operaciones aritméticas.

3.3 El Tratamiento de la Información Incompleta por medio de Conjuntos Difusos

En el capítulo anterior planteábamos las aproximaciones clásicas al problema del tratamiento de la información incompleta. En este apartado retomamos el problema analizando algunas de las aproximaciones surgidas bajo la teoría de los conjuntos difusos. Estas aproximaciones, no solo pueden representar el hecho de que se disponga o no de una información, sino que pueden matizar el grado en que se posee dicha información. Es decir, pueden manejar tipos de información que no son proporcionados al sistema en forma precisa. Por ejemplo, supongamos que yo sé que “Luis no tiene más de 40 años”, esta información no podría ser almacenada como tal en un RDBMS clásico, la única posibilidad que nos proporciona el sistema es la de catalogar un desconocimiento total por nuestra parte con respecto a la edad de Luis, por tanto, en la casilla correspondiente almacenaría un valor “desconocido pero aplicable”. Los modelos de bases de datos relacionales difusas van más lejos y nos proporcionan la posibilidad de almacenar y manejar la información imprecisa antes mencionada. Pero, antes de estudiar como realizan esto, vamos a ver de que manera pueden representar los tipos de información “ausente” estudiados a lo largo del capítulo anterior. Dejaremos para los próximos apartados la forma en que los diferentes modelos de bases de datos relacionales difusas tratan la información imprecisa.

3.3.1 El Enfoque Posibilístico del Modelo de Prade-Testemale

Estos autores, cuyo modelo de bases de datos relacionales difusas trataremos en profundidad en el apartado quinto, incorporan en su modelo una representación específica para lo que denominan datos “incompletos” o “inciertos” en el ámbito de la Teoría de la Posibilidad. Esbozaremos las líneas generales de dicha representación, en [Prade 84b] y en [Prade, Testemale, 87] se puede encontrar un tratamiento más detallado.

Consideremos un atributo A cuyo dominio es D , donde D es, por ejemplo, el conjunto de todos los posibles valores que puede tomar A . Todo nuestro conocimiento disponible acerca del valor que toma A para un objeto x puede ser representado mediante una distribución de posibilidad $\pi_{A(x)}$ sobre $D \cup \{e\}$, donde e es un elemento especial que denota el caso en que A no se aplica a x . En otras palabras, $\pi_{A(x)}$ es una aplicación que va de $D \cup \{e\}$ al intervalo $[0, 1]$.

Veremos mediante un ejemplo como podemos utilizar el anterior esquema para representar diferentes tipos de conocimiento acerca de un hecho. En las siguientes situaciones vamos a modelar nuestro conocimiento acerca de la “*edad del coche de Luis*”.

1. “*El coche de Luis tiene 2 años*”. Esta situación muestra el caso clásico y se representa con un grado de posibilidad 1 para el valor 2 y 0 para el resto de los valores de dominio.

$$\begin{aligned}\pi_{edad_del_coche(Luis)}(e) &= 0 \\ \pi_{edad_del_coche(Luis)}(2) &= 1 \\ \pi_{edad_del_coche(Luis)}(d) &= 0 \text{ resto de valores } d \in D\end{aligned}$$

2. “*Luis no tiene coche*”. Esto implica que la propiedad “*edad_del_coche*” no es aplicable, esto nos remite al concepto de “desconocido y no aplicable”, y se representa mediante una distribución de posibilidad que toma valor 1 para el elemento e y 0 para el resto de los valores del Dominio D .

$$\begin{aligned}\pi_{edad_del_coche(Luis)}(e) &= 1 \\ \pi_{edad_del_coche(Luis)}(d) &= 0 \quad \forall d \in D\end{aligned}$$

3. “Luis tiene coche pero no sé la edad del mismo”. Esta situación hace referencia al concepto de “desconocido pero aplicable”. La representación quedaría como sigue:

$$\begin{aligned}\pi_{edad_del_coche(Luis)}(e) &= 0 \\ \pi_{edad_del_coche(Luis)}(d) &= 1 \quad \forall d \in D\end{aligned}$$

4. “No sé si Luis tiene coche y, por tanto no sé la edad del coche”. Mi conocimiento sobre la situación abre las siguientes posibilidades:

- (a) Luis puede tener coche y su edad puede ser cualquiera.
- (b) Luis puede no tener coche en cuyo caso no es aplicable la edad.

Esta situación queda representada mediante una distribución de posibilidad que toma el valor 1 para todos los posibles valores del dominio D y para el elemento especial e . Esto es, cualquier cosa es posible.

$$\pi_{edad_del_coche(Luis)}(d) = 1, \quad \forall d \in D \cup \{e\}$$

5. “Luis tiene coche y sé que tiene entre 2 y 4 años”. El modelo permite representar este tipo de conocimiento mediante la expresión:

$$\begin{aligned}\pi_{edad_del_coche(Luis)}(e) &= 0 \\ \pi_{edad_del_coche(Luis)}(d) &= \begin{cases} 1 & \text{si } d \in [2, 4] \subseteq D \\ 0 & \text{en caso contrario} \end{cases}\end{aligned}$$

6. “Luis tiene coche y sé que es nuevo”. El conocimiento que tengo sobre la edad del coche es impreciso, característica que puede ser modelada mediante distribuciones de posibilidad en la forma:

$$\begin{aligned}\pi_{edad_del_coche(Luis)}(e) &= 0 \\ \pi_{edad_del_coche(Luis)}(d) &= \mu_{nuevo}(d) \quad \forall d \in D\end{aligned}$$

donde $\mu_{nuevo}(d)$ representa la función de pertenencia del predicado difuso “nuevo”.

7. “La posibilidad de que Luis no tenga coche es λ y, caso de que lo tenga es nuevo”. En este caso puede ser modelado también el grado de creencia sobre el hecho de que Luis tenga coche.

$$\begin{aligned}\pi_{edad_del_coche(Luis)}(\epsilon) &= \lambda, \quad \lambda > 0 \\ \pi_{edad_del_coche(Luis)}(d) &= \mu_{nuevo}(d) \quad \forall d \in D\end{aligned}$$

donde $\mu_{nuevo}(d)$ conserva el significado mostrado anteriormente.

Los casos 1, 2 y 3 modelan las situaciones tratadas en el capítulo anterior, mientras que el resto supone un muestrario de las capacidades de representación del modelo. La manipulación que realiza el modelo de Prade-Testemale sobre la información representada mediante distribuciones de posibilidad será objeto de estudio en el quinto apartado de este capítulo.

3.3.2 El Modelo de Umano et al.

Uno de los primeros modelos de bases de datos relacionales difusas fue el que presentaron Umanno, Fukami et al. en [Umano 82], en dicho modelo, cuyo estudio abordaremos en el próximo apartado, se recoge un esquema para la representación de la información “ausente” algo distinta de la propuesta por Prade et al.

En esta propuesta utiliza las distribuciones de posibilidad para modelar el conocimiento sobre la información en forma similar a como lo hacen Prade et al. La diferencia reside en cómo concibe la información “desconocida por no aplicable”. Umano et al. entiende que este tipo de información puede ser modelada por una distribución de posibilidad sobre el dominio considerado, en la que cada valor de dominio aparece con un valor de posibilidad igual a 0. Es decir, si U es el universo de discurso de $A(x)$ y $\pi_{A(x)}(u)$ representa la posibilidad de que $A(x)$ tome el valor u en U , entonces existe un caso especial de distribución de posibilidad denominado “indefinido” (undefined) y que se representa como:

$$\pi_{A(x)}(u) = 0 \quad \forall u \in U$$

Para los valores “desconocidos y aplicables”, que denomina “unknown”, emplea la misma representación que en Prade et al. O sea:

$$\pi_{A(x)}(u) = 1 \quad \forall u \in U$$

Para representar la situación en la que no se conoce incluso si una “ausencia” es “aplicable” o no, Umano et al. emplean un valor especial que denominan “NULL” y que modelan como:

$$NULL = \{1/unknown, 1/undefined\}$$

Para el resto de los casos de información “imprecisa” adopta una representación similar a la de Prade et al.

3.3.3 Comentarios

La tabla 3.2 esquematiza la solución que cada modelo proporciona a la representación de cada una de las situaciones mostradas en los ejemplos del subapartado 3.3.1. La propuesta de Prade et al. parece una extensión, en el ámbito de la teoría de la posibilidad, de la representación adoptada en el esquema de Codd mediante el empleo de las “I-marcas”. No obstante, añade más significado al concepto de información “incompleta” puesto que puede tratar con información “imprecisa” en una acepción más amplia. La forma en que Umano et al. tratan la información “ausente” parece más intuitiva puesto que la representación del concepto “indefinido” resulta más acorde con el significado de las distribuciones de posibilidad. Por contra, el concepto “NULL” se construye como una distribución de posibilidad sobre las distribuciones de posibilidad que representan los conceptos “unknown” y “undefined”. El modelo que se presentará en esta memoria utiliza una representación para la información “ausente” basada en el esquema de Umano et al. En cualquier caso, la potencialidad de la teoría de conjuntos difusos para representar información incompleta en bases de datos se hace patente en estos ejemplos. Por ello, se muestra como una alternativa importante para resolver los problemas planteados por este tipo de información en las bases de datos.

3.4 Características de un Modelo de Base de Datos Relacional Difuso

La información difusa forma parte de nuestra vida cotidiana y se manifiesta frecuentemente en cualquier acto de comunicación humana. Los aspectos más importantes de

Ejemplo	Mod. Prade-Testemale	Mod. Umano-Fukami
1) “El coche de Luis tiene 2 años”	$\pi_{edad_del_coche}(Luis)(e) = 0$ $\pi_{edad_del_coche}(Luis)(2) = 1$ $\pi_{edad_del_coche}(Luis)(d) = 0,$ resto valores $d \in D$	$\pi_{edad_del_coche}(Luis)(d) =$ $\{1/2\}_P$
2) “Luis no tiene coche”	$\pi_{edad_del_coche}(Luis)(e) = 1$ $\pi_{edad_del_coche}(Luis)(d) = 0,$ $\forall d \in D$	Undefined
3) “Luis tiene coche pero no sé la edad del mismo”	$\pi_{edad_del_coche}(Luis)(e) = 0$ $\pi_{edad_del_coche}(Luis)(d) = 1,$ $\forall d \in D$	Unknown
4) “No sé si Luis tiene coche y, por tanto no sé la edad del coche”	$\pi_{edad_del_coche}(Luis)(d) = 1,$ $\forall d \in D \cup \{e\}$	Null
5) “Luis tiene coche y sé que tiene entre 2 y 4 años”	$\pi_{edad_del_coche}(Luis)(e) = 0$ $\pi_{edad_del_coche}(Luis)(d) =$ $\begin{cases} 1 & \text{si } d \in [2, 4] \subseteq D \\ 0 & \text{en caso contrario} \end{cases}$	$\pi_{edad_del_coche}(Luis)(d) =$ $\begin{cases} 1 & \text{si } d \in [2, 4] \subseteq D \\ 0 & \text{en caso contrario} \end{cases}$
6) “Luis tiene coche y sé que es nuevo”	$\pi_{edad_del_coche}(Luis)(e) = 0$ $\pi_{edad_del_coche}(Luis)(d) =$ $\mu_{nuevo}(d) \quad \forall d \in D$	$\pi_{edad_del_coche}(Luis)(d) =$ $\mu_{nuevo}(d) \quad \forall d \in D$
7) “La posibilidad de que Luis no tenga coche es λ y, caso de que lo tenga es nuevo”	$\pi_{edad_del_coche}(Luis)(e) = \lambda,$ $\pi_{edad_del_coche}(Luis)(d) =$ $\mu_{nuevo}(d) \quad \forall d \in D$	No representable

Tabla 3.2: Representación de información incompleta mod. posibilísticos

la información difusa que habitualmente manejamos son: la incertidumbre y la imprecisión en nuestras apreciaciones. El primero se deriva de apreciaciones realizadas sobre nuestra observación de la realidad que no pueden aportar un grado de certidumbre total sobre lo que afirmamos, por ejemplo, “*Luis es profesor, creo que de Universidad pero podría ser de Instituto*”. El segundo aspecto se manifiesta a través del enunciado de conceptos que, o no se encuentran bien diferenciados o definidos, o bien resultan subjetivos, por ejemplo, la afirmación “*Luis es castaño*”, no precisa exactamente el color del pelo. A veces en nuestras conversaciones se combinan ambos aspectos. Es obvio que los enunciados mostrados en los dos ejemplos proporcionan un grado de información sobre el mundo real. En algunos casos la información que proporcionan puede resultar suficiente y en otros puede no serlo, en cualquier caso aumentan nuestra información sobre el universo, ahora sabemos, por ejemplo, que Luis no tiene el pelo, ni completamente negro ni rubio, y tenemos una idea acerca del color de su pelo, también sabemos que nuestro interlocutor cree más en la posibilidad de que sea profesor de Universidad que en la de que sea profesor de Instituto.

Todas estas consideraciones nos conducen a plantear la conveniencia de que el sistema satisfaga la siguiente característica:

1.- Un SBDRD debe proporcionar los mecanismos adecuados para poder representar información difusa en todas sus vertientes.

No es misión del SBDRD el dotar a esa información de ninguna semántica particular. Esta tarea ha de realizarla el usuario a la hora de diseñar la aplicación particular que resuelva su problema. Sin embargo, el SBDRD debe proporcionar un entorno adecuado para representar dicha semántica. Por tanto:

2.- Un SBDRD deberá representar un marco adecuado para el almacenamiento del significado de la información difusa que albergue.

La información, en si misma, no aporta ninguna utilidad si no se dispone de mecanismos adecuados para recuperarla y para operar sobre ella. De poco nos sirve poder almacenar de alguna forma un enunciado difuso como: “*Luis es alto*”, si, como respuesta a una consulta del tipo: “*Dame los individuos algo altos*” no aparece *Luis* satisfaciéndola en mayor o menor grado. A la luz de estas consideraciones hemos de establecer el siguiente requisito a satisfacer por un SDBRD:

3.- Un SDBRD tendrá que facilitar un conjunto mínimo de operadores para recuperar y tratar la información atendiendo a la naturaleza difusa de la misma.

Las características anteriormente relacionadas hacen referencia a la naturaleza difusa de la información a manejar. En ningún momento se exige un carácter “relacional” al modelo que las satisfaga. De hecho estas características podrían ser cumplidas por modelos de bases de datos que operen bajo una filosofía distinta. Por tanto para que un SBDRD pueda ser llamado “relacional” con propiedad, deberá operar en el marco establecido por el “modelo relacional”. Como vimos que este modelo presenta varias aproximaciones y los sistemas implementados satisfacen en diversos grados los requisitos del mismo, es lógico pensar que los modelos de SBDRD presentarán, así mismo, distinto grado de “relacionalidad”. Por tanto hay que añadir una característica más a un SBDRD:

4.- Un SBDRD deberá satisfacer los requisitos del “modelo relacional” en la mayor medida posible.

El cumplimiento de esta última característica requiere de un gran esfuerzo en la definición del modelo de SBDRD. Esto se debe a que hay elementos básicos del “modelo relacional clásico” que son claramente incompatibles con una representación imprecisa de la información. Por ejemplo, si una relación almacena para todos sus atributos información difusa, se hará imposible la definición de claves primarias y no se podrá garantizar el acceso a cada tupla de la relación, por el mismo motivo, también se puede ver comprometido el concepto de “unicidad”. Algunos operadores relacionales clásicos pueden presentar problemas para su extensión difusa, tal es el caso del operador *REUNIÓN*. No obstante, se puede llegar a compromisos que no resten significado “relacional” al modelo de SBDRD.

3.5 Principales Modelos de Bases de Datos Difusas

En esta sección vamos a realizar un análisis de los principales modelos existentes en la actualidad. Veremos como ninguno de ellos satisface completamente todas las características mencionadas en la sección anterior.

Los diferentes niveles “difusos” que se pueden cubrir mediante un modelo de SBDRD son:

- La obtención, en consulta, de información difusa a partir de datos almacenados

en forma precisa.

- La representación y recuperación de información difusa.
- La representación y recuperación de información difusa y el tratamiento de la misma (en el sentido de inferir nueva información).

En [Hamon 86], [Bosc 88], [Kacprzyk 86], [Tahani 77], tenemos algunos ejemplos de desarrollos en torno al primer nivel. Mediante estos modelos se elaboran mecanismos de consulta, a través de extensiones de lenguajes ya existentes, como SQL, o de lenguajes nuevos, para obtener en forma difusa la información contenida en una base de datos relacional. Lo más interesante de estos modelos, lo constituye quizás, la traducción de consultas SQL difusas a SQL clásico que se realiza en [Hamon 86] y [Bosc 88].

Sobre el tercer nivel existen algunas aproximaciones incompletas de la mano de implementaciones de “Bases de Datos Lógicas Relacionales Difusas” con Fuzzy-Prolog, [Li, Liu, 90]. Actualmente nuestro grupo de trabajo está desarrollando un modelo teórico [Vila et al. 1994c], que sirve como marco para el desarrollo de implementaciones que realicen inferencias sobre la información difusa, almacenada a través de predicados de primer orden. Así mismo, el modelo de SBDRD que se presentará en el capítulo cuarto, posibilita el montaje de mecanismos de inferencia difusos mediante el empleo de estructuras coherentes con la filosofía de representación del conocimiento adoptada.

En el segundo nivel es donde actualmente se realizan mayores esfuerzos para desarrollar un modelo adecuado. En los siguientes subapartados vamos a reflejar el estado del arte en cuanto a los modelos existentes.

Las soluciones propuestas se agrupan en torno a tres enfoques diferentes:

- *El Modelo Relacional Difuso*
- *El Modelo de Unificación Mediante Relaciones de Similitud*
- *Modelos Relacionales Sobre Distribuciones de Posibilidad*

3.5.1 Modelo Relacional Difuso

Este modelo contempla la extensión difusa de una relación mediante la presencia de grados de pertenencia. Según este modelo, cada tupla presenta un grado de pertenencia

a la relación que normalmente se encuentra comprendido entre 0 y 1. En forma más precisa se enuncia:

Una Base de Datos Difusa DB está constituida por un conjunto de relaciones:

$$DB = \{R_1, R_2, \dots, R_m\} \quad (3.20)$$

Donde cada R es una relación difusa caracterizada por la función de pertenencia:

$$\mu_R : U_1 \times U_2 \times \dots \times U_n \rightarrow [0, 1] \quad (3.21)$$

siendo U_i el dominio del atributo i -ésimo de la relación R , y \times el Producto Cartesiano.

Aunque en [Zadeh 78] y [Baldwin 87] se recogen algunos sistemas construidos según este modelo estos se han realizado con otro objeto.

La extensión difusa que representa este modelo no satisface completamente las características reseñadas en el apartado anterior. Concretamente, no ofrece capacidad para representar convenientemente gran parte de los tipos difusos, el manejo de la información difusa se realiza en forma artificiosa, ya que, una tupla asume su carácter difuso en forma global sin que podamos en forma alguna determinar cual es la aportación difusa de cada uno de los atributos que la constituyen. Además, resulta incapaz de satisfacer los requisitos del “modelo relacional” relativos a la *unicidad*, a la *accesibilidad* de cada tupla y a la implementación de operadores relacionales tal como la REUNIÓN.

3.5.2 Unificación Mediante Relaciones de Similitud. Modelo de Buckles-Petry

Este modelo propuesto por Buckles y Petry [Buckles, Petry, 82], [Buckles, Petry, 84], define una *Relación Difusa* como un subconjunto del producto cartesiano $(2^{D_1} \times \dots \times 2^{D_m})$, donde 2^{D_i} es cualquier elemento del conjunto de las partes del dominio D_i , $\mathcal{P}(D_i)$. Los tipos de dominios iniciales soportados por el modelo eran:

1. Conjunto finito de escalares. Ej. $DE = \{\text{rubio, castaño, pelirrojo, moreno}\}$.
2. Conjunto finito de números. Ej. $DN = \{15, 16, 17\}$.

Más tarde extendieron el modelo a un tercer tipo de dominio [Buckles, Petry, 84].

3. Conjunto de números difusos. Ej. $DD = \{\text{muy alto, alto, medio, bajo, muy bajo}\}$

La forma de **representar y manejar la imprecisión** se realiza a través de *Relaciones de Similitud* [Zadeh 71] sobre dominios escalares o numéricos. En [Buckles, Petry, 84] proponen una extensión para tratamiento de números difusos.

Mediante las *Relaciones de Similitud* se establece en que medida se parecen entre sí los diferentes valores del dominio sobre el que están definidas. En [Shenoi, Melton, 89] y [Shenoi, Melton, 90] se muestra que se puede suprimir la “transitividad max-min” aunque los dominios no se pueden particionar en clases de equivalencia. Las clases de equivalencia sobre un dominio se construyen a partir de las *Relaciones de Similitud* y de un *umbral de similitud* mínimo establecido en la consulta. Veámoslo mediante un ejemplo: supongamos que tenemos un atributo definido sobre el dominio *color de pelo* compuesto por los valores: {rubio, castaño, pelirrojo, moreno}, y que, sobre los valores que toma este dominio, tenemos definida la *Relación de Similitud* dada en la tabla 3.3.

s	RU	PR	CA	MO
RU	1	.6	.4	0
PR	.6	1	.5	.1
CA	.4	.5	1	.8
MO	0	.1	.8	1

Tabla 3.3: Relación de Similitud sobre el dominio “color de pelo”

Los valores tomados por la *Relación de similitud* deberán ser suministrados por el usuario y ofrecen la medida en que los valores del dominio se parecen según la apreciación del observador. A partir de la *Relación de Similitud* dada, el color de pelo *castaño* presenta un grado de similitud de 0.4 con respecto al color de pelo *rubio*, esto significa que para un umbral de similitud inferior o igual a 0.4 ambos colores son iguales para el observador.

En definitiva, mediante las *Relaciones de Similitud* se construyen clases de equivalencia en cada dominio de forma que los elementos de una misma clase son indistinguibles para el grado de similitud establecido.

En el proceso de consulta el usuario pregunta por aquellas tuplas que satisfacen una determinada condición para unos umbrales de similitud dados. Un sistema realizado según este modelo tomaría la condición de consulta y, mediante las *relaciones de similitud* definidas sobre los dominios y los umbrales de similitud establecidos para cada atributo implicado, agruparía en clases de equivalencia las tuplas de la relación de partida. A continuación operaría sobre las clases con los operadores clásicos del algebra relacional. Siguiendo con el ejemplo, supongamos que la *relación de similitud* dada pertenece al atributo *color de pelo* de una relación o tabla que contiene además información de la *talla* y de la *edad* de un grupo de individuos. Sea pues la relación $R(\text{Nombre}, \text{Edad}, \text{Altura}, \text{Color_de_pelo})$ mostrada en la tabla 3.4. Veamos

Nombre	Edad	Altura	Color_de_pelo
Juan	16	{Muy Alto, Alto}	Castaño
José	17	Bajo	Moreno
Pedro	15	Muy Bajo	Rubio
Mario	{15,16}	Media	Pelirrojo

Tabla 3.4: Relación Persona

como se resolvería la siguiente consulta: “*Dame las personas y su altura con un color de pelo similar a castaño en grado .6*”. Esto implica $Umbral(D_{color_de_pelo}) = .6$ y $Umbral(D_i) = 0 \forall i \neq color_de_pelo$.

- En primer lugar se crean las clases de equivalencia a partir de las relaciones de similitud y de los umbrales de similaridad. Esto da como resultado la relación intermedia mostrada en la tabla 3.5. Obsérvese que los umbrales de similaridad impuestos generan tres clases de equivalencia; una que agrupa a Juan y José, otra a Pedro y, otra que contiene a Mario.
- Después se aplican los operadores relacionales clásicos, en nuestro ejemplo se aplica el operador selección que devuelve el conjunto de las tuplas cuyo color de pelo es similar a castaño en grado .6 y a este conjunto se le aplica la proyección sobre los campos *nombre* y *altura*. El resultado final queda expresado en la tabla 3.6

Nombre	Edad	Altura	Color_de_pelo
{Juan, José}	{16, 17}	{Muy Alto, Alto, Bajo}	{Castaño, Moreno}
Pedro	15	Muy Bajo	Rubio
Mario	{15, 16}	Media	Pelirrojo

Tabla 3.5: Relación intermedia

Nombre	Altura
{Juan, José}	{Muy Alto, Alto, Bajo}

Tabla 3.6: Resultado final de la consulta

En [Buckles, Petry, Sachar, 89] se desarrolla un Cálculo Relacional para este modelo.

Análisis Crítico

1. Los problemas que presenta este modelo residen en los siguientes puntos:
 - No modela bien todos los aspectos difusos de la información. Por ejemplo, no representa bien conceptos lingüísticos tales como *anciano*, *alto*, *mucho*, *rápido*, *etc.*, ya que, aunque puede definir relaciones de similitud en los dominios que los acojan (estas relaciones se construyen estableciendo el grado de similitud entre los diferentes números difusos existentes en el dominio a través del grado de compatibilidad, en el sentido de Zadeh, existente entre ellos [Buckles, Petry, 84]), no resulta práctico cuando se trata de compararlos con los valores precisos del dominio sobre el que están contruidos.
 - La definición de tupla viola la atomicidad en la representación de la información, ya que permite que los atributos que la componen contengan valores que representan subconjuntos del dominio sobre el que están definidos. Además, como resultado de una consulta, se agrupan las tuplas en clases de equivalencia de las cuales se pueden extraer

diferentes interpretaciones. En nuestro ejemplo el resultado final puede sugerir varias interpretaciones:

- {Juan, Muy Alto}
- {José, Alto}
- {Pedro, Muy Alto}. etc.

Todo esto conduciría a asignaciones de valores de atributo erróneas y a una degradación irreversible de la información que se posee, con cada operación que se realice sobre las relaciones originales. Al final no sabemos lo que realmente representa cada tupla.

- Consecuencia de lo anterior es el hecho de que no se pueden definir claves primarias sobre las relaciones. Por tanto resulta difícil garantizar la integridad de una base de datos construida según este modelo.
- Por otro lado si trabajamos sobre las clases de equivalencia que se obtienen de la aplicación de los umbrales de similaridad establecidos en la consulta, las operaciones del Álgebra Relacional actúan correctamente desde el punto de vista matemático, pero el resultado de dichas operaciones puede dar lugar a relaciones que resulten confusas en lugar de difusas. Imagínese el resultado de operar a través de la REUNIÓN sobre dos relaciones cuyas tuplas representen clases de equivalencia. La cuestión en este caso es: ¿Que tuplas reunimos?.

2. El modelo sin embargo presenta dos características interesantes:

- El empleo de *relaciones de similitud* proporciona una herramienta adecuada e intuitiva para representar la imprecisión de los conceptos en el sentido de “proximidad” de los mismos. De hecho en nuestro modelo haremos uso de *relaciones de semejanza* (un tipo de relación difusa que no fuerza la “transitividad max-min”) para modelar la representación de este tipo de información difusa.
- El uso de diferentes umbrales para cada uno de los atributos implicados en una consulta. En nuestro modelo emplearemos un enfoque similar en el sentido de posibilitar el establecer de un grado de cumplimiento diferente para cada atributo implicado en la consulta. Con ello proporcionamos al

usuario un mecanismo con el que establecer que atributos han de satisfacer una consulta en una forma más precisa.

3.5.3 Modelos Relacionales Sobre Distribuciones de Posibilidad

Estos modelos representan la “*ambigüedad*” de los datos mediante el uso de *distribuciones de posibilidad* [Zadeh 78]. Se han propuesto varios modelos relacionales donde los valores de dominio vienen representados por distribuciones de posibilidad. La estructura general de una base de datos perteneciente a este modelo es como sigue.

Una Base de Datos Difusa DB está constituida por un conjunto de relaciones

$$DB = \{R_1, R_2, \dots, R_m\} \quad (3.22)$$

donde cada relación R que contenga distribuciones de posibilidad, (a estas relaciones a veces se les denomina *relaciones extendidas*) se define como:

$$R \subseteq \mathcal{P}(U_1) \times \mathcal{P}(U_2) \times \dots \times \mathcal{P}(U_n) \quad (3.23)$$

donde $\mathcal{P}(U_i)$ representa la familia de todas las distribuciones de posibilidad en el dominio U_i del atributo i -ésimo de la relación R_i y \times es el operador Producto Cartesiano.

A continuación vamos a dar un repaso a los diferentes modelos aparecidos al amparo de este enfoque. Las diferencias que presentan estos modelos entre sí radican básicamente en los siguientes aspectos:

- Representación de las distribuciones de posibilidad y manejo de información desconocida, indefinida y nula.
- Planteamiento de la consulta.
- Elección de los operadores de selección con los que evaluar las condiciones impuestas en una consulta.

3.5.3.1 Modelo de Umano y Fukami

Umano et al fueron los primeros en proponer el modelo de bases de datos relacional de distribuciones de posibilidad y el metodo de recuperación correspondiente [Umano 82], [Fukami, et al, 79]. El lenguaje de manipulación de datos está implementado mediante un sistema de manipulación de conjuntos difusos llamado FSTDS [Umano 77], [Umano 78].

- **Representación de la información**

Veamos mediante un ejemplo tomado de [Fukami, et al, 79] como se representa la información en este modelo. Sea la relación *Persona(Nombre, Edad, Nombre_hijos)* mostrada en la tabla 3.7.

Nombre	Edad	Nombre_hijos
Tom	23	Ted
Susan	35	John
Susan	35	Mike
Richard	40	{Judy, Anna} _p
Raymond	joven	unknown
Victor	unknown	undefined
Smith	{50, 51} _p	NULL

Tabla 3.7: Ejemplo de relación para el modelo de Umano et al.

El nombre de los hijos de Richard {Judy, Anna}_p es una distribución de posibilidad, indicando que el valor puede ser Judy o Anna (si tuviera dos hijos estos aparecerían dos veces como ocurre con Susan). La edad de Raymond es la distribución de posibilidad que se corresponde con la etiqueta joven, la cual, por ejemplo, puede venir definida por:

$$\text{joven} = \{0.3/15, 0.6/16, 0.8/17, 1/18, 1/19, 1/20, 1/21, 1/22, 1/23, 0.9/24, \\ 0.8/25, 0.7/26, 0.5/27, 0.3/28, 0.1/29\}_p$$

Unknown (desconocido), undefined (indefinido) y NULL (cuando no sabemos si un valor está indefinido o es desconocido) vienen definidos también por distribuciones de posibilidad:

$$\begin{aligned}\text{unknown} &= \{1/u : u \in U\}_p \\ \text{undefined} &= \{0/u : u \in U\}_p \\ \text{NULL} &= \{1/\text{unknown}, 1/\text{undefined}\}_p\end{aligned}$$

los cuales son procesados en forma específica por la implementación del sistema.

- **Planteamiento de la consulta**

El modelo resuelve la consulta, expresada en términos difusos o precisos, dividiendo en tres subconjuntos las relaciones implicadas en la misma: un primer subconjunto contiene las tuplas que claramente satisfacen la consulta, un segundo conjunto agrupa las tuplas que posiblemente satisfacen la misma y el tercero contiene las tuplas que claramente no la satisfacen. Veamos dos ejemplos de consulta sobre la relación anterior. Para la consulta, expresada en términos precisos, *Dame las personas que tienen más de 25 años*, la clasificación que se obtendría sería:

$$\begin{aligned}\text{claramente satisfacen} &= \{1/\text{Susan}, 1/\text{Richard}, 1/\text{Smith}\} \\ \text{posiblemente satisfacen} &= \{1/\text{Raymond}, 1/\text{Victor}\} \\ \text{claramente no satisfacen} &= \{1/\text{Tom}\}\end{aligned}$$

como la consulta no es difusa los grados son 0 ó 1. Los datos que claramente no satisfacen la consulta no son devueltos.

Una consulta difusa como *Encontrar las personas “jóvenes”* precisa de la definición previa del predicado “joven”, que es diferente de la distribución de posibilidad “joven” incluida en la relación. En nuestro ejemplo supongámoslo definido como:

$$\text{joven} = \{1/19, 1/20, 1/21, 1/22, 0.8/23, 0.6/24, 0.3/25\}$$

El resultado es:

$$\begin{aligned}\text{claramente satisfacen} &= \{0.8/\text{Tom}\} \\ \text{posiblemente satisfacen} &= \{1/\text{Raymond}, 1/\text{Victor}\}\end{aligned}$$

Para Raymond y Victor se emplea el máximo grado de “Joven” de entre los disponibles. La distribución de posibilidad que representa la edad de Raymond proporciona un valor de posibilidad 1 para las edades 18,19,20,21,22,23 el mismo que tiene asociado el predicado “joven” a las edades 19,20,21,22, como hay coincidencia al menos para un valor de ambas distribuciones de posibilidad y el grado de posibilidad máximo es 1, entonces Raymond posiblemente satisface la consulta con grado 1. En el caso de Victor ocurre que como tiene una edad desconocida posiblemente satisface la consulta.

• Operadores de selección

Este modelo asocia a cada ítem envuelto en una consulta un valor de verdad constituido por un par de elementos, (denominado par de verdad), $\langle c, t \rangle$. El factor c toma el valor T para aquellos ítems que reflejan certeza y toma el valor P para los que reflejan posibilidad, el factor t asocia el valor de verdad en el sentido de la lógica multivaluada y toma valor dentro del intervalo $[0,1]$.

El procedimiento para evaluar una proposición que contenga predicados difusos es como sigue. Sean t_1, t_2, \dots, t_n los valores de verdad obtenidos de evaluar el predicado para todas las combinaciones posibles de elementos u_x y u_y obtenidas de las distribuciones de posibilidad $\Pi_{A(x)}$ y $\Pi_{A(y)}$.

- Si $t_1 = t_2 = \dots = t_n$ entonces el sistema evalúa la proposición sobre los datos dados por las distribuciones de posibilidad $\Pi_{A(x)}$ y $\Pi_{A(y)}$ como **cierta** y le asocia un par de verdad $\langle T, t_1 \rangle$.
- Si no es así, entonces el sistema evalúa la proposición sobre los datos dados por las distribuciones de posibilidad $\Pi_{A(x)}$ y $\Pi_{A(y)}$ como **posible** y le asocia un par de verdad $\langle P, \max(t_1, t_2, \dots, t_n) \rangle$

(3.24)

La condición que se establece en una consulta puede estar compuesta de varias condiciones atómicas. Cuando la condición es compuesta se evalúa mediante los conectivos conjunción, disyunción y negación, que se definen como sigue:

1. *Conjunción* \wedge

$\langle c_1, t_1 \rangle \wedge \langle c_2, t_2 \rangle = \langle \min(c_1, c_2), \min(t_1, t_2) \rangle$
donde $\min(T, T) = T$, $\min(T, P) = P$ $\min(P, P) = P$.

2. *Disyunción* \vee

(a) Para $c_1 = c_2$, tenemos

$$\langle c_1, t_1 \rangle \vee \langle c_2, t_2 \rangle = \langle c_1, \max(t_1, t_2) \rangle$$

(b) Para $c_1 \neq c_2$, si $c_1 = t$ y $c_2 = P$, tenemos

i. si $t_1 \geq t_2$, o $t_1 < t_2$ y $t_1 \geq 0.5$ entonces

$$\langle T, t_1 \rangle \vee \langle P, t_2 \rangle = \langle T, t_1 \rangle$$

ii. si no,

$$\langle T, t_1 \rangle \vee \langle P, t_2 \rangle = \langle P, t_2 \rangle$$

3. *Negación* \neg

$$\neg \langle T, t \rangle = \langle T, 1 - t \rangle$$

$$\neg \langle P, t \rangle = \langle P, 1 \rangle$$

El modelo contempla los siguientes predicados:

- **SETEQ, DISJOINT, CONTAIN**. Los cuales operan sobre dos conjuntos difusos S_1 y S_2 devolviendo un par de verdad $\langle T, 1 \rangle$ o $\langle T, 0 \rangle$ en función de que se cumpla que $S_1 = S_2$, $S_1 \cap S_2 = \emptyset$ y $S_1 \supseteq S_2$, respectivamente.
- **EQ, GE, GT**. Estos predicados pueden operar sobre valores precisos o sobre conjuntos difusos. En el caso de que operen sobre dos valores precisos, x e y , devolverán un par de verdad $\langle T, 1 \rangle$ o $\langle T, 0 \rangle$ según se cumpla o no $x = y$, $x \geq y$ $x > y$, respectivamente. Cuando estos predicados operan sobre datos precisos y difusos pueden devolver los pares de verdad $\langle T, 1 \rangle$, $\langle P, 1 \rangle$ o $\langle T, 0 \rangle$ caso de que el predicado correspondiente se satisfaga estrictamente, se pueda satisfacer o no sea satisfecho en modo alguno, respectivamente.
- **FCONT, FEQ**. El primer operador aplicado sobre dos conjuntos difusos F_1 y F_2 nos devuelve un par de verdad $\langle T, t \rangle$ donde t , que se denomina *compatibilidad*, refleja el grado de inclusión de un conjunto en otro y se calcula mediante la expresión:

$$FCONT(F_1, F_2) = \frac{\sum_i (\mu_{F_1}(u_i) \wedge \mu_{F_2}(u_i))}{\sum_i \mu_{F_1}(u_i)} \quad (3.25)$$

La compatibilidad de $FEQ(F_1, F_2)$ se define como el mínimo de las compatibilidades $FCONT(F_1, F_2)$ y $FCONT(F_2, F_1)$.

3.5.3.2 Modelo de Prade-Testemale

La estructura de datos sobre la que opera este modelo es parecida a la adoptada por el modelo de Umano et al, con la salvedad de la definición específica del dominio vista en el apartado tercero. Sin embargo, este modelo emplea medidas de posibilidad y de necesidad para la satisfacción de las condiciones establecidas en la consulta [Prade, Testemale, 84]. El lenguaje de manipulación de datos está escrito en MACLISP sobre un DPS8. Veamos que particularidades presenta en los diferentes aspectos.

- **Representación de la Información**

Analizaremos la capacidad que presenta este modelo para representar información difusa mediante un ejemplo. Consideremos la relación *Estudiantes* de la tabla 3.8 donde tenemos representados los mismos con su nombre, edad y sus calificaciones en Matemáticas y Física (en un rango entre 0 y 20):

Nombre	Edad	Matemáticas	Física
Tom	joven	16	[14,16]
David	20	ligeramente mal	no aplicable
Bob	22	no muy mal	bueno
Jane	sobre 21	casi bueno	[10,12]
Jill	joven	alrededor de 10	alrededor de 12
Joe	sobre 24	[14,16]	15
Jack	[22,25]	desconocido	casi bueno

Tabla 3.8: Ejemplo de relación para el modelo de Prade-Testemale

En esta relación [12,15] y [14,16] representan intervalos de valores y “joven”, “bueno”, etc. son distribuciones de posibilidad previamente definidas. El concepto de “desconocido” y el de “no aplicable” es el mismo visto en el apartado tercero.

- **Planteamiento de la consulta**

Para resolver la consulta “encontrar los nombres de las personas cuya clasificación en Matemáticas es mucho mayor que buena”, tendríamos que partir de una definición de la distribución de posibilidad que representa a una calificación “buena”. Después, el modelo precisa de una definición para el comparador difuso *mucho mayor que*. En nuestro ejemplo tomaremos las siguientes:

$$buena = \{0.3/13, 1/14, 1/15, 1/16, 0.5/16.5\}_p$$

$$\mu_{mucho_mayor_que}(u, v) = \begin{cases} 0 & u - v \leq 2 \\ 0.5 & u - v = 3 \\ 1 & u - v \geq 4 \end{cases}$$

En primer lugar el modelo resuelve la composición del comparador difuso *mucho mayor que* con la distribución de posibilidad dada por *buena*, esto se realiza utilizando la regla de composición dada en [Zadeh 75]. Según esta regla, la composición de un subconjunto difuso A dado en el universo de discurso U y un operador relacional difuso R sobre $U \times V$ viene dada por:

$$A \circ R = \{\max_{u \in U}(\mu_A(u) \wedge \mu_R(u, v)) / v : v \in V\}$$

El resultado de aplicar la composición del comparador difuso *mucho mayor que* con la distribución de posibilidad dada por *buena* es:

$$\begin{aligned} F &= mucho_mayor_que \circ buena \\ &= \{0.3/16, 0.5/17, 1/18, 1/19, 1/20\}_p \end{aligned}$$

después recupera de la relación las tuplas que satisfacen la consulta agrupadas en dos conjuntos dados por las medidas de *necesidad* y de *posibilidad*. La medida de *posibilidad* viene dada por la expresión:

$$\Pi(F | A) = \sup_{u \in D} \{\mu_F(u) \wedge \pi_A(u)\}$$

donde D es el dominio de F y de A . Esta ecuación representa el grado máximo en la intersección de F y A . El conjunto de posibilidad devuelto para la consulta dada es:

$$\Pi = \{0.3/Tom, 0.3/Joe, 1/Jack\}$$

Por otro la medida de *necesidad* se calcula:

$$N(F | A) = \inf_{u \in D} \{\mu_F(u) \vee (1 - \pi_A(u))\}$$

que representa el grado mínimo de la unión de F y el complemento de A , lo cual nos proporciona para la consulta un conjunto de *necesidad*:

$$N = \{0.3/Tom\}$$

• Operadores de selección

La SELECCIÓN relacional difusa está modelada como sigue. La condición que han de satisfacer las tuplas para ser recuperadas a partir de una consulta puede ser atómica o compuesta. Esta última consta de varias condiciones atómicas unidas por los conectivos AND (\wedge), OR (\vee) y NOT (\neg), dados por la T-norma del mínimo, la T-conorma del máximo y por el complemento de la función de pertenencia respectivamente.

En cuanto a las condiciones atómicas el modelo distingue entre los siguientes tipos:

- $A \Theta a$
- $A \Theta B$

donde A y B representan sendos atributos a comparar, a es una constante (difusa o no) y Θ es un comparador (difuso o clásico) que en cualquier caso viene dado por una función de pertenencia μ_Θ definida sobre el Producto Cartesiano de dos dominios y tomando valores en el intervalo $[0,1]$, y con el que se pueden modelar comparadores difusos como “*aproximadamente igual*” y “*mucho mayor que*”.

Veamos los dos casos (en lo que sigue el símbolo $|$ no denota opción entre dos alternativas, sino que separa el componente “consulta”, de los items de la base de datos):

- $A\Theta a$.

La *posibilidad* de que el valor de atributo A para el objeto x caiga en el conjunto de elementos que están en relación Θ con al menos un elemento de a viene dada

por:

$$\Pi(a \circ \Theta \mid A(x)) = \sup_{d \in D} \min(\mu_{a \circ \Theta}(d), \pi_{A(x)}(d)) \quad (3.26)$$

con

$$\mu_{a \circ \Theta}(d) = \sup_{d' \in D} \min(\mu_{\Theta}(d, d'), \mu_a(d')) \quad (3.27)$$

donde D es el dominio del atributo A , μ_{Θ} es la función de pertenencia (difusa o no) definida sobre $D \times D$, y $\pi_{A(x)}$ es la distribución de posibilidad que restringe para x los posibles valores del Atributo A .

La necesidad del mismo evento viene dada por:

$$N(a \circ \Theta \mid A(x)) = \inf_{d \in D} \max(\mu_{a \circ \Theta}(d), 1 - \pi_{A(x)}(d)) \quad (3.28)$$

- $A\Theta B$.

De forma análoga, la *posibilidad (necesidad)* de que el valor de atributo A para el objeto x esté en relación Θ con el valor de atributo B para el mismo objeto x viene dada por:

$$\Pi(\Theta \mid (A(x), B(x))) = \sup_{(d, d') \in D \times D} \min(\mu_{\Theta}(d, d'), \pi_{(A(x), B(x))}(d, d')) \quad (3.29)$$

$$N(\Theta \mid (A(x), B(x))) = \inf_{(d, d') \in D \times D} \max(\mu_{\Theta}(d, d'), 1 - \pi_{(A(x), B(x))}(d, d')) \quad (3.30)$$

para atributos no interactivos (que no exista dependencia de los valores que toma uno con respecto a los que toma el otro) la expresión queda:

$$\Pi(\Theta \mid (A(x), B(x))) = \sup_{(d, d') \in D \times D} \min(\mu_{\Theta}(d, d'), \pi_{A(x)}(d), \pi_{B(x)}(d')) \quad (3.31)$$

$$N(\Theta \mid (A(x), B(x))) = \inf_{(d, d') \in D \times D} \max(\mu_{\Theta}(d, d'), 1 - \pi_{A(x)}(d), 1 - \pi_{B(x)}(d')) \quad (3.32)$$

3.5.3.3 Modelo de Zemankova-Kaendel

Este modelo de base de datos, (en [Zemankova 84], [Zemankova 85]), se compone de tres partes: (a) una base de datos de valores (VDB) donde se organizan los datos en forma similar al resto de los modelos posibilísticos, (b) una base de datos explicativa, donde se almacenan las definiciones para los subconjuntos difusos y relaciones difusas y (c) un conjunto de reglas de traducción, que se emplean para el manejo de adjetivos, etc. La manipulación de datos está basada en el Álgebra Relacional. Este lenguaje está implementado en una forma extendida del sistema RIM (Relational Information Management) desarrollado por la Boeing Co.

- **Planteamiento de la consulta**

Este modelo plantea la recuperación en una consulta en forma similar al de Prade-Testemale solo que la medida de **posibilidad** que emplea para encontrar la compatibilidad del subconjunto difuso F de la condición con el valor del atributo A para cada tupla en la relación viene dada por:

$$p_A(F) = \sup_{u \in D} \{\mu_F(u) \cdot \pi_A(u)\} \quad (3.33)$$

y la medida de **certeza**

$$c_A(F) = \max_{u \in D} \{0, \inf\{\mu_F(u) \cdot \pi_A(u)\} > 0\} \quad (3.34)$$

es usada en lugar de la medida de necesidad de Prade-Testemale.

El resultado de una consulta se presenta en forma de relaciones difusas las cuales contienen dos campos en los que se recojen los valores de **posibilidad** y **certeza** que presenta cada tupla para la consulta dada. Sobre estas relaciones se pueden establecer unos umbrales mínimos a satisfacer para las tuplas que se recuperen.

- **Operadores de Selección**

En el estudio que hacen de las condiciones impuestas en la selección parten de una relación Θ de similaridad sobre $D \times D$ y a partir de ella construyen cualquier otra relación de comparación [Zemankova 84]. Dan tres ejemplos: “*aprox. igual*”, “*mayor que*” y “*menor que*”. Si s es la relación de similitud de la que se parte la

relación “*mayor que*” se construiría:

$$\mu_{mayor_que}(x, y) = \begin{cases} 1 - 0.5 \cdot s(x, y) & \text{si } x \geq y \\ 0.5 \cdot s(x, y) & \text{si } x < y \end{cases} \quad (3.35)$$

Las correspondientes medidas de posibilidad se computarían utilizando la expresión (3.33) lo que daría para condiciones de tipo $A\Theta f$ con f atómico:

$$p_{\Theta f}(A(x)) = \sup_{d \in D} \{\mu_{\Theta}(d, f) \cdot \pi_{A(x)}(d)\} \quad (3.36)$$

para condiciones del tipo $A\Theta F$ donde F no sea atómico, no da ninguna regla de cálculo.

Comentarios

Los modelos analizados a lo largo de este subapartado presentan una serie de propiedades que, a juicio del autor de esta memoria, resultan interesantes a la hora de encontrar un modelo de bases de datos relacional difuso. Dichas propiedades son:

- La forma de representar la información difusa mediante el empleo de dominios contruidos sobre distribuciones de posibilidad parece una forma apropiada para captar el significado de una amplia variedad de este tipo de información. De todas formas existe cierto tipo de información difusa, que, si bien puede representarse de acuerdo con los esquemas propuestos por los modelos posibilísticos, resulta mas natural recurrir al empleo de relaciones de similitud o de semejanza para modelarla, (p.e., resulta más natural establecer una relación de semejanza sobre el dominio que subyace al atributo `color_de_pelo` de la tabla 3.4, que construir distribuciones de posibilidad sobre dicho dominio). En el modelo que se propondrá en el próximo capítulo serán integradas ambas herramientas de representación de información difusa.
- La extensión del concepto de relación adoptada por las aproximaciones analizadas es, salvo matices, similar. Al respecto, el modelo que será propuesto en esta memoria, incorporará el concepto de “atributos de compatibilidad”, que, como se verá, servirá de gran ayuda para el seguimiento de las operaciones de manipulación.

- Con respecto a la manipulación de la información difusa, cada aproximación aporta una perspectiva diferente basada cada una de ellas en la concepción empleada para evaluar la medida en que cada par de datos difusos se encuentran en correspondencia con respecto a una propiedad dada. Habida cuenta la disparidad semántica de la información difusa que un SBDRD ha de tratar, nos parece limitado restringir el aspecto manipulativo a una única aproximación, por este motivo abogaremos por un modelo que proporcione una gran variedad de mecanismos para acometer las tareas de manipulación de acorde con las características semánticas de la información.

En definitiva, el modelo de bases de datos difusas que se presentará en el próximo capítulo surge con una vocación integradora de las aproximaciones visitadas a lo largo de este apartado. Además se incluyen algunas características importantes, tanto estructurales, como de manipulación, con objeto de proporcionar un marco efectivo para el desarrollo de SBDRD operativos.

Capítulo 4

GEFRED. Modelo Generalizado de Bases de Datos Relacionales Difusas

4.1 Introducción

En este capítulo vamos a introducir los elementos principales del modelo teórico de Bases de Datos Relacionales Difusas sobre el se sustentan los desarrollos específicos de software que presentaremos en esta memoria. GEFRED, (A GEneralized model for Fuzzy RElational Databases), [Medina et al, 93(b)], como lo denominaremos en lo sucesivo, representa una síntesis entre las diferentes tendencias aparecidas para abordar el problema de representación y tratamiento de información difusa mediante Bases de Datos Relacionales. Se ha realizado un esfuerzo de abstracción y de generalización tendente a integrar en un mismo desarrollo enfoques que aparecían en principio como dispares, todo ello con el único fin de facilitar un soporte coherente para la implementación de Sistemas de Bases de Datos Relacionales Difusas. Hay que entender el modelo que vamos a presentar como un *“entorno teórico sobre el que articular las diferentes aproximaciones aparecidas para resolver el problema del tratamiento de la información difusa en el ámbito de las bases de datos relacionales”*. Es importante tener presente que en la actualidad no se encuentran resueltos de forma satisfactoria todos los aspectos del tratamiento de la información difusa desde la perspectiva relacional. Existen problemas para los que han aparecido diferentes soluciones parciales

que sacrifican algunos aspectos en beneficio de otros. Es precisamente la naturaleza difusa de la información a tratar la que plantea mayores dificultades en el tema de las operaciones de selección, en el problema de la Reunión (natural y externa), en el problema de la descomposición (sin pérdidas), en el problema de la normalización y en el de la redundancia. Bajo esta perspectiva, la filosofía que adopta este modelo es la de aglutinar en torno a un esquema común la mayoría de las propuestas contrastadas, dejando margen para que se delimiten algunos aspectos del modelo o para que se incorporen otros nuevos mediante propuestas teóricas específicas. En este sentido, algunas de las características del modelo no se encuentran totalmente cerradas, tal es el caso del concepto de duplicidad, de la forma concreta que adoptan los *comparadores difusos generalizados* así como las variantes más conflictivas de la Reunión. Como contrapartida, el modelo presenta la ventaja de que, prácticamente cualquier aproximación teórica a los problemas anteriores puede particularizarse de forma sencilla dentro del modelo.

Las líneas básicas que sustentan GEFRED se desglosan en los apartados relativos a la estructura de datos y a la manipulación de los mismos.

4.2 Estructura de Datos

Uno de los objetivos del modelo es el de proporcionar una amplia cobertura, en cuanto a la representación y tratamiento, de información difusa de toda índole. De acuerdo con esto, los tipos de datos que deberá ser capaz de representar se relacionan en la tabla 4.1.

Los elementos del modelo que estructuran toda esta información son el *Dominio Difuso Generalizado* y la *Relación Difusa Generalizada*.

4.2.1 Dominio Difuso Generalizado, D_G

El Dominio Difuso Generalizado constituye el elemento estructural sobre el que se articulará la representación de los datos recogidos en la tabla 4.1.

Definición 4.1 Si D es el dominio de discurso, $\tilde{\mathcal{P}}(D)$ el conjunto de todas las distribuciones de posibilidad definidas sobre D , incluidas las que definen los tipos *Unknown* y *Undefined*, (Tipos 8 y 9 resp. de la tabla 4.1), y *NULL* el tipo cuya definición

1. Un escalar simple (Ej. Aptitud=buena, representado mediante la distribución de posibilidad $1/\text{buena}$).
2. Un número simple (Ej. Edad=28, representado mediante la distribución de posibilidad $1/28$).
3. Un conjunto de posibles asignaciones excluyentes de escalares (Ej. Aptitud= {mala,buena}, se expresa $\{1/\text{mala},1/\text{buena}\}$).
4. Un conjunto de posibles asignaciones excluyentes de números (Ej. Edad= {20,21}, representado mediante $\{1/20,1/21\}$)
5. Una distribución de posibilidad en el dominio de los escalares (Ej. Aptitud= {0.6/mala,0.7/regular}).
6. Una distribución de posibilidad en el dominio de los números (Ej. Edad= {0.4/23, 1.0/24, 0.8/25}, números difusos, etiquetas lingüísticas)
7. Un número Real $\in [0, 1]$ representando grados de cumplimiento (Ej. Calidad=0.9).
8. Un valor desconocido **Unknown** dado por la distribución de posibilidad $\mathbf{Unknown}=\{1/u : u \in U\}$ sobre el dominio considerado.
9. Un valor indefinido **Undefined** dado por la distribución de posibilidad $\mathbf{Undefined}=\{0/u : u \in U\}$ sobre el dominio considerado.
10. Un valor nulo **NULL** dado por la expresión, $\mathbf{NULL}=\{1/\text{Unknown},1/\text{Undefined}\}$

Tabla 4.1: Tipos de datos

aparece en la citada tabla, llamaremos **Dominio Difuso Generalizado**, D_G , a $D_G \subseteq \tilde{\mathcal{P}}(D) \cup NULL$.

La tabla 4.1 sintetiza el tipo de información para la que el modelo puede proporcionar tratamiento. Esa información cae dentro de alguna de las siguientes categorías:

- **Información precisa.** En este sentido el modelo conserva la capacidad para representar y manipular información precisa en los mismos términos en que lo hace el modelo relacional clásico. De acuerdo con su naturaleza podemos clasificar la información de tipo preciso en dos grandes categorías: información cuyo origen o representación es numérico como son los dominios definidos sobre subconjuntos de los enteros o las representaciones parciales de dominios contruidos sobre la recta real, y por otro lado, información escalar, que permite representar conocimiento cuyo carácter no es numérico aunque en algún caso pudiera serlo su representación. Esta última categoría permite representar una amplia variedad tipos de información. El Dominio Difuso Generalizado no plantea límites en cuanto a las implicaciones semánticas que pudieran derivarse de la definición de un dominio preciso, cuya naturaleza necesite de la definición de una serie de operaciones especiales, a realizar sobre los elementos de dicho dominio. En el marco en el que se formula GEFRED es posible, por tanto, la definición de los tipos de datos extendidos, [Codd 90], en el sentido de incorporar más significado a la representación de los datos. De esta forma disponemos de potentes herramientas para preservar la integridad a nivel de dominio, podemos así mismo, restringir las operaciones realizables sobre un dominio determinado y, en definitiva, poseemos mayor capacidad para dotar a la representación de los datos de una semántica más próxima a la realidad que modelan. En cualquier caso, todas estas capacidades de representación encontrarán en el *Comparador Difuso Generalizado* el mecanismo mediante el que modelar su actuación. Dicho comparador será definido más adelante.
- **Información con proximidad.** El ámbito impreciso sobre el que opera el modelo, pretende dar cabida a aquel tipo de información que recoge los datos como apreciaciones subjetivas de la realidad del universo a representar. Estas apreciaciones se obtienen de dominios, generalmente escalares, contruidos sobre conceptos próximos o adyacentes en cuanto a su significado. Frecuentemente este

tipo de información responde a percepciones de la realidad cuya magnitud no puede ser establecida en forma absoluta. Un ejemplo de este tipo de información puede ser el color del cabello de un individuo. Además de ser éste un concepto relativo, su magnitud no puede ser establecida en forma absoluta, sin embargo, podemos encontrar cierto grado de parecido entre algunos de los posibles valores que podría adoptar, un pelo castaño oscuro encierra algún parecido con un cabello moreno. Como veremos más adelante, esta proximidad entre los valores de dominio, puede ser modelada mediante el empleo de relaciones de semejanza definidas sobre los valores de dominio.

- **Información Imprecisa.** Cuando nosotros suministramos al sistema un dato que cae dentro de la categoría de información precisa, no solo proporcionamos un valor de dominio concreto, si no que además, indirectamente, estamos aportando el grado con el que conocemos ese dato. En el caso de la información precisa, el conocimiento sobre que el dato toma ese valor es total, por lo que el grado de conocimiento ha de ser máximo. Cuando nuestro conocimiento de la realidad no permite o no necesita determinar un valor preciso de entre los posibles del dominio, podemos modelar dicho conocimiento mediante el empleo de una colección de valores de dominio junto con los respectivos grados en que pensamos que cada valor es compatible con la información que queremos representar. El anterior concepto puede modelarse mediante el empleo de las distribuciones de posibilidad. Este será el mecanismo con el que representaremos el conocimiento impreciso en el ámbito del modelo que estamos definiendo. La forma adoptada para representar la imprecisión se construye sobre un universo de discurso que puede pertenecer a cualquiera de las categorías anteriores.
- **Información no proporcionada.** Cuando no suministramos un valor a un atributo de una tupla determinada puede ser debido a varias razones. De todas ellas, este modelo representa tres: que desconozcamos completamente el valor que toma dicha información pero que sepamos que puede tomar alguno, que conozcamos que dicho atributo no puede adoptar ningún valor y, que desconozcamos completamente el valor que toma dicha información y si puede tomar alguno. Los dos primeros casos pueden ser modelados como casos particulares de distribuciones de posibilidad como puede verse en los tipos 8 y 9 de la tabla 4.1, el último se define como una distribución de posibilidad construida sobre los tipos

anteriores (tipo 10). La filosofía de representación para estos datos se corresponde con la adoptada en [Umano 82].

4.2.2 Relación Difusa Generalizada, R_{FG}

La *Relación Difusa Generalizada* representa el elemento estructural principal mediante el cual podemos organizar la información con todas sus connotaciones. En la siguiente definición tratamos de recoger los diferentes aspectos que confluyen en esta estructura de representación.

Definición 4.2 Una *Relación Difusa Generalizada*, R_{FG} , viene dada por un par de conjuntos “cabecera”, (\mathcal{H}) y “cuerpo”, (\mathcal{B}) , $R_{FG} = (\mathcal{H}, \mathcal{B})$ definidos como sigue:

- La “cabecera” es un conjunto fijo de ternas atributo-dominio-compatibilidad,

$$\mathcal{H} = \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), (A_{G2} : D_{G2} [, C_{A_{G2}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \quad (4.1)$$

donde a cada atributo A_{Gj} , le subyace un dominio difuso generalizado, no necesariamente distinto, D_{Gj} ($j=1, 2, \dots, n$) y $C_{A_{Gj}}$ es un “atributo de compatibilidad” que toma valores en el intervalo $[0, 1]$.

- El “cuerpo” consiste en un conjunto de tuplas difusas generalizadas “distintas”, donde cada tupla está compuesta por un conjunto de ternas atributo-valor-grado,

$$\mathcal{B} = \{(A_{G1} : \tilde{d}_{i1} [, c_{i1}]), (A_{G2} : \tilde{d}_{i2} [, c_{i2}]), \dots, (A_{Gn} : \tilde{d}_{in} [, c_{in}])\} \quad (4.2)$$

con ($i = 1, 2, \dots, m$, siendo m el número de tuplas de la relación), donde \tilde{d}_{ij} representa el valor de dominio que toma la tupla i sobre el atributo A_{Gj} y c_{ij} el grado de compatibilidad asociado a este valor.

Observaciones:

1. Los corchetes que encierran a $C_{A_{Gj}}$ y a c_{ij} denotan un caracter opcional para el atributo de compatibilidad y, por tanto, para los grados de compatibilidad, en cuanto a su aparición, de forma explícita, en la cabecera y cuerpo de la relación

respectivamente. Ello es debido a que dichos atributos, y sus correspondientes grados, son generados en los procesos de manipulación sobre las relaciones. Una R_{FG} que unicamente contenga información de partida, (una relación “base”), no presenta atributos de compatibilidad.

2. Un valor 0 para un grado de compatibilidad en una tupla l será interpretado como que, para esa tupla y para el atributo asociado, las operaciones de manipulación efectuadas sobre dicho atributo no han superado los umbrales establecidos sobre las mismas. Ello no implica que dicha tupla no haya satisfecho globalmente la operación formulada sobre la relación en cuestión.
3. La definición de *Relación Difusa Generalizada* contempla las relaciones clásicas como un caso particular, donde los dominios clásicos sobre los que se construyen los atributos que constituyen la *cabecera* de la relación, son casos particulares de la definición de *Dominio Difuso Generalizado* y donde no aparecen los *atributos de compatibilidad* en la *cabecera* ni, por tanto, los *grados de compatibilidad* en el *cuerpo*.

4.2.2.1 Significado de la Relación Difusa Generalizada

Como ocurriera con el concepto de *relación* en el modelo relacional, el concepto de *Relación Difusa Generalizada* se revela como particularmente importante en tanto que constituye el principal mecanismo para representar la información en todas sus vertientes. Por ello vamos a detenernos a examinar los principales aspectos semánticos que concurren en su definición.

- El *Relación Difusa Generalizada* es la única estructura proporcionada por el modelo para representar la información de partida en la base de datos, (relaciones base), la información recuperada de la misma mediante un proceso de manipulación, así como la información intermedia generada en dicho proceso.
- La *cabecera* recoge basicamente la estructura de la relación (R_{FG}). Cada atributo de la relación supone la instanciación de un *dominio difuso generalizado* con todas las características que lleve asociado dicho dominio (datos y operaciones). Así mismo, cada atributo puede llevar asociado un atributo de compatibilidad definido sobre el intervalo unidad. Con este atributo el modelo pro-

porciona un mecanismo uniforme para registrar las operaciones de manipulación sobre la relación. Almacenará los estados parciales y finales consecuencia de las diferentes operaciones realizadas sobre la relación.

