



COPPE/UFRJ

ALIANÇA - UMA PROPOSTA DE ARQUITETURA PARA BANCO DE
DADOS NEBULOSOS

Raquel Defelippo Rodrigues

Tese de Doutorado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Doutor em Engenharia de Sistemas e Computação.

Orientadora: Marcia Helena Costa Fampa

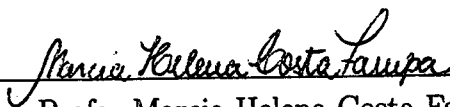
Rio de Janeiro
Dezembro de 2008

ALIANÇA - UMA PROPOSTA DE ARQUITETURA PARA BANCO DE
DADOS NEBULOSOS

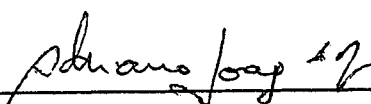
Raquel Defelippo Rodrigues

TESE SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ
COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE)
DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS
REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR
EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.


Aprovada por:



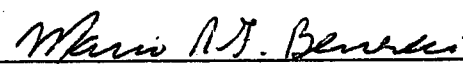
Prof. Marcia Helena Costa Fampa, D.Sc.



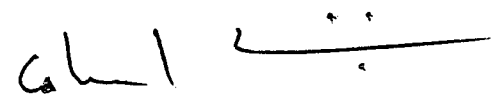
Prof. Adriano Joaquim de Oliveira Cruz, Ph.D.



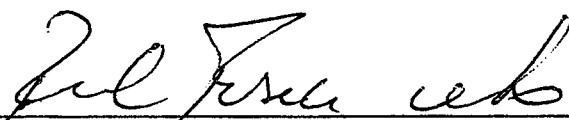
Prof. Geraldo Bonorino Xexéo, D.Sc.



Prof. Mário Roberto Folhadela Benevides, Ph.D.



Prof. Josefino Cabral Melo Lima, Ph.D.



Prof. Raul Fonseca Neto, D.Sc.

RIO DE JANEIRO, RJ - BRASIL

DEZEMBRO DE 2008

Rodrigues, Raquel Defelippo

Aliança - Uma Proposta de Arquitetura Para Banco de Dados Nebulosos/ Raquel Defelippo Rodrigues. - Rio de Janeiro: UFRJ/COPPE, 2008.

XVI, 156 p.: il.; 29,7 cm.

Orientadora: Marcia Helena Costa Fampa

Tese (doutorado) - UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2008.

Referências Bibliográficas: p. 139-144.

1. Bancos de Dados Nebulosos. 2. Comparadores Nebulosos. 3. FSQL. I. Fampa, Marcia Helena Costa. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

“Quando alguém encontra seu caminho precisa ter coragem suficiente para dar passos errados. As decepções, as derrotas, o desânimo são ferramentas que Deus utiliza para mostrar a estrada”

Paulo Coelho

*Às pessoas que mais amo:
meu marido Marcio e
meus pais Elizabete e Fábio*

AGRADECIMENTOS

A Deus, pelo término de mais uma etapa da minha vida.

Ao meu marido Marcio, por estar ao meu lado a todo momento, me dando forças para seguir em frente.

Aos meus pais Fábio e Elizabete e aos meus irmãos Marcos e Diogo, pelo apoio constante durante este trabalho e por toda a minha vida.

Ao professor Adriano Cruz, pela dedicação e pelos conhecimentos transmitidos durante o desenvolvimento deste trabalho. Muito mais do que mestre se tornou um grande amigo.

Aos professores da UFRJ, em especial a professora Marcia Fampa, pelo incentivo.

Aos colegas da UFRJ, por participarem ativamente da minha vida acadêmica. Um agradecimento especial aos amigos Austeclynio e Rafael Cavalcanti.

Resumo da Tese apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Doutor em Ciências (D.Sc.)

ALIANÇA - UMA PROPOSTA DE ARQUITETURA PARA BANCO DE DADOS NEBULOSOS

Raquel Defelippo Rodrigues

Dezembro/2008

Orientadora: Marcia Helena Costa Fampa

Programa: Engenharia de Sistemas e Computação

Este trabalho apresenta uma nova arquitetura para bancos de dados nebulosos chamada *Aliança*. Bancos de dados nebulosos usualmente armazenam informações e sua associada metainformação para adicionar contexto a todos os diferentes tipos de dados que podem ser armazenados neste banco de dados. Mas, os sistemas de bancos de dados nebulosos se tornam muito complexos e difíceis de manter porque conceitos nebulosos são complicados de serem representados usando formas tradicionais de bancos de dados. *Aliança*, porém, introduz um novo caminho para representar esta base de metainformação nebulosa, que simplifica muito as tarefas de gerenciamento de dados. As principais vantagens desta nova representação são a facilidade de entendimento, implementação, uso e suporte. A nova base de metaconhecimento organiza as informações no formato XML, que adiciona uma vantagem extra de portabilidade. O FSQL usado assume um convencional banco de dados e adiciona ferramentas externas para a manipulação de dados nebulosos, sendo baseado na relação de similaridade e na teoria da possibilidade introduzida por Zadeh.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

ALIANÇA: A PROPOSAL FOR A FUZZY DATABASE ARCHITECTURE

Raquel Defelippo Rodrigues

December/2008

Advisor: Marcia Helena Costa Fampa

Department: Computing and Systems Engineering

This work presents a new fuzzy database architecture called *Aliança*. Fuzzy databases usually store information and its associated metainformation in order to add context to the whole range of different types that can be stored in these databases. However, the fuzzy database systems became very complex and difficult to maintain because fuzzy concepts are very difficult to represent using traditional databases representation forms. *Aliança*, however, introduces a new way to represent this fuzzy metainformation base, that greatly simplifies data management tasks. The main advantages of these new representation are the easyness of understanding, implementation, use and support. This new metainformation base organizes information using the XML format, that adds the extra advantage of portability. FSQL assumes a conventional database and adds external tools to handle fuzzy data and is based on similarity relations and the theory of possibility introduced by Zadeh.

Sumário

1	Introdução	1
1.1	Motivação e Objetivo	1
1.2	Descrição dos Capítulos	2
2	Fundamentos	4
2.1	Lógica Clássica \times Lógica Nebulosa	4
2.1.1	Breve Histórico	5
2.1.2	Conjuntos Clássicos	5
2.1.2.1	Operações com Conjuntos Clássicos	6
2.1.2.2	Propriedades de Conjuntos Clássicos	7
2.1.3	Conjuntos Nebulosos	9
2.1.3.1	Conceitos sobre Conjuntos Nebulosos	12
2.1.3.2	Operações com Conjuntos Nebulosos	13
2.1.3.3	Propriedades de Conjuntos Nebulosos	14
2.1.3.4	Formulação e Parametrização das Funções de In- clusão	16
2.1.3.5	Variáveis Lingüísticas	19
2.1.3.6	Teoria de Possibilidade	19
2.1.3.7	Comparadores Estendidos	22
2.2	Visão Geral de SGBDs Relacionais	36
2.2.1	Histórico	36
2.2.2	Modelo Relacional	39
2.2.3	Estruturas de Dados	40

2.2.4	Manipulação dos Dados	41
2.2.4.1	Álgebra Relacional	41
2.2.4.2	Cálculo Relacional	43
2.2.5	A Linguagem SQL	44
2.3	XML - <i>Extensible Markup Language</i>	46
2.3.1	O que é XML?	46
2.3.2	HTML \times XML	47
2.3.3	Pontos Fortes de XML	48
2.4	Bancos de Dados Nebulosos	50
2.4.1	Informações Imprecisas	50
2.4.1.1	Relações Nebulosas Baseadas em Similaridade . . .	51
2.4.1.2	Relações Nebulosas Baseadas em Possibilidade . . .	53
2.4.2	Consultas Imprecisas	55
2.4.2.1	O Problema da Redundância de Dados	59
3	Histórico	60
3.1	Modelo de Buckles-Petry	61
3.2	Modelo de Prade-Testemale	62
3.2.1	Representação de dados por Distribuição de Possibilidade . .	62
3.3	Modelo de Umano-Fukami	64
3.4	Modelo de Zemankova-Kaendel	66
3.4.1	Arquitetura	67
3.4.2	Tipos de Dados	68
3.5	Modelo GEFRED	69
3.5.1	Arquitetura do Banco de Dados Relacional Difuso: O Servi- dor FSQL (Fuzzy SQL)	69
3.5.2	Representação do Conhecimento Impreciso	71
3.5.3	Comparadores Nebulosos Generalizados	73
3.5.4	Implementação do Conhecimento Impreciso na Base de Dados	75

3.5.5	Implementação do Conhecimento Impreciso na Base de Metaconhecimento Difuso (FMB)	77
3.5.6	Exemplo de Implementação no FIRST da BD e FMB	79
3.6	Trabalho de Turowski e Weng	85
3.6.1	A modelagem da informação nebulosa	87
3.6.2	DTD	88
3.7	O mapeamento entre um banco de dados nebulosos e um arquivo XML nebuloso - Um Trabalho de Gauray e Alhajj	89
3.7.1	Mapeamento de Bancos de Dados Nebuloso Relacionais para arquivos XML Nebuloso	90
4	O Sistema de Gerenciamento de Banco de Dados Nebulosos <i>Aliança</i>	95
4.1	Objetivo	95
4.2	Arquitetura	96
4.2.1	Representação do Conhecimento Impreciso	97
4.3	Base do Metaconhecimento Nebuloso	100
4.3.1	Estrutura da BMN	101
4.3.2	Descrição dos Arquivos XML	104
4.3.3	Representação Interna da Base de Dados	107
4.4	Comparadores Nebulosos	110
4.5	Servidor FSQL	119
4.5.1	<i>FSQL</i> - Fuzzy SQL	119
4.5.2	Tratamento Especial dado às Etiquetas Lingüísticas com Distribuições de Possibilidade	120
4.6	Modelo de Dados	123
4.6.1	<i>FUML</i> - Fuzzy UML	123
4.6.2	Diagrama de Classes Nebulosas	124
4.7	Implementação do Sistema <i>Aliança</i>	126
4.7.1	Arquitetura	126
4.7.2	Tecnologias Utilizadas	127

4.7.2.1	A Linguagem Java	127
4.7.2.2	JavaCC	129
4.7.2.3	API JDOM	130
4.7.2.4	MySQL	131
4.7.3	Exemplos	131
5	Conclusões e Trabalhos Futuros	137
	Referências Bibliográficas	139
	A Publicação	145

Lista de Figuras

2.1	Conjunto clássico das pessoas altas	7
2.2	Conjunto nebuloso das pessoas altas	10
2.3	Conjuntos: pessoas adultas e pessoas não adultas	16
2.4	Função de inclusão triangular $Triângulo(x; 20, 60, 80)$	17
2.5	Função de inclusão trapezoidal $Trapézio(x; 10, 20, 60, 100)$	18
2.6	Função de inclusão intervalar $Intervalo(x; 10, 40)$	18
2.7	Número preciso X	23
2.8	$X_{trap} = Trapézio(x; a, b, c, d)$	23
2.9	$X_{triang} = Triângulo(x; d - margem, d, d + margem)$	23
2.10	$X_{inter} = Intervalo(x; m, n)$	24
2.11	Comparador estendido $\geq (X)$	24
2.12	Comparador estendido $\geq (X_{trap})$	25
2.13	Comparador estendido $\geq (X_{triang})$	26
2.14	Comparador estendido $\geq (X_{inter})$	26
2.15	Comparador estendido $\leq (X)$	27
2.16	Comparador estendido $\leq (X_{trap})$	27
2.17	Comparador estendido $\leq (X_{triang})$	28
2.18	Comparador estendido $\leq (X_{inter})$	29
2.19	Comparador $> (X)$	30
2.20	Comparador $> (X_{trap})$	30
2.21	Comparador $> (X_{triang})$	30
2.22	Comparador $> (X_{inter})$	30
2.23	Comparador $< (X)$	32

2.24	Comparador $< (X_{trap})$	32
2.25	Comparador $< (X_{triang})$	32
2.26	Comparador $< (X_{inter})$	32
2.27	Comparador $>_{muito} (X)$	34
2.28	Comparador $>_{muito} (X_{trap})$	34
2.29	Comparador $>_{muito} (X_{triang})$	34
2.30	Comparador $>_{muito} (X_{inter})$	34
2.31	Comparador $<_{muito} (X)$	36
2.32	Comparador $<_{muito} (X_{trap})$	36
2.33	Comparador $<_{muito} (X_{triang})$	36
2.34	Comparador $<_{muito} (X_{inter})$	36
2.35	Definição de etiquetas sobre o atributo População	57
2.36	Definição de etiquetas sobre o atributo PIB	57
2.37	0,7-corte PIB baixo	58
2.38	0,7-corte População grande	58
3.1	Esquema Geral do FIRST	70
3.2	Esquema da Base do Metaconhecimento Difuso (FMB)	78
3.3	Definição de etiquetas sobre o atributo Salário	81
3.4	Definição de etiquetas sobre o atributo Idade	81
3.5	Exemplo de um conjunto nebuloso contínuo	87
4.1	Arquitetura Geral do Banco de Dados Nebulosos <i>Aliança</i>	96
4.2	Distribuição de possibilidade para o tipo Unknown	98
4.3	Distribuição de possibilidade para o tipo Undefined	98
4.4	Um exemplo de um rótulo lingüístico para o conceito “Médio”	99
4.5	Um exemplo de intervalo de possibilidade	99
4.6	Um exemplo de valor aproximado	100
4.7	Estrutura da BMN	102
4.8	Definição das possíveis etiquetas usadas para o atributo Preço	103
4.9	Definição das possíveis etiquetas usadas para o atributo Idade	103

4.10	Um exemplo do operador FEQ	111
4.11	Valor aproximado X e etiqueta lingüística Y	112
4.12	$\mu_{FGEQ}(X, Y)$	112
4.13	Valor preciso X e intervalo de possibilidade Y	112
4.14	$\mu_{FLEQ}(X, Y)$	112
4.15	Valor aproximado X e etiqueta lingüística Y	114
4.16	$\mu_{MGT}(X, Y)$	114
4.17	Intervalo de possibilidade X e etiqueta lingüística Y	114
4.18	$\mu_{MLT}(X, Y)$	114
4.19	Valor aproximado X e intervalo de possibilidade Y	115
4.20	$\mu_{NFEQ}(X, Y)$	115
4.21	Intervalo de possibilidade X e etiqueta lingüística Y	116
4.22	$\mu_{NFGT}(X, Y)$	116
4.23	Etiqueta lingüística X e valor preciso Y	116
4.24	$\mu_{NFLT}(X, Y)$	116
4.25	Valor aproximado X e etiqueta lingüística Y	118
4.26	$\mu_{NMGT}(X, Y)$	118
4.27	Intervalo de possibilidade X e etiqueta lingüística Y	118
4.28	$\mu_{NMLT}(X, Y)$	118
4.29	Modelo FUML do Diagrama de Classes	125
4.30	Arquitetura Geral do Banco de Dados Nebulosos <i>Aliança</i> - ênfase Implementação	126

Lista de Tabelas

2.1	Relação Perfil dos Funcionários de uma Empresa	40
2.2	Animais em Extinção	51
2.3	Relação de Profissionais Liberais	52
2.4	Matriz de Similaridade para o Atributo Especialidade	52
2.5	Relação dos Alunos de uma Escola	55
2.6	Matriz de Similaridade para o Atributo Participação em Sala	56
2.7	Resultado da consulta	56
3.1	Tipos de Dados Representados pelo GEFRED	71
3.2	Representação de Atributos do Tipo 2	75
3.3	Representação de Atributos do Tipo 3	77
3.4	Relação Empregados ¹	80
3.5	Relação de Similaridade s_r sobre o atributo Rendimento	81
3.6	Exemplo para a tabela FUZZY_COL_LIST	82
3.7	Exemplo para a tabela FUZZY_APPROX_MUCH	82
3.8	Exemplo para a tabela FUZZY_OBJECT_LIST	82
3.9	Exemplo para a tabela FUZZY_LABEL_DEF	83
3.10	Exemplo para a tabela FUZZY_NEARNESS_DEF	83
3.11	Representação Interna Real da relação Empregados no SGBD com a extensão do FIRST	84
3.12	Uma relação nebulosa ‘ <i>Likes</i> ’ com tuplas nebulosas	90
3.13	Uma relação nebulosa ‘ <i>Employees</i> ’ com atributos nebulosas	91

3.14	Uma relação nebulosa ‘Employees_Body’ com relação de similaridade para o atributo ‘Health’ dada na Tabela 3.15	92
3.15	Relação de similaridade associada ao atributo ‘Health’	93
4.1	Informações armazenadas na BMN	101
4.2	Carros Antigos ¹	102
4.3	Relação de Similaridade s_r definida sobre o atributo Eficiência	104
4.4	Representação interna da relação Carros Antigos no SGBD proposto <i>Aliança</i>	108
4.5	Descrição dos novos atributos usados representação interna dos valores nebulosos	109
4.6	Relação de Semelhança sobre o comparador nebuloso “($F EQ(d, d')$) - <i>Possivelmente igual a</i> ”	121
4.7	Relação de Semelhança sobre o comparador nebuloso “($F GEQ(d, d')$) - <i>Possivelmente maior ou igual a</i> ”	121
4.8	Carros Antigos - Resultado da Consulta	135

Capítulo 1

Introdução

1.1 Motivação e Objetivo

Tradicionalmente os bancos de dados foram designados para o armazenamento eficiente, a modificação segura e a adequada recuperação de grandes amostras de dados precisos.