- El *cuervo* constituye la información presente en la relación. Como vimos anteriormente, la información de partida, la que se recupera mediante procesos de manipulación, así como la intermedia generada durante dichos procesos, pueden constituir el contenido del *cuervo* de una relación. El *grado de compatibilidad* es un valor de dominio para cada tupla que puede ser asociado a cada atributo como consecuencia de los procesos de manipulación realizados sobre la relación. Por tanto, solo tiene sentido su empleo en los procesos de manipulación, tanto para presentar los resultados finales, como para almacenar los resultados intermedios generados en el proceso. Dicho de otra forma las relaciones de partida no poseen atributos de compatibilidad, siendo dichos grados generados durante los procesos de manipulación. De todas formas, nada impide que podamos almacenar en una relación base los grados devueltos por un proceso de manipulación empleando para ello atributos compatibles con los valores obtenidos mediante dicho proceso.
- El modelo no precisa explícitamente el concepto de “tuplas difusas generalizadas distintas”. Este aspecto, que nos remite al concepto de redundancia, es básico para preservar el espíritu del modelo relacional. Existen algunos términos del modelo, como la definición de clave primaria, el estudio de las dependencias funcionales, la normalización y la descomposición sin pérdidas, cuyos fundamentos descansan sobre el concepto de redundancia. Desde el punto de vista “difuso”, el problema de considerar cuando dos tuplas son redundantes es más complejo que cuando los valores a manejar son precisos. El origen del problema está en cuando considerar que dos conjuntos difusos son iguales, más concretamente, en que entendemos por igualdad entre dos conjuntos difusos. Estrictamente hablando, como vimos el capítulo anterior, dos conjuntos difusos son iguales ($\tilde{A} = \tilde{B}$), sii

$$\forall x \in \Omega, \mu_{\tilde{A}}(x) = \mu_{\tilde{B}}(x) \quad (4.3)$$

En base a este criterio, dos tuplas serían iguales y, por lo tanto habría que eliminar una de ellas, si, para cada valor de dominio, fueran iguales en el sentido anterior. No obstante, este criterio de redundancia puede parecer poco restrictivo para

gran parte de los casos. Cuando hemos de extender conceptos cuyo origen es preciso, como es el caso de la igualdad, al campo de los conjuntos difusos, una de las líneas frecuentemente empleadas es la de considerar la compatibilidad entre conjuntos difusos. Existen varios criterios de compatibilidad entre conjuntos difusos cada uno de los cuales tiene sus ventajas y sus inconvenientes. Algunos autores [Prade, Testemale, 87b] basan su concepto de redundancia en uno de estos criterios de compatibilidad entre conjuntos difusos. En este sentido establecen que dos distribuciones de posibilidad π y π' sobre D son aproximadamente iguales si

$$\sup_{d \in D} |\pi(d) - \pi'(d)| \leq \varepsilon_D \quad (4.4)$$

donde ε_D es un umbral dependiente del dominio. De esta forma afirman que dos distribuciones de posibilidad son aproximadamente iguales si son compatibles en un grado superior a un valor $1 - \varepsilon_D$ establecido sobre el dominio en que se encuentran definidas. En base a esto dicen que dos tuplas son redundantes si encontramos que son aproximadamente iguales para cada componente. El problema que presenta este criterio de compatibilidad es que no cumple la propiedad transitiva, por lo que no define una equivalencia. En [Buckles, Petry, 82] se pueden encontrar otras aproximaciones al problema dentro del modelo propuesto por estos autores.

Actualmente, [Cubero, Medina, Vila 93], [Cubero, Pons, Vila 94], se encuentra en estudio por miembros de nuestro grupo de investigación, una propuesta teórica destinada a paliar algunos de los problemas que presentan las dependencias funcionales difusas, las operaciones de normalización y las descomposiciones.

La aproximación plantea una solución al problema, basada en el empleo de dos conceptos:

- Dos valores difusos se consideran “*redundantes*” si uno está incluido en el otro, por lo que este último se considera la “*dispersión*” del primero. Donde la inclusión se define en los términos expuestos en el capítulo anterior.
- Sobre cada Dominio se define un “*nivel de granularidad*” con el que podemos acotar el proceso de degradación de la información en las sucesivas manipulaciones.

Con el primer concepto se introduce un criterio de “*redundancia*” cuyos límites establece mediante el concepto de “*granularidad*”. Este último, restringe la tendencia natural de los valores difusos a la dispersión mediante la imposición de unos límites, (“*niveles de granularidad*”), dependientes del dominio sobre el que están definidos.

Basándonos en los elementos anteriores, dos tuplas serán redundantes, si cada par de valores correspondiente a cada atributo, lo es en el sentido anteriormente expuesto, como consecuencia, almacenaremos una única tupla que contendrá la “*dispersión*” de ese par de valores.

Como resultado del análisis realizado sobre el tema de la redundancia, podemos concluir lo siguiente:

- El problema de la eliminación de redundancia sobre relaciones difusas en general, está estrechamente ligado a que acepción se tome del concepto de igualdad en ambientes difusos.
- El autor de esta memoria no conoce, hasta la fecha, una aproximación que resulte plenamente satisfactoria para todos los casos.
- Atendiendo al espíritu del modelo que se está presentando, no se incluye una forma específica para detectar y eliminar la redundancia, aunque se insta a que las realizaciones específicas del modelo adopten alguno de los criterios aparecidos o por aparecer, para eliminar o reducir los efectos de la redundancia.

4.2.3 Clave Primaria Generalizada K_G

El concepto de clave primaria es de suma importancia en el modelo relacional clásico, puesto que garantiza la accesibilidad de cada tupla de una relación, mediante la combinación de uno o más valores, que la determinan inequívocamente. El problema de extender este concepto a bases de datos difusas radica, precisamente, en como tratar la posibilidad de que puedan formar parte de la clave primaria atributos susceptibles de aceptar valores difusos. Dicho de otro modo, como puede, por ejemplo, una colección de distribuciones de posibilidad determinar, de forma inequívoca, una tupla frente al resto de las tuplas de la relación. Sobre este asunto, al igual que ocurre con el tema

de la redundancia, concepto del que depende, planea el problema de como extender el concepto de igualdad a valores difusos. La alternativa adoptada en este modelo para la definición de una clave primaria, muestra, nuevamente, un talante conservador basado en los siguientes criterios:

- Con una relación difusa generalizada con información de partida compuesta únicamente por atributos de naturaleza imprecisa no podemos garantizar la accesibilidad de cada tupla.
- Por tanto, habremos de poder identificar cada tupla en dicha relación mediante el empleo de una colección de valores precisos.
- Según esto, aunque alguno de los atributos de la clave primaria difusa pudiera ser susceptible de tratamiento difuso, cuando se considere como componente de la clave primaria, habrá de garantizar la distinguibilidad para cada uno de los valores de dominio que pueda adoptar.
- El modelo deja una línea abierta a la definición de claves difusas en base a otros criterios de redundancia.

Con la siguiente definición introducimos una notación que nos será útil para definir próximos conceptos.

Definición 4.3 Sea R_{FG} una relación difusa generalizada dada por:

$$R_{FG} = \begin{cases} \mathcal{H} &= \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} &= \{(A_{G1} : \tilde{d}_{i1} [, c_{i1}]), \dots, (A_{Gn} : \tilde{d}_{in} [, c_{in}])\} \end{cases}$$

llamaremos **Componente de valor** de una relación difusa generalizada y lo notaremos, R_{FG}^v , a la parte de la relación dada por:

$$R_{FG}^v = \begin{cases} \mathcal{H}^v &= \{(A_{G1} : D_{G1}), \dots, (A_{Gn} : D_{Gn})\} \\ \mathcal{B}^v &= \{(A_{G1} : \tilde{d}_{i1}), \dots, (A_{Gn} : \tilde{d}_{in})\} \end{cases} \quad (4.5)$$

\mathcal{H}^v y \mathcal{B}^v son las **componentes de valor** de la “cabecera” y el “cuerpo” respectivamente.

Así mismo, llamaremos **Componente de compatibilidad** de una relación difusa generalizada y lo notaremos, R_{FG}^c , a la parte de la relación dada por:

$$R_{FG}^c = \begin{cases} \mathcal{H}^c & = \{[C_{A_{G_1}}], \dots, [C_{A_{G_n}}]\} \\ \mathcal{B}^c & = \{[c_{i1}], \dots, [c_{in}]\} \end{cases} \quad (4.6)$$

\mathcal{H}^c y \mathcal{B}^c son las **componentes de compatibilidad** de la “cabecera” y el “cuerpo” respectivamente.

Definición 4.4 Llamaremos **relación base** a una “relación difusa generalizada”, R_{FG} , que solo posea “componente de valor”, $R_{FG} = R_{FG}^v$

Estamos ahora en condiciones de introducir la definición adoptada para la clave primaria.

Definición 4.5 Sea R_{FG} una relación difusa generalizada dada por:

$$R_{FG} = \begin{cases} \mathcal{H} & = \{(A_{G_1} : D_{G_1} [, C_{A_{G_1}}]), \dots, (A_{G_n} : D_{G_n} [, C_{A_{G_n}}])\} \\ \mathcal{B} & = \{(A_{G_1} : \tilde{d}_{r1} [, c_{r1}]), \dots, (A_{G_n} : \tilde{d}_{rn} [, c_{rn}])\} \end{cases}$$

con $r = 1, 2, \dots, m$, siendo m el número de tuplas de la relación,

llamaremos **Clave Primaria Generalizada**, K_G , a un subconjunto de \mathcal{H} expresado como:

$$K_G \subseteq \mathcal{H}, K_G = \{(A_{G_s} : D_{G_s}) : s \in S \subseteq \{1, \dots, n\}\} \quad (4.7)$$

que cumple:

1. $\forall s \in S$, D_{G_s} es un dominio numérico o escalar simple, tipos 1 y 2 de la tabla 4.1.
2. $\forall i, i' \in \{1, \dots, m\}$, $\exists s \in S : (A_{G_s} : d_{is}) \neq (A_{G_s} : d_{i's})$, donde el operador \neq recoge el significado clásico de la desigualdad.

Con la clave primaria definida de esta forma se garantiza, de un lado, la accesibilidad de cada una de las tuplas de la relación y, de otro, la posibilidad de proporcionar un tratamiento difuso a los atributos incluidos en la clave primaria cuando esto resulte conveniente. Esto es así puesto que, dichos atributos pueden construirse sobre dominios escalares que soporten la definición de relaciones de semejanza sobre sus elementos. De esta forma, para garantizar el concepto de clave primaria, cada tupla de la relación forzará la distinguibilidad sobre los valores que tomen sobre dichos atributos, esto

implica que cada valor unicamente será similar a si mismo, (umbral de semejanza igual a 1), fuera de su vertiente de clave primaria, cada atributo que la compone podrá ser objeto, de tratamientos “difusos” basados en la relación de semejanza que, en su caso, se encuentre definida sobre el dominio que subyace al atributo.

La existencia de una clave primaria posibilita que se pueda exigir, a una Base de Datos Relacional Difusa realizada según el modelo propuesto, que satisfaga la regla de Identidad.

4.3 Manipulación de los Datos

El Modelo contempla la manipulación de las $R_{FG,s}$ mediante el empleo del Álgebra Relacional Difusa Generalizada. Las operaciones clásicas, *Unión*, *Intersección*, *Diferencia*, *Producto Cartesiano*, *Proyección*, *Reunión* y *Selección*, quedan extendidas a fin de que puedan operar sobre las $R_{FG,s}$ en forma coherente.

La *Selección* y la *Reunión* basan su funcionamiento en el empleo de comparadores difusos contruidos en torno a alguna de las medidas de compatibilidad difusas aparecidas en la literatura. Nuestro modelo adopta un marco básico, sobre el que construir los comparadores difusos, partiendo del significado de las operaciones de comparación definidas sobre el dominio de discurso, y extendiendo dicho significado al tratamiento de valores del *Dominio Generalizado Difuso*.

4.3.1 Operadores de Comparación Difusa

Un aspecto crítico que ha de abordar cualquier SBDRD es la definición de los operadores que se ven envueltos en las operaciones de *Selección* y *Reunión*. Este modelo resuelve el problema mediante la definición operadores de comparación a dos niveles: a nivel del dominio de discurso D y a nivel de D_G . Sobre el primer nivel definiremos el *Comparador Extendido*, θ_e y con su ayuda construiremos el *Comparador Difuso Generalizado*, Θ_G en el segundo.

4.3.1.1 Comparador Extendido, θ_e

El **Comparador Extendido** recoge todas las modalidades de relación existentes entre los valores del Dominio de discurso D considerado, atendiendo a la naturaleza de dichos

valores y al caracter, clásico, difuso o de semejanza, de la relación existente entre los mismos.

Definición 4.6 *Sea D el dominio de discurso considerado, llamaremos **Comparador Extendido**, θ_e , a cualquier relación difusa definida sobre D , expresada en la forma:*

$$\theta_e : D \times D \longrightarrow [0, 1] \quad (4.8)$$

$$\theta_e(d_i, d_j) \longmapsto [0, 1]$$

con $d_i, d_j \in D$

Así definido el **Comparador Extendido** (θ_e) permite modelar en forma consistente los siguientes operadores de comparación:

- Comparadores Clásicos del Álgebra Relacional tales como: $=, \neq, >, \geq, <, \leq$. Por ejemplo, el operador *igual extendido* quedaría expresado mediante $=_e(d_i, d_j) = \delta(d_i, d_j)$ donde $\delta(d_i, d_j) = 1$ para $d_i = d_j$ y $\delta(d_i, d_j) = 0$ para $d_i \neq d_j$.
- Comparadores Difusos tales como “*aproximadamente igual*”, “*mucho mayor que*”, etc. Este tipo de comparadores vendrán expresados mediante funciones de pertenencia. Por ejemplo, se podría modelar el operador “*aproximadamente igual*” mediante el correspondiente comparador extendido \simeq_e con la siguiente función de pertenencia:

$$\mu_{\simeq_e}(d_i, d_j) = e^{-\beta|d_i - d_j|} \text{ con } \beta > 0$$

- Operadores de Semejanza, los cuales operan sobre datos escalares en los que se ha establecido una *relación de semejanza* [Zadeh 71].

4.3.1.2 Comparador Difuso Generalizado, $\Theta_G^{\theta_e}$

El *Comparador Difuso Generalizado* encarna gran parte de la filosofía del modelo que estamos presentando en esta memoria. En nuestra opinión, la mayoría de los problemas que conlleva la extensión del modelo relacional al ámbito de la información imprecisa, residen en la forma en que extendamos los conceptos clásicos empleados en la manipulación de los datos (el concepto de igualdad, mayor que, menor que, etc.).

Dicha extensión no es trivial ni única, el desarrollo de la teoría de conjuntos difusos ha proporcionado varias formas para llevarla a cabo, sin embargo, ninguna de ellas puede considerarse como definitiva en cuanto a que pueda captar todos los matices semánticos del concepto clásico que extiende. Así, por ejemplo, en cuanto a las medidas de “compatibilidad” adoptadas por los diferentes modelos de bases de datos relacionales difusas analizados en el capítulo anterior, el catálogo es extenso. Como vimos, Buckles-Petry basan su modelo en la definición de relaciones de similitud entre los valores del dominio subyacente a un atributo de la base de datos. Umano-Fukami definen varios operadores para captar el concepto compatibilidad, como se vio en el capítulo anterior, todas las operaciones devuelven dos conjuntos, uno constituido por los datos que según ellos estrictamente satisfacen la condición, y otro, para los datos que posiblemente la satisfacen. Prade-Testemale establecen dos medidas de compatibilidad entre valores difusos: la medida de necesidad, más restrictiva, y la medida de posibilidad. Zemankova-kandel por su parte emplean sendas medidas denominadas de posibilidad y certeza cuya definición, al igual que la definición de las medidas anteriores, puede encontrarse en el capítulo anterior.

A la vista de lo comentado, está claro que la vía para extender estos conceptos no está totalmente delimitada, por tanto no parece adecuado limitar a una sola aproximación la forma en que podemos manipular la información imprecisa existente en una base de datos. En este sentido, consideramos además, que la información imprecisa añada nuevas connotaciones semánticas distintas de las que poseen los datos precisos. Esto nos insta a diversificar los mecanismos de manipulación a fin de que puedan retener un mayor significado y una mayor capacidad para resolver los aspectos específicos que concurren en la representación y manipulación de cada valor difuso.

En virtud de lo anteriormente expuesto, consideramos que los únicos aspectos que debe recoger la definición de comparador difuso incluida en nuestro modelo son el hecho de que debe partir del concepto clásico que extiende y el de que, como consecuencia, debe operar de forma consistente con información precisa de la misma forma en que lo hace el operador clásico del que parte. Ambos aspectos quedan reflejados en la siguiente definición.

Definición 4.7 *Sea D el dominio de discurso considerado, sea D_G el dominio difuso generalizado construido sobre él y sea θ_e un comparador extendido definido en D . Consideremos una función $\Theta_G^{\theta_e}$ definida:*

$$\Theta_G^{\theta_e} : D_G \times D_G \rightarrow [0, 1] \quad (4.9)$$

$$\Theta_G^{\theta_e}(\tilde{d}_1, \tilde{d}_2) \in [0, 1]$$

diremos que $\Theta_G^{\theta_e}$ es un **Comparador Difuso Generalizado** sobre D_G inducido por el comparador extendido θ_e , si cumple:

$$\Theta_G^{\theta_e}(\tilde{d}_1, \tilde{d}_2) = \theta_e(d_1, d_2) \quad \forall d_1, d_2 \in D \quad (4.10)$$

donde \tilde{d}_1, \tilde{d}_2 representan las distribuciones de posibilidad $1/d_1, 1/d_2$, inducidas, respectivamente, por los valores d_1, d_2 .

Como podemos ver, la anterior definición no limita el *comparador difuso generalizado* a una medida concreta. Esto nos permite modelar cada operación de comparación empleando la medida adecuada a cada problema.

4.3.2 Álgebra Relacional Difusa Generalizada

Con los comparadores difusos anteriormente definidos y con la estructura de datos adoptada por el modelo, estamos en condiciones de extender el significado de los operadores del Álgebra Relacional, a fin de que podamos manipular en forma conveniente las “relaciones difusas generalizadas”.

Para definir la Unión, Intersección y Diferencia en GEFRED es preciso introducir el concepto de compatibilidad respecto a la unión de dos relaciones difusas generalizadas:

Definición 4.8 *Dos relaciones difusas generalizadas R_{FG} y R'_{FG} se dicen compatibles respecto a la unión si lo son, (en el sentido clásico), sus respectivas componentes valor, R_{FG}^v y R'_{FG}^v*

4.3.2.1 Unión Difusa Generalizada, \cup_G

La definición de Unión incluida en el modelo propuesto precisa que la estructura de las R_{FG}, s que intervienen sea compatible respecto a la unión. Como ocurre con el resto de los operadores del algebra que estamos describiendo, opera sobre R_{FG}, s y como resultado se obtienen nuevamente R_{FG}, s . En términos precisos queda definida como sigue:

Definición 4.9 Sean R_{FG} y R'_{FG} dos relaciones difusas generalizadas dadas por:

$$R_{FG} = \begin{cases} \mathcal{H} &= \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} &= \{(A_{G1} : \tilde{d}'_{i1} [, c_{i1}]), \dots, (A_{Gn} : \tilde{d}'_{in} [, c_{in}])\} \end{cases}$$

$$R'_{FG} = \begin{cases} \mathcal{H}' &= \{(A'_{G1} : D_{G1} [, C_{A'_{G1}}]), \dots, (A'_{Gn} : D_{Gn} [, C_{A'_{Gn}}])\} \\ \mathcal{B}' &= \{(A'_{G1} : \tilde{d}'_{k1} [, c'_{k1}]), \dots, (A'_{Gn} : \tilde{d}'_{kn} [, c'_{kn}])\} \end{cases}$$

con $i = 1, \dots, m$ y $k = 1, \dots, m'$, siendo m y m' las respectivas cardinalidades, entonces, la **Unión Difusa Generalizada** de R_{FG} y R'_{FG} vendrá definida por:

$$R_{FG} \cup_G R'_{FG} = \begin{cases} \mathcal{H}_{\cup_G} &= \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B}_{\cup_G} &= \begin{cases} \mathcal{B}_{\cup_G}^v &= \mathcal{B}^v \cup \mathcal{B}'^v \\ \mathcal{B}_{\cup_G}^c &= \{[c''_{l1}], \dots, [c''_{ln}]\} \end{cases} \end{cases} \quad (4.11)$$

con $l = 1, \dots, m''$, siendo m'' la cardinalidad de la unión, donde

$$c''_{lj} = \begin{cases} \max\{c_{lj}, c'_{lj}\} & \text{si } \exists c_{lj} \text{ y } \exists c'_{lj} \\ c_{lj} & \text{si } \exists c_{lj} \text{ y } \nexists c'_{lj} \\ c'_{lj} & \text{si } \exists c'_{lj} \text{ y } \nexists c_{lj} \\ 0 & \text{si } \nexists c_{lj}, \nexists c'_{lj} \text{ y, } \exists C_{A_{G_j}} \text{ ó } \exists C_{A'_{G_j}} \end{cases} \quad (4.12)$$

siendo c_{lj} , si existe, el grado de compatibilidad de d''_{lj} para la tupla l en la relación R_{FG} y c'_{lj} , si existe, el grado de compatibilidad de d''_{lj} para la tupla l en la relación R'_{FG} .

Obsérvese que para obtener cada atributo de compatibilidad de la unión es preciso que exista al menos uno de los atributos de los que procede en alguna de las relaciones que intervienen en la unión. Además puede verse que, para la *componente de valor* de las relaciones a unir, se emplea el concepto de unión clásico. Mediante este concepto únicamente se considerarán tuplas redundantes aquellas que posean el mismo valor para cada uno de los atributos que las componen. Como comentábamos al introducir la *relación difusa generalizada*, se pueden emplear otros criterios para eliminar redundancia en la relación resultante de la unión descrita.

• Propiedades

1. *Propiedad Conmutativa.* Sean R_{FG} y R'_{FG} dos relaciones difusas generalizadas, entonces se cumple:

$$R_{FG} \cup_G R'_{FG} = R'_{FG} \cup_G R_{FG} \quad (4.13)$$

Demostración

Puesto que las dos relaciones han de ser compatibles respecto a la unión, podemos centrarnos en comprobar el cumplimiento de la propiedad para los cuerpos de las relaciones.

En efecto, como la componente de valor del cuerpo de la *unión difusa generalizada* se obtiene de la unión clásica de las componentes de valor de los cuerpos de las relaciones de partida, dicha componente de valor será conmutativa. Por otro lado es preciso probar que el cálculo de la componente de compatibilidad del cuerpo de la *unión difusa generalizada* es conmutativo. Esto es inmediato a partir de la ec. 4.11, donde el cálculo de los grados de compatibilidad c_{ij} es idéntico si se intercambian c_{ij} por c'_{ij} , con lo cual es conmutativo y, por tanto, se concluye que la *unión difusa generalizada* también lo es.

2. *Propiedad Asociativa.* Sean R_{FG} , R'_{FG} y R''_{FG} tres relaciones difusas generalizadas, entonces se cumple:

$$R_{FG} \cup_G (R'_{FG} \cup_G R''_{FG}) = (R_{FG} \cup_G R'_{FG}) \cup_G R''_{FG} \quad (4.14)$$

Demostración

Por definición, en función de las componentes de valor de los cuerpos de las relaciones participantes la ec. 4.14 queda expresada:

$$\mathcal{B}^v \cup (\mathcal{B}'^v \cup \mathcal{B}''^v) = (\mathcal{B}^v \cup \mathcal{B}'^v) \cup \mathcal{B}''^v \quad (4.15)$$

Dicha expresión se cumple puesto que las componentes indicadas tienen una estructura compatible con las relaciones clásicas y la unión clásica satisface la propiedad asociativa.

Por otro lado hemos de probar que el cálculo de la componente de compatibilidad es asociativa. Para ello habremos de comprobar todas las posibilidades de la expresión 4.11. En las expresiones que siguen emplearemos la siguiente notación:

Sea l una tupla cualquiera perteneciente a $\mathcal{B}^v \cup \mathcal{B}^{'v} \cup \mathcal{B}^{''v}$ y sea j un atributo cualquiera de la anterior unión, entonces c_{lj} identificará, si existe, al grado de compatibilidad del atributo j para la tupla l en la relación R_{FG} , c'_{lj} significará lo mismo para la relación R'_{FG} y c''_{lj} para la R''_{FG} , así mismo, u_{lj} , si existe, identificará al grado de compatibilidad del atributo j para la tupla $l \in R'_{FG} \cup_G R''_{FG}$ y u_{lj} hará lo mismo para una tupla $l \in R_{FG} \cup_G R'_{FG}$.

Una vez establecida la notación, veamos de forma exhaustiva como se verifica la asociatividad para cada caso:

- (a) Si $\exists c_{lj}$, $\exists c'_{lj}$ y $\exists c''_{lj}$, entonces de 4.11 tenemos que la expresión 4.14 para la componente de compatibilidad cumple:

$$\max\{\max\{c'_{lj}, c''_{lj}\}, c_{lj}\} = \max\{c'_{lj}, c''_{lj}, c_{lj}\} = \max\{\max\{c_{lj}, c'_{lj}\}, c''_{lj}\}$$

se cumple la asociatividad para este caso.

- (b) Si $\exists c_{lj}$, $\nexists c'_{lj}$ y $\exists c''_{lj}$ tenemos que $u_{lj} = c''_{lj}$ y, por tanto, la parte izquierda de la expresión 4.14 queda:

$$\max\{c_{lj}u_{lj}\} = \max\{c_{lj}, c''_{lj}\}$$

por otro lado tenemos que $u'_{lj} = c_{lj}$, con lo que para la parte derecha de la expresión 4.14 se obtiene:

$$\max\{u'_{lj}, c''_{lj}\} = \max\{c_{lj}, c''_{lj}\}$$

lo cual demuestra la asociatividad para este caso.

- (c) Si $\nexists c_{lj}$, $\exists c'_{lj}$ y $\exists c''_{lj}$, obtenemos que $u_{lj} = \max\{c'_{lj}, c''_{lj}\}$, con lo que la parte izquierda resulta ser:

$$u_{lj} = \max\{c'_{lj}, c''_{lj}\}$$

así mismo obtenemos, $u'_{lj} = c'_{lj}$ lo que lleva a que la parte derecha sea:

$$\max\{u'_{lj}, c''_{lj}\} = \max\{c'_{lj}, c''_{lj}\}$$

se cumple la asociatividad para este caso.

- (d) Si $\exists c_{lj}$, $\exists c'_{lj}$ y $\bar{\exists} c''_{lj}$, entonces $u_{lj} = c_{lj}$ y la expresión izquierda de 4.14

$$\max\{c_{lj}, u'_{lj}\} = \max\{c_{lj}, c'_{lj}\}$$

$u'_{lj} = \max\{c_{lj}, c'_{lj}\}$ y la parte derecha por tanto

$$u'_{lj} = \max\{c_{lj}, c'_{lj}\}$$

lo cual prueba la asociatividad para este caso.

- (e) Si $\bar{\exists} c_{lj}$, $\bar{\exists} c'_{lj}$, $\exists C_{A_{G_j}}$ ó $\exists C_{A'_{G_j}}$ y $\exists c''_{lj}$, entonces $u_{lj} = c''_{lj}$, luego la parte izquierda queda

$$u_{lj} = c''_{lj}$$

puesto que $u'_{lj} = 0$, para la parte derecha tenemos

$$\max\{u'_{lj}, c''_{lj}\} = \max\{0, c''_{lj}\} = c''_{lj}$$

verificándose por tanto la asociatividad para este caso.

- (f) Si $\bar{\exists} c_{lj}$, $\bar{\exists} c''_{lj}$, $\exists C_{A_{G_j}}$ ó $\exists C_{A''_{G_j}}$ y $\exists c'_{lj}$, entonces $u_{lj} = c'_{lj}$ y la parte izquierda de la expresión

$$u_{lj} = c'_{lj}$$

como $u'_{lj} = c'_{lj}$ la parte derecha se reduce a

$$u'_{lj} = c'_{lj}$$

probando la asociatividad para este caso.

- (g) Si $\bar{\exists} c'_{lj}$, $\bar{\exists} c''_{lj}$, $\exists C_{A'_{G_j}}$ ó $\exists C_{A''_{G_j}}$ y $\exists c_{lj}$, entonces $u_{lj} = 0$ y para la parte izquierda de 4.14 obtenemos

$$\max\{c_{lj}, 0\} = c_{lj}$$

puesto que $u'_{lj} = c_{lj}$ para la parte derecha se obtiene

$$u'_{lj} = c_{lj}$$

lo que prueba la asociatividad para este caso.

- (h) Si $\bar{\exists} c_{lj}$, $\bar{\exists} c'_{lj}$, $\bar{\exists} c''_{lj}$ y $\exists C_{A_{G_j}}$ ó $\exists C_{A'_{G_j}}$ ó $\exists C_{A''_{G_j}}$ entonces el resultado para la unión en cualquiera de los casos es 0.

Con lo cual se demuestra que la *Unión Difusa Generalizada* es asociativa. ■

Así definida, la *unión difusa generalizada* permite, como veremos cuando introduzcamos el concepto de selección adoptado, modelar el concepto de disyunción en una consulta. Además, contribuye a uniformizar el tratamiento empleado para resolver una consulta. Esto es así, ya que, como el resto de los operadores que definiremos, emplea únicamente el concepto de *relación difusa generalizada* para representar los estados por los que transita la información en el proceso de manipulación. En el apartado 4.4 mostraremos a través de un ejemplo como se opera con el concepto introducido.

4.3.2.2 Intersección Difusa Generalizada, \cap_G

El concepto de intersección de relaciones queda extendido en el marco del modelo según se recoge en la siguiente definición:

Definición 4.10 Sean R_{FG} y R'_{FG} dos relaciones difusas generalizadas compatibles respecto a la unión dadas por:

$$R_{FG} = \begin{cases} \mathcal{H} &= \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} &= \{(A_{G1} : \tilde{d}_{i1} [, c_{i1}]), \dots, (A_{Gn} : \tilde{d}_{in} [, c_{in}])\} \end{cases}$$

$$R'_{FG} = \begin{cases} \mathcal{H}' &= \{(A'_{G1} : D_{G1} [, C_{A'_{G1}}]), \dots, (A'_{Gn} : D_{Gn} [, C_{A'_{Gn}}])\} \\ \mathcal{B}' &= \{(A'_{G1} : \tilde{d}'_{k1} [, c'_{k1}]), \dots, (A'_{Gn} : \tilde{d}'_{kn} [, c'_{kn}])\} \end{cases}$$

con $i = 1, \dots, m$ y $k = 1, \dots, m'$, siendo m y m' las respectivas cardinalidades, entonces, la **Intersección Difusa Generalizada** de R_{FG} y R'_{FG} vendrá definida por:

$$R_{FG} \cap_G R'_{FG} = \begin{cases} \mathcal{H}_{\cap_G} &= \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B}_{\cap_G} &= \begin{cases} \mathcal{B}_{\cap_G}^v &= \mathcal{B}^v \cap \mathcal{B}'^v \\ \mathcal{B}_{\cap_G}^c &= \{[c''_{l1}], \dots, [c''_{ln}]\} \end{cases} \end{cases} \quad (4.16)$$

con $l = 1, \dots, m''$, siendo m'' la cardinalidad de la intersección, donde

$$c''_{ij} = \begin{cases} \min\{c_{ij}, c'_{ij}\} & \text{si } \exists c_{ij} \text{ y } \exists c'_{ij} \\ c_{ij} & \text{si } \exists c_{ij} \text{ y } \nexists c'_{ij} \\ c'_{ij} & \text{si } \exists c'_{ij} \text{ y } \nexists c_{ij} \end{cases} \quad (4.17)$$

siendo c_{ij} , si existe, el grado de compatibilidad de d''_{ij} para la tupla l en la relación R_{FG} y c'_{ij} , si existe, el grado de compatibilidad de d''_{ij} para la tupla l en la relación R'_{FG} .

Nuevamente, se parte del concepto clásico para la *componente de valor* de las relaciones a intersectar. Para la componente de compatibilidad, cada atributo de la intersección procede de combinar al menos un atributo de alguno de los atributos correspondientes de las relaciones de partida.

• Propiedades

1. *Propiedad Conmutativa.* Sean R_{FG} y R'_{FG} dos relaciones difusas generalizadas, entonces se cumple:

$$R_{FG} \cap_G R'_{FG} = R'_{FG} \cap_G R_{FG} \quad (4.18)$$

Demostración

Al igual que vimos para la *unión difusa generalizada*, la propiedad conmutativa se verifica para la componente de valor de la *intersección difusa generalizada*. Por otro lado, el cálculo de la componente de compatibilidad del cuerpo de la *unión difusa generalizada* es conmutativo. Esto es inmediato a partir de la ec. 4.17, donde el cálculo de los grados de compatibilidad c_{ij} es idéntico si se intercambian c_{ij} por c'_{ij} , con lo cual es conmutativo y, por tanto, se concluye que la *intersección difusa generalizada* también lo es.

2. *Propiedad Asociativa.* Sean R_{FG} , R'_{FG} y R''_{FG} tres relaciones difusas generalizadas, entonces se cumple:

$$R_{FG} \cap_G (R'_{FG} \cap_G R''_{FG}) = (R_{FG} \cap_G R'_{FG}) \cap_G R''_{FG} \quad (4.19)$$

Demostración

Nuevamente, se observa de la definición, que la propiedad asociativa se satisface sobre las componentes de valor de las relaciones que intervienen.

Para probar la propiedad asociativa sobre las componentes de compatibilidad es preciso que se verifique la igualdad 4.19 para todos los casos posibles. Para ello nos remitiremos a la siguiente notación:

Sea l una tupla cualquiera perteneciente a $\mathcal{B}^v \cap \mathcal{B}'^v \cap \mathcal{B}''^v$ y sea j un atributo cualquiera de la anterior intersección, entonces c_{lj} identificará, si existe, al grado de compatibilidad del atributo j para la tupla l en la relación R_{FG} , c'_{lj} significará lo mismo para la relación R'_{FG} y c''_{lj} para la R''_{FG} , así mismo, u_{lj} , si existe, identificará al grado de compatibilidad del atributo j para la tupla $l \in R'_{FG} \cap_G R''_{FG}$ y u_{lj} hará lo mismo para una tupla $l \in R_{FG} \cap_G R'_{FG}$.

Los casos posibles son:

- (a) Si $\exists c_{lj}$, $\exists c'_{lj}$ y $\exists c''_{lj}$, entonces de 4.17 tenemos que $u_{lj} = \min\{c'_{lj}, c''_{lj}\}$ lo cual lleva a que la parte izquierda de la expresión 4.19 tome, para el grado de compatibilidad correspondiente, la forma siguiente

$$\min\{c_{lj}, u_{lj}\} = \min\{c_{lj}, \min\{c'_{lj}, c''_{lj}\}\} = \min\{c_{lj}, c'_{lj}, c''_{lj}\}$$

con respecto a la parte derecha, como $u'_{lj} = \min\{c_{lj}, c'_{lj}\}$, entonces queda

$$\min\{u'_{lj}, c''_{lj}\} = \min\{\min\{c_{lj}, c'_{lj}\}, c''_{lj}\} = \min\{c_{lj}, c'_{lj}, c''_{lj}\}$$

lo cual prueba la asociatividad para este caso.

- (b) Si $\nexists c_{lj}$, $\exists c'_{lj}$ y $\exists c''_{lj}$, entonces $u_{lj} = \min\{c'_{lj}, c''_{lj}\}$ y, por tanto, la parte izquierda de la expresión 4.19 queda

$$u_{lj} = \min\{c'_{lj}, c''_{lj}\}$$

por otro lado, $u'_{lj} = c'_{lj}$ por lo que la parte derecha se expresa

$$\min\{u'_{lj}, c''_{lj}\} = \min\{c'_{lj}, c''_{lj}\}$$

quedando probada la propiedad para este caso.

- (c) Si $\exists c_{lj}$, $\nexists c'_{lj}$ y $\exists c''_{lj}$, entonces $u_{lj} = c''_{lj}$ y la parte izquierda adopta la forma

$$\min\{c_{lj}, u_{lj}\} = \min\{c_{lj}, c''_{lj}\}$$

y, puesto que $u'_{lj} = c_{lj}$, para la parte derecha se obtiene

$$\min\{u'_{lj}, c''_{lj}\} = \min\{c_{lj}, c''_{lj}\}$$

(d) $\exists c_{lj}$, $\exists c'_{lj}$ y $\nexists c''_{lj}$, entonces $u_{lj} = c'_{lj}$ y la parte izquierda de 4.19

$$\min\{c_{lj}, u_{lj}\} = \min\{c_{lj}, c'_{lj}\}$$

y, como $u'_{lj} = \min\{c_{lj}, c'_{lj}\}$, la parte derecha da

$$u'_{lj} = \min\{c_{lj}, c'_{lj}\}$$

lo que prueba la propiedad para este caso

(e) Si sólo existe grado de compatibilidad para una de la relaciones participantes en la intersección el resultado es que es este el grado que toma la intersección, independientemente de como asociemos la intersección de las relaciones que intervienen.

Con lo cual queda demostrada la propiedad. ■

A través del concepto de intersección de relaciones introducido, podemos materializar, además, el concepto de conjunción. Algunos de los aspectos de su empleo serán ilustrados a través del ejemplo descrito en el apartado 4.4.

4.3.2.3 Diferencia Difusa Generalizada, $-_G$

La diferencia de relaciones en el ámbito de nuestro modelo queda extendida como muestra la siguiente definición.

Definición 4.11 Sean R_{FG} y R'_{FG} dos relaciones difusas generalizadas dadas por:

$$R_{FG} = \begin{cases} \mathcal{H} & = \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} & = \{(A_{G1} : \tilde{d}_{i1} [, c_{i1}]), \dots, (A_{Gn} : \tilde{d}_{in} [, c_{in}])\} \end{cases}$$

$$R'_{FG} = \begin{cases} \mathcal{H}' & = \{(A'_{G1} : D_{G1} [, C_{A'_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B}' & = \{(A'_{G1} : \tilde{d}'_{k1} [, c'_{k1}]), \dots, (A'_{Gn} : \tilde{d}'_{kn} [, c'_{kn}])\} \end{cases}$$

con $i = 1, \dots, m$ y $k = 1, \dots, m'$, siendo m y m' las respectivas cardinalidades, entonces, la **Diferencia Difusa Generalizada** de R_{FG} y R'_{FG} vendrá definida por:

$$R_{FG} -_G R'_{FG} = \begin{cases} \mathcal{H}_{-G} &= \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B}_{-G} &= \begin{cases} \mathcal{B}_{-G}^v &= \mathcal{B}^v \\ \mathcal{B}_{-G}^c &= \{[c''_{l1}], \dots, [c''_{ln}]\} \end{cases} \end{cases} \quad (4.20)$$

con $l = 1, \dots, m''$, siendo m'' la cardinalidad de la diferencia, donde

$$c''_{lj} = \begin{cases} \min\{c_{lj}, (1 - c'_{lj})\} & \text{si } \exists C_{A_{Gj}}, \exists C_{A'_{Gj}} \text{ y } l \in \mathcal{B}^v \cap \mathcal{B}'^v \\ c_{lj} & \text{si } \exists C_{A_{Gj}} \text{ y } \nexists C_{A'_{Gj}} \\ 1 & \text{si } \nexists C_{A_{Gj}}, \exists C_{A'_{Gj}} \text{ y } l \in \mathcal{B}^v - \mathcal{B}'^v \\ 1 - c'_{lj} & \text{si } \nexists C_{A_{Gj}}, \exists C_{A'_{Gj}} \text{ y } l \in \mathcal{B}^v \cap \mathcal{B}'^v \end{cases} \quad (4.21)$$

siendo c_{lj} el grado de compatibilidad de d''_{lj} para la tupla l en la relación R_{FG} y c'_{lj} el grado de compatibilidad de d'_{lj} para la tupla l en la relación R'_{FG} .

Como se ve, para el cálculo de la diferencia es preciso considerar si la tupla l pertenece a la intersección clásica de las componentes de valor de las dos relaciones o a la diferencia clásica de las mismas. Teniendo esto en cuenta, el cálculo de la componente de compatibilidad de la relación resultante dependerá de la tupla considerada y de la existencia o no de los atributos de compatibilidad correspondientes en las relaciones de partida.

4.3.2.4 Producto Difuso Generalizado, \times_G

El producto de define como sigue:

Definición 4.12 Sean R_{FG} y R'_{FG} dos relaciones difusas generalizadas dadas por:

$$R_{FG} = \begin{cases} \mathcal{H} &= \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} &= \{(A_{G1} : \tilde{d}_{i1} [, c_{i1}]), \dots, (A_{Gn} : \tilde{d}_{in} [, c_{in}])\} \end{cases}$$

$$R'_{FG} = \begin{cases} \mathcal{H}' &= \{(A'_{G1} : D'_{G1} [, C_{A'_{G1}}]), \dots, (A'_{Gn'} : D'_{Gn'} [, C_{A'_{Gn'}}])\} \\ \mathcal{B}' &= \{(A'_{G1} : \tilde{d}'_{k1} [, c'_{k1}]), \dots, (A'_{Gn'} : \tilde{d}'_{kn'} [, c'_{kn'}])\} \end{cases}$$

con $i = 1, \dots, m$ y $k = 1, \dots, m'$, siendo m y m' las respectivas cardinalidades y n , n' los respectivos grados,

entonces, el **Producto Cartesiano Difuso Generalizado** se define como:

$$R_{FG} \times_G R'_{FG} = \begin{cases} \mathcal{H}_{\times_G} &= \mathcal{H} \times \mathcal{H}' \\ \mathcal{B}_{\times_G} &= \mathcal{B} \times \mathcal{B}' \end{cases} \quad (4.22)$$

Como se puede observar, el **Producto Cartesiano Difuso Generalizado** opera igual que el producto cartesiano usual, solo que considera a los atributos de compatibilidad y a sus valores como partes integrantes de las relaciones. Por tanto, también satisface las propiedades conmutativa y asociativa.

4.3.2.5 Proyección Difusa Generalizada, \mathcal{P}_G

Definición 4.13 Sea R_{FG} una relación difusa generalizada dada por:

$$R_{FG} = \begin{cases} \mathcal{H} &= \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} &= \{(A_{G1} : \tilde{d}_{r1} [, c_{r1}]), \dots, (A_{Gn} : \tilde{d}_{rn} [, c_{rn}])\} \end{cases}$$

con $r = 1, 2, \dots, m$, siendo m el número de tuplas de la relación, sea X un subconjunto de \mathcal{H} expresado como:

$$X \subseteq \mathcal{H}, X = \{(A_{Gs} : D_{Gs} [, C_{A_{G_s}}]) : s \in S, s' \in S'; S, S' \subseteq \{1, \dots, n\}\}$$

entonces, diremos que la **Proyección Difusa Generalizada** de R_{FG} sobre X , $\mathcal{P}_G(R_{FG}; X)$ es una relación difusa generalizada dada por:

$$\mathcal{P}_G(R_{FG}; X) = \begin{cases} \mathcal{H}_{\mathcal{P}} &= X \\ \mathcal{B}_{\mathcal{P}} &= \{(A_{Gs} : \tilde{d}_{rs} [, c_{rs'}])\} \end{cases} \quad (4.23)$$

con $s \in S, s' \in S', S, S' \subseteq \{1, \dots, n\}$.

La proyección se puede realizar sobre cualquier subconjunto de la *cabecera*, incluyendo éste “atributos de valor” y sus correspondientes “atributos de compatibilidad”. El resultado de la proyección es una “selección vertical” sobre el *cuerpo* de la relación. El resultado de una proyección es una *relación difusa generalizada*, por lo que se le pueden aplicar las mismas consideraciones hechas en el subapartado 4.2.2.1, incluidas las relativas al problema de la redundancia. Mediante el ejemplo descrito en el apartado 4.4 podemos encontrar la forma en que opera la proyección introducida.

4.3.2.6 Selección Difusa Generalizada, \mathcal{S}_G

La selección es una de las operaciones que exigen un mayor esfuerzo en su definición. Esto es debido a que implementan los mecanismos por los que una consulta expresada en términos imprecisos, debe restringir las tuplas devueltas, a aquellas que satisfacen los requerimientos de la misma. En el modelo que estamos enunciando, dichos mecanismos proporcionan tres grados de libertad a la hora de formular una consulta. El primero de ellos reside en la variedad de tipos de información susceptible de ser objeto de consulta. El segundo permite elegir el modelo de comparador a emplear para ajustarnos más a la idea que se persigue en cada consulta. Con el tercero es posible acotar la precisión con la se ha de satisfacer cada condición involucrada en una consulta. Precisamos los aspectos fundamentales de la selección mediante la siguiente definición:

Definición 4.14 Sea R_{FG} una relación difusa generalizada dada por:

$$R_{FG} = \begin{cases} \mathcal{H} & = \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} & = \{(A_{G1} : \tilde{d}_{r1} [, c_{r1}]), \dots, (A_{Gn} : \tilde{d}_{rn} [, c_{rn}])\} \end{cases}$$

con $r = 1, 2, \dots, m$, siendo m el número de tuplas de la relación,

sea $\tilde{a} \in D_G$ una constante, sea $\Theta_G^{\theta_e}$ un “comparador difuso generalizado” y sea $\gamma \in [0, 1]$ un “umbral de cumplimiento”,

entonces, la **Selección Difusa Generalizada** realizada sobre R_{FG} por la condición inducida por $\Theta_G^{\theta_e}$ compuesto con \tilde{a} sobre el atributo A_{Gi} y cualificada por γ , $\mathcal{S}_G(R_{FG}; \Theta_G^{\theta_e}(A_i, \tilde{a}) \geq \gamma)$, es una relación $R_{FG}^{\mathcal{S}}$ dada por:

$$R_{FG}^{\mathcal{S}} = \begin{cases} \mathcal{H} & = \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} & = \{(A_{G1} : \tilde{d}_{r'1} [, c_{r'1}]), \dots, (A_{Gi} : \tilde{d}_{r'i} [, c'_{r'i}]), \dots, (A_{Gn} : \tilde{d}_{r'n} [, c_{r'n}])\} \end{cases} \quad (4.24)$$

con

$$c'_{r'i} = \Theta_G^{\theta_e}(\tilde{d}_{r'i}, \tilde{a}) \geq \gamma, \quad (4.25)$$

$r' = 1, \dots, m'$, siendo m' el número de tuplas de la selección.

La definición anterior permite materializar sobre una R_{FG} una restricción compuesta por una sola condición. Para formular consultas complejas compuestas por más

de una condición simple el modelo establece la forma en que se resuelven mediante las siguientes reglas de composición:

• **Reglas de Composición sobre la Selección Generalizada, \mathcal{S}_G**

Definición 4.15 Sea R_{FG} una relación difusa generalizada dada por:

$$R_{FG} = \begin{cases} \mathcal{H} & = \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} & = \{(A_{G1} : \tilde{d}_{r1} [, c_{r1}]), \dots, (A_{Gn} : \tilde{d}_{rn} [, c_{rn}])\} \end{cases}$$

con $r = 1, 2, \dots, m$, siendo m el número de tuplas de la relación,

- sea $S \subseteq \{1, \dots, n\}$ siendo n el “grado” de R_{FG} ,
- sean $\tilde{a}_i \in D_{Gi}$, $i \in S$ un conjunto de constantes no necesariamente distintas y A_{Gi} , $i \in S$ un conjunto de atributos de la cabecera de R_{FG} no necesariamente distintos,
- sean $\Theta_{Gi}^{\theta_e}$, $i \in S$ un conjunto de comparadores difusos generalizados, no necesariamente distintos,
- sean $\gamma_i \in [0, 1]$, un conjunto de “umbrales de cumplimiento”,
- y sean $\mathcal{S}_G(R_{FG}; \Theta_{Gi}^{\theta_e}(A_i, \tilde{a}_i) \geq \gamma_i)$, $i \in S$ el conjunto de selecciones difusas generalizadas inducidas por las constantes \tilde{a}_i y los comparadores difusos generalizados $\Theta_{Gi}^{\theta_e}$ sobre los atributos A_{Gi} de R_{FG} y cualificadas por γ_i ,

entonces, definimos las siguientes composiciones sobre la Selección Generalizada:

1. Llamaremos **Complemento de la Selección Generalizada**,

$\neg \mathcal{S}_G(R_{FG}; \Theta_{Gi}^{\theta_e}(A_i, \tilde{a}_i) \geq \gamma_i) = \mathcal{S}_G(R_{FG}; \neg \Theta_{Gi}^{\theta_e}(A_i, \tilde{a}_i) \geq \gamma_i)$, a la relación difusa generalizada, $R_{FG}^{\neg \mathcal{S}}$ dada por:

$$R_{FG}^{\neg \mathcal{S}} = \begin{cases} \mathcal{H} & = \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} & = \{(A_{G1} : \tilde{d}_{r'1} [, c_{r'1}]), \dots, (A_{Gi} : \tilde{d}_{r'i} [, c'_{r'i}]), \dots, (A_{Gn} : \tilde{d}_{r'n} [, c_{r'n}])\} \end{cases} \quad (4.26)$$

con

$$c'_{r'i} = 1 - \Theta_{Gi}^{\theta_e}(\tilde{d}_{r'i}, \tilde{a}_i) \geq \gamma_i \quad (4.27)$$

$r' = 1, \dots, m'$, siendo m' el número de tuplas de la selección.

2. Se define la **Conjunción sobre la Selección Generalizada** mediante la expresión:

$$\bigwedge_{i \in S} \mathcal{S}_G(R_{FG}; \Theta_{G_i}^{\theta_e}(A_i, \tilde{a}_i) \geq \gamma_i) = \bigcap_{i \in S} \mathcal{S}_G(R_{FG}; \Theta_{G_i}^{\theta_e}(A_i, \tilde{a}_i) \geq \gamma_i) \quad (4.28)$$

3. La **Disyunción sobre la Selección Generalizada** viene dada por la expresión:

$$\bigvee_{i \in S} \mathcal{S}_G(R_{FG}; \Theta_{G_i}^{\theta_e}(A_i, \tilde{a}_i) \geq \gamma_i) = \bigcup_{i \in S} \mathcal{S}_G(R_{FG}; \Theta_{G_i}^{\theta_e}(A_i, \tilde{a}_i) \geq \gamma_i) \quad (4.29)$$

La **Selección Difusa Generalizada** así definida, proporciona un mecanismo por el cual se ajustan, los “grados de compatibilidad” de los “valores de atributo” según las condiciones impuestas en la consulta y se eliminan aquellas tuplas que no satisfacen un “umbral de cumplimiento” establecido. Para ello, se hace uso de los “comparadores difusos generalizados” empleados para modelar las condiciones. También se han introducido reglas mediante las cuales componer consultas complejas a partir de unas más simples. Obsérvese que en todo el proceso no interviene otro mecanismo de representación que no sea el proporcionado por la *relación difusa generalizada*. En el apartado 4.4 se resuelve un ejemplo de consulta empleando los mecanismos establecidos en la definición que acabamos de introducir. Veremos también, como algunos de los modelos de consulta estudiados en el capítulo anterior pueden formularse mediante el empleo de los operadores del *álgebra difusa generalizada* que estamos definiendo.

4.3.2.7 Reunión Difusa Generalizada, \bowtie_G

En el modelo objeto de estudio introduciremos, por los motivos aducidos al principio del capítulo, una versión *conservadora* de la Reunión. Este calificativo proviene del hecho de que el proceso de reunión de dos relaciones difusas generalizadas preserva la información de partida presente en cada uno de los atributos implicados. Aunque para algunos aspectos del modelo relacional, (dependencias funcionales, procesos de normalización), es preciso llegar más lejos en la extensión del concepto, también es cierto que dichos aspectos no han encontrado, de momento, una solución satisfactoria. En consonancia con el espíritu que anima al modelo que enunciarnos, adoptamos una definición abierta en este sentido y susceptible de ser perfilada mediante sucesivas aproximaciones. En definitiva, por el momento, el modelo que estamos enunciando no va a establecer la

forma en que la información presente en los atributos difusos implicados en la reunión va a converger en una representación única que los aglutine. A nuestro entender, este proceso presenta grandes dificultades que solo pueden ser resueltas parcialmente, por lo que no consideramos oportuno recoger una única aproximación en el modelo. Sin embargo, a partir de la definición que introducimos a continuación, pueden construirse aproximaciones que recojan aspectos específicos de la reunión.

Definición 4.16 Sean R_{FG} y R'_{FG} dos relaciones difusas generalizadas, no necesariamente distintas, dadas por:

$$R_{FG} = \begin{cases} \mathcal{H} & = \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} & = \{(A_{G1} : \tilde{d}_{j1} [, c_{j1}]), \dots, (A_{Gn} : \tilde{d}_{jn} [, c_{jn}])\} \end{cases}$$

$$R'_{FG} = \begin{cases} \mathcal{H}' & = \{(A'_{G1} : D'_{G1} [, C_{A'_{G1}}]), \dots, (A'_{Gn'} : D'_{Gn'} [, C_{A'_{Gn'}}])\} \\ \mathcal{B}' & = \{(A'_{G1} : \tilde{d}'_{k1} [, c'_{k1}]), \dots, (A'_{Gn'} : \tilde{d}'_{kn'} [, c'_{kn'}])\} \end{cases}$$

con $j = 1, \dots, m$ y $k = 1, \dots, m'$, siendo m y m' las respectivas cardinalidades y n, n' los respectivos grados, sea $\Theta_G^{\theta_c}$ un “comparador difuso generalizado” y sea $\gamma \in [0, 1]$ un “umbral de cumplimiento”.

Entonces, la **Reunión Difusa Generalizada** realizada sobre $R_{FG} \times_G R'_{FG}$ mediante la condición $\Theta_G^{\theta_c}$ inducida sobre los atributos A_{Gi} de R_{FG} y $A'_{Gi'}$ de R'_{FG} , cualificada por γ , $\bowtie_G (R_{FG} \times_G R'_{FG}; \Theta_G^{\theta_c}(A_i, A'_{i'}) \geq \gamma)$, viene dada por:

$$R_{FG}^{\bowtie} = \begin{cases} \mathcal{H}^{\bowtie} & = \mathcal{H} \times \mathcal{H}' \\ \mathcal{B}^{\bowtie} & = \{(A_{G1} : \tilde{d}_{r1} [, c_{r1}]), \dots, (A_{Gi} : \tilde{d}_{ri} [, c''_{ri}), \dots, \\ & (A'_{Gi'} : \tilde{d}'_{ri'} [, c''_{ri'}), \dots, (A'_{Gn''} : \tilde{d}'_{rn''} [, c'_{rn''}])\} \end{cases} \quad (4.30)$$

con $n'' = n + n'$, donde r es la cardinalidad de la Reunión y

$$c''_{ri} = c''_{ri'} = \Theta_G^{\theta_c}(\tilde{d}_{ri}, \tilde{d}'_{ri'}) \geq \gamma \quad (4.31)$$

La *Reunión Difusa Generalizada* así definida se puede entender como una especie de *Selección Difusa Generalizada* realizada sobre el *Producto Cartesiano Difuso Generalizado* de las dos relaciones implicadas y, cuyo resultado, es la variación de los grados de compatibilidad de los atributos sobre los que se ejecuta la Reunión y la supresión de aquellas tuplas que no presenten un “grado de Reunión” superior o igual a un umbral

establecido. Como siempre, el criterio por el que se elimine la posible redundancia puede venir establecido mediante una formulación del modelo más específica en ese sentido. Así mismo, en combinación con el aspecto anterior, es posible obtener expresiones parciales para la extensión del concepto de Reunión natural en el ámbito de la definición introducida.

4.3.3 Cardinalidad Difusa Generalizada

En este apartado vamos a abordar la definición de la cardinalidad de una “relación difusa generalizada”. Mediante este concepto se pretende establecer el punto de partida sobre el que se articulan una serie de operaciones de manipulación cuyo fin es evaluar aspectos relativos a la cantidad de tuplas que satisfacen una determinada condición.

En el caso clásico, la forma de evaluar la cardinalidad de un conjunto no presenta problemas, puesto que la pertenencia de un elemento a un conjunto solo puede evaluarse como 0 o 1. En el ámbito del tratamiento difuso, el problema se plantea a la hora de considerar en que grado una “tupla difusa” incrementa la cardinalidad del conjunto considerado. En [Zadeh 83] se proponen varias fórmulas para evaluar la cardinalidad de un conjunto difuso. De todas estas, el modelo que estamos describiendo va a adoptar, para el cálculo de la cardinalidad de una relación, una definición que está basada en un tipo de cardinalidad denominada *sigma-count* [DeLuca 72] y [Zadeh 72]. Esta cardinalidad se define como sigue:

Definición 4.17 *Sea un conjunto difuso F expresado mediante:*

$$F = \mu_1/u_1 + \dots + \mu_n/u_n$$

*entonces se define la **sigma-count absoluta** de dicho conjunto como:*

$$\sum Count(F) \triangleq \sum_i \mu_i \quad i = 1, \dots, n \quad (4.32)$$

La siguiente definición precisa el concepto de cardinalidad adoptado para GEFRED:

Definición 4.18 *Sea R_{FG} una “relación difusa generalizada”, dada por*

$$R_{FG} = \begin{cases} \mathcal{H} & = \{(A_{G1} : D_{G1} [, C_{A_{G1}}]), \dots, (A_{Gn} : D_{Gn} [, C_{A_{Gn}}])\} \\ \mathcal{B} & = \{(A_{G1} : \tilde{d}_{l1} [, c_{l1}]), \dots, (A_{Gn} : \tilde{d}_{ln} [, c_{ln}])\} \end{cases}$$

con $l = 1, \dots, m$, siendo m el número de tuplas de la relación.

entonces, la **Cardinalidad Difusa Generalizada** de dicha relación, $\text{card}(R_{FG})$, viene expresada como:

$$\text{card}(R_{FG}) = \sum_{l=1}^m \mu_{R_{FG}}(t_l) \quad (4.33)$$

donde:

1. $\mu_{R_{FG}}(t_l) = 1$ si R_{FG} es una “relación base”.
2. $\mu_{R_{FG}}(t_l) = c_{lj}$ si R_{FG} es una “selección atómica” que implica al atributo A_{Gj} , siendo c_{lj} el “grado de compatibilidad” de la tupla t_l para dicha selección.
3. $\mu_{R_{FG}}(t_l) = \max\{\mu_{R'_{FG}}(t_l), \mu_{R''_{FG}}(t_l)\}$ si R_{FG} es la “unión difusa generalizada” de las relaciones R'_{FG} y R''_{FG}
4. $\mu_{R_{FG}}(t_l) = \min\{\mu_{R'_{FG}}(t_l), \mu_{R''_{FG}}(t_l)\}$ si R_{FG} es la “intersección difusa generalizada” de las relaciones R'_{FG} y R''_{FG}

Mediante la anterior definición, combinamos los diferentes grados de compatibilidad de una tupla dada, para obtener un grado de pertenencia a la relación de dicha tupla. La forma en que combinamos dichos grados está en función de los conectivos empleados entre las condiciones impuestas sobre cada atributo. Una vez obtenido dicho grado obtenemos la “cardinalidad difusa generalizada” sumando los grados obtenidos para cada una de las tuplas de la relación.

• Cardinalidad Relativa Generalizada

A veces es preciso evaluar que proporción de la cantidad de tuplas que verifican una condición C_1 , verifica también otra condición C_2 . Esto nos conduce al concepto de “cardinalidad relativa”. La cardinalidad “relativa” se puede construir a partir de la cardinalidad “absoluta”. En [Zadeh 83] se recogen varias alternativas para evaluar dicha cardinalidad en el ámbito difuso. En concreto, si partimos del empleo de la *sigma-count* para evaluar la cardinalidad “absoluta”, podemos encontrar para la cardinalidad “relativa” la siguiente definición ([DeLuca 72] y [Zadeh 72]):

Definición 4.19 Sean F y G dos conjuntos difusos entonces la expresión de la **sigma-count relativa** de F con respecto a G se define como:

$$\sum Count(F/G) = \frac{\sum Count(F \cap G)}{\sum Count(G)} \quad (4.34)$$

con la \cap dada por la T -norma del mínimo.

Como puede verse de su definición, la *sigma-count* relativa toma valores en el intervalo $[0,1]$.

De la anterior expresión y de la def. 4.18 resulta natural definir la cardinalidad relativa de dos “relaciones difusa generalizadas” como sigue:

Definición 4.20 Sean R_{FG} y R'_{FG} dos “relaciones difusa generalizadas”, entonces la expresión de la **Cardinalidad Relativa Generalizada** de R_{FG} con respecto a R'_{FG} , $card(R_{FG}/R'_{FG})$, se define como:

$$card(R_{FG}/R'_{FG}) = \frac{card(R_{FG} \cap_G R'_{FG})}{card(R'_{FG})} \quad (4.35)$$

donde \cap_G representa la “intersección difusa generalizada”.

4.4 Ejemplo de Representación y Manipulación sobre GEFRED

Para ilustrar el funcionamiento del modelo vamos a emplear el ejemplo de una relación que recoge datos, algunos de ellos de naturaleza difusa, sobre un conjunto de empleados (tabla 4.2). Este ejemplo no agota todas las posibilidades de GEFRED, tanto en la representación, como en la manipulación de información difusa.

Suponemos que la información que almacenan los atributos NOMBRE y CALLE son de naturaleza precisa. Los dominios que subyacen a EDAD y SALARIO, son numéricos y tienen definidos sobre ellos las distribuciones de posibilidad, (expresadas en forma trapezoidal), etiquetadas y definidas como muestra la figura 4.1. Se contempla la presencia de valores “Unknown” en ambos atributos pero, dada la naturaleza de la relación, no existen valores “undefined” y por tanto tampoco “NULL”. El dominio que subyace al atributo RENDIMIENTO, es del tipo 1 de la tabla 4.1, en la tabla 4.3

\mathcal{H}	NOMBRE	CALLE	EDAD	RENDIMIENTO	SALARIO
\mathcal{B}	Luis	Recogidas	&.8/30,1/31	Bueno	110000
	Antonio	Reyes Católicos	Maduro	Regular	100000
	Juan Carlos	Camino Ronda	Joven	Malo	90000
	Francisco	P. A. Alarcón	Mayor	Excelente	150000
	Julia	Puerta Real	Joven	Bueno	130000
	Inés	Manuel de Falla	#28	Bueno	125000
	Javier	Gran Vía	*30,35	Regular	105000

El símbolo # significa “aproximadamente” y su definición aparece en la figura 4.1, & indica distribución de posibilidad y * intervalo.

Tabla 4.2: Tabla de Empleados, Emp

$s_e(d, d')$	Malo	Regular	Bueno	Excelente
Malo	1	0.8	0.5	0.1
Regular	0.8	1	0.7	0.5
Bueno	0.5	0.7	1	0.8
Excelente	0.1	0.5	0.8	1

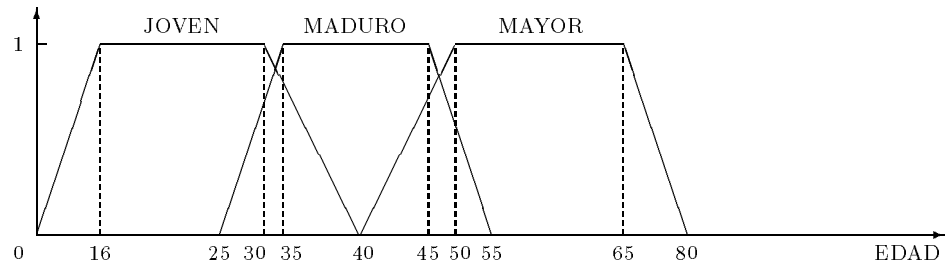
Tabla 4.3: Relación de semejanza sobre el dominio RENDIMIENTO.

se especifican los valores del dominio así como la relación de semejanza que existe entre ellos. También permite en principio valores “Unknown”.

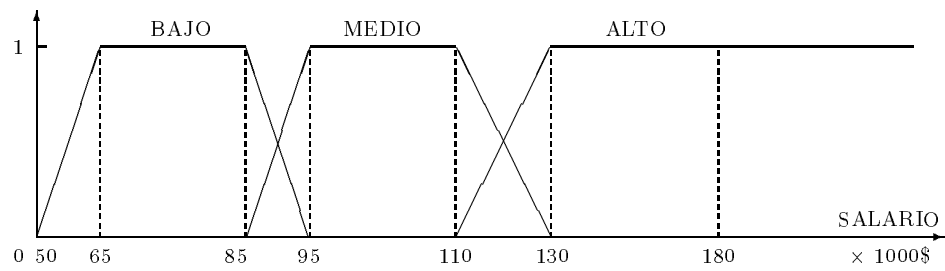
De acuerdo con el modelo GEFRED, la “cabecera” de la relación difusa generalizada que estructura la información vendría dada por:

$$\mathcal{H} = \{ \text{NOMBRE} : D_{\text{NOMBRE}}, \text{CALLE} : D_{\text{CALLE}}, \text{EDAD} : D_{\text{EDAD}}, \text{RENDIMIENTO} : D_{\text{RENDIMIENTO}}, \text{SALARIO} : D_{\text{SALARIO}} \} \quad (4.36)$$

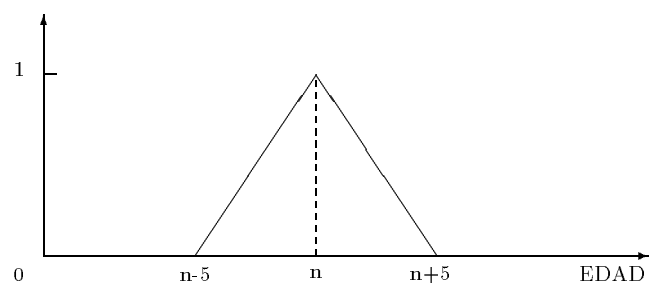
Donde, no aparecen atributos de compatibilidad puesto que se trata de una relación de partida (relación base). El “cuerpo” de la relación está constituido por todas las tuplas mostradas en la relación de la tabla 4.2.



a)



b)



c)

Figura 4.1: Definición de etiquetas sobre los atributos EDAD y SALARIO.

Para poner de manifiesto el funcionamiento del *Álgebra Relacional Difusa Generalizada*, vamos a estudiar el proceso de resolución de una consulta ejemplo.

La consulta en cuestión pregunta por los empleados que, con un rendimiento “bueno”, perciban un salario “alto”, o que, con una edad “mayor”, tengan un salario “medio”. En términos más precisos, podría ser formulada:

Consulta 4.1 *Dame el NOMBRE, la EDAD, el RENDIMIENTO y el SALARIO, así como el grado en que cada atributo satisface la condición, de aquellos empleados que presenten RENDIMIENTO “bueno” (con un grado ≥ 0.9) y un SALARIO “alto” (con un grado ≥ 0.7) ó que tengan una EDAD “mayor” (con un grado ≥ 0.6) y un SALARIO “medio” (con un grado ≥ 0.7)*

Los “comparadores difusos generalizados” implícitos en la consulta deberán, o estar definidos en la implementación, o bien estar almacenados como componentes del meta-conocimiento de la Base de Datos Difusa. GEFRED, en este aspecto, no precisa como se ha de modelar el comparador de forma específica. Esto es así, porque la naturaleza de la información, condiciona, además de la representación de los datos, la forma de operar con ellos. En nuestro ejemplo vamos a emplear los siguientes modelos para los comparadores implicados en la consulta:

- Para la compatibilidad en los atributos EDAD y SALARIO empleamos el “comparador extendido” habitual $=_e(d, d') = \delta(d, d')$ y el “comparador difuso generalizado” inducido:

$$\Theta^{=e}(\tilde{d}, \tilde{d}') = \sup_{(d, d') \in D \times D} \min(=_e(d, d'), \pi_{\tilde{d}}(d), \pi_{\tilde{d}'}(d')) \quad (4.37)$$

donde, $\pi_{\tilde{d}}(d)$ y $\pi_{\tilde{d}'}(d')$, son las distribuciones de posibilidad asociadas a \tilde{d} y a \tilde{d}' , respectivamente.

- Para RENDIMIENTO emplearemos el “comparador extendido” $s_e(d, d')$ dado en la Tabla 4.3. El “comparador difuso generalizado” usado será:

$$\Theta^{s_e}(\tilde{d}, \tilde{d}') = \sup_{(d, d') \in D \times D} \min(s_e(d, d'), \pi_{\tilde{d}}(d), \pi_{\tilde{d}'}(d')) \quad (4.38)$$

donde nuevamente, $\pi_{\tilde{d}}(d)$ y $\pi_{\tilde{d}'}(d')$, son las distribuciones de posibilidad asociadas a \tilde{d} y a \tilde{d}' , respectivamente.

Una vez precisados los comparadores, la consulta, en los términos del Álgebra de GEFRED, queda expresada como sigue:

$$\mathcal{P}_G((\mathcal{S}_1 \wedge_G \mathcal{S}_2) \vee_G (\mathcal{S}_3 \wedge_G \mathcal{S}_4) ; X) \quad (4.39)$$

donde:

- $\mathcal{S}_1 = \mathcal{S}_G(Emp; \Theta^{se}(RENDIMIENTO, Bueno) \geq 0.9)$ con “Bueno” $\in D_{REND}$.
- $\mathcal{S}_2 = \mathcal{S}_G(Emp; \Theta^{=e}(SALARIO, Alto) \geq 0.7)$ siendo “Alto” la etiqueta que referencia la distribución de posibilidad mostrada en la figura 4.1.
- $\mathcal{S}_3 = \mathcal{S}_G(Emp; \Theta^{=e}(EDAD, Mayor) \geq 0.6)$ siendo “Mayor” la etiqueta que referencia la distribución de posibilidad mostrada en la figura 4.1.
- $\mathcal{S}_4 = \mathcal{S}_G(Emp; \Theta^{=e}(SALARIO, Medio) \geq 0.7)$ siendo “Medio” la etiqueta que referencia la distribución de posibilidad mostrada en la figura 4.1.
- $X = \{NOMB, EDAD, c_{EDAD}, REND, c_{REND}, SAL, c_{SAL}\}$

En las tablas 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10 y 4.11, se detallan las diferentes operaciones llevadas a cabo para resolver la consulta.

\mathcal{H}	NOMBRE	CALLE	EDAD	REND	c_{REND}	SALARIO
\mathcal{B}	Luis	Recogidas	&.8/30,1/31	Bueno	1	110000
	Julia	Puerta Real	Joven	Bueno	1	130000
	Inés	Manuel de Falla	#28	Bueno	1	125000

Los valores de c_{REND} se obtienen de la relación de semejanza mostrada en la Tabla 4.3. Las tuplas que presentan un valor $< \gamma = 0.9$ para el grado de pertenencia del atributo RENDIMIENTO no pertenecen a la relación.

Tabla 4.4: Cálculo de \mathcal{S}_1

\mathcal{H}	NOMBRE	CALLE	EDAD	RENDIMIENTO	SALARIO	c_{SAL}
\mathcal{B}	Francisco	P. A. Alarcón	Mayor	Excelente	150000	1
	Julia	Puerta Real	Joven	Bueno	130000	1
	Inés	Manuel de Falla	#28	Bueno	125000	0.75

Los valores de c_{SAL} , que se calculan según la ecuación 4.37, para etiquetas trapezoidales pueden ser calculados de forma trigonométrica. Las tuplas que presentan un valor $< \gamma = 0.7$ para el grado de pertenencia del atributo SAL no pertenecen a la relación.

Tabla 4.5: Cálculo de \mathcal{S}_2

\mathcal{H}	NOMBRE	CALLE	EDAD	c_{EDAD}	REND.	SALARIO
\mathcal{B}	Antonio	Reyes Católicos	Maduro	0.75	Regular	100000
	Francisco	P. A. Alarcón	Mayor	1	Excelente	150000

Los valores de c_{EDAD} , que se calculan según la ecuación 4.37, para etiquetas trapezoidales pueden ser calculados de forma trigonométrica. Las tuplas que presentan un valor $< \gamma = 0.6$ para el grado de pertenencia del atributo EDAD no pertenecen a la relación.

Tabla 4.6: Cálculo de \mathcal{S}_3

\mathcal{H}	NOMBRE	CALLE	EDAD	REND.	SALARIO	c_{SAL}
\mathcal{B}	Luis	Recogidas	&.8/30,1/31	Bueno	110000	1
	Antonio	Reyes Católicos	Maduro	Regular	100000	1
	Javier	Gran Vía	*30,35	Regular	105000	1

Los valores de c_{SAL} , que se calculan según la ecuación 4.37, para etiquetas trapecoidales pueden ser calculados de forma trigonométrica. Las tuplas que presentan un valor $< \gamma = 0.7$ para el grado de pertenencia del atributo SAL no pertenecen a la relación.

Tabla 4.7: Cálculo de \mathcal{S}_4

\mathcal{H}	NOMBRE	CALLE	EDAD	REND	c_{REND}	SALARIO	c_{SAL}
\mathcal{B}	Julia	Puerta Real	Joven	Bueno	1	130000	1
	Inés	Manuel de Falla	#28	Bueno	1	125000	0.75

Para el cálculo de la “Conjunción Generalizada Difusa” se halla la “Intersección Generalizada Difusa” de las relaciones calculadas por las consultas \mathcal{S}_1 y \mathcal{S}_2 .

Tabla 4.8: Cálculo de $\mathcal{S}_1 \wedge_G \mathcal{S}_2$

\mathcal{H}	NOMBRE	CALLE	EDAD	c_{EDAD}	REND	SALARIO	c_{SAL}
\mathcal{B}	Antonio	Reyes Católicos	Maduro	0.75	Regular	100000	1

Para el cálculo de la “Conjunción Generalizada Difusa” se halla la “Intersección Generalizada Difusa” de las relaciones calculadas por las consultas \mathcal{S}_3 y \mathcal{S}_4 .

Tabla 4.9: Cálculo de $\mathcal{S}_3 \wedge_G \mathcal{S}_4$

\mathcal{H}	...	EDAD	c_{EDAD}	RENDIMIENTO	c_{REND}	SALARIO	c_{SAL}
\mathcal{B}	...	Maduro	0.75	Regular	0	100000	1
	...	Joven	0	Bueno	1	130000	1
	...	#28	0	Bueno	1	125000	0.75

Para el cálculo de la “Disyunción Difusa Generalizada” se halla la “Unión Difusa Generalizada” de $(\mathcal{S}_1 \wedge_G \mathcal{S}_2)$ y $(\mathcal{S}_3 \wedge_G \mathcal{S}_4)$.

Tabla 4.10: Cálculo de $((\mathcal{S}_1 \wedge_G \mathcal{S}_2) \vee_G (\mathcal{S}_3 \wedge_G \mathcal{S}_4))$

\mathcal{H}	NOMBRE	EDAD	c_{EDAD}	RENDIMIENTO	c_{REND}	SALARIO	c_{SAL}
\mathcal{B}	Antonio	Maduro	0.75	Regular	0	100000	1
	Julia	Joven	0	Bueno	1	130000	1
	Inés	#28	0	Bueno	1	125000	0.75

Se proyecta sobre los atributos, ya sean de valor ó de compatibilidad.

Tabla 4.11: $\mathcal{P}_G((\mathcal{S}_1 \wedge_G \mathcal{S}_2) \vee_G (\mathcal{S}_3 \wedge_G \mathcal{S}_4) ; X)$

4.5 Estudio de GEFRED frente a los Modelos Teóricos Precedentes

Para ilustrar la capacidad del modelo introducido, vamos a dedicar este apartado a analizar algunos de los aspectos de GEFRED confrontándolos con los modelos previos más importantes. Pondremos mayor énfasis en el carácter general que proporciona, tanto al tratamiento, como a la manipulación de la información. En este sentido mostraremos, en contraste con los modelos precedentes, como GEFRED puede ofrecer tratamiento para casi todas las vertientes que constituyen la información difusa. Veremos también como proporciona herramientas para adoptar en cada momento el comparador más apropiado para recoger el significado de cada operación de consulta. En esta línea, demostraremos que puede elaborar los mecanismos de consulta de modelos anteriores como casos particulares del mecanismo global adoptado. Concretamente, el *álgebra difusa generalizada* en combinación con la definición de *comparador difuso generalizado* proporciona una potente herramienta para elaborar la información.

El análisis comparativo que presentamos a continuación no pretende ser exhaustivo. Nos ceñiremos al estudio de los aspectos distintivos de cada modelo desde la óptica de GEFRED. Tales aspectos son los tipos de datos que representan y como lo hacen, los operadores que definen y como funcionan, y la forma en que resuelven las consultas.

4.5.1 GEFRED vs. Modelo de Buckles-Petry

Este modelo representa una vertiente distinta, en cuanto a la forma de representar y manipular la información imprecisa en el ámbito relacional. Aunque la manera en que resuelve los aspectos de manipulación resulta elegante desde el punto de vista formal matemático, la variedad de tipos de información sobre los que opera y el trato semántico que reciben es muy limitado y poco operativo. Veamos sus aspectos más importantes bajo la perspectiva de GEFRED.

4.5.1.1 Estructura de Datos

Como vimos en el capítulo anterior, este modelo operaba en principio sobre conjuntos de escalares y de números, (después se incorporará el tratamiento para números difusos, [Buckles, Petry, 84]). De cualquier forma, existe un aspecto de esta representación que

GEFRED no contempla en los mismos términos. En este modelo el concepto de conjunto tiene una connotación de “no excluyente” frente a la connotación de “excluyente” asociada a las distribuciones de posibilidad sobre las que se contruye la definición de *dominio difuso generalizado*. Es decir que, por ejemplo, este modelo puede representar mediante un valor de dominio que “Luis habla inglés y francés”, cosa que en GEFRED requiere del empleo de dos tuplas ya que, mediante distribuciones de posibilidad, lo que podríamos representar como valor de dominio es que “Luis habla inglés o francés”. La forma en que opera el modelo de Buckles-Petry con este tipo de información nos parece contraria al principio de la atomicidad de la información preconizado por el modelo relacional clásico. Además, esto plantea problemas a los procesos de normalización y de detección de dependencias funcionales. Por ello, sin renunciar a poder representar información de este tipo, GEFRED evita la forma en que se resuelve en el modelo a estudio.

En este modelo el concepto de “relación difusa” queda extendido como un subconjunto del producto cartesiano $2^{D_1} \times \dots \times 2^{D_n}$, siendo 2^{D_i} un subconjunto del dominio de discurso D_i . Después, con la incorporación de tratamiento para números difusos, el dominio extendería su capacidad para representar este tipo de datos. Con la salvedad introducida en el párrafo anterior, GEFRED puede articular el concepto de “relación difusa” como un caso particular de la definición de *relación difusa generalizada*.

4.5.1.2 Operaciones Algebraicas

El modelo de Buckles-Petry basa las operaciones de manipulación en el concepto de similaridad. Formalmente recurre al uso de las *relaciones de similitud* para “relajar” el concepto de “igualdad” entre los valores de un dominio dado. Mediante las relaciones de similitud, el grado en que dos valores de dominio son iguales toma un valor comprendido entre 0 y 1. Por tanto, si D es el dominio de discurso, que en este modelo es discreto y finito, se define la “similaridad” entre dos valores de dicho dominio como una relación de difusa dada por

$$s : D \times D \longrightarrow [0, 1] \quad (4.40)$$

$$s(d_i, d_j) \longmapsto [0, 1]$$

con $d_i, d_j \in D$, cumpliendo:

1. Reflexiva: $s(d_i, d_i) = 1$

2. Simétrica: $s(d_i, d_j) = s(d_j, d_i)$

3. Transitividad Max-Min: $s(d_i, d_j) \geq \max_{\forall d_k \in D} \{\min[s(d_i, d_k), s(d_k, d_j)]\}$

Como puede verse, una relación de similaridad es una relación difusa y por tanto un comparador extendido en el sentido expresado en la ecuación 4.8. Puesto que un comparador extendido es un caso particular de *comparador difuso generalizado*, podemos emplearlo en el *álgebra difusa generalizada* para resolver una consulta en términos parecidos a los empleados en el modelo de Buckles-Petry.

4.5.1.3 Resolución de Consultas

Utilizaremos el ejemplo del subapartado 3.5.2 para ilustrar como podría resolverse tal consulta mediante el empleo de GEFRED. En primer lugar, expresaremos la relación mostrada en la tabla 3.4 en los términos establecidos por la *relación difusa generalizada*. La tabla 4.12 muestra la nueva representación donde, el conjunto representado por los valores {alto,muy alto} ha sido reemplazado por el concepto “*bastante alto*”, dado por una distribución de posibilidad. Obsérvese que no hay pérdida en el cambio de representación, más bien al contrario, ya que el nuevo valor adoptado permite representar el concepto de que un individuo posee una altura que está entre lo que podemos considerar como “*alto*” y “*muy alto*”. Además el concepto anterior se adecuaba más a la aproximación de considerar las distribuciones de posibilidad como construidas sobre valores *excluyentes* que sobre *no excluyentes*. En efecto, la altura de un individuo toma un único valor y no dos a la vez. Como habíamos comentado, nos parece que uno de los mayores inconvenientes que presenta el modelo de Buckles-Petry es el empleo que hace de este concepto. De acuerdo con lo anteriormente expuesto, la edad de Mario también puede ser representada mediante la distribución de posibilidad mostrada en la tabla 4.12

Los valores “*bastante alto*”, “*medio*”, “*bajo*” y “*muy bajo*” son “*etiquetas lingüísticas*”. La forma que toman las distribuciones de posibilidad asociadas a dichas etiquetas no es relevante en el ejemplo, puesto que no intervienen en las condiciones de la consulta. Los valores “*castaño*”, “*moreno*”, “*rubio*” y “*pelirrojo*” son escalares que representan conceptos próximos en el sentido apuntado en el subapartado 4.2.1. Dicha proximidad es modelada mediante la relación de similitud mostrada en la tabla 3.3 que reproducimos nuevamente.

H	Nombre	Edad	Altura	Color_de_pelo
B	Juan	16	Bastante Alto	Castaño
	José	17	Bajo	Moreno
	Pedro	15	Muy Bajo	Rubio
	Mario	{1/15,1/16}	Media	Pelirrojo

Tabla 4.12: Relación Persona

s	RU	PR	CA	MO
RU	1	.6	.4	0
PR	.6	1	.5	.1
CA	.4	.5	1	.8
MO	0	.1	.8	1

Tabla 4.13: Relación de Similitud sobre el dominio “color de pelo”

Como vemos la relación de similitud mostrada se puede emplear como un *comparador extendido*, s_e , podemos, por ejemplo, construir sobre ella el siguiente *comparador difuso generalizado*:

$$\Theta^{s_e}(\tilde{d}, \tilde{d}') = \sup_{(d,d') \in D \times D} \min(s_e(d, d'), \pi_{\tilde{d}}(d), \pi_{\tilde{d}'}(d')) \tag{4.41}$$

siendo d y d' valores del dominio color del pelo.

Este comparador, como veremos en el subapartado dedicado al modelo de Prade-Testemale, se puede demostrar que satisface la definición de *comparador difuso generalizado*.

La consulta del ejemplo:

Consulta 4.2 Dame las personas y su altura con un color de pelo similar a castaño en grado 0.6.

se formularía en términos del álgebra de GEFRED de la siguiente manera:

$$\mathcal{P}_G(\mathcal{S}_{color} ; X) \tag{4.42}$$

donde

- $\mathcal{S}_{color} = \mathcal{S}_G(Persona; \Theta^{se}(\text{Color_de_pelo, castaño}) \geq 0.6)$
con “castaño” $\in D_{Color_de_pelo}$.
- $X = \{Nombre, Altura\}$

Dicha consulta se resolvería como muestran las tablas 4.14 y 4.15.

H	Nombre	Edad	Altura	Color_de_pelo	c_{col_pel}
B	Juan	16	Bastante Alto	Castaño	1
	José	17	Bajo	Moreno	0.8

Tabla 4.14: Cálculo de la Selección \mathcal{S}_{color}

H	Nombre	Altura
B	Juan	Bastante Alto
	José	Bajo

Tabla 4.15: Resultado final de la Consulta

El resultado obtenido sobre este ejemplo mediante el empleo de GEFRED, presenta varias diferencias con respecto al resultado mostrado en la tabla 3.6.

1. Con el modelo de Buckles-Petry obtenemos una “*tupla de equivalencia*”, la cual puede interpretarse de varias maneras. Entre ellas, por ejemplo, sería válida la interpretación: *(José, muy alto)* información que a todas luces no se encuentra presente en la relación de partida. Además, está en contradicción con la información de partida *(José, bajo)*. Sin embargo, mediante el empleo de GEFRED obtenemos las interpretaciones que son consistentes con la información de partida presente en la base de datos y que es compatible con la consulta formulada.
2. El proceso de elaboración de la consulta es más coherente mediante el empleo de GEFRED, puesto que utiliza la misma estructura de representación, la *relación difusa generalizada*, para todos los pasos intermedios a cubrir por el cálculo.

3. GEFRED permite, al igual que ocurre con el modelo de Buckles-Petry, establecer umbrales de consulta para cada atributo envuelto en la misma. Sin embargo, GEFRED va más allá al poder recoger el grado con el que cada atributo de cada tupla satisface la consulta.
4. La representación de la información de partida y los resultados que se obtienen por medio de las consultas mediante GEFRED, resultan más próximos a las necesidades que intuitivamente se podrían plantear en la utilización de un sistema de almacenamiento de información difusa.

Como conclusión, GEFRED recoge los aspectos positivos del modelo de Buckles-Petry y proporciona mecanismos alternativos para resolver algunos de los problemas que este modelo plantea.

4.5.2 GEFRED vs. Modelo de Umano-Fukami

El modelo de Umano-Fukami presenta una aproximación posibilística, cuyo aspecto principal es la forma en que se construyen los predicados envueltos en los procesos de consulta y en como clasifica las respuestas a dicha consulta. Estudiaremos, bajo la perspectiva de GEFRED, los aspectos relativos a la representación de los datos, a la manipulación de los mismos y a la forma en que resuelve las consultas.

4.5.2.1 Estructura de Datos

En este aspecto el modelo en estudio opera sobre un tipo de dominio compatible con el definido en el subapartado 4.2.1. A partir de aquí, introduce el concepto de relación extendida, R_i , ([Umano 82]), que es definida como un subconjunto del producto Cartesiano de una colección de distribuciones de posibilidad dado por

$$R_i \subseteq (\mathcal{P}(U_{i1}) \cup \{NULL\}) \times \mathcal{P}(U_{i2}) \cup \{NULL\} \times \dots \times \mathcal{P}(U_{im}) \cup \{NULL\} \quad (4.43)$$

donde los símbolos \times , \cup y \subseteq denotan el producto Cartesiano, la unión y subconjunto, respectivamente, $\mathcal{P}(U_{ij})$, $j = 1, 2, \dots, m$, son colecciones de todas las distribuciones de posibilidad en el universo de discurso U_{ij} y NULL es un valor especial que representa la situación en la que no conocemos ni siquiera si un valor es aplicable o no.

Como vemos esta estructura de relación puede ser asimilada por el concepto de *relación difusa generalizada* como un caso particular en el que no aparecen atributos

de compatibilidad. Por tanto, GEFRED puede representar cualquier tipo de relación soportada por el modelo de Umano-Fukami.

4.5.2.2 Operaciones Algebraicas

El modelo de Umano-Fukami contempla una serie de predicados u operadores para la manipulación de datos. Según esté definido cada predicado obtiene los datos clasificados en dos conjuntos: los que estrictamente satisfacen dicho predicado y los que posiblemente lo satisfacen. Vamos a mostrar como algunos de los predicados propuestos en el mencionado modelo pueden ser formulados empleando comparadores difusos generalizados. Para hacerlo, construiremos dos versiones de *comparador difuso generalizado* para cada predicado. Una, calcula el grado en que se satisface estrictamente un predicado, (si se satisface), y la otra, el grado en que posiblemente se satisface el mismo predicado. Como ejemplo, a continuación detallamos el proceso para el predicado EQ.

Según el modelo de Umano-Fukami si \tilde{d} y \tilde{d}' son distribuciones de posibilidad, el predicado $EQ(\tilde{d}, \tilde{d}')$ puede evaluarse de alguna de las siguientes maneras:

1. $\langle T, 0 \rangle$ si $\tilde{d} \cap \tilde{d}' = \emptyset$
2. $\langle T, 1 \rangle$ si $\tilde{d} = \{1/d_i\}_p$ y $\tilde{d}' = \{1/d_i\}_p$ con $d_i \in D$
3. $\langle P, 1 \rangle$ en cualquier otro caso.

De acuerdo con esto, mediante el predicado EQ podemos establecer de forma categórica que dos datos son iguales si son precisos y poseen el mismo valor en el dominio de discurso, que dos datos son distintos si las distribuciones de posibilidad que los representan no poseen ningun valor en común en el dominio de discurso y, que *pueden* ser igual en caso de que no se de ninguna de las anteriores circunstancias. En definitiva, el conjunto de distribuciones de posibilidad que no hacen falso este predicado se clasifica en dos conjuntos, uno que agrupa los valores que satisfacen en forma *estricta* dicho predicado y otro que *posiblemente* lo satisface.

Vamos a ver como, empleando comparadores difusos generalizados, podemos calcular el grado con el que dos datos satisfacen el predicado para cada uno de los conjuntos anteriores. Para ello vamos a construir dos comparadores. El primero de ellos, que denominaremos versión *estricta* del comparador EQ, lo notaremos como EQ_T^s . El

superíndice δ indica que este comparador difuso está inducido por el *comparador extendido* habitual definido para la igualdad clásica sobre el dominio de discurso. La forma concreta en que opera este comparador viene dada por la siguiente expresión:

$$EQ_T^\delta : D_G \times D_G \rightarrow [0,1]$$

$$EQ_T^\delta(\tilde{d}, \tilde{d}') = \begin{cases} 1 & \text{si } \tilde{d} = 1/u \text{ y } \tilde{d}' = 1/u \text{ } u \in D \\ 0 & \text{en cualquier otro caso} \end{cases} \quad (4.44)$$

El segundo comparador, denominado versión *posible* del comparador EQ, vendrá notado como EQ_P^δ y se define de la siguiente forma:

$$EQ_P^\delta : D_G \times D_G \rightarrow [0,1]$$

$$EQ_P^\delta(\tilde{d}, \tilde{d}') = \begin{cases} 1 & \text{si } \tilde{d} \cap \tilde{d}' \neq \emptyset \\ 0 & \text{en cualquier otro caso} \end{cases} \quad (4.45)$$

donde la intersección viene dada por la T-norma del mínimo. Es fácil comprobar que ambos comparadores satisfacen la definición de comparador difuso generalizado. Efectivamente, se aplican sobre dominios difusos generalizados proporcionando valores en el intervalo unidad y cumplen respectivamente:

$$EQ_T^\delta(\tilde{d}, \tilde{d}') = \delta(d, d')$$

y

$$EQ_P^\delta(\tilde{d}, \tilde{d}') = \delta(d, d')$$

que valdrán 1 en el caso de que los valores d y d' sean iguales y 0 en caso contrario. En efecto, si \tilde{d} y \tilde{d}' son las distribuciones de posibilidad inducidas por los valores d y $d' \in D$, la T-norma del mínimo devuelve un valor igual a 1 si ambos coinciden y 0 si son distintos. Este resultado es igual al que se obtiene mediante el comparador extendido, δ definido sobre el dominio de discurso.

Como consecuencia de las anteriores definiciones, podemos comprobar que un par de valores satisfacen el predicado EQ con un par de verdad $\langle T, 1 \rangle$ si satisfacen con grado 1 la versión *estricta* del comparador EQ, EQ_T^δ , que lo satisfacen con el par de verdad $\langle P, 1 \rangle$ si satisfacen con grado 1 la versión *posible* de EQ, EQ_P^δ , y con grado 0 la versión *estricta* y, que lo satisfacen con el par de verdad $\langle T, 0 \rangle$ si satisfacen ambas versiones con grado 0.

Veamos a continuación como GEFRED puede modelar algunos de los aspectos envueltos en una consulta resuelta por medio del modelo de Umano-Fukami.

4.5.2.3 Resolución de Consultas

El modelo analizado construye las consultas por medio de una serie de predicados definidos en el mismo o por el usuario. Atendiendo a un predicado simple, (condición simple), una tupla puede ser clasificada como que ciertamente satisface la condición con un grado t , como que posiblemente la satisface y lo hace con un grado t o que ciertamente no la satisface. Como hemos visto, estas tres situaciones se encuentran representadas en el modelo mediante un par de verdad asociado a la tupla para ese predicado con un valor $\langle T, t \rangle$, $\langle P, t \rangle$ y $\langle T, 0 \rangle$, respectivamente.

\mathcal{H}	NOMBRE	EDAD	NOMBRE_HIJOS
\mathcal{B}	Tom	23	Ted
	Susan	35	John
	Susan	35	Mike
	Richard	40	{1/Judy, 1/Anna}
	Raymond	joven	UNKNOWN
	Victor	UNKNOWN	UNDEFINED
	Smith	{1/50, 1/51}	NULL

Tabla 4.16: Relación Persona

Vamos a utilizar el ejemplo del subapartado 3.5.3.1, cuya relación, expresada como una *relación difusa generalizada*, reproducimos en la tabla 4.16, para ilustrar como el empleo de predicados difusos en una consulta puede resolverse usando GEFRED. Para ello veamos como se resuelve la petición:

Consulta 4.3 “*Dame las personas jóvenes*”

Donde el predicado “*joven*” viene dado por la distribución de posibilidad

$$joven = \{1/19, 1/20, 1/21, 1/22, 0.8/23, 0.6/24, 0.3/25\} \quad (4.46)$$

Veamos como GEFRED es capaz de resolver la consulta anterior en los mismos términos que los planteados por el modelo de Umano-Fukami. Para ello, hemos de introducir un par de *comparadores difusos generalizados* para resolver el predicado

“*ser joven*” para los dos tipos de conjuntos devueltos por el modelo. La definición de ambos viene dada por:

$$FEQ_T^\delta(\tilde{d}, \tilde{J}) = \begin{cases} t & \text{si } \mu_J(x) = t \forall x \in \tilde{d} \cap \tilde{J} \\ 0 & \text{en cualquier otro caso} \end{cases} \quad (4.47)$$

$$FEQ_P^\delta(\tilde{d}, \tilde{J}) = \max_{x \in \tilde{d} \cap \tilde{J}} (\mu_J(x)) \quad (4.48)$$

donde, δ es el comparador extendido habitual para la igualdad clásica, \tilde{J} es la distribución de posibilidad que modela el predicado “*Joven*” y \cap es la intersección dada por el mínimo.

Se puede observar como las expresiones 4.44 y 4.45 son casos particulares de las 4.47 y 4.48 respectivamente. Las últimas “*dispersan*” el concepto de igualdad.

Vamos a demostrar en primer lugar que las ec.s, 4.47 y 4.48 constituyen *comparadores difusos generalizados*. Para ello comprobamos:

1. Que operan sobre datos incluidos en la definición de dominio Difuso. Esto es cierto en tanto que operan sobre distribuciones de posibilidad.
2. Devuelven valores en el intervalo unidad. Si las distribuciones de posibilidad están normalizadas, las operaciones realizadas entregan valores en dicho intervalo.
3. Que sobre valores discretos se obtenga resultados compatibles con el *comparador extendido* sobre el que están inducidas:

$$FEQ_T^\delta(\tilde{d}, \tilde{d}') = \delta(d, d') \quad (4.49)$$

$$FEQ_P^\delta(\tilde{d}, \tilde{d}') = \delta(d, d') \quad (4.50)$$

es fácil ver que se satisface puesto que, para valores precisos, d y d' , la ec. 4.47 da un valor $\neq 0$ solo en el caso en que $d = d' \Rightarrow \delta(d, d') = 1$. En la ecuación 4.48, para valores precisos, la intersección $\tilde{d} \cap \tilde{d}'$ es distinta de vacío y vale 1 únicamente cuando $d = d' \Rightarrow \delta(d, d') = 1$.

Vamos a mostrar como, mediante empleo de los comparadores introducidos en 4.47 y 4.48, podemos obtener para la consulta 4.3 el mismo resultado que el proporcionado por el modelo de Umamo-Fukami. Dicho resultado, calculado en el apartado 3.5.3.1, venía dado por los siguientes conjuntos de tuplas:

claramente satisfacen = {0.8/Tom}
 posiblemente satisfacen = {1/Raymond,1/Victor}

Para obtener el primer conjunto, el álgebra de GEFRED formularía la siguiente consulta:

$$\mathcal{P}(\mathcal{S}_T^{es-joven}; X) \quad (4.51)$$

donde

- $\mathcal{S}_T^{es-joven} = \mathcal{S}(Persona; FEQ_T^\delta(Edad, "joven") \geq 0.5^1)$, con FEQ_T^δ dado por la ec. 4.47 y "joven" por la distribución de posibilidad 4.46.
- $X = \{Nombre, C_{Nombre}\}$

La resolución de esta consulta se detalla en las tablas 4.17 y 4.18.

\mathcal{H}	NOMBRE	EDAD	C_{EDAD}	NOMBRE_HIJOS
\mathcal{B}	Tom	23	0.8	Ted

Tabla 4.17: Cálculo de $\mathcal{S}_T^{es-joven}$

\mathcal{H}	NOMBRE	C_{EDAD}
\mathcal{B}	Tom	0.8

Tabla 4.18: Cálculo de $\mathcal{P}(\mathcal{S}_T^{es-joven}; X)$

Para obtener el conjunto de individuos que *posiblemente* sean "jóvenes" es preciso emplear el comparador definido en 4.48 y al resultado descontarle los que *claramente* satisfacen la condición. En definitiva, resolver la expresión:

$$\mathcal{P}(\mathcal{S}_P^{es-joven} -_G \mathcal{S}_T^{es-joven}; X) \quad (4.52)$$

donde

¹El modelo de Umano-Fukami emplea este valor umbral por defecto

- $\mathcal{S}_P^{es-joven} = \mathcal{S}(Persona; FEQ_P^\delta(Edad, "joven") \geq 0.5)$, con FEQ_P^δ dado por la ec. 4.48 y "joven" por la distribución de posibilidad 4.46.
- $\mathcal{S}_T^{es-joven}$ se calcula como en 4.51.
- $-_G$ es la *diferencia difusa generalizada*.
- $X = \{Nombre, C_{Nombre}\}$

La tabla 4.19 muestra el resultado de resolver la restricción con el comparador definido en 4.48.

\mathcal{H}	NOMBRE	EDAD	C_{EDAD}	NOMBRE_HIJOS
\mathcal{B}	Tom	23	0.8	Ted
	Raymond	joven	1	UNKNOWN
	Victor	UNKNOWN	1	UNDEFINED

Tabla 4.19: Cálculo de $\mathcal{S}_P^{es-joven}$

La tabla 4.17 refleja el resultado de realizar la selección mediante el operador definido en 4.47 y la tabla 4.20 encuentra la relación diferencia de ambas, para terminar calculando la proyección que muestra el resultado final de la consulta en la tabla 4.21

\mathcal{H}	NOMBRE	EDAD	C_{EDAD}	NOMBRE_HIJOS
\mathcal{B}	Raymond	joven	1	UNKNOWN
	Victor	UNKNOWN	1	UNDEFINED

Tabla 4.20: Cálculo de $\mathcal{S}_P^{es-joven} -_G \mathcal{S}_T^{es-joven}$

El estudio precedente nos sugiere los siguientes comentarios:

1. El dominio considerado por el modelo de Umano-Fukami es un subconjunto del empleado por GEFRED. La relación definida también se puede formular en los términos de GEFRED.

\mathcal{H}	NOMBRE	C_{EDAD}
\mathcal{B}	Raymond	1
	Victor	1

Tabla 4.21: Cálculo de $\mathcal{P}(\mathcal{S}_P^{es-joven} -_G \mathcal{S}_T^{es-joven}; X)$

- Algunos aspectos relativos a la construcción de los operadores de comparación pueden ser elaborados mediante el empleo de nuestro modelo. En este sentido, consideramos que la definición de los predicados difusos no resulta demasiado acertada en tanto que resulta un tanto artificiosa. Cuando, en el ejemplo, se opera con la definición del predicado “*joven*”, mostrada en ec. 4.46, vemos que la forma en que se obtiene el grado en que una tupla satisface “*claramente*” o “*posiblemente*” dicho predicado, no tiene en cuenta para su cálculo, los grados de pertenencia de la distribución de posibilidad que representa cada dato comparado. Es decir, que para el conjunto de “*certeza*” obtendríamos un mismo valor, 1, si la edad de Tom fuera 19 años que si fuera $\{0.2/19, 0.3/20, 0.5/21, 1/22\}$. Por otro lado, resulta aún más extraña la forma en que una tupla puede pasar, de satisfacer “*claramente*” una consulta a “*posiblemente*” satisfacerla. Por ejemplo, si la edad de Tom fuera $\{1/20, 1/21, 1/22\}$ satisfaría “*claramente*” la consulta con grado 1, por contra, si fuera $\{1/20, 1/21, 1/22, 0.2/23\}$ pasaría a “*posiblemente*” satisfacerla con grado 1
- El proceso de resolución de consultas es un poco complicado sobre todo cuando se combinan predicados que, para una misma tupla, son satisfechos “*claramente*” por unos atributos y “*posiblemente*” por otros.

4.5.3 GEFRED vs. Modelo de Prade-Testemale

Los aspectos diferenciadores de este modelo provienen de las medidas de necesidad y de posibilidad empleadas para obtener diferentes respuestas a una condición en una consulta dada. La forma en que construye los operadores de comparación también es una característica importante. Para cada operador construye dos versiones atendiendo a cada una de las medidas anteriores.

4.5.3.1 Estructura de Datos

El marco sobre el que está construida la representación del modelo de Prade-Testemale está constituido por el conjunto de las distribuciones de posibilidad definidas sobre el dominio de discurso considerado, D , más un elemento adicional, $\{e\}$, con el que representa la situación de información “no aplicable”. En el subapartado 3.3.1 se realizó un análisis sobre la forma en que este modelo representa la información incompleta. El mecanismo empleado por GEFRED difiere en cuanto a capacidad para manipular algún matiz de la información ausente.

Concretamente, y remitiendonos a lo visto en el subapartado 3.3, no puede representar información como la mostrada en el siguiente ejemplo: “La posibilidad de que Luis no tenga coche es λ y, caso de que lo tenga es nuevo”. Aunque el modelo de Prade-Testemale incrementa las posibilidades de representación para este tipo de matices, nos parece poco consistente la forma en que extiende el dominio de discurso mediante el elemento $\{e\}$ que representa “no aplicable”. Esto es así porque mezcla datos de naturaleza diferente como pueden ser dominios numéricos con un valor $\{e\}$, cuyas propiedades nada tienen que ver con las que puedan manifestar los números. Consideramos además, que este tipo de aproximación puede dar lugar a problemas similares a los que presenta la aproximación clásica de “valores por defecto” (subapartado 2.4.2). Esto es así debido a que ambas extienden el dominio con un valor artificial que en una representa el caso de que se desconoce el valor y en la otra que se desconoce y es inaplicable.

En cualquier caso la representación adoptada en GEFRED parece mas coherente en cuanto que emplea dominios numéricos y escalares para modelar la información del universo a representar. Sobre esos dominios subyacentes se representan los aspectos relacionados con el *estado del conocimiento* que el sistema posee acerca de la información que almacena, tales aspectos se refieren a la precisión con que se puede aportar una información al sistema. El estado de *valor desconocido* se modela como un caso extremo en el que el conocimiento sobre el valor que puede tomar un dato es nulo y que, por tanto, es posible que adopte cualquier valor. Todos estos aspectos quedan representados mediante distribuciones de posibilidad, incluso el hecho de que una información no sea aplicable a un atributo y el de que se desconozca a que es debido la ausencia de una información, (el que no aparezca porque se desconozca o porque no sea aplicable).

Partiendo del concepto de dominio introducido, el modelo de Prade-Testemale define su propio concepto de “*relación extendida*” como una relación ordinaria definida sobre un producto Cartesiano cuyos elementos son distribuciones de posibilidad construidas sobre $D \cup \{e\}$:

$$R \subseteq \mathcal{P}(D_1 \cup \{e\}) \times \dots \times \mathcal{P}(D_n \cup \{e\}) \quad (4.53)$$

donde R es una *relación extendida*, $D_i, i = 1, \dots, n$ son los dominios de discurso y \times es producto Cartesiano habitual.

GEFRED puede representar la estructura de relación propuesta por el modelo de Prade-Testemale, omitiendo los atributos de compatibilidad, con la salvedad hecha con respecto a la forma en que ambos modelos construyen sus dominios. En lo sucesivo veremos como, obviando esa diferencia, ambos modelos presentan algunas características comunes.

4.5.3.2 Operaciones Algebraicas

El modelo de Prade-Testemale construye sus operadores de comparación a partir de relaciones difusas definidas sobre el dominio de discurso. Esto lo hace en términos similares a los utilizados en la definición del *comparador extendido*. Ofrece dos versiones para dichos operadores, una menos restrictiva basada en la *medida de posibilidad* y otra más fuerte basada en la *medida de necesidad*. La posibilidad de que un atributo A para un objeto x se encuentre en relación Θ con el mismo valor para un atributo B viene dada por la expresión:

$$\Pi(\Theta \mid (A(x), B(x))) = \sup_{(d, d') \in D \times D} \min(\mu_{\Theta}(d, d'), \pi_{A(x)}(d), \pi_{B(x)}(d')) \quad (4.54)$$

donde $\pi_{A(x)}(d)$ y $\pi_{B(x)}(d')$ son las distribuciones de posibilidad correspondientes a $A(x)$ y $B(x)$ respectivamente, y $\mu_{\Theta}(d, d')$ es la relación difusa que representa la relación Θ .

Así mismo, la necesidad de que dos atributos estén relacionados en los mismos términos se expresa:

$$N(\Theta \mid (A(x), B(x))) = \inf_{(d, d') \in D \times D} \max(\mu_{\Theta}(d, d'), 1 - \pi_{A(x)}(d), 1 - \pi_{B(x)}(d')) \quad (4.55)$$

Vamos a demostrar que, si no consideramos la existencia del elemento $\{\epsilon\}$, ambas medidas son casos particulares del concepto de *comparador difuso generalizado*. Para ello deben satisfacerse las siguientes condiciones:

1. Que operen sobre valores considerados dentro de la definición de *dominio difuso generalizado*
2. Que los valores que generen estén incluidos en el intervalo unidad
3. Que operen sobre los valores del dominio de discurso de igual forma en que lo hace la relación difusa sobre la que se construye. O sea, que si $\mu_{\Theta}(d, d')$ es el *comparador extendido* del que parten, cumplan

$$\Pi(\Theta \mid (A(x), B(x))) = \mu_{\Theta}(d_1, d_2) \quad (4.56)$$

$$N(\Theta \mid (A(x), B(x))) = \mu_{\Theta}(d_1, d_2) \quad (4.57)$$

cuando $A(x)$ toma el valor $d_1 \in D$ y $B(x)$ toma el valor $d_2 \in D$

Como la definición de ambas medidas se construye sobre distribuciones de posibilidad y estas, si no se considera el elemento $\{\epsilon\}$, forman parte de la definición del *dominio difuso generalizado*, la primera condición queda satisfecha. De igual forma, de la definición de relación difusa y supuesto que se opera con distribuciones de posibilidad normalizadas, ambas medidas entregan valores en el intervalo unidad, por lo que la segunda condición también se cumple. La tercera condición requiere en primer lugar que la relación difusa dada por la función de pertenencia $\mu_{\Theta}(d, d')$ sea un *comparador extendido*, cosa que es fácil probar puesto que satisface dicha definición. En segundo lugar, si suponemos que las medidas están construidas a partir de dicha relación, habremos de probar que se cumplen las igualdades 4.56 y 4.57. Efectivamente, si d_1 y d_2 son valores precisos pertenecientes al dominio de discurso D , podemos expresarlos mediante las distribuciones de posibilidad $1/d_1, 1/d_2 \in \tilde{\mathcal{P}}(D)$, sustituyendo estos valores en 4.54 y 4.55 obtenemos las siguientes expresiones:

$$\Pi(\Theta \mid (A(x), B(x))) = \sup_{(d, d') \in D \times D} \min(\mu_{\Theta}(d, d'), 1/d_1, 1/d_2) \quad (4.58)$$

$$N(\Theta \mid (A(x), B(x))) = \inf_{(d, d') \in D \times D} \max(\mu_{\Theta}(d, d'), 1 - 1/d_1, 1 - 1/d_2) \quad (4.59)$$

En primer lugar comprobaremos que la expresión 4.58 conduce a la 4.56.

Por definición, la distribución de posibilidad $1/d_1$ toma un valor distinto de 0 únicamente para $d_1 \in D$ en cuyo caso vale 1, lo mismo se puede decir para $1/d_2$ con respecto a d_2 . Así pues, $\min(\mu_\Theta(d, d'), 1/d_1, 1/d_2)$ toma un valor distinto de 0 únicamente para el par $(d_1, d_2) \in D \times D$, siendo dicho valor $\mu_\Theta(d_1, d_2)$, luego, por tanto obtenemos

$$\sup_{(d, d') \in D \times D} \min(\mu_\Theta(d, d'), 1/d_1, 1/d_2) = \min(\mu_\Theta(d_1, d_2), 1, 1) = \mu_\Theta(d_1, d_2) \quad (4.60)$$

con lo cual queda confirmada la igualdad 4.56.

En términos similares podemos obtener 4.57 partiendo de 4.59. Para ello, basta con tener en cuenta que la expresión

$$\max(\mu_\Theta(d, d'), 1 - 1/d_1, 1 - 1/d_2) \quad (4.61)$$

obtenida de la ecuación 4.59, solo puede ser distinta de 1 para el par $(d_1, d_2) \in D \times D$, en cuyo caso obtenemos

$$\max(\mu_\Theta(d_1, d_2), 0, 0) = \mu_\Theta(d_1, d_2) \quad (4.62)$$

como, para el resto de pares $(d, d') \in D \times D$ la expresión 4.61 siempre se evalúa como 1, el ínfimo será

$$\inf_{(d, d') \in D \times D} \max(\mu_\Theta(d, d'), 1 - 1/d_1, 1 - 1/d_2) = \max(\mu_\Theta(d_1, d_2), 0, 0) = \mu_\Theta(d_1, d_2) \quad (4.63)$$

lo cual, en virtud de la expresión 4.62 nos permite aseverar que para valores precisos se cumple la expresión 4.57.

4.5.3.3 Resolución de Consultas

Vamos a emplear el ejemplo descrito en el subapartado 3.5.3.2 para comparar la forma en que los dos modelos resuelven la consulta propuesta en dicho ejemplo. Para ello reproducimos en la tabla 4.22 la relación de dicho ejemplo según la representación adoptada por GEFRED.

La figura 4.2 muestra las funciones de pertenencia que definen las calificaciones cuyas etiquetas lingüísticas aparecen en los atributos Matemáticas y Física. Para el resto de las etiquetas que aparecen en la relación también emplea una representación

\mathcal{H}	NOMBRE	EDAD	MATEMÁTICAS	FÍSICA
\mathcal{B}	Tom	joven	16	[14,16]
	David	20	ligeramente mal	UNDEFINED
	Bob	22	no muy mal	bueno
	Jane	sobre 21	casi bueno	[10,12]
	Jill	joven	alrrededor de 10	alrrededor de 12
	Joe	sobre 24	[14,16]	15
	Jack	[22,25]	UNKNOWN	casi bueno

Tabla 4.22: Relación Estudiantes

trapezoidal descrita por cuadruplas (a, b, c, d) donde, a y b representan los extremos de la porcion de la distribución de posibilidad para la que los valores subyacentes toman valor de pertenencia igual a 1 y, los valores c y d almacenan la distancia desde los anteriores puntos al los límites respectivos a partir de los cuales la función de pertenencia vale 0. De esta forma representa valores precisos, intervalos y valores difusos. Por ejemplo el valor 10 vendría representado mediante una distribución trapezoidal por la cuadrupla $(10, 10, 0, 0)$, el intervalo $[11, 14]$ por $(11, 14, 0, 0)$ y la etiqueta *bueno* de la figura 4.2 por $(14, 16, 1.5, 1)$.

Según esto el resto de los valores difusos de la relación Estudiantes queda descrito como sigue:

- “*Joven*” $\rightarrow (18, 23, 2, 2)$
- “*sobre 21*” $\rightarrow (20, 22, 1, 1)$
- “*sobre 24*” $\rightarrow (23, 25, 1, 1)$

En el subapartado 3.5.3.2 vimos que el modelo de Prade-Testemale resolvía la siguiente petición:

Consulta 4.4 “*Encontrar los nombres de las personas cuya clasificación en Matemáticas es mucho mayor que buena*”

proporcionado los siguientes conjuntos posibilidad y necesidad:

$$\Pi = \{0.3/Tom, 0.3/Joe, 1/Jack\} \quad (4.64)$$

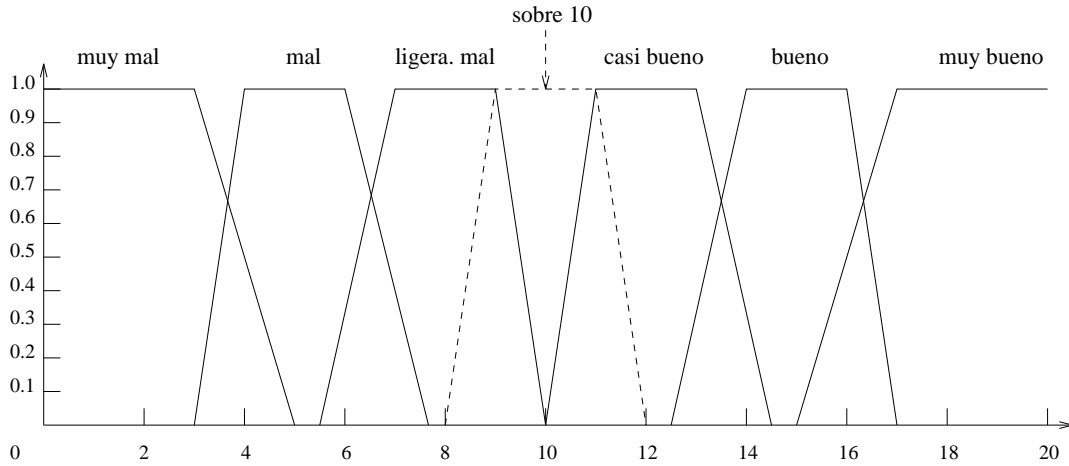


Figura 4.2: Etiquetas lingüísticas definidas sobre Calificación

$$N = \{0.3/Tom\} \tag{4.65}$$

Vamos a mostrar la forma en que GEFRED puede obtener los mismos resultados.

En primer lugar todos los tipos de datos aparecidos en la relación Estudiantes pueden ser modelados mediante el empleo de los conceptos de dominio y relación definidos en GEFRED.

A continuación, para resolver esta consulta, es preciso construir el *comparador difuso generalizado* “mucho mayor que”. Es evidente que, como *comparador extendido*, podemos partir de la expresión definida a nivel de dominio de discurso:

$$\mu_{\gg}(u, v) = \begin{cases} 0 & u - v \leq 2 \\ 0.5 & u - v = 3 \\ 1 & u - v \geq 4 \end{cases} \tag{4.66}$$

Como vimos en el apartado anterior (ec.s 4.57 y 4.56), las expresiones de las medidas de necesidad y posibilidad basadas en la relación difusa 4.66 son comparadores difusos generalizados, por lo que podemos construir los siguientes comparadores basados en dichas medidas:

$$\Theta_N^{\gg}(\tilde{d}_1, \tilde{d}_2) = \inf_{(d,d') \in D \times D} \max(\mu_{\gg}(d, d'), 1 - \pi_{\tilde{d}_1}(d), 1 - \pi_{\tilde{d}_2}(d')) \quad (4.67)$$

$$\Theta_{\Pi}^{\gg}(\tilde{d}_1, \tilde{d}_2) = \sup_{(d,d') \in D \times D} \min(\mu_{\gg}(d, d'), \pi_{\tilde{d}_1}(d), \pi_{\tilde{d}_2}(d')) \quad (4.68)$$

Con estos comparadores estamos en condiciones de formular la consulta 4.4 empleando el álgebra de GEFRED. Para encontrar las personas que necesariamente satisfacen la consulta escribimos:

$$\mathcal{P}_G(\mathcal{S}_N^{\gg \text{bueno}} ; X) \quad (4.69)$$

donde

- $\mathcal{S}_N^{\gg \text{bueno}} = \mathcal{S}_G(\text{Estudiantes}; \Theta_N^{\gg}(\text{Matemáticas}, \text{“bueno”}) > 0)$ con “bueno” definido según la figura 4.2
- $X = \{\text{Nombre}, C_{\text{MATE}}\}$

Las tablas 4.23 y 4.24 detallan el proceso de cálculo.

\mathcal{H}	NOMBRE	EDAD	MATEMÁTICAS	C_{MATE}	FÍSICA
\mathcal{B}	Tom	joven	16	0.3	[14,16]

Tabla 4.23: Cálculo de $\mathcal{S}_N^{\gg \text{bueno}}$

\mathcal{H}	NOMBRE	C_{MATE}
\mathcal{B}	Tom	0.3

Tabla 4.24: Cálculo de $\mathcal{P}_G(\mathcal{S}_N^{\gg \text{bueno}} ; X)$

Para las tuplas que posiblemente satisfacen la consulta se formula de la siguiente manera:

$$\mathcal{P}_G(\mathcal{S}_{\Pi}^{\gg \text{bueno}} ; X) \quad (4.70)$$

donde

- $\mathcal{S}_{\Pi}^{\gg \text{bueno}} = \mathcal{S}_G(\text{Estudiantes}; \Theta_{\Pi}^{\gg}(\text{Matemáticas}, \text{“bueno”}) > 0)$ con “bueno” definido según la figura 4.2
- $X = \{\text{Nombre}\}$

Las tablas 4.25 y 4.26 detallan el cálculo de la consulta.

\mathcal{H}	NOMBRE	EDAD	MATEMÁTICAS	C_{MATE}	FÍSICA
\mathcal{B}	Tom	joven	16	0.3	[14,16]
	Joe	sobre 24	[14,16]	0.3	15
	Jack	[22,25]	UNKNOWN	1	casi bueno

Tabla 4.25: Cálculo de $\mathcal{S}_{\Pi}^{\gg \text{bueno}}$

\mathcal{H}	NOMBRE	C_{MATE}
\mathcal{B}	Tom	0.3
	Joe	0.3
	Jack	1

Tabla 4.26: Cálculo de $\mathcal{S}_{\Pi}^{\gg \text{bueno}}$

De lo mostrado en el ejemplo se puede comentar lo siguiente:

1. El modelo de Prade-Testemale calcula un grado global de satisfacción para cada una de las tuplas consultadas. Sobre dicho grado puede establecer un umbral mínimo de satisfacción. El valor del grado de satisfacción de una consulta sobre un atributo concreto se extiende a toda la tupla empleando el mínimo de los grados obtenidos en la consulta para cada atributo de la tupla. GEFRED almacena un grado para cada atributo implicado en la consulta. Esto le permite establecer umbrales de cumplimiento diferentes para cada uno de los atributos implicados en la misma. Podemos pues, actuar con mayor flexibilidad a la hora de seleccionar las tuplas que recuperamos mediante la consulta. Además, podemos calcular un grado global para la tupla recurriendo al concepto de cardinalidad introducido

en la definición del modelo. En este sentido, el grado almacenado en el atributo C_{MATE} puede globalizarse para toda la tupla puesto que, en este caso, representa el mínimo de los asumidos por cada uno de los atributos en cada tupla.

2. Con las limitaciones debidas a la concepción del elemento $\{e\}$ por parte del modelo de Prade-Testemale, GEFRED puede emplear el esquema de representación y manipulación de información imprecisa como un caso particular de los mecanismos introducidos con su definición.
3. GEFRED incorpora un esquema para llevar a cabo las operaciones mas coherente, consistente y flexible que el que incorpora el modelo de Prade-Testemale.

4.5.4 GEFRED vs. Modelo de Zemankova-Kandel

Este modelo, descrito en [Zemankova 84] y [Zemankova 85], se estructura sobre datos contruidos mediante distribuciones de posibilidad. Los aspectos diferenciadores más importantes residen en el empleo de dos medidas para evaluar, la compatibilidad de una condición, F , (representada por una distribución de posibilidad), con los valores del dominio. Estas medidas se denominan de “*posibilidad*”, definida de distinta forma que en el modelo de Prade-Testemale, y de “*certeza*” para la que propone un definición específica y cuyo estudio abordaremos en el apartado dedicado a las operaciones algebraicas. También recurre al empleo de modificadores que alteran la fisonomía de una distribución, C , a la que se aplican para modelar conceptos como: “*muy/mucho C*”, “*aproximadamente C*” y “*no C*”. Analizaremos estos y otros aspectos bajo la óptica de GEFRED en los próximos subapartados

4.5.4.1 Estructura de Datos

El modelo de Zemankova-Kandel se construye sobre dominios clasificados en los siguientes tipos:

1. Escalares discretos (ej. Color={rojo, blanco, azul})
2. Conjuntos discretos de números, finitos o infinitos (limitados por la capacidad del ordenador que lo implemente).
3. El intervalo unidad $[0, 1]$.

A partir de aquí permite operar con distribuciones de posibilidad construidas sobre dichos dominios. Además, en operaciones de consulta, incorpora tratamiento para los valores nulos, empleando para ello, el concepto de “*valor possibilístico esperado*”, (PEV). Este valor que está basado en el concepto de “*valor difuso esperado*”, (FEV) desarrollado por Kandel en [Kandel, Byatt] y [Kandel 79], sustituye la información “*ausente*” en un atributo de una tupla dada, por un valor calculado sobre los valores contenidos en resto de las tuplas por medio de una expresión que devuelve una distribución de posibilidad. Los detalles de como se calcula dicha expresión pueden obtenerse de la bibliografía referenciada. En cualquier caso, el valor que representa un “*nulo*” es una distribución de posibilidad. Por tanto, toda la información abordada por el modelo de Zemankova-Kandel puede ser representada mediante el empleo de GEFRED.

El concepto de relación adoptado viene expresado mediante la siguiente definición:

Definición 4.21 *Dados los conjuntos de dominios D_1, \dots, D_n , (no necesariamente distintos), de tipos escalares, tipos numéricos o intervalos unidad, R es una “relación possibilística” de esos conjuntos si es un conjunto de n -tuplas $(\Pi_{A_1}, \dots, \Pi_{A_n})$ donde Π_{A_i} ($i = 1, \dots, n$) es una distribución de posibilidad representando los posibles valores del atributo A_i definido sobre el dominio D_i .*

Además, exige que alguno de los atributos contengan información precisa para poder definir la clave primaria. Como vemos GEFRED incluye como caso particular la definición de “*relación possibilística*” introducida.

4.5.4.2 Operaciones Algebraicas

El modelo de Zemankova-Kandel construye las dos medidas de satisfacción de los requerimientos de una consulta, a partir de varios conceptos de relación definidos sobre el dominio subyacente y modelados mediante tres tipos de relaciones difusas:

- *Relaciones de Similitud*. Satisfaciendo las propiedades reflexiva, simétrica y transitividad max-min como está recogido en [Zadeh 71]. Las utiliza sobre dominios escalares.
- *Relaciones de “Proximidad”*. Estas son casos particulares de *relaciones de similitud* construidas sobre dominios numéricos. Con ellas modela mediante unas funciones el concepto de de proximidad difusa entre dos valores numéricos. Ofrece

dos de estas funciones para describir los conceptos *relativo* y *absoluto* de dicha proximidad.

- Enlaces (Link). Estas relaciones difusas no fuerzan simetría ni reflexividad pero si un cierto tipo de *transitividad reformada*.

Las tres categorías de relaciones difusas mostradas son compatibles con el concepto de *comparador extendido*. En efecto, son relaciones difusas y actúan sobre valores del dominio subyacente compatibles.

A partir de aquí contruye operadores relacionales difusos. Si X es un conjunto de objetos del sistema descrito, los tipos de condiciones soportadas son:

$$(i) \quad X \quad IS \ m \ F \quad (4.71)$$

$$(ii) \quad X.A \quad IS \ m \ \Theta \ valor \quad (4.72)$$

$$(iii) \quad X.A_1 \quad IS \ m \ \Theta \ X.A_2 \quad (4.73)$$

$$(iv) \quad X \quad IS \ m_1 \ AS \ F_1 \ AS \ m_2 \ F_2 \quad (4.74)$$

$$(v) \quad X \quad IS \ m_1 \ MORE \ m_2 \ F_1 \ THAN \ m_3 \ F_2 \quad (4.75)$$

$$(vi) \quad X \quad IS \ m_1 \ LESS \ m_2 \ F_1 \ THAN \ m_3 \ F_2 \quad (4.76)$$

donde m_i es un modificador, (ej. “Aproximadamente”, “no mucho”, etc.), F_i es un conjunto difuso definido sobre atributos del objeto X , y $X.A_i$ son atributos clasificando objetos X cuyos valores tienen que ser del tipo especificado para $X.A$. El símbolo Θ es un operador relacional distinto de IS.

Las condiciones 4.72 y 4.73 se construyen empleando las *relaciones de similitud* y de *proximidad* sobre el dominio subyacente según sea este escalar o numérico, respectivamente. Los operadores envueltos en las condiciones 4.74, 4.75 y 4.76 se construyen sobre relaciones de proximidad definidas en el dominio subyacente. En concreto para los *operadores relacionales difusos*: ES, ES MAYOR QUE y ES MENOR QUE, parte de las *relaciones difusas de comparación* a nivel de dominio subyacente definidas como sigue:

Definición 4.22 Sean $s(x, y)$ y $p(x, y)$ relaciones de similitud y de proximidad definidas para elementos de dominios escalares y numéricos, respectivamente. Entonces las relaciones difusas de comparación empleadas en los operadores *ES* (\simeq), *ES MAYOR QUE* ($\tilde{>}$) y *ES MENOR QUE* ($\tilde{<}$) están definidas por:

$$\mu_{\simeq}(x, y) = \begin{cases} s(x, y) & \text{con } x, y \in D \text{ escalar} \\ p(x, y) & \text{con } x, y \in D \text{ numérico} \end{cases} \quad (4.77)$$

$$\mu_{\tilde{>}}(x, y) = \begin{cases} 1 - 0.5 \cdot p(x, y) & x \geq y \\ 0.5 \cdot p(x, y) & x < y \end{cases} \quad (4.78)$$

$$\mu_{\tilde{<}}(x, y) = \begin{cases} 1 - 0.5 \cdot p(x, y) & x \leq y \\ 0.5 \cdot p(x, y) & x > y \end{cases} \quad (4.79)$$

Con los operadores así definidos establece dos criterios para clasificar las tuplas que satisfacen una consulta. Dichos criterios son formalizados mediante la definición de las medidas de “*posibilidad*” y de “*certeza*”. El modelo no proporciona muchos detalles en cuanto a la forma en que se construyen las condiciones a partir de estos operadores. Todo parece apuntar a que las condiciones se construyen de igual forma que en el modelo de Prade-Testemale. Así, por ejemplo, si f es un valor difuso sobre un dominio escalar D , cuya función de pertenencia es $\mu_f(x)$, y $s(x, y)$ es la relación de similitud definida sobre dicho dominio, la condición “*es f*” se traslada mediante la sintaxis empleada en dicho modelo a “*es \simeq f*” con \simeq modelado según la expresión 4.77. La función de pertenencia que representa dicha condición vendría dada por:

$$\mu_{\simeq f}(y) = \max_{x \in D} \{ \mu_f(x) \wedge \mu_{\simeq}(x, y) \} \quad (4.80)$$

A partir de aquí introduce las medidas de “*posibilidad*” y de “*certeza*” en que un valor X satisface la condición “*ser \simeq f*” como:

$$p_X(\simeq f) = \sup_{y \in D} \{ \mu_{\simeq f}(y) \cdot \Pi_X(y) \} \quad (4.81)$$

y

$$c_X(\simeq f) = \max\{0, \inf_{y \in D} \{ \mu_{\simeq f}(y) \cdot \Pi_X(y) \} > 0\} \quad (4.82)$$

respectivamente.

Ambas medidas pueden ser elaboradas mediante el empleo de los recursos de GEFRED. En efecto, puesto que se construyen a partir de relaciones difusas sobre el dominio subyacente, basta comprobar que valores precisos de f y de X , y que para un operador Θ definido según la def. 4.22 se satisface en general:

$$p_X(\Theta f) = \Theta(X, f) \quad (4.83)$$

y

$$c_X(\Theta f) = \Theta(X, f) \quad (4.84)$$

Para valores precisos la condición inducida por Θ sobre f viene dada por:

$$\mu_{\Theta f}(y) = \{\Theta(f, y); y \in D\} \quad (4.85)$$

con lo que tenemos las siguientes medidas de “posibilidad” y de “certeza”:

$$p_X(\Theta f) = \sup_{y \in D} \{\Theta(f, y) \cdot 1/x\} \quad (4.86)$$

$$c_X(\Theta f) = \max\{0, \inf_{y \in D} \{\Theta(f, y) \cdot 1/x\} > 0\} \quad (4.87)$$

La expresión 4.86 solo es distinta de 0 para $y = x$ con lo que se cumple 4.83. En la expresión 4.87 x es el único valor del dominio para el que el ínfimo del producto puede ser distinto de 0. Para ese valor se obtiene el resultado 4.84. Por tanto la forma de construir las operaciones de comparación puede modelarse a través del *comparador difuso generalizado*.