Os modelos de bancos de dados tradicionais procuram modelar o mundo real, mas no dia a dia estamos sempre rodeados pela incerteza. E manipular incerteza não é uma tarefa fácil, visto que apenas o homem é capaz de tomar certas decisões em relação a informações incertas.

Uma das mais importantes e usuais operações em banco de dados são aquelas destinadas às consultas de informações. Por exemplo, podemos efetuar perguntas do tipo “*Quais são as pessoas que têm mais de 20 e menos de 30 anos?*”. Talvez fosse mais interessante para nós se pudéssemos efetuar uma consulta que se aproximasse do raciocínio humano tal como “*Quais são as pessoas jovens?*”. O conceito de “*jovem*” indica claramente que a pessoa não tem 80 anos, mas não indica explicitamente a sua idade exata. Nos bancos de dados tradicionais a etiqueta *jovem*, se pudesse ser armazenada, complicaria o armazenamento dos dados numéricos normais, impossibilitando o tratamento apropriado desta informação. E ainda não acrescentaria nada ao banco, pois a semântica da palavra *jovem* não estaria armazenada e assim não teríamos como saber se uma pessoa que possui 25

anos é jovem ou não.

O desejo de integrar as funcionalidades que são oferecidas pelos bancos de dados tradicionais com os conceitos nebulosos que os humanos manipulam de forma cotidiana e natural foi a principal motivação por trás do surgimento dos **Banco de Dados Nebulosos**.

Os conceitos imprecisos são bem representados pela teoria dos conjuntos nebulosos, uma vez que a lógica nebulosa (*Fuzzy Logic*) é uma técnica inteligente que se apresenta capaz de manipular informações imprecisas e permite apresentar uma resposta aproximada baseada no conhecimento inexato, incompleto, ou não confiável.

A união das funcionalidades oferecidas pelos bancos de dados tradicionais com a lógica nebulosa torna os novos bancos de dados nebulosos capazes de armazenar, tratar e consultar informações de forma flexível.

Este trabalho apresenta uma nova arquitetura para bancos de dados nebulosos chamada *Aliança*. *Aliança* introduz um novo caminho para representar a base de metainformação nebulosa, que simplifica as tarefas de gerenciamento de dados. As principais vantagens desta nova representação são a facilidade de entendimento, implementação, uso e suporte.

1.2 Descrição dos Capítulos

De uma forma rápida e geral, podemos descrever o conteúdo deste trabalho explicando brevemente o conteúdo de seus capítulos:

- **Capítulo 1:** As motivações deste trabalho são apresentados.
- **Capítulo 2:** São introduzidos os conceitos fundamentais necessários para uma boa compreensão deste trabalho. Os princípios da lógica clássica e da lógica nebulosa são apresentados. Uma revisão dos sistemas de gerenciamento de bancos de dados relacionais é feita. Os fundamentos da linguagem XML são colocados de uma forma breve e objetiva. Alguns conceitos sobre

a área de incerteza em bancos de dados são exibidos.

- **Capítulo 3:** É feita uma revisão histórica dos sistemas de gerenciamento de banco de dados relacionais nebulosos desenvolvidos sobre os quais esta tese se baseia. Uma especial atenção ao modelo GEFRED, que foi criado por Medina em [1] e aperfeiçoado por Galindo em [2], é dada por ser um modelo bem completo.
- **Capítulo 4:** É proposta uma nova arquitetura para banco de dados nebulosos chamada *Aliança*. São apresentados seus elementos básicos e como eles se relacionam. A viabilidade de tal proposta de arquitetura é confirmada com a apresentação de um protótipo do sistema.
- **Capítulo 5:** As conclusões e as propostas para trabalhos futuros são apresentadas.
- **Apêndice A:** É apresentado o artigo *Aliança: A proposal for a fuzzy database architecture incorporating XML* que será publicado em janeiro de 2009 em FUZZY SETS AND SYSTEMS - An International Journal in Information Science and Engineering Official Publication of the International Fuzzy Systems Association (IFSA).

Capítulo 2

Fundamentos

Neste capítulo, serão introduzidas algumas noções elementares da teoria dos conjuntos nebulosos (*Fuzzy Sets*), assim como a notação utilizada ao longo deste trabalho. A matriz de similaridade juntamente com as medidas de possibilidade e necessidade, criadas por Zadeh em [3] e [4], também serão apresentadas. Será feita uma breve revisão dos sistemas de gerenciamento de bancos de dados relacionais. Os fundamentos da linguagem XML serão colocados de uma forma objetiva e alguns conceitos sobre a área de incerteza em bancos de dados são exibidos.

Todo o conteúdo abordado neste capítulo é considerado essencial para a plena compreensão deste trabalho.

2.1 Lógica Clássica \times Lógica Nebulosa

A teoria da lógica nebulosa é um super conjunto da lógica clássica que foi estendida para incluir o conceito de verdade parcial, valores entre totalmente verdade e totalmente falso.

A lógica nebulosa é a lógica que suporta os modos de raciocínio que são aproximados ao invés de exatos.

2.1.1 Breve Histórico

Aristóteles, filósofo grego (384 - 322 a.C.), foi o fundador da ciência da lógica, e estabeleceu um conjunto de regras rígidas para que conclusões pudessem ser aceitas logicamente válidas. O emprego da lógica de Aristóteles levava a uma linha de raciocínio lógico baseada em premissas e conclusões. O exemplo clássico é apresentado a seguir:

$$\begin{array}{ll} \text{“Todo ser vivo é mortal”} & \text{(Premissa 1)} \\ \text{“Sarah é um ser vivo”} & \text{(Premissa 2)} \\ \hline \text{“Sarah é mortal”} & \text{(Conclusão)} \end{array}$$

Desde então, a lógica de Aristóteles tem sido binária, ou seja, uma declaração é falsa ou verdadeira, não podendo ser ao mesmo tempo parcialmente verdadeira e parcialmente falsa. Esta proposição e a lei da não contradição, que coloca que “ A e não A ” cobrem todas as possibilidades, formam a base do pensamento lógico clássico.

Assim, na lógica Aristoteleana os objetos são classificados em categorias muito bem definidas. Com isto, um elemento pertence a um conjunto ou não. Uma figura geométrica é um círculo ou não. Mas na realidade, esta lógica falha em um grande número de situações. Como é possível definir exatamente quem é *jovem* ou *baixo*?

Diante de questões desse tipo Lotfi A. Zadeh (Universidade da Califórnia, Berkeley) observou que os recursos tecnológicos disponíveis eram incapazes de compreender situações ambíguas, uma vez que todo o processamento era feito através da lógica computacional fundamentada na lógica clássica. E foi em 1965, ver [5], que Zadeh introduziu os conceitos de conjuntos nebulosos.

2.1.2 Conjuntos Clássicos

Um conjunto clássico A é uma coleção de elementos x existentes em um universo de discurso Ω , tal que ($A \subset \Omega$).

Definição 2.1 Função de Inclusão: Um conjunto clássico A , tal que $(A \subset \Omega)$, é definido pela sua função de inclusão $\mathcal{X}_A(\cdot) : \Omega \rightarrow \{0, 1\}$ que mapeia cada elemento de Ω em 1 ou 0 dependendo se o elemento é ou não um membro do conjunto. Que pode ser representado por:

$$\mathcal{X}_A(x) = \begin{cases} 1, & \text{se } x \in A \\ 0, & \text{se } x \notin A \end{cases} \quad (2.1.1)$$

Um conjunto clássico introduz um limiar que especifica se um elemento pertence ou não a este conjunto.

Exemplo 2.1 *Pode-se considerar que altos são todos os seres humanos maiores que 1,70m.*

Logo, o conjunto das medidas das pessoas altas é definido como:

$$ALTO = \{x \mid x \in \Omega \text{ e } x \geq 1,70m\}$$

sendo Ω o conjunto das alturas das pessoas em metros.

Assim a função de inclusão do conjunto das pessoas altas é dada por:

$$\mathcal{X}_{ALTO}(x) = \begin{cases} 1, & \text{se } x \geq 1,70 \\ 0, & \text{se } x < 1,70 \end{cases}$$

que está graficamente representada na Figura 2.1.

Por se tratar de um conjunto clássico existe uma fronteira rígida onde uma pessoa com 1,70m de altura pertence ao conjunto de pessoas altas, enquanto que uma pessoa com 1,699999m de altura não pertence a este mesmo conjunto.

2.1.2.1 Operações com Conjuntos Clássicos

Sejam A e B conjuntos clássicos definidos sobre o universo de discurso Ω . Os operadores booleanos convencionais aplicados em conjuntos nítidos são os seguintes:

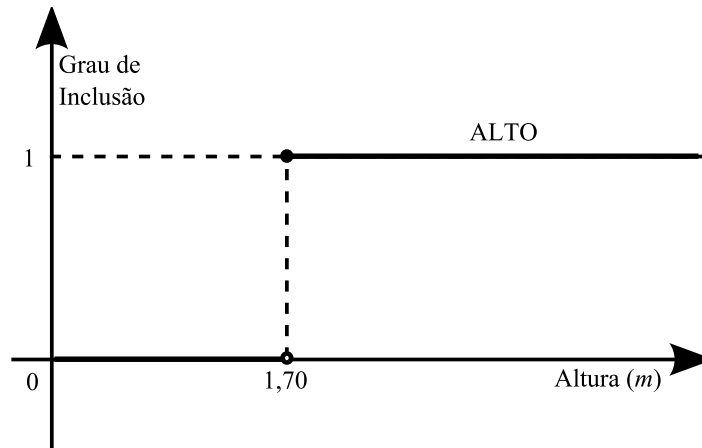


Figura 2.1: Conjunto clássico das pessoas altas

- União:

$$A \cup B = \{ x \mid x \in A \text{ ou } x \in B \} \quad (2.1.2)$$

- Interseção:

$$A \cap B = \{ x \mid x \in A \text{ e } x \in B \} \quad (2.1.3)$$

- Complemento:

$$\bar{A} = \{ x \mid x \notin A \text{ e } x \in \Omega \} \quad (2.1.4)$$

- Diferença:

$$A|B = \{ x \mid x \in A \text{ e } x \notin B \} \quad (2.1.5)$$

- União Exclusiva:

$$A \oplus B = \{ x \mid (x \in A \text{ e } x \notin B) \text{ ou } (x \notin A \text{ e } x \in B) \} \quad (2.1.6)$$

2.1.2.2 Propriedades de Conjuntos Clássicos

Considerando que A , B e C são conjuntos contidos no universo Ω , e que \emptyset representa o conjunto vazio. Os conjuntos nitidamente definidos possuem as seguintes propriedades:

1. Comutatividade:

$$A \cup B = B \cup A$$

$$A \cap B = B \cap A$$

2. Associatividade:

$$A \cup (B \cup C) = (A \cup B) \cup C$$

$$A \cap (B \cap C) = (A \cap B) \cap C$$

3. Idempotência:

$$A \cup A = A$$

$$A \cap A = A$$

4. Distributividade:

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

5. Absorção:

$$A \cup (A \cap B) = A$$

$$A \cap (A \cup B) = A$$

6. Involução:

$$\overline{\overline{A}} = A$$

7. Elemento Neutro:

$$A \cup \emptyset = A$$

$$A \cap \Omega = A$$

8. Elemento Nulo para a Interseção:

$$A \cap \emptyset = \emptyset$$

9. Dominação:

$$A \cup \Omega = \Omega$$

Três propriedades importantes de conjuntos nitidamente definidos estão apresentadas a seguir:

- **Teorema de DeMorgan:** *Permite que se transforme uma união em interseção, e vice versa*

$$\overline{A \cap B} = \overline{A} \cup \overline{B}$$

$$\overline{A \cup B} = \overline{A} \cap \overline{B}$$

- **Lei da Não Contradição:** *A interseção de um conjunto nítido com seu complemento resulta em um conjunto vazio*

$$A \cap \overline{A} = \emptyset$$

- **Lei da Exclusão do Meio:** *A união de um conjunto nítido com seu complemento resulta no conjunto Universo do domínio*

$$A \cup \overline{A} = \Omega$$

A Lei da Não Contradição estabelece que um elemento ou pertence a um conjunto nítido ou ao seu complemento.

Uma vasta bibliografia pode ser encontrada sobre lógica clássica, como por exemplo [6] e [7].

2.1.3 Conjuntos Nebulosos

Na teoria dos conjuntos clássicos a fronteira de decisão entre um elemento pertencer a um conjunto ou não pertencer é rígida, ou seja, um elemento pertence a um conjunto ou não e ponto final. Em conjuntos nebulosos esta transição ocorre de forma gradual.

Definição 2.2 Função de Inclusão em Conjuntos Nebulosos: *A função de inclusão de um conjunto nebuloso \tilde{A} é definida em seu universo de discurso Ω e é caracterizada pela função $\mu_{\tilde{A}}(\cdot) : \Omega \rightarrow [0, 1]$ que mapeia cada elemento de Ω em um valor real entre 0 e 1. Para um determinado elemento x , a função de inclusão $\mu_{\tilde{A}}(x)$ representa o grau de inclusão do elemento x no conjunto \tilde{A} .*

Exemplo 2.2 O exemplo 2.1, apresentado anteriormente para conjuntos nítidos, definia que altos eram todos os seres humanos maiores que 1,70m.

Trabalhando com conjuntos nebulosos, a rigidez desta fronteira pode ser relaxada. Assim, o conjunto das pessoas altas é então representado, por exemplo, pela seguinte função de inclusão:

$$\mu_{ALTO}(x) = \begin{cases} 0, & \text{se } x \leq 1,60 \\ \frac{x - 1,60}{0,2}, & \text{se } 1,60 < x < 1,80 \\ 1, & \text{se } x \geq 1,80 \end{cases}$$

que está graficamente representada na Figura 2.2.

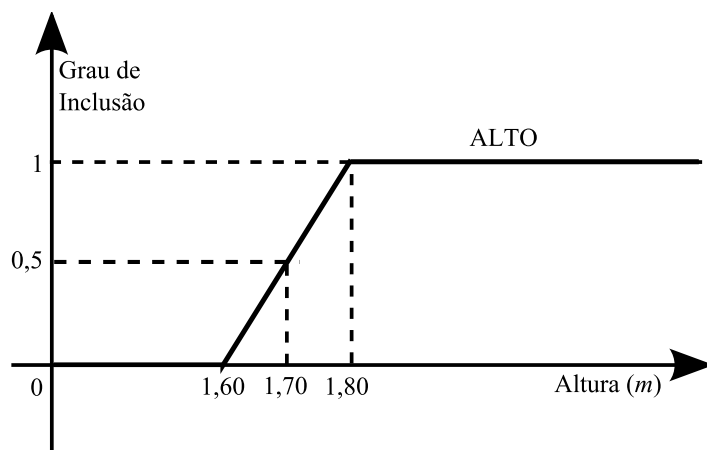


Figura 2.2: Conjunto nebuloso das pessoas altas

Desta maneira, uma pessoa com 1,80m de altura representa muito bem o conceito de uma pessoa alta, assim ela possui grau de inclusão igual a 1 no conjunto das pessoas altas, enquanto que uma pessoa de 1,70m de altura possui o grau de inclusão igual a 0,5 no conjunto das pessoas altas, pois esta não representa tão bem o conceito de pessoa alta.

Com esta representação, tanto uma pessoa com 1,70m quanto uma com 1,699999m de altura estarão incluídas no conjunto das pessoas altas, mas cada uma com o seu grau de inclusão neste conjunto. A fronteira rígida ocorrida no exemplo de conjunto clássico, apresentado na Figura 2.1, é então evitada. Esta pequena diferença entre 1,70m e 1,699999m seria realmente muito importante?

Os conjuntos nebulosos permitem os seguintes tipos de inclusão:

- **Inclusão com Grau:** Um elemento pode pertencer a um conjunto com um certo grau de inclusão.
- **Inclusão em Diversos Conjuntos:** Um elemento pode ser membro parcial de vários conjuntos.

Definição 2.3 Conjuntos Nebulosos *Seja uma coleção de objetos indicados genericamente por Ω . Então um conjunto nebuloso \tilde{A} em Ω pode ser representado continuamente por equações matemáticas, como apresentado no Exemplo 2.2, ou pode ser representado de maneira discreta, em forma de par ordenado ou fração, como apresentado a seguir:*

$$\tilde{A} = \{(x, \mu_{\tilde{A}}(x)) | x \in \Omega\} \quad (2.1.7)$$

$$\tilde{A} = \left\{ \frac{\mu_{\tilde{A}}(x)}{x} | x \in \Omega \right\} \quad (2.1.8)$$

sendo x um elemento de Ω , e $\mu_{\tilde{A}}(x)$ o grau de inclusão do elemento x no conjunto \tilde{A} , as frações não possuem o significado matemático são apenas representações.