4.5.4.3 Resolución de Consultas

Para comparar la forma de operar en el marco del modelo de Zemankova-Kandel con la de GEFRED, vamos a emplear un ejemplo tomado de [Zemankova 84]. La tabla 4.27 muestra una relación en la que se recogen diferentes posibilidades de representación del modelo.

Los atributos Nombre y Edad almacenan valores precisos, conteniendo este último un valor nulo (representado mediante –). Los valores asumidos por el atributo Residencia permiten listas de valores. La lista *Seattle, Portland* representa que Tom vive

Nombre	Edad	Residencia	Color_pelo	Buen_Asp.
Tom	25	Seattle, Portland	.8/castaño + .6/moreno	.6
Bob	30	Tallahassee	.3/rubio + .7/pelirrojo	.4
Al	35	Boston, New York	1/castaño	.9
Ron	–	Miami	.4/castaño + .9/pelirrojo	.2

Tabla 4.27: Relación Persona

en *Seattle* o en *Portland*. Esto puede ser modelado en GEFRED por la distribución de posibilidad ($1/Seattle, 1/Portland$). Para representar los posibles matices del color de pelo construye distribuciones de posibilidad sobre un dominio escalar, así el color de pelo de Tom es moreno tirando a castaño, lo cual se representa como $(.8/castaño + .6/moreno)$. Sobre dicho dominio tiene definida la relación de similitud mostrada en la tabla 4.28. GEFRED permite también construir este tipo de datos sobre escalares. El último atributo recoge el grado en que cada individuo concuerda con el concepto de *buén aspecto*.

s	RU	PR	CA	MO
RU	1	.6	.4	0
PR	.6	1	.5	.1
CA	.4	.5	1	.8
MO	0	.1	.8	1

Tabla 4.28: Relación de Similitud sobre el dominio “color de pelo”

Vamos a resolver la siguiente cuestión sobre la tabla 4.27:

Consulta 4.5 “Encontrar las personas y el grado en que satisfacen que su pelo sea *castaño*”

El ejemplo estudiado solo detalla el cálculo de las medidas en que se satisface tal condición para la tupla relativa a Bob. Obteniendo que Bob satisface dicha condición con un valor para la medida de *posibilidad* igual a .35, siendo .12 el valor obtenido para la medida de *certeza*.

A continuación mostramos el resultado que se obtiene empleando los recursos de GEFRED para resolver dicha consulta en los mismos términos que el modelo de Zemankova-Kandel.

Para las tuplas *posibles* la consulta se formula:

$$\mathcal{P}_G(\mathcal{S}_{\tilde{P}}^{\tilde{casta\~{n}o}} ; X) \quad (4.88)$$

donde

- $\mathcal{S}_{\tilde{P}}^{\tilde{casta\~{n}o}} = \mathcal{S}_G(Persona; \Theta_{\tilde{P}}^{\tilde{casta\~{n}o}}(Color_pelo, "casta\~{n}o") \geq 0)$
- $\Theta_{\tilde{P}}^{\tilde{casta\~{n}o}}(Color_pelo, "casta\~{n}o") = \sup_{y \in D_{Color_pelo}} \{\mu_{\tilde{casta\~{n}o}}(y) \cdot \Pi_{Color_pelo}(X)(y)\}$ con X recorriendo cada tupla de la relación Persona.
- $X = \{Nombre, C_{Color_pelo}\}$

El resultado final se muestra en la tabla 4.29.

\mathcal{H}	NOMBRE	C_{Color_pelo}
\mathcal{B}	Tom	0.8
	Bob	0.35
	Al	1
	Ron	0.45

Tabla 4.29: Cálculo de $\mathcal{P}_G(\mathcal{S}_{\tilde{P}}^{\tilde{casta\~{n}o}} ; X)$

Para las tuplas *ciertas* la consulta se formula:

$$\mathcal{P}_G(\mathcal{S}_{\tilde{C}}^{\tilde{casta\~{n}o}} ; X) \quad (4.89)$$

donde

- $\mathcal{S}_{\tilde{C}}^{\tilde{casta\~{n}o}} = \mathcal{S}_G(Persona; \Theta_{\tilde{C}}^{\tilde{casta\~{n}o}}(Color_pelo, "casta\~{n}o") \geq 0)$
- $\Theta_{\tilde{C}}^{\tilde{casta\~{n}o}}(Color_pelo, "casta\~{n}o") = \max\{0, \inf_{y \in D_{Color_pelo}} \{\mu_{\tilde{casta\~{n}o}}(y) \cdot \Pi_{Color_pelo}(X)(y)\} > 0\}$, con X recorriendo cada tupla de la relación Persona.
- $X = \{Nombre, C_{Color_pelo}\}$

El resultado final se muestra en la tabla 4.30

\mathcal{H}	NOMBRE	C_{Color_pelo}
\mathcal{B}	Tom	0.48
	Bob	0.12
	Al	1
	Ron	0.4

Tabla 4.30: Cálculo de $\mathcal{P}_G(\mathcal{S}_{\tilde{C}}^{\tilde{casta\tilde{n}o}}; X)$

4.6 Conclusiones

El modelo que hemos presentado en este capítulo aporta las siguientes características frente a otras propuestas recogidas en la literatura:

- Proporciona tratamiento difuso a la información: tanto a los datos difusos, como a las relaciones que existen entre ellos.
- Los tipos de datos con los que opera es bastante extenso en comparación con otras propuestas. Es decir, la variedad de tipos de datos difusos que puede manejar es más amplia que los modelos precedentes.
- Organiza en forma consistente la información difusa. Las “relaciones difusas generalizadas” estructuran, tanto la información de partida, como la que resulta de las operaciones realizadas sobre la misma. Además, la relación que resulta de una operación de manipulación refleja en su estructura, mediante los grados de compatibilidad, el grado en que cada atributo involucrado en dicha operación satisface los requisitos establecidos en la misma.
- Dentro del conjunto de metainformación que el experto o usuario debe proporcionar a la Base de Datos se encuentra, además de la definición que adopta para las diferentes etiquetas, la del modelo de comparador difuso que va asociar a cada una de las comparaciones a establecer sobre cada dominio definido.
- El modelo presenta gran flexibilidad para el tratamiento y evaluación de la información difusa, basado en la libertad a la hora de seleccionar un comparador

difuso generalizado u otro y en que se puede controlar el grado en que se satisfacen las condiciones individuales de una consulta.

- Por último una gran parte de los modelos precedentes se pueden considerar, bajo ciertas condiciones, como casos particulares de GEFRED. Por tanto, este modelo se puede emplear para representarlos.

Capítulo 5

Un Sistema Gestor de BD Relacionales Difusas Basado en GEFRED

5.1 Introducción

En el capítulo 3 hemos analizado las propuestas teóricas presentes en la literatura para extender el "Modelo Relacional" a fin de que pueda incorporar tratamiento para información de naturaleza difusa. En el capítulo 4 hemos propuesto y cotejado un modelo, GEFRED, con el que se buscaba una síntesis entre los modelos precedentes. En la segunda parte de esta memoria nos proponemos presentar los elementos básicos de la implementación de un Gestor de Bases de Datos Relacionales Difusas, SBDRD, basado en los postulados teóricos de GEFRED.

Si con la formulación de GEFRED se perseguía aglutinar en un entorno común los aspectos más interesantes de las diferentes aproximaciones aparecidas sobre el tema, con el sistema que vamos a proponer particularizaremos alguno de los extremos de GEFRED a fin de materializar un SBDRD operativo.

5.2 FIRST: Un Interface Difuso para Sistemas Relacionales

A la hora de abordar el problema de realizar un prototipo de SBDRD nos planteamos la consecución de una serie de objetivos, a saber:

- Realizar la implementación basándonos en GEFRED.
- Procurar la operatividad del mismo a fin de que pueda servir para evaluar la bondad de GEFRED.
- Siguiendo la filosofía de GEFRED, se hará uso de los elementos del "Modelo Relacional Clásico" para la implementación del SBDRD.
- Respetar en el desarrollo final las operaciones primitivas del Modelo Relacional.
- Buscar la simplicidad en las soluciones adoptadas.
- Primar la eficiencia en la ejecución de las operaciones en el SBDRD, sobre cualquier otra consideración.
- En orden a conseguir esto, adoptar para los ítem difusos, los criterios de representación más asequibles y generales.
- Facilitar el desarrollo del sistema sobre un RDBMS comercial. En el apéndice de esta memoria recogeremos los aspectos de un prototipo desarrollado sobre un RDBMS particular (Oracle).
- Dotar de portabilidad a la implementación. Respetar una organización estructurada en módulos. Esto permitirá resolver cada petición "difusa" al sistema mediante el empleo de los módulos que sean precisos para transformar dicha petición en las oportunas llamadas clásicas al RDBMS anfitrión.

Para alcanzar los objetivos propuestos hemos adoptado una serie de criterios en la organización del SBDRD, tanto en la representación del conocimiento impreciso, como en la implementación del mismo sobre cada módulo del Gestor. Con dichos criterios estableceremos los límites de una implementación particular basada en GEFRED denominada **FIRST**, acrónimo de **F**uzzy **I**nterface for **R**elational **Sis**Tems, (Interface

Difuso para Sistemas Relacionales). FIRST es un módulo que permite extender la capacidad de un RDBMS “clásico” a fin de que pueda representar y manipular información “imprecisa”. Para la construcción de FIRST nos valdremos unicamente de los postulados teóricos formulados en GEFRED, de los recursos proporcionados por el modelo relacional “clásico” para poder representar este tipo de información y de los procedimientos contenidos en éste y en sus implementaciones para manipular dicha información.

Con esta implementación, que no pretende agotar las posibilidades de GEFRED, perseguimos demostrar lo siguiente:

- Es factible construir un sistema real que implemente los postulados de GEFRED.
- Puesto que GEFRED extiende el modelo relacional “clásico”, es posible que un sistema basado en él respete la representación y tratamiento para datos considerados en el esquema relacional.
- Bajo ciertos criterios de representación, compatibles con la definición de GEFRED, es posible construir un sistema que unicamente emplee los recursos proporcionados por un RDBMS “clásico”.
- En definitiva, se puede partir de un RDBMS “comercial” para construir un SBDRD. Por supuesto que también es posible construir un sistema realizado “ad hoc” para GEFRED.

5.2.1 Esquema General de FIRST

La figura 5.1 muestra el esquema general de FIRST. Como la idea de partida es la de construirlo sobre un gestor de bases de datos convencional, todos los desarrollos a realizar toman a dicho gestor como el elemento principal al que van dirigidas todas las peticiones.

A continuación haremos un breve recorrido por los diferentes módulos que constituyen el esquema propuesto:

- **RDBMS.** Como hemos dicho, todas las operaciones que hayamos concebido para la extensión difusa que representa nuestra implementación, se traducirán a peticiones al RDBMS anfitrión. Estas peticiones se realizarán empleando el

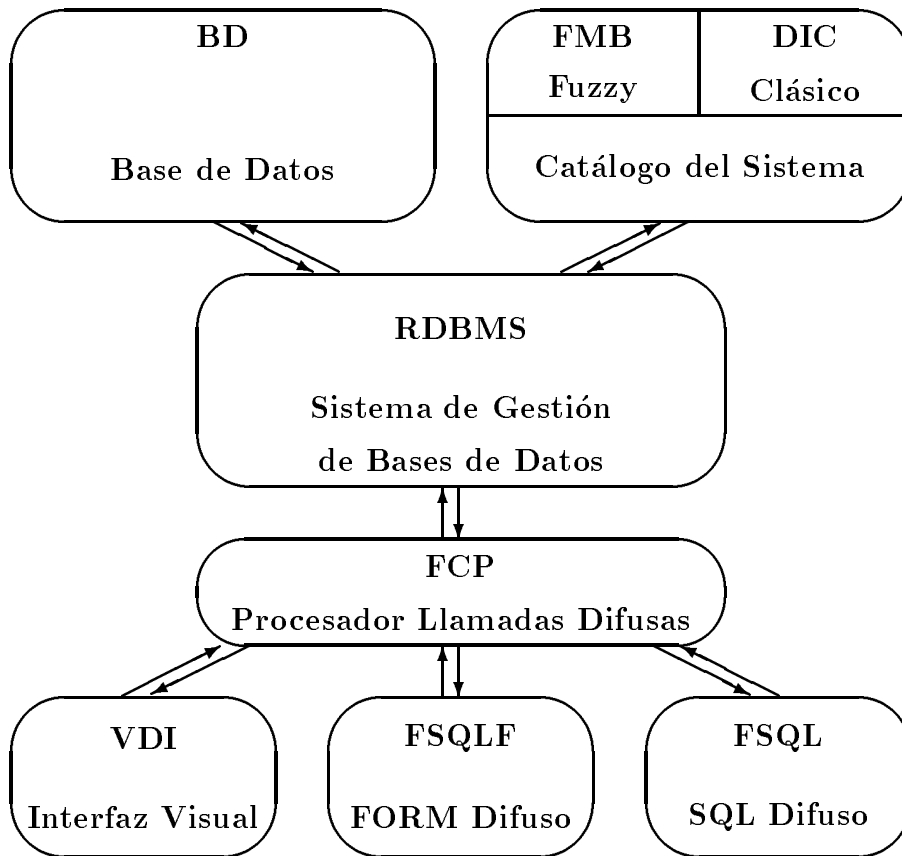


Figura 5.1: Esquema General de FIRST

leguaje de soporte del mismo. Generalmente, y este es nuestro caso, todas las peticiones al sistema se resuelven mediante sentencias SQL clásicas. Al final, todo el procesamiento, "difuso" o no, se traducirá a una serie de peticiones al sistema en lenguaje SQL.

- **BD.** La base de datos aglutina toda la información permanente, "difusa" o no, mediante las tablas de la base de datos. La forma en que se representan los datos en dichas tablas dependerá de la naturaleza y del tipo de los mismos. Nosotros usaremos los recursos disponibles en el sistema anfitrión para representar la información "difusa" que queramos en FIRST. La forma en que haremos esto se

verá en los próximos apartados.

- **FMB.** El diccionario o catálogo de un RDBMS representa aquella parte del sistema que almacena información sobre los datos recogidos en la Base de Datos, así como otro tipo de informaciones: usuarios, permisos, accesos, datos de control, etc. En nuestro caso extenderemos esta parte del sistema a fin de que pueda recoger aquella información necesaria relacionada con la naturaleza "imprecisa" de la nueva colección de datos a procesar. Denominaremos esta extensión como Base de Metaconocimiento Difuso, (FMB), y se organizará, como veremos más adelante, siguiendo la filosofía imperante en el catálogo del RDBMS anfitrión.
- **FCP.** Pretendemos con nuestra implementación, transformar el conjunto de operaciones "difusas" a soportar, en un conjunto de peticiones al RDBMS anfitrión incluido en el conjunto de instrucciones que éste recoge. La misión del FCP consiste en traducir las operaciones "difusas" formuladas sobre FIRST en peticiones ordinarias al sistema anfitrión. Esta traducción tiene en cuenta la operación "difusa" requerida, los ítem difusos implicados en dicha operación, la representación de los mismos en la base de datos y la información que sobre estos posea la FMB. Por tanto, el FCP lleva implementada la forma de proceder para resolver en términos clásicos las peticiones difusas.
- **FSQL** Se contempla como una extensión del lenguaje de consulta estructurado SQL que proporciona una sintáxis adicional y modificada para articular las peticiones de naturaleza "difusa" a FIRST. El módulo que se encarga de traducir esa sintáxis a operaciones sobre el RDBMS se denomina procesador de FSQL y hará uso para ello de las rutinas del FCP.

Hasta aquí lo que podríamos considerar como módulos básicos de nuestra implementación.

Con la introducción de un lenguaje de consulta y manipulación tal como FSQL, tenemos resuelto en primera instancia el problema de la interfaz de FIRST con el usuario. A partir de aquí, se puede pensar en la realización de otros tipos de interfaces o de lenguajes de consulta basados en entornos gráficos, lenguajes de 4ª Generación, etc.

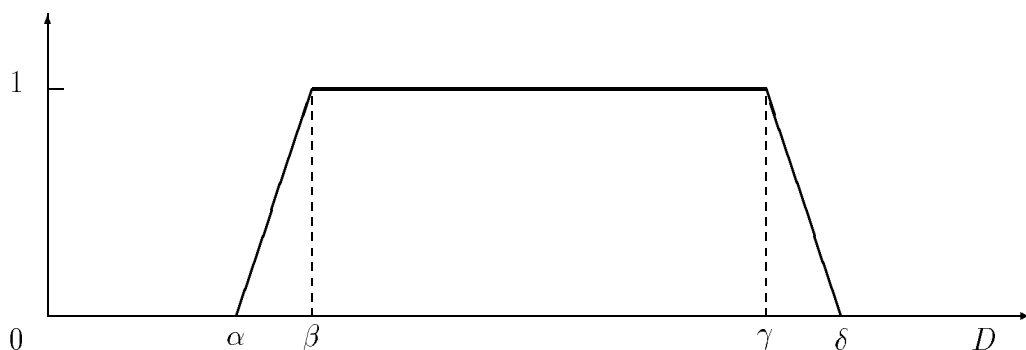


Figura 5.2: Distribución Trapezoidal

5.2.2 Representación del Conocimiento Impreciso

Los diferentes elementos que forman parte del tratamiento impreciso pueden recibir diferentes representaciones. Así pues, p. e., una distribución de posibilidad normalizada puede venir representada por diferentes tipos de funciones, nosotros emplearemos para la misma una representación trapezoidal. Lo mismo se puede decir para la forma en que modelamos las operaciones relacionales difusas, así como para el resto de los ítem "difusos" que va a utilizar nuestro sistema. Pudiera parecer que el criterio de representación expuesto supone una fuerte restricción, sin embargo, esto no es así dada la naturaleza "imprecisa" de la información con la que tratamos. No parece muy acertado modelar unos datos, que en sí son imprecisos, con una representación extremadamente precisa dada por funciones no lineales.

En este apartado vamos a mostrar los criterios de representación empleados en nuestra implementación. Estos criterios no son exclusivos de una implementación concreta, como FIRST, sino que representan la base sobre la que construir la implementación de los mismos de acuerdo con el esquema diseñado para la realización de un SBDRD. Podríamos decir, por tanto, que constituyen un paso intermedio a cubrir entre la formulación de un modelo de Bases de Datos Relacionales Difusas y la implementación efectiva de un sistema basado en él, en [Medina et al, 93(a)] se encuentran algunos de los aspectos relacionados con la representación e implementación de un SBDRD.

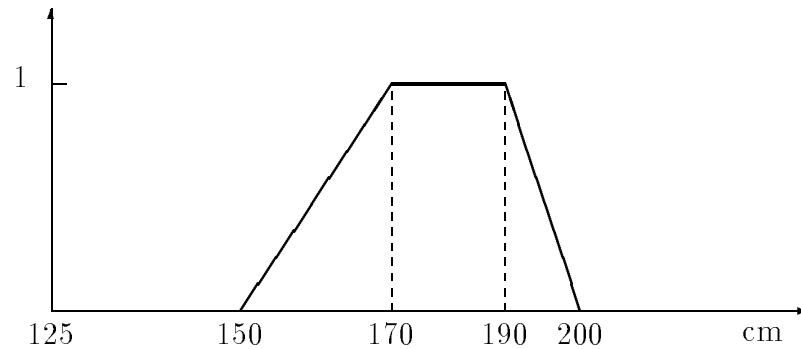


Figura 5.3: Etiqueta Lingüística "alto"

A continuación describiremos los criterios adoptados para la representación de conocimiento impreciso: Datos difusos, comparadores difusos, cualificadores de la consulta y cuantificadores de la misma.

5.2.2.1 Representación de los Datos

Para los diferentes tipos de datos que constituían la definición de *dominio difuso generalizado*, recogidos en la tabla 4.1, acordamos los siguientes criterios de representación:

- **Datos Precisos**

Se empleará la representación que proporcione el RDBMS anfitrión. Es decir, se adoptarán los formatos para cadenas alfanuméricas, valores numéricos, fechas, horas, etc. soportados por el sistema sobre el que se construya FIRST.

- **Datos Imprecisos**

Los datos de naturaleza imprecisa soportados por GEFRED pueden ser clasificados en dos grupos: sobre referencial ordenado y sobre referencial discreto. FIRST proporciona una representación diferente para cada uno de ellos.

- *DATOS IMPRECISOS SOBRE REFERENCIAL ORDENADO*

Este grupo de datos contiene distribuciones de posibilidad sobre dominios continuos o discretos sobre los que existe una relación de orden. A este grupo

pertenecen el tipo 6 de la tabla 4.1. Cada dato de este tipo tiene asociada una función de pertenencia. Por cuestiones de simplicidad de representación y de eficiencia en el cálculo, adoptaremos las siguientes representaciones para este tipo de datos:

– *Distribución de Posibilidad Trapezoidal.*

Esta representación determina la función de pertenencia asociada al dato mediante el uso de cuatro parámetros ($\alpha, \beta, \gamma, \delta$) ver figura 5.2. Utilizamos funciones de pertenencia normalizadas, aquellas que poseen un núcleo no vacío.

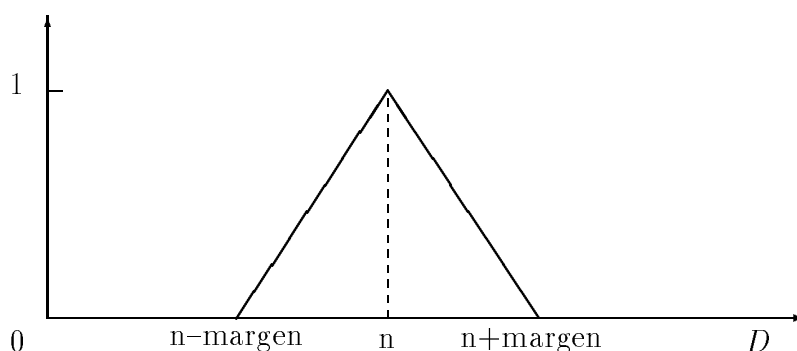


Figura 5.4: "Aproximadamente n"

– *Etiqueta Lingüística.*

Los datos expresados mediante una etiqueta lingüística hacen referencia a un concepto impreciso, (a veces subjetivo), que lleva asociado una distribución de posibilidad. Por ejemplo, la etiqueta lingüística "alto" puede llevar asociada la distribución de posibilidad en representación trapezoidal que muestra la figura 5.3.

– *Valores Aproximados*

Dado un valor, n , perteneciente al dominio subyacente, podemos dar una representación del concepto impreciso "aproximadamente n " mediante un

valor que llamaremos "*margen*" a partir del cual podemos construir su función de pertenencia como vemos en la figura 5.4. Nuevamente empleamos funciones de pertenencia normalizadas.

– *Conjuntos Intervalares.*

Adoptan la representación mostrada en la figura 5.5. Se opera con ellos de forma similar a como se hace con la distribuciones de posibilidad trapezoidales.

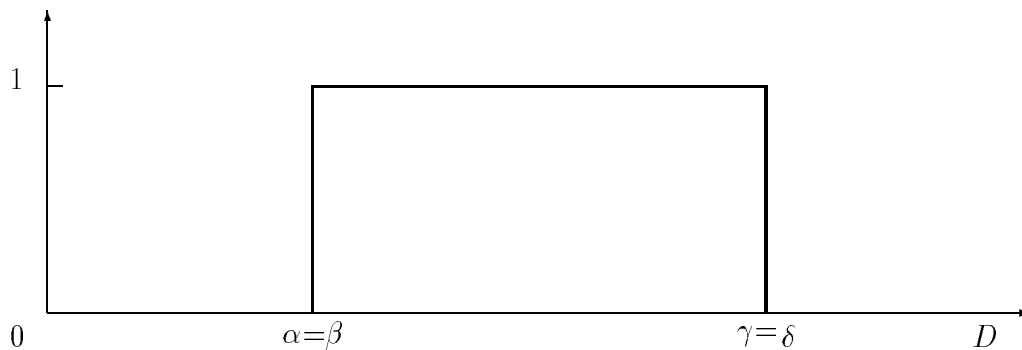


Figura 5.5: Distribución Intervalar

• *DATOS CON ANALOGÍA SOBRE DOMINIO DISCRETO*

Este grupo de datos está construido sobre dominios discretos en los que se encuentran definidas "*relaciones de semejanza*" entre los valores que lo constituyen. En el subapartado 5.2.2.2 introduciremos dicho concepto en el contexto que nos ocupa. Para este tipo de datos tendremos que proporcionar almacenamiento, para la representación de los mismos, así como para las "*relaciones de semejanza*" definidas sobre los valores del dominio. Los diferentes tipos que podemos representar dentro de este grupo son:

– *Escalares Simples.*

Estos datos se representan utilizando el esquema de representación del RDB-MS anfitrión. Únicamente habremos de añadir al sistema información para que opere con las "relaciones de semejanza" definidas sobre el dominio que le subyace.

– *Distribución de Posibilidad sobre Discretos*

A un dato impreciso de este tipo le asociamos una representación en la que se describen los valores del dominio de discurso que la componen con los respectivos valores de posibilidad para cada uno de ellos, $((p_1, d_1), \dots, (p_n, d_n))$.

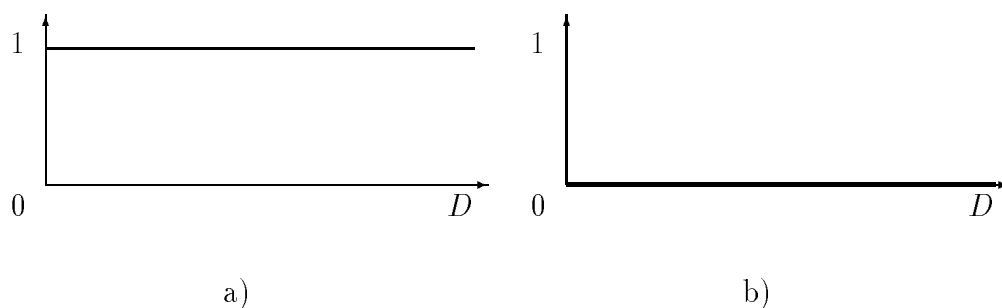


Figura 5.6: Tipos UNKNOWN y UNDEFINED

• *DATOS TIPO UNKNOWN*

Un dato de este tipo refleja el desconocimiento con respecto al valor que toma un atributo. Sabemos, sin embargo, que el atributo puede tomar algún valor del dominio de discurso. Esto implica que es posible que tome cualquiera de ellos, por tanto representaremos el tipo UNKNOWN mediante la distribución de posibilidad, $\{1/u \mid u \in U\}$ donde U es el dominio subyacente. La figura 5.6 a) muestra gráficamente esta distribución de posibilidad.

• *DATOS TIPO UNDEFINED*

Cuando un atributo toma el valor UNDEFINED refleja el hecho de que ninguno de los valores de dominio sobre el que está definido es aplicable. Esto se puede

entender como que ninguno de los valores de dominio es posible, por lo que lo representaremos mediante la distribución de posibilidad, $\{0/u \mid u \in U\}$ donde U es el dominio subyacente. La distribución de posibilidad se muestra en la figura 5.6 b).

- *DATOS TIPO NULL*

Sobre un atributo tenemos un valor NULL cuando no aportamos información, bien porque no la conocemos (UNKNOWN) o porque no es aplicable (UNDEFINED). Mediante el conjunto $\{1/UNKNOWN, 1/UNDEFINED\}$ podemos modelar este tipo de dato.

5.2.2.2 Comparadores Difusos Generalizados

En la literatura pueden encontrarse diversos métodos de comparación de números difusos, los cuales pueden clasificarse en dos categorías: los que emplean una función del conjunto de números difusos a un conjunto ordenado y los que utilizan relaciones difusas para el proceso de comparación. Al primer tipo pertenecen las propuestas recogidas en [Adamo 80], [Yager 78] y [Yager 81]. Sobre el segundo tipo se encuentran diferentes aproximaciones en [Baldwin 79], [Bass, Kwakernaak, 77], [Delgado, Verdegay, Vila 88] y [Dubois, Prade, 83]. Nosotros basaremos la implementación que estamos describiendo en comparadores construidos en torno al segundo tipo. Concretamente, partiremos del esquema formulado en [Prade, Testemale, 84], si bien su representación ha sido simplificada y adaptada a las peculiaridades tanto de GEFRED como de FIRST.

Como vimos en el capítulo dedicado al estudio de GEFRED, la definición de *comparador difuso generalizado* permitía modelar una amplia variedad de modalidades de comparación. Sin embargo, de acuerdo con la idea de simplicidad que inspira a FIRST, seleccionaremos los comparadores más representativos a efectos de implementarlos en el sistema. A continuación se muestra la representación que adoptamos para los diferentes comparadores empleados en FIRST:

- Los datos con analogía sobre dominios discretos tendrán modelada esa analogía mediante el empleo de *"relaciones de semejanza"*.

Una *"relación de semejanza"* en el contexto de GEFRED es un *comparador extendido* que viene representado por una relación difusa binaria que satisface las propiedades reflexiva y simétrica. En nuestro caso emplearemos la *"relación de*

semejanza” unicamente sobre universos de discurso finitos. Por tanto podemos modelar dicha relación mediante una representación matricial en la forma $[R]$ cuyo término genérico $[R]_{ij}$ es $\mu_R(x_i, y_j) = r_{ij}$, $i = 1, n$, $j = 1, m$, donde $| X | = n$ y $| Y | = m$.

- *”Igual a”*. Este operador modela el concepto de igualdad para datos de naturaleza imprecisa. Formalmente viene expresado mediante la función de pertenencia dada por:

$$\mu_{igual_a}(\tilde{d}, \tilde{d}') = \sup_{(d, d') \in D \times D} \min\{p(d, d'), \pi_{\tilde{d}}(d), \pi_{\tilde{d}'}(d')\} \tag{5.1}$$

donde $p(d, d')$ es una *”relación de semejanza”* y, $\pi_{\tilde{d}}(d)$ y $\pi_{\tilde{d}'}(d')$ son las respectivas distribuciones de posibilidad definidas sobre el dominio de discurso D .

- Para datos imprecisos definidos sobre referencial ordenado, $p(d, d') = \delta(d, d')$, siendo δ una delta de Dirac. Debido a la representación que damos a este tipo de datos, se puede obtener de forma geométrica el resultado de la operación *”igual a”*. En la figura 5.7 a) se muestra graficamente como se obtiene el grado en que dos distribuciones de posibilidad son *”iguales”*.
- Para datos con analogía sobre referencial discreto, $p(d, d')$ es la representación matricial de la *”relación de semejanza”* que existe definida sobre el dominio de discurso D .

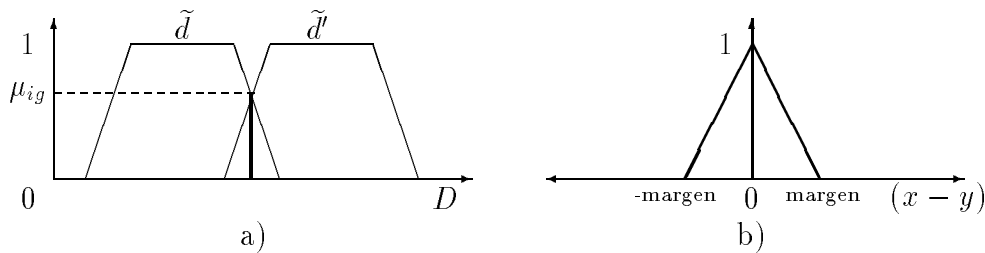


Figura 5.7: Operadores *”igual”* y *”aproximadamente igual”*

- *”Aproximadamente igual”*. Este operador proporciona el grado en que dos valores numéricos precisos son aproximadamente iguales. Lo calcula según la siguiente expresión:

$$\mu_{aprox_igual}(x, y) = \begin{cases} 0 & \text{si } |x - y| > \text{margen} \\ 1 - |x - y| / \text{margen} & \text{si } |x - y| \leq \text{margen} \end{cases} \quad (5.2)$$

La fig. 5.7 b) muestra en forma gráfica su forma de operación. El parámetro *margen* ajusta el operador al dominio en que se encuentra definido.

- "*Mayor o igual*". Está definido sobre dominios ordenados. La función de pertenencia de este operador viene dada por la relación difusa:

$$\mu_{\geq}(A, B) = \sup_{(x,y) \in X \times Y} \min\{\geq(x, y), \pi_A(x), \pi_B(y)\} \quad (5.3)$$

donde A y B son datos imprecisos sobre referencial ordenado o datos numéricos precisos, $\pi_A(x)$, $\pi_B(y)$ sus respectivas representaciones posibilísticas y \geq es el operador "*Mayor o igual*" clásico dado por:

$$\geq(x, y) = \begin{cases} 0 & \text{si } x < y \\ 1 & \text{si } x \geq y \end{cases} \quad (5.4)$$

Este operador puede resolver las siguientes comparaciones:

- Grado en que un número preciso es "*mayor o igual*" a una distribución de posibilidad.
- Grado en que una distribución de posibilidad es "*mayor o igual*" a un número preciso.
- Grado en que una distribución de posibilidad es "*mayor o igual*" a otra distribución de posibilidad.
- "*Menor o igual*". Está definido sobre dominios ordenados. La función de pertenencia de este operador viene dada por la relación difusa:

$$\mu_{\leq}(A, B) = \sup_{(x,y) \in X \times Y} \min\{\leq(x, y), \pi_A(x), \pi_B(y)\} \quad (5.5)$$

donde A y B son datos imprecisos sobre referencial ordenado o datos numéricos precisos, $\pi_A(x)$, $\pi_B(y)$ sus respectivas representaciones posibilísticas y \leq es el operador "*Mayor o igual*" clásico dado por:

$$\leq(x, y) = \begin{cases} 0 & \text{si } x > y \\ 1 & \text{si } x \leq y \end{cases} \quad (5.6)$$

Este operador puede resolver comparaciones sobre los mismos tipos que el operador "*Mayor o igual*"

- "*Mayor que*". Definimos este operador a partir del operador "*Menor o igual*". Para ello calculamos el complemento de dicho operador:

$$\mu_{>}(A, B) = 1 - \mu_{\leq}(A, B) \quad (5.7)$$

- "*Menor que*". Se define como el complemento del operador "*Mayor o igual*":

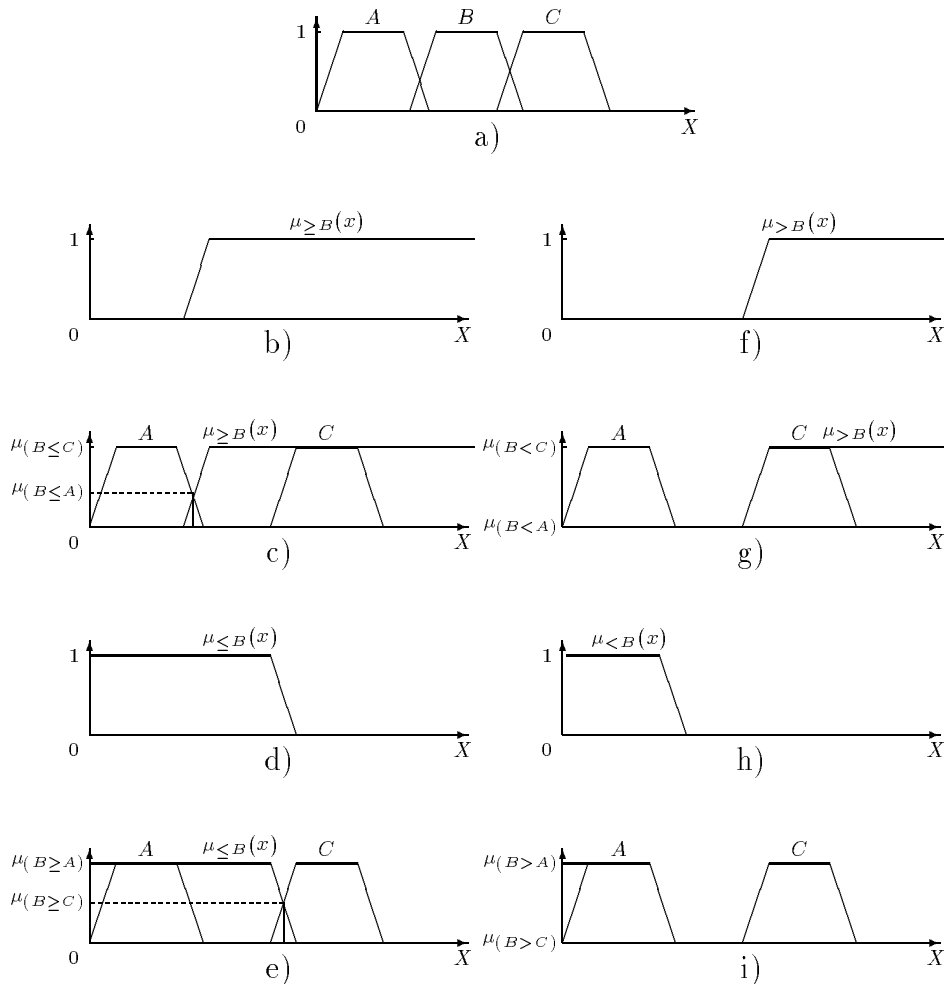
$$\mu_{<}(A, B) = 1 - \mu_{\geq}(A, B) \quad (5.8)$$

La figura 5.8 muestra gráficamente el funcionamiento de los operadores relacionales difusos descritos anteriormente.

5.2.2.3 Cualificadores del Umbral de Consulta

Cuando planteamos una consulta en una Base de Datos Imprecisa se establecen una serie de condiciones a cumplir por las tuplas que la satisfagan. Dada la naturaleza imprecisa de los operadores y de los datos sobre los que operamos, existe un grado de cumplimiento para cada condición involucrada en una consulta. Este grado de cumplimiento se halla comprendido entre 0 y 1. Mediante el empleo de un umbral mínimo para el grado de cumplimiento podemos ejercer algún control sobre la precisión con que se satisfacen cada una de las condiciones de la consulta. Si establecemos un umbral de cumplimiento 1 para una condición envuelta en una consulta, eliminaremos aquellas tuplas que no igualen o superen el umbral para esa condición.

Denominaremos "*cualificación de una condición*" a la acción de establecer un umbral de cumplimiento para una condición atómica (simple) de una consulta. A dicho umbral lo denominaremos "*cualificador*". Un "*cualificador*", que es un valor entre 0 y 1, puede ser representado mediante un valor lingüístico, por ejemp., si exigimos que el grado con el que se satisfaga una condición simple sea "alto", estaremos indicando que aceptaremos tuplas que la satisfagan con un grado igual o superior 0.8. En definitiva,



La figura a) muestra las tres distribuciones de posibilidad sobre las que operaremos, en la fig. b) se muestra la función de pertenencia del operador \geq aplicado a B , la fig. c) muestra los grados en que A y C son "mayores o iguales" a B , la fig. d) muestra la función de pertenencia de \leq aplicado a B , la fig. e) los grados en que A y C son "menores o iguales" a B . En las figuras f) y h) mostramos, respectivamente, las funciones de pertenencia de los operadores $>$ y $<$ construidos sobre B , en las figuras g) e i) se muestra como se calculan los grados en que A y C son "mayores" y "menores", respectivamente, a B .

Figura 5.8: Operadores Relacionales Difusos

podemos asociar valores lingüísticos a los *cualificadores*. El valor umbral que asociamos a cada valor lingüístico debe estar almacenado en el sistema y tiene, al igual que las etiquetas lingüísticas, una connotación subjetiva.

5.2.2.4 Cuantificadores difusos de la Consulta

Mediante el concepto de cuantificador es posible construir consultas que requieren efectuar un cálculo sobre el número de tuplas que satisfacen una determinada condición. Algunos ejemplos de consultas basadas en este concepto se muestran a continuación:

Consulta 5.1 *“Dime cuantos alumnos ha superado la Física en Junio”*

Consulta 5.2 *“Dime cuantos alumnos ha suspendido las Matemáticas y han suspendido la Física en Junio”*

Consulta 5.3 *“Dame los alumnos que han superado todas las asignaturas en Junio”*

Consulta 5.4 *“Dame los alumnos que han superado alguna asignatura en Junio”*

El Modelo Relacional Clásico permite realizar dos tipos de cuantificaciones: *“absoluta”* y *“relativa”*. Mediante la cuantificación *“absoluta”* es posible obtener la cardinalidad del conjunto de tuplas que satisfacen una consulta. Así pues, proporciona mecanismos para calcular el número de alumnos que satisfacen la condición envuelta en las consultas 5.1 y 5.2. Las consultas 5.3 y 5.4 implican el empleo de los dos casos de cuantificación *“relativa”* contemplados específicamente por el modelo relacional clásico: el cuantificador *“universal”*, empleado en la consulta 5.3, y el cuantificador *“existencial”* empleado en la consulta 5.4. El cuantificador *“universal”* es un cuantificador *“relativo”* en el siguiente sentido: partiendo de la consulta 5.3, para evaluar la respuesta es preciso de un lado, determinar cuales son las asignaturas consideradas y, de otro, cuantas de ellas han sido superadas por cada alumno, el cociente entre la segunda cantidad y la primera proporciona un valor relativo para cada alumno, dicho valor, comprendido entre 0 y 1, establece la proporción de asignaturas aprobadas. Cuando esta proporción toma el valor 1 nos encontramos ante el caso en el que ese alumno ha superado todas las asignaturas consideradas, ese alumno satisface la *“cuantificación universal”*, y, por tanto, aparece como resultado a la consulta. La consulta 5.4 representa un ejemplo en el se adopta una *“cuantificación existencial”*, en este caso,

el resultado se obtiene también de encontrar, para cada alumno, el cociente entre la cantidad de asignaturas que ha aprobado y la cantidad de asignaturas consideradas, solo que dicho cociente únicamente ha de ser superior a 0 para que un alumno sea considerado en la respuesta, indicando en este caso que, para ese alumno, *existe al menos una* de las asignaturas consideradas en la que ha aprobado.

El empleo de una aproximación difusa para el tratamiento de información en el ámbito relacional permite que algunos de los conceptos relativos a la cuantificación de las operaciones de manipulación pueda verse enriquecido. En efecto, pensemos en las posibilidades que se abrirían si pudieramos manejar consultas como:

Consulta 5.5 *“Dime en que asignaturas han habido muchos aprobados”*

Consulta 5.6 *“Dime si han habido pocos alumnos que hayan superado las Matemáticas y hayan suspendido la Física”*

Consulta 5.7 *“Dime si hay alguna asignatura para la que la mayoría de los alumnos que la han superado lo hayan hecho con aprobado”*

Consulta 5.8 *“¿Como de cierto es que la mayoría de los alumnos que han superado las asignaturas por parciales han aprobado con buena nota?”*

Donde podemos modelar las calificaciones mediante distribuciones de posibilidad. Los dos primeros ejemplos ilustran casos en los que se cuantifican, mediante valores difusos, cardinalidades *“absolutas”*. Los dos últimos casos emplean expresiones difusas construidas sobre cardinalidades *“relativas”*.

El estudio de los cuantificadores difusos permite modelar las anteriores proposiciones. Sin embargo, como ocurre en otros tantos aspectos del tratamiento difuso de la información, las soluciones propuestas no son únicas ni universales. Cada solución puede adoptar un criterio particular sobre cada uno de los siguientes aspectos:

1. Como representar un cuantificador difuso.
2. Como evaluar la cardinalidad de un conjunto difuso.
3. Como componer los cuantificadores difusos.

En [Zadeh 83] se realiza un estudio en profundidad sobre el tema de los cuantificadores difusos en el que se proponen los siguientes criterios para cada uno de los puntos anteriores.

Sobre el primer punto Zadeh adopta el empleo de "cuantificadores difusos de tipo 1" para modelar cuantificadores "absolutos" y, de "cuantificadores difusos de tipo 2", (cuantificadores que toman valor sobre los valores proporcionados por los de tipo 1), para modelar los cuantificadores "relativos". De esta forma se pueden representar mediante funciones de pertenencia de tipo 1 los cuantificadores que representan los siguientes conceptos: "varios", "pocos", "algunos", "no muchos", "aproximadamente cinco", "cerca de 10", "un gran número", etc. Mientras que mediante funciones de pertenencia de tipo 2 es posible cuantificar conceptos "relativos" como: "la mayoría", "una gran proporción", "frecuentemente", "de vez en cuando", "muchos de ellos", etc.

Con respecto a la cardinalidad, en el apartado 4.3.3 se introdujo la definición adoptada para GEFRED, dicha definición, basada en la *sigma-count*, nos proporciona un mecanismo adecuado para encontrar la cantidad de tuplas que satisfacen determinadas condiciones. En FIRST, como es natural, basaremos las operaciones de cuantificación en dicha definición. Además, vamos a permitir evaluar las cardinalidades difusas mediante el empleo de etiquetas lingüísticas que denominaremos "cuantificadores difusos".

Un cuantificador difuso *absoluto* modela mediante un subconjunto difuso alguna de las expresiones lingüísticas denominadas por Zadeh como de tipo 1: "varios", "pocos", "algunos", etc. En FIRST dicho subconjunto difuso, que lleva asociada una etiqueta lingüística, se representa por medio de una distribución de posibilidad trapezoidal. El dominio sobre el que se construye esta constituido por el conjunto posible de valores que puede tomar la cardinalidad de la consulta. En el ámbito de FIRST, definimos formalmente el concepto como sigue:

Definición 5.1 Sea R_{FG} una "relación base", cuya "cardinalidad difusa generalizada" es n , llamaremos **cuantificador difuso absoluto** a la etiqueta lingüística que identifica a una distribución de posibilidad trapezoidal construida sobre el intervalo $[0, n]$.

Con esta definición, el proceso de cuantizar una operación de manipulación se reduce a evaluar el grado en que son compatibles la "cardinalidad difusa generalizada" obtenida, con la distribución de posibilidad asociada a la etiqueta, que se emplea como cuantificador. Dicha compatibilidad se obtiene mediante el grado en que la cardinalidad obtenida pertenece a la distribución de posibilidad asociada a dicha etiqueta.

Vamos a ver como podemos resolver la consulta 5.5 empleando los recursos introducidos.

Ejemplo 5.1 Para el estudio de la consulta 5.5 vamos a partir de un esquema basado en las siguientes suposiciones:

- Supongamos que partimos de una relación cuya cabecera es: $\mathcal{H} = \{Alumno : D_{Nombre}, Calif_Fisica : D_{Notas}, Calif_Matem : D_{Notas}\}$, y cuyo cuerpo contiene 40 tuplas (40 alumnos).
- Supongamos que tenemos definidas sobre el dominio *Notas*, (intervalo [0,10]), las distribuciones trapezoidales correspondientes a las etiquetas lingüísticas “suspenso”, “aprobado”, “notable”, “sobresaliente” y “M.H.”. La forma explícita de dichas distribuciones no es relevante para ilustrar este ejemplo.
- Sea “muchos” un “cuantificador difuso absoluto” definido mediante la distribución trapezoidal que muestra la figura 5.9.
- Asumamos que hemos obtenido en Física el valor 15 para la “cardinalidad difusa generalizada” de la relación que resulta de la condición $Calif_Fisica \geq$ “aprobado”¹.
- Asumamos también que, para Matemáticas, la cardinalidad obtenida bajo la misma condición, $Calif_Matem \geq$ “aprobado”, es 18.

Entonces, a partir de la figura 5.9, podemos decir:

1. Que el número de aprobados en Física ha sido “mucho” con un grado 0.67.
2. Que el número de aprobados en Matemáticas ha sido “mucho” con un grado 0.87.

Se observa que podemos establecer un umbral bajo el cual consideremos que la condición no se satisface para el cuantificador “muchos”. Por ejemplo, si dicho umbral tomara el valor 0.8, la respuesta a la consulta 5.5 sería {Matemáticas}.

■

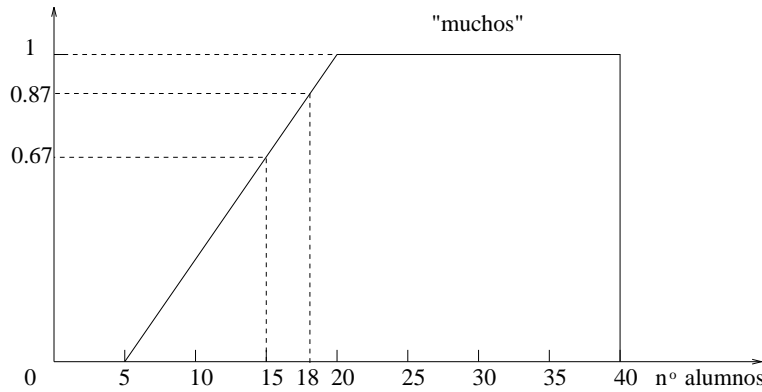


Figura 5.9: Etiqueta lingüística para el cuantificador “muchos”

Con respecto al “*cuantificador difuso relativo*”, en FIRST se operará en el marco de la siguiente definición:

Definición 5.2 Sean R_{FG} y R'_{FG} dos “*relaciones difusa generalizadas*”, llamaremos **cuantificador difuso relativo** de R_{FG} con respecto a R'_{FG} a la etiqueta lingüística que identifica a una distribución de posibilidad trapezoidal construida sobre el intervalo $[0,1]$, representando dicho intervalo el dominio de la “*cardinalidad relativa generalizada*” de R_{FG} con respecto a R'_{FG} .

El siguiente ejemplo ilustra la resolución de los aspectos concernientes a la “*cuantización relativa*” envuelta en la consulta 5.7.

Ejemplo 5.2 Para el estudio de la consulta 5.7 partiremos del esquema adoptado en el ejemplo 5.1. La figura 5.10 muestra la definición adoptada para la distribución asociada al “*cuantificador difuso relativo*” “*mayoría*”.

En otros términos, la consulta 5.7 requiere determinar, para cada asignatura, la “*cardinalidad relativa generalizada*” entre la relación de alumnos que han obtenido “*aprobado*” con respecto a la relación de alumnos han superado la asignatura. A

¹En el contexto en el que se establece la consulta 5.5 entenderemos por “*aprobados*” los alumnos han superado la asignatura, en ese sentido se emplea el operador \geq

continuación, determinar el grado en que dicha cardinalidad concuerda con el concepto “mayoría” mostrado en la figura 5.10. Es evidente que, en este caso, la relación de alumnos que han obtenido “aprobado” está incluida en la de los alumnos que han superado la asignatura, sin embargo no siempre tiene que ser así.

Vamos a detallar los pasos que nos conducirán a la resolución de la consulta 5.7:

1. Hemos de obtener la “cardinalidad relativa generalizada” para cada asignatura. Para ello emplearemos la expresión 4.35, la cual en el caso que nos ocupa adopta la forma:

$$card(\mathcal{S}_1/\mathcal{S}_2) = \frac{card(\mathcal{S}_1 \cap_G \mathcal{S}_2)}{card(\mathcal{S}_2)} = \frac{card(\mathcal{S}_1)}{card(\mathcal{S}_2)}$$

donde, para la asignatura Física,

- $\mathcal{S}_1 = \mathcal{S}(Actas; \Theta^{\leq \epsilon}(Calif_Física, “aprobado”))$
- $\mathcal{S}_2 = \mathcal{S}(Actas; \Theta^{\geq \epsilon}(Calif_Física, “aprobado”))$

y, para Matemáticas,

- $\mathcal{S}_1 = \mathcal{S}(Actas; \Theta^{\leq \epsilon}(Calif_Matem, “aprobado”))$
- $\mathcal{S}_2 = \mathcal{S}(Actas; \Theta^{\geq \epsilon}(Calif_Matem, “aprobado”))$

2. Lo cual nos conduce al cálculo de las cardinalidades $card(\mathcal{S}_1)$ y $card(\mathcal{S}_2)$ para cada asignatura. Supongamos que obtenemos:
 - Para Física $card(\mathcal{S}_1) = 12$ y $card(\mathcal{S}_2) = 15$ lo que nos da para la cardinalidad relativa un valor $12/15 = 0.8$.
 - Para Matemáticas $card(\mathcal{S}_1) = 10$ y $card(\mathcal{S}_2) = 18$, por lo que la cardinalidad relativa será $10/18 = 0.56$
3. Calculamos el grado en que es compatible cada cardinalidad relativa obtenida con el concepto “mayoría”, modelado por la distribución de posibilidad mostrada en la figura 5.10, y obtenemos los grados 1 y 0.2 para Física y Matemáticas, respectivamente.
4. Nuevamente, si establecemos un umbral de satisfacción, como por ejemplo 0.5, la consulta 5.7 entregaría el resultado {Física}.

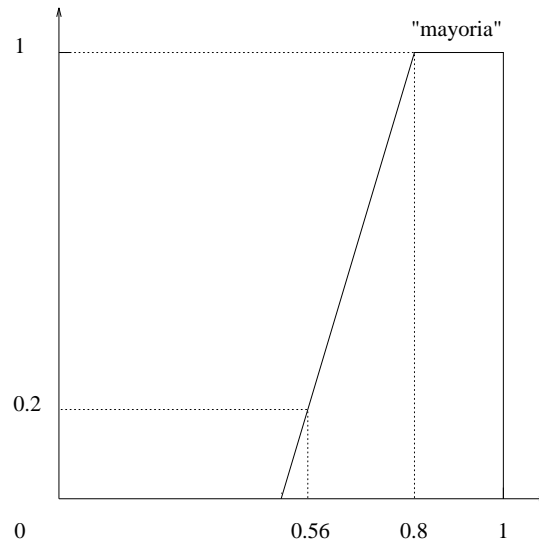


Figura 5.10: Etiqueta lingüística para el cuantificador relativo “mayoría”

■

Comentarios

Sobre los conceptos introducidos a lo largo de este subapartado es preciso realizar las siguientes consideraciones:

- El proceso de resolver una cuantificación en una consulta se contruye sobre las *selecciones difusas generalizadas* generadas a partir de las condiciones impuestas a las tuplas envueltas en dicha cuantificación. Como dichas selecciones pueden ser cualificadas por medio de un valor umbral, es posible actuar de forma que, aquellas tuplas que no satisfagan la condición con un grado suficientemente alto no sean recuperadas mediante la *selección difusa generalizada* y que, por tanto, no contribuyan a incrementar la cardinalidad calculada sobre la relación obtenida como respuesta a esa condición. De esta forma, mediante la cualificación de las *selecciones difusas generalizadas* podemos ejercer un control sobre el proceso de cuantificación.

Por otro lado, como hemos apuntado antes, el grado con que la cardinalidad calculada sobre el conjunto recuperado mediante una condición satisface un cuantificador, puede ser acotado inferiormente mediante un valor umbral. Así pues, también es posible controlar la precisión con que queremos obtener la satisfacción de un cuantificador. Los ejemplos 5.1 y 5.2 ilustran esta posibilidad.

Vemos pues que podemos actuar a dos niveles sobre el proceso de cuantificación en una consulta. Mediante el primero, implícito en el proceso de selección, podemos ignorar las tuplas que no satisfacen la condición con un grado suficientemente alto, (establecido mediante el umbral de consulta). En el ejemplo 5.1 puede ser conveniente ignorar los alumnos que satisfagan la condición \geq “*aprobado*” con un grado pequeño puesto que esto podría implicar que estubieramos contabilizando alumnos con notas más próximas al concepto de suspenso que al de apto. En el segundo nivel de control es posible adaptar el concepto representado por el cuantificador a las exigencias de precisión que exija cada problema. Así, en el ejemplo 5.1, es posible que, actuando sobre el valor del umbral, obtengamos como respuesta a la consulta una, las dos o ninguna de las asignaturas.

En resumen, la representación adoptada permite un mayor grado de flexibilidad para encontrar un solución para cada problema.

- Como se ha apuntado en el punto anterior, el proceso de cuantificación se realiza mediante el empleo de los mecanismos que proporciona el *álgebra difusa generalizada* (*selección e intersección difusos generalizados*), y de los operadores aritméticos habituales (suma y división).
- Los *cuantificadores difusos absolutos* son subjetivos y su definición puede depender de la cardinalidad de la relación sobre la que se aplica. Por ejemplo, el concepto “muchos” no puede ser modelado de igual forma si la relación sobre la que opera tiene 10 tuplas que si tuviera 1000. Por este motivo, en FIRST podremos definir los cuantificadores en el ámbito de cada relación.
- Como los *cuantificadores difusos relativos* se construyen sobre proporciones, su carácter es más general. Cuando establezco que la “mayoría” de las tuplas que satisfacen una condición c_1 satisfacen otra condición c_2 , lo importante no es determinar la cardinalidad de ambos conjuntos si no el cociente, (la proporción), entre

ambos. Por ejemplo, si dicho cociente es 0.8 da igual que se haya calculado de 8/10 que de 80/100, en ambos casos representa bien el concepto del cuantificador “mayoría”. Por este motivo en FIRST los *cuantificadores relativos* se contruyen en el ámbito de la Base de Datos.

- Los cuantificadores clásicos *existencial*, (\exists) y *universal* (\forall) pueden ser modelados mediante *cuatificadores difusos relativos*. El primero lleva asociada la función de pertenencia que modela el intervalo (0,1] con un umbral 0 para las consultas. El segundo se representa mediante la distribución de posibilidad {1/1} con un umbral 1 para las cuantificaciones.

5.3 Implementacion de FIRST

En esta sección abordaremos el estudio de los criterios adoptados para implementar los elementos del tratamiento impreciso de FIRST mediante el empleo de los recursos que proporcione el RDBMS que lo soporte, ([Medina et al, 93(a)]). Estos criterios deben satisfacer las siguientes condiciones:

- Ser coherentes con la formulación desarrollada en GEFRED.
- Emplear unicamente los recursos proporcionados por el RDBMS anfitrión. Estos recursos se agrupan en las siguientes categorías:
 - Tipos de datos soportados por el RDBMS.
 - Juego de sentencias disponible para definición, manipulación y control de datos.
 - Herramientas de construcción de programas. Estas herramientas pueden ser, p. e., SQL inmerso en algún lenguaje de alto nivel (C, Pascal, Fortran, etc.).
- Obtener resultados eficientes y operativos.

Para establecer los fundamentos de FIRST ha sido preciso elaborar un modelo que incorpore los procedimientos “difusos” en el ámbito relacional (GEFRED). Después, adoptar una representación específica para el conocimiento impreciso, para la que, en

nuestro caso, se ha adoptado un criterio de simplicidad. Nos encontramos ahora, en la fase en que debemos determinar como implementar dicha representación en el RDBMS anfitrión respetando los términos formulados en GEFRED.

A continuación vamos a describir en líneas generales, cuales son los niveles sobre los que podemos actuar para llevar a cabo la implementación de los principales módulos de FIRST empleando los recursos antes mencionados:

- A nivel de la Base de Datos. La Base de Datos es aquella colección de datos persistentes que constituyen la representación de una fracción de conocimiento del universo. Como nuestro sistema contempla la representación de conocimiento impreciso, debemos determinar la forma en que éste se almacena en la Base de Datos. Por tanto, habrá que extender la representación de los datos para albergar este tipo de información.
- A nivel del Catálogo del Sistema. En un RDBMS clásico existe una parte del sistema en la que se recoge toda aquella información que el gestor necesita saber acerca de los datos que almacena, "datos sobre los datos" (denominados a veces "*metadatos*"). Esta parte representa habitualmente dicha información mediante el empleo de tablas o relaciones organizadas siguiendo un esquema similar al empleado por la propia base de datos.

FIRST necesita tener información sobre los elementos de la Base de Datos que contienen información imprecisa así como de la naturaleza y representación de dicha información. Denominaremos Base de Metaconocimiento Difuso (FMB), a aquella extensión del Catálogo del Sistema que recoge la información necesaria sobre los datos de naturaleza imprecisa existentes en la Base de Datos.

- A nivel del Gestor de FIRST. El gestor posee conocimiento implementado sobre las operaciones de naturaleza imprecisa que puede realizar y de como hacerlo. El Procesador de Llamadas Difusas, FCP, es la parte del sistema que contiene las rutinas que se encargan de implementar esas operaciones.

En el subapartado 5.2.2 hemos mostrado los criterios empleados en FIRST para representar los diversos elementos del conocimiento impreciso. En los próximos subapartados centraremos nuestra atención en la descripción de la implementación del conocimiento impreciso en la Base de Datos, en el catálogo del sistema (mediante la FMB) y en los procedimientos del Gestor.

5.3.1 Implementación del Conocimiento Impreciso en la Base de Datos

En el sistema que se está describiendo existen tres tipos de atributos susceptibles de tratamiento impreciso. Se clasifican según el tipo del dominio que les subyace.

1. Atributos con "*datos precisos*" que tienen *etiquetas lingüísticas* definidas sobre ellos. Los llamaremos atributos de **Tipo 1**. Este tipo de atributos reciben una representación similar a la de los datos precisos. Sin embargo llevan asociada una información adicional en forma de etiquetas lingüísticas. La Base de Metaconocimiento Difuso se hará cargo de la representación de estas etiquetas, también recogerá información acerca de la naturaleza de estos atributos.

Tipo de Dato	F_TYPE	F_1	F_2	F_3	F_4
UNKNOWN	0	NULL	NULL	NULL	NULL
UNDEFINED	1	NULL	NULL	NULL	NULL
NULL	2	NULL	NULL	NULL	NULL
CRISP	3	d	NULL	NULL	NULL
LABEL	4	FUZZY_ID	NULL	NULL	NULL
INTERVALO[A,B]	5	A	0	0	B
APROX(d)	6	d	d-margen	d+margen	NULL
DIFUSO	7	α	$\beta - \alpha$	$\gamma - \delta$	δ

Para un atributo F de tipo 2 la representación crea un atributo F_TYPE para almacenar el código de tipo que le corresponde a cada valor y los atributos F_1, F_2, F_3, F_4 para almacenar los parámetros de cada dato. Los valores NULL que aparecen en los atributos que describen el dato tienen el significado de valor "no-aplicable" en el RDBMS anfitrión. Para el tipo LABEL el código FUZZY_ID representa un identificador para la etiqueta lingüística que se encuentra definida en la Base de Metaconocimiento. "Margen" es un parámetro que también se encuentra almacenado en la FMB.

Tabla 5.1: Representación de Atributos del tipo 2.

2. Atributos que pueden recoger "*datos imprecisos sobre referencial ordenado*". A estos atributos los denominaremos como de **Tipo 2**. Estos atributos recogen el

tipo de datos cuya representación ha sido descrita en el subapartado 5.2.2, permiten también, la representación de información incompleta en forma de datos de tipo UNKNOWN, UNDEFINED y NULL. Además, pueden contener “*información precisa*”. En la implementación de dichos atributos, para cada dato, tenemos que almacenar en la Base de Datos la siguiente información:

- El **tipo** que corresponde al dato que queremos almacenar. Según lo visto, éste puede ser: UNKNOWN, UNDEFINED, NULL, "CRISP", LABEL, INTERVAL, APROX., TRAPEZOIDAL.
- La **descripción** de los parámetros que definen el dato de acuerdo con el tipo al que pertenezca. P.e., si es un dato de tipo TRAPEZOIDAL, habrá que almacenar los valores de los parámetros α , β , γ , δ que lo describen.

Por otro lado, en la Base de Metaconocimiento habrá de quedar constancia de la presencia de estos atributos en la Base de Datos y de las características de los mismos.

La forma en que codifiquemos toda esta información dependerá de cada implementación y de que características del sistema queramos potenciar. En la tabla 5.1 mostramos la alternativa de implementación que hemos adoptado para FIRST en la que primamos los aspectos siguientes:

- Velocidad de ejecución frente a economía de almacenamiento. Para alguno de los tipos que puede recoger este atributo, se podría emplear una representación más compacta, sin embargo, esto ralentizaría la mayoría de las operaciones implicadas en una consulta.
- Uniformidad en la representación. Empleamos cinco atributos clásicos para representar cada atributo difuso de este tipo.
- Uso de los elementos del RDBMS anfitrión para representar la información respetando en dicha representación el esquema relacional. Este criterio, unido a otros adoptados a lo largo de esta exposición, posibilitará el reducir cualquier operación de naturaleza imprecisa a términos del modelo relacional clásico.

3. Atributos sobre *"dominios discretos con analogía"*. Atributos de **Tipo 3**. Estos atributos recogen datos escalares o distribuciones de posibilidad sobre dominios escalares, representados en la forma que se describe en el apartado 5.2.2. También aceptan datos de tipo UNKNOWN, UNDEFINED y NULL. Para atributos de este tipo tendremos que almacenar en la Base de Datos el tipo y la representación asociada a cada dato. La Base de Metaconocimiento Difuso contabilizará cada atributo de este tipo que aparezca en la Base de Datos. También almacenará las *"relaciones de semejanza"* definidas sobre el dominio subyacente.

En la Tabla 5.2 mostramos la alternativa de implementación adoptada para este tipo de atributos.

Tipo de Dato	F_TYPE	F_P1	F_1	...	F_Pn	F_n
UNKNOWN	0	NULL	NULL	...	NULL	NULL
UNDEFINED	1	NULL	NULL	...	NULL	NULL
NULL	2	NULL	NULL	...	NULL	NULL
SIMPLE	3	1	d	...	NULL	NULL
POS. DISTR.	4	p_1	d_1	...	p_n	d_n

El valor n es el máximo número de pares (grado posibilidad, valor) que puede representar el atributo instanciado. Este valor, que tiene que ser establecido en el momento en que se declara el atributo, acota la capacidad de ese atributo para representar distribuciones de posibilidad.

Tabla 5.2: Representación para atributos de Tipo 3

5.3.2 Implementación del Conocimiento Impreciso en el Catálogo de la BD. Base de Metaconocimiento Difuso (FMB)

Como hemos visto en el subapartado anterior, existe cierto tipo de información sobre los atributos descritos, que precisa ser almacenada de una forma accesible por el sistema. La Base de Metaconocimiento Difuso, FMB, va a ser la encargada de organizar toda

aquella información relacionada con la naturaleza imprecisa de estos atributos. En FIRST contemplamos la Base de Metaconocimiento Difuso como una extensión del Catálogo del sistema, por ello, organizaremos la información mediante el uso de tablas o relaciones. Los elementos del tratamiento impreciso que se almacenan en la Base de Metaconocimiento son los siguientes:

- Que atributos de la Base de Datos reciben tratamiento impreciso.
- Que clase de información imprecisa recogen. De que tipo son estos atributos (tipo 1, 2 ó 3).
- Objetos definidos en el ámbito de la Base de Datos, p.e., cuantificadores difusos de consulta.
- Que objetos difusos hay definidos sobre cada atributo:
 - Etiquetas lingüísticas.
 - Valores aproximados.
 - Relaciones de semejanza
 - Cualificadores del umbral de satisfacción de una consulta.
- La descripción de esos objetos.

5.3.2.1 Tablas de la Base de Metaconocimiento Difuso

A continuación detallamos como FIRST implementa la FMB. La Organización de las tablas que la constituyen se muestran en la figura 5.11. Procederemos a describir la estructura y significado de cada tabla de la FMB.

• FUZZY_COL

Esta tabla contiene una descripción de aquellos atributos de la Base de Datos que son susceptibles de tratamiento difuso. Esta descripción se realiza en términos análogos a los empleados en los diccionarios de los RDBMS convencionales. La tabla recoge una clasificación para cada atributo difuso en la que se establece el tipo de tratamiento de que será objeto.

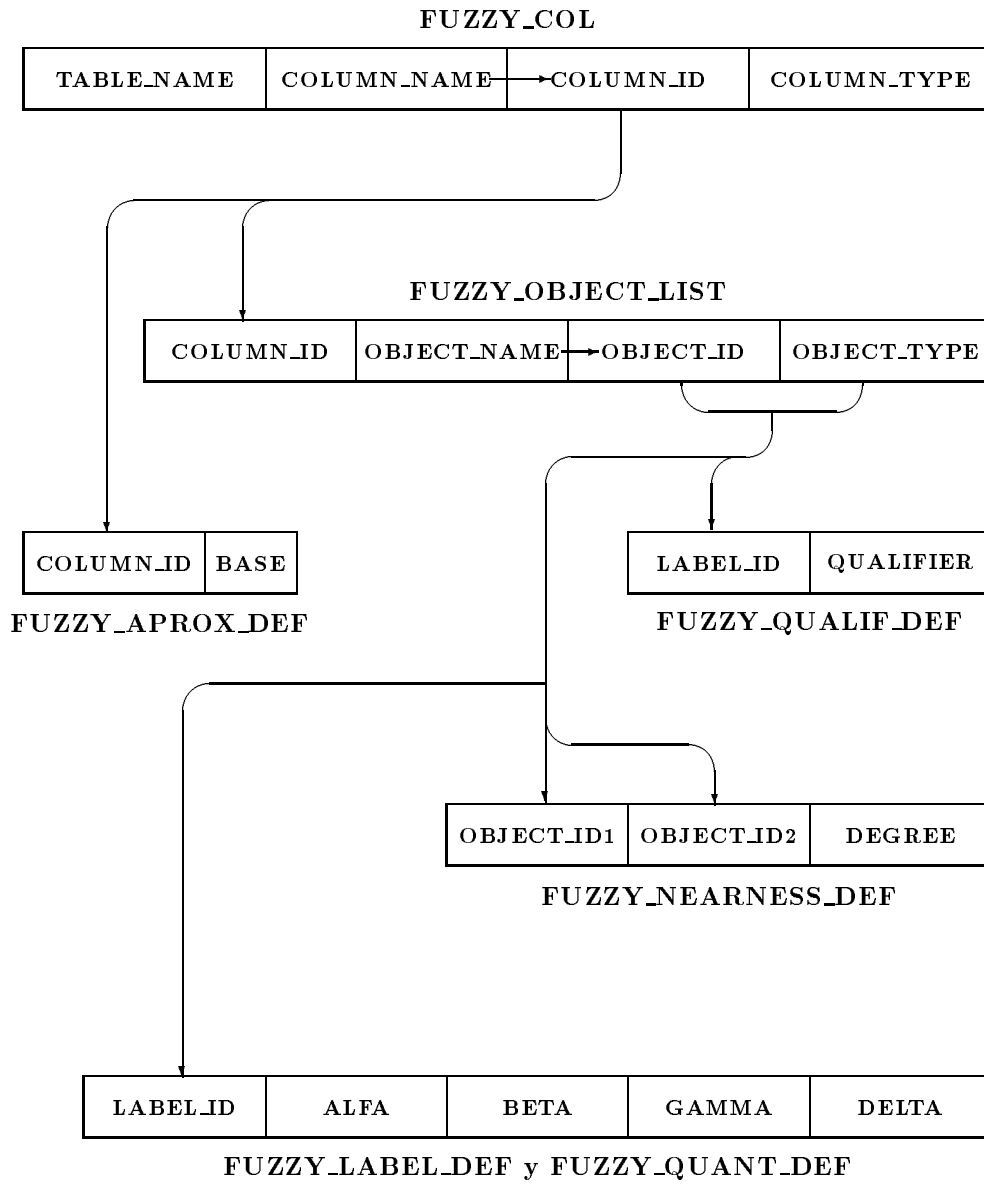


Figura 5.11: Esquema de la Base de Metaconocimiento FMB.

En una primera concepción, la tabla consta de cuatro atributos en los que se van a reflejar las características antes apuntadas. Estos atributos son:

- **TABLE_NAME**: Este campo es de tipo caracter; su longitud es compatible con la longitud que el sistema donde se implemente permita para los identificadores de las tablas.

Este campo contiene el nombre de la tabla a la cual pertenece el campo difuso que se referencia en el atributo **COLUMN_NAME**.

- **COLUMN_NAME**: También es de tipo carcter y responde a las mismas consideraciones en cuanto al tipo de identificadores que puede albergar. Los identificadores contenidos en este campo referencian a aquellas columnas que van a tener un tratamiento difuso, bien porque contengan información difusa, o bien porque, aún no teniendola, puedan ser objeto de consultas difusas; la naturaleza de estos campos se ve reflejada en el campo **COLUMN_TYPE**.
- **COLUMN_ID**: Será de tipo numérico positivo y con un rango por determinar en función de las necesidades del sistema a implementar (6 ó 7 dígitos pueden ser más que suficientes). Es la llave primaria de la tabla **FUZZY_COL** y asocia un identificador numérico al campo referenciado en el campo **COLUMN_NAME**; su misión será la de actuar en cualquier tabla de definición como referencia al campo al que va asociado, es decir, referenciamos a una columna fuzzy por su **COLUMN_ID**.
- **COLUMN_TYPE**: Este campo, de un carácter, tiene gran importancia, puesto que contiene información con respecto al tipo de datos y tratamientos que van a recibir las columnas referenciadas por el **COLUMN_ID**. Sobre este punto hay que detenerse a analizar el tipo de datos, difusos o no, que vamos a manejar, así como el tipo de tratamiento que van a recibir.

Tipos de Columnas

Además de los datos que convencionalmente trata la BD y que son de tipo "crisp", hemos de considerar los que se añadan dada la nueva concepción difusa de la BD. La clasificación de los diferentes tipos de columnas atenderá a criterios referentes, tanto a los posibles tipos de datos a albergar, como al tipo de tratamiento a que

van a estar sujetos los mismos. La clasificación para las diferentes columnas es la misma que la adoptada para los atributos en la sección 5.3.1:

- * **Tipo 1.** Serán aquellas columnas cuyos datos, aún siendo de tipo "crisp", admitirán (en consulta) tratamiento difuso y, por tanto, aparecerán en la descripción que se refleja en la tabla FUZZY_COL. Estos datos pueden ser, en principio, del tipo de los contemplados en la base de datos clásica de partida. Esta condición se traducirá por un "1" en el campo COLUMN_TYPE de la tabla FUZZY_COL. El dominio de este tipo de datos será el propio de la base de datos de partida.

El tratamiento que reciben los datos de los campos pertenecientes a este tipo, además de las funciones convencionales, viene determinado por las etiquetas lingüísticas que se definen sobre los mismos y que aparecerán convenientemente relacionadas en la tabla FUZZY_OBJECT_LIST, descrita más adelante.

- * **Tipo 2.** Esta clasificación ofrece al sistema el medio de reconocer en los campos referenciados, los datos difusos, tales como etiquetas lingüísticas y distribuciones de posibilidad, así como, en coordinación con la información contenida en la tabla FUZZY_OBJECT_LIST, el tratamiento a dispensarles. A este tipo de columnas el sistema les asignará como identificativo un "2" en el campo COLUMN_TYPE de la tabla FUZZY_COL.
- * **Tipo 3.** Este tipo de columnas necesitará la definición de unas relaciones de semejanza sobre los elementos de cada campo. La información relativa a estas definiciones, así como la relativa a los valores de dominio aceptados, aparece estructurada en las tablas FUZZY_OBJECT_LIST y FUZZY_NEARNNESS_DEF. Los campos de este tipo presentarán un "3" en el campo COLUMN_TYPE de la tabla FUZZY_COL.

● FUZZY_OBJECT_LIST

Esta tabla contiene una lista de los objetos de tipo difuso que hay definidos en las columnas de la base de datos. Además, refleja una clasificación de estos objetos mediante el campo OBJECT_TYPE. La información se estructura como se detalla a continuación:

- COLUMN_ID: NUMBER(6). Contiene el número que identifica a la columna sobre la que se define el objeto cuyo nombre aparece en el campo OBJECT_NAME de esta misma tabla. Constituye una clave externa de la tabla FUZZY_COL.
- OBJECT_NAME: CHARACTER(30). Contiene el nombre del objeto, el cual caerá dentro de alguno de los tipos contemplados en el campo OBJECT_TYPE.
- OBJECT_ID: NUMBER(6). Asociará un número a cada objeto, que servirá para referenciarlo en el resto de las tablas. Este campo junto con el campo COLUMN_ID constituye la llave primaria de esta tabla.
- OBJECT_TYPE: NUMBER(1). Especifica el tipo del objeto identificado por el campo OBJECT_ID; por tanto, dará una indicación al sistema sobre la tabla a la que habrá de dirigirse para encontrar la definición apropiada del objeto en cuestión.

Los valores que podrá tomar serán:

- * 0 para etiquetas lingüísticas de tipo trapezoidal.
- * 1 para escalares sujetos a tratamiento mediante relaciones de semejanza.
- * 2 para cualificadores definidos sobre el índice de cumplimiento en la consulta.
- * 3 para etiquetas lingüísticas definidas sobre cuantificadores relativos.
- * 4 para etiquetas definidas lingüísticas sobre cuantificadores absolutos.

• FUZZY_LABEL_DEF

Esta tabla contiene los puntos que determinan la función de pertenencia correspondiente a las etiquetas lingüísticas del tipo trapezoidal correspondientes a los tipos de objetos 0 y 4 de la tabla FUZZY_OBJECT_LIST. Los campos de esta tabla son:

- LABEL_ID: Number(6). Contiene el número que identifica a la etiqueta mediante el campo OBJECT_ID en la tabla FUZZY_OBJECT_LIST. Este campo constituye una llave externa para la tabla que nos ocupa y el campo LABEL_ID es una llave primaria de la misma.
- ALPHA: Number. $\alpha = \inf\{x : x \in \text{support}(label)\}$
- BETA: Number. $\beta = \inf\{x : x \in \text{kernel}(label)\}$

- GAMMA: Number. $\gamma = \sup\{x : x \in \text{kernel}(\text{label})\}$
- DELTA: Number. $\delta = \sup\{x : x \in \text{support}(\text{label})\}$

(Ver fig. 5.2 en la sección 5.2.2).

• FUZZY_APPROX_DEF

El contenido de esta tabla aporta información sobre la función de pertenencia de las etiquetas del tipo "aproximadamente". Estas funciones de pertenencia son de tipo triangular, con valor de pertenencia 1 para el valor sobre el que se considera la aproximación y están caracterizadas por este valor que ha de ser aportado en la consulta, y por la anchura de lo que podríamos considerar la base del triángulo. Por tanto, esta tabla constará de un identificador de etiqueta, LABEL_ID, que será la llave primaria de la misma, y de un campo que contendrá el valor MARGIN.

(Ver fig. 5.4 en la sección 5.2.2).

• FUZZY_NEARNESS_DEF

Esta tabla representa las medidas de proximidad o semejanza entre los diferentes valores de dominio permitidos sobre los campos de tipo "3" de la tabla FUZZY_COL. La información se estructura de la siguiente forma:

Dos campos se ocupan de establecer la relación entre cada dos valores del dominio y un tercer campo contiene el índice de semejanza de ambos valores. Se parte del concepto de semejanza por su carácter menos restrictivo en el sentido de que únicamente obliga a las relaciones a que cumplan la propiedad Simétrica y la Reflexiva. Por ello no habrá que aportar información en torno a los pares de valores cuyo grado de cercanía pueda ser deducido a partir de estas dos propiedades: $R(x, x) = 1$, $R(x, y) = R(y, x)$.

Así pues los campos se estructuran como sigue:

- OBJECT_ID1: Number. Contiene el primer valor de la pareja a relacionar.
- OBJECT_ID2: Number. Contiene el segundo valor de la pareja.
- DEGREE: Number ([0,1]). Contiene el grado en que son próximos los conceptos relacionados mediante OBJECT_ID1 y OBJECT_ID2.

Nótese que la llave primaria de la tabla que nos ocupa está formada por la combinación de los campos OBJECT_ID1 y OBJECT_ID2.

• FUZZY_QUALIFIERS_DEF

Esta tabla contiene la definición del *cualificador* asociada al valor lingüístico almacenado en la tabla FUZZY_OBJECT_LIST. El concepto de *cualificador* fué introducido en el subapartado 5.2.2. La figura 5.12 a) muestra un ejemplo de representación para este ítem. Los atributos de esta tabla son:

- LABEL_ID: Number(6). Es la clave primaria e identifica la etiqueta sobre la que está realizada la definición.
- QUALIFIER: Number[0,1]. Cualificador asociado a la etiqueta referenciada en el atributo LABEL_ID.

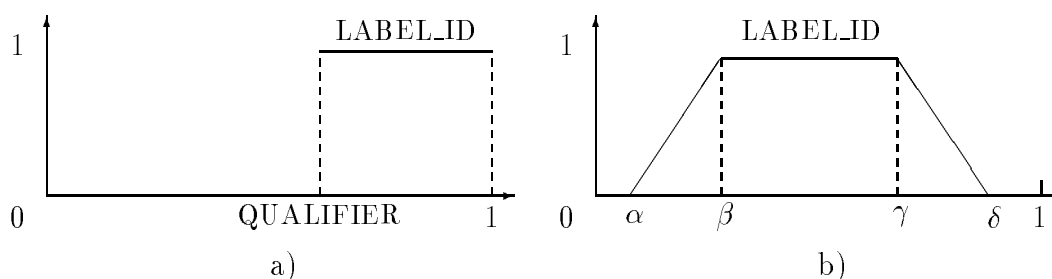


Figura 5.12: Definiciones de FUZZY_QUALIF_DEF y FUZZY_QUANT_DEF

• FUZZY_QUANTIFIERS_DEF

Esta tabla contiene la definición para las etiquetas usadas como cuantificadores relativo y absoluto en la consulta. Estas etiquetas identifican distribuciones de posibilidad trapezoidales sobre el dominio $[0,1]$. En el subapartado 5.2.2 se introdujo la definición de este tipo de etiquetas. Por tanto tenemos:

- LABEL_ID: Number(6). Es la clave primaria e identifica la etiqueta sobre la que está construida la definición.
- ALPHA: Number[0,1]. $\alpha = \inf\{x : x \in \text{support}(\text{label})\}$
- BETA: Number[0,1]. $\beta = \inf\{x : x \in \text{kernel}(\text{label})\}$
- GAMMA: Number[0,1]. $\gamma = \sup\{x : x \in \text{kernel}(\text{label})\}$
- DELTA: Number[0,1]. $\delta = \sup\{x : x \in \text{support}(\text{label})\}$

(Ver fig. 5.12 b)).

5.3.3 Ejemplo de Implementación en FIRST de la BD y la FMB

En este apartado vamos a ilustrar como se representan en la BD y en la FMB los elementos de la implementación introducidos. Para ello vamos a emplear el ejemplo que se muestra en la tabla 5.3 que consiste en una relación que recoge datos, algunos de ellos de naturaleza difusa, sobre un conjunto de empleados. A lo largo de este ejemplo vamos a ver como se implementa la información que recoge dicha relación. Veremos la estructura interna que adoptan los campos difusos en la base de datos. Por último, veremos como se actualiza la Base de Metaconocimiento Difusa para albergar la información relativa a los atributos y demás item difusos contemplados en dicha relación.

Los atributos NOMBRE y DIRECCIÓN contienen información "crisp", siendo el primero, la clave primaria de la relación. La información que contienen se implementa en la Base de Datos de la misma forma que lo hace el RDBMS anfitrión puesto que no recibirá tratamiento "difuso". Los atributos EDAD y SALARIO almacenan información "difusa" y precisarán de la definición de las de etiquetas lingüísticas que almacenan, son, por tanto, atributos de **tipo 2**.

Las figuras 5.13 a), b) y c) muestran, respectivamente, la definición de las etiquetas lingüísticas empleadas en los atributos EDAD y SALARIO y la función de pertenencia para la expresión "aproximadamente".

El atributo RENDIMIENTO permite también información "difusa", pero sobre un dominio escalar, (atributo de **tipo 3**), por tanto, para poder realizar una consulta

NOMBRE	DIRECCIÓN	EDAD	RENDIMIENTO	SALARIO
Luis	Recogidas	31	Bueno	Alto
Antonio	Reyes Católicos	Maduro	Regular	100000
Juan Carlos	Camino Ronda	Joven	Malo	90000
Francisco	P. A. Alarcón	Mayor	Excelente	Bajo
Julia	Puerta Real	Joven	Bueno	Medio
Inés	Manuel de Falla	#28	Bueno	125000
Javier	Gran Vía	*30,35	Regular	105000

El símbolo # significa "aproximadamente" está definido en la fig. 5.13. El símbolo * representa un valor intervalar.

Tabla 5.3: EMPLEADOS

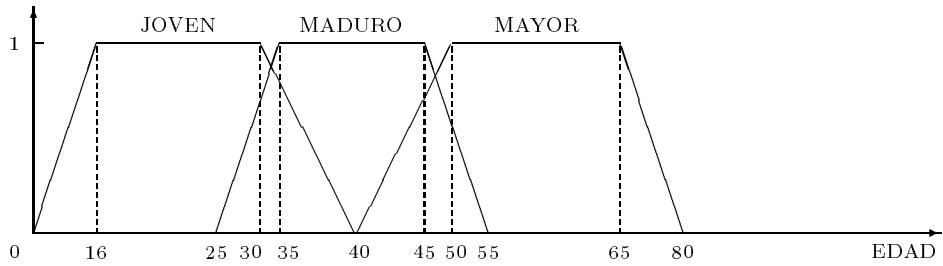
que contenga dicho atributo, necesitaré tener definida previamente una "relación de semejanza" sobre su dominio. Dicha relación se puede ver en la tabla 5.4.

Una vez hechas esas definiciones, la relación EMPLEADOS queda implementada en el RDBMS anfitrión como refleja la tabla 5.5, donde hemos adoptado para los atributos EDAD, SALARIO y RENDIMIENTO, la implementación mostrada en las tablas 5.1 y 5.2 del apartado 5.3.1.

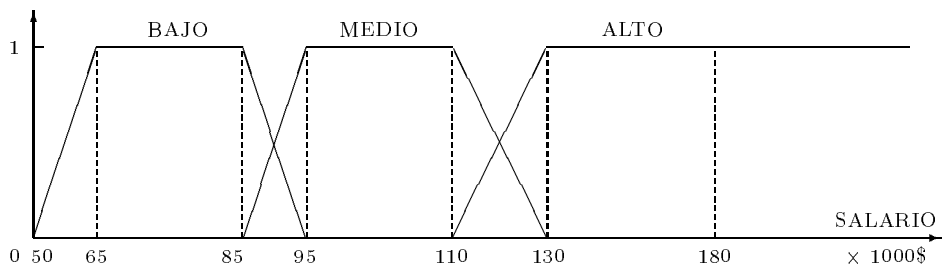
Las tablas 5.6, 5.7, 5.8, 5.9 y 5.10 muestran como queda almacenada en las tablas de la FMB toda la información concerniente a los item empleados sobre la relación EMPLEADOS.

$s_e(d, d')$	Malo	Regular	Bueno	Excelente
Malo	1	0.8	0.5	0.1
Regular	0.8	1	0.7	0.5
Bueno	0.5	0.7	1	0.8
Excelente	0.1	0.5	0.8	1

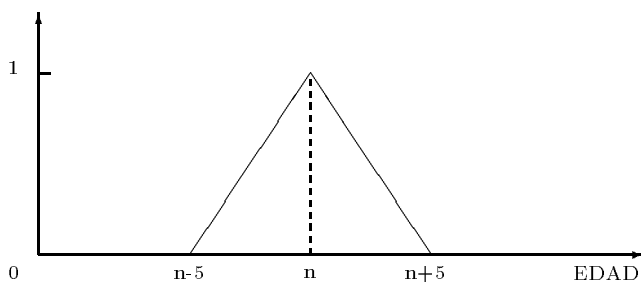
Tabla 5.4: Relación de Semejanza sobre RENDIMIENTO.



a)



b)



c)

Figura 5.13: Definición de etiquetas sobre los atributos EDAD y SALARIO.

...	EDAD_TIP	EDAD_1	EDAD_2	EDAD_3	EDAD_4	REND_TYPE	...
...	3	31	NULL	NULL	NULL	3	...
...	4	4	NULL	NULL	NULL	3	...
...	4	3	NULL	NULL	NULL	3	...
...	4	5	NULL	NULL	NULL	3	...
...	4	3	NULL	NULL	NULL	3	...
...	6	23	28	23	33	NULL	...
...	5	30	0	0	35	3	...

...	REND_P1	REND_1	SAL_TIP	SAL_1	SAL_2	SAL_3	SAL_4
...	1	Bueno	4	2	NULL	NULL	NULL
...	1	Regular	3	100000	NULL	NULL	NULL
...	1	Malo	3	90000	NULL	NULL	NULL
...	1	Excelente	4	0	NULL	NULL	NULL
...	1	Bueno	4	1	NULL	NULL	NULL
...	1	Bueno	3	125000	NULL	NULL	NULL
...	1	Regular	3	105000	NULL	NULL	NULL

Ver en la tabla FUZZY_OBJECT_LIST (5.7) los IDs para las etiquetas que aparecen en EDAD_1 y SAL_1.

Tabla 5.5: EMPLEADOS

TABLE_NAME	COLUMN_NAME	COLUMN_ID	COLUMN_TYPE
Empleados	Salario	0	2
Empleados	Edad	1	2
Empleados	Rendimiento	2	3

Tabla 5.6: FUZZY_COL

COLUMN_ID	OBJECT_NAME	OBJECT_ID	OBJECT_TYPE
0	Bajo	0	0
0	Medio	1	0
0	Alto	2	0
1	Joven	0	0
1	Maduro	1	0
1	Mayor	2	0
2	Malo	0	1
2	Regular	1	1
2	Bueno	2	1
2	Excelente	3	1

Tabla 5.7: FUZZY_OBJECT_LIST

COLUMN_ID	LABEL_ID	ALPHA	BETA	GAMMA	DELTA
0	0	5000	65000	85000	95000
0	1	85000	95000	110000	130000
0	2	110000	130000	180000	400000
1	0	0	16	30	40
1	1	25	35	45	55
1	2	40	50	65	80

Tabla 5.8: FUZZY_LABEL_DEF

COLUMN_ID	OBJECT_ID1	OBJECT_ID2	DEGREE
2	0	1	0.8
2	0	2	0.5
2	0	3	0.1
2	1	2	0.7
2	1	3	0.5
2	2	3	0.8

Tabla 5.9: FUZZY_NEARNESS_DEF

COLUMN_ID	MARGIN
1	5

Tabla 5.10: FUZZY_APROX_DEF

5.3.4 Implementación del Conocimiento Impreciso en los Procedimientos del Gestor. El Procesador de Llamadas Difusas (FCP)

FIRST debe habilitar al usuario para llevar a cabo todas las tareas relacionadas con la actualización, mantenimiento y recuperación de datos. Hasta ahora hemos abordado, basicamente, la implementación en FIRST de una versión de la estructura de datos formulada en GEFRED.

A continuación vamos a describir como implementamos los mecanismos para la manipulación de información "difusa". Los principales elementos de GEFRED que se encargan de la manipulación de datos son: los *comparadores difusos generalizados*, los cuales modelan las operaciones de comparación sobre datos "difusos", y el *Álgebra Difusa Generalizada*, que extiende las operaciones del Álgebra Relacional clásica a fin de que puedan tratar datos "difusos". Sobre los *comparadores difusos generalizados* hay que resaltar la siguiente circunstancia: en el modelo propuesto en GEFRED se adopta la definición de estos comparadores con un talante abierto a fin de que esta pueda acoger un amplio abanico de operadores de comparación sobre los diversos tipos

de datos "difusos" soportados. Por tanto, para realizar la implementación real de un *comparador difuso generalizado* concreto en el ámbito de FIRST, es necesario delimitar su descripción según la operación que queramos que realice: establecer el índice de compatibilidad de dos números difusos, ver en que grado un número difuso es mayor a otro, etc. En el subapartado 5.2.2 se expuso la forma en que modelamos los operadores "difusos" empleados en FIRST.

Una vez que hemos decidido como vamos a representar los operadores "difusos" a considerar, tenemos que analizar los recursos del Gestor anfitrión a emplear para poder implementarlos en FIRST. Siguiendo la filosofía que inspira la implementación que estamos describiendo, la forma de proceder consiste en traducir las operaciones de manipulación "difusas" en operaciones de manipulación clásicas. En nuestro caso las operaciones de manipulación clásicas se realizan mediante sentencias en SQL proporcionadas por el RDBMS anfitrión.

En base a lo anteriormente expuesto, precisamos de una serie de procedimientos que traduzcan las operaciones de manipulación de datos "difusas" a una serie de peticiones SQL clásicas al RDBMS. En este subapartado vamos a dar una descripción del módulo de FIRST encargado de implementar estos procedimientos. Se trata del procesador de llamadas difusas (Fuzzy Call Procesor, FCP). El FCP se encarga por tanto de, dada una operación de manipulación "difusa", obtener de la Base de Metaconocimiento y de la Base de Datos la información necesaria acerca de la representación y del significado de los elementos "difusos" implicados y traducir dicha operación a un conjunto de sentencias SQL a ejecutar por el RDBMS anfitrión. Se ha adoptado SQL para resolver las sentencias difusas porque, a pesar de presentar algunos inconvenientes como lenguaje relacional, presenta un alto grado de difusión en el ámbito de las implementaciones comerciales del modelo relacional. Esto permite que la implementación de las operaciones de manipulación difusas de FIRST que vamos a describir, sea fácilmente portable a un gran número de plataformas. En el apéndice A mostramos un ejemplo realizado sobre el RDBMS Oracle.

La amplia difusión de SQL ha motivado la aparición de múltiples versiones y dialectos. Esto ha desencadenado la introducción de diversas normas con el propósito de estandarizar su definición. Una de las normas más extendidas es la sintaxis ANSI/ISO de SQL. Desgraciadamente, la evolución de los desarrollos suele ir por delante de los esfuerzos de estandarización, por lo que cada fabricante acostumbra a ofrecer su propia

versión de SQL a la que enriquece con nuevas posibilidades, el precio que se paga por ello es la pérdida de portabilidad. En la descripción de la implementación del FCP que se abordará a continuación, se intentará recurrir a las operaciones de SQL estándar, en algunos casos se emplearán funciones que, por no estar muy desarrolladas en el estándar, constituyen uno de los aspectos más enriquecidos en las versiones comerciales. Cuando sea posible por el carácter estándar de una sentencia o cláusula, se empleará, para la misma, la notación adoptada por dicho estándar. Cuando el elemento empleado no esté estandarizado se utilizará una notación suficientemente explicativa sin menoscabo de la sintaxis que éste adopte en cada implementación de SQL.

A continuación vamos describir el proceso seguido por el FCP para trasladar las diferentes operaciones de manipulación difusas a operaciones clásicas.

5.3.4.1 Procesado de las Operaciones de Manipulación Difusas

Las *operaciones de manipulación difusas* permiten resolver los aspectos relacionados con la introducción de información “difusa” en la base de datos, con la actualización (modificación y supresión) de la misma y con la consulta de la información presente en la base de datos. Las sentencias que desencadenan tales acciones reciben por parte del FCP un tratamiento que puede ser dividido en las siguientes etapas:

1. Comprobar la corrección sintáctica² de la sentencia formulada. Emitir, si hubiere lugar a ello, los oportunos mensajes de error.
2. Verificar y extraer la interpretación semántica de la sentencia a procesar. Para ello, debe comprobarse la existencia y características de los ítems implicados en la operación, así como, la compatibilidad entre los atributos y las operaciones requeridas sobre los mismos. Todos estos aspectos se consiguen mediante la exploración en la FMB de los elementos semánticos que constituyen la petición formulada al sistema.
3. Una vez superada esta etapa, el FCP posee la información que precisa sobre todos los elementos que intervienen en proceso a resolver. También tiene una descripción en términos que comprende de la tarea que se pretende.

²En este apartado no vamos a introducir una sintaxis concreta para expresar las operaciones de manipulación difusas. La resolución de esta etapa depende, por tanto, de la sintaxis empleada.

4. Con toda esta información, el FCP esta en condiciones de expresar la operación formulada, mediante un conjunto de sentencias SQL, con objeto de obtener la respuesta deseada del RDBMS anfitrión y con la presentación adoptada para los diferentes tipos de datos considerados.

Como quiera que la consulta es el aspecto que representa mayor dificultad en el tratamiento difuso de la información, centraremos nuestros esfuerzos en describir como el FCP resuelve ese aspecto. Cabe decir que, en la descripción que sigue, se ilustran unas directrices de como resolver algunos de los puntos relativos a la manipulación, en este sentido, los apartados que siguen constituyen ejemplos de como desarrollar esas directrices, por tanto, no establecen una forma única ni óptima de llevarlas a cabo, la implementación efectiva en cada plataforma, proporcionará las herramientas definitivas para conseguir los mejores resultados para las operaciones descritas en los apartados siguientes. A continuación, abordaremos los elementos implicados en el proceso de consulta y la forma en que el FCP resuelve este tipo de operaciones.

5.3.4.2 Implementación de la Selección Difusa Generalizada

Tanto en el caso clásico como en el difuso, la resolución de una consulta puede involucrar todos los recursos del álgebra o del cálculo relacional. En este estudio empezaremos por analizar el proceso de resolución de la selección o restricción. En nuestro caso, el análisis se efectúa sobre la *selección difusa generalizada*. Dicha selección basa su funcionamiento en el empleo del *comparador difuso generalizado*. El subapartado 5.2.2 muestra el criterio adoptado en FIRST para la representación de dichos comparadores, (la fig. 5.8 ilustra de que forma actúan).

En el ámbito de la *selección difusa generalizada*, el cometido de cada comparador es el de evaluar las tuplas que satisfacen la condición expresada por encima de un valor umbral. El resultado debe ser una *relación difusa generalizada* en la que aparecen los *atributos de compatibilidad* calculados de la condición expresada mediante dicho comparador. Basándonos en la estructura de datos descrita en los subapartados 5.3.1 y 5.3.2, vamos a mostrar el proceso desencadenado en el FCP por una *selección difusa generalizada* para cada uno de los comparadores adoptados en FIRST. Nos vamos a detener unicamente en la última de las etapas analizadas en el apartado 5.3.4.1, por lo que suponemos cubiertas por el FCP las etapas anteriores. Esto implica que el FCP tiene almacenada toda la información concerniente a la estructura de los atributos

implicados, así como del resto de los ítems difusos empleados. La forma en que el FCP organiza en memoria toda esa información depende del lenguaje de programación empleado, del interface usado con el gestor anfitrión y de los compromisos adoptados por la implementación desarrollada. El apéndice A muestra un ejemplo de realización práctica de algunas de las rutinas implicadas en dicho proceso.

Bajo la hipótesis anterior, el algoritmo general³ propuesto para la traducción de una *selección difusa generalizada* a una sentencia SQL clásica se describe a continuación:

Algoritmo 5.1 *Consta de los siguientes pasos:*

1. *Descomponer la relación objeto de restricción en subrelaciones disjuntas conteniendo cada una de ellas datos de un único tipo. Esto se puede realizar mediante la condición clásica: $F_TYPE=t$, donde, F_TYPE es el campo que contiene el tipo de dato que almacena cada tupla para el atributo difuso F y t es el código para cada tipo según muestran las tablas 5.1 y 5.2.*
2. *Extraer de cada una de las subrelaciones anteriores el conjunto de las tuplas que satisfacen la condición impuesta en la selección difusa generalizada particionado en tantos conjuntos disjuntos de tuplas como requiera el cálculo de los respectivos grados de compatibilidad. La forma en que se realizan la mencionadas particiones depende de la condición difusa establecida en la selección y del tipo de dato con el que se esté evaluando. En cualquier caso, las condiciones clásicas que realizan dicha partición son calculadas por el FCP a partir de los elementos que constituyen dicha selección.*
3. *En base a la implementación para atributos difusos mostrada en las tablas 5.1 y 5.2, construir, para cada tipo de dato consultado, la presentación adoptada en orden a visualizarlo con un formato concreto. Para realizar esto fué preciso realizar el paso 1.*
4. *Expresar mediante funciones SQL el cálculo de los “grados de compatibilidad” para cada una de las particiones realizadas al efecto.*

³En el apéndice A se recoge una variación de este algoritmo adaptada a las características específicas del RDBMS empleado.

5. Aglutinar las sentencias SQL generadas para satisfacer cada uno de los pasos anteriores. Las condiciones que generan cada una de las particiones se conectan mediante el operador lógico AND. Cada partición genera una consulta SQL clásica en la que la proyección sobre los atributos difusos y de “compatibilidad” está generada como se indica en los pasos 3 y 4, respectivamente. Cada una de estas particiones se conectan mediante el operador UNION de SQL .

Seguidamente, detallaremos los aspectos más relevantes de este algoritmo para cada una de las selecciones disponibles en FIRST.

- *SELECCIÓN BASADA EN EL COMPARADOR “IGUAL A”*

La expresión 5.1 del subapartado 5.2.2 muestra la forma del comparador “Igual a”. En lo sucesivo emplearemos para él la notación FEQ(<campo_difuso> ,<constante> ,<umbral>). Donde:

- <campo_difuso> es el identificador de un atributo de tipo 1, 2 ó 3.
- <constante> es el identificador de un dato constante perteneciente a alguno de los tipos descritos en 5.2.2 y compatible con los valores soportados por <campo_difuso>.
- <umbral> es el identificador de un valor umbral, (etiqueta o número), que cualifica la comparación.

Para analizar la forma en que se resuelve la selección, es preciso distinguir entre los diferentes tipos de atributos difusos a los que puede aplicarse:

- **Atributos Tipo 1.** Estos atributos solo contienen información precisa de tipo numérico por lo que, para su tratamiento, solo hay que evaluar la comparación de un número con los tipos de datos difusos considerados. No es preciso, por tanto, realizar la partición del paso 1 del algoritmo 5.1. La tabla 5.11 esquematiza el cálculo y las sentencias obtenidas para resolver la selección sobre este tipo de atributos para cada una de las condiciones posibles. Para este tipo de atributos el operador FEQ lleva implícito el comparador “Aproximadamente igual” (expr. 5.2). Esto ocurre cuando comparamos dos números precisos, en este caso emplearemos, si existe, la definición del *margen* dada la tabla FUZZY_APROX_DEF de la FMB.

TIPO DATO	CÁLCULO	SENTENCIA QUE CALCULA
CRISP/ APROX(n)	<p>$a = margen * (1 - u)$</p>	<p>SELECT ..., F, $1 - ABS(F-n)/margen$, ... FROM tabla WHERE $ABS(F-n) \leq a$</p>
LABEL/ DIFUSO	<p>$a = u * (\beta - \alpha) + \alpha$ $b = \delta - u * (\delta - \gamma)$</p>	<p>SELECT ..., F, $(F-\alpha)/(\beta - \alpha)$, ... FROM tabla WHERE $F \geq a$ AND $F < \beta$</p> <p>UNION</p> <p>SELECT ..., F, $(\delta-F)/(\gamma - \delta)$, ... FROM tabla WHERE $F \leq b$ AND $F > \gamma$</p> <p>UNION</p> <p>SELECT ..., F, 1, ... FROM tabla WHERE F BETWEEN β AND γ</p>
INTER VALO[r,s]		<p>SELECT ..., F, 1, ... FROM tabla WHERE F BETWEEN r AND s</p>

Tabla 5.11: Selección basada en FEQ sobre atributos Tipo 1

Sobre el esquema de la tabla 5.11 es conveniente hacer las siguientes consideraciones:

1. La primera columna muestra los diferentes tipos de <constante> soportados por el operador.
2. La segunda, ilustra graficamente y calcula, los elementos mas importantes de la operación para cada tipo de dato.
3. La tercera muestra un modelo de sentencia SQL con la que se obtienen: los *valores de dominio* del atributo considerado, F, en este caso y, los *grados*

de *compatibilidad* obtenidos en la selección para un `<umbral>` u . Para ello, realiza las particiones a que dan lugar los cálculos anteriores.

4. Los valores a y b delimitan el intervalo del dominio subyacente que presenta un *grado de compatibilidad* superior o igual al `<umbral>`, u , establecido en la condición. Dichos valores son calculados por el FCP a partir de la expresión del operador FEQ proporcionada en la consulta.
 5. El FCP resuelve de forma similar la condición para los tipos LABEL y DIFUSO. La diferencia entre uno y otro estriba en que, para el tipo LABEL precisa obtener de la FMB los parámetros $(\alpha, \beta, \gamma, \delta)$ mientras que, con el tipo DIFUSO dichos parámetros son proporcionados por la consulta.
 6. Las sentencias clásicas resueltas emplean operadores aritméticos comunes de SQL, así como la función ABS que devuelve el valor absoluto de su argumento. En el caso poco probable de que una implementación no contenga dicha función, la sentencia puede ser particionada en dos sentencias que no contengan tal función.
 7. El campo difuso F puede estar indexado en el RDBMS anfitrión, en cuyo caso aumentaría la eficiencia de la selección considerada en el esquema.
- **Atributos Tipo 2.** Estos atributos pueden almacenar los tipos de datos mostrados en la tabla 5.1. La selección sobre dichos atributos se construye a partir del operador de comparación expresado en la forma `FEQ(<campo_2>, <constante>, <umbral>)`, donde, `<campo_2>` identifica a un atributo de tipo 2 y `<constante>` es un valor constante aportado en la consulta que puede ser de cualquiera de los tipos contemplados en la tabla 5.1 salvo los tipos UNKNOWN, UNDEFINED y NULL para los que se construirá un operador especial. Para cada tipo constante considerado en el segundo argumento del operador FEQ pueden obtenerse diferentes sentencias clásicas. A continuación estudiaremos el proceso para cada tipo de dato.

La tabla 5.12 esquematiza la forma en que el FCP calcula las sentencias clásicas que resuelven la selección basada en la comparación con un valor constante de tipo preciso (“crisp”). Es preciso hacer algunas consideraciones a fin de aclarar el significado de lo expresado en dicha tabla:

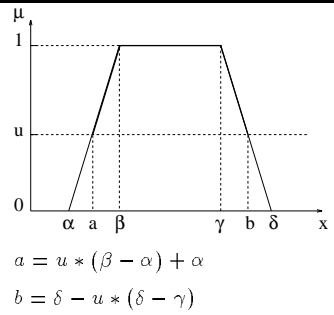
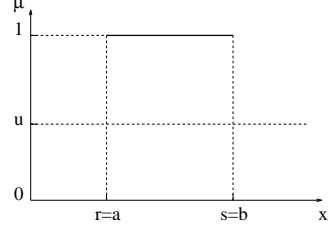
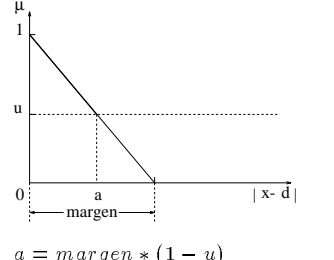
FEQ(F, "crisp", u)		
TIPO de DATO	CÁLCULO	SENTENCIA QUE CALCULA
UNKNOWN	FEQ(UNKNOWN, n,u)=1 $\forall n \in D, \forall u \in [0, 1]$	SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F _t =0
UNDEFINED	FEQ(UNDEFINED, n,u)=0 $\forall n \in D, \forall u \in [0, 1]$	No es preciso recuperar tuplas de esta partición
NULL	FEQ(NULL, n,u)=1 $\forall n \in D, \forall u \in [0, 1]$	SELECT ..., 'NULL', 1, ... FROM tabla WHERE F _t =2
CRISP	FEQ(m,n,u)= $\delta(m,n)$ $\forall n, m \in D, \forall u \in [0, 1]$	SELECT ..., F ₁ , 1, ... FROM tabla WHERE F _t =3 AND F ₁ =n
LABEL/ DIFUSO	 $a = u * (\beta - \alpha) + \alpha$ $b = \delta - u * (\delta - \gamma)$	SELECT ..., <i>etiqueta</i> , $(n-\alpha)/(\beta - \alpha)$, ... FROM tabla WHERE F _t =t AND $n \geq a$ AND $n < \beta$ UNION SELECT ..., <i>etiqueta</i> , $(\delta-n)/(\gamma-\delta)$, ... FROM tabla WHERE F _t =t AND $n \leq b$ AND $n > \gamma$ UNION SELECT ..., <i>etiqueta</i> , 1, ... FROM tabla WHERE F _t =t AND n BETWEEN β AND γ
INTER VALO[r,s]	 $r=a$ $s=b$	SELECT ..., <i>intervalo</i> , 1, ... FROM tabla WHERE F _t =5 AND n BETWEEN F ₁ AND F ₄
APROX(d)	 $a = margen * (1 - u)$	SELECT ..., <i>aprox</i> , $1-ABS(F_1+margen-n)/margen$, ... FROM tabla WHERE F _t =6 AND $ABS(F_1+margen-n) \leq a$

Tabla 5.12: Selección usando FEQ(F, "crisp", u) sobre atributos de Tipo 2

1. La tabla 5.12 clasifica las sentencias generadas para resolver la comparación con cada tipo de dato soportado por el atributo F. Para cada tipo de dato se realiza una partición según lo expuesto en el paso 1 del algoritmo 5.1. La condición $F_t = tipo$ se encarga de realizar dicha partición. El resultado final de la selección se obtendrá mediante la union de las sentencias que genera cada partición.
2. Para realizar el paso 2 del algoritmo 5.1, se divide la partición resultante para cada dato en tantas sentencias como sea necesario para expresar el cálculo de los “*grados de compatibilidad*”. Dichas sentencias se muestran en la tabla 5.12 unidas por el operador de UNION de SQL.
3. La presentación para los diferentes tipos de datos contenidos en la tabla consultada puede reconstruirse a partir de la información proporcionada por los atributos F_t, F_1, F_2, F_3, F_4. En la tabla 5.12 notamos esta circunstancia mediante el empleo de los identificadores: *etiqueta*, *intervalo* y *aprox*. En el apéndice A mostraremos como construir una presentación para cada tipo de datos utilizando algunas cláusulas de SQL.
4. Como veremos en el próximo punto, algunas de las sentencias calculadas precisan del empleo de una función especial que no se encuentra definida en el estándar ANSI de SQL. Dicha función, que expresaremos mediante la sintaxis `DECODE(expr, valor_1, result_1, ..., valor_n, result_n, defecto)`, decodifica la expresión *expr* de forma que si esta toma el valor *valor_1* devuelve el valor *result_1*, si toma el valor *valor_2* devuelve el valor *result_2* y así sucesivamente, si *expr* no toma ninguno de los valores de la lista, la función devuelve el valor *defecto*. La existencia de esta función facilita la conversión de sentencias. La implementación de FIRST sobre la plataforma Oracle que se mostrará en el apéndice A empleará dicha función.
5. En la fila etiquetada como LABEL/DIFUSO se muestra un tratamiento análogo para este tipo de datos que difiere unicamente en la expresión de los parámetros: *etiqueta*, *t*, α , β , γ , δ , *a* y *b* en la columna SENTENCIA QUE CALCULA para cada uno de los tipos de datos:
 - Para datos de tipo DIFUSO dichos parámetros representan lo siguiente: *etiqueta* es la presentación que se adopta para una distribución de posibilidad reconstruida como se explica en el punto 3, *t* es “7” (ver tabla

5.1), α es “F_1”, β es “F_2+F_1”, γ es “F_3+F_4”, δ es “F_4”, a es “ $u * F_2 + F_1$ ” y b es “ $u * F_3 + F_4$ ”.

- Para datos de tipo LABEL: *etiqueta* es la reconstrucción a partir de la FMB del nombre asociado al atributo F_1, t es “4”, α debe calcularse mediante la expresión “ $\text{DECODE}(F_1, l_1, \alpha_1, \dots, l_r, \alpha_r)$ ”⁴, β mediante “ $\text{DECODE}(F_1, l_1, \beta_1, \dots, l_r, \beta_r)$ ”, γ por “ $\text{DECODE}(F_1, l_1, \gamma_1, \dots, l_r, \gamma_r)$ ”, δ se sustituye por “ $\text{DECODE}(F_1, l_1, \delta_1, \dots, l_r, \delta_r)$ ”, a se calcula mediante la sentencia “ $u * \text{DECODE}(F_1, l_1, \beta_1, \dots, l_r, \beta_r) + \text{DECODE}(F_1, l_1, \alpha_1, \dots, l_r, \alpha_r) * (1-u)$ ” y b por la sentencia “ $u * \text{DECODE}(F_1, l_1, \gamma_1, \dots, l_r, \gamma_r) + \text{DECODE}(F_1, l_1, \delta_1, \dots, l_r, \delta_r) * (1-u)$ ”

6. Puede incrementarse la eficiencia en el proceso de búsqueda de tuplas que satisfacen la selección, mediante el empleo de índices sobre los atributos F_t, F_1, F_2, F_3, F_4. También es posible restringir el número de tuplas sobre las que operar empleando condiciones adicionales. Por ejemplo, para la fila etiquetada mediante LABEL/DIFUSO, está claro que si el valor “*crisp*”, n , con el que se compara no está entre α y δ , no se recuperará la tupla que contenga ese valor difuso, por tanto se evita un tratamiento ulterior para dicha tupla. Dicha condición podría ser expresada mediante la cláusula “AND n BETWEEN α AND δ ”. De todas formas, la efectividad o no de esta operación depende de que el optimizador de consultas de la plataforma que estemos empleando tenga en cuenta estas circunstancias al evaluar una condición compuesta, por lo que no es posible generalizar los mecanismos descritos a todas las plataformas. En el apéndice A adoptaremos las soluciones que nos optimicen el proceso de acuerdo con el planificador de Oracle.

Las tablas 5.13 y 5.14 resumen la forma en que el FCP calcula las sentencias clásicas que resuelven la selección basada en la comparación con un valor constante de tipo “distribución trapezoidal”, (“DIFUSO” en tabla 5.1), o de tipo “LABEL” en dicha tabla. De igual modo, las tablas 5.15 y 5.16 ilustran el proceso para la selección basada en la comparación con un valor constante de

⁴La lista de argumentos que sigue al primero, F_1, son extraídos por el FCP de la FMB, α_i es el valor contenido en el campo ALPHA de la tabla FUZZY-LABEL-DEF para la etiqueta identificada con el valor l_i

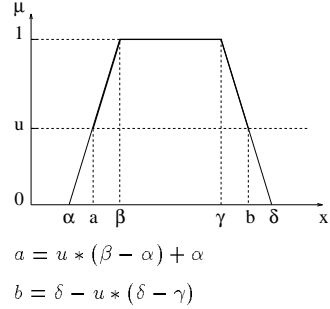
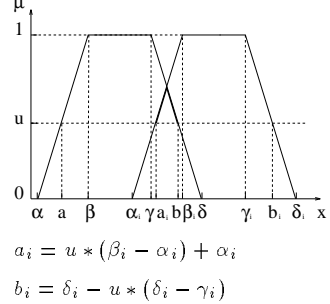
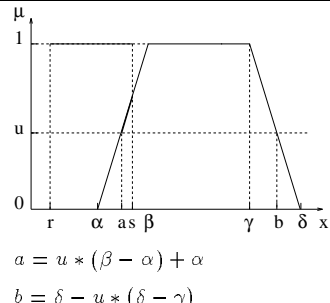
FEQ(F, "label", u), FEQ(F, "[α, β, γ, δ]", u)		
TIPO de DATO	CÁLCULO	SENTENCIA QUE CALCULA
UNKNOWN	FEQ(UNK., "difuso", u)=1 ∀u ∈ [0, 1]	SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0
UNDEFINED	FEQ(UNDEF., "difuso", u)=0 ∀u ∈ [0, 1]	No es preciso recuperar tuplas de esta partición
NULL	FEQ(NULL, "difuso", u)=1 ∀u ∈ [0, 1]	SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2
CRISP	 <p> $a = u * (\beta - \alpha) + \alpha$ $b = \delta - u * (\delta - \gamma)$ </p>	SELECT ..., F_1, (F_1-α)/(β-α), ... FROM tabla WHERE F_t=3 AND F_1 ≥ a AND F_1 < β UNION SELECT ..., F_1, (δ-F_1)/(γ-δ), ... FROM tabla WHERE F_t=3 AND F_1 ≥ b AND F_1 < γ UNION SELECT ..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1 BETWEEN β AND γ
LABEL/ DIFUSO	 <p> $a_i = u * (\beta_i - \alpha_i) + \alpha_i$ $b_i = \delta_i - u * (\delta_i - \gamma_i)$ </p>	SELECT ..., etiqueta, (δ_i - α_i)/((β_i - α_i)-(γ_i - δ_i)), ... FROM tabla WHERE F_t=t AND b_i ≥ a AND γ_i < β UNION SELECT ..., etiqueta, (δ - α_i)/((β_i - α_i)-(γ - δ)), ... FROM tabla WHERE F_t=t AND a_i ≤ b AND α_i > γ UNION SELECT ..., etiqueta, 1, ... FROM tabla WHERE F_t=t AND β ≤ γ_i AND α_i ≤ γ
INTER VALO[r,s]	 <p> $a = u * (\beta - \alpha) + \alpha$ $b = \delta - u * (\delta - \gamma)$ </p>	SELECT ..., intervalo, (F_4-α)/(β-α), ... FROM tabla WHERE F_t=5 AND F_4 < β AND F_4 ≥ a UNION SELECT ..., intervalo, (F_1-δ)/(γ-δ), ... FROM tabla WHERE F_t=5 AND F_1 > γ AND F_1 ≤ b UNION SELECT ..., intervalo, 1, ... FROM tabla WHERE F_t=5 AND F_1 ≤ γ AND F_4 ≥ β

Tabla 5.13: Selección usando FEQ(F, "difuso", u) sobre atributos de Tipo 2

FEQ(F, "label", u), FEQ(F, "[α, β, γ, δ]", u)		
TIPO de DATO	CALCULO	SENTENCIA QUE CALCULA
APROX(d)	<p> $a = u * ((\beta - \alpha) + margen) + \alpha - margen$ $b = u * ((\gamma - \delta) - margen) + \delta + margen$ </p>	<pre> SELECT ..., aprox, (F_A-α)/(β - α + margen), ... FROM tabla WHERE F_t=6 AND F_1 ≥ (a - margen) AND F_1 < (β - margen) UNION SELECT ..., aprox, (F_1-δ)/(γ - δ - margen), ... FROM tabla WHERE F_t=6 AND F_1 ≤ (b - margen) AND F_1 > (γ - margen) UNION SELECT ..., aprox, 1, ... FROM tabla WHERE F_t=6 AND F_1 BETWEEN (β - margen) AND (γ - margen) </pre>

Tabla 5.14: Continuación de la tabla 5.13

tipo “intervalo[r,s]”. Por último, la comparación con un valor constante de tipo “aprox(d)” queda reflejada en las tablas 5.17 y 5.18.

Además de las consideraciones realizadas en torno a la tabla 5.12, es preciso comentar los siguientes aspectos específicos relativos a las tablas 5.13 a 5.18:

1. En las tablas 5.13 y 5.14 se muestra un tratamiento similar para los valores <constante>, LABEL y DIFUSO, esto es así por que dicho tratamiento está basado en los parametros α , β , γ y δ que son aportados directamente al operador FEQ, en el caso de tratarse del tipo DIFUSO, o extraídos de la FMB cuando se compara con una constante de tipo LABEL.
 2. La fila etiquetada como LABEL/DIFUSO adopta la convención expresada en el punto 5 de las consideraciones vertidas sobre la tabla 5.12, con la única salvedad de que, para la tabla 5.13, se aplican a los parámetros a_i , b_i , α_i , β_i , γ_i y δ_i en lugar de a los parámetros a , b , α , β , γ y δ .
- **Atributos Tipo 3.** La tabla 5.2 muestra el tipo de información que puede recoger este tipo de atributos. La expresión del operador de comparación queda en la forma FEQ(<campo_3>, <constante>, <umbral>), donde, <campo_3> identifica a un atributo de tipo 3 y <constante> es un valor constante aportado en la consulta que puede ser un escalar simple o una distribución de posibilidad. Para

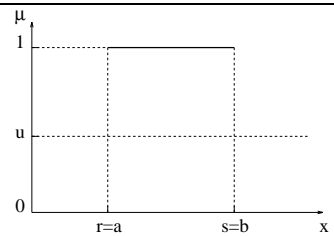
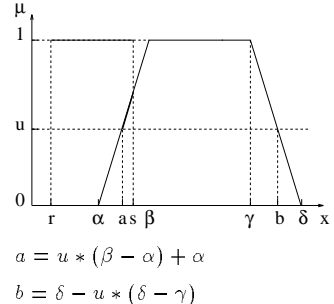
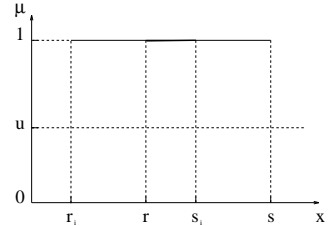
FEQ(F, "intervalo[r,s]", u)		
TIPO de DATO	CÁLCULO	SENTENCIA QUE CALCULA
UNKNOWN	FEQ(UNKNOWN, <i>intev</i> , u)=1 $\forall u \in [0, 1]$	SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0
UNDEFINED	FEQ(UNDEFINED, <i>intev</i> , u)=0 $\forall u \in [0, 1]$	No es preciso recuperar tuplas de esta partición
NULL	FEQ(NULL, <i>intev</i> , u)=1 $\forall u \in [0, 1]$	SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2
CRISP		SELECT ..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1 BETWEEN r AND s
LABEL/ DIFUSO	 $a = u * (\beta - \alpha) + \alpha$ $b = \delta - u * (\delta - \gamma)$	SELECT ..., <i>etiqueta</i> , (r- δ)/(γ - δ), ... FROM tabla WHERE F_t=t AND $b \geq r$ AND $\gamma < r$ UNION SELECT ..., <i>etiqueta</i> , (s- α)/(β - α), ... FROM tabla WHERE F_t=t AND $a \leq s$ AND $\beta > s$ UNION SELECT ..., <i>etiqueta</i> , 1, ... FROM tabla WHERE F_t=t AND $\gamma \geq r$ AND $\beta \leq s$
INTER VALO[r,s]		SELECT ..., <i>intervalo</i> , 1, ... FROM tabla WHERE F_t=5 AND F_1 $\leq s$ AND F_4 $\geq r$

Tabla 5.15: Selección usando FEQ(F, "intervalo[r,s]", u) sobre atributos de Tipo 2

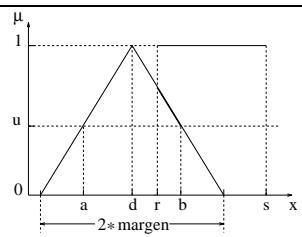
FEQ(F, "intervalo[r,s]", u)		
TIPO de DATO	CALCULO	SENTENCIA QUE CALCULA
APROX(d)	 <p> $c(x) \neq 0$ $\forall x \in [r - margen * (1 - u), s + margen * (1 - u)]$ </p>	<pre> SELECT ..., aprox, 1-r/F_4,... FROM tabla WHERE F_t=6 AND F_1 ≥ r - margen * (2 - u) AND F_1 < (r - margen) UNION SELECT ..., aprox, s/F_4-1,... FROM tabla WHERE F_t=6 AND F_1 ≤ s - margen * u AND F_1 > (s - margen) UNION SELECT ..., aprox, 1,... FROM tabla WHERE F_t=6 AND F_1 BETWEEN r - margen * (2 - u) AND s + margen * u </pre>

Tabla 5.16: Continuación de la tabla 5.15

los tipos UNKNOWN, UNDEFINED y NULL se empleará un operador especial. Las tablas 5.19 y 5.20 clasifican las sentencias generadas para resolver una selección basada en este operador para todas las posibles opciones. Para llegar a las expresiones correspondientes a la segunda columna de la tabla hay que tener en cuenta los siguientes puntos:

1. En cada tabla, el tratamiento aplicado a cada valor de dominio depende de su clasificación en la tabla 5.2, y de que sobre el atributo considerado exista definida, o no, una relación de semejanza entre los valores del dominio subyacente. Esta circunstancia se verá reflejada en la FMB, tablas FUZZY_OBJECT_LIST y FUZZY_NEARNESS_DEF. El operador FEQ, al igual que ocurre con su expresión para los atributos de tipo 1 y 2, está basado en la expresión 5.1. Sin embargo se adoptan versiones de dicho operador particularizadas a cada caso. De esta forma, las sentencias calculadas se basan en las siguientes expresiones de dicho operador:
 - Valor <constante> de tipo SIMPLE sin relación de semejanza en el dominio: $\mu_{igual_a}(d, n) = \delta(d, n)$ para comparación con datos de tipo SIMPLE y $\mu_{igual_a}(\tilde{d}, n) = \pi_{\tilde{d}}(n)$ para datos de tipo distribución de posibilidad. (Tabla 5.19)
 - Valor <constante> de tipo SIMPLE con relación de semejanza en el do-

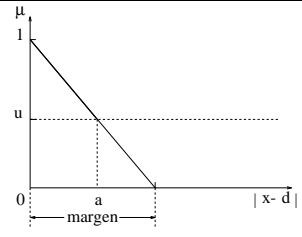
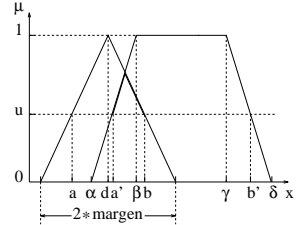
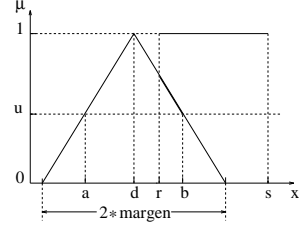
FEQ(F, "aprox(d)", u)		
TIPO de DATO	CÁLCULO	SENTENCIA QUE CALCULA
UNKNOWN	FEQ(UNKNOWN, $apr(d), u$)=1 $\forall u \in [0, 1]$	SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0
UNDEFINED	FEQ(UNDEFINED, $apr(d), u$)=0 $\forall u \in [0, 1]$	No es preciso recuperar tuplas de esta partición
NULL	FEQ(NULL, $apr(d), u$)=1 $\forall u \in [0, 1]$	SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2
CRISP	 <p>$a = margen * (1 - u)$</p>	SELECT ..., F_1, 1-ABS(F_1-d)/margen, ... FROM tabla WHERE F_t=3 AND ABS(F_1-d) ≤ a
LABEL/ DIFUSO	 <p>$a = d - margen * (1 - u)$ $b = d + margen * (1 - u)$</p>	SELECT ..., <i>etiqueta</i> , $((\delta - (d - margen)) / (margen - (\gamma - \delta)))$, ... FROM tabla WHERE F_t=t AND $b' \geq a$ AND $\gamma < d$ UNION SELECT ..., <i>etiqueta</i> , $((d + margen) - \alpha) / (margen + (\beta - \alpha))$, ... FROM tabla WHERE F_t=t AND $a' \leq b$ AND $\beta > d$ UNION SELECT ..., <i>etiqueta</i> , 1, ... FROM tabla WHERE F_t=t AND $\gamma \geq d$ AND $\beta \leq d$
INTER VALO[r,s]	 <p>$a = d - margen * (1 - u)$ $b = d + margen * (1 - u)$</p>	SELECT ..., <i>intervalo</i> , $(F_A - (d - margen)) / margen$, ... FROM tabla WHERE F_t=5 AND $F_A \geq a$ AND $F_A < d$ UNION SELECT ..., <i>intervalo</i> , $((d + margen) - F_1) / margen$, ... FROM tabla WHERE F_t=5 AND $F_1 > d$ AND $F_1 \leq b$ UNION SELECT ..., <i>intervalo</i> , 1, ... FROM tabla WHERE F_t=5 AND d BETWEEN F_1 AND F_A

Tabla 5.17: Selección usando FEQ(F, "aprox(d)", u) sobre atributos de Tipo 2

FEQ(F, "aprox(d)", u)		
TIPO de DATO	CALCULO	SENTENCIA QUE CALCULA
APROX(d')	<p> $c(x) \neq 0$ $\forall d' \in [d - 2 * margen * (1 - u),$ $s + 2 * margen * (1 - u)]$ </p>	<p>SELECT ..., <i>aprox</i>, $1.5+(F_{-1}-d)/(2*margen),...$ FROM <i>tabla</i> WHERE $F_{-t}=6$ AND $F_{-1} \geq (d - margen) - 2*margen*(1-u)$ AND $F_{-1} \leq (d - margen)$ UNION SELECT ..., <i>aprox</i>, $0.5+(d-F_{-1})/(2*margen),...$ FROM <i>tabla</i> WHERE $F_{-t}=6$ AND $F_{-1} > (d - margen)$ AND $F_{-1} \leq (d - margen) + 2 * margen * (1 - u)$</p>

Tabla 5.18: Continuación de la tabla 5.17

minio: $\mu_{igual_a}(d, n) = s_n(d)$ para comparación con datos de tipo SIMPLE, donde $s_n(d)$ son los grados de semejanza de los valores del dominio D con el valor constante n , y $\mu_{igual_a}(\tilde{d}, n) = \sup_{d \in D} \min\{s_n(d), \pi_{\tilde{d}}(d)\}$ para datos de tipo distribución de posibilidad. (Tabla 5.19)

- Valor <constante> de tipo DISTR. POS. sin relación de semejanza en el dominio: $\mu_{igual_a}([d_1/w_1, \dots, d_s/w_s], d) = d_i$ si $\delta(w_i, d) = 1$; 0 en caso contrario, para comparación con datos de tipo SIMPLE y $\mu_{igual_a}([d_1/w_1, \dots, d_s/w_s], \tilde{d}) = \sup_{(d,d') \in \{w_1, \dots, w_s\} \times D} \min\{\pi_{\tilde{d}}(d), \pi_{\tilde{d}'}(d')\}$ para datos de tipo distribución de posibilidad. (Tabla 5.20)
- Valor <constante> de tipo DISTR. POS. con relación de semejanza en el dominio: $\mu_{igual_a}([d_1/w_1, \dots, d_s/w_s], d') = \sup_{(d,d') \in \{w_1, \dots, w_s\} \times D} \min\{s(d, d'), \pi_{\tilde{d}}(d), 1/d'\}$, para comparación con datos de tipo SIMPLE y $\mu_{igual_a}([d_1/w_1, \dots, d_s/w_s], \tilde{d}') = \sup_{(d,d') \in \{w_1, \dots, w_s\} \times D} \min\{s(d, d'), \pi_{\tilde{d}}(d), \pi_{\tilde{d}'}(d')\}$. (Tabla 5.20)

2. Para construir las sentencias clásicas se recurre al empleo de las funciones DECODE, (cuyo funcionamiento quedó descrito), GREATEST, LEAST y NVL. La función GREATEST($expr_1, \dots, expr_n$) se evalúa como $expr_j$ siendo éste el máximo de todas las $expr_i, i = 1, \dots, n$. Con una estructura similar la función LEAST entrega el valor mínimo. Por último, NVL(*atrib*, k) devuelve el valor k para cada tupla que adopte un valor NULL, (en el sentido adoptado por el sistema anfitrión) ,en el atributo *atrib*, esto puede ser necesario

FEQ(F,n,u)	
TIPO de DATO	SENTENCIA QUE CALCULA
Sin Relación de Semejanza en el Dominio de F	
UNKNOWN	SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0
UNDEFINED	No es preciso recuperar tuplas de esta partición
NULL	SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2
SIMPLE	SELECT ..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1=n
DISTR. POS	SELECT ..., <i>distrib</i> , GREATEST(DECODER(F_1,n,F_P1,0), ..., DECODE(F_r,n, F_Pr,0)), ... FROM tabla WHERE F_t=4 AND n IN (F_1, ..., F_r) AND GREATEST(NVL(F_P1,0), ..., NVL(F_Pr,0)) ≥ u
Con Relación de Semejanza en el Dominio de F	
UNKNOWN	SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0
UNDEFINED	No es preciso recuperar tuplas de esta partición
NULL	SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2
SIMPLE	SELECT ..., F_1, DECODE(F_1,v_1,s_1, ..., v_m,s_m), ... FROM tabla WHERE F_t=3 AND F_1 IN (v_1, ..., v_m)
DISTR. POS	SELECT ..., <i>distrib</i> , GREATEST(LEAST(DECODER(F_1,n,F_P1,0), DECODE(F_1, v_1,s_1, ..., v_s,s_s,0)), ..., LEAST(DECODER(F_r,n,F_Pr,0), DECODE(F_r,v_1,s_1, ..., v_s,s_s,0))), ... FROM tabla WHERE F_t=4 AND GREATEST(LEAST(DECODER(F_1, n,F_P1,0), DECODE(F_1,v_1,s_1, ..., v_s,s_s,0)), ..., LEAST(DECODER(F_r,n,F_Pr,0), DECODE(F_r,v_1,s_1, ..., v_s,s_s,0))) ≥ u

Tabla 5.19: Selección usando FEQ(F, “simple”, u) sobre atributos de Tipo 3

porque para valores NULL algunas funciones se evalúan como NULL.