Definição 2.4 Etiqueta Lingüística *São chamadas de etiquetas lingüísticas as palavras que expressam conjuntos nebulosos, que podem estar formalmente definidos ou não.*

Exemplo 2.3 *Se consideramos que a “idade” (em anos inteiros) é o universo de discurso de “jovem”, o conjunto nebuloso que representa este conceito poderia ter a seguinte forma:*

$$jovem = \left\{ \frac{1}{20}, \frac{1}{25}, \frac{0.9}{26}, \frac{0.8}{27}, \frac{0.6}{28}, \frac{0.5}{30}, \dots, \frac{0.1}{34} \right\}$$

O identificador “jovem”, com a conotação de que leva associado um conjunto nebuloso, recebe a denominação de **etiqueta lingüística**. O elemento $\frac{1}{20}$ é uma representação para indicar que o elemento 20 possui grau de inclusão 1 no conjunto jovem.

No dia a dia, pode-se perceber que múltiplas etiquetas lingüísticas são usadas, tais como: “jovem”, “velho”, “frio”, “quente”, “barato”, “caro”, “baixo”, “alto”, “grande”, “pequeno” e muitas outras.

A definição intuitiva dessas etiquetas pode variar de pessoa para pessoa, de momento e de acordo com o contexto a que se aplica. Por exemplo, sabemos que uma pessoa *alta* não possui a mesma altura que um prédio *alto*.

A representação de conjuntos nebulosos pode ser variada e depende fundamentalmente da natureza do universo de discurso sobre o qual o conjunto é definido.

2.1.3.1 Conceitos sobre Conjuntos Nebulosos

Definição 2.5 Igualdade de Conjuntos Nebulosos *Dois conjuntos nebulosos \tilde{A} e \tilde{B} sobre Ω são ditos iguais se cumprem:*

$$\tilde{A} = \tilde{B} \Leftrightarrow \mu_{\tilde{A}}(x) = \mu_{\tilde{B}}(x), \forall x \in \Omega \quad (2.1.9)$$

Definição 2.6 Inclusão de um Conjunto Nebuloso em Outro *Dados dois conjuntos nebulosos \tilde{A} e \tilde{B} sobre Ω , dizemos que \tilde{A} está incluído em \tilde{B} quando:*

$$\tilde{A} \subseteq \tilde{B} \Leftrightarrow \mu_{\tilde{A}}(x) \leq \mu_{\tilde{B}}(x), \forall x \in \Omega \quad (2.1.10)$$

Definição 2.7 Suporte de um Conjunto Nebuloso *O suporte de um conjunto nebuloso \tilde{A} definido sobre Ω é um subconjunto de Ω que satisfaz:*

$$\text{suporte}(\tilde{A}) = \{x \mid x \in \Omega \text{ e } \mu_{\tilde{A}}(x) > 0\} \quad (2.1.11)$$

Definição 2.8 α – Corte de um Conjunto Nebuloso *Um α – corte de um conjunto nebuloso \tilde{A} sobre Ω é um subconjunto não nebuloso de elementos de Ω , cuja função de inclusão toma um valor maior ou igual a algum valor concreto α :*

$$\tilde{A}_\alpha = \{x \mid x \in \Omega \text{ e } \mu_{\tilde{A}}(x) \geq \alpha\} \quad (2.1.12)$$

Definição 2.9 Núcleo de um Conjunto Nebuloso *O núcleo de um conjunto nebuloso \tilde{A} definido sobre Ω é um subconjunto de Ω que satisfaz:*

$$\text{núcleo}(\tilde{A}) = \{x \mid x \in \Omega \text{ e } \mu_{\tilde{A}}(x) = 1\} \quad (2.1.13)$$

Definição 2.10 Altura de um Conjunto Nebuloso A altura de um conjunto nebuloso \tilde{A} definido sobre Ω é dada por:

$$\text{altura}(\tilde{A}) = \sup_{x \in \Omega} (\mu_{\tilde{A}}(x)) \quad (2.1.14)$$

Definição 2.11 Conjuntos Nebulosos Normalizados Um conjunto nebuloso \tilde{A} definido sobre Ω é dito normalizado se e somente se:

$$\exists x \in \Omega, \mu_{\tilde{A}}(x) = \text{altura}(\tilde{A}) = 1 \quad (2.1.15)$$

Definição 2.12 Conjuntos Nebulosos Convexos Um conjunto nebuloso \tilde{A} é convexo se e somente se $x_1, x_2 \in \Omega$ e $\lambda \in [0, 1]$,

$$\mu_{\tilde{A}}(\lambda x_1 + (1 - \lambda)x_2) \geq \min\{\mu_{\tilde{A}}(x_1), \mu_{\tilde{A}}(x_2)\} \quad (2.1.16)$$

2.1.3.2 Operações com Conjuntos Nebulosos

As operações com conjuntos nebulosos são definidas em termos das funções de inclusão. As operações definidas por Zadeh são:

Definição 2.13 União: Se \tilde{A} e \tilde{B} são dois conjuntos nebulosos definidos sobre um universo de discurso Ω , a função de inclusão $\mu_U(x)$ da união desses dois conjuntos nebulosos, sendo $U = \tilde{A} \cup \tilde{B}$, é definida como

$$\mu_U(x) = \max(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)), \quad \forall x \in \Omega \quad (2.1.17)$$

Definição 2.14 Interseção: Se \tilde{A} e \tilde{B} são dois conjuntos nebulosos definidos sobre um universo de discurso Ω , a função de inclusão $\mu_I(x)$ da interseção desses dois conjuntos nebulosos, sendo $I = \tilde{A} \cap \tilde{B}$, é definida como

$$\mu_I(x) = \min(\mu_{\tilde{A}}(x), \mu_{\tilde{B}}(x)), \quad \forall x \in \Omega \quad (2.1.18)$$

Definição 2.15 Complemento: A função de inclusão $\mu_C(x)$ do complemento do conjunto nebuloso \tilde{A} definido sobre Ω , sendo $C = \overline{\tilde{A}}$, é dada como

$$\mu_C(x) = 1 - \mu_{\tilde{A}}(x), \quad \forall x \in \Omega \quad (2.1.19)$$

Existem outras definições para a União e Interseção, ver [8] e [9], como por exemplo

$$\tilde{A} \cap \tilde{B} = \mu_{\tilde{A}}(x) \cdot \mu_{\tilde{B}}(x) \quad \text{e} \quad \tilde{A} \cup \tilde{B} = \mu_{\tilde{A}}(x) + \mu_{\tilde{B}}(x) - (\mu_{\tilde{A}}(x) \cdot \mu_{\tilde{B}}(x))$$

2.1.3.3 Propriedades de Conjuntos Nebulosos

Considerando que \tilde{A} , \tilde{B} e \tilde{C} são conjuntos nebulosos contidos no universo Ω , e que \emptyset representa o conjunto vazio. As seguintes propriedades continuam valendo para conjuntos nebulosos:

1. Comutatividade:

$$\tilde{A} \cup \tilde{B} = \tilde{B} \cup \tilde{A}$$

$$\tilde{A} \cap \tilde{B} = \tilde{B} \cap \tilde{A}$$

2. Associatividade:

$$\tilde{A} \cup (\tilde{B} \cup \tilde{C}) = (\tilde{A} \cup \tilde{B}) \cup \tilde{C}$$

$$\tilde{A} \cap (\tilde{B} \cap \tilde{C}) = (\tilde{A} \cap \tilde{B}) \cap \tilde{C}$$

3. Idempotência:

$$\tilde{A} \cup \tilde{A} = \tilde{A}$$

$$\tilde{A} \cap \tilde{A} = \tilde{A}$$

4. Distributividade:

$$\tilde{A} \cup (\tilde{B} \cap \tilde{C}) = (\tilde{A} \cup \tilde{B}) \cap (\tilde{A} \cup \tilde{C})$$

$$\tilde{A} \cap (\tilde{B} \cup \tilde{C}) = (\tilde{A} \cap \tilde{B}) \cup (\tilde{A} \cap \tilde{C})$$

5. Absorção:

$$\tilde{A} \cup (\tilde{A} \cap \tilde{B}) = \tilde{A}$$

$$\tilde{A} \cap (\tilde{A} \cup \tilde{B}) = \tilde{A}$$

6. Involução:

$$\overline{\overline{\tilde{A}}} = \tilde{A}$$

7. Elemento Neutro:

$$\tilde{A} \cup \emptyset = \tilde{A}$$

$$\tilde{A} \cap \Omega = \tilde{A}$$

8. Elemento Nulo para a Interseção:

$$\tilde{A} \cap \emptyset = \emptyset$$

9. Dominação:

$$\tilde{A} \cup \Omega = \Omega$$

10. DeMorgan:

$$\overline{\tilde{A} \cap \tilde{B}} = \overline{\tilde{A}} \cup \overline{\tilde{B}}$$

$$\overline{\tilde{A} \cup \tilde{B}} = \overline{\tilde{A}} \cap \overline{\tilde{B}}$$

As duas leis a seguir **não** são válidas para conjuntos nebulosos:

- **Lei da Não Contradição:** $\tilde{A} \cap \overline{\tilde{A}} = \emptyset$

- **Lei da Exclusão do Meio:** $\tilde{A} \cup \overline{\tilde{A}} = \Omega$

Exemplo 2.4 *Considerando o conjunto dos adultos e não adultos apresentados na Figura 2.3, as arestas superiores do triângulo (representado pela área sombreada) formam a interseção dos conjuntos. É possível observar que a interseção não é vazia e portanto uma pessoa de 20 anos pode ser considerada adulta e não adulta ao mesmo tempo.*

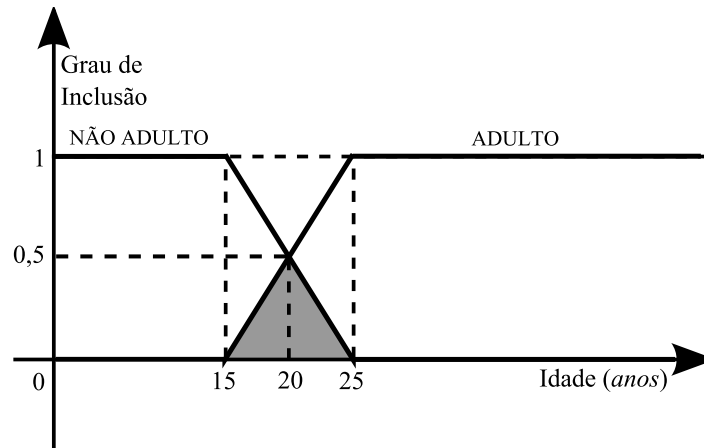


Figura 2.3: Conjuntos: pessoas adultas e pessoas não adultas

2.1.3.4 Formulação e Parametrização das Funções de Inclusão

Definição 2.16 **Números Nebulosos** *Um número nebuloso \tilde{A} é um conjunto nebuloso na reta real (\mathbb{R}) que satisfaz as condições para normalidade, dada por 2.1.15, e convexidade, dada por 2.1.16.*

De uma forma geral, um conjunto nebuloso é completamente caracterizado por sua função de inclusão. Como a maioria dos conjuntos nebulosos utilizados são descritos no universo de discurso \mathbb{R} , torna-se impraticável listar todos os pares para a definição da função de inclusão. Assim sendo, uma maneira mais conveniente de definir uma função de inclusão é expressá-la através de uma fórmula matemática.

Existem várias formas de se parametrizar uma função de inclusão, ver [8], mas nesta tese apenas as formas triangular, trapezoidal e intervalar serão utilizadas.

Definição 2.17 **Funções de Inclusão Triangulares** *Uma função de inclusão triangular é especificada por três parâmetros $\{a, b, c\}$ como segue:*

$$Triângulo(x; a, b, c) \begin{cases} 0, & x < a \\ \frac{x-a}{b-a}, & a \leq x < b \\ \frac{c-x}{c-b}, & b \leq x \leq c \\ 0, & x > c \end{cases} \quad (2.1.20)$$

Os parâmetros a, b, c (com $a < b < c$) determinam as coordenadas x de forma a desenharem um triângulo.

Exemplo 2.5 A função de inclusão triangular definida por $\text{Triângulo}(x; 20, 60, 80)$ é mostrada na Figura 2.4.

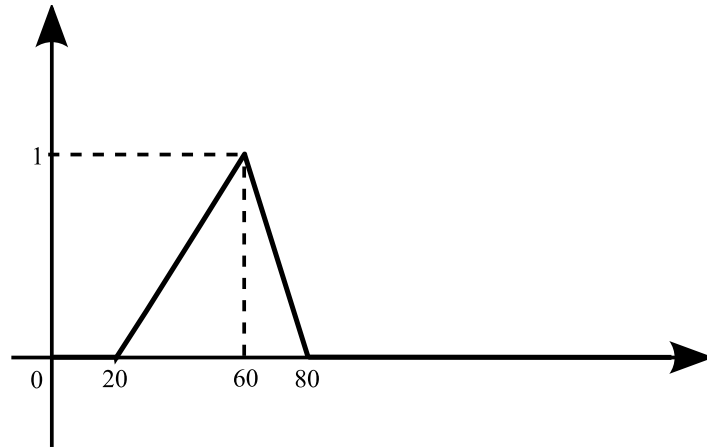


Figura 2.4: Função de inclusão triangular $\text{Triângulo}(x; 20, 60, 80)$

Definição 2.18 Funções de Inclusão Trapezoidais Uma função de inclusão trapezoidal é especificada por quatro parâmetros $\{a, b, c, d\}$ como segue:

$$\text{Trapézio}(x; a, b, c, d) \begin{cases} 0, & x < a \\ \frac{x-a}{b-a}, & a \leq x < b \\ 1, & b \leq x < c \\ \frac{d-x}{d-c}, & c \leq x \leq d \\ 0, & x > d \end{cases} \quad (2.1.21)$$

Os parâmetros a, b, c, d (com $a < b < c < d$) determinam as coordenadas x de forma a desenharem um trapézio.

Exemplo 2.6 A função de inclusão trapezoidal definida por $\text{Trapézio}(x; 10, 20, 60, 100)$ é mostrada na Figura 2.5.

Definição 2.19 Funções de Inclusão Intervalares Uma função de inclusão intervalar é uma função de inclusão Trapezoidal, ver Definição 2.18, onde $a = b$ e $c = d$. Portanto é especificada por apenas dois parâmetros $\{a, b\}$ como segue:

$$\text{Intervalo}(x; a, b) \begin{cases} 0, & x < a \text{ ou } x > b \\ 1, & a \leq x \leq b \end{cases} \quad (2.1.22)$$

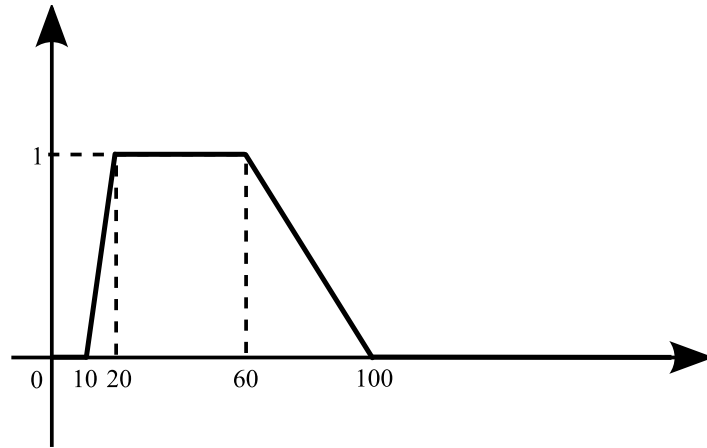


Figura 2.5: Função de inclusão trapezoidal $Trapézio(x; 10, 20, 60, 100)$

Os parâmetros a e b (com $a < b$) determinam as coordenadas x de forma a desenharem um intervalo.

Exemplo 2.7 A função de inclusão intervalar definida por $Intervalo(x; 10, 40)$ é mostrada na Figura 2.6.

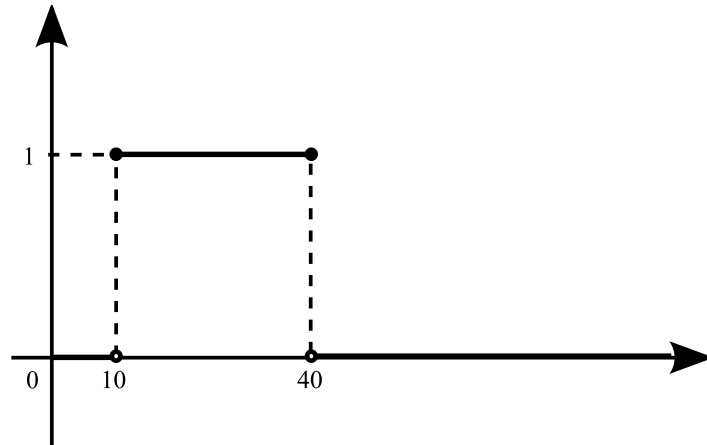


Figura 2.6: Função de inclusão intervalar $Intervalo(x; 10, 40)$

Definição 2.20 Conjuntos Nebulosos Simétricos Um conjunto nebuloso $\tilde{A} \subset \Omega$ é simétrico se sua função de inclusão é simétrica ao redor de um certo ponto $c \in \Omega$, ou seja,

$$\mu_{\tilde{A}}(c + x) = \mu_{\tilde{A}}(c - x), \quad \forall x \in \Omega \quad (2.1.23)$$

Existem vários trabalhos que mostram maiores detalhes sobre a Teoria da Lógica Nebulosa, como por exemplo [10] e [11].