3. El identificador *distrib* hace referencia a la forma en que se reconstruye la presentación de una distribución de posibilidad a partir de los atributos que almacenan sus parámetros, (ver tabla 5.2), de acuerdo con lo expuesto en el punto 3 del algoritmo 5.1.
4. Para la tabla 5.19, en la lista $v_1, c_1, \dots, v_s, c_s$, el valor c_i representa el grado de semejanza entre la constante n , aportada en la selección, y el valor $v_i, i = 1, \dots, s$. Dichos grados son extraídos por el FCP de la FMB, concretamente de la tabla FUZZY_NEARNESS_DEF. En la tabla 5.20 se emplea, aquella fracción de la relación de semejanza establecida sobre el dominio que subyace al atributo difuso F que afecta a la distribución de posibilidad pro-

FEQ(F, “[$d_1/w_1, \dots, d_s/w_s$]”, u)	
TIPO de DATO	SENTENCIA QUE CALCULA
Sin Relación de Semejanza en el Dominio de F	
UNKNOWN	SELECT ..., 'UNKNOWN', max{ d_1, \dots, d_s }, ... FROM tabla WHERE F _t =0
UNDEFINED	No es preciso recuperar tuplas de esta partición
NULL	SELECT ..., 'NULL', max{ d_1, \dots, d_s }, ... FROM tabla WHERE F _t =2
SIMPLE	SELECT ..., F _l , DECODE(F _l , $w_1, d_1, \dots, w_s, d_s$), ... FROM tabla WHERE F _t =3 AND DECODE(F _l , $w_1, d_1, \dots, w_s, d_s$) ≥ u
DISTR. POS	SELECT ..., <i>distrib</i> , GREATEST(LEAST(DECODE(F _l , $w_1, d_1, \dots, w_s, d_s$), NVL(F _{P1} , 0)), ..., LEAST(DECODE(F _r , $w_1, d_1, \dots, w_s, d_s$), NVL(F _{Pr} , 0))) ... FROM tabla WHERE F _t =4 AND GREATEST(LEAST(DECODE(F _l , $w_1, d_1, \dots, w_s, d_s$), NVL(F _{P1} , 0)), ..., LEAST(DECODE(F _r , $w_1, d_1, \dots, w_s, d_s$), NVL(F _{Pr} , 0))) ≥ u
Con Relación de Semejanza en el Dominio de F	
UNKNOWN	SELECT ..., 'UNKNOWN', max{ d_1, \dots, d_s }, ... FROM tabla WHERE F _t =0
UNDEFINED	No es preciso recuperar tuplas de esta partición
NULL	SELECT ..., 'NULL', max{ d_1, \dots, d_s }, ... FROM tabla WHERE F _t =2
SIMPLE	SELECT ..., F _l , GREATEST(LEAST(DECODE(F _l , $v_{11}, s_{11}, \dots, v_{1n}, s_{1n}$, 0), d_1), ..., LEAST(DECODE(F _l , $v_{s1}, s_{s1}, \dots, v_{sn}, s_{sn}$, 0), d_s)), ... FROM tabla WHERE F _t =3 AND GREATEST(LEAST(DECODE(F _l , $v_{11}, s_{11}, \dots, v_{1n}, s_{1n}$, 0), d_1), ..., LEAST(DECODE(F _l , $v_{s1}, s_{s1}, \dots, v_{sn}, s_{sn}$, 0), d_s)) ≥ u
DISTR. POS	SELECT ..., <i>distrib</i> , GREATEST(LEAST(DECODE(F _l , $v_{11}, s_{11}, \dots, v_{1n}, s_{1n}$, 0), NVL(F _{P1} , 0), d_1), ..., LEAST(DECODE(F _l , $v_{s1}, s_{s1}, \dots, v_{sn}, s_{sn}$, 0), NVL(F _{P1} , 0), d_s)), ..., LEAST(DECODE(F _r , $v_{11}, s_{11}, \dots, v_{sn}, s_{sn}$, 0), NVL(F _{Pr} , 0), d_1), ..., LEAST(DECODE(F _r , $v_{s1}, s_{s1}, \dots, v_{sn}, s_{sn}$, 0), NVL(F _{Pr} , 0), d_s)), ... FROM tabla WHERE F _t =4 AND GREATEST(LEAST(DECODE(F _l , $v_{11}, s_{11}, \dots, v_{sn}, s_{sn}$, 0), NVL(F _{P1} , 0), d_1), ..., LEAST(DECODE(F _l , $v_{s1}, s_{s1}, \dots, v_{sn}, s_{sn}$, 0), NVL(F _{P1} , 0), d_s)), ..., LEAST(DECODE(F _r , $v_{11}, s_{11}, \dots, v_{sn}, s_{sn}$, 0), NVL(F _{Pr} , 0), d_1), ..., LEAST(DECODE(F _r , $v_{s1}, s_{s1}, \dots, v_{sn}, s_{sn}$, 0), NVL(F _{Pr} , 0), d_s)) ≥ u

Tabla 5.20: Selección usando FEQ(F, “distr. posib.”, u) sobre atributos de Tipo 3

porcionada en la selección. La forma en que se expresa queda determinada por una colección de listas de constantes $v_{i1}, s_{i1}, \dots, v_{in}, s_{in}, i = 1, \dots, s$ donde cada lista expresa los grados de similitud de cada valor de posibilidad de la distribución, $[d_1/w_1, \dots, d_s/w_s]$, con el resto de los n valores de dominio soportados por el atributo F. Todos estos datos se extraen de la FMB en el momento de analizar la selección, por lo que son constantes en el momento de evaluar la consulta en el sistema anfitrión.

5. La expresión $\max\{d_1, \dots, d_s\}$ es un valor constante que representa el máximo de los grados de posibilidad de la distribución, $[d_1/w_1, \dots, d_s/w_s]$, proporcionada en la selección.

- *SELECCIÓN BASADA EN EL COMPARADOR “MAYOR O IGUAL”*

Para notar dicha operación emplearemos la expresión $\text{FGEQ}(\langle \text{campo_difuso} \rangle, \langle \text{constante} \rangle, \langle \text{umbral} \rangle)$. Donde:

- $\langle \text{campo_difuso} \rangle$ es el identificador de un atributo de tipo 1 ó 2. (En principio, FIRST no soporta esta operación sobre atributos de tipo 3)
- $\langle \text{constante} \rangle$ es el identificador de un dato constante perteneciente a alguno de los tipos descritos en 5.2.2 y compatible con los tipos soportados por $\langle \text{campo_difuso} \rangle$.
- $\langle \text{umbral} \rangle$ es el identificador de un valor umbral, (etiqueta o número), que cualifica la comparación.

Para resolver esta selección es preciso hacer uso de la expresión 5.3 de la página 165. El proceso para trasladar una selección basada en este comparador a una consulta clásica es similar al descrito para el operador FEQ. Por este motivo nos detendremos únicamente en los aspectos particulares del operador FGEQ. Puesto que la comparación se realiza con un valor constante, a nivel teórico, el proceso a seguir consiste en descomponer la expresión 5.3 en dos partes: una dependiente de los valores que vaya asumiendo el atributo y otra construida mediante la composición del “*comparador extendido*” \geq con el valor constante difuso, k , aportado en la consulta. Con esta última composición obtenemos la distribución de posibilidad que representa el concepto difuso “*mayor o igual que la constante k*”, (la figura 5.21 muestra la composición para los tipos de datos

TIPO de DATO	FUNCIÓN DE PERTENENCIA	TIPO de DATO	FUNCIÓN DE PERTENENCIA
CRISP	<p>μ "mayor o igual que n" (x)</p>	INTER VALO[r,s]	<p>μ "mayor o igual que [r,s]" (x)</p>
LABEL/ DIFUSO	<p>μ "mayor o igual que [$\alpha,\beta,\gamma,\delta$]" ($x$)</p> <p>$a = u * (\beta - \alpha) + \alpha$</p>	APROX(d)	<p>μ "mayor o igual que aprox(d)" (x)</p> <p>$a = d - margen * (1 - u)$</p>

Tabla 5.21: Composición del comparador “mayor o igual” con los diferentes tipos

a los que se aplica). A continuación lo único que resta es encontrar el grado en que cada dato difuso asumido por el atributo consultado es compatible con el anterior concepto. Las expresiones SQL que resuelven la selección difusa en términos clásicos adoptan una sistemática similar a la descrita para la selección basada en el operador “igual a”. Nuevamente distinguiremos el tratamiento para cada tipo de atributo (1 y 2):

- **Atributos Tipo 1.** Expresaremos las sentencias SQL que resuelven la selección basada en el operador FGEQ(<campo_difuso> ,<constante> ,<umbral>) para los diferentes tipos que puede asumir el parámetro <constante>.
 - Si <constante> es del tipo “crisp”, (d), puesto que el operador FGEQ es difuso, entenderemos que se pregunta por las tuplas que poseen un valor en el atributo sometido a consulta “mayor o igual” que d , entendido éste como valor aproximado. Esto significa encontrar la sentencia SQL que, mediante el empleo del algoritmo 5.1, seleccione aquellas tuplas que presenten un valor compatible, en el sentido mostrado en la expresión 5.3, con la distribución mostrada en la segunda columna de la segunda fila de la tabla 5.21 que supere o iguale el umbral u establecido como tercer parámetro. El resultado, en el que, como muestra la citada figura, se distinguen dos particiones, viene expresado por la sentencia:

```
SELECT ..., F, (F - (d - margen))/margen, ... FROM tabla WHERE
F ≥ d - margen * (1 - u) AND F < d UNION SELECT ..., F, 1, ... FROM
tabla WHERE F ≥ d.
```

- Si <constante> es del tipo “*label*” o “*difuso*” con una distribución de posibilidad trapezoidal dada por los valores $[\alpha, \beta, \gamma, \delta]$ la sentencia que resuelve la selección en términos clásicos es:

```
SELECT ..., F, (F-α)/(β-α), ... FROM tabla WHERE F ≥ u * (β - α) + α
AND F < β UNION SELECT ..., F, 1, ... FROM tabla WHERE F ≥ β
```

- Si <constante> es del tipo “*intervalo*[*r,s*]” la expresión obtenida es:

```
SELECT ..., F, 1, ... FROM tabla WHERE F ≥ r
```

- Para <constante> del tipo “*aprox*(*n*)” el resultado es idéntico al obtenido para valor <constante> del tipo “*crisp*”

- **Atributos Tipo 2.** Debido a la variedad de tipos que puede soportar esta clase de atributos, las sentencias resultantes resultan mas complejas. No obstante el mecanismo seguido es similar al expuesto para la resolución del comparador “*igual a*” notado mediante el operador FEQ. Las sentencias SQL elaboradas por el FCP a partir del valor constante expresado en el comparador FGEO, se mostrarán clasificadas según el tipo dicho valor:

- <constante> de tipo “*crisp*”, (*n*). De acuerdo con lo expuesto en el algoritmo 5.1, es preciso expresar el cálculo de las tuplas que presentan, en el atributo implicado, un grado de compatibilidad con la condición “*mayor o igual que n*”, (fig. 5.21), superior al umbral *u* establecido. La sentencia que resuelve esta operación queda expresada como sigue:

```
SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0 UNION
SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2 UNION SE-
LECT ..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1 ≥ n UNION
SELECT ..., etiqueta, 1, ... FROM tabla WHERE F_t=t AND n ≤ γ
UNION SELECT ..., etiqueta, (δ-n)/(γ-delta), ... FROM tabla WHERE
F_t=t AND n > γ AND n ≤ b UNION SELECT ..., intervalo, 1, ...
FROM tabla WHERE F_t=5 AND F_4 ≥ n UNION SELECT ..., aprox,
1, ... FROM tabla WHERE F_t=6 AND F_1 ≥ n - margen UNION SE-
```

LECT ..., *aprox*, $(F_4 - n) / \text{margen}$ WHERE $F_t = 6$ AND $F_1 < n - \text{margen}$
AND $F_1 \geq n - \text{margen} * (2 - u)$

Para la obtención de los parámetros que configuran cada tipo de dato del atributo F sobre el que se opera, nos remitimos al esquema presentado en la columna 2 de la tabla 5.12. Así mismo, las consideraciones vertidas entonces sobre los parámetros α , β , γ , δ , t , a y b son aplicables a la sentencia mostrada arriba.

- **<constante>** de tipo “*difuso*”, $([\alpha, \beta, \gamma, \delta])$. En este caso es necesario precisar en una sentencia SQL el cálculo de la compatibilidad de los diferentes tipos de datos presentes en el atributo con la condición “*mayor o igual que $[\alpha, \beta, \gamma, \delta]$* ” cuya representación, para un umbral u , se muestra la primera columna de la segunda fila de la figura 5.21:

```
SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0 UNION SE-
LECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2 UNION SELECT
..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1 ≥ β UNION SE-
LECT ..., F_1, (F_1 - α)/(β - α), ... FROM tabla WHERE F_t=3 AND F_1
≥ a AND F_1 < β UNION SELECT ..., etiqueta,  $(\delta_i - \alpha) / ((\beta - \alpha) - (\gamma_i - \delta_i))$ , ...
FROM tabla WHERE F_t=t AND  $b_i \geq a$  AND  $\gamma_i < \beta$  UNION SELECT
..., etiqueta, 1, ... FROM tabla WHERE F_t=t AND  $\gamma_i \geq \beta$  UNION SE-
LECT ..., intervalo, 1, ... FROM tabla WHERE F_t=5 AND F_4 UNION
SELECT ..., intervalo,  $(F_4 - \alpha) / (\beta - \alpha)$ , ... FROM tabla WHERE F_t=5
AND  $F_4 \geq a$  AND  $F_4 < \beta \geq \beta$  UNION SELECT ..., aprox, 1, ...
FROM tabla WHERE F_t=6 AND  $F_1 \geq \beta - \text{margen}$  UNION SELECT
..., aprox,  $(F_4 - \alpha) / (\beta - \alpha + \text{margen})$ , ... FROM tabla WHERE F_t=6 AND
 $F_1 \geq a - \text{margen}$  AND  $F_1 < \beta - \text{margen}$ 
```

La sentencia mostrada emplea las convenciones establecidas para el operador FEQ sobre constantes de tipo “*difuso*”. Las tablas 5.13 y 5.14 muestran una notación compatible con los parámetros empleados en la sentencia anterior.

- **<constante>** de tipo “*intervalo*”, $([r, s])$. La sentencia que realiza el cálculo prescrito por el operador difuso es:

```
SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0 UNION SE-
LECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2 UNION SELECT
..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1 ≥ r UNION SE-
```



```
LECT ..., etiqueta, 1, ... FROM tabla WHERE F_t=t AND  $\gamma \geq r$  UNION
SELECT ..., etiqueta,  $(r-\delta)/(\gamma-\delta)$ , ... FROM tabla WHERE F_t=t AND
 $b \geq r$  AND  $\gamma < r$  UNION SELECT ..., intervalo, 1, ... FROM tabla
WHERE F_t=5 AND  $F_A \geq r$  UNION SELECT ..., aprox, 1, ... FROM
tabla WHERE F_t=6 AND  $F_1 \geq r$  UNION SELECT ..., aprox,  $1-r/F_A$ ,
... FROM tabla WHERE F_t=6 AND  $F_1 \geq r - margen * (2 - u)$  AND
 $F_1 < r - margen$ 
```

Las mismas consideraciones hechas sobre los resultados resumidos en las tablas 5.15 y 5.16 son aplicables al caso que nos ocupa.

- <constante> de tipo “*aprox*”, (*aprox*(*d*)). Teniendo en cuenta la distribución de la tabla 5.21 que representa el concepto “*mayor o igual que aprox*(*d*)”, la sentencia que resulta de aplicarlo sobre los diferentes tipos de datos que puede soportar el atributo F es la siguiente:

```
SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0 UNION SE-
LECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2 UNION SELECT
..., F_1, 1, ... FROM tabla WHERE F_t=3 AND  $F_1 \geq d$  UNION SE-
LECT ..., F_1,  $1+(F_1-d)/margen$ , ... FROM tabla WHERE F_t=3 AND
 $F_1 \geq d - margen * (1 - u)$  AND  $F_1 < d$  UNION SELECT ..., etiqueta, 1,
... FROM tabla WHERE F_t=t AND  $\gamma \geq d$  UNION SELECT ..., etiqueta,
 $((\delta-(d-margen))/(margen-(\gamma-\delta)))$ , ... FROM tabla WHERE F_t=t AND  $b' \geq a$  AND  $\gamma < d$  UNION SELECT ..., intervalo, 1, ... FROM tabla
WHERE F_t=5 AND  $F_A \geq d$  UNION SELECT ..., intervalo,  $(F_A-(d-
margen))/margen$ , ... FROM tabla WHERE F_t=5 AND  $F_A \geq a$  AND  $F_A < d$  UNION SELECT ..., aprox, 1, ... FROM tabla WHERE F_t=6 AND
 $F_1 \geq d - margen$  UNION SELECT ..., aprox,  $1.5+(F_1-d)/(2*margen)$ ,
... FROM tabla WHERE F_t=6 AND  $F_1 \geq (d - margen) - 2 * margen * (1 - u)$  AND  $F_1 < (d - margen)$ 
```

El significado de los parámetros envueltos en esta sentencia está proporcionado por las representaciones mostradas en las tablas 5.21, 5.17, 5.18 y por las consideraciones enunciadas respecto a las dos últimas tablas.

- **SELECCIÓN BASADA EN EL COMPARADOR “MENOR QUE”**

Para notar dicha operación emplearemos la expresión FLT(<campo_difuso>, <cons-

tante>, <umbral>). Donde:

- <campo_difuso> es el identificador de un atributo de tipo 1 ó 2. (En principio FIRST no soporta esta operación sobre atributos de tipo 3)
- <constante> es el identificador de un dato constante perteneciente a alguno de los tipos descritos en 5.2.2 y compatible con los tipos soportados por <campo_difuso>.
- <umbral> es el identificador de un valor (etiqueta o número) umbral que cualifica la comparación.

TIPO de DATO	FUNCIÓN DE PERTENENCIA	TIPO de DATO	FUNCIÓN DE PERTENENCIA
CRISP	<p>μ "menor que n" (x)</p>	INTER VALO[r,s]	<p>μ "menor que [r,s]" (x)</p>
LABEL/ DIFUSO	<p>μ "menor que [α,β,γ,δ]" (x)</p> <p>$a = \beta - u * (\beta - \alpha)$</p>	APROX(d)	<p>μ "menor que aprox(d)" (x)</p> <p>$a = d - u * \text{margin}$</p>

Tabla 5.22: Composición del comparador “menor que” con los diferentes tipos

La resolución de este tipo de selección está construida sobre la expresión 5.8 de la página 166. En este caso, la obtención del concepto “menor que k”, con k siendo una constante difusa o no, se realiza a partir del complemento de la distribución de posibilidad obtenida para el concepto “mayor o igual que k”. Según esto, partiendo de la tabla 5.21, las funciones de pertenencia calculadas para cada tipo se clasifican en la tabla 5.22. De igual forma que en el caso del operador FGEQ el proceso a seguir consiste en construir una sentencia clásica en SQL que permita obtener aquellas tuplas que, para el atributo considerado, presentan un grado de compatibilidad con la función de pertenencia que modela el concepto “menor que k” superior o igual al

umbral aportado en la selección. Los resultados obtenidos se clasificaran atendiendo al tipo de atributo considerado:

- **Atributos Tipo 1.** Remitiendonos a la tabla 5.22, expresaremos las sentencias SQL que resuelven la selección basada en el operador $FGEQ(\langle \text{campo_difuso} \rangle, \langle \text{constante} \rangle, \langle \text{umbral} \rangle)$ para los diferentes tipos que puede asumir el parámetro $\langle \text{constante} \rangle$:
 - Para $\langle \text{constante} \rangle$ del tipo “*crisp*”, (d). La sentencia que devuelve las tuplas que satisfacen, con el umbral exigido, la condición impuesta al atributo considerado de ser “*aproximadamente menor que d*” es:


```
SELECT ..., F, 1, ... FROM tabla WHERE F ≤ d - margen SELECT
..., F, (d-F)/margen, ... FROM tabla WHERE F > d - margen AND F
≤ d - u * margen
```
 - Si $\langle \text{constante} \rangle$ es del tipo “*label*” o “*difuso*” con una distribución de posibilidad trapezoidal dada por los valores $[\alpha, \beta, \gamma, \delta]$ la sentencia que resuelve la selección en términos clásicos es:


```
SELECT ..., F, 1, ... FROM tabla WHERE F ≤ α UNION SELECT ...,
F, (β-F)/(β-α), ... FROM tabla WHERE F > α AND F ≤ β - u * (β - α
```
 - Para $\langle \text{constante} \rangle$ del tipo “*intervalo[r,s]*” la expresión obtenida es:


```
SELECT ..., F, 1, ... FROM tabla WHERE F < r
```
 - Para $\langle \text{constante} \rangle$ del tipo “*aprox(n)*” el resultado es idéntico al obtenido para valor $\langle \text{constante} \rangle$ del tipo “*crisp*”
- **Atributos Tipo 2.** Mostraremos las sentencias elaboradas por el FCP para resolver la selección sobre este tipo de atributos clasificadas según el tipo de la constante expresada en la operación:
 - $\langle \text{constante} \rangle$ de tipo “*crisp*”, (n). De acuerdo con lo expuesto en el algoritmo 5.1, es preciso expresar el cálculo de las tuplas que presentan, en el atributo implicado, un grado de compatibilidad con la condición “*menor que n*”, (fig. 5.22), superior al umbral u establecido. La sentencia que resuelve esta operación queda expresada como sigue:

```

SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0 UNION
SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2 UNION SELECT
..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1 < n UNION SELECT
..., etiqueta, 1, ... FROM tabla WHERE F_t=t AND n ≥ β UNION SE-
LECT ..., etiqueta, (n-α)/(β-α), ... FROM tabla WHERE F_t=t AND
n ≥ a AND n < β UNION SELECT ..., intervalo, 1, ... FROM tabla
WHERE F_t=5 AND F_1 < n UNION SELECT ..., aprox, 1, ... FROM
tabla WHERE F_t=6 AND F_1 < n - margen UNION SELECT ..., aprox,
(F_4-n)/margen WHERE F_t=6 AND F_1 ≤ n - margen * (1 - u) AND F_1
> n - margen

```

Para la obtención de los parámetros que configuran cada tipo de dato del atributo F sobre el que se opera, nos remitimos al esquema presentado en la columna 2 de la tabla 5.12. Así mismo, las consideraciones vertidas entonces sobre los parámetros α , β , γ , δ , t , a y b son aplicables a la sentencia mostrada arriba.

- <constante> de tipo “difuso”, $([\alpha, \beta, \gamma, \delta])$. La sentencia SQL que lleva a cabo la operación a estudio ha de efectuar el cálculo de la compatibilidad de los diferentes tipos de datos presentes en el atributo con la condición “menor que $[\alpha, \beta, \gamma, \delta]$ ” cuya representación, para un umbral u , se muestra la primera columna de la segunda fila de la figura 5.22:

```

SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0 UNION
SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2 UNION SELECT
..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1 < α UNION SELECT
..., F_1, (β-F_1)/(β-α), ... FROM tabla WHERE F_t=3 AND F_1 > α
AND F_1 ≤ β - u * (β - α) UNION SELECT ..., etiqueta, 1, ... FROM
tabla WHERE F_t=t AND β_i ≤ α UNION SELECT ..., etiqueta, (β-
α_i)/((β-α)+(β_i-α_i)), ... FROM tabla WHERE F_t=t AND a_i > α AND
a_i ≤ β - u * (β - α) UNION SELECT ..., intervalo, 1, ... FROM tabla
WHERE F_t=5 AND F_1 ≤ α UNION SELECT ..., intervalo, (β-F_1)/(β-
α), ... FROM tabla WHERE F_t=5 AND F_1 > α AND F_4 ≤ a UNION
SELECT ..., aprox, 1, ... FROM tabla WHERE F_t=6 AND F_1 ≤ α -
margen UNION SELECT ..., aprox, (β-F_1)/(β-α+margen), ... FROM
tabla WHERE F_t=6 AND F_1 ≤ u * (α - β - margen) + β AND F_1

```

$> \alpha - margen$

- **<constante>** de tipo “*intervalo*”, $([r,s])$. La sentencia que realiza el cálculo prescrito por el operador difuso es:

```
SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0 UNION SE-
LECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2 UNION SELECT
..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1 < r UNION SE-
LECT ..., etiqueta, 1, ... FROM tabla WHERE F_t=t AND r > beta UNION
SELECT ..., etiqueta, (r-alpha)/(beta-alpha), ... FROM tabla WHERE F_t=t AND
u * (beta - alpha) + alpha <= r AND r < beta UNION SELECT ..., intervalo, 1, ...
FROM tabla WHERE F_t=5 AND F_1 < r UNION SELECT ..., aprox, 1, ...
FROM tabla WHERE F_t=6 AND F_1 < r - margen UNION SELECT
..., aprox, (beta-F_1)/(beta-alpha+margen), ... FROM tabla WHERE F_t=6 AND
F_1 < r - u * margen AND F_1 > r - margen
```

- **<constante>** de tipo “*aprox*”, $(aprox(d))$. Teniendo en cuenta la distribución de la tabla 5.22 que representa el concepto “*menor que aprox(d)*”, la sentencia que resulta de aplicarlo sobre los diferentes tipos de datos que puede soportar el atributo F es la siguiente:

```
SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0 UNION
SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2 UNION SELECT
..., F_1, 1, ... FROM tabla WHERE F_t=3 AND F_1 < d-margen UNION
SELECT ..., F_1, (d-F_1)/margen, ... FROM tabla WHERE F_t=3 AND
F_1 > d-margen AND F_1 < d-u*margen UNION SELECT ..., etiqueta,
1, ... FROM tabla WHERE F_t=t AND F_2 < d - margen UNION SE-
LECT ..., etiqueta, (d-alpha)/(beta-alpha+margen), ... FROM tabla WHERE F_t=t
AND beta > d - margen AND u * (beta - alpha) + alpha < d - u * margen UNION
SELECT ..., intervalo, 1, ... FROM tabla WHERE F_t=5 AND F_1
< d - margen UNION SELECT ..., intervalo, (d-F_1)/margen, ... FROM
tabla WHERE F_t=5 AND F_1 > d - margen AND F_1 < d - u * margen
UNION SELECT ..., aprox, 1, ... FROM tabla WHERE F_t=6 AND
F_1 < d - margen UNION SELECT ..., aprox, (d-F_1)/(2 * margen),
... FROM tabla WHERE F_t=6 AND F_1 > d - 2 * margen AND F_1
> d - 2 * u * margen
```

- *SELECCIÓN BASADA EN EL COMPARADOR “MENOR O IGUAL” Y EN EL COMPARADOR “MAYOR QUE”*

El objetivo perseguido en la exposición de este subapartado no es sino ilustrar algunos aspectos del proceso de traducción de una operación de selección difusa a un conjunto de sentencias clásicas. Por otro lado, los factores relativos a la implementación pueden sufrir ligeras modificaciones para adaptarlos a una plataforma u otra, con lo que las sentencias mostradas a lo largo de este subapartado poseen un carácter meramente indicativo, siendo preciso adaptarlas a cada caso concreto. En definitiva, lo que se pretende demostrar es que es posible, bajo ciertos criterios de representación, resolver las operaciones de manipulación difusas mediante el empleo de sentencias SQL clásicas.

Las sentencias que desencadenan la selección basada en el comparador “*menor o igual*” presentan una sintaxis simétrica con respecto a la mostrada para la selección basada en el comparador “*mayor o igual*”. Consideramos que los aspectos principales envueltos en el proceso seguido por el FCP para traducir cada modalidad de comparación a expresiones clásicas han sido suficientemente ilustrados, por este motivo no vamos a abundar en la expresión de las sentencias calculadas para este tipo de selección.

Lo mismo se puede decir sobre la selección basada en el comparador “*mayor que*” en relación al carácter simétrico presentado con respecto al comparador “*menor que*”.

- *SELECCIÓN BASADA EN COMPARADORES ESPECIALES*

Las diferentes modalidades de selecciones revisadas en los puntos anteriores presentan un carácter uniforme en cuanto al tratamiento de los tipos de datos “*especiales*”, entendiéndose por tales a los datos de tipo UNKNOWN, UNDEFINED y NULL. Dicho tratamiento, que se deriva de la representación posibilística adoptada para los mismos, recupera en todos los casos las tuplas que poseen un valor UNKNOWN o NULL para el atributo objeto de evaluación. En ningún caso se recuperan las tuplas que poseen un valor UNDEFINED para el atributo sobre el que se articula la consulta. Es evidente que se precisa algún tipo de operador que ponga de relieve la condición especial de este tipo de datos. Para ello FIRST contempla la existencia de una serie de operadores que posibilitan seleccionar o excluir expresamente aquellas tuplas que posean un dato de tipo “*especial*” para el atributo considerado. De acuerdo con la notación empleada

para el resto de las selecciones es posible representarlos mediante la sintaxis:

- `UNK(<campo_difuso>)`. Devuelve cada tupla que presente un dato de tipo UNKNOWN como valor para el atributo notado como `<campo_difuso>`. El “*grado de compatibilidad*” es 1 para dicho atributo. La sentencia calculada por el FCP para resolver este operador en términos clásicos emplea únicamente el atributo `F_TYPE` para obtener las tuplas que lo satisfacen. Su expresión inmediata es:

```
SELECT ..., 'UNKNOWN', 1, ... FROM tabla WHERE F_t=0
```

- `NULL(<campo_difuso>)`. En este caso realiza el test para datos de tipo NULL en el atributo considerado, de igual forma, el “*grado de compatibilidad*” devuelto para dicho atributo es 1. Su expresión en términos clásicos es:

```
SELECT ..., 'NULL', 1, ... FROM tabla WHERE F_t=2
```

- `UNDEF(<campo_difuso>)`. Para las tuplas que presenten un dato de tipo UNDEFINED en el atributo considerado devolverá un “*grado de compatibilidad*” igual a 1. Se expresa mediante la consulta clásica:

```
SELECT ..., 'UNDEFINED', 1, ... FROM tabla WHERE F_t=1
```

Es evidente que estos operadores pueden formularse para encontrar las tuplas que no contienen en el atributo `<campo_difuso>` los respectivos datos especiales considerados. Esto se hará en combinación con el complemento o negación. De esta forma podemos recuperar las tuplas que no contengan un valor UNKNOWN, NULL o UNDEFINED, respectivamente, aplicando la negación de los anteriores operadores. El resultado se podría notar como `NOT UNK(<campo_difuso>)`, `NOT NULL(<campo_difuso>)` y `NOT UNDEF(<campo_difuso>)`, respectivamente. Con la serie de operadores introducidos es posible filtrar la presencia de datos UNKNOWN y NULL en los resultados de las consultas, así mismo, es posible detectar la presencia de valores UNDEFINED en los atributos.

• COMPOSICIÓN DE LA SELECCIÓN

De acuerdo con lo formulado en GEFRED, la “*selección difusa generalizada*” se compone mediante los conectivos \wedge (AND) y \vee (OR) empleando la “*intersección difusa generalizada*” y la “*unión difusa generalizada*”, respectivamente. Dichas operaciones algebraicas obtienen la intersección y la unión respectiva de las relaciones generalizadas

calculadas de cada condición atómica y evalúan el “*grado de compatibilidad*” resultante como el mínimo o el máximo, en cada caso, de los presentes en las selecciones atómicas de partida. El operador complemento \neg (NOT) obtiene, sobre el atributo considerado, el complemento a uno del correspondiente “*grado de compatibilidad*”. En lo que respecta al FCP, esto se traduce en aplicar el siguiente procedimiento sobre las restricciones parciales calculadas para cada condición atómica:

1. Obtener por separado las expresiones clásicas que permiten recuperar las tuplas que satisfacen cada condición atómica por encima del umbral establecido para cada una de ellas.
2. Combinar las expresiones de acuerdo con el nivel de precedencia habitual:
 - (a) ()
 - (b) NOT
 - (c) AND
 - (d) OR
 - (e) para igual precedencia de izquierda a derecha

La manera en que dos sentencias iniciales se combinan mediante un conector requiere considerar por separado cada parte de las sentencias SQL de partida. De esta forma, dadas dos “*selecciones difusas generalizadas*” S_1 y S_2 resueltas por sendas sentencias SQL clásicas, entonces, para cada una de las particiones prescritas por los pasos 1 y 2 del algoritmo 5.1, habremos de combinar la parte correspondiente a la cláusula WHERE de cada una de las sentencias mediante el empleo del conector clásico AND para resolver la “*intersección difusa generalizada*” y mediante el conector OR para resolver la “*unión difusa generalizada*”. Simultáneamente, hemos de combinar las respectivas cláusulas que en cada partición evalúan los “*grados de compatibilidad*” para cada selección de forma que, para cada tupla, obtengamos el mínimo de los grados con que se satisface cada selección atómica, para el caso de la intersección y el máximo para el caso de unión. Para la negación o complemento difuso existen dos maneras de encontrar la sentencia SQL que resuelve la selección:

- (a) Emplear los pasos descritos en el algoritmo 5.1 para encontrar la sentencia que evalúa las tuplas que satisfacen la condición complementaria con un umbral u . El proceso a seguir sería similar al descrito para los operadores analizados a lo largo de este subapartado. Habría que encontrar una expresión para cada tipo de atributo y dato.
- (b) Construir una sentencia compuesta por la diferencia relacional clásica entre la sentencia que calcula la selección para un umbral 0 con sus respectivos grados de compatibilidad complementados a 1 y la sentencia que devuelve las tuplas que satisfacen la condición sin complementar pero para un umbral $1-u$

La primera aproximación resulta más eficiente en ejecución puesto que solo requiere evaluar una vez las tuplas. Sin embargo precisa que se consideren nuevas sentencias para las comparaciones complementadas. La segunda permite el empleo de las sentencias ya introducidas a lo largo del subapartado, sin embargo requiere que se evalúen dos veces las tuplas. Evitaremos repetir el proceso que nos conduce al cálculo de estas nuevas sentencias ya que no aporta nada esencialmente nuevo.

5.3.4.3 Cardinalidad Absoluta y Relativa en FIRST

FIRST proporciona soporte para las definiciones de cardinalidad introducidas en el apartado 4.3.3 del estudio de GEFRED (pag. 113). En este subapartado mostraremos las pautas seguidas para expresar su cálculo, para ello introduciremos dos operadores FACOUNT y FRCOUNT que nos servirán para notar las cardinalidades difusas absoluta y relativa, respectivamente. Concretamente expresaremos estas operaciones de la siguiente forma:

- Cardinalidad Absoluta: `FACOUNT(<subconsulta>)` donde, `<subconsulta>` es la selección atómica o compuesta de la que se pretende evaluar la cardinalidad.
- Cardinalidad Relativa: `FRCOUNT(<subconsulta1>,<subconsulta2>)` con `<subconsulta1>` como la consulta cuya cardinalidad se pretende relativizar y `<subconsulta2>` es la consulta sobre la que se evalúa la cardinalidad relativa.

El cálculo que se emplea en FIRST para resolver ambas cardinalidades está basado en las expresiones 4.18 y 4.35 adoptadas en el apartado 4.3.3. A continuación mostramos el aspecto que podrían adoptar las sentencias SQL que se encargan de calcular cada una de las cardinalidades:

- Cardinalidad Absoluta:

```
SELECT SUM( $\mu_t$ ) FROM <lista_tablas> WHERE <subconsulta>
```

donde, μ_t es el grado expresado en las ec. 4.18, cuyo cálculo para composiciones basadas en la disyunción y en la conjunción, respectivamente, se realiza en función de las condiciones impuestas en <subconsulta>, mediante el empleo de los operadores GREATEST, (max), y LEAST, (min) de SQL.

- Cardinalidad Relativa:

```
SELECT SUM(LEAST( $\mu_{1t}, \mu_{2t}$ )/ $\mu_{2t}$ ) FROM <lista_tablas>
```

```
WHERE <subconsulta1> OR <subconsulta2>
```

donde μ_{1t} representa la expresión que calcula el grado para la relación <subconsulta1> y μ_{2t} expresa lo mismo con respecto a <subconsulta2>

Veamos como quedaría compuesta la sentencia mediante un ejemplo para cada tipo de cardinalidad.

Ejemplo 5.3 Partiendo de los supuestos establecidos en el ejemplo 5.1 (pag. 171), veamos como quedaría expresada y resuelta la siguiente consulta:

Consulta 5.9 “Determinar cuantos alumnos ha superado física (grado 0.8) y matemáticas (grado 0.8)”

Las mismas consideraciones vertidas entonces son válidas para el caso que nos ocupa. El concepto “mayor o igual que aprobado” se construye, a partir de la etiqueta “aprobado”, siguiendo el esquema que se muestra en la figura 5.21. Supondremos que operamos sobre la relación **Alumnos**(nombre, curso, cal_fisica, cal_mate) en la que cada atributo conteniendo las calificaciones de cada asignatura es de tipo 1, o sea que contiene valores “crisp”.

La consulta empleando la notación introducida podría expresarse:

```
SELECT FACOUNT( FGEQ( cal_fisica, "aprobado", 0.8) AND FGEQ( cal_matem,
"aprobado", 0.8)) FROM alumnos
```

La sentencia resultante tendría la forma:

```
SELECT SUM( LEAST( Ccal_fisica, Ccal_matem)) FROM alumnos WHERE cal_fisica
≥ 4.7 AND cal_matem ≥ 4.7
```

donde, C_{cal_fisica} , C_{cal_matem} son los atributos de compatibilidad que se obtienen para las condiciones “aprobar física” y “aprobar matemáticas”, respectivamente y SUM es una función de agregación de SQL que calcula la suma que toma la expresión argumento para todos los valores de la relación. Mediante el empleo de las funciones DECODE y SIGN, cuyo funcionamiento fue descrito anteriormemnte, podemos expresar el cálculo de dichos grados de compatibilidad de la siguiente manera:

```
DECODE( SIGN( cal_fisica-4.7), -1, 0, DECODE( SIGN( cal_fisica-5), -1, (cal_fisica-
5)/1.5, 1))
```

para cal_fisica y, para cal_matem,

```
DECODE( SIGN( cal_matem-4.7), -1, 0, DECODE( SIGN( cal_matem-5), -1, (cal-
matem-5)/1.5, 1))
```

obsérvese para ambos casos que, la combinación de las fuciones DECODE y SIGN permiten adaptar el cálculo de los grados de compatibilidad para cada valor considerado en la cláusula WHERE.

Por tanto, la sentencia SQL final que expresa el cálculo de la consulta ejemplo presenta la forma:

```
SELECT SUM( LEAST( DECODE( SIGN(cal_fisica-4.7), -1, 0, DECODE( SIGN(
cal_fisica-5), -1, (cal_fisica-5)/1.5, 1)), DECODE( SIGN( cal_matem-4.7), -1, 0, DE-
CODE( SIGN( cal_matem-5), -1, (cal_matem-5)/1.5, 1))) FROM alumnos WHERE
cal_fisica ≥ 4.7 AND cal_matem ≥ 4.7
```

■

El siguiente ejemplo ilustra el cálculo de la cardinalidad relativa en FIRST.

Ejemplo 5.4 En los mismos términos que los expuestos en el ejemplo anterior, vamos a estudiar los pasos que nos conducirán a expresar la sentencia SQL que resuleve la consulta:

Consulta 5.10 “Determinar que proporción de los alumnos que han superado la física (grado 0.8), han superado las matemáticas (grado 0.8)”

La consulta anterior podría notarse de la siguiente manera:

```
SELECT FRCOUNT( FGEEQ( cal_fisica, "aprobado", 0.8), FGEEQ( cal_matem,
"aprobado", 0.8)) FROM alumnos
```

Esta consulta pide determinar cuantos alumnos han superado las matemáticas y la física respecto de los que sólo han superado las matemáticas. Por tanto es preciso determinar, por un lado, la cardinalidad absoluta de la intersección dada por los alumnos que han superado ambas asignaturas y, por otro, la de los alumnos que sólo han superado las matemáticas. El cálculo de la primera cardinalidad mencionada se lleva a cabo mediante la sentencia encontrada para el ejemplo 5.3. La sentencia que calcula la segunda cardinalidad sería:

```
SELECT SUM( DECODE( SIGN( cal_matem-4.7), -1, 0, DECODE( SIGN( cal_ma-
tem-5), -1, (cal_matem-5)/1.5, 1))) FROM alumnos WHERE cal_matem ≥ 4.7
```

La cardinalidad relativa buscada vendrá expresada mediante el cociente de las cardinalidades absolutas anteriores:

```
SELECT SUM( LEAST( DECODE( SIGN(cal_fisica-4.7), -1, 0, DECODE( SIGN(
cal_fisica-5), -1, (cal_fisica-5)/1.5, 1)), DECODE( SIGN( cal_matem-4.7), -1, 0, DE-
CODE( SIGN( cal_matem-5), -1, (cal_matem-5)/1.5, 1))))/ SUM( DECODE( SIGN(
cal_matem-4.7), -1, 0, DECODE( SIGN( cal_matem-5), -1, (calmatem-5)/1.5, 1)))
FROM alumnos WHERE cal_fisica ≥ 4.7 OR cal_matem ≥ 4.7
```

En esta sentencia SQL podemos observar lo siguiente:

- Mediante el término que sigue a la cláusula WHERE se seleccionan todos los alumnos que han aprobado alguna o las dos asignaturas. Esto acota las tuplas (alumnos) que van a intervenir en el cómputo de la cardinalidad relativa, ninguna otra tupla interviene en ese cálculo.
- A través del término que sigue a la cláusula SELECT se realiza el cálculo de la cardinalidad relativa. Para ello, de acuerdo con el significado de la intersección en GEFRED, mediante la función LEAST seleccionamos para cada tupla el menor grado de compatibilidad que corresponde a las condiciones “aprobar la física” y “aprobar las matemáticas”.
- Las expresiones que calculan los grados de compatibilidad para las dos condiciones consideradas entregan un valor 0 para los valores de atributo que no satisfacen cada condición. Por este motivo, el numerador del término SELECT entregará un

valor $\text{LEAST}(0,0)=0$ para las tuplas que no pertenecen a la intersección difusa generalizada, con lo cual esas tuplas no aumentarán la cardinalidad absoluta representada por dicho numerador.

- El denominador del término que sigue a la cláusula SELECT, $\text{SUM}(\text{DECODE}(\text{SIGN}(\text{cal_matem}-4.7), -1 \dots))$, toma en cuenta únicamente las tuplas que satisfacen la condición “aprobar las matemáticas”, por este motivo la suma se realiza únicamente sobre los grados de compatibilidad calculados para dichas tuplas.
- De lo anteriormente expuesto, se comprueba que la sentencia mostrada expresa la sentencia SQL clásica que resuelve la cardinalidad relativa requerida por la consulta ejemplo según los términos establecidos por su definición en GEFRED.

■

5.3.4.4 La Cuantificación en FIRST

Como vimos en el subapartado 5.2.2.4 (pag. 168), FIRST contempla el empleo de cuantificadores difusos de las consultas. Para mostrar la forma en que puede resolver una cuantificación en cualquiera de sus modalidades, absoluta y relativa, adoptaremos para cada una de ellas la siguiente notación:

- Cuantificación Absoluta: $\text{FAQT}(\langle \text{subconsulta} \rangle, \langle \text{etiqueta} \rangle, \langle \text{umbral} \rangle)$ donde, $\langle \text{subconsulta} \rangle$ es la selección atómica o compuesta que se pretende cuantificar, $\langle \text{etiqueta} \rangle$ es una etiqueta lingüística identificada en la tabla FUZZY_OBJECT_LIST y definida en la tabla FUZZY_QUANTIFIERS_DEF como una distribución de posibilidad trapezoidal sobre un dominio que representa la cardinalidad de la subconsulta, $\langle \text{umbral} \rangle$ es un valor que establece el umbral mínimo a satisfacer por la cuantificación.
- Cuantificación Relativa: $\text{FRQT}(\langle \text{subconsulta1} \rangle, \langle \text{subconsulta2} \rangle, \langle \text{etiqueta} \rangle, \langle \text{umbral} \rangle)$ donde $\langle \text{subconsulta1} \rangle$ es la consulta cuya cardinalidad se pretende relativizar, $\langle \text{subconsulta2} \rangle$ es la consulta sobre la que se evalúa la cardinalidad relativa, $\langle \text{etiqueta} \rangle$ identificada en la tabla FUZZY_OBJECT_LIST como un cuantificador relativo y $\langle \text{umbral} \rangle$ tiene el significado habitual.

Nótese que, según ha sido introducida, la cuantificación absoluta de una relación representa el grado en que la cardinalidad absoluta de dicha relación es compatible con una etiqueta lingüística representando una distribución de posibilidad (cuantificador absoluto). Por tanto se da la equivalencia:

$$\begin{aligned} \text{FAQT}(\langle \text{subconsulta} \rangle, \langle \text{etiqueta} \rangle, \langle \text{umbral} \rangle) = \\ \text{FEQ}(\text{FACOUNT}(\langle \text{subconsulta} \rangle), \langle \text{etiqueta} \rangle, \langle \text{umbral} \rangle) \end{aligned}$$

Para la cuantificación relativa se puede decir lo mismo:

$$\begin{aligned} \text{FRQT}(\langle \text{subconsulta1} \rangle, \langle \text{subconsulta2} \rangle, \langle \text{etiqueta} \rangle, \langle \text{umbral} \rangle) = \\ \text{FEQ}(\text{FRCOUNT}(\langle \text{subconsulta1} \rangle, \langle \text{subconsulta2} \rangle), \langle \text{etiqueta} \rangle, \langle \text{umbral} \rangle) \end{aligned}$$

De lo anteriormente expuesto, el proceso para resolver una cuantificación se reduce a resolver dos pasos:

1. Obtener la expresión que permite evaluar la cardinalidad (absoluta o relativa) de la/s subconsulta/s implicada/s.
2. Construir la sentencia que evalúa la “compatibilidad” de dicha cardinalidad con la *etiqueta* (cuantificador absoluto o relativo) por encima del *umbral* establecido.

Los siguientes ejemplos resuelven cuantificaciones absoluta y relativa, respectivamente.

Ejemplo 5.5 Basándonos en los mismos supuestos, veamos como se resuelve la siguiente variante de la consulta 5.9:

Consulta 5.11 *“Encuentra las asignaturas en las que han habido muchos (grado 0.7) alumnos aprobados (grado 0.8)”*

Para resolver esta consulta hemos de calcular la cardinalidad absoluta de los alumnos que han aprobado física y de los que han aprobado matemáticas y despues confrontar dichas cardinalidades con el cuantificador “muchos” que, en nuestro caso, queda modelado por la distribución de posibilidad que muestra la figura 5.14.

Con la notación introducida, la consulta anterior podría quedar expresada:

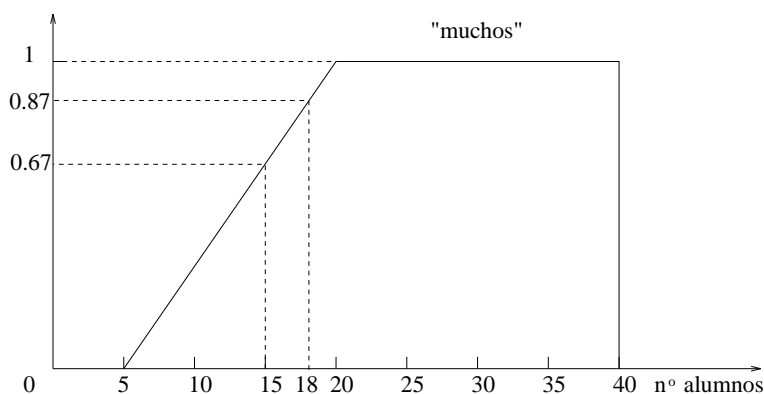


Figura 5.14: Etiqueta lingüística para el cuantificador “muchos”

`FAQT(<subconsulta>, ‘muchos’, 0.7)`

donde, <subconsulta> es “alumnos que han superado la física” y también “alumnos que han superado las matemáticas”. En términos de cardinalidades se puede expresar:

`FEQ(FACOUNT(<subconsulta>), ‘muchos’, 0.7)`

con el mismo significado para <subconsulta>. Para física resolvemos la cardinalidad absoluta mediante la sentencia:

```
SELECT SUM( DECODE( SIGN(cal_fisica-4.7), -1, 0, DECODE( SIGN( cal_fisica-5), -1, (cal_fisica-5)/1.5, 1))) FROM alumnos WHERE cal_fisica ≥ 4.7
```

y, para matemáticas

```
SELECT SUM( DECODE( SIGN(cal_matem-4.7), -1, 0, DECODE( SIGN( cal_matem-5), -1, (cal_matem-5)/1.5, 1))) FROM alumnos WHERE cal_matem ≥ 4.7
```

Ahora hemos de calcular para cada asignatura si la cardinalidad calculada es compatible con el cuantificador “muchos” en un grado mayor igual al <umbral>. Para física dicho cálculo se lleva a cabo mediante la sentencia:

```
SELECT 'FISICA' FROM dual WHERE 15.5 ≤ (SELECT SUM( DECODE( SIGN( cal_fisica-4.7), -1, 0, DECODE( SIGN( cal_fisica-5), -1, (cal_fisica-5)/1.5, 1))) FROM alumnos WHERE cal_fisica ≥ 4.7)
```

y, para matemáticas

```
SELECT 'MATEMATICAS' FROM dual WHERE 15.5 ≤ (SELECT SUM( DECODE( SIGN(cal_matem-4.7), -1, 0, DECODE( SIGN( cal_matem-5), -1, (cal_matem-5)/1.5, 1))) FROM alumnos WHERE cal_matem ≥ 4.7)
```

por lo que la sentencia global vendrá dada por la unión de las sentencias anteriores:

```
SELECT 'FISICA' FROM dual WHERE 15.5 ≤ (SELECT SUM( DECODE( SIGN( cal_fisica-4.7), -1, 0, DECODE( SIGN( cal_fisica-5), -1, (cal_fisica-5)/1.5, 1))) FROM alumnos WHERE cal_fisica ≥ 4.7) UNION SELECT 'MATEMATICAS' FROM dual WHERE 15.5 ≤ (SELECT SUM( DECODE( SIGN(cal_matem-4.7), -1, 0, DECODE( SIGN( cal_matem-5), -1, (cal_matem-5)/1.5, 1))) FROM alumnos WHERE cal_matem ≥ 4.7)
```

La sentencia anterior entrega, como resultado de su ejecución, los literales 'FISICA' y/o 'MATEMATICAS' si las condiciones expresadas en las respectivas cláusulas WHERE se satisfacen, y ello ocurre cuando las cardinalidades absolutas superan o igualan el valor determinado por el umbral establecido sobre el cuantizador “muchos”. La tabla “dual” es una tabla simbólica que permite realizar operaciones en las que no interviene ninguna relación.

■

El próximo ejemplo ilustra la resolución por parte del FCP de una cuantificación relativa.

Ejemplo 5.6 Vamos a cuantificar la consulta mostrada en el ejemplo 5.4, para ello emplearemos el cuantificador relativo “mayoría” cuya representación muestra la figura 5.15. La consulta a estudio queda expresada en los siguientes términos:

Consulta 5.12 *“Determinar si la mayoría (grado 0.9) de los alumnos que ha superado la física (grado 0.8) han superado las matemáticas (grado 0.8)”*

Según la notación introducida, la consulta podría notarse de la siguiente manera:

```
FRQT( FGEQ( cal_fisica, “aprobado”, 0.8), FGEQ( cal_mate, “aprobado”, 0.8), “mayoría”, 0.9)
```

o bien:

```
FEQ( FRCOUNT( FGEQ( cal_fisica, “aprobado”, 0.8), FGEQ( cal_mate, “aprobado”, 0.8)), “mayoría”, 0.9)
```

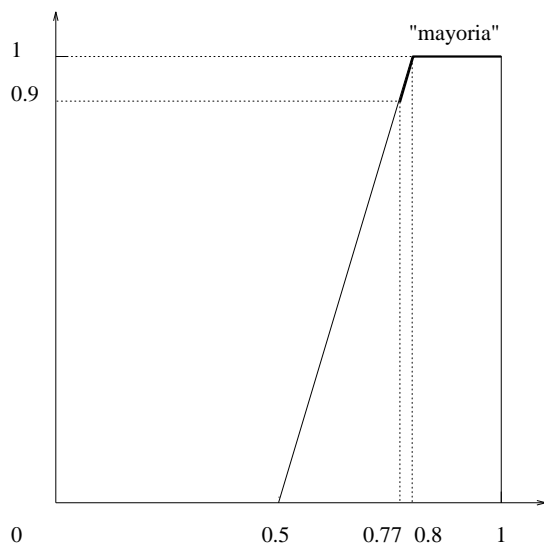



Figura 5.15: Etiqueta lingüística para el cuantificador relativo “mayoría”

El cálculo de la cardinalidad relativa se resuelve mediante la misma sentencia que la encontrada en el ejemplo 5.4:

```
SELECT SUM( LEAST( DECODE( SIGN(cal_fisica-4.7), -1, 0, DECODE( SIGN(
cal_fisica-5), -1, (cal_fisica-5)/1.5, 1)), DECODE( SIGN( cal_matem-4.7), -1, 0, DE-
CODE( SIGN( cal_matem-5), -1, (cal_matem-5)/1.5, 1))) / SUM( DECODE( SIGN(
cal_matem-4.7), -1, 0, DECODE( SIGN( cal_matem-5), -1, (calmatem-5)/1.5, 1)))
FROM alumnos WHERE cal_fisica ≥ 4.7 OR cal_matem ≥ 4.7
```

Por lo que únicamente precisamos encontrar la sentencia que evalúe la compatibilidad de esta cardinalidad con el cuantificador relativo “mayoría” y determine si la pregunta es cierta o falsa para el umbral establecido. Dicho umbral establece que para cardinalidades mayores o iguales a 0.77 se verifica la consulta, por tanto la sentencia que resuelve la consulta ejemplo es:

```
SELECT 'TRUE' FROM dual WHERE 0.77 ≤ (SELECT SUM( LEAST( DE-
CODE( SIGN(cal_fisica-4.7), -1, 0, DECODE( SIGN( cal_fisica-5), -1, (cal_fisica-5)/1.5,
1)), DECODE( SIGN( cal_matem-4.7), -1, 0, DECODE( SIGN( cal_matem-5), -1,
(cal_matem-5)/1.5, 1))) / SUM( DECODE( SIGN( cal_matem-4.7), -1, 0, DECODE(
```

`SIGN(cal_matem-5), -1, (cal_matem-5)/1.5, 1))) FROM alumnos WHERE cal_fisica \geq 4.7 OR cal_matem \geq 4.7)`

Si la cardinalidad relativa obtenida para la condición expresada supera o iguala el valor 0.77 entonces la ejecución de la consulta devuelve “TRUE” en caso contrario no devolverá nada lo que será interpretado como que no se satisface la condición de cuantificación establecida.

■

5.3.5 Ejemplo de Procesamiento de Consulta en FIRST

Para mostrar como opera el FCP vamos a detallar como se resuelve una consulta simple enunciada como sigue:

Consulta 5.13 *“Dame el NOMBRE, la DIRECCIÓN la EDAD, así como el grado en que se satisface la condición para el atributo EDAD, de aquellos empleados con una EDAD ”madura” (con un grado \geq 0.8)”*

Partimos de la tabla Empleados y de las etiquetas y demas elementos introducidos en el ejemplo ilustrado en el subapartado 5.3.3. Los pasos a seguir para resolver esta consulta son los siguientes:

1. Formular dicha consulta siguiendo una sintaxis determinada. Dicha sintaxis puede ser formulada como una extensión ”difusa” de algún lenguaje de consulta relacional. Por ejemplo, se podría pensar en un SQL ”difuso”. En esta memoria no vamos a introducir una sintaxis detallada de esa extensión. En el apéndice A introduciremos una sintaxis elemental para ilustrar una implementación de FIRST en un gestor comercial.
2. Una vez formulada dicha sintaxis o cualquier otro mecanismo para articular la consulta en sus componentes elementales, el FCP se encargará resolver tales componentes en sentencias clásicas.
3. A continuación, dichas sentencias se agruparan, siguiendo el criterio establecido en la consulta, en una sentencia global que se enviará al RDBMS anfitrión.

4. De la ejecución de esa sentencia se obtendrá el resultado de la consulta anteriormente formulada.

De los pasos enumerados anteriormente el único paso que resuelve el FCP es el segundo. En concreto, esta consulta precisa del FCP para resolver los siguientes componentes "difusos" de la misma:

1. La comparación difusa expresada mediante la expresión "*empleados con una EDAD "madura" (con un grado ≥ 0.8)*".
2. El cálculo del grado con que satisface el atributo EDAD dicha consulta para cada tupla. Osea, en términos de GEFRED, la proyección sobre el atributo de compatibilidad asociado al atributo EDAD.
3. La proyección sobre el atributo EDAD. Con dicha tarea el FCP debe dar a cada dato presente en dicho atributo un formato legible. Dicho formato debe obtenerlo de la información almacenada en los atributos de la Base de Datos que ha servido para implementar este tipo de atributo "difuso".

El FCP no resuelve las componentes de la consulta de forma independiente, sino que el conjunto de sentencias que devuelve para cada componente es función de que se soliciten las otras. Así por ejemplo, las sentencias clásicas que tendríamos como resultado de la resolución por parte del FCP de la primera componente, serían distintas, si solo queremos obtener la proyección sobre el atributo EDAD, de las que obtendríamos si solicitamos además la proyección sobre el atributo de compatibilidad asociado a EDAD.

Por razones de eficiencia, el FCP opera realizando una partición disjunta de la relación en conjuntos de tuplas que satisfacen la condición con un grado superior o igual al umbral establecido en consulta. Dicha partición va clasificada según el tipo de los datos que contienen las tuplas para el atributo implicado en la consulta, (EDAD en nuestro caso). De esta forma puede aplicar las fórmulas de conversión adecuadas para cada tipo de dato "difuso" y puede además calcular el grado de compatibilidad correspondiente, si es el caso.

Para resolver la primera de las componentes enumeradas de la consulta, el FCP necesita obtener de la FMB la información sobre el atributo EDAD que le indique si es de tipo "difuso" y si lo es, debe obtener que tipo de atributo "difuso" es,

(para ello consulta la tabla FUZZY_COL_LIST). A continuación debe comprobar si la etiqueta lingüística "maduro" se encuentra definida sobre dicho atributo (información recogida en la tabla FUZZY_OBJECT_LIST). Y por último, obtener de la tabla FUZZY_LABEL_DEF la distribución de posibilidad asociada a dicha etiqueta.

Una vez que posee esa información procede a elaborar los fragmentos de sentencias SQL que recuperarán las tuplas que satisfacen la condición expresada en la consulta. Esos fragmentos serán combinados con los que se generen a partir de otros elementos de la consulta (otras condiciones) para ensamblar, al final del proceso, la sentencia global a enviar al Gestor anfitrión.

A continuación mostramos la sentencia global que resuelve la consulta propuesta. Los fragmentos elaborados por el FCP se encuentran en mayúscula:

```
select nombre ,direccion ,DECODE(EDADT,0,'UNKNOWN',1,'UNDEFINED',
2,'NULL',3,TO_CHAR(EDAD1),4,DECODE(EDAD1,0,'JOVEN',1,'MADURO',2,
'MAYOR'),5,['||TO_CHAR(EDAD1)||','||TO_CHAR(EDAD4)||'],'6,
['||TO_CHAR(EDAD1)||','||TO_CHAR(EDAD1+EDAD2)||','||TO_CHAR(
EDAD4)||'],'7,['||TO_CHAR(EDAD1)||','||TO_CHAR(EDAD2+EDAD1)||
','||TO_CHAR(EDAD3+EDAD4)||','||TO_CHAR(EDAD4)||'],' "EDAD"
,ROUND(DECODE(EDADT,0,1,2,1,3,DECODE(SIGN((EDAD1-33)*(47-EDAD1)),
-1,0,DECODE(SIGN(35-EDAD1),1,(EDAD1-25)/10,DECODE(SIGN(EDAD1-45),
1,(EDAD1-55)/-10,1))),4,DECODE(EDAD1,1,1.00,0),5,DECODE(SIGN(
(EDAD4-33)*(47-EDAD1)),-1,0,DECODE(SIGN(35-EDAD4),1,(EDAD4-25)/10,
DECODE(SIGN(EDAD1-45),1,(EDAD1-55)/-10,1))),6,DECODE(SIGN((EDAD4-
37)*(53-EDAD4)), -1,0,DECODE(SIGN(35-EDAD4+5),1,(EDAD4-25)/(10+5),
DECODE(SIGN(45-EDAD4+5),1,1,(EDAD1-55)/(-10-5))))),7,DECODE(SIGN(
(0.80*EDAD3+EDAD4-33)*(47-0.80*EDAD2+EDAD1)), -1,0,DECODE(SIGN(
(EDAD3+EDAD4-35)*(45-EDAD2+EDAD1)), -1,DECODE(SIGN((0.80*EDAD3+
EDAD4-33)*(35-EDAD3+EDAD4)), -1,DECODE(SIGN((47-0.80*EDAD2+EDAD1)
*(EDAD2+EDAD1-45)),1,(55-EDAD1)/(EDAD2--10)),(EDAD4-25)/(10-
EDAD3)),1)),0),2) "CDEG(EDAD)" from empleados where (EDADT=0
OR EDADT=2 OR EDADT=3 AND EDAD1 BETWEEN 33 AND 47 OR EDADT=4
AND EDAD1 IN (1) OR EDADT=5 AND EDAD1<=47 AND EDAD4>=33 OR
EDADT=6 AND EDAD4 BETWEEN 37 AND 53 OR EDADT=7 AND EDAD4>=33
AND EDAD1<=47 AND 0.80*EDAD3+EDAD4>=33 AND 0.80*EDAD2+EDAD1<=47)
```

El resultado⁵ de ejecutar esta sentencia, que como puede verse es de sintaxis clásica, es el que se muestra en la tabla 5.23

NOMBRE	DIRECCIÓN	EDAD	CEDG(EDAD)
Antonio	Reyes Catolicos	Maduro	1
Javier	Gran Via	[30,35]	1

Tabla 5.23: Resultado de la consulta ejemplo

⁵Esta sentencia y su ejecución ha sido elaborada sobre el ejemplo de implementación de FIRST desarrollado en la plataforma Oracle y cuyos detalles se exponen en el apéndice A

Conclusiones y Direcciones Futuras

Los resultados presentados en esta memoria pueden resumirse en los siguientes puntos:

- **GEFRED.** Este modelo presenta las siguientes ventajas:
 - Los tipos de datos con los que opera es bastante extenso en comparación con otras propuestas. Su naturaleza difusa es más variada que en modelos precedentes.
 - Organiza en forma consistente la información difusa. Las “relaciones difusas generalizadas” estructuran, tanto la información de partida, como la que resulta de las operaciones realizadas sobre la misma.
 - El modelo presenta gran flexibilidad para el tratamiento y evaluación de la información difusa, basado en la libertad a la hora de seleccionar un comparador difuso generalizado u otro y en que se puede controlar el grado en que se satisfacen las condiciones individuales de una consulta.
 - Por último, una gran parte de los modelos precedentes se pueden considerar, bajo ciertas condiciones, como casos particulares de GEFRED. Por tanto, este modelo se puede emplear para representarlos.
- **FIRST.** Esta implementación posee las siguientes características:
 - Ofrece amplias posibilidades para la representación y manipulación de datos difusos.
 - Proporciona un tratamiento coherente y unificado para los datos difusos atendiendo a la naturaleza de los mismos.
 - Selecciona una representación simplificada de los items difusos, lo cual facilita el cálculo y no resta significado a las operaciones.

- Se construye con las herramientas disponibles en cualquier RDBMS relacional comercial.
- Resulta de fácil implementación y es portable de una plataforma a otra.
- Respeto todas las operaciones originales del RDBMS anfitrión.
- Constituye un prototipo ideal para experimentar con diferentes tipos de medidas y operaciones difusas.

En definitiva, a la luz de los resultados obtenidos, podemos, en líneas generales, concluir lo siguiente:

1. Resulta ineludible la extensión del Modelo Relacional para que manipule información imprecisa.
2. Es viable esta extensión mediante el empleo de los recursos que proporciona la teoría de conjuntos difusos.
3. En el ámbito de dicha teoría, el modelo que introducimos en esta memoria constituye una buena forma de realizar dicha extensión.
4. Es factible la construcción de sistemas relacionales de bases de datos difusas basados en dicho modelo.
5. Es posible que estos sistemas se beneficien de los recursos actualmente disponibles en los RDBMS convencionales y que coexistan con ellos preservando las inversiones realizadas sobre dichos sistemas.

Líneas Futuras de Investigación.

En el ámbito teórico, se encuentran en estudio los siguientes aspectos:

- Aunque la división relacional no constituye una primitiva del álgebra relacional, está previsto introducir una definición de dicho operador en el ámbito del modelo propuesto.
- También se trabaja en la introducción de un Cálculo Difuso Relacional para GEFRED.

- Así mismo se contempla un estudio en profundidad sobre la construcción de *comparadores difusos generalizados* que permitan cubrir todas las necesidades que presenta la variada semántica de la manipulación de datos imprecisos.

En cuanto a los aspectos relacionados con la implementación del modelo, algunos de los caminos a seguir son:

- Desarrollar un prototipo completo de FIRST que incorpore todos los módulos contemplados en su especificación.
- Incrementar la variedad de los operadores implementados y la aplicabilidad de los mismos, incorporando tales operadores al FCP.
- Desarrollar una extensión difusa de SQL completa incluyendo todas las operaciones de los sublenguajes DDL, DCL y DML.
- Incorporar herramientas para el desarrollo de aplicaciones sobre bases de datos difusas.
- Adaptar la especificación de FIRST a los RDBMS convencionales más difundidos.
- Estudiar la viabilidad de desarrollos a bajo nivel de implementaciones basadas en GEFRED.

Apéndice A

Implementación de FIRST sobre un RDBMS comercial(ORACLE)

A.1 Introducción

La memoria que estamos presentando propone un modelo teórico de bases de datos relacionales difusas e introduce un esquema para la descripción y puesta en práctica del mismo. El contenido de este trabajo se nos antoja incompleto si no mostramos de alguna forma que los postulados introducidos pueden dar lugar a una propuesta práctica. El objeto de este apéndice es el de mostrar un prototipo de FIRST construido entorno a un RDBMS comercial (Oracle). La complejidad, y sobre todo, la extensión, que supone llevar a la práctica toda la capacidad implícita en GEFRED y en la implementación FIRST, nos ha llevado a acotar la implementación que nos proponemos describir. El criterio seguido a la hora de delimitar el prototipo, ha sido el de proporcionar una idea de como podrían llevarse a cabo algunos de los procesos de manipulación modelados en GEFRED y adoptados en FIRST. En este sentido, nos hemos centrado en implementar algunos de los aspectos relativos al proceso de consulta con información “difusa” de por medio. Por el momento, y por los motivos aducidos anteriormente, no hemos implementado las sentencias correspondientes al Lenguaje de Definición de Datos (DDL), esto no significa que no se contemple su implementación, sino que nos parecía prioritario la construcción de parte de las sentencias del Lenguaje de Manipulación de Datos (DML).

La implementación que vamos a describir se ha realizado sobre la versión 6 del

RDBMS Oracle, de Oracle Corporation, ejecutándose sobre plataforma SPARC, de SUN Microsystem, y S.O. SunOS, (variante de UNIX). Los motivos que nos han conducido a la citada elección son:

- Oracle Corporation es una empresa pionera y refutada en el ámbito de las bases de datos relacionales. Posee productos bastante consolidados y con amplia cobertura en casi todas las plataformas “hardware” y “software”.
- Recoge bastante bien los postulados del modelo relacional, aunque la versión empleada no fuerza la integridad referencial, (la versión 7, actualmente en vigor, si contempla dicha regla).
- Proporciona numerosas herramientas de valor añadido para el desarrollo de aplicaciones sobre dicho gestor.
- Se ha llevado a cabo la implementación sobre la versión 6 puesto que esta era la que estaba disponible en el momento de su desarrollo, además, esto añade generalidad a los mecanismos adoptados para materializarla.
- La arquitectura SPARC y el sistema operativo UNIX gozan de gran aceptación y difusión en todo el mundo. Además, tales estaciones de trabajo constituyen un soporte bastante adecuado para la ejecución de RDBMS. Esto es así por sus posibilidades tanto de cálculo, como de almacenamiento y de soporte de conexión en red, lo que las hace ideales para experimentar con las posibilidades que encierra el tratamiento multiproceso y multiusuario de la información gestionada por un RDBMS.

Además del gestor indicado, ha sido preciso el empleo de las siguientes herramientas para llevar a cabo el prototipo que se describe en este apéndice:

- Entorno de programación C en UNIX, (cc, make, dbx, editores, etc.).
- Herramientas para el desarrollo de analizadores lexicos y sintácticos (LEX y YACC de UNIX).
- Herramientas del entorno Oracle para el desarrollo de programas C con SQL inmerso (PROC).

- Herramientas interactivas de acceso al RDBMS (SQLPLUS) así como de desarrollo de aplicaciones con lenguajes de 4^a Generación (SQLFORMS, SQLMENU).

Con los elementos citados anteriormente, se ha desarrollado un prototipo de FIRST que permite representar toda la gama de información difusa considerada en el capítulo 5 y que proporciona un interprete de comandos, que es capaz de procesar consultas en una variante de SQL, que contempla la sintaxis ofrecida por el SQL de Oracle, combinada con un conjunto de cláusulas SQL difusas para materializar consultas difusas. El esquema adoptado se corresponde con el propuesto por FIRST, si bien, como hemos indicado, acotado en sus posibilidades.

A continuación presentaremos los diferentes elementos que constituyen la implementación del prototipo propuesto. Comenzaremos por introducir un ejemplo de base de datos que nos servirá para ilustrar los diferentes aspectos del prototipo, así como para poder realizar algunos ejemplos de consultas sobre él. Después, describiremos, mediante ese ejemplo, la representación que adoptamos en Oracle para una base de datos con información difusa, también se mostrará el esquema de la FMB adoptado en el prototipo. El siguiente apartado mostrará los criterios empleados para realizar los aspectos del FCP contemplados en el prototipo. Tras la descripción de un procesador elemental de FSQL, procederemos a ilustrar su funcionamiento mediante la ejecución de varias consultas sobre la base de datos ejemplo.

A.2 Base de Datos Ejemplo

En este apartado vamos a introducir la base de datos que vamos a emplear para ilustrar los aspectos concernientes al prototipo de FIRST que vamos a describir. Esta base de datos no pretende, ni ser un modelo de la forma en que puede emplearse un SBDRD, ni aspira a agotar las posibilidades, tanto del modelo, (GEFRED), como de la implementación, (FIRST). Por tanto, con la adopción de este ejemplo de base de datos se pretende cubrir los siguientes objetivos:

- Mostrar como se representa la información estructural de una base de datos en FIRST.
- Comprender como se almacenan en la FMB todas las definiciones que conciernen a una base de datos con información difusa de diversa índole.

- Poner de manifiesto los mecanismos habilitados en esta implementación real para obtener toda la información que contiene el sistema (representación, definiciones, objetos difusos, etc.).
- Ilustrar algunas de las operaciones de manipulación de que dispone en la actualidad el prototipo mostrado.
- Poner de manifiesto algunos de los mecanismos de manipulación descritos, como se han implementado y como funcionan.

Con estos propósitos en mente, la información que recoge la base de datos ejemplo presenta un aspecto parcial de la información que una empresa pudiera contener acerca de la estructura laboral de la misma. Las tablas que recogen dicha información tienen la siguiente estructura:

- **Personal:** Esta tabla contiene información básica acerca del personal de la empresa. Se compone de los atributos:
 - *EMP#*: Tipo CHAR(4). Es la clave primaria identificando cada empleado de la tabla. Es clave externa para la tabla *Aptitud*.
 - *NOMBRE*: Tipo CHAR(20). Es el nombre del empleado considerado.
 - *SEXO*: Tipo CHAR(1). El valor 'V' representa "Hombre" y 'H' "Mujer".
 - *EDAD*: Atributo difuso de Tipo 2. Tiene definidas etiquetas lingüísticas sobre el dominio subyacente.
 - *CIUDAD*: Tipo CHAR(15). Ciudad de Origen.
- **Aptitud:** Almacena las principales características de formación y de capacitación de los empleados:
 - *EMP#*: Tipo CHAR(4). Es la clave primaria identificando cada empleado en la tabla y es clave externa para la tabla *Personal*.
 - *ESTUDIOS*: De Tipo 3. Almacena el nivel de formación académica de cada empleado. Tiene definida una relación de semejanza mediante la cual se modela la proximidad que pueden presentar formaciones diferentes para determinadas actividades.

- *PROF*: De Tipo 3. Contiene la profesión de cada empleado. También se encuentra definida una relación de semejanza sobre el dominio que constituyen las diferentes profesiones consideradas.
- *EXPERIENCIA*: De Tipo 2. Almacena el periodo de tiempo que cada empleado lleva en el desempeño de su actividad su profesión.
- **Dpto**: Contiene la descripción de los diferentes departamentos que configuran la empresa:
 - *DPTO#*: CHAR(4). Es el código de departamento, constituye la clave primaria de la tabla en cuestión.
 - *NOMBRE*: CHAR(20). Es la designación del departamento.
 - *LOCALIZ*: CHAR(15). Ciudad en la que se ubica.
 - *BENEFICIOS*: NUMBER(10), es un atributo de Tipo 1. Recoge los beneficios por ejercicio de cada departamento, permite la definición de etiquetas lingüísticas sobre el dominio numérico que le subyace.
- **Puestos**: Esta tabla recoge los diferentes cargos disponibles en la empresa y la cualificación recomendable:
 - *PUESTO#*: CHAR(4). Es la clave primaria y codifica cada uno de los cargos.
 - *NOMBRE*: CHAR(20). Es la denominación del cargo.
 - *ESTUDIOS*: Tipo 3. Mismas características que el atributo del mismo nombre en la tabla *Aptitud*.
 - *PROF*: Tipo 3. Análogo al atributo de la tabla *Aptitud*.
 - *EXPERIENCIA*: Experiencia recomendable, similar al atributo denominado de igual modo en la tabla *Aptitud*.
- **Empleos**: Esta tabla muestra los empleos cubiertos actualmente; que empleo, en que departamento, quién lo ocupa y cómo:
 - *EMP#*: CHAR(4). Clave externa a la tabla *Personal* y *Aptitud*.
 - *DPTO#*: CHAR(4). Clave externa a la tabla *Dpto*.

- *PUESTO#*: CHAR(4). Clave externa a la tabla *Puestos*.
 - *SUELDO*: NUMBER(7), es un atributo de Tipo 1. Representa el sueldo bruto mensual. Posee una serie de etiquetas definidas sobre el dominio subyacente.
 - *COMISION*: De Tipo 2. Almacena los complementos al sueldo básico. Tiene definidas igualmente etiquetas sobre el dominio.
 - *RENDIMIENTO*: De Tipo 3. Almacena evaluaciones subjetivas en cuanto al rendimiento laboral del empleado. Se define una relación de semejanza entre los valores asumidos por el dominio.
- **Ciudades:** Recoge las distancias que separan las ciudades que alojan a los empleados y a los departamentos:
 - *DISTANCIA*: NUMBER(4) atributo de Tipo 1. Distancia que separa las ciudades que figuran en los atributos *CIUDAD1* y *CIUDAD2*. Poseerá etiquetas lingüísticas sobre el dominio distancia.
 - *CIUDAD1*: CHAR(15). Ciudad sobre la que se evalúa la distancia.
 - *CIUDAD2*: CHAR(15). La otra ciudad sobre la que se evalúa la distancia.

En las tablas A.1, A.2, A.3, A.4, A.5 y A.6 se recoge el contenido de las relaciones del ejemplo considerado. Dicho contenido no se corresponde con ningún caso real, por lo que las conclusiones que de él se extraigan sólo tienen validez a título de ejemplo.

EMP#	NOMBRE	SEXO	EDAD	CIUDAD
E0	VICTOR	Hombre	APRENDIZ	BARCELONA
E1	SIXTO	Hombre	17	GRANADA
E2	OLGA	Mujer	27	GRANADA
E3	JOSE	Hombre	[25,30]	MALAGA
E4	SALVADOR	Hombre	[40,35,38,45]	VALENCIA
E5	MATILDE	Mujer	[25,30,35]	MADRID
E6	ANA	Mujer	UNKNOWN	BARCELONA
E7	CANDIDO	Hombre	45	GRANADA
E8	LUIS	Hombre	18	MALAGA
E9	FEDERICO	Hombre	23	VALENCIA
E10	ALVARO	Hombre	JOVEN	GRANADA
E11	ELADIO	Hombre	MAYOR	MADRID
E12	CARLOS	Hombre	MUY_MAYOR	MADRID
E13	CRISTINA	Mujer	MADURO	MALAGA
E14	INES	Mujer	JOVEN	VALENCIA
E15	JULIAN	Hombre	31	BARCELONA
E16	ANTONIO	Hombre	55	MALAGA
E17	SILVIA	Mujer	[30,35,40]	VALENCIA
E18	PALOMA	Mujer	27	GRANADA
E19	FERNANDO	Hombre	JOVEN	BARCELONA
E20	FELIPE	Hombre	60	VALENCIA
E21	RICARDO	Hombre	48	VALENCIA
E22	MERCEDES	Mujer	17	GRANADA

Tabla A.1: Tabla *Personal*

EMP#	ESTUDIOS	PROF	EXPERIENCIA
E0	EGB	VENDEDOR	POCA
E1	EGB	ADMINISTRATIVO	POCA
E2	LICENCIADO	INGENIERO	ALGUNA
E3	LICENCIADO	ADMINISTRATIVO	3
E4	BACHILLER	VENDEDOR	20
E5	BACHILLER	VENDEDOR	5
E6	EGB	SECRETARIO	UNKNOWN
E7	DOCTOR	DIRECTOR	BASTANTE
E8	BACHILLER	VENDEDOR	1
E9	BACHILLER	ADMINISTRATIVO	[0,2,4]
E10	BACHILLER	SECRETARIO	ALGUNA
E11	LICENCIADO	DIRECTOR	BASTANTE
E12	LICENCIADO	INGENIERO	20
E13	DOCTOR	DIRECTOR	15
E14	BACHILLER	SECRETARIO	[1,3,5]
E15	LICENCIADO	INGENIERO	[2,6,13,7]
E16	BACHILLER	ADMINISTRATIVO	BASTANTE
E17	LICENCIADO	INGENIERO	7
E18	LICENCIADO	TECNICO	4
E19	BACHILLER	VENDEDOR	ALGUNA
E20	LICENCIADO	TECNICO	GRANDE
E21	LICENCIADO	TECNICO	BASTANTE
E22	EGB	SECRETARIO	1

Tabla A.2: Tabla *Aptitud*

DPTO#	NOMBRE	LOCALIZ	BENEFICIOS
D0	EXPORTACION	MADRID	35000000
D1	CENTRO	MADRID	30000000
D2	SUR	MALAGA	40000000
D3	NORTE	BARCELONA	35000000
D4	LEVANTE	VALENCIA	10000000

Tabla A.3: Tabla *Dpto.*

P#	NOMBRE	ESTUDIOS	PROF	EXPER
P0	DIRECTOR	LICENCIADO	DIRECTOR	BASTANTE
P1	INGENIERO	LICENCIADO	INGENIERO	ALGUNA
P2	TECNICO	LICENCIADO	TECNICO	ALGUNA
P3	ADMINISTRATIVO	BACHILLER	ADMINISTRATIVO	ALGUNA
P4	SECRETARIO	BACHILLER	SECRETARIO	ALGUNA
P5	VENDEDOR	EGB	VENDEDOR	ALGUNA

Tabla A.4: Tabla *Puestos*

EMP#	DPTO#	P#	SUELDO	COMISION	REND
E0	D1	P5	60000	10000	REGULAR
E1	D1	P3	80000	5000	BUENO
E2	D3	P1	200000	NORMAL	BUENO
E3	D4	P3	120000	10000	BUENO
E4	D0	P5	160000	ALTA	EXCELENTE
E5	D2	P5	140000	30000	BUENO
E6	D0	P4	120000	REDUCIDA	REGULAR
E7	D0	P0	500000	ELEVADA	EXCELENTE
E8	D2	P5	90000	35000	BUENO
E9	D1	P3	120000	[0,5000,10000]	BUENO
E10	D3	P4	110000	BAJA	BUENO
E11	D1	P0	450000	ELEVADA	BUENO
E12	D3	P1	350000	40000	BUENO
E13	D4	P0	475000	50000	BUENO
E14	D3	P4	100000	BAJA	BUENO
E15	D2	P1	230000	[50000,60000,85000,100000]	BUENO
E16	D1	P3	170000	NORMAL	EXCELENTE
E17	D3	P1	180000	REDUCIDA	MALO
E18	D2	P2	160000	20000	BUENO
E19	D2	P5	130000	ALTA	EXCELENTE
E20	D0	P2	230000	NORMAL	BUENO
E21	D3	P2	270000	[20000,25000,30000]	BUENO
E22	D1	P4	85000	BAJA	BUENO

Tabla A.5: Tabla *Empleos*

CIUDAD1	CIUDAD2	DISTANCIA
BARCELONA	MADRID	621
BARCELONA	MALAGA	995
BARCELONA	GRANADA	875
BARCELONA	VALENCIA	352
MADRID	MALAGA	542
MADRID	GRANADA	432
MADRID	VALENCIA	350
MALAGA	GRANADA	128
MALAGA	VALENCIA	643
GRANADA	VALENCIA	528

Tabla A.6: Tabla *Ciudades*

En las tablas A.7, A.8 y A.9 se encuentran definidas a relaciones de semejanza adoptadas para los dominios *Estudios*, *Profesión* y *Rendimiento*. En los dos primeros casos, estas relaciones de semejanza modelan la cercanía que puedan presentar las diferentes preparaciones: académica y profesional. La tercera relación modela el parecido entre los diferentes grados de rendimiento adoptados.

s	EGB	Bachiller	Licenciado	Doctor
EGB	1	.5	.2	.1
Bachiller	.5	1	.6	.3
Licenciado	.2	.6	1	.8
Doctor	.1	.3	.8	1

Tabla A.7: Relación de semejanza sobre el dominio *Estudios*

s	Director	Ingeniero	Técnico	Administ.	Secret.	Vendedor
Director	1	.8	.7	.6	.4	.4
Ingeniero	.8	1	.8	.2	.2	.2
Técnico	.7	.8	1	.8	.6	.5
Administ.	.6	.2	.8	1	.7	.6
Secret.	.4	.2	.6	.7	1	.6
Vendedor	.4	.2	.5	.6	.6	1

Tabla A.8: Relación de semejanza sobre el dominio *Profesión*

s	Malo	Regular	Bueno	Excelente
Malo	1	.8	.5	.1
Regular	.8	1	.7	.5
Bueno	.5	.7	1	.8
Excelente	.1	.5	.8	1

Tabla A.9: Relación de semejanza sobre el dominio *Rendimiento*

Las figuras A.1, A.2, A.3, A.4, A.5 y A.6 contienen la representación de las diferentes etiquetas adoptadas para los dominios considerados en el ejemplo.

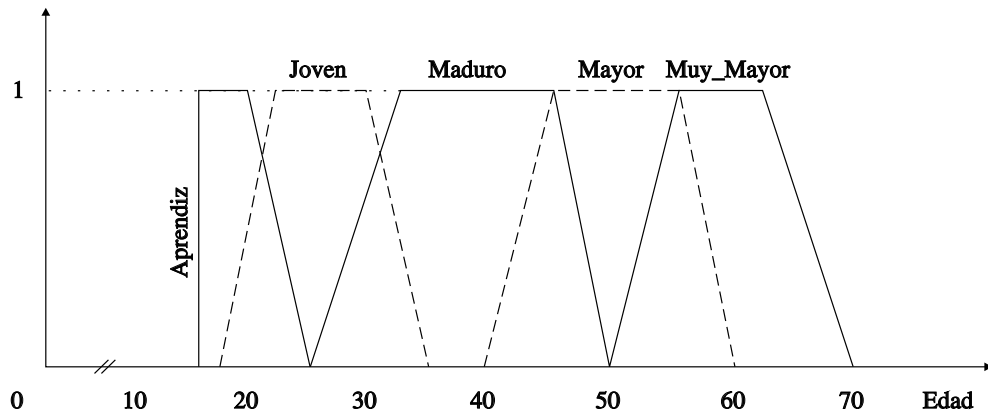


Figura A.1: Etiquetas sobre el dominio *Edad*, (en años)

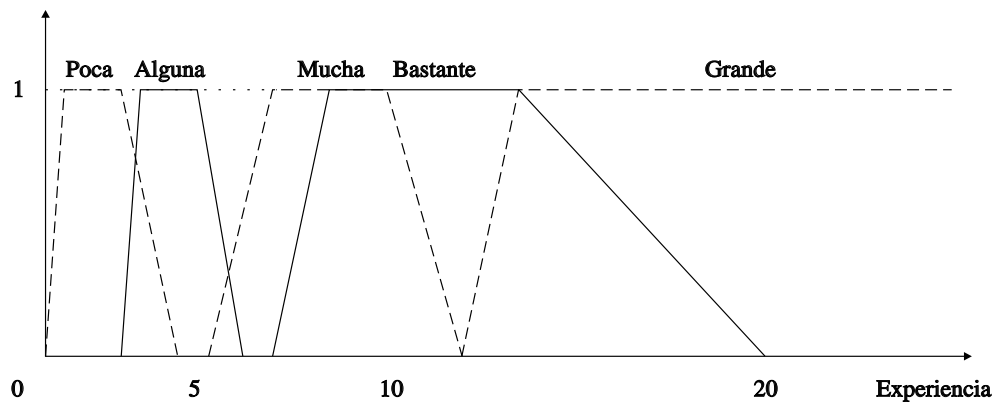


Figura A.2: Etiquetas sobre el dominio *Experiencia*, (en años)

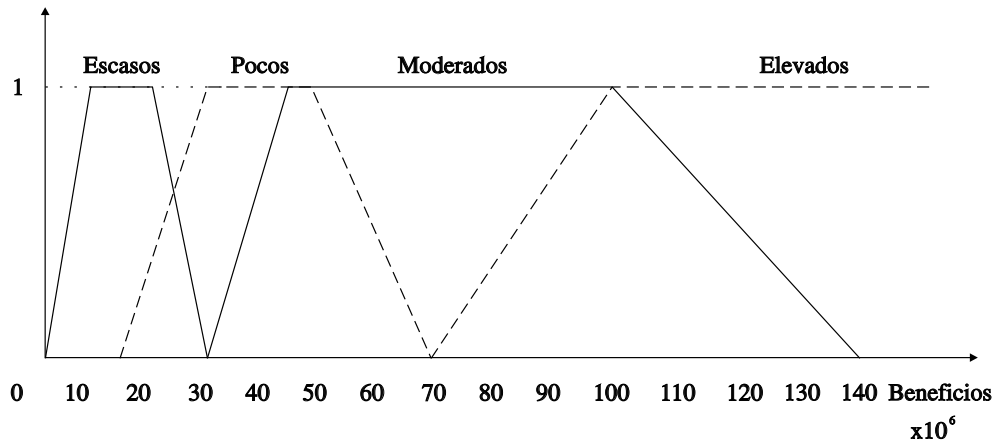


Figura A.3: Etiquetas sobre el dominio *Beneficios*

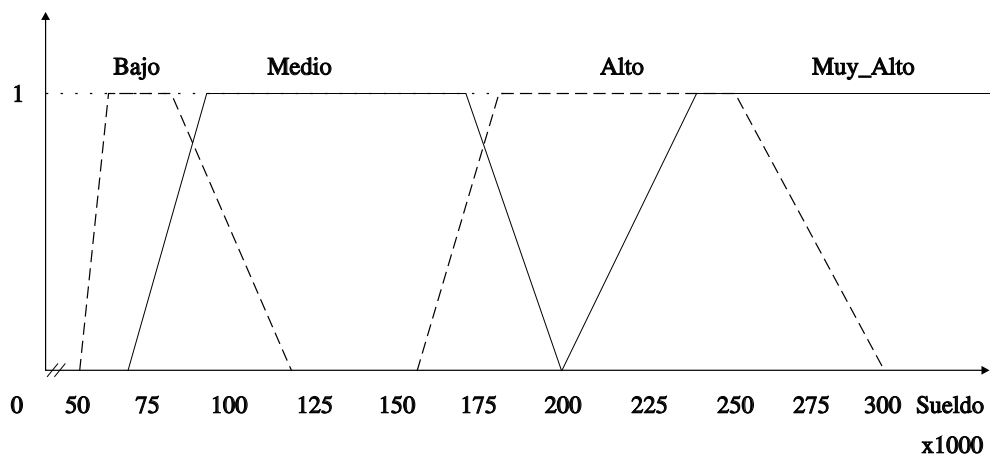


Figura A.4: Etiquetas sobre el dominio *Sueldo*

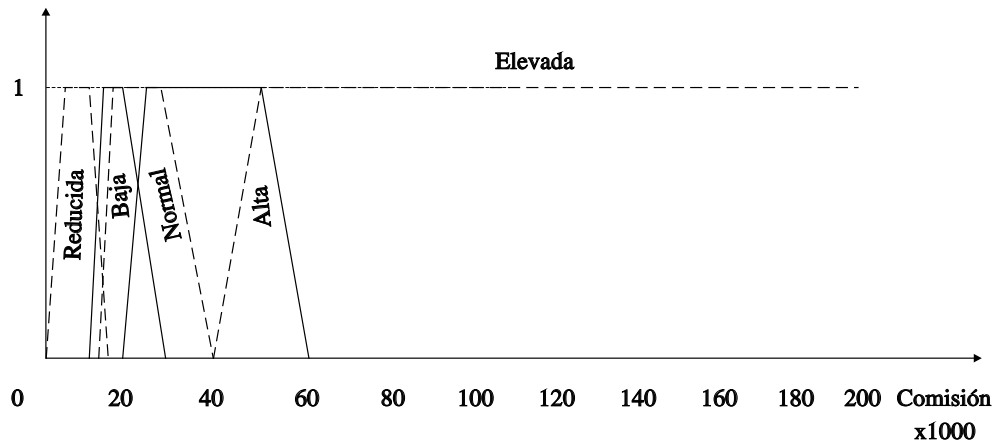


Figura A.5: Etiquetas sobre el dominio *Comisión*

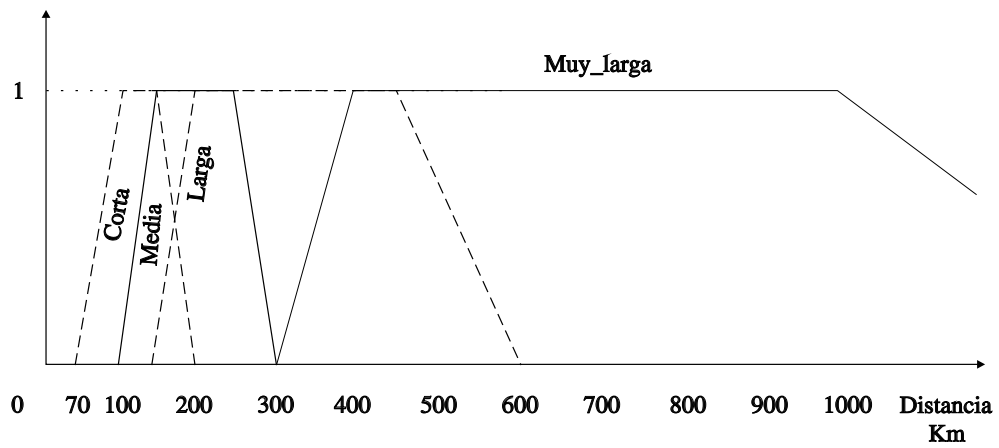


Figura A.6: Etiquetas sobre el dominio *Distancia*

A.3 Estructura de Datos de la FMB y de la BD

En este apartado mostraremos como FIRST representa de forma efectiva la información contenida en las relaciones. La implementación de la FMB es esencialmente igual a la estudiada en el capítulo 5. Únicamente indicar que los atributos `obj#` y `col#` que identifican a cada atributo difuso en la FMB, adoptan, para cada item difuso, el código que le asigna Oracle en el momento de su creación.

Para facilitar al usuario la obtención de información acerca de items difusos a partir de la FMB, se ha creado un conjunto de vistas que la muestran en una forma comprensible. La construcción de estas vistas se realiza a partir de las tablas de la FMB y del catálogo de Oracle. Estas vistas, en forma similar a como ocurre con el catálogo de Oracle, están clasificadas en tres categorías: User, All y DBA. Emplearemos alguna de estas vistas para mostrar la forma en que se almacena la información difusa contenida en el ejemplo del apartado anterior. Las tablas A.10, A.11, A.12, A.13, A.14 y A.15 muestran dicha información.

TABLE_NAME	COL_NAME	OBJ_ID	COL_ID	COL_TYPE
APTITUD	ESTUDIOST	863	2	NEARNESS
APTITUD	EXPERIENCIAT	863	6	POSSIBILISTIC
APTITUD	PROFT	863	4	NEARNESS
CIUDADES	DISTANCIA	867	3	CRISP
DPTO	BENEFICIOS	865	4	CRISP
EMPLEOS	COMISIONT	862	5	POSSIBILISTIC
EMPLEOS	RENDIMIENOT	862	10	NEARNESS
EMPLEOS	SUELDO	862	4	CRISP
PERSONAL	EDADT	866	4	POSSIBILISTIC
PUESTOS	ESTUDIOST	864	3	NEARNESS
PUESTOS	EXPERIENCIAT	864	7	POSSIBILISTIC
PUESTOS	PROFT	864	5	NEARNESS

Tabla A.10: Contenido de la tabla Fuzzy_Col_List relativo a la BD ejemplo

TABLE_NAME	COL_NAME	ITEM_NAME	ITEM_TYPE	OBJ_ID	COL_ID	ITEM_ID
APTITUD	ESTUDIOS	EGB	NLABEL	863	2	0
APTITUD	ESTUDIOS	BACHILLER	NLABEL	863	2	1
APTITUD	ESTUDIOS	LINCENCIADO	NLABEL	863	2	2
APTITUD	ESTUDIOS	DOCTOR	NLABEL	863	2	3
APTITUD	EXPERIENCIAT	POCA	PLABEL	863	6	0
APTITUD	EXPERIENCIAT	ALGUNA	PLABEL	863	6	1
APTITUD	EXPERIENCIAT	MUCHA	PLABEL	863	6	2
APTITUD	EXPERIENCIAT	BASTANTE	PLABEL	863	6	3
APTITUD	EXPERIENCIAT	GRANDE	PLABEL	863	6	4
APTITUD	PROFT	DIRECTOR	NLABEL	863	4	0
APTITUD	PROFT	INGENIERO	NLABEL	863	4	1
APTITUD	PROFT	TECNICO	NLABEL	863	4	2
APTITUD	PROFT	ADMINISTRATIVO	NLABEL	863	4	3
APTITUD	PROFT	SECRETARIO	NLABEL	863	4	4
APTITUD	PROFT	VENDEDOR	NLABEL	863	4	5
CIUDADES	DISTANCIA	CORTA	PLABEL	867	3	0
CIUDADES	DISTANCIA	MEDIA	PLABEL	867	3	1
CIUDADES	DISTANCIA	LARGA	PLABEL	867	3	2
CIUDADES	DISTANCIA	MUY_LARGA	PLABEL	867	3	3
DPTO	BENEFICIOS	ESCASOS	PLABEL	865	4	0
DPTO	BENEFICIOS	POCOS	PLABEL	865	4	1
DPTO	BENEFICIOS	MODERADOS	PLABEL	865	4	2
DPTO	BENEFICIOS	ELEVADOS	PLABEL	865	4	3
EMPLEOS	COMISIONT	REDUCIDA	PLABEL	862	5	0
EMPLEOS	COMISIONT	BAJA	PLABEL	862	5	1
EMPLEOS	COMISIONT	NORMAL	PLABEL	862	5	2
EMPLEOS	COMISIONT	ALTA	PLABEL	862	5	3
EMPLEOS	COMISIONT	ELEVADA	PLABEL	862	5	4
EMPLEOS	RENDIMIENOT	MALO	NLABEL	862	10	0
EMPLEOS	RENDIMIENOT	REGULAR	NLABEL	862	10	1
EMPLEOS	RENDIMIENOT	BUENO	NLABEL	862	10	2
EMPLEOS	RENDIMIENOT	EXCELENTE	NLABEL	862	10	3
EMPLEOS	SUELDO	BAJO	PLABEL	862	4	0
EMPLEOS	SUELDO	MEDIO	PLABEL	862	4	1
EMPLEOS	SUELDO	ALTO	PLABEL	862	4	2
EMPLEOS	SUELDO	MUY_ALTO	PLABEL	862	4	3
PERSONAL	EDADT	APRENDIZ	PLABEL	866	4	0
PERSONAL	EDADT	JOVEN	PLABEL	866	4	1
PERSONAL	EDADT	MADURO	PLABEL	866	4	2
PERSONAL	EDADT	MAYOR	PLABEL	866	4	3
PERSONAL	EDADT	MUY_MAYOR	PLABEL	866	4	4
PUESTOS	ESTUDIOS	EGB	NLABEL	864	3	0
PUESTOS	ESTUDIOS	BACHILLER	NLABEL	864	3	1
PUESTOS	ESTUDIOS	LICENCIADO	NLABEL	864	3	2
PUESTOS	ESTUDIOS	DOCTOR	NLABEL	864	3	3

Tabla A.11: Contenido de la tabla Fuzzy_Object_List relativo a la BD ejemplo

TABLE_NAME	COL_NAME	ITEM_NAME	ITEM_TYPE	OBJ_ID	COL_ID	ITEM_ID
PUESTOS	EXPERIENCIAT	POCA	PLABEL	864	7	0
PUESTOS	EXPERIENCIAT	ALGUNA	PLABEL	864	7	1
PUESTOS	EXPERIENCIAT	MUCHA	PLABEL	864	7	2
PUESTOS	EXPERIENCIAT	BASTANTE	PLABEL	864	7	3
PUESTOS	EXPERIENCIAT	GRANDE	PLABEL	864	7	4
PUESTOS	PROFT	DIRECTOR	NLABEL	864	5	0
PUESTOS	PROFT	INGENIERO	NLABEL	864	5	1
PUESTOS	PROFT	TECNICO	NLABEL	864	5	2
PUESTOS	PROFT	ADMINISTRATIVO	NLABEL	864	5	3
PUESTOS	PROFT	SECRETARIO	NLABEL	864	5	4
PUESTOS	PROFT	VENDEDOR	NLABEL	864	5	5

Tabla A.12: Continuación de la tabla A.11

COL_NAME	ITEM_NAME	ITEM_ID	ALFA	BETA	GAMMA	DELTA
BENEFICIOS	ESCASOS	0	0	10000000	20000000	30000000
BENEFICIOS	POCOS	1	15000000	30000000	50000000	70000000
BENEFICIOS	MODERADOS	2	30000000	45000000	100000000	140000000
BENEFICIOS	ELEVADOS	3	70000000	100000000	1000000000	1000000000
COMISIONT	REDUCIDA	0	0	5000	10000	15000
COMISIONT	BAJA	1	11000	15000	20000	30000
COMISIONT	NORMAL	2	15000	17000	30000	40000
COMISIONT	ALTA	3	20000	32000	50000	60000
COMISIONT	ELEVADA	4	40000	50000	200000	300000
DISTANCIA	CORTA	0	70	100	150	200
DISTANCIA	MEDIA	1	100	150	250	300
DISTANCIA	LARGA	2	150	200	450	600
DISTANCIA	MUY_LARGA	3	300	400	1000	1500
EDADT	APRENDIZ	0	16	16	20	26
EDADT	JOVEN	1	18	22	30	35
EDADT	MADURO	2	25	32	45	50
EDADT	MAYOR	3	40	45	55	60
EDADT	MUY_MAYOR	4	50	55	62	70
EXPERIENCIAT	POCA	0	0	1	2	4
EXPERIENCIAT	ALGUNA	1	2	3	5	6
EXPERIENCIAT	MUCHA	2	5	7	10	12
EXPERIENCIAT	BASTANTE	3	7	8	15	20
EXPERIENCIAT	GRANDE	4	12	15	50	50
SUELDO	BAJO	0	50000	55000	80000	120000
SUELDO	MEDIO	1	70000	90000	170000	200000
SUELDO	ALTO	2	160000	180000	250000	300000
SUELDO	MUY_ALTO	3	200000	240000	1000000	1000000

Tabla A.13: Contenido de la tabla Fuzzy_Label_Def relativo a la BD ejemplo

COLUMN_NAME	LABEL1_NAME	LABEL2_NAME	LABEL1_ID	LABEL2_ID	DEGREE
ESTUDIOST	EGB	BACHILLER	0	1	.5
ESTUDIOST	EGB	LICENCIADO	0	2	.2
ESTUDIOST	EGB	LINCENCIADO	0	2	.2
ESTUDIOST	EGB	DOCTOR	0	3	.1
ESTUDIOST	BACHILLER	LICENCIADO	1	2	.6
ESTUDIOST	BACHILLER	LINCENCIADO	1	2	.6
ESTUDIOST	BACHILLER	DOCTOR	1	3	.3
ESTUDIOST	LICENCIADO	DOCTOR	2	3	.8
ESTUDIOST	LINCENCIADO	DOCTOR	2	3	.8
PROFT	DIRECTOR	INGENIERO	0	1	.8
PROFT	DIRECTOR	TECNICO	0	2	.7
PROFT	DIRECTOR	ADMINISTRATIVO	0	3	.6
PROFT	DIRECTOR	SECRETARIO	0	4	.4
PROFT	DIRECTOR	VENDEDOR	0	5	.4
PROFT	INGENIERO	TECNICO	1	2	.8
PROFT	INGENIERO	ADMINISTRATIVO	1	3	.2
PROFT	INGENIERO	SECRETARIO	1	4	.2
PROFT	INGENIERO	VENDEDOR	1	5	.2
PROFT	TECNICO	ADMINISTRATIVO	2	3	.8
PROFT	TECNICO	SECRETARIO	2	4	.6
PROFT	TECNICO	VENDEDOR	2	5	.5
PROFT	ADMINISTRATIVO	SECRETARIO	3	4	.7
PROFT	ADMINISTRATIVO	VENDEDOR	3	5	.6
PROFT	SECRETARIO	VENDEDOR	4	5	.6
RENDIMIENOT	MALO	REGULAR	0	1	.8
RENDIMIENOT	MALO	BUENO	0	2	.5
RENDIMIENOT	MALO	EXCELENTE	0	3	.1
RENDIMIENOT	REGULAR	BUENO	1	2	.7
RENDIMIENOT	REGULAR	EXCELENTE	1	3	.5
RENDIMIENOT	BUENO	EXCELENTE	2	3	.8

Tabla A.14: Contenido de la tabla Fuzzy_Nearness_def relativo a la BD ejemplo

TABLE_NAME	COLUMN_NAME	BASE
APTITUD	EXPERIENCIAT	2
CIUDADES	DISTANCIA	25
DPTO	BENEFICIOS	10000000
EMPLEOS	COMISIONT	5000
EMPLEOS	SUELDO	10000
PERSONAL	EDADT	5
PUESTOS	EXPERIENCIAT	2

Tabla A.15: Contenido de la tabla Fuzzy_Aprox_def relativo a la BD ejemplo

En cuanto a la estructura de la BD del ejemplo, las tablas A.16, A.17, A.18, A.19, A.20 y A.21 recogen la forma en que dicha estructura se traduce a la representación de FIRST adoptada en Oracle.

COLUMN_NAME	TIPO
EMP#	CHAR(4)
NOMBRE	CHAR(20)
SEXO	CHAR(1)
EDADT	NUMBER(1)
EDAD1	NUMBER(3)
EDAD2	NUMBER(3)
EDAD3	NUMBER(3)
EDAD4	NUMBER(3)
CIUDAD	CHAR(15)

Tabla A.16: Estructura en Oracle de la tabla *Personal*

COLUMN_NAME	TIPO
EMP#	CHAR(5)
ESTUDIOS	NUMBER(1)
ESTUDIOS1	NUMBER(3)
PROFT	NUMBER(1)
PROF1	NUMBER(3)
EXPERIENCIAT	NUMBER(1)
EXPERIENCIA1	NUMBER(2)
EXPERIENCIA2	NUMBER(2)
EXPERIENCIA3	NUMBER(2)
EXPERIENCIA4	NUMBER(2)

Tabla A.17: Estructura en Oracle de la tabla *Aptitud*

COLUMN_NAME	TIPO
DPTO#	CHAR(4)
NOMBRE	CHAR(20)
LOCALIZ	CHAR(15)
BENEFICIOS	NUMBER(10)

Tabla A.18: Estructura en Oracle de la tabla *Dpto.*

COLUMN_NAME	TIPO
PUESTO#	CHAR(4)
NOMBRE	CHAR(20)
ESTUDIOST	NUMBER(1)
ESTUDIOS1	NUMBER(3)
PROFT	NUMBER(1)
PROF1	NUMBER(3)
EXPERIENCIAT	NUMBER(1)
EXPERIENCIA1	NUMBER(2)
EXPERIENCIA2	NUMBER(2)
EXPERIENCIA3	NUMBER(2)
EXPERIENCIA4	NUMBER(2)

Tabla A.19: Estructura en Oracle de la tabla *Puestos*

COLUMN_NAME	TIPO
EMP#	CHAR(5)
DPTO#	CHAR(4)
PUESTO#	CHAR(4)
SUELDO	NUMBER(7)
COMISIONT	NUMBER(1)
COMISION1	NUMBER(7)
COMISION2	NUMBER(7)
COMISION3	NUMBER(7)
COMISION4	NUMBER(7)
RENDIMIENOT	NUMBER(1)
RENDIMIENTO1	NUMBER(3)

Tabla A.20: Estructura en Oracle de la tabla *Empleos*

COLUMN_NAME	TIPO
CIUDAD1	CHAR(15)
CIUDAD2	CHAR(15)
DISTANCIA	NUMBER(4)

Tabla A.21: Estructura en Oracle de la tabla *Ciudades*

La tabla A.22 sirve de ejemplo para mostrar la forma en que quedan almacenados los datos de la tabla *Empleos* en la estructura correspondiente del RDBMS anfitrión.

• **Observaciones:**

- Las prioridades establecidas en esta implementación se dirigen a evaluar algunos de los aspectos relativos a la consulta “difusa”, por tanto se ha dejado para más adelante la implementación de un DDL “difuso”. Toda la información relativa al ejemplo mostrada en este apartado ha sido introducida “a mano”, es decir, utilizando el DDL de Oracle. No obstante, es preciso señalar que, una vez creada la FMB, el proceso de inserción de información en las “verdaderas” tablas de la base de datos, (las tablas creadas en el gestor anfitrión), no difiere del mecanismo adoptado para el caso clásico.
- Como dijimos anteriormente, para mostrar como quedan almacenados en la FMB los datos relativos al ejemplo, se han empleado una serie de vistas que presentan dicha información de una forma más visible. En esas vistas la denominación de algunos de los atributos que organizan la “metainformación” difiere ligeramente de la adoptada en el capítulo 5, no obstante la correspondencia existente es fácilmente identificable.
- La información mostrada aquí es la necesaria y suficiente para permitir la resolución, sobre el ejemplo empleado, de todos los procesos de consulta que se mostrarán en los próximos apartados.

EMP#	DT#	PTO#	SUELDO	COMT	COM1	COM2	COM3	COM4	RENTD	REND1
E0	D1	P5	60000	3	10000				3	1
E1	D1	P3	80000	3	5000				3	2
E2	D3	P1	200000	4	2				3	2
E3	D4	P3	120000	3	10000				3	2
E4	D0	P5	160000	4	3				3	3
E5	D2	P5	140000	3	30000				3	2
E6	D0	P4	120000	4	0				3	1
E7	D0	P0	500000	4	4				3	3
E8	D2	P5	90000	3	35000				3	2
E9	D1	P3	120000	6	0	5000	-5000	10000	3	2
E10	D3	P4	110000	4	1				3	2
E11	D1	P0	450000	4	4				3	2
E12	D3	P1	350000	3	40000				3	2
E13	D4	P0	475000	3	50000				3	2
E14	D3	P4	100000	4	1				3	2
E15	D2	P1	230000	7	50000	10000	-15000	100000	3	2
E16	D1	P3	170000	4	2				3	3
E17	D3	P1	180000	4	0				3	0
E18	D2	P2	160000	3	20000				3	2
E19	D2	P5	130000	4	3				3	3
E20	D0	P2	230000	4	2				3	2
E21	D3	P2	270000	6	20000	5000	-5000	30000	3	2
E22	D1	P4	85000	4	1				3	2

Tabla A.22: Representación en Oracle de los datos de la tabla *Empleos*

A.4 El Procesador de Llamadas Difusas (FCP)

Como se dijo en el capítulo 5, el FCP se encarga de traducir las peticiones “difusas” a sentencias SQL clásicas directamente ejecutables por el RDBMS anfitrión. Dicho proceso de traducción se encuentra fuertemente ligado a la representación adoptada y al dialecto de SQL disponible en el RDBMS anfitrión. El primer aspecto ha quedado reflejado en los apartados anteriores. En cuanto al dialecto de SQL recogido en la versión de Oracle empleada, una descripción exhaustiva puede encontrarse en [Oracle]. En el caso que nos ocupa hemos procurado confeccionar las rutinas del FCP empleando las características estándar de SQL. No obstante, ha sido preciso recurrir al empleo de algunas funciones específicas del citado dialecto para elaborar algunas de las mencionadas rutinas. Particularmente poderosas se han mostrado las funciones DECODE y SIGN del SQL de Oracle. Mediante estas funciones ha sido posible construir sentencias que condensan los pasos enunciados en el algoritmo 5.1 en la pag. 198. Dicho algoritmo adopta la siguiente forma:

Algoritmo A.1 Consta de los siguientes pasos:

1. Elaborar una condición compuesta por la disyunción (OR) sobre un conjunto de condiciones elementales que restrinjan las tuplas que satisfacen la condición “difusa” para cada tipo de dato soportado. La distinción entre los diferentes tipos de valores para un atributo “difuso” se realiza mediante la condición clásica: $F_TYPE=t$, donde, F_TYPE es el campo que contiene el tipo de dato que almacena cada tupla para el atributo difuso F y t es el código para cada tipo según muestran las tablas 5.1 y 5.2.
2. En base a la implementación para atributos difusos mostrada en las tablas 5.1 y 5.2, construir una sentencia basada en la función DECODE que obtenga, para cada tipo de dato consultado, la presentación adoptada en orden a visualizarlo con un formato concreto.
3. Expresar mediante las funciones SQL (DECODE y SIGN) la sentencia SQL clásica que realiza el cálculo de los “grados de compatibilidad” atendiendo al tipo de dato considerado.

El primer paso del algoritmo anterior hace referencia al proceso de transformación de una condición difusa a una condición SQL clásica, el segundo a la elaboración de una sentencia que reconstruya la información presente en la base de datos Oracle y la muestre según un formato determinado y, el tercero alude al mecanismo mediante el cual construir una sentencia que efectúe, en ejecución, el cálculo de los “grados de compatibilidad” correspondientes. Los pasos mencionados pueden ser necesarios o no, dependiendo de los requisitos expresados en la consulta. Así, si no se establece un condición “difusa” en la consulta y sólo se nos pide mostrar el contenido de atributo “difuso”, los pasos 1 y 3 serán innecesarios. De igual forma si la consulta no requiere el cálculo de los “grados de compatibilidad” el tercer paso puede ser obviado.

La FMB en la Estructura de Datos del Prototipo

Para que el FCP lleve a cabo su misión, precisa disponer de la información contenida en la FMB, relativa a los elementos que constituyen la sentencia a traducir. Dicha información ha sido necesaria antes, para realizar el proceso de validación semántica de la cláusula considerada, (sobre este aspecto discutiremos en el próximo apartado). Llegado a este punto, el FCP tiene que conocer las definiciones pertinentes a los items “difusos” envueltos en la cláusula “difusa”. Existen varias alternativas para facilitar dicha información al FCP:

- Realizar una búsqueda en la FMB conforme esta información es requerida, e incorporarla a una estructura de datos dinámica directamente procesable desde el lenguaje anfitrión, (C en el caso que nos ocupa).
- Mantener una copia de la FMB en una estructura de datos dinámica de forma que sea en todo momento reflejo del contenido de la FMB. En el momento de la ejecución del programa de acceso a FIRST, se vuelca toda la información en dicha estructura y durante el proceso se actualiza su contenido así como el de la FMB de acuerdo con las modificaciones que se produzcan.

En el prototipo que estamos describiendo, nos hemos decantado por una variante restringida de la segunda alternativa. Habida cuenta el carácter experimental del mismo, no se ha contemplado la actualización de acuerdo con las variaciones que las operaciones de manipulación puedan producir en la FMB, por otra parte, de momento

no se encuentran implementadas dichas operaciones. Así pues, desde el momento en que entramos en el interprete de comandos de FIRST volcamos el contenido de la FMB en una estructura de datos dinámica.

A.4.1 Funciones del FCP Implementadas

Actualmente el FCP tiene implementadas las funciones necesarias para realizar selecciones basadas en el comparador “Igual a” para atributos de Tipo 1, 2 y 3, con valores constantes de tipo “etiqueta”, notado en 5.3.4.2 como FEQ(<campo_difuso>, <etiqueta>, <umbral>). La mecánica empleada para desarrollar las funciones implicadas se basa en el esquema presentado en las tablas 5.11, 5.13, 5.14 y 5.19. Seguidamente daremos una breve descripción de las funciones diseñadas al efecto.

Funciones de Conversión de Condición Difusa a Condición Clásica

Existen tres funciones, una por cada tipo de atributo, que construyen la sentencia SQL clásica que lleva a cabo la condición difusa solicitada. Además, existen una serie de funciones auxiliares que realizan cálculos adicionales necesarios para dichas funciones. Las funciones de conversión son: `void feq1labcond()`, `void feq2labcond()` y `void feq3labcond()`. Veamos en ese orden como operan:

- La estructura de la primera función es:

```
void feq1labcond(cname,stmchg,a,b)
char *cname,*stmchg,*a,*b;
```

donde, `cname` es un puntero a una cadena de caracteres que identifica el atributo sobre el que se evalúa la condición difusa, `stmchg` es un puntero a la cadena que contendrá la sentencia SQL resultante de evaluar la condición difusa y, `a` y `b` son punteros a cadenas que contienen los alfacortes, determinados, sobre la etiqueta lingüística aportada en la condición difusa, por umbral establecido en la misma (ver fig. A.7).

Con esos parámetros de entrada, la función devuelve en `stmchg` la cadena “`cname BETWEEN a AND b`”.

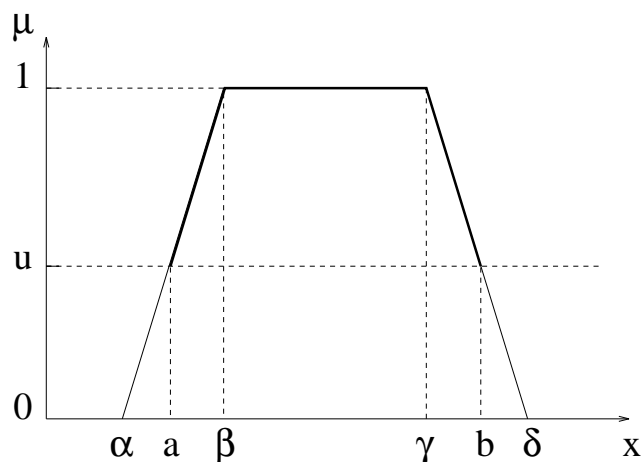


Figura A.7: Alfacortes para un umbral u

Ejemplo A.1 Para la condición $\text{FEQ}(\text{sueldo}, "alto", 0.7)$ sobre la tabla *Empleos*, el resultado de la anterior función es la cadena:

“SUELDO BETWEEN 174000 AND 265000”

■

- La segunda función presenta la siguiente estructura:

```
void feq2labcond(cname,newnear,stmchg,a,b,i1,i2,mu)
char *cname;
id_near *newnear;
char *stmchg;
char *a,*b,*i1,*i2;
float mu;
```

donde, *cname* contiene el nombre del atributo, *col* es un puntero a una estructura que recoge la información difusa relacionada con el atributo *cname*, *newnear* recoge información de aquellas etiquetas, tales que presentan un grado de compatibilidad con la consulta, superior o igual al umbral, *mu*, exigido, *stmchg* es un puntero a la cadena que devuelve el fragmento de sentencia convertida, *a* y *b* son los alfacortes e, *i1* e *i2* son valores del dominio particulares para la consulta con

datos tipo "aprox.". Los principios en que basa su funcionamiento se encuentran esquematizados en las tablas 5.13 y 5.14.

Para ver como opera nos valdremos del siguiente ejemplo:

Ejemplo A.2 Sea la condición difusa $FEQ(edad, "maduro", 0.8)$ sobre la tabla *Personal*, la sentencia clásica que genera la función sobre esta condición es:

“(EDADT=0 OR EDADT=2 OR EDADT=3 AND EDAD1 BETWEEN 30.6 AND 46 OR EDADT=4 AND EDAD1 IN (1,2,3) OR EDADT=5 AND EDAD1 <= 46 AND EDAD4 >= 30.6 OR EDADT=6 AND EDAD4 BETWEEN 34.6 AND 52 OR EDADT=7 AND EDAD4 >= 30.6 AND EDAD1 <= 46 AND $0.80*EDAD3+EDAD4 >= 30.6$ AND $0.80*EDAD2+EDAD1 <= 46$)”

■

- La tercera tiene la cabecera:

```
void feq3labcond(cname,newnear,stmchg)
char *cname;
id_near *newnear;
char *stmchg;
```

donde los parámetros mostrados tienen el mismo significado que en las anteriores funciones. El esquema de su funcionamiento se encuentra ilustrado en la tabla 5.19.

Ejemplo A.3 Para la condición $FEQ(rendimiento, "bueno", 0.5)$ sobre la tabla *Empleos*, la función devuelve la siguiente cadena SQL:

“RENDIMIENTO1 IN (0,1,2,3)”

donde 0,1,2,3 identifican a las etiquetas definidas con esos códigos en la FMB.

■

Funciones de Formato

Los datos difusos tienen en el RDBMS anfitrión la representación interna mostrada en las tablas 5.1 y 5.2, (págs. 178 y 180 resp.), en la tabla A.22 puede verse un ejemplo de su empleo. Para hacer legible dicha información es preciso adoptar un formato de salida de acuerdo con un patrón determinado. En el caso que nos ocupa, hemos adoptado el formato que muestra la tabla A.23. Las funciones que elaboran las sentencias SQL, capaces de proporcionar el mencionado formato, para los diferentes tipos de atributos, son: `void fuz_col_format()` para atributos de tipo 2 y `void fuz_col_format3()` para atributos de tipo 3. No es necesario adecuar la representación interna a la salida para atributos de tipo 1, puesto que ambas son de tipo numérico simple.

TIPO DE DATO	FORMATO SALIDA
UNKNOWN	UNKNOWN
UNDEFINED	UNDEFINED
NULL	NULL
CRISP	número
LABEL	nombre_etiqueta
INTERVALO[A,B]	[A,B]
APROD(d)	[d-margen,d,d+margen]
DIFUSO	$[\alpha, \beta, \gamma, \delta]$

Tabla A.23: Formato de salida para datos difusos

Seguidamente estudiamos de cerca las dos funciones anteriormente mencionadas.

- La primera de ellas presenta la siguiente cabecera:

```
void fuz_col_format(col,cname,nnear,lselect)
column *col;
char *cname;
id_near *nnear;
char *lselect;
```

donde, `col` es un puntero a una estructura que almacena toda la información existente en la FMB acerca del atributo apuntado por `cname`, `nnear` es un puntero a una lista que contiene los identificadores de todas las etiquetas definidas sobre `cname` y `lselect` es un puntero que recoge la sentencia SQL generada.

El siguiente ejemplo muestra su funcionamiento.

Ejemplo A.4 La salida que genera la función para el atributo de tipo 2 “experiencia” de la relación *Puestos* es:

```
“DECODE(EXPERIENCIAT, 0, 'UNKNOWN', 1, 'UNDEFINED', 2, 'NULL',
3, TO_CHAR( EXPERIENCIA1), 4, DECODE( EXPERIENCIA1, 0, 'POCA',
1, 'ALGUNA', 2, 'MUCHA', 3, 'BASTANTE', 4, 'GRANDE'), 5, '['|| TOCHAR(
EXPERIENCIA1)||', '||TO_CHAR( EXPERIENCIA4)||']', 6, '['||TO_CHAR( EX-
PERIENCIA1)||', '||TO_CHAR( EXPERIENCIA1+ EXPERIENCIA2)||', '||TO-
CHAR( EXPERIENCIA4)||']', 7, '['||TO_CHAR( EXPERIENCIA1)||', '||TO-
CHAR( EXPERIENCIA2 + EXPERIENCIA1 )||', '|| TO_CHAR( EXPERIEN-
CIA3+ EXPERIENCIA4)||', '||TO_CHAR( EXPERIENCIA4)||']”
```

donde, la función `DECODE` es empleada para determinar el formato a utilizar dependiendo del valor que tome el atributo `EXPERIENCIAT` para cada tupla, `TO_CHAR` es una función que pasa un dato de tipo numérico a tipo carácter para poder ser concatenado mediante el operador concatenación, `||`, de SQL.

■

- La cabecera de la segunda función es:

```
void fuz_col_format3(col,cname,nnear,lselect)
column *col;
char *cname;
id_near *nnear;
char *lselect;
```

donde los parámetros tienen el mismo significado que en la sentencia anterior.

Ejemplo A.5 Sea el atributo Estudios de la tabla *Aptitud*, el resultado de aplicar la función anterior es:

```
“DECODE(ESTUDIOS1, 0, 'EGB', 1, 'BACHILLER', 2, 'LINCENCIADO', 3, 'DOCTOR' )”
```

Funciones para la Obtención de los Grados de Compatibilidad

El cálculo de los grados de compatibilidad para un atributo “difuso”, mediante una única sentencia SQL clásica, requiere del empleo de una estrategia facilitada por la utilización combinada de las funciones DECODE y SIGN de SQL. Si ambas funciones se utilizan de forma adecuada, es posible construir estructuras selectivas SQL, que operen en la cláusula SELECT, para aplicar diferentes expresiones a cada tupla, dependiendo del valor que tome para determinados atributos. En las funciones que describiremos en este punto se hace un empleo profuso de esta técnica. Las funciones que obtienen las sentencias que evalúan los grados de compatibilidad son: `void feq1labcddeg()` para atributos de tipo 1, `void feq2labcddeg()` para atributos de tipo 2 y `void feq3labcddeg()` para atributos de tipo 3.