2.1.3.5 Variáveis Lingüísticas

Como um conjunto convencional, um conjunto nebuloso pode ser usado para descrever o valor de uma variável. Por exemplo, a sentença “A Temperatura está ALTA” usa o conjunto nebuloso “ALTA” para descrever a temperatura.

A variável *Temperatura* exemplifica um importante conceito na lógica nebulosa: a **variável lingüística**.

A variável lingüística permite que seu valor seja descrito tanto qualitativamente por uma etiqueta lingüística (isto é, um símbolo que serve como nome de um conjunto nebuloso, ver Definição 2.4) quanto quantitativamente por uma função de inclusão correspondente (que expressa o significado do conjunto nebuloso, ver Definição 2.2).

A variável lingüística é usada para expressar conceitos e conhecimentos na comunicação humana.

2.1.3.6 Teoria de Possibilidade

A teoria de possibilidade é uma forma de teoria de informação que está relacionada, mas é independente, a conjuntos nebulosos e a teoria de probabilidade. Tecnicamente, uma distribuição de possibilidade é um conjunto nebuloso normal (com ao menos um elemento com grau de inclusão igual a 1, ver Definição 2.11). Por exemplo, todos os números nebulosos são distribuições de possibilidade. Porém, a teoria de possibilidade pode também ser derivada sem referência a conjuntos nebulosos, o que não ocorre neste trabalho uma vez que uma distribuições de possibilidade estará sempre relacionada a conjuntos nebulosos.

Um conjunto nebuloso associa-se a uma variável lingüística contendo o valor da variável, assim como um conjunto nitidamente definido faz. A diferença entre as duas associações porém é que, para conjuntos nebulosos, a noção de valores possíveis versus impossíveis vem relacionada a um grau. Esta questão será ilustrada

através de um exemplo.

Exemplo 2.8 *Supõe-se que nos arquivos policiais exista um suspeito de cometer um crime com idade entre 20 e 30 anos. Esta idade pode ser representada pelo intervalo $[20,30]$. Assim, existe a certeza que o suspeito não possui 19 nem 31 anos, por outro lado ele pode ter 20, 21, 22, ..., 30 anos. Esta situação apresenta alguns valores que são possíveis e outros que são impossíveis para a idade do suspeito. Tal raciocínio introduz uma fronteira rígida entre valores possíveis e impossíveis.*

*Para situações onde este limite é difícil de ser determinado, a lógica nebulosa oferece uma alternativa que seria o acréscimo de graus aos valores. Logo, se for acrescentado, para cada idade pertencente ao intervalo $[20,30]$, um grau de inclusão neste conjunto e sua fronteira rígida sofrer um pequeno relaxamento, será gerada uma distinção entre possível e impossível dando a cada uma delas um grau chamado de **possibilidade**.*

Por exemplo, considera-se a etiqueta Jovem definida como mostrado a seguir:

$$Jovem = \left\{ \frac{0.2}{17}, \frac{0.4}{18}, \frac{0.6}{19}, \frac{0.8}{20}, \frac{1.0}{21}, \frac{1.0}{22}, \frac{1.0}{23}, \frac{1.0}{24}, \frac{1.0}{25}, \frac{1.0}{26}, \frac{1.0}{27}, \frac{1.0}{28}, \frac{0.7}{29}, \frac{0.6}{30}, \frac{0.4}{31}, \frac{0.2}{32} \right\}$$

*Se o conjunto nebuloso **Jovem** for associado à idade do suspeito, será obtida a distribuição sobre o grau de possibilidade da idade do suspeito. Ou seja,*

$$\Pi_{Idade(Suspeito)}(x_i) = \mu_{Jovem}(x_i) \quad (2.1.24)$$

onde Π denota a distribuição de possibilidade da idade do suspeito, e x_i é a variável que representa a idade das pessoas.

Em geral, quando um conjunto nebuloso \tilde{A} é associado a uma variável X , esta associação resulta em uma distribuição de possibilidade de X , que é definida pela função de inclusão de \tilde{A} :

$$\Pi_X(x_i) = \mu_{\tilde{A}}(x_i) \quad (2.1.25)$$

A definição formal de distribuição de possibilidade está apresentada a seguir.

Definição 2.21 Distribuição de Possibilidade *Seja um conjunto nebuloso \tilde{A} definido sobre Ω com sua função de inclusão $\mu_{\tilde{A}}(x)$ e uma variável X sobre Ω (que*

desconhecemos seu valor). Então, a proposição “ X é \tilde{A} ” define uma **distribuição de possibilidade**, e desta forma conclui-se que a possibilidade de X ser \tilde{A} é $\mu_{\tilde{A}}(u)$, para todo $u \in \Omega$.

Medida de Possibilidade e Medida de Necessidade

De maneira geral, se uma distribuição de possibilidade é definida sobre um conjunto \tilde{A} e existe um elemento $x \in \tilde{A}$, a condição “ x é a ” envolve dois tipos de pergunta:

(Q1) Dada a distribuição de possibilidade $\Pi_{\tilde{A}}(x)$, é **possível** que x seja a ?

(Q2) Dada a distribuição de possibilidade $\Pi_{\tilde{A}}(x)$, é **necessário** que x seja a ?

Estas duas questões serão ilustradas usando um exemplo não nebuloso. Supõe-se que:

A idade de Maria está entre 20 e 25 anos

A seguinte condição é considerada:

Queremos uma pessoa cuja idade excede os 22 anos

A idade de Maria atende a essa condição? Existe uma informação imprecisa a respeito da idade de Maria, não é possível encontrar uma resposta exata para a questão. Mais especificamente, pode-se afirmar que é **possível** que a idade de Maria exceda 22 anos, mas não é **necessariamente** o caso.

A resposta da pergunta Q1 é chamada de uma “possibilidade”, e a resposta da pergunta Q2 de uma “necessidade”. Essas medidas de possibilidade e de necessidade foram definidas por Zadeh em [3] e [4] como apresentadas a seguir.

Definição 2.22 Medida de Possibilidade *A medida de possibilidade para uma variável X satisfazer a condição “ X é A ”, dada uma distribuição de possibilidade Π_X , é definida como*

$$Pos(A|\Pi_X) = \sup_{x_i \in \Omega} [\min(\mu_A(x_i), \Pi_X(x_i))] \quad (2.1.26)$$

sendo que Ω denota o universo de discurso da variável X .

Possibilidade e necessidade são duas medidas relacionadas. Em primeiro lugar, é possível concluir que **total necessidade implica em total possibilidade**. Segundo, que **nenhuma possibilidade implica em nenhuma necessidade**. E terceiro, a **uma variável não é possível ser “não a” se e somente se ela é necessariamente a**.

Uma generalização pode ser feita, a partir da terceira afirmação:

$$Nec(A|\Pi_X) = 1 - Pos(\neg A|\Pi_X) \quad (2.1.27)$$

Logo, é possível definir a medida de necessidade a partir da medida de possibilidade.

Definição 2.23 Medida de Necessidade *A medida de necessidade para uma variável X satisfazer a condição “ X é A ” dada uma distribuição de possibilidade Π_X é definida como*

$$Nec(A|\Pi_X) = \inf_{x_i \in \Omega} [\max(\mu_A(x_i), 1 - \Pi_X(x_i))] \quad (2.1.28)$$

sendo que Ω denota o universo de discurso da variável X .

2.1.3.7 Comparadores Estendidos

Os comparadores clássicos ($<$, $>$, \geq e \leq) necessitam ter seus conceitos estendidos para que estes possam ser aplicados sobre funções de inclusão nebulosas.

A seguir, são apresentados os comparadores nebulosos estendidos aplicados sobre X . Sendo X uma função de inclusão definida sobre o universo de discurso Ω , que pode ser um número preciso, uma função de inclusão trapezoidal, uma função de inclusão triangular e uma função de inclusão intervalar, como apresentado nas Figuras 2.7, 2.8, 2.9 e 2.10 respectivamente.

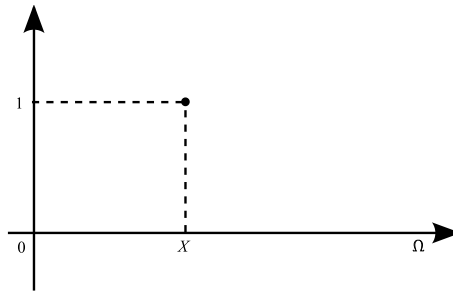


Figura 2.7: Número preciso X

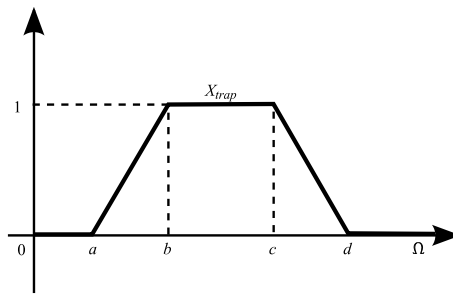


Figura 2.8: $X_{trap} = Trapézio(x; a, b, c, d)$

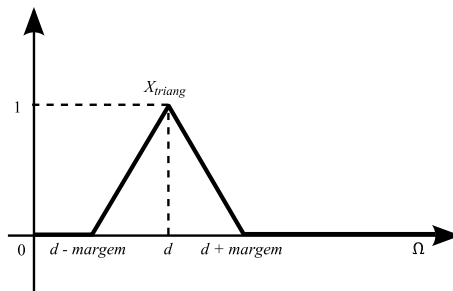


Figura 2.9: $X_{triang} = Triângulo(x; d - margem, d, d + margem)$

1. **Comparador estendido maior ou igual a ($\geq (X)$):** A distribuição de possibilidade que representa o conceito nebuloso *maior ou igual a constante* X varia de acordo com os tipos de dados que X pode assumir:

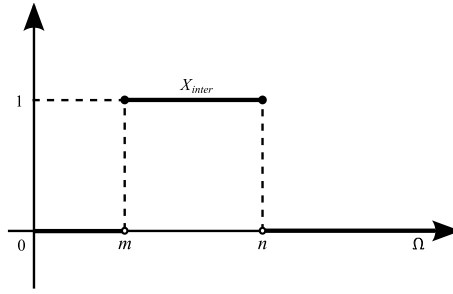


Figura 2.10: $X_{inter} = Intervalo(x; m, n)$

- **Número preciso X :**

A formalização do comparador estendido $\geq (X)$ é dada por:

$$\geq (X) = \mu_{\geq(X)}(x_i) = \begin{cases} 0, & \text{se } x_i < X \\ 1, & \text{se } x_i \geq X \end{cases} \quad (2.1.29)$$

sendo Ω o universo de discurso de X e $x_i \in \Omega$.

A função de inclusão do comparador estendido $\geq (X)$ é apresentada na Figura 2.11

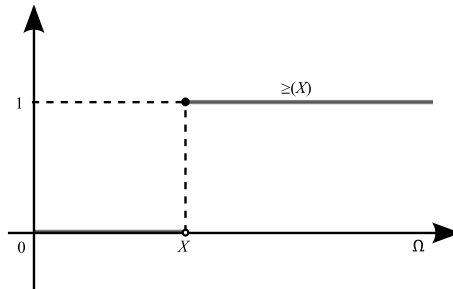


Figura 2.11: Comparador estendido $\geq (X)$

Neste caso, \geq é o próprio operador “maior ou igual” clássico.

- **Função de Inclusão Trapezoidal $X_{trap} = Trapézio(x; a, b, c, d)$:**

A formalização do comparador estendido $\geq (X_{trap})$ é dada por:

$$\geq (X_{trap}) = \mu_{\geq(X_{trap})}(x_i) = \begin{cases} 0, & \text{se } x_i \leq a \\ \frac{x_i - a}{b - a}, & \text{se } a < x_i < b \\ 1, & \text{se } x_i \geq b \end{cases} \quad (2.1.30)$$

sendo Ω o universo de discurso de X_{trap} e $x_i \in \Omega$.

A função de inclusão do comparador estendido $\geq (X_{trap})$ é apresentada na Figura 2.12

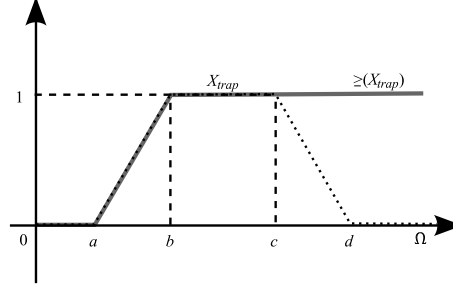


Figura 2.12: Comparador estendido $\geq (X_{trap})$

- **Função de Inclusão Triangular** $X_{triang} = Triângulo(x; d - margem, d, d + margem)$:

A formalização do comparador estendido $\geq (X_{triang})$ é dada por:

$$\geq (X_{triang}) = \mu_{\geq(X_{triang})}(x_i) = \begin{cases} 0, & \text{se } x_i \leq d - m_d \\ \frac{x_i - d + m_d}{m_d}, & \text{se } d - m_d < x_i < d \\ 1, & \text{se } x_i \geq d \end{cases} \quad (2.1.31)$$

sendo Ω o universo de discurso de X_{triang} , $x_i \in \Omega$ e m_d a *margem* considerada.

A função de inclusão do comparador estendido $\geq (X_{triang})$ é apresentada na Figura 2.13

- **Função de Inclusão Intervalar** $X_{inter} = Intervalo(x; m, n)$:

A formalização do comparador estendido $\geq (X_{inter})$ é dada por:

$$\geq (X_{inter}) = \mu_{\geq(X_{inter})}(x_i) = \begin{cases} 0, & \text{se } x_i < m \\ 1, & \text{se } x_i \geq m \end{cases} \quad (2.1.32)$$

sendo Ω o universo de discurso de X_{inter} e $x_i \in \Omega$.

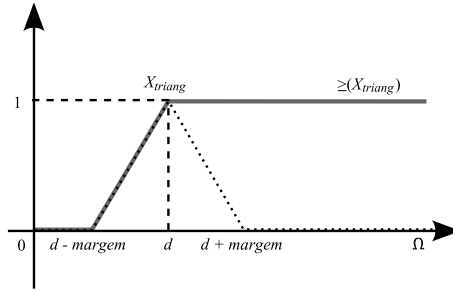


Figura 2.13: Comparador estendido $\geq (X_{triang})$

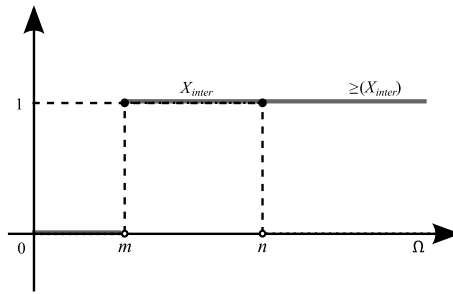


Figura 2.14: Comparador estendido $\geq (X_{inter})$

A função de inclusão do comparador estendido $\geq (X_{inter})$ é apresentada na Figura 2.14

2. **Comparador estendido menor ou igual a ($\leq (X)$):** A distribuição de possibilidade que representa o conceito nebuloso *menor ou igual a constante* X varia de acordo com o tipo do dado X :

- **Número preciso X :**

A formalização do comparador estendido $\leq (X)$ é dada por:

$$\leq (X) = \mu_{\leq(X)}(x_i) = \begin{cases} 0, & \text{se } x_i > X \\ 1, & \text{se } x_i \leq X \end{cases} \quad (2.1.33)$$

sendo Ω o universo de discurso de X e $x_i \in \Omega$.

A função de inclusão do comparador estendido $\leq (X)$ é apresentada na Figura 2.15

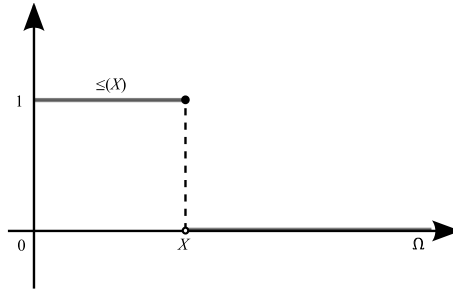


Figura 2.15: Comparador estendido $\leq (X)$

Neste caso, \leq é o operador “menor ou igual” clássico.

- **Função de Inclusão Trapezoidal** $X_{trap} = \text{Trapézio}(x; a, b, c, d)$:

A formalização do comparador estendido $\leq (X_{trap})$ é dada por:

$$\leq (X_{trap}) = \mu_{\leq(X_{trap})}(x_i) = \begin{cases} 0, & \text{se } x_i \geq d \\ \frac{d - x_i}{d - c}, & \text{se } c < x_i < d \\ 1, & \text{se } x_i \leq c \end{cases} \quad (2.1.34)$$

sendo Ω o universo de discurso de X_{trap} e $x_i \in \Omega$.

A função de inclusão do comparador estendido $\leq (X_{trap})$ é apresentada na Figura 2.16

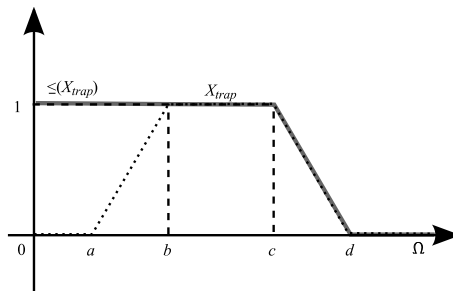


Figura 2.16: Comparador estendido $\leq (X_{trap})$

- **Função de Inclusão Triangular** $X_{triang} = \text{Triângulo}(x; d - \text{margem}, d, d + \text{margem})$:

A formalização do comparador estendido $\leq (X_{triang})$ é dada por:

$$\leq (X_{triang}) = \mu_{\leq(X_{triang})}(x_i) = \begin{cases} 0, & \text{se } x_i \geq d + m_d \\ \frac{d + m_d - x_i}{m_d}, & \text{se } d < x_i < d + m_d \\ 1, & \text{se } x_i \leq d \end{cases} \quad (2.1.35)$$

sendo Ω o universo de discurso de X_{triang} , $x_i \in \Omega$ e m_d a *margem* considerada.

A função de inclusão do comparador estendido $\leq (X_{triang})$ é apresentada na Figura 2.17

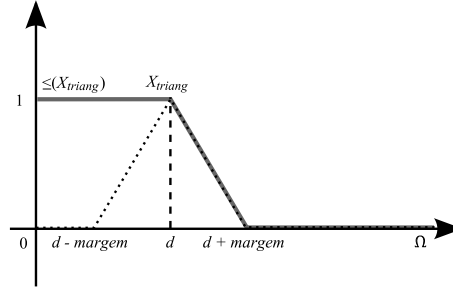


Figura 2.17: Comparador estendido $\leq (X_{triang})$

- **Função de Inclusão Intervalar** $X_{inter} = Interlo(x; m, n)$:

A formalização do comparador estendido $\leq (X_{inter})$ é dada por:

$$\leq (X_{inter}) = \mu_{\leq(X_{inter})}(x_i) = \begin{cases} 0, & \text{se } x_i > n \\ 1, & \text{se } x_i \leq n \end{cases} \quad (2.1.36)$$

sendo Ω o universo de discurso de X_{inter} e $x_i \in \Omega$.

A função de inclusão do comparador estendido $\leq (X_{inter})$ é apresentada na Figura 2.18

3. **Comparador estendido maior que ($> (X)$):** Este operador é definido a partir do complemento do operador “menor ou igual a”:

$$\mu_{>(X)}(x_i) = > (X) = 1 - \leq (X) \quad (2.1.37)$$

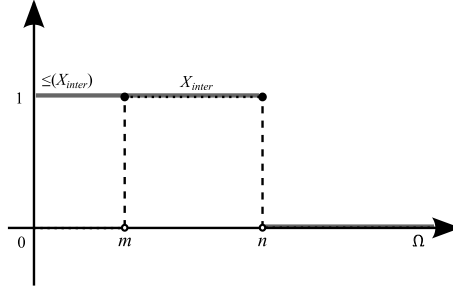


Figura 2.18: Comparador estendido $\leq (X_{inter})$

sendo Ω o universo de discurso de X e $x_i \in \Omega$.

A distribuição de possibilidade que representa o conceito nebuloso *maior que a constante* X varia de acordo com o tipo do dado X .

A formalização do comparador estendido *maior que* está apresentada nas Equações 2.1.38, 2.1.39, 2.1.40 e 2.1.41 de acordo com os tipos de dados que X pode assumir, sendo respectivamente um número preciso, uma função de inclusão trapezoidal, uma função de inclusão triangular e uma função de inclusão intervalar:

$$> (X) = \mu_{>(X)}(x_i) = \begin{cases} 0, & \text{se } x_i \leq X \\ 1, & \text{se } x_i > X \end{cases} \quad (2.1.38)$$

sendo Ω o universo de discurso de X e $x_i \in \Omega$.

$$> (X_{trap}) = \mu_{>(X_{trap})}(x_i) = \begin{cases} 0, & \text{se } x_i \leq c \\ \frac{x_i - c}{d - c}, & \text{se } c < x_i < d \\ 1, & \text{se } x_i \geq d \end{cases} \quad (2.1.39)$$

sendo Ω o universo de discurso de X_{trap} e $x_i \in \Omega$.

$$> (X_{triang}) = \mu_{>(X_{triang})}(x_i) = \begin{cases} 0, & \text{se } x_i \leq d \\ \frac{x_i - d}{m_d}, & \text{se } d < x_i < d + m_d \\ 1, & \text{se } x_i \geq d + m_d \end{cases} \quad (2.1.40)$$

sendo Ω o universo de discurso de X_{triang} , $x_i \in \Omega$ e m_d a *margem* considerada.

$$\gt (X_{inter}) = \mu_{\gt(X_{inter})}(x_i) = \begin{cases} 0, & \text{se } x_i \leq n \\ 1, & \text{se } x_i > n \end{cases} \quad (2.1.41)$$

sendo Ω o universo de discurso de X_{Inter} e $x_i \in \Omega$.

As funções de inclusão do comparador estendido *maior que* estão apresentadas nas Figuras 2.19, 2.20, 2.21 e 2.22 de acordo com os tipos de dados que X pode assumir, sendo respectivamente um número preciso, uma função de inclusão trapezoidal, uma função de inclusão triangular e uma função de inclusão intervalar:

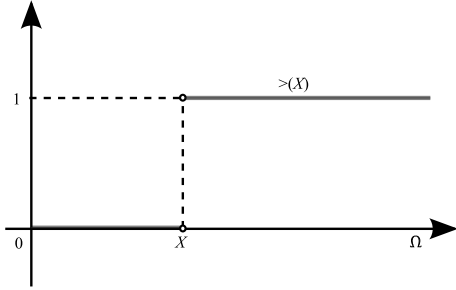


Figura 2.19: Comparador $\gt (X)$

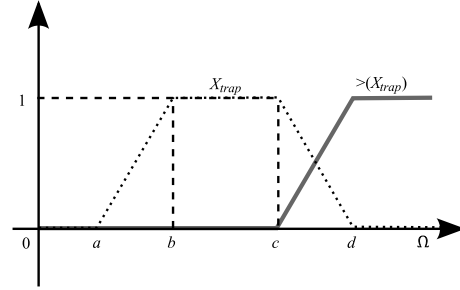


Figura 2.20: Comparador $\gt (X_{trap})$

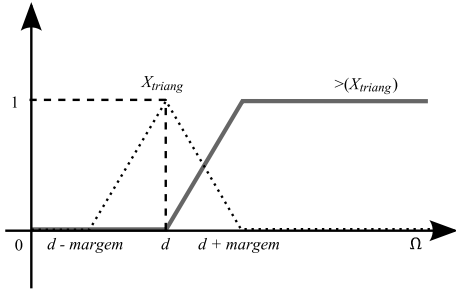


Figura 2.21: Comparador $\gt (X_{triang})$

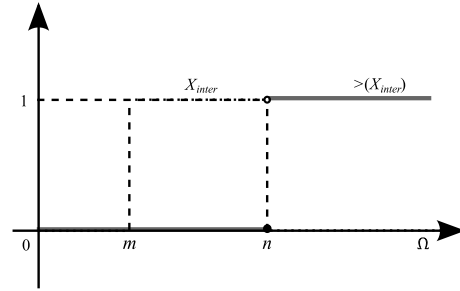


Figura 2.22: Comparador $\gt (X_{inter})$

4. **Comparador estendido menor que ($\lt (X)$):** Este operador é definido a partir do complemento do operador “*maior ou igual a*”:

$$\mu_{\lt(X)}(x_i) = \lt (X) = 1 - \geq (X) \quad (2.1.42)$$

sendo Ω o universo de discurso de X e $x_i \in \Omega$.

A distribuição de possibilidade que representa o conceito nebuloso *menor que a constante* X varia de acordo com o tipo do dado X .

A formalização do comparador estendido *menor que* está apresentada nas Equações 2.1.43, 2.1.44, 2.1.45 e 2.1.46 de acordo com os tipos de dados que X pode assumir, sendo respectivamente um número preciso, uma função de inclusão trapezoidal, uma função de inclusão triangular e uma função de inclusão intervalar:

$$\langle (X) = \mu_{\langle(X)}(x_i) = \begin{cases} 0, & \text{se } x_i \geq X \\ 1, & \text{se } x_i < X \end{cases} \quad (2.1.43)$$

sendo Ω o universo de discurso de X e $x_i \in \Omega$.

$$\langle (X_{trap}) = \mu_{\langle(X_{trap})}(x_i) = \begin{cases} 0, & \text{se } x_i \geq b \\ \frac{b - x_i}{b - a}, & \text{se } a < x_i < b \\ 1, & \text{se } x_i \leq a \end{cases} \quad (2.1.44)$$

sendo Ω o universo de discurso de X_{trap} e $x_i \in \Omega$.

$$\langle (X_{triang}) = \mu_{\langle(X_{triang})}(x_i) = \begin{cases} 0, & \text{se } x_i \geq d \\ \frac{d - x_i}{m_d}, & \text{se } d - m_d < x_i < d \\ 1, & \text{se } x_i \leq d - m_d \end{cases} \quad (2.1.45)$$

sendo Ω o universo de discurso de X_{triang} , $x_i \in \Omega$ e m_d a *margem* considerada.

$$\langle (X_{inter}) = \mu_{\langle(X_{inter})}(x_i) = \begin{cases} 0, & \text{se } x_i \geq m \\ 1, & \text{se } x_i < m \end{cases} \quad (2.1.46)$$

sendo Ω o universo de discurso de X_{inter} e $x_i \in \Omega$.

As funções de inclusão do comparador estendido *menor que* estão apresentadas nas Figuras 2.23, 2.24, 2.25 e 2.26 de acordo com os tipos de dados

que X pode assumir, sendo respectivamente um número preciso, uma função de inclusão trapezoidal, uma função de inclusão triangular e uma função de inclusão intervalar:

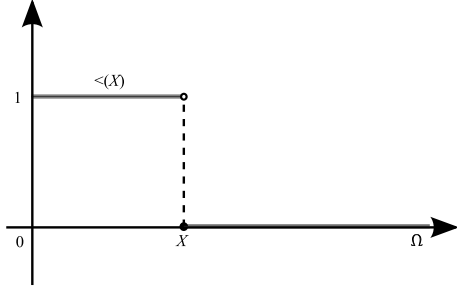


Figura 2.23: Comparador $<(X)$

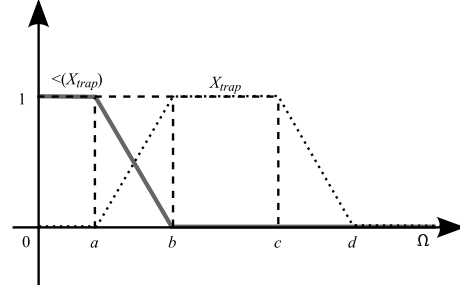


Figura 2.24: Comparador $<(X_{trap})$

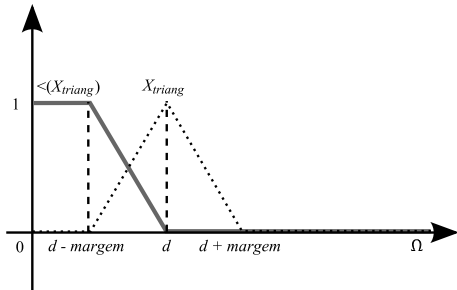


Figura 2.25: Comparador $<(X_{triang})$

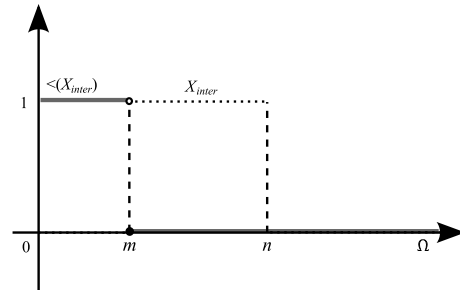


Figura 2.26: Comparador $<(X_{inter})$

5. **Comparador estendido muito maior que ($>_{muito}(X)$):** Este operador é definido a partir do operador “maior que”. É acrescentado ao operador maior que uma distância M :

$$\mu_{>_{muito}(X)}(x_i) = >_{muito}(X) = >(X) + M \quad (2.1.47)$$

sendo Ω o universo de discurso de X , $x_i \in \Omega$, e M a distância considerada.

A distribuição de possibilidade que representa o conceito nebuloso *muito maior que a constante X* varia de acordo com o tipo do dado X .

A formalização do comparador estendido *muito maior que* está apresentada nas Equações 2.1.48, 2.1.49, 2.1.50 e 2.1.51 de acordo com os tipos de dados que X pode assumir, sendo respectivamente um número preciso, uma função

de inclusão trapezoidal, uma função de inclusão triangular e uma função de inclusão intervalar:

$${}_{>}\text{muito}(X) = \mu_{>}\text{muito}(X)(x_i) = \begin{cases} 0, & \text{se } x_i \leq X + M \\ 1, & \text{se } x_i > X + M \end{cases} \quad (2.1.48)$$

sendo Ω o universo de discurso de X , $x_i \in \Omega$, e M a distância considerada.

$${}_{>}\text{muito}(X_{\text{trap}}) = \mu_{>}\text{muito}(X_{\text{trap}})(x_i) = \begin{cases} 0, & \text{se } x_i \leq c + M \\ \frac{x_i - c - M}{d - c}, & \text{se } c + M < x_i < d + M \\ 1, & \text{se } x_i \geq d + M \end{cases} \quad (2.1.49)$$

sendo Ω o universo de discurso de X_{trap} , $x_i \in \Omega$ e M a distância considerada.

$${}_{>}\text{muito}(X_{\text{triang}}) = \mu_{>}\text{muito}(X_{\text{triang}})(x_i) = \begin{cases} 0, & \text{se } x_i \leq d + M \\ \frac{x_i - d - M}{m_d}, & \text{se } d + M < x_i \text{ e} \\ & x_i < d + m_d + M \\ 1, & \text{se } x_i \geq d + m_d + M \end{cases} \quad (2.1.50)$$

sendo Ω o universo de discurso de X_{triang} , $x_i \in \Omega$, m_d a *margem* e M a distância considerada.

$${}_{>}\text{muito}(X_{\text{inter}}) = \mu_{>}\text{muito}(X_{\text{inter}})(x_i) = \begin{cases} 0, & \text{se } x_i \leq n + M \\ 1, & \text{se } x_i > n + M \end{cases} \quad (2.1.51)$$

sendo Ω o universo de discurso de X_{inter} , $x_i \in \Omega$ e M a distância considerada.

As funções de inclusão do comparador estendido *muito maior que* estão apresentadas nas Figuras 2.27, 2.28, 2.29 e 2.30 de acordo com os tipos de dados que X pode assumir, sendo respectivamente um número preciso, uma função de inclusão trapezoidal, uma função de inclusão triangular e uma função de inclusão intervalar:

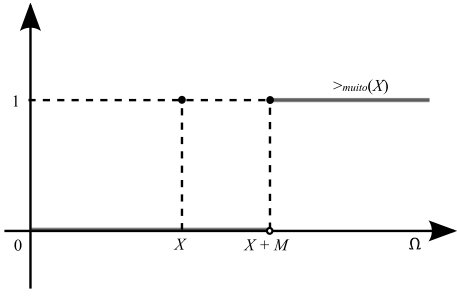


Figura 2.27: Comparador $>_{muito}(X)$

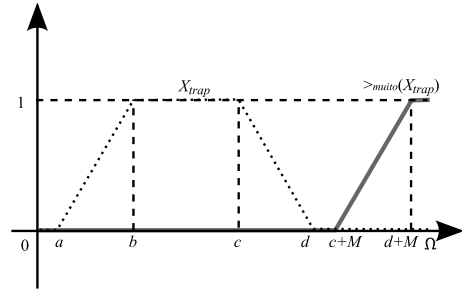


Figura 2.28: Comparador $>_{muito}(X_{trap})$

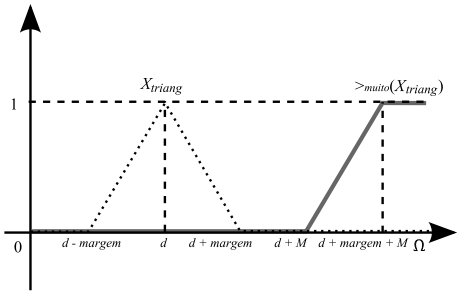


Figura 2.29: Comparador $>_{muito}(X_{triang})$

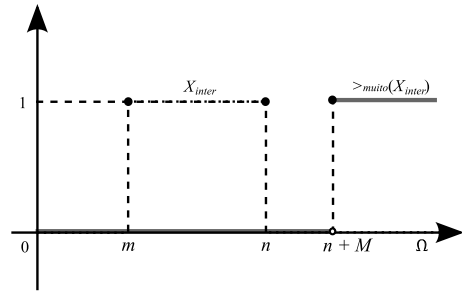


Figura 2.30: Comparador $>_{muito}(X_{inter})$

6. **Comparador estendido muito menor que ($<_{muito}(X)$):** Este operador é definido a partir do operador “menor que”. É retirado ao operador menor que uma distância M :

$$\mu_{<_{muito}(X)}(x_i) = <_{muito}(X) = <(X) - M \quad (2.1.52)$$

sendo Ω o universo de discurso de X , $x_i \in \Omega$, e M a distância considerada.

A distribuição de possibilidade que representa o conceito nebuloso *muito menor que a constante X* varia de acordo com o tipo do dado X .

A formalização do comparador estendido *muito menor que* está apresentada nas Equações 2.1.53, 2.1.54, 2.1.55 e 2.1.56 de acordo com os tipos de dados que X pode assumir, sendo respectivamente um número preciso, uma função de inclusão trapezoidal, uma função de inclusão triangular e uma função de inclusão intervalar:

$$\leq_{\text{muito}}(X) = \mu_{\leq_{\text{muito}}(X)}(x_i) = \begin{cases} 0, & \text{se } x_i \geq X - M \\ 1, & \text{se } x_i < X - M \end{cases} \quad (2.1.53)$$

sendo Ω o universo de discurso de X , $x_i \in \Omega$, e M a distância considerada.

$$\leq_{\text{muito}}(X_{\text{trap}}) = \mu_{\leq_{\text{muito}}(X_{\text{trap}})}(x_i) = \begin{cases} 0, & \text{se } x_i \geq b - M \\ \frac{b - M - x_i}{b - a}, & \text{se } a - M < x_i < b - M \\ 1, & \text{se } x_i \leq a - M \end{cases} \quad (2.1.54)$$

sendo Ω o universo de discurso de X_{trap} , $x_i \in \Omega$ e M a distância considerada.

$$\leq_{\text{muito}}(X_{\text{triang}}) = \mu_{\leq_{\text{muito}}(X_{\text{triang}})}(x_i) = \begin{cases} 0, & \text{se } x_i \geq d - M \\ \frac{d - M - x_i}{m_d}, & \text{se } d - m_d - M < x_i \\ & \text{e } x_i < d - M \\ 1, & \text{se } x_i \leq d - m_d - M \end{cases} \quad (2.1.55)$$

onde Ω o universo de discurso de X_{triang} , $x_i \in \Omega$, m_d a *margem* e M a distância considerada.

$$\leq_{\text{muito}}(X_{\text{inter}}) = \mu_{\leq_{\text{muito}}(X_{\text{inter}})}(x_i) = \begin{cases} 0, & \text{se } x_i \geq m - M \\ 1, & \text{se } x_i < m - M \end{cases} \quad (2.1.56)$$

onde Ω o universo de discurso de X_{inter} , $x_i \in \Omega$ e M a distância considerada.

As funções de inclusão do comparador estendido *muito menor que* estão apresentadas nas Figuras 2.31, 2.32, 2.33 e 2.34 de acordo com os tipos de dados que X pode assumir, sendo respectivamente um número preciso, uma função de inclusão trapezoidal, uma função de inclusão triangular e uma função de inclusão intervalar:

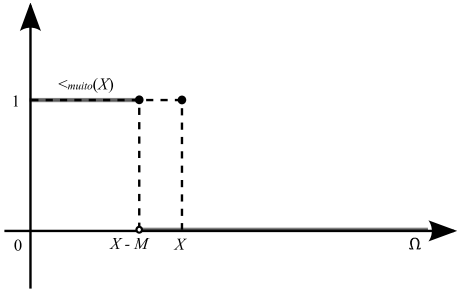


Figura 2.31: Comparador $\langle_{\text{muito}}(X)$

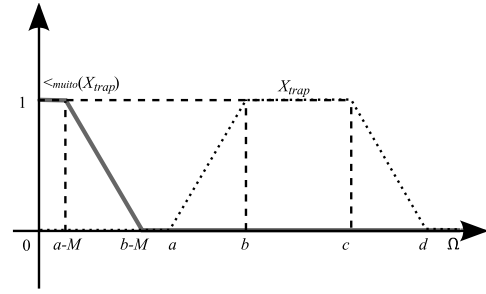


Figura 2.32: Comparador $\langle_{\text{muito}}(X_{\text{trap}})$

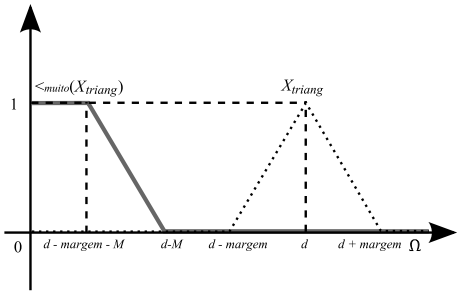


Figura 2.33: Comparador $\langle_{\text{muito}}(X_{\text{triang}})$

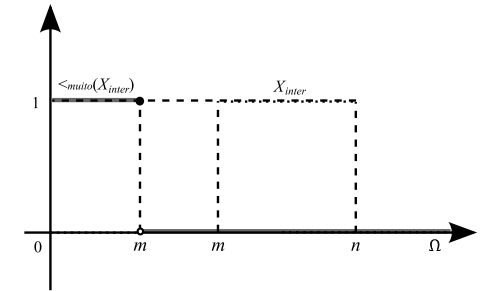


Figura 2.34: Comparador $\langle_{\text{muito}}(X_{\text{inter}})$

Os Comparadores Estendidos serão utilizados mais adiante na Seção 4.4 como base para a definição dos Comparadores Nebulosos usados na construção da linguagem FSQL do Banco de Dados Nebulosos - *Aliança*.

2.2 Visão Geral de SGBDs Relacionais

O objetivo desta seção é apresentar as principais características dos SGBD Relacionais. Maiores detalhes de toda esta teoria podem ser encontrados na literatura, por exemplo em [12] e [13].

2.2.1 Histórico

Os bancos de dados possuem suas origens em bibliotecas, negócios, informações de governos, e anotações médicas. Existe uma longa história do armazenamento, indexação, e recuperação de informação, com seus sucessos e falhas.

Nos anos 60, os computadores se tornaram financeiramente acessíveis para companhias privadas e sua capacidade de armazenamento foi ampliada. Dois principais modelos de dados foram desenvolvidos: modelo de redes (CODASYL) e hierárquico (IMS). O acesso aos dados era feito através de operações de baixo nível entre ligações de ponteiros e registros. Os detalhes do armazenamento dependiam dos tipos de dados a serem armazenados. Assim, adicionar um campo extra a base de dados requeria reescrever o esquema subjacente de acesso/modificação. A ênfase estava nos registros que seriam processados. Um usuário necessitava conhecer a estrutura física da base de dados para poder executar consultas sobre as informações.

Entre 1970 e 1972 o modelo relacional proposto por Codd em [14] para bases de dados teve um papel fundamental. Codd separou o esquema (organização lógica) de uma base de dados de seus métodos físicos de armazenamento. Este sistema vem sendo o padrão desde então.

Nos anos 70, a teoria das bases de dados foi conduzida aos projetos de pesquisa. Dois importantes protótipos para sistemas relacionais foram desenvolvidos durante 1974-77 O Sistema Ingres [15] e o Sistema R [16].

O Sistema Ingres, desenvolvido em UCB, gerou a Ingres Corp., Sybase, MS SQL Server, Britton-Lee, Wang's PACE. Este sistema usava QUEL como linguagem de consulta.

O Sistema R foi desenvolvido pela IBM San Jose e gerou SQL/DS da IBM & DB2, Oracle, HP's Allbase, Tandem's Non-Stop SQL. A linguagem de consulta SEQUEL era usada por esse sistema.

O termo "sistema de gerenciamento de banco de dados relacional" (SGBDR) foi inventado durante este período.

Em 1976, Chen [17] propôs o modelo do Entidade-Relacionamento (ER) para o projeto de base de dados, o que deu uma outra visão importante aos modelos conceituais dos dados. Permitindo ao desenvolvedor modelar em alto nível, fazendo com que este se concentrasse no uso dos dados em vez da estrutura lógica da tabela.

No início dos anos 80 a comercialização dos sistemas relacionais sofreu um crescimento acentuado.

Nos anos 80, o SQL (Structured Query Language) tornou-se padrão. O DB2 tornou-se o produto de maior importância da IBM. Os modelos de rede e o hierárquico foram para o segundo plano, e hoje em dia não há essencialmente nenhum desenvolvimento destes sistemas, apesar de ainda serem usados.

No início dos anos 90, a indústria sofreu transformações e poucas companhias sobreviveram devido ao fato de oferecerem produtos cada vez mais complexos e com preços elevados. Muitos dos desenvolvimentos, durante este período, tiveram seu foco em ferramentas para o cliente com a finalidade do desenvolvimento de aplicações tais como PowerBuilder (Sybase), Oracle Developer, VB (Microsoft) etc. O modelo cliente servidor passou a ser amplamente utilizado. Ocorreu o surgimento de ferramentas pessoais de produtividade tais como Excel/Access (MS) e ODBC. O que marcou também o começo de protótipos dos sistemas de gerência da base de dados orientados a objeto.

Em meados da década de 90 ocorreu o surgimento da Internet. O acesso remoto aos sistemas computadorizados passou a ser permitido. O modelo cliente servidor alcançou os usuários médios, com pouca paciência para complexidade, enquanto Web/DB cresceu de forma exponencial.

No final dos anos 90 ocorreu um grande investimento das companhias na Internet, o que causou o crescimento do mercado das ferramentas para conectores de Web/ Internet/ BD. Active Server Pages, Front Page, Java Servlets, JDBC, Enterprise Java Beans, ColdFusion, Dream Weaver, Oracle Developer 2000, etc são exemplos de tais ferramentas. A solução código fonte aberto veio junto com o uso difundido do gcc, cgi, Apache, MySQL etc. O processamento de transações online (OLTP) e o processamento analítico online (OLAP) veio com o uso da tecnologia point-of-sale (POS) por muitos comerciantes em bases diárias.

No início do século XXI aplicações mais interativas aparecem com uso de PDAs, transações de POS, consolidação dos vendedores, etc.

2.2.2 Modelo Relacional

Um banco de dados é uma coleção de dados relacionados a alguns fenômenos reais que estamos tentando modelar.

O **Modelo relacional** de banco de dados foi proposto por Cood da IBM em 1970 [14], com o intuito de simplificar a estrutura de bancos de dados, que chegava a ser muito complexa em outros modelos.

O modelo relacional pode ser caracterizado por pelo menos três recursos poderosos, ver [12]:

1. Suas estruturas de dados são simples. Elas são relações formadas por janelas bidimensionais cujos elementos são itens de dados. Isso permite um alto grau de independência da representação de dados.
2. O modelo relacional oferece uma fundação sólida para a consistência de dados. O projeto de banco de dados é auxiliado pelo processo de normalização que elimina anomalias de dados. Além disso, estados consistentes de um banco de dados podem ser definidos de maneira uniforme e mantidos por regras de integridade.
3. O modelo relacional permite a manipulação de relações orientadas a conjuntos. Essa característica levou ao desenvolvimento de linguagens não procedurais poderosas baseadas na teoria dos conjuntos (álgebra relacional) ou em lógica (cálculo relacional).

Definição 2.24 Banco de Dados Relacional *É um banco de dados onde todos os dados visíveis ao usuário estão armazenados em tabelas (ou relações) de dados e onde todas as operações do banco de dados trabalham sobre as tabelas e/ou produzem novas tabelas. O SGBD relacional é um sistema de gerenciamento de banco de dados que trabalha sobre banco de dados relacionais.*

2.2.3 Estruturas de Dados

A estrutura de dados, em um banco de dados relacional, está associada ao conceito de tabelas. Uma vez que um modelo relacional organiza dados em tabelas, cada coluna é chamada de atributo, e as linhas são as instâncias, chamadas de tuplas. Ver Tabela 2.1.

Tabela 2.1: Relação Perfil dos Funcionários de uma Empresa

	Atributo			
	↓			
	Nome	Sobrenome	Tipo de trabalho	Especialidade
Tupla ⇒	Paulo	Oliveira	Acadêmico	IA
	José	Reis	Indústria	Estatística

Definição 2.25 Domínio *É o conjunto de todos os valores possíveis que podem ser atribuídos a um atributo. Assim, todo atributo possui um domínio associado a ele.*

Definição 2.26 Atributo *É a instância de um domínio em uma tabela. Atributos são cada uma das características de um objeto que são armazenadas em um banco de dados. Corresponde a cada uma das colunas de uma tabela.*

Definição 2.27 Tupla *É o conjunto de todos os valores concretos de todos os atributos de um objeto. Corresponde a cada uma das linhas de uma tabela.*

Definição 2.28 Tabela ou Relação *É composta de atributos e domínios, cuja instância é constituída de uma tupla.*

Definição 2.29 Chave Primária *É o subconjunto não-vazio mínimo de atributos de uma tabela, tal que os valores dos atributos que constituem a chave sejam capazes de identificar univocamente cada tupla da tabela.*

Definição 2.30 Chave Estrangeira *É o subconjunto de atributos de uma tabela que é chave primária em outra tabela distinta e que é usado para relacionar os dados de ambas as tabelas.*

2.2.4 Manipulação dos Dados

As linguagens de manipulação de dados desenvolvidas para o modelo relacional se dividem em dois grupos: são as linguagens baseadas na *álgebra relacional* e as linguagens baseadas no *cálculo relacional*. Ambas as linguagens foram propostas originalmente por Codd, que também provou que elas eram equivalentes em termos de poder de expressão.

2.2.4.1 Álgebra Relacional

A álgebra relacional é procedural. No sentido que o usuário deve especificar, empregando certos operadores de alto nível, o modo como o resultado deve ser obtido.

A álgebra relacional consiste em um conjunto de operações que usam uma ou duas relações como entrada e produzem uma nova relação como resultado.

As operações fundamentais da álgebra relacional são:

- **Seleção.** Esta operação seleciona tuplas de uma relação R que satisfazem uma certa condição X , que pode ser simples ou composta. A operação de seleção é representada por: $\sigma_X(R)$.
- **Projeção.** Esta é uma operação unária. Fornece como resultado uma relação R , com certas colunas deixadas de fora, ou seja, produz um subconjunto “vertical” da relação R .
A operação de projeção é representada por: $\Pi_{atributo_1, atributo_2, \dots, atributo_n}(R)$.
- **Produto Cartesiano.** O produto cartesiano é uma operação binária. O produto cartesiano de duas relações R e S , representado por $R \times S$, consiste de todas as possíveis combinações de pares de tuplas, uma de cada relação.
- **União.** A união é uma operação binária. A união de duas relações R e S , representada por $R \cup S$, é o conjunto de todas as tuplas que estão em R ou em S , ou em ambas.

Para a operação $R \cup S$ ser válida, são necessárias duas condições:

1. As relações R e S devem possuir o mesmo número de atributos.
2. Os domínios do i -ésimo atributo de R e do i -ésimo atributo de S devem ser os mesmos.

- **Diferença.** A diferença é uma operação binária. A diferença de duas relações R e S , representada por $R - S$, é o conjunto de todas as tuplas que estão em R , mas não estão em S .

Existem outras operações além das fundamentais:

- **Interseção.** A interseção de duas relações R e S , representada por $R \cap S$, nos dá como resultado o conjunto daquelas tuplas que pertencem a R e a S . A expressão que usa interseção pode ser reescrita substituindo-se a operação interseção por um par de diferenças de conjuntos: $R \cap S = R - (R - S)$

- **Junção Natural.** A operação de junção natural, representada por $R \bowtie S$, forma um produto cartesiano de R e S , faz uma seleção forçando uma igualdade sobre os atributos que aparecem em ambas relações e finalmente remove colunas duplicadas.

Se R e S são relações que não possuem atributos em comum, então $R \bowtie S = R \times S$.

- **Divisão.** Tomemos as duas relações R e S , onde R possui o número de atributos igual a r e S possui o número de atributos igual a s , sendo $r > s$ e $s \neq 0$. A divisão de R por S , representada por $R \div S$ é o conjunto de $(r-s)$ -tuplas t tais que, para todas as s -tuplas u em S , a tupla tu está em R . A operação de divisão pode ser especificada em termos de operadores fundamentais como segue: $R \div S = \Pi_{r-s}(R) - \Pi_{r-s}(\Pi_{r-s}(R) \times S) - R$

- **Atribuição.** Representado por \leftarrow , funciona de maneira similar à atribuição numa linguagem de programação. A expressão $T \leftarrow (R \cup S)$ indica que o resultado da operação $R \cup S$ é atribuído a variável do tipo de relação T .

2.2.4.2 Cálculo Relacional

O cálculo relacional está baseado em um ramo da Matemática chamado de Cálculo dos Predicados.

Quando se escreve uma expressão da álgebra relacional, especifica-se uma seqüência de procedimentos que geram respostas à nossa consulta. O cálculo relacional, por contraste, é uma linguagem de consulta não-procedural. Ele descreve a informação desejada sem dar um procedimento específico para obter tal informação.

Um banco de dados pode ser visto como uma coleção de tuplas ou uma coleção de domínios, assim as linguagens do cálculo relacional se dividem em dois grupos: *cálculo relacional de tuplas* e *cálculo relacional de domínios*.

O *cálculo relacional de tuplas* interpreta uma variável em uma fórmula como uma tupla em uma relação, enquanto o *cálculo relacional de domínios* interpreta uma variável como o valor de um domínio.

A seguir daremos um breve resumo das características principais do cálculo relacional de tuplas, por ser de mais simples entendimento.

Uma consulta do cálculo relacional de tuplas é representada como

$$\{t \mid P(t)\}$$

que é o conjunto de todas as tuplas t para as quais o predicado P é verdadeiro.

Os elementos sobre os quais se constrói o cálculo relacional de tuplas são os seguintes:

- **Variável Tupla.** Consiste em uma variável sobre a qual se pode associar valores específicos assumidos por uma tupla. Cada variável tupla vem restringida por um domínio.
- **Quantificadores.** Uma variável tupla é chamada de variável livre, a menos que esteja quantificada por um \forall ou um \exists .
- **Expressões** Uma fórmula do cálculo relacional é construída a partir de átomos. Um átomo tem uma das seguintes formas:

- $s \in R$, onde s é uma variável tupla e R é uma relação.
- $s[x] \Theta u[y]$, onde s e u são variáveis tupla, x é um atributo em que s é definido, y é um atributo no qual u é definido, e Θ é um operador de comparação ($=, <, >, \neq, \leq, \geq$). É necessário que os atributos x e y tenham domínios cujos membros possam ser comparados por Θ .
- $s[x] \Theta c$, onde s é uma variável tupla, x é um atributo no qual s é definido, Θ é um operador de comparação, e c é uma constante no domínio de atributo x .

As fórmulas são construídas a partir de átomos seguindo uma das seguintes regras:

- Um átomo é uma fórmula.
- Se P_1 é uma fórmula, então também são $\neg P_1$ e (P_1) .
- Se P_1 e P_2 são fórmulas, então também são $P_1 \wedge P_2$, $P_1 \vee P_2$ e $P_1 = P_2$.
- Se $P_1(s)$ é uma fórmula contendo uma variável tupla livre s , então também são $\exists s \in r(P_1(s))$ e $\forall s \in r(P_1(s))$.

Há várias linguagens baseadas no cálculo relacional de tuplas, sendo a mais popular delas a SQL.

2.2.5 A Linguagem SQL

A linguagem SQL (*Structured Query Language*) oferece uma abordagem uniforme para a manipulação de dados (recuperação, atualização), definição de dados (manipulação de esquema) e controle (autorização, integridade etc.).

Através de comandos SQL os usuários podem montar consultas poderosas sem a necessidade da criação de um programa, ou ainda utilizar comandos SQL embutidos em programas de aplicação que acessem os dados armazenados.

Devido ao fato de possuir várias aplicações, a linguagem SQL oferece suporte a várias funções de um SGBD. Ela consiste de:

- DDL (linguagem de definição de dados), onde os dados a serem armazenados são definidos e estruturados;
- DML (linguagem de manipulação de dados), que permite a inclusão, remoção, seleção ou atualização de dados armazenados no banco de dados;
- Controle de acesso, permitindo proteção dos dados de manipulação não autorizadas;
- Restrições de integridade, que auxiliam no processo de definição da integridade dos dados, protegendo contra corrupções, inconsistências e falhas no sistema de computação.

Toda consulta SQL é baseada no modelo relacional, ou seja, retorna uma tabela como resposta. Para definir uma consulta basta informar o que queremos e não como fazer, pois SQL é uma linguagem não procedural.

O comando SELECT: A estrutura básica de um comando SQL é

```
SELECT <lista de atributos>
FROM <lista de tabelas>
[WHERE <condição>]
```

SELECT. Seleciona as colunas que deverão compor a tabela resultante. Os nomes dos atributos devem ser os mesmos definidos nos bancos de dados. É uma cláusula obrigatória em qualquer consulta.

FROM indica as tabelas do banco de dados que devem ser consultadas. Também é obrigatória em qualquer consulta.

WHERE indica uma condição pela qual os dados serão filtrados. A condição especificada deve retornar verdadeiro ou falso. Para cada linha da tabela, o interpretador SQL verifica se atende a condição especificada nesta cláusula e adiciona a linha na resposta caso seja verdadeira a avaliação. É uma cláusula opcional na consulta.

Observações:

- Caso a cláusula **where** seja omitida, a condição é considerada como verdadeiro.
- A lista de atributos na cláusula **select** pode ser substituída por um asterisco (*) para selecionar todos os atributos de todas as relações da cláusula **from**.
- SQL forma o produto cartesiano das relações chamadas na cláusula **from**, verifica a condição dada na cláusula **where**, e então, projeta o resultado para os atributos da cláusula **select**.
- O resultado de uma consulta SQL é sempre uma tabela.
- Os operadores lógicos **and** e **or** e os operadores de comparação $>$, $>=$, $<$, $<=$, $=$ e \neq podem ser usados na cláusula **where**.

2.3 XML - *Extensible Markup Language*

O objetivo desta seção é apresentar os fundamentos da linguagem XML. Sendo mostrado apenas o suficiente para o entendimento desta tese.

A linguagem XML é muito mais poderosa e maiores detalhes de toda esta teoria podem ser encontrados na literatura, por exemplo em [18].

2.3.1 O que é XML?

XML é um conjunto de regras para a definição semântica de *tags* que quebram um documento em partes e identificam as diferentes partes deste documento.

As *tags* se diferenciam dos dados formados por caracteres (textos não marcados) por estarem contidas em colchetes como “<” e “>” como “< aqui >”. Portanto, um documento é formado por *tags* e dados que combinados formam elementos. O elemento começa com uma *tag* de abertura e termina com uma de fechamento.

XML é uma *meta-markup language*. É uma linguagem em que podemos criar as *tags* que precisarmos, dando a elas o nome que achamos mais conveniente de forma que tenham um significado extra de acordo com o contexto. Estas *tags*

precisam ser organizadas de acordo com certos princípios gerais, mas estes são bem flexíveis.

XML pode ser designado para ser usado na Internet, no entanto tem outros importantes usos:

- Uma forma de armazenamento para processadores de palavras;
- Um formato para a troca de dados entre diferentes programas;
- Um caminho para preservar dados num modo de leitura humana.

2.3.2 HTML × XML

XML não é apenas outra *markup language* como HTML (*Hypertext Markup Language*) que define um número fixo de *tags* que descreve um número fixo de elementos.

XML é muito mais flexível e acessível para os vários usos do que HTML.

HTML é realmente bom somente para descrever *layout*. Usando HTML é possível tornar uma palavra **negrito** ou *itálico*. Em XML, como as marcações são definidas em outros documentos este tipo de formatação pode ser feita de uma maneira mais clara e de fácil manutenção.

Exemplo 2.9 *Um exemplo em HTML:*

Uma música pode ser descrita usando a definição de título, de data, uma lista não ordenada e uma lista de itens. Deste modo nenhuma tag tem realmente relação com música.

```
<dt> Dança Espanhola  
<dd> por Tchaikovsky  
<ul>  
<li> Orquestra: Sinfônica de Berlim  
<li> Regente: Gunther von Clidows  
<li> Duração: 3'54
```


 CD: Jóias da Música

Exemplo 2.10 *O mesmo exemplo em XML:*

Em vez de tags genéricas como <dt> e , este exemplo usa tags com significado como <MÚSICA>, e <TÍTULO>. Que traz vantagens, como a facilidade da leitura do código.

<MÚSICA>

<TÍTULO> Dança Espanhola </TÍTULO>

<COMPOSITOR> Tchaikovsky </COMPOSITOR>

<ORQUESTRA> Orquestra Sinfônica de Berlim </ORQUESTRA>

<REGENTE> Gunther von Clidows </REGENTE>

<DURAÇÃO> 3'54 </DURAÇÃO>

<CD> Jóias da Música </CD>

</MÚSICA>

XML é, em um nível básico, um incrível formato simples de dados. Em um alto nível, XML se auto descreve. Se uma pessoa sem nenhum conhecimento de XML encontrar o código acima escrito em uma folha de papel, esta pessoa será capaz de perceber que se trata da descrição de uma música, cujo título é Dança Espanhola.

2.3.3 Pontos Fortes de XML

- **Inteligência:** a XML é inteligente para qualquer nível de complexidade. A marcação pode ser alterada de uma marcação mais geral como

“<CÃO> Lassie </CÃO>”

para uma mais detalhista, como

“<CÃO> <COLLIE> Lassie </COLLIE> </CÃO>”.

- **Adaptação:** A adaptação de XML é infinita. Marcações personalizadas podem ser criadas para qualquer necessidade. Se for preciso fazer uso de uma marcação que descreva uma música clássica de maneira diferente de uma música folclórica, ela pode ser feita.
- **Manutenção:** XML é de fácil manutenção, pois isola o conteúdo da formatação. O que simplifica o desenvolvimento e a manutenção. Pessoas diferentes com experiências diferentes podem trabalhar independentemente nas informações de um documento e no formato, estilo e estética.
- **Simplicidade:** XML é uma evolução de SGML - *Standard Generalized Markup Language*. O problema com SGML é que esta usa uma forma de gerenciamento profissional da informação. XML representa uma tentativa de simplificar SGML a um nível que esta possa ser usada facilmente. O resultado é uma versão simplificada de SGML que contém todas as partes de SGML que as pessoas estavam habituadas a usar.
- **Transferência de Dados:** As duas principais vantagens da XML sobre as outras linguagens de transferência de dados é que ela é muito expressiva e extensível. Na verdade, é possível dizer qualquer coisa sobre qualquer assunto. Ela também possui uma sintaxe consistente e de fácil compreensão que a torna simples de ser analisada. Portanto:
 - A XML pode capturar o tipo de informação que é transferida entre aplicativos.
 - Os documentos XML podem ser personalizados para se ajustarem a necessidades muito específicas.
 - Os analisadores XML e outras ferramentas genéricas já estão disponíveis.

2.4 Bancos de Dados Nebulosos

Nesta seção, são apresentados alguns conceitos sobre a área de incerteza em bancos de dados, mostrando algumas das diferenças entre estes e os bancos de dados tradicionais, maiores detalhes em [9].

A introdução da lógica nebulosa nos bancos de dados aconteceu com o intuito de tornar possível a manipulação de informações incompletas, incertas ou imprecisas.

A incerteza em bancos de dados pode aparecer por diversas razões:

- **Imprecisão dos dados reais:** *Por exemplo, quando tratamos de uma medição, às vezes, esta seria melhor descrita com termos lingüísticos tais como “em torno de”.*
- **Resultar de julgamentos:** *Por exemplo, quando deseja-se saber se uma cidade possui uma qualidade de vida boa, é preciso julgar a qualidade das escolas, o saneamento básico, os meios de transporte, etc.*
- **A informação que o usuário está interessado é imprecisa:** *Por exemplo, um usuário gostaria de obter uma lista de escolas que possuam um bom ensino e com um preço razoável.*

A lógica nebulosa vem sendo utilizada para estender os sistemas de bancos de dados em duas áreas:

- Armazenamento e recuperação de **informações imprecisas por natureza.**
- Processamento de **consultas que são imprecisas.**

2.4.1 Informações Imprecisas

As informações imprecisas aparecem nos bancos de dados nebulosos de duas maneiras:

- Valores imprecisos de um atributo em uma tupla.
- Inclusão parcial de uma tupla em uma relação.

Um exemplo será apresentado a seguir.

Exemplo 2.11 *A relação Animais em Extinção, apresentada na Tabela 2.2, possui quatro atributos: Animal, Número de Exemplares, Região e Peso da Tupla.*

Tabela 2.2: Animais em Extinção

	Atributo			
	nebuloso			
	↓			
	Animal	Nº de Exemplares	Região	Peso da
		(em 2002)		Tupla
	Ararinha Azul	0	Norte da Bahia	1
Tupla nebulosa ⇒	Peixe Boi	aproximadamente 1500	Amazônia	0.6
	Cervo do Pantanal	aproximadamente 600	Pantanal	0.8

O atributo Número de exemplares é um atributo nebuloso porque nem sempre é possível saber exatamente quantos animais habitam uma região, assim este atributo aceita termos lingüísticos tais como aproximadamente.

Nesta relação, as tuplas possuem um grau de pertinência, por exemplo, o Peixe Boi não está em extinção na região do Pantanal, mas possui um número de exemplares bem reduzido o que faz com que ele pertença a tabela dos animais em extinção com um grau de pertinência igual a 0.6.

Valores imprecisos de um atributo em uma tupla podem ser representados por duas maneiras diferentes. Através de relações nebulosas baseadas em Similaridades, ver [3], ou em Possibilidades, ver [4], ambas definidas por Zadeh e apresentadas a seguir.

2.4.1.1 Relações Nebulosas Baseadas em Similaridade

Nas relações nebulosas baseadas em similaridade a imprecisão dos valores dos atributos está primariamente em seus significados.

Esta teoria será explicada através de um exemplo.

Exemplo 2.12 *Suponhamos que uma empresa está interessada em armazenar informações a respeito de profissionais liberais que já prestaram serviços a ela. A Tabela 2.3 armazena o primeiro nome, o último nome, telefone e a especialidade desses profissionais.*

Tabela 2.3: Relação de Profissionais Liberais

Nome	Último_Nome	Telefone	Especialidade
Pedro	Rocha	(021) 2234-2125	IA
Júlio	Cruz	(032) 3261-0023	Sistemas Especialistas
Maria	Cimino	(021) 6223-7877	Estatística
Ana	Bento	(021) 2635-0910	Robótica

A matriz de similaridades para o atributo Especialidade, cujo domínio é {Robótica, Sistemas Especialistas, Inteligência Artificial, Estatística}, apresenta o grau de sobreposição das diferentes áreas de especialidades. Ou seja, o grau em que uma área pode ser considerada similar a outra de acordo com o seu significado semântico.

Uma matriz de similaridade para o atributo Especialidade está apresentada na Tabela 2.4.

Tabela 2.4: Matriz de Similaridade para o Atributo Especialidade

	Robótica	Sistemas Especialistas	IA	Estatística
Robótica	1,0	0,6	0,6	0,2
Sistemas Especialistas	0,6	1,0	0,9	0,2
IA	0,6	0,9	1,0	0,2
Estatística	0,2	0,2	0,2	1,0

De acordo com Zadeh, uma relação de similaridade não pode ser construída livremente, ela precisa respeitar as três propriedades apresentadas a seguir.

Definição 2.31 Relação de similaridade *Uma relação de similaridade s_r para um domínio D é uma relação nebulosa que mapeia pares de valores do domínio D no intervalo unitário, isto é, $s_r : D \times D \rightarrow [0, 1]$, com as seguintes três propriedades:*

1. *Reflexiva:* $s_r(x, x) = 1$
2. *Simétrica:* $s_r(x, y) = s_r(y, x)$
3. *Transitiva:* $s_r(x, z) \geq \max_{y \in D} (\min(s_r(x, y), s_r(y, z)))$
sendo $x, y, z \in D$.

2.4.1.2 Relações Nebulosas Baseadas em Possibilidade

Uma aproximação baseada em possibilidade está diretamente relacionada com a teoria de possibilidade. As aproximações possibilidade/necessidade são mais gerais que a aproximação por similaridade. Uma vez que são capazes de manipular todos os tipos de informação, ao contrário da aproximação baseada em similaridade que somente se aplicam à domínios com valores discretos.

A relação nebulosa baseada em possibilidade generaliza uma relação permitindo que o valor de um atributo A seja uma distribuição de possibilidade $\Pi_{A(t)}$ sobre o domínio deste atributo.

O interesse por aproximações baseadas em possibilidade é causado por sua habilidade de representar, de modo unificado, valores precisos e valores NULL, tão bem quanto valores nebulosos.

É possível notar que, uma vez que os dados são imprecisos, então o resultado de uma consulta será também impreciso. Na teoria, ambas as medidas de possibilidade e necessidade podem ser usadas no processamento de consultas em banco de dados.

Para caracterizar a imprecisão da resposta de uma consulta, usamos as medidas de possibilidade e necessidade.

Exemplo 2.13 *Se for dada a distribuição de possibilidade da idade de um suspeito e o universo de discurso Ω da idade das pessoas, e deseja-se encontrar suspeitos jovens, é possível calcular o grau de possibilidade e de necessidade como a seguir:*

$$Poss(jovem|\Pi_{idade}) = \sup_{x_i \in \Omega} [\min(\mu_{jovem}(x_i), \Pi_{idade}(x_i))] \quad (2.4.57)$$

$$Nec(jovem|\Pi_{idade}) = \inf_{x_i \in \Omega} [\max(\mu_{jovem}(x_i), 1 - \Pi_{idade}(x_i))] \quad (2.4.58)$$

Seja o universo de discurso da idade das pessoas neste exemplo dado por:

$$\Omega = \{10, 15, 20, 25, 30, 35, 40, 45, 50\}$$

E a distribuição de possibilidade da idade do suspeito S é:

$$\Pi_{Idade}(S) = \left\{ \frac{0.2}{15}, \frac{0.5}{20}, \frac{1}{25}, \frac{0.8}{30} \right\}$$

Seja a função de inclusão da etiqueta lingüística *jovem* definida como o seguinte conjunto nebuloso discreto:

$$jovem = \left\{ \frac{1}{10}, \frac{1}{15}, \frac{1}{20}, \frac{0.8}{25}, \frac{0.4}{30}, \frac{0.2}{35} \right\}$$

Usando a Equação 2.4.57, pode-se calcular o grau de possibilidade para que o suspeito S satisfaça a condição de “suspeito jovem”:

$$Poss(jovem|\Pi_{idade}) = \sup \{ \min(0.2, 1), \min(0.5, 1), \min(1, 0.8), \min(0.8, 0.4) \}$$

$$Poss(jovem|\Pi_{idade}) = \sup \{ 0.2, 0.5, 0.8, 0.4 \}$$

$$Poss(jovem|\Pi_{idade}) = 0.8$$

Para calcular a medida de necessidade, primeiro é preciso calcular o complemento da distribuição de possibilidade da idade do suspeito S :

$$1 - \Pi_{Idade}(S) = \left\{ \frac{1}{10}, \frac{0.8}{15}, \frac{0.5}{20}, \frac{0}{25}, \frac{0.2}{30}, \frac{1}{35}, \frac{1}{40}, \frac{1}{45}, \frac{1}{50} \right\}$$

A medida de necessidade pode ser obtida usando a Equação 2.4.58:

$$Nec(jovem|\Pi_{idade}) = \inf \{ \max(1, 1), \max(1, 0.8), \max(1, 0.5), \max(0.8, 0), \\ \max(0.4, 0.2), \max(0.2, 1), \max(0, 1), \max(0, 1), \max(0, 1) \}$$

$$Nec(jovem|\Pi_{idade}) = \inf \{ 1, 1, 1, 0.8, 0.4, 1, 1, 1, 1 \}$$

$$Nec(jovem|\Pi_{idade}) = 0.4$$

Logo, pode-se concluir que a possibilidade do suspeito S ser jovem é 0.8, enquanto que a necessidade dele ser jovem é 0.4.

2.4.2 Consultas Imprecisas

As consultas imprecisas aparecem nos bancos de dados nebulosos de três maneiras diferentes:

- **Condições Imprecisas:** *Por exemplo, deseja-se encontrar os brasileiros que possuem renda baixa.*
- **Operadores Imprecisos:** *Por exemplo, deseja-se encontrar os brasileiros que possuem renda quase igual a R\$1.000,00.*
- **Quantidades Imprecisas:** *Por exemplo, deseja-se encontrar os países cuja maioria dos aposentados possui casa própria.*

Para bancos de dados relacionais nebulosos, os conceitos da linguagem SQL são estendidos e esta passa a ser chamada de SQL Nebuloso.

Exemplo 2.14 *A relação Alunos, apresentada na Tabela 2.5, é composta de 5 atributos: Nome, Idade, Participação em Sala, Nota_Matemática e Nota_Português.*

Tabela 2.5: Relação dos Alunos de uma Escola

Nome	Idade	Participação em Sala	Nota_Matemática	Nota_Português
Juan Dias	9	boa	5	9
Cezar Silva	8	ruim	6	7
Maria Lima	8	excelente	10	8
Ana Goes	7	ruim	3.5	7
Marília Tavares	9	regular	5.5	4

O atributo Participação em Sala é um atributo nebuloso e possui domínio discreto D , dado por:

$$D = \{ruim, regular, boa, excelente\}$$

Tabela 2.6: Matriz de Similaridade para o Atributo Participação em Sala

	ruim	regular	boa	excelente
ruim	1,0	0,8	0,5	0,1
regular	0,8	1,0	0,7	0,5
boa	0,5	0,7	1,0	0,8
excelente	0,1	0,5	0,8	1,0

Ao domínio D está associada uma matriz de similaridade apresentada na Tabela 2.6.

Supõe-se a seguinte consulta: “Recuperar da relação *Alunos*, o nome e a idade dos alunos cuja participação em sala de aula é **boa** com grau pelo menos igual a **0.7**”.

Em SQL nebuloso, esta consulta poderia ser representada da seguinte forma:

```

SELECT Nome, Idade
FROM Alunos
WHERE Participação em Sala = 'boa' (0.7)

```

Neste exemplo, procura-se alunos cujo grau de similaridade entre sua participação em sala é igual a boa com grau pelo menos igual a 0.7. Logo, o resultado desta consulta, apresentado na Tabela 2.7, conterà alunos cuja participação em sala é regular, boa e excelente de acordo com a matriz de similaridade apresentada.

Tabela 2.7: Resultado da consulta

Nome	Idade
Juan Dias	9
Maria Lima	8
Marília Tavares	9

Exemplo 2.15 *Seja a seguinte consulta sobre a relação de alunos de uma escola apresentada em Tabela 2.5: “Recuperar da relação Alunos, o nome dos dois primeiros alunos cuja participação em sala de aula é **ruim** com grau pelo menos igual a **0.8**”*

```
SELECT 2 Nome
FROM Alunos
WHERE Participação em sala = 'ruim' (0.8)
```

Esta consulta busca as duas respostas que melhor respondem a ela, ou seja, de todos os alunos cuja participação em sala se assemelha a ruim a consulta retornará apenas os dois alunos que possuírem os maiores valores da similaridade entre sua participação em sala e ruim. E ainda, sua participação em sala deve se assemelhar com ruim com grau, no mínimo, igual a 0.8.

O processo de consultas em SQL Nebuloso é baseado em uma ou mais consultas regulares SQL.

Exemplo 2.16 *Supõe-se que a relação Países seja composta de 4 atributos: Nome, Capital, População e PIB (Produto interno bruto). Sendo que os atributos População e PIB são atributos nebulosos e possuem etiquetas lingüísticas associadas a seus domínios, como apresentado nas Figuras 2.35 e 2.36.*

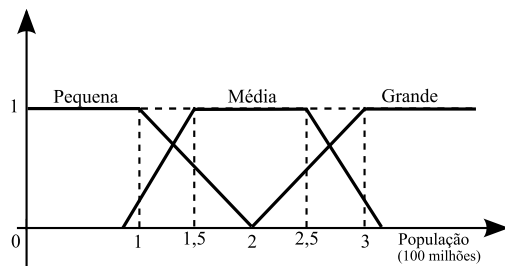


Figura 2.35: Definição de etiquetas sobre o atributo **População**

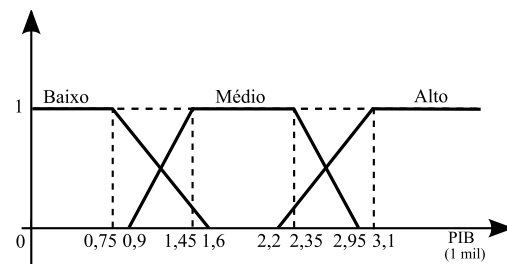


Figura 2.36: Definição de etiquetas sobre o atributo **PIB**

É possível fazer a seguinte consulta: “Recuperar da relação Países o nome e a capital das nações que possuem PIB **baixo** com grau, no mínimo, igual a (0,7) e População **grande** com grau, no mínimo, igual a (0,7).”

```
SELECT Nome, Capital
FROM Países
WHERE População = Grande (0,7) AND PIB = Baixo (0,7)
```

Esta consulta pode ser transformada em uma consulta SQL tradicional encontrando o 0,7-corte da População grande e 0,7-corte do PIB baixo. Estes α -cortes estão apresentados nas Figuras 2.37 e 2.38.

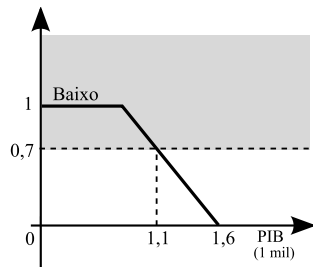


Figura 2.37: 0,7-corte PIB **baixo**

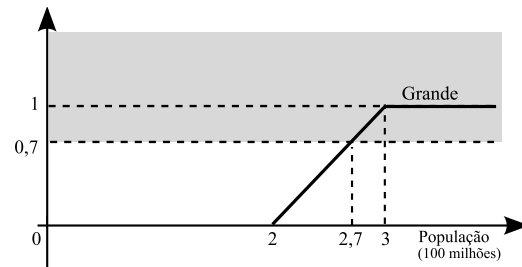


Figura 2.38: 0,7-corte População **grande**

Baseados nas funções de inclusão, os dois 0,7-cortes obtidos são:

$$População_{Grande_{0,7}}(p) = \{p \mid p \geq 270.000.000\}$$

$$PIB_{Baixo_{0,7}}(p) = \{p \mid p \leq 1.100\}$$

Assim, a consulta em SQL nebuloso, apresentada anteriormente nada mais é do que a seguinte consulta em SQL tradicional:

```
SELECT País
FROM Nação
WHERE População ≥ 270.000.000 AND PIB ≤ 1.100
```

2.4.2.1 O Problema da Redundância de Dados

Em bancos de dados nebulosos duas tuplas podem ser parcialmente iguais e parcialmente diferentes ao mesmo tempo, no caso de uma relação possuir todos os seus atributos nebulosos.

O problema é: “Quão similar duas tuplas nebulosas precisam ser para serem consideradas idênticas?”.

Para responder esta questão primeiro é necessário definir igualdade para tuplas nebulosas.

Definição 2.32 Igualdade entre Tuplas *Sejam t_1 e t_2 duas tuplas nebulosas na relação com atributos $a_1, a_2, a_3, \dots, a_k$. O grau de igualdade entre as duas tuplas, é dado por*

$$Equal(t_1, t_2) = \bigotimes_{i=1}^k Eq(a_i(t_1), a_i(t_2))$$

onde $Eq(a_i(t_1), a_i(t_2))$ denota uma medida de similaridade ou de possibilidade, ver [9], entre os i -ésimos atributos das tuplas t_1 e t_2 , e \otimes denota o operador min.

É preciso escolher um limiar para o teste de redundância que é chamado de limiar de redundância.

Definição 2.33 Tuplas Redundantes *Uma tupla nebulosa t_1 é redundante na relação R se existe outra tupla t_2 tal que*

$$Equal(t_1, t_2) \geq \alpha$$

onde α é o limiar de redundância.

Depois de identificadas as tuplas redundantes, apenas uma deve permanecer na relação, sendo aquela que possuir o maior grau de inclusão.

Neste trabalho o problema das tuplas redundantes é eliminado permitindo que as chaves primárias, de todas as tabelas, admitam apenas atributos precisos (*crisp*). O que torna todas as tuplas distintas em uma tabela e elimina o problema da redundância.

Capítulo 3

Histórico

Neste capítulo, uma revisão histórica dos principais sistemas de gerenciamento de bancos de dados relacionais nebulosos é discutida. Além disso, são apresentados os trabalhos de Turowski e Weng [20] e Gaurav e Alhajj [21] que usam o formato XML para tratar informações nebulosas.

O problema da representação e do tratamento de incerteza em bancos de dados vem sendo amplamente estudado e é possível encontrar inúmeras referências bibliográficas tais como Tashiro et al. [22], Yazici et al. [23], Chaudhry et al. [24], Kacprzyk e Zadrozny [25], Yang et al. [26], Yager [27], Raschia e Mouaddib [28] entre outros.

A primeira intensão de representar informação imprecisa em uma base de dados foi apresentada por Codd em [29] e ampliada em [30] e [31]. No entanto, seu modelo não faz uso da teoria dos conjuntos nebulosos. Um valor NULL foi introduzido, significando que quando um atributo assume NULL seu valor pode ser qualquer um pertencente ao domínio. Qualquer comparação com um valor NULL, origina um resultado que não é nem verdadeiro nem falso.

Nas próximas seções são apresentados alguns modelos de bancos de dados que fazem uso da lógica nebulosa para tratar as incertezas que foram utilizados como base para a elaboração desta tese.

3.1 Modelo de Buckles-Petry

O modelo de Buckles-Petry, apresentado em [32], [33] e [34], faz uso da medida de similaridade definida por Zadeh em [3].

Buckles e Petry definiram uma relação nebulosa da seguinte maneira:

Definição 3.1 Relação Nebulosa *Uma relação nebulosa, R , é um subconjunto do conjunto do produto cartesiano $2^{D_1} \times 2^{D_2} \times \dots \times 2^{D_m}$, onde 2^{D_i} denota qualquer elemento não nulo do conjunto potência (conjunto das partes) $P(D_i)$ do domínio D_i .*

A inclusão em uma relação específica R , é determinada pela semântica da relação. Por exemplo, se D_1 é o conjunto das maiores cidades, D_2 é o conjunto dos países e R é definida em $2^{D_1} \times 2^{D_2}$ como sendo a relação dos países e suas maiores cidades, então $(Paris, Itália) \in 2^{D_1} \times 2^{D_2}$, mas não pertencem a R .

Qualquer membro da relação é chamado de tupla.

Definição 3.2 Tupla Nebulosa *Uma tupla nebulosa, t , é qualquer membro de R e $2^{D_1} \times 2^{D_2} \times \dots \times 2^{D_m}$.*

Uma tupla arbitrária é da forma $t_i = (d_{i_1}, d_{i_2}, \dots, d_{i_m})$ onde $d_{i_j} \subseteq D_j$.

Os tipos de dados que foram inicialmente suportados pelo modelo são:

- Conjunto finito de escalares. Ex. $DE = \{loiro, ruivo, castanho, negro\}$
- Conjunto finito de números. Ex. $DN = \{15, 16, 17\}$

A forma de representar e manipular a informação era feita através das *Relações de Similaridade* sobre domínios escalares ou numéricos. Limiares de aceitação com relação a similaridades eram usados quando as consultas são efetuadas.

Posteriormente em [33] o modelo passou a suportar também um terceiro tipo de dado.

- Conjunto de números nebulosos (etiquetas associadas a uma função de inclusão). Ex. $DD = \{muito alto, alto, médio, baixo\}$

O modelo de Buckles-Petry levanta a questão da redundância entre tuplas nebulosas tratando esta redundância através do uso de limiares de aceitação para o grau de similaridades entre os atributos que farão as junções das tabelas a serem consultadas.

Definição 3.3 Tuplas Redundantes *Duas tuplas distintas $t_i = (d_{i_1}, d_{i_2}, \dots, d_{i_m})$ e $t_k = (d_{k_1}, d_{k_2}, \dots, d_{k_m})$, são redundantes se*

$$LEVEL(D_j) \leq \min_{x,y \in d_{i_j} \cup d_{k_j}} [s(x, y)]$$

para $j = 1, 2, \dots, m$ e $LEVEL(D_j)$ dado a priori.

Em [35] foi apresentado um Cálculo Relacional para este modelo.

Assim como o modelo proposto por Buckles e Petry, *Aliança* também faz uso das relações de similaridade entre elementos de um domínio, mas trata o problema das tuplas redundantes considerando as chaves primárias de suas tabelas como nítidas da mesma forma que os bancos de dados clássicos.

3.2 Modelo de Prade-Testemale

O modelo de Prade-Testemale, que foi publicado em [36], aceita que valores nebulosos sejam associados aos atributos.

Em seus atributos, tanto os valores precisos quanto os parciais (imprecisos, desconhecidos) são representados pela distribuição de possibilidade de Zadeh [4].

3.2.1 Representação de dados por Distribuição de Possibilidade

Seja A um atributo e D seu domínio. Os valores válidos relativos ao valor do atributo A sobre o objeto x serão representados por uma distribuição de possibilidade $\Pi_{A(x)}$ sobre $D \cup \{e\}$, onde e é um elemento especial que representa o caso em que A não se aplica a x . Em outras palavras, $\Pi_{A(x)}$ é uma função que vai de $D \cup \{e\}$ ao intervalo $[0, 1]$.

Seja um exemplo onde a representação do conhecimento a respeito da idade do carro de um usuário chamado Paulo é dado. Algumas situações são apresentadas a seguir, onde a distribuição de possibilidades da idade do carro de Paulo $\Pi_{idade-do-carro}(Paulo)$ é representada apenas por Π .

1. Não se sabe se Paulo possui ou não um carro. Se ele possuir, a idade de seu carro pode ser:

$$\Pi(d) = 1; \quad \forall d \in D \cup \{e\}$$

2. É completamente possível que Paulo possua um carro, mas existe a possibilidade igual a $\lambda > 0$ que seu carro tenha mais de 5 anos:

$$\Pi(e) = 1; \quad \Pi(d) = \begin{cases} \lambda & \forall d \geq 5 \\ 0 & \forall d < 5 \end{cases}$$

3. É certo que Paulo não tem carro.

$$\Pi(e) = 1$$

O que corresponde que o valor null não se aplica.

4. É completamente possível que Paulo tenha um carro, mas existe a possibilidade não nula λ dele não possuir carro:

$$\Pi(e) = \lambda, \lambda > 0; \quad \Pi(d) = \mu_{novo}(d), \forall d \in D$$

onde μ_{novo} é uma função de inclusão que representa o predicado vago “novo”.

5. É certo que Paulo possui um carro, mas não há nenhuma informação a respeito de sua idade:

$$\Pi(e) = 0; \quad \Pi(d) = 1 \forall d \in D$$

O valor NULL corresponde a “desconhecido”.

6. Está certo que Paulo possui um carro, e existe a informação não nebulosa parcial que a idade de seu carro está entre 2 e 4 anos:

$$\Pi(e) = 0; \quad \Pi(d) = \begin{cases} 1 & \forall d \in [2, 4] \subseteq D \\ 0 & \text{caso contrário} \end{cases}$$

7. É certo que Paulo possui um carro, e existe a informação nebulosa de que ele é novo:

$$\Pi(e) = 0; \quad \Pi(d) = \mu_{\text{novo}}(d) \quad \forall d \in D$$

8. É certo que Paulo possui um carro, e que este tem 2 anos:

$$\Pi(e) = 0; \quad \Pi(2) = 1$$

sendo Π igual a zero para qualquer outro valor em D .

Assim, em cada caso a distribuição de possibilidade é normalizada em $D \cup \{e\}$; que é natural, uma vez que $D \cup \{e\}$ descreve todas as possíveis alternativas.

Consultas vagas, cujos conteúdos são também representados por distribuições de possibilidade, podem ser feitas.

As operações básicas da álgebra