- La primera de ellas presenta la siguiente cabecera:

```
void feq1labcddeg(cname,col,stmchg,litem,a,b,bet_alf,gam_del)
char *cname;
column *col;
char *stmchg;
int litem;
char *a,*b,*bet_alf,*gam_del;
```

donde, `bet_alf`, y `gam_del` son punteros a cadenas que contienen el cálculo $\beta - \alpha$ y $\gamma - \delta$ para los parámetros de la etiqueta, sobre la que se evalúa la condición difusa, la sentencia generada se recoge en `stmchg`. Veamos en el siguiente ejemplo la sentencia que genera esta función

Ejemplo A.6 Sea `FEQ(distancia,"larga",0.7)` una condición difusa expresada para la tabla *Ciudades*, la función genera la siguiente sentencia SQL para calcular los grados de compatibilidad sobre el atributo “distancia”:

“ROUND(DECODE(SIGN((DISTANCIA-185) * (495 - DISTANCIA)), -1, 0, DECODE(SIGN(200 - DISTANCIA), 1, (DISTANCIA - 150)/ 50, DECODE(SIGN(DISTANCIA - 450), 1, (DISTANCIA - 600)/ -150, 1))), 2)”

donde la función ROUND(*expr*,2) redondea la parte fraccionaria de *expr* a dos cifras decimales.

■

Como puede verse, en esta sentencia se hace un uso intensivo de las funciones DECODE y SIGN. Cada parte proporciona una expresión diferente para el cálculo de los grados de compatibilidad. Así, la expresión SIGN((DISTANCIA-185) * (495 - DISTANCIA)) puede evaluarse como -1, 0 ó 1 según sea negativo, cero o positivo el argumento de la función SIGN. Si, para un valor determinado del atributo DISTANCIA, se obtiene un resultado negativo para esa expresión, significa que ese valor es: o menor que 185 o mayor que 495, por lo que se sale del rango de validez de la etiqueta “larga” para el umbral establecido, por este motivo, la función DECODE hace que se devuelva un 0. Si no ocurre así, nos encontramos en la parte de la sentencia, ..., DECODE(SIGN(200 - DISTANCIA), 1, ... que evalúa si el valor asumido por DISTANCIA es menor que 200, en cuyo caso el grado de compatibilidad se calcula mediante la expresión (DISTANCIA - 150)/ 50. Si esto no ocurre tampoco, el fragmento ..., DECODE(SIGN(DISTANCIA - 450), ... comprueba si el valor asumido por DISTANCIA es mayor que 450, y en ese caso el grado de compatibilidad se calcula mediante la expresión (DISTANCIA - 600)/ -150. Por último, si no ha ocurrido ninguna de las anteriores condiciones nos encontramos con que DISTANCIA asume un valor comprendido entre 200 y 450 por lo que el grado de compatibilidad vale 1.

- La cabecera para la segunda función es:

```
void feq2labcddeg(cname,col,newnear,stmchg,litem,a,b,i1,i2,bet_alf,
                 gam_del,mu)
char   *cname;
column *col;
id_near *newnear;
```

```

char *stmchg;
int litem;
char *a,*b,*i1,*i2,*bet_alf,*gam_del;
float mu;

```

donde todos los parámetros presentes tiene significado conocido.

La sentencia que genera esta función es bastante más compleja que la que genera la función anteriormente estudiada, esto es así porque los atributos de tipo 2 admiten mayor variedad de tipos “difusos”. De todas formas la estrategia empleada para generar la sentencia SQL que efectúa el cálculo es la misma, el siguiente ejemplo muestra los resultados que es capaz de proporcionar la función que estamos estudiando.

Ejemplo A.7 Consideremos la condición $FEQ(edad, "mayor", 0.8)$ evaluada sobre la tabla *Personal*, la sentencia SQL, que compone la función analizada, para obtener los grados de compatibilidad con que se satisface dicha condición es:

```

"ROUND( DECODE( EDADT, 0, 1, 2, 1, 3, DECODE( SIGN( ( EDAD1 - 44)
* (56 - EDAD1) ), -1, 0, DECODE( SIGN( 45 - EDAD1), 1, (EDAD1 - 40)/
5, DECODE( SIGN( EDAD1 - 55), 1, ( EDAD1 - 60)/ -5,1 ) ) ), 4, DECODE(
EDAD1, 2, 1.00, 3, 1.00, 4, 1.00, 0), 5, DECODE( SIGN( ( EDAD4 - 44) *
(56 - EDAD1) ), -1, 0, DECODE( SIGN( 45 - EDAD4), 1,( EDAD4 - 40)/ 5,
DECODE( SIGN( EDAD1 - 55), 1,( EDAD1 - 60)/ -5, 1 ) ) ), 6, DECODE(
SIGN( ( EDAD4 - 48) *(62 - EDAD4) ), -1, 0, DECODE( SIGN( 45 - EDAD4
+ 5), 1,( EDAD4 - 40)/ (5 +5) ), DECODE( SIGN( 55 - EDAD4 + 5), 1, 1,(
EDAD1-60 )/( -5-5) ) ) ), 7, DECODE( SIGN( ( 0.80 * EDAD3 + EDAD4 -44) *
(56 - 0.80 * EDAD2 + EDAD1) ), -1, 0, DECODE( SIGN( ( EDAD3 + EDAD4
- 45) * (55 - EDAD2 + EDAD1) ), -1, DECODE( SIGN( (0.80 * EDAD3 +
EDAD4 -44) * ( 45 - EDAD3 + EDAD4) ), -1, DECODE( SIGN( (56 - 0.80 *
EDAD2 + EDAD1) * (EDAD2 + EDAD1 - 55) ), 1, (60 - EDAD1)/ ( EDAD2 -
5) ),( EDAD4 - 40)/( 5 - EDAD3) ), 1) ), 0), 2)"

```

La filosofía adoptada para obtener la sentencia anterior está basada en lo expuesto en el apartado 5.3.4.2 y esquematizado en las tablas 5.13 y 5.14. En próximos apartados veremos el resultado de ejecutar las sentencias generadas de esta forma.

- La función escrita para atributos de tipo 3 tiene la siguiente cabecera:

```
void feq3labcddeg(cname,newnear,stmchg)
char *cname;
id_near *newnear;
char *stmchg;
```

Esta función extrae de la FMB los grados de semejanza que presentan aquellas etiquetas, que son compatibles con la etiqueta argumento, en un grado superior o igual al umbral establecido.

Ejemplo A.8 Dada la condición $\text{FEQ}(\text{prof}, \text{"ingeniero"}, 0.8)$, la función genera la siguiente sentencia SQL:

“ $\text{DECODE}(\text{PROF1}, 0, 0.80, 1, 1.00, 2, 0.80, 0)$ ”

■

Funciones Auxiliares

Para cumplir con su cometido, el FCP necesita de un conjunto de funciones auxiliares que calculan parámetros o dimensionan variables. De ellas, las más importantes son:

- Funciones que calculan la dimensión de las cadenas generadas por las funciones del FCP:
 - `int feq1labcond_alloc()` y `int feq1labcddeg_alloc()`, que devuelven el número de caracteres necesitados para almacenar la salida de las funciones `void feq1labcond()` y `void feq1labcddeg`, respectivamente.
 - `int feq2labcond_alloc()`, `int fuz_col_alloc()` y `int feq2labcddeg_alloc()`, para dimensionar las cadenas generadas por las funciones `void feq2labcond()`, `void fuz_col_format()` y `void feq2labcddeg()`, respectivamente.

- `int feq3labcond_alloc()`, `int fuz_col_alloc3()` y `int feq3labcddeg_alloc()` para las funciones `void feq3labcond()`, `void fuz_col_format()` y `void feq3labcddeg()`, respectivamente.
- `void select_id()` , esta función determina las etiquetas que son compatibles con una etiqueta dada a partir del umbral establecido. También realiza cálculos del número de caracteres que ocupan esas etiquetas y sus grados.
- `void calcula_cte()` realiza cálculos en el RDBMS tendentes a obtener los valores de las constantes `a`, `b`, `i1`, `i2`, `bet_alf` y `gam_del`, empleados en las funciones descritas en este subapartado.

A.5 Un Procesador Elemental de FSQL

Para completar los elementos que constituyen este prototipo, se ha desarrollado un procesador de SQL “difuso” que soporta una sintaxis híbrida, que integra en el SQL de Oracle un conjunto de cláusulas y funciones, que permiten resolver consultas “difusas”. Así pues, mediante el procesador de FSQL implementado, es posible realizar consultas que involucren cláusulas “difusas” y cláusulas clásicas.

Las nuevas cláusulas “difusas” que incorpora este procesador tienen la siguiente sintaxis (en formato YACC):

```
fcond_simp  :   fcond_sin umbral
              |   fcond_sin
              ;

fcond_sin   :   column_item fcomp fexpr
              ;

umbral      :   THOLD NUMERO
              ;

fexpr       :   '$' ID
              |   NUMERO
              |   '#' NUMERO
```

```

        |    '[' NUMERO ',' NUMERO ']'
        |    '$' '[' NUMERO ',' NUMERO ',' NUMERO ',' NUMERO ']'
        ;

fcomp      :    FEQ
        |    FGT
        |    FLT
        |    FGEQ
        |    FLEQ
        ;

grado_comp :    CDEG '(' column_item ')'
        ;

column_item :    id_usuario '.' id_tabla '.' id_atributo
        |    id_tabla '.' id_atributo
        |    tabla_alias '.' id_atributo
        |    id_atributo
        ;

```

donde, los símbolos no terminales `id_usuario`, `id_tabla`, `id_atributo` y `tabla_alias`, son identificadores de los objetos: usuario, tabla, atributo y alias de tabla, respectivamente, y siguen las reglas de formación establecidas en la sintaxis de Oracle. El símbolo terminal `NUMERO` identifica en la sintaxis a un dato con formato numérico, el analizador semántico se encargará de comprobar el rango y tipo para cada ocurrencia en la entrada. Los símbolos entre comillas simples son literales que pertenecen a la regla de producción. Los símbolos terminales restantes son palabras reservadas de la nueva sintaxis que permiten expresar condiciones, modificadores y funciones “difusas”. Las siguientes frases serían sentencias válidas de la sintaxis de FSQL:

```
SELECT edad, CDEG(edad) FROM personal WHERE edad FEQ $joven THOLD 0.8
```

```
SELECT a.emp#, nombre, estudios, CDEG(estudios)
```

```
FROM personal a, aptitud b
WHERE a.emp#=b.emp# AND estudios FEQ $licenciado THOLD 0.6

SELECT emp#,dpto#,puesto#,sueldo,CDEG(sueldo),comision,CDEG(comision)
FROM empleos WHERE sueldo FEQ $alto AND comision FEQ $baja THOLD 0.8
```

El símbolo no terminal `fcomp` representa la batería de comparadores difusos contemplados en la sintaxis de FSQL. La sintaxis completa de una comparación difusa atómica tiene la forma:

```
<atributo> fcomp <cte_difusa> [THOLD <umbral>]
```

donde los corchetes indican que el umbral puede aparecer o no en la condición, en este último caso, se toma un valor umbral por defecto que, en el caso que nos ocupa, vale 0.5.

Las condiciones compuestas se obtienen mediante el empleo de los conectivos NOT, AND y OR, pudiéndose conectar condiciones atómicas “difusas” con otras que no lo sean.

Para visualizar los grados de compatibilidad que presenta un atributo determinado se emplea la sintaxis:

```
CDEG(<atributo>)
```

donde, `<atributo>` identifica al atributo del que queremos presentar los grados de compatibilidad.

A.5.1 Funcionamiento del Procesador de FSQL

La figura A.8 muestra el esquema del funcionamiento del procesador de FSQL. En dicha figura se muestran los módulos implicados en el procesado de una sentencia expresada en términos de FSQL, desde la entrada, hasta la obtención de los resultados, si los hubiera, en la salida.

Ayudados por la figura A.8 vamos recorrer el procesado de una sentencia FSQL a través del Procesador de FSQL.

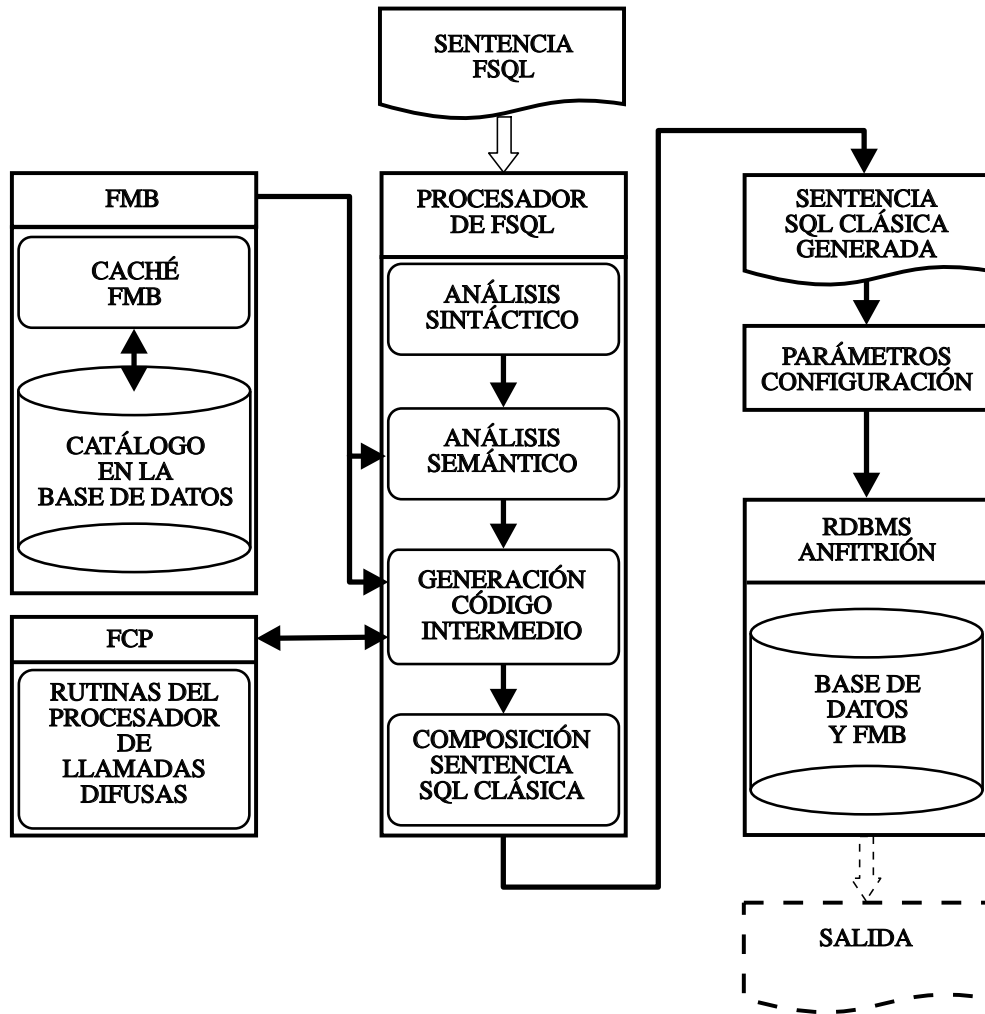


Figura A.8: Esquema del Procesado de una Sentencia FSQL

1. Dada una sentencia supuestamente expresada en FSQL, de acuerdo con la sintaxis introducida en el subapartado anterior, lo primero que hace el PFSQL, (Procesador de FSQL, como lo llamaremos en adelante), es determinar si adopta la sintaxis adecuada. Esto se realiza mediante el analizador sintáctico incorporado.
2. Simultáneamente a este proceso, el PFSQL, mediante el analizador semántico, se encarga de comprobar que todos los elementos “difusos” implicados se encuentran correctamente empleados. Para ello recurre a la información presente en la FMB acerca de dichos elementos.
3. Conforme se cumplimentan los procesos anteriores, el PFSQL vá generando una estructura de datos que organiza la información de partida en dos grupos de proceso:
 - El grupo de cláusulas presentes en la sentencia original que no precisan tratamiento “difuso”. Sobre este grupo no se realizará ninguna operación especial.
 - Y, el grupo relativo a las cláusulas “difusas” inmersas en la sentencia de entrada.

A partir de este último grupo, el PFSQL extrae de la FMB y de las propias cláusulas “difusas”, los parámetros que se encuentren definidos al respecto, y los envía al FCP, para que genere los fragmentos de sentencias SQL correspondientes a la función requerida por dichas cláusulas. Toda la información generada queda convenientemente organizada mediante la mencionada estructura de datos.

4. Una vez realizados los pasos anteriores, el PFSQL, a partir de la información de que dispone, está en condiciones de reconstruir la sentencia SQL final.
5. La sentencia generada es directamente ejecutable por el RDBMS anfitrión, por lo que puede ser enviada al mismo para su inmediato procesado. No obstante, su ejecución podría verse modificada mediante la adición de una serie de parámetros de configuración que influyan sobre el formato de salida. Dichos parámetros pueden tener una fuerte dependencia del sistema empleado para ejecutarla, por lo que no entraremos en ellos.

Dependiendo del tipo de sentencia procesada, DDL, DCL o DML, la ejecución de la sentencia generada puede, o no, generar una salida, por este motivo dicha acción se encuentra representada mediante línea discontinua en la figura A.8. Cuando se trata de sentencias DDL y DCL, es posible que la FMB sufra alteraciones en su contenido de acuerdo con las modificaciones introducidas.

En los pasos anteriormente descritos están contemplados mecanismos de detección de errores que funcionan a varios niveles. A nivel del analizador sintáctico, el PFSQL detecta errores sintácticos en la entrada, aunque, por el momento, no incorpora un gestor de errores demasiado explicativo acerca del origen de los mismos. Los errores semánticos son localizados, en primera instancia, cuando tienen su origen en el incorrecto empleo de items “difusos”, posteriormente, en el proceso de ejecución por parte del RDBMS anfitrión, serán detectados el resto de los errores semánticos. También el gestor anfitrión será el encargado de detectar posibles errores de ejecución, por lo que habrá que remitirse a los códigos de error generados por el mismo.

A.5.2 Rutinas del Procesador de FSQL

De acuerdo con el esquema de procesado presentado, el PFSQL precisa de una serie de rutinas que implementan los módulos que lo componen. En este subapartado, daremos una breve descripción de las mismas agrupadas por módulos.

Análisis Sintáctico

El análisis sintáctico ha sido implementado mediante la ayuda de un par de herramientas proporcionadas por el S.O. UNIX: el generador de analizadores léxicos LEX y el generador de analizadores sintácticos YACC. El primero, en base a una entrada expresada según un formato específico, genera un fichero en C, el cual, debidamente compilado, permite construir un módulo que analice la entrada detectando los “tokens” establecidos en la sintaxis adoptada. El segundo, que suele actuar en combinación con LEX, genera un analizador sintáctico a partir de un fichero de entrada, que expresa la sintaxis a reconocer por dicho generador, en términos parecidos a la notación BNF. YACC permite, además, la inclusión en las reglas de producción, de bloques de código C, que realicen determinadas operaciones, de acuerdo con la resolución de dichas reglas. Esta característica ha sido especialmente útil para llevar a cabo el análisis semántico

y la construcción de la estructura de datos que representa la sentencia procesada.

De acuerdo con todo esto, la realización del prototipo a estudio, ha precisado del desarrollo de sendos ficheros, con las especificaciones de la sintaxis adoptada para FSQL, así como, de los “tokens” reconocidos por el analizador léxico. El primero de estos ficheros, incorpora numerosas llamadas a una serie de rutinas, que han sido diseñadas con objeto de llevar a cabo, el análisis semántico, la construcción de la representación interna de la sentencia procesada, la generación del código SQL necesario y el enlace de la sentencia SQL final.

Funciones del Analizador Semántico

Como hemos indicado anteriormente, el análisis semántico es una tarea que el PF-SQL acomete unicamente para los items “difusos”. Los elementos semánticos de la sentencia de entrada que no reciben tratamiento “difuso” son analizados posteriormente por el analizador del gestor anfitrión. En este sentido, el prototipo que estamos describiendo, posee un conjunto de funciones, que se encargan de determinar la validez de las operaciones “difusas”, a que son sometidas los diferentes elementos considerados. También se ha previsto la generación de mensajes de error, de acuerdo con los fallos semánticos detectados. Someramente, las rutinas más importantes encargadas de tales menesteres son:

- `column *busca_column();`

Esta rutina busca en la caché de la FMB si el atributo, que se le pasa como argumento, se cataloga como “difuso”. Caso de ser así, devuelve un puntero a una estructura, que almacena toda la información disponible en la FMB relativa a dicho atributo, en caso contrario devuelve NULL.

- `int busca_etiq();`

Esta función determina si la etiqueta que se le pasa como argumento está definida sobre el atributo que también es pasado como argumento. Si está definida, devuelve el identificador de dicha etiqueta en la estructura de datos, si no lo está, devuelve -1, que será interpretado como error.

- `void fuz_error();`

Esta función genera los correspondientes mensajes de error producidos en el análisis semántico. Como parámetros de entrada tiene el código generado durante el proceso de análisis. Según sea dicho código dispone de los siguientes mensajes de error:

1. “Expresion sobre atributo difuso no soportada”
2. “Referencia ambigua de atributo difuso”
3. “Comparacion difusa no soportada por atributo no difuso”
4. “Tipo no soportado en la version actual”
5. “Etiqueta no definida sobre atributo”
6. “Umbral fuera del rango [0,1]”
7. “Expresion difusa no soportada en la version actual”

Funciones para Generar la Representación y para Componer la Sentencia

Como hemos dicho anteriormente, el PFSQL genera una estructura de datos interna que representa la sentencia que está procesando. Dicha estructura tiene principalmente dos vertientes: una, orientada a organizar los diferentes fragmentos de cadenas de texto en que se divide la sentencia original, para facilitar la ulterior recombinación y, la otra, para organizar las diferentes acciones a que son sometidos los atributos “difusos”. Las funciones desarrolladas para gestionar la primera estructura de datos son:

- `void crea_nodo();`

Esta función crea un nuevo nodo en la lista que mantiene los diferentes fragmentos de sentencias que se van organizando. Como parámetros de entrada recibe los datos que caracterizan ese fragmento.

- `void libera_nodo();`

Esta rutina es empleada para liberar el espacio ocupado por el nodo que se le pasa como parámetro. De esta forma, una vez procesada la entrada, las estructuras auxiliares dejan disponible la memoria empleada.

Las funciones utilizadas para gestionar la segunda estructura de datos son:

- `void crea_fuz_column();`

Crea e inicializa un nodo en la lista que recoge información acerca de los atributos objeto de tratamiento “difuso”. Esta lista recoge toda la información que el PF-SQL precisa, para analizar semanticamente las cláusulas “difusas” y para enviar al FCP, los parámetros necesarios para generar los fragmentos SQL precisos para construir la sentencia final.

- `int act_cdeg();`

Esta función genera una estructura, que se añade aquellos nodos, que referencian atributos, para los que se requiere calcular el “grado de compatibilidad”. Esta función recoge la estructura de la condición compuesta a que fuera sometido dicho atributo.

- `void libera_fuz_column();`

Esta rutina realiza la misma función que `void libera_nodo();` para la estructura descrita en segundo lugar.

Además de las anteriores funciones, existen otras auxiliares que se encargan de crear y gestionar el árbol que almacena la estructura que representa una condición compuesta, (diferentes condiciones atómicas combinadas mediante los conectivos AND y OR). Dicha estructura es necesaria para generar la condición SQL equivalente y, sobre todo, para generar la sentencia que calcule los “grados de compatibilidad”. Las siguientes funciones se encargan de tales cometidos:

- `void crea_arbol();`

Crea un nodo en el árbol que organiza la información aludida anteriormente. Ese nodo posee toda la información necesaria para, posteriormente, reconstruir la sentencia SQL que calcule las condiciones y “grados de compatibilidad” “difusos”.

- `int organiza_arbol();`

Esta función reorganiza el árbol general, de forma que se subdivide en tantos subárboles, como atributos implicados en el cálculo de los “grados de compatibilidad”.

- `void libera_arb();`

Libera el espacio ocupado por el árbol una vez que este ya no es necesario.

A continuación, describimos un conjunto de funciones, que han sido desarrolladas para ensamblar los fragmentos de sentencias generados, para producir la sentencia global, que obtiene del RDBMS anfitrión los resultados requeridos.

- `int genera_clasica();`

Esta función invoca al FCP para obtener los fragmentos SQL que calculan los diferentes componentes expresados en la entrada. Para llamar a esta función, es preciso haber finalizado el proceso de análisis de la entrada, y haber construido las estructuras internas mencionadas al principio de este subapartado. Como resultado de su ejecución, obtenemos los fragmentos generados por el FCP debidamente acomodados en dichas estructuras. Esta función devuelve el número de caracteres que precisan los fragmentos generados.

- `void compone_sent();`

Esta función combina en una única cadena de caracteres todos los fragmentos provenientes de la entrada y los generados por el FCP.

El último paso que queda por cubrir en este proceso, es el de ejecutar la sentencia SQL generada en el gestor anfitrión. De esto se encarga la función `void ejecuta();`, la cual recibe como parámetro de entrada un puntero al literal que contiene dicha sentencia. La función, escrita en SQL inmerso en C, procede a la ejecución de la sentencia y muestra los resultados por salida estándar.

A.6 FIRST en Acción.

Para terminar con el estudio del prototipo de FIRST implementado, vamos a dedicar este apartado a describir su manejo e ilustrar su comportamiento, mediante la resolución de algunas consultas, formuladas sobre el ejemplo de base de datos introducido en el apartado A.2.

Con respecto a la ejecución del prototipo propiamente dicha, ésta no puede ser mas simple. El único requisito para su puesta en funcionamiento es que el servidor anfitrión

se encuentre “levantado”. Después, supuesto que disponemos de los permisos adecuados, ejecutaremos el comando `first`. Este comando puede verse alterado mediante el empleo del modificador `-v`, directiva esta, que provoca una ejecución del programa en la que se muestran las sentencias SQL generadas, antes de proceder a su ejecución.

Una vez que hemos ejecutado cualquiera de las versiones del mencionado comando, el programa procede a recuperar la FMB en una estructura dinámica. Esta situación se pone de manifiesto por el mensaje: “Leyendo información de la Base de Datos. Espere por favor...”. Cuando finaliza este proceso, muestra el mensaje: “Información leída”. A partir de este momento entramos en un intérprete de comandos con el “prompt” `SQLFIRST>`, el cual nos solicita la entrada de sentencias escritas de acuerdo con la sintaxis descrita en el apartado A.5. Cada entrada, finalizada con la orden “intro”, pone en marcha el PFSQL y provoca la ejecución de la sentencia requerida, (con las limitaciones expresadas a lo largo de este apéndice).

Para salir del intérprete se ejecutará el comando `EXIT`¹.

A.6.1 Ejemplos de Consultas

Como hemos indicado al principio de este apartado, resolveremos mediante este prototipo una serie de consultas formuladas sobre la base de datos ejemplo. Con estos ejemplos se pretende únicamente mostrar la operatividad, (restringida), del prototipo y, sobre todo, apuntar la potencialidad y viabilidad de desarrollos más complejos basados en las ideas que lo sustentan. Con esta idea en mente, estructuraremos los siguientes ejemplos de acuerdo con el esquema:

1. Consulta expresada en términos informales.
2. Expresión de la misma en términos de FSQL.
3. Sentencia SQL generada por el PFSQL.
4. Resultados obtenidos de su ejecución.

Consulta A.1 “Mostrar *emp#*, nombre, edad, (grado de compatibilidad) de aquel personal “joven” (umbral 0.8)”

¹El intérprete no distingue entre mayúsculas y minúsculas

En FSQL:

```
SELECT emp#, nombre, edad, CDEG(edad)
FROM personal
WHERE edad FEQ $joven THOLD 0.8
```

Salida SQL:

```
SELECT EMP# ,NOMBRE, DECODE(EDADT,0,'UNKNOWN',1,'UNDEFINED',2,
'NULL',3,TO_CHAR(EDAD1),4,DECODE(EDAD1,0,'APRENDIZ',1,'JOVEN',
2,'MADURO',3,'MAYOR',4,'MUY_MAYOR'),5,['||TO_CHAR(EDAD1)||',
' ||TO_CHAR(EDAD4)||'],'6,[' ||TO_CHAR(EDAD1)||', ' ||TO_CHAR(EDAD1+
EDAD2)||', ' ||TO_CHAR(EDAD4)||'],'7,[' ||TO_CHAR(EDAD1)||',
' ||TO_CHAR(EDAD2+EDAD1)||', ' ||TO_CHAR(EDAD3+EDAD4)||', ' ||TO_CHAR(
EDAD4)||'],'8) "EDAD", ROUND(DECODE(EDADT,0,1,2,1,3,DECODE(SIGN(
(EDAD1-21.2)*(31-EDAD1)),-1,0,DECODE(SIGN(22-EDAD1),1,(EDAD1-18)/4,
DECODE(SIGN(EDAD1-30),1,(EDAD1-35)/-5,1))),4,DECODE(EDAD1,0,0.80,
1,1.00,2,0.83,0),5,DECODE(SIGN((EDAD4-21.2)*(31-EDAD1)),-1,0,
DECODE(SIGN(22-EDAD4),1,(EDAD4-18)/4,DECODE(SIGN(EDAD1-30),1,
(EDAD1-35)/-5,1))),6,DECODE(SIGN((EDAD4-25.2)*(37-EDAD4)),-1,0,
DECODE(SIGN(22-EDAD4+5),1,(EDAD4-18)/(4+5),DECODE(SIGN(30-EDAD4+5),
1,1,(EDAD1-35)/(-5-5))),7,DECODE(SIGN((0.80*EDAD3+EDAD4-21.2)*
(31-0.80*EDAD2+EDAD1)),-1,0,DECODE(SIGN((EDAD3+EDAD4-22)*(30-EDAD2+
EDAD1)), -1,DECODE(SIGN((0.80*EDAD3+EDAD4-21.2)*(22-EDAD3+EDAD4)),
-1,DECODE(SIGN((31-0.80*EDAD2+EDAD1)*(EDAD2+EDAD1-30)),1,(35-
EDAD1)/(EDAD2--5)),(EDAD4-18)/(4-EDAD3)),1)),0),2) FROM PERSONAL
WHERE (EDADT=0 OR EDADT=2 OR EDADT=3 AND EDAD1 BETWEEN 21.2 AND
31 OR EDADT=4 AND EDAD1 IN (0,1,2) OR EDADT=5 AND EDAD1<=31 AND
EDAD4>=21.2 OR EDADT=6 AND EDAD4 BETWEEN 25.2 AND 37 OR EDADT=7
AND EDAD4>=21.2 AND EDAD1<=31 AND 0.80*EDAD3+EDAD4>=21.2 AND
0.80*EDAD2+EDAD1<=31)
```

Consulta A.2 “Mostrar persona, puesto, dpto, sueldo, (su grado compatibilidad) de aquellos empleos con sueldo “alto” (umbral 0.8) y comision “alta” (umbral 0.6)”

EMP_	NOMBRE	EDAD	CDEG(EDAD)
E0	VICTOR	APRENDIZ	.8
E2	OLGA	27	1
E3	JOSE	[25,30]	1
E5	MATILDE	[25,30,35]	1
E6	ANA	UNKNOWN	1
E9	FEDERICO	23	1
E10	ALVARO	JOVEN	1
E13	CRISTINA	MADURO	.83
E14	INES	JOVEN	1
E15	JULIAN	31	.8
E18	PALOMA	27	1
E19	FERNANDO	JOVEN	1

Tabla A.24: Resultado de la Consulta A.1

En FSQL:

```
SELECT p.nombre, pu.nombre, d.nombre, sueldo, CDEG(sueldo)
FROM empleos e, puestos pu, personal p, dpto d
WHERE e.emp#=p.emp# AND e.puesto#=pu.puesto# AND e.dpto#=d.dpto#
AND sueldo FEQ $alto THOLD 0.8 AND comision FEQ $alta THOLD 0.6
```

Salida SQL:

```
SELECT P.NOMBRE ,PU.NOMBRE ,D.NOMBRE ,SUELDO ,ROUND(Decode(SIGN(
(SUELDO-176000)*(260000-SUELDO)), -1,0,Decode(SIGN(180000-SUELDO),
1,(SUELDO-160000)/20000,Decode(SIGN(SUELDO-250000),1,(SUELDO-
300000)/-50000,1))),2) FROM EMPLEOS E ,PUESTOS PU ,PERSONAL P ,
DPTO D WHERE E.EMP# = P.EMP# AND E.PUESTO# = PU.PUESTO# AND
E.DPTO# = D.DPTO# AND SUELDO BETWEEN 176000 AND 260000 AND
(COMISIONT=0 OR COMISIONT=2 OR COMISIONT=3 AND COMISION1 BETWEEN
27200 AND 54000 OR COMISIONT=4 AND COMISION1 IN (2,3,4) OR
COMISIONT=5 AND COMISION1<=54000 AND COMISION4>=27200 OR
```

```
COMISIONT=6 AND COMISION4 BETWEEN 30200 AND 61000 OR COMISIONT=7
AND COMISION4>=27200 AND COMISION1<=54000 AND 0.60*COMISION3+
COMISION4>=27200 AND 0.60*COMISION2+COMISION1<=54000)
```

P.NOMBRE	PU.NOMBRE	D.NOMBRE	SUELDO	CDEG(SUELDO)
OLGA	INGENIERO	NORTE	200000	1
FELIPE	TECNICO	EXPORTACION	230000	1

Tabla A.25: Resultado de la Consulta A.2

Consulta A.3 *“Mostrar nombre, ciudad de origen, ciudad de trabajo, distancia (con su grado de compatibilidad) de aquellos empleados que trabajan en una ciudad a una distancia “larga” (umbral 0.6) de donde nacieron”*

En FSQL:

```
SELECT p.nombre, p.ciudad, d.localiz, c.distancia, CDEG(c.distancia)
FROM personal p,dpto d, empleos e, ciudades c WHERE e.emp#=p.emp#
AND e.dpto#=d.dpto# AND (p.ciudad=c.ciudad1 AND d.localiz=c.ciudad2
OR p.ciudad=c.ciudad2 AND d.localiz=c.ciudad1) AND c.distancia FEQ
$larga THOLD 0.6
```

Salida SQL:

```
SELECT P.NOMBRE ,P.CIUDAD ,D.LOCALIZ ,C.DISTANCIA ,ROUND(Decode(
SIGN((C.DISTANCIA-180)*(510-C.DISTANCIA)), -1,0,Decode(SIGN(200-
C.DISTANCIA), 1,(C.DISTANCIA-150)/50,Decode(SIGN(C.DISTANCIA-450),
1,(C.DISTANCIA-600)/-150,1))),2) FROM PERSONAL P ,DPTO D ,EMPLEOS
E ,CIUDADES C WHERE E.EMP# = P.EMP# AND E.DPTO# = D.DPTO# AND
(P.CIUDAD = C.CIUDAD1 AND D.LOCALIZ = C.CIUDAD2 OR P.CIUDAD =
C.CIUDAD2 AND D.LOCALIZ = C.CIUDAD1 ) AND C.DISTANCIA BETWEEN
180 AND 510
```

NOMBRE	CIUDAD_OR	LOCALIZ	DISTANCIA	CDEG(DIST)
SIXTO	GRANADA	MADRID	432	1
MERCEDES	GRANADA	MADRID	432	1
CANDIDO	GRANADA	MADRID	432	1
INES	VALENCIA	BARCELONA	352	1
SILVIA	VALENCIA	BARCELONA	352	1
FELIPE	VALENCIA	MADRID	350	1
RICARDO	VALENCIA	BARCELONA	352	1
SALVADOR	VALENCIA	MADRID	350	1
FEDERICO	VALENCIA	MADRID	350	1

Tabla A.26: Resultado de la Consulta A.3

Consulta A.4 *Mostrar nombre, prof, edad, exper., rend. (grados respectivos) de aquellos empleados que sean “jóvenes” (0.7) con “alguna” experiencia (0.5) y rendimiento “excelente” (0.5)”*

En FSQL

```
SELECT p.nombre, a.prof, p.edad, CDEG(p.edad),
a.experiencia, CDEG(a.experiencia), e.rendimiento,
CDEG(e.rendimiento)
FROM personal p, aptitud a, empleos e
WHERE p.emp#=a.emp# AND p.emp#=e.emp# AND p.edad
FEQ $joven THOLD 0.7 AND a.experiencia FEQ $alguna
THOLD 0.5 AND e.rendimiento FEQ $excelente THOLD 0.7
```

Salida SQL:

```
SELECT P.NOMBRE ,DECODE(A.PROF1,0,'DIRECTOR',1,'INGENIERO',
2,'TECNICO',3,'ADMINISTRATIVO',4,'SECRETARIO',5,'VENDEDOR')
,DECODE(P.EDAD1,0,'UNKNOWN',1,'UNDEFINED',2,'NULL',3,
TO_CHAR(P.EDAD1),4,DECODE(P.EDAD1,0,'APRENDIZ',1,'JOVEN',
2,'MADURO',3,'MAYOR',4,'MUY_MAYOR'),5,' '|TO_CHAR(P.EDAD1)
```

```

||', '||TO_CHAR(P.EDAD4)||']',6,'[ '||TO_CHAR(P.EDAD1)||', '||
TO_CHAR(P.EDAD1+P.EDAD2)||', '||TO_CHAR(P.EDAD4)||']',7,'[ '
||TO_CHAR(P.EDAD1)||', '||TO_CHAR(P.EDAD2+P.EDAD1)||', '||
TO_CHAR(P.EDAD3+P.EDAD4)||', '||TO_CHAR(P.EDAD4)||']')
"P.EDAD" ,ROUND(DECODE(P.EDADT,0,1,2,1,3,DECODE(SIGN((P.EDAD1
-20.8)*(31.5-P.EDAD1)), -1,0,DECODE(SIGN(22-P.EDAD1),1,
(P.EDAD1-18)/4,DECODE(SIGN(P.EDAD1-30),1,(P.EDAD1-35)/-5,
1))),4,DECODE(P.EDAD1,0,0.80,1,1.00,2,0.83,0),5,DECODE(
SIGN((P.EDAD4-20.8)*(31.5-P.EDAD1)), -1,0,DECODE(SIGN(22-
P.EDAD4),1,(P.EDAD4-18)/4,DECODE(SIGN(P.EDAD1-30),1,
(P.EDAD1-35)/-5,1))),6,DECODE(SIGN((P.EDAD4-24.3)*(38-
P.EDAD4)), -1,0,DECODE(SIGN(22-P.EDAD4+5),1,(P.EDAD4-18)/
(4+5),DECODE(SIGN(30-P.EDAD4+5),1,1,(P.EDAD1-35)/(-5-5))),
7,DECODE(SIGN((0.70*P.EDAD3+P.EDAD4-20.8)*(31.5-0.70*
P.EDAD2+P.EDAD1)), -1,0,DECODE(SIGN((P.EDAD3+P.EDAD4-22)*
(30-P.EDAD2+P.EDAD1)), -1,DECODE(SIGN((0.70*P.EDAD3+P.EDAD4-
20.8)*(22-P.EDAD3+P.EDAD4)), -1,DECODE(SIGN((31.5-0.70*
P.EDAD2+P.EDAD1)*(P.EDAD2+P.EDAD1-30)),1,(35-P.EDAD1)/
(P.EDAD2--5)),(P.EDAD4-18)/(4-P.EDAD3)),1)),0),2)
,DECODE(A.EXPERIENCIAT,0,'UNKNOWN',1,'UNDEFINED',2,'NULL',
3,TO_CHAR(A.EXPERIENCIA1),4,DECODE(A.EXPERIENCIA1,0,'POCA',
1,'ALGUNA',2,'MUCHA',3,'BASTANTE',4,'GRANDE'),5,'[ '||
TO_CHAR(A.EXPERIENCIA1)||', '||TO_CHAR(A.EXPERIENCIA4)||']',
6,'[ '||TO_CHAR(A.EXPERIENCIA1)||', '||TO_CHAR(A.EXPERIENCIA1+
A.EXPERIENCIA2)||', '||TO_CHAR(A.EXPERIENCIA4)||']',7,'[ '||
TO_CHAR(A.EXPERIENCIA1)||', '||TO_CHAR(A.EXPERIENCIA2+
A.EXPERIENCIA1)||', '||TO_CHAR(A.EXPERIENCIA3+A.EXPERIENCIA4)
||', '||TO_CHAR(A.EXPERIENCIA4)||']') "A.EXPERIENCIA"
,ROUND(DECODE(A.EXPERIENCIAT,0,1,2,1,3,DECODE(SIGN(
(A.EXPERIENCIA1-2.5)*(5.5-A.EXPERIENCIA1)), -1,0,DECODE(
SIGN(3-A.EXPERIENCIA1),1,(A.EXPERIENCIA1-2)/1,DECODE(SIGN(
A.EXPERIENCIA1-5),1,(A.EXPERIENCIA1-6)/-1,1))),4,DECODE(
A.EXPERIENCIA1,0,0.67,1,1.00,0),5,DECODE(SIGN((

```

```

A.EXPERIENCIA4-2.5)*(5.5-A.EXPERIENCIA1)), -1, 0, DECODE(
SIGN(3-A.EXPERIENCIA4), 1, (A.EXPERIENCIA4-2)/1, DECODE(
SIGN(A.EXPERIENCIA1-5), 1, (A.EXPERIENCIA1-6)/-1, 1))) , 6,
DECODE(SIGN((A.EXPERIENCIA4-3.5)*(8.5-A.EXPERIENCIA4)),
-1, 0, DECODE(SIGN(3-A.EXPERIENCIA4+2), 1, (A.EXPERIENCIA4-
2)/(1+2), DECODE(SIGN(5-A.EXPERIENCIA4+2), 1, 1,
(A.EXPERIENCIA1-6)/(-1-2))))), 7, DECODE(SIGN((0.50*
A.EXPERIENCIA3+A.EXPERIENCIA4-2.5)*(5.5-0.50*A.EXPERIENCIA2
+A.EXPERIENCIA1)), -1, 0, DECODE(SIGN((A.EXPERIENCIA3+
A.EXPERIENCIA4-3)*(5-A.EXPERIENCIA2+A.EXPERIENCIA1)), -1,
DECODE(SIGN((0.50*A.EXPERIENCIA3+A.EXPERIENCIA4-2.5)*
(3-A.EXPERIENCIA3+A.EXPERIENCIA4)), -1, DECODE(SIGN((5.5-
0.50*A.EXPERIENCIA2+A.EXPERIENCIA1)*(A.EXPERIENCIA2+
A.EXPERIENCIA1-5)), 1, (6-A.EXPERIENCIA1)/(A.EXPERIENCIA2-
-1)), (A.EXPERIENCIA4-2)/(1-A.EXPERIENCIA3)), 1)), 0), 2)
, DECODE(E.RENDIMIENTO1, 0, 'MALO', 1, 'REGULAR', 2, 'BUENO',
3, 'EXCELENTE') , DECODE(E.RENDIMIENTO1, 2, 0.80, 3, 1.00, 0)
FROM PERSONAL P , APTITUD A , EMPLEOS E WHERE P.EMP# =
A.EMP# AND P.EMP# = E.EMP# AND (P.EDADT=0 OR P.EDADT=2
OR P.EDADT=3 AND P.EDAD1 BETWEEN 20.8 AND 31.5 OR P.EDADT=
4 AND P.EDAD1 IN (0, 1, 2) OR P.EDADT=5 AND P.EDAD1<=31.5
AND P.EDAD4>=20.8 OR P.EDADT=6 AND P.EDAD4 BETWEEN 24.3
AND 38 OR P.EDADT=7 AND P.EDAD4>=20.8 AND P.EDAD1<=31.5
AND 0.70*P.EDAD3+P.EDAD4>=20.8 AND 0.70*P.EDAD2+P.EDAD1<=
31.5) AND (A.EXPERIENCIAT=0 OR A.EXPERIENCIAT=2 OR
A.EXPERIENCIAT=3 AND A.EXPERIENCIA1 BETWEEN 2.5 AND 5.5
OR A.EXPERIENCIAT=4 AND A.EXPERIENCIA1 IN (0, 1) OR
A.EXPERIENCIAT=5 AND A.EXPERIENCIA1<=5.5 AND
A.EXPERIENCIA4>=2.5 OR A.EXPERIENCIAT=6 AND
A.EXPERIENCIA4 BETWEEN 3.5 AND 8.5 OR A.EXPERIENCIAT=7
AND A.EXPERIENCIA4>=2.5 AND A.EXPERIENCIA1<=5.5 AND
0.50*A.EXPERIENCIA3+A.EXPERIENCIA4>=2.5 AND 0.50*
A.EXPERIENCIA2+A.EXPERIENCIA1<=5.5) AND E.RENDIMIENTO1

```

IN (2,3)

NOMBRE	PROF	EDAD	C(ED)	EXPER	C(EXP)	REND	C(RND)
ALVARO	SECRETARIO	JOVEN	1	ALGUNA	1	BUENO	.8
INES	SECRETARIO	JOVEN	1	[1,3,5]	1	BUENO	.8
JULIAN	INGENIERO	31	.8	[2,6,13,7]	1	BUENO	.8
PALOMA	TECNICO	27	1	4	1	BUENO	.8
FERNANDO	VENDEDOR	JOVEN	1	ALGUNA	1	EXCELENTE	1
OLGA	INGENIERO	27	1	ALGUNA	1	BUENO	.8
JOSE	ADMINISTRATIVO	[25,30]	1	3	1	BUENO	.8
MATILDE	VENDEDOR	[25,30,35]	1	5	1	BUENO	.8
FEDERICO	ADMINISTRATIVO	23	1	[0,2,4]	.67	BUENO	.8

Tabla A.27: Resultado de la Consulta A.4

Bibliografía

- [Adamo 80] Adamo J.M. "Fuzzy Decision Trees." *Fuzzy Sets & Systems* 4, 207-219. (1980)
- [Anvari, Rose, 87] Anvari M., Rose G.F. "Fuzzy Relational Databases". *Analysis of Fuzzy Information*. Bezdek ed. Vol II CRC Press. (1987)
- [Baldwin 79] Baldwin J.F., Guild N.C.F "Comparison of Fuzzy Sets on the Same Decision Space", *Fuzzy Sets and Systems* 2, 213-233. (1979)
- [Baldwin 87] Baldwin J.F., "FRIL-A Fuzzy Relational Inference Language", *Fuzzy Sets and Systems*, Vol. 14, pp. 155-174. (1987).
- [Bass, Kwakernaak, 77] Bass S.M., Kwakernaak H., "Rating and Ranking of Multiple-Aspect Alternatives Using Fuzzy Sets". *Automatica* 13, 47-58. (1977)
- [Bellman, Giertz, 73] Bellman R. and Giertz M. "On the Analytic Formalism of the Theory of Fuzzy Sets." *Information Sciences*. 5. 149-156. (1973)
- [Bosc 88] Bosc P., Galibourg M., Hamon G. "Fuzzy Querying with SQL: Extensions and Implementation Aspects". *Fuzzy Sets and Systems*. v.28 pp. 333-349. (1988)
- [Buckles, Petry, 82] Buckles B.P., Petry F.E. "A Fuzzy Representation of Data for Relational Databases". *Fuzzy Sets and Systems*, 7. 213-226. (1982)
- [Buckles, Petry, 84] Buckles B.P., Petry F.E. "Extending the Fuzzy Database with Fuzzy Numbers". *Information Sciences*, 34. 145-155. (1984)

- [Buckles, Petry, Sachar, 89] Buckles B.P., Petry F.E., Sachar H.S. "A Domain Calculus for Fuzzy Relational Databases". *Fuzzy Sets and Systems*, 29. 327-340. (1989)
- [Chang 76] Chang C. L. "DEDUCE-A Deductive Query Language for Relational Data Bases". *Pattern Recognition and Artificial Intelligence* (ed., C. H. Chen). New York: Academic Press (1976)
- [Codd 70] Codd E.F. "A Relational Model of Data for Large Shared Data Banks". *Commun. ACM*, 13. (6). pp. 377-387 (1970)
- [Codd 71] Codd E.F. "Relational Completeness of Data Sublanguages". En *Courant Computer Science Symposia 6, Data Base Systems* (New York Mayo 24-25) Prentice Hall (1971)
- [Codd 71b] Codd E.F. "ALPHA: A Data Base Sublanguage Founded on the Relational Calculus." In *Proc. 1971 ACM SIGFIDET Workshop* (San Diego, Nov 11-12) (1971)
- [Codd 79] Codd E.F. "Extending the Database Relational Model to Capture More Meaning." *ACM Trans. on Database Systems* 4:4 (1979)
- [Codd 82] Codd E.F. "Relational Database: A Practical Foundation for Productivity." *Comm. ACM* 25:2 (1982)
- [Codd 85] Codd E.F. "How Relational Is Your Database Management System?." *Computerworld*, October 14 and 21 (1985)
- [Codd 86a] Codd E.F. "Missing Information (Applicable and Inapplicable) in Relational Databases." *ACM SIGMOD Record*, Vol. 15, no. 4 (1986)
- [Codd 86b] Codd E.F. "The Twelve Rules for Relational DBMS." San Jose, The Relational Institute, Technical Report EFC-6. (1986)
- [Codd 87] Codd E.F. "More Commentary on Missing Information in Relational Databases." *ACM SIGMOD Record*, Vol. 16, no. 1. (1987)

- [Codd 90] Codd E.F. "The Relational Model for Database Management, Version 2." Reading, Mass.: Addison-Wesley (1990)
- [Cubero, Medina, Vila 93] Cubero J. C., Medina J. M., Vila M. A. "Influences of Granularity Level in Fuzzy Functional Dependencies" en "Symbolic and Quatitative Approach to Reasoning and Uncertainty", Letures Notes in Computer Sciences, 747. Springer Verlag, pp. 73-78. (1993)
- [Cubero, Pons, Vila 94] Cubero J. C., Pons O., Vila M. A. "Weak and Strong Resemblances and Fuzzy Functional Dependencies". IEEE'94 International Conference, Florida. Por aparecer (1994)
- [Cubero, Vila 94] Cubero J. C., Vila M. A. "A New Definition of Fuzzy Functional Dependencies in Fuzzy Relational Databases". International Journal of Intelligent Systems, 9(5), pp 441-448. (1994)
- [Date 86] Date C.J. "Null Values in Database Management." En C.J. Date, Relational Database: Selected Writings. Reading, Mass.: Addison-Wesley (1986)
- [Date 86II] Date C.J. "An Introduction to Database Systems", Volume II, Fourth Edition, Adisson-Wesley (1986)
- [Date 90I] Date C.J. "An Introduction to Database Systems", Volume I, Fifth Edition, Adisson-Wesley (1990)
- [Date 89] Date C.J. "NOT is Not "Not"! (Notes on Three-Valued Logic and Related Matters)." En C.J. Date, Relational Database Writings 1985-1989. Reading, Mass.: Addison-Wesley (1989)
- [Delgado, Verdegay, Vila 88] Delgado M., Verdegay J.L., Vila M.A. "A Procedure for Ranking Fuzzy Numbers Ussing Fuzzy Relations". Fuzzy Sets & Systems 26, 49-62. (1988)
- [DeLuca 72] DeLuca A. and Termini S., "A Definition of Non-probabilistic Entropy in the Setting of Fuzzy Sets Theory". Inform. Control 20, pp. 301-312. (1972)

- [Dubois, Prade, 78] Dubois D., Prade H. "Operations on Fuzzy Numbers", *Int. J. Syst. Sci.* 9, No. 6, 613-626. (1978)
- [Dubois, Prade, 79] Dubois D., Prade H. "Fuzzy Sets and Systems. Theory and Applications", Academic Press, N. Y. (1979)
- [Dubois, Prade, 79b] Dubois D., Prade H. "Fuzzy Real Algebra: Some Results", *Fuzzy Sets and Systems* 2, 327-348. (1979)
- [Dubois, Prade, 83] Dubois D., Prade H. "Ranking Fuzzy Numbers in the Setting of Possibility Theory", *Information Sciences* 30, 183-224. (1983)
- [Dubois, Prade, 88] Dubois D., Prade H. "Fuzzy Number. An Overview, the Analisis of Fuzzy Information", J.C. Bezdek CRS Press, Boca Raton Fl. USA (1985)
- [Dubois, Prade, 88] Dubois D., Prade H. "Possibility Theory. An Approach to Computerized Processing of Uncertainty", Plenum Press, N. Y. (1988)
- [Fukami, et al, 79] Fukami S., Umamo M., Muzimoto M. and Tanaka H., "Fuzzy Database Retrieval and Manipulation Languaje", *IEICE Technical Reports*, Vol. 78, No. 233, pp. 65-72, AL-78-85 (Automata and Language) (1979).
- [Gallaire, Minker, Nicolas 84] Gallaire H., Minker J., Nicolas J.M., "Logic and Databases: A deductive Approach". *ACM Computing Surveys*. Vol. 16(2), pp.153-185 (1984).
- [Hamon 86] Hamon Guy. "Extension d'un langage d'interrogation de Base de Donnees en vue de l'utilisation de questions imprecises", Tesis Doctoral, (1986).
- [Kacprzyk 86] Kacprzyk J., Ziolkowski A. "Database Queries with Fuzzy Linguistic Quantifiers". *IEEE Transactions on Syst., Man, and Cyb.* Vol SMC-16 n°3. (1986)
- [Kandel, Byatt] Kandel A. y Byatt W.J. "Fuzzy Sets, Fuzzy Algebra, and Fuzzy Statistics." *Proc. IEEE*, vol. 66, no. 12, 1978, pp. 1619-1639. (1978)

- [Kandel 79] Kandel A. "On Fuzzy Statistics." en *Advances Set Theory and Applications*, Gupta, M.M., et. al., eds., North-Holland Publishing Company. (1979)
- [Kim 90] Kim W. "Object-Oriented Databases: Definition and Research Directions". *IEEE Trans. on Knowledge and Data Engeneering* Vol. 2:3 pp. 327-341 (1990).
- [Kuhns] Kuhns J.L. "Answering Questions by Computer: A Logical Study". Informe RM-5428-PR, Rand Corp., Santa Monica, Calif. (1967)
- [Lacroix, Pirotte, 77(a)] Lacroix M., Pirotte A. "Domain-Oriented Relational Languages". *Proc. 3rd International Conference on Very Large Data Bases* (1977)
- [Lacroix, Pirotte 77(b)] Lacroix M., Pirotte A. "ILL: An English Structured Language for Relational Data Bases". En *Architecture and Models in Data Base Management Systems* (ed., G.M. Nijssen). New York. North-Holland (1977)
- [Li, Liu, 90] Li D., Liu. "A Fuzzy Prolog Database System". ed. John Wiley & Sons. New York. (1990)
- [Medina et al, 91] Medina J. M., Vila M.A., "Un Modelo de Bases de Datos Difusa Aplicado a Información Médica". 1^{er} Congreso Español sobre Tecnologías y Lógica Fuzzy, pp 1-6. Granada. (1991)
- [Medina et al, 93(a)] Medina J. M., Vila M.A., Cubero J.C., Pons O. "Towards the Implementation of a Generalized Fuzzy Relational Database Model". *Sometido a Fuzzy Sets & Systems*. (1993)
- [Medina et al, 93(b)] Medina J. M., Pons O., Vila M.A. "GEFRED. A Generalized Model of Fuzzy Relational Data Bases". *Information Sciences*,76,1-2, pp 87-109. (1994)
- [Oracle] "Oracle RDBMS. SQL Language Reference Manual vs. 6.0". (1990).

- [Pons, 93] Pons O., Vila M.A., Delgado M. "Inferencia a Partir de una Base de Datos Difusa, Utilizando Reglas Difusas." *3^{er} Congreso en Tecnologías y Lógica Fuzzy*. Santiago de Compostela. (1993)
- [Pons, 94] Pons O., Medina J.M., Vila M.A. "Handling Imprecise Medical Information in the Framework of Logic Fuzzy Databases". Por aparecer en *International Journal of Fuzzy Systems*. (1994)
- [Parsaye et al 89] Parsaye K. et al "Intelligent Databases Object-Oriented, Deductive Hypermedia Technologies" J. Wiley New York (1989)
- [Pirotte, Wodom] Pirotte A., Wodon P. "A Comprehensive Formal Query Language for a Relational Data Base". *R.A.I.R.O. Informatique/Compute Science* 11, num. 2 (1977)
- [Prade, Testemale, 84] Prade H., Testemale C. "Generalizing Database Relational Algebra for the Treatment of Incomplete/Uncertain Information and Vague Queries". *Information Sciences*, 34. 115-143. (1984)
- [Prade 84b] Prade H. "Lipski's Approach to Incomplete Information Databases Restated and Generalized in the Setting of Zadeh's Possibility Theory". *Inf. Syst.* 9(1),27-42.
- [Prade, Testemale, 87] Prade H., Testemale C. "Representation of Soft Constraints and Fuzzy Attribute Values by means of Possibility Distributions in Databases". Bezdek ed. *Analysis of Fuzzy Information. Vol II* CRC Press. (1987)
- [Prade, Testemale, 87b] Prade H., Testemale C. "Fuzzy Relational Databases: Representational Issues and Reduction Using Similarity Measures". *J. Am. Soc. Inf. Sci.* 38(2),118-126. (1987)
- [Reiter 84] Reiter R. "Towards a Logical Reconstruction of Relational Database Theory", En *Conceptual Modelling*. Eds. Brodie, Mylopoulos y Schmidt, pp 193-238 (1984)
- [Raju, Majumdar, 87] Raju K., Majumdar A. "The Study of Joins in Fuzzy Relational Databases". *Fuzzy Sets and Systems* 21. pp19 – 34.(1987)

- [Rundensteiner, Hawkes, Bandler, 89] Rundensteiner E.A., Hawkes L. W., Bandler W. "On Nearness Measures in Fuzzy Relational Data Models". *International Journal of Approximate Reasoning*, 3.pp267 – 298.(1989)
- [Schweizer, Sklar] Schweizer B., Sklar A. "Probabilistic Metric Spaces", North-Holland (1983)
- [Shenoi, Melton, 89] Shenoi S., Melton A. "Proximity Relations in the Fuzzy Relational Database Model". *Fuzzy Sets and System*, 31. 285-296. (1989)
- [Shenoi, Melton, 90] Shenoi S., Melton A. "An Extended Version of the Fuzzy Relational Database Model". *Information Sciences*, 52. 35-52. (1990)
- [Tahani 77] V. Tahani V. "A Conceptual Framework for Fuzzy Query Processing-A step toward Very Intelligent Database Systems". *Information Procesing & Management*, vol. 13, pp. 289-303 (1977).
- [Trillas 79] Trillas E. "Sobre Funciones de Negación en la Teoría de Conjuntos Difusos". *Stochastica*, Vol 3, n 1, 47-59. (1979)
- [Ullman] Ullman J. D. "Principles of Database and Knowledge Base Systems: Volume I: Rockville, Md.: Computer Science Press (1988)
- [Umano 77] Umano M., Mizumoto M. and Tanaka. "Implementation of Fuzzy-Set Manipulation System", *Journal of Information Processing Society of Japan*, Vol. 18, pp. 884-892. (1977).
- [Umano 78] Umano M., Mizumoto M. and Tanaka. "FSTDS System: A Fuzzy-Set Manipulation System", *Information Sciences*. Vol. 14, pp. 115-159 (1978).
- [Umano 82] Umano M. "Freedom-0: A Fuzzy Database System". *Fuzzy Information and Decision Processes*. Gupta-Sanchez edit. North-Holland Pub. Comp. (1982)
- [Vassiliou 80] Vassiliou Y. "Fuctional Dependences and Incomplete Information." *Proc. 6th International Conference on Very Large Data Bases* (1980)

- [Vila Cátedra, 92] Vila M.A., “Las Bases de Datos Relacionales y su Tratamiento Lógico”. Proyecto de Cátedra. (1992).
- [Vila et al. 1993] Vila M.A., Cubero J.C., Medina J.M., Pons O. “Logic and Fuzzy Relational Databases: a new Language and a new Definition”. En P. Bosc y Kacprzyk, editors, *Fuzzy Sets and Possibility Theory in Database Management Systems*. Physica Verlag. (1993).
- [Vila et al. 1993b] Vila M.A., Cubero J.C., Medina J.M., Pons O. “On the Use of a Logical Definition of Fuzzy Relational Databases”. En el 2º IEEE International Conference on Fuzzy Systems, S. Francisco, pp 489-499. (1993).
- [Vila et al. 1994] Vila M.A., Cubero J.C., Medina J.M., Pons O. “The Generalized Selection: An Alternative Way for the Quotient Operations in Fuzzy Relational Databases”. En International Conference on Information Systems, IPMU’94. Por aparecer. (1994).
- [Vila et al. 1994b] Vila M.A., Cubero J.C., Medina J.M., Pons O. “The Generalized Selection: An Alternative Way for the Quotient Operations in Fuzzy Relational Databases”. En International Conference on Information Systems, IPMU’94. Por aparecer. (1994).
- [Vila et al. 1994c] Vila M.A. et al. “A Logic Approach to Fuzzy Relational Databases”, *International Journal of Intelligent Systems*,9(5), pp 449-461, (1994).
- [Vila et al. 1994d] Vila M.A., Cubero J.C., Medina J.M., Pons O. “Some Recent Advances in Fuzzy Relational and Fuzzy Deductive Databases”, *European Research Consortium for Informatics and Mathematics*, pp.161-176, Barcelona, 1-2 November. (1994).
- [Yager 78] Yager R.R. “Ranking Fuzzy Subsets Over the Unit Interval.” *Proc. CDC* 1435-1437. (1978)
- [Yager 81] Yager R.R. “A Procedure for Ordering Fuzzy Subsets of the Unit Interval”. *Information Sciences* 24, 143-161. (1981)

- [Yager 87] Yager R.R., Ovchinnikov S y otros, "Fuzzy Sets and Applications: Selected Pappers by L. A. Zadeh", Wiley Intersc. (1987)
- [Zadeh 65] Zadeh L.A. "Fuzzy Sets". Information and Control, 8, 338-353. (1965)
- [Zadeh 71] Zadeh L.A. "Similarity Relations and Fuzzy Orderings". Information Sciences. vol. 3 177-200. (1971)
- [Zadeh 72] Zadeh L.A., "Fuzzy Languajes and Their Relation to Human and Machine Intelligence". Proc. Int. Conf. on Man and Computer, Bordeaux, France, pp. 130-165. Karger Basel. (1972)
- [Zadeh 75] Zadeh L.A., "The Concept of a Linguistic Variable and Its Application to Approximate Reasoning", Information Sciences, Vol. 8, pp. 199-248; Vol. 9, pp. 43-80. (1975).
- [Zadeh 78] Zadeh L.A. "Fuzzy Sets as a Basis for a Theory of Possibility". Fuzzy Sets and Systems. 1, 3-28. (1978)
- [Zadeh 83] Zadeh L.A. "A Computacional Approach to Fuzzy Quantifiers in Natural Languages". Computers and Mathematics with Applications 9, pp. 149-184. (1983)
- [Zemankova 84] Zemankova M., Kandel A. "Fuzzy Relational Data Bases - A Key to Expert Systems-". Verlag TUV Rheinland. (1984)
- [Zemankova 85] Zemankova M., Kandel A. "Implementing Imprecision in Information Systems", Information Sciences, Vol. 37, pp. 107-141. (1985)
- [Zimmermann 91] Zimmermann H. J. "Fuzzy Set Theory and its Applications. 2nd Edition". Ed. Kluwer Academic Publishers. (1991)
- [Zloof] Zloof M.M. "Query By Example". Proc. NCC 44, Anaheim, Calif. (mayo 1975). Montvale, N.J.:AFIPS Press (1977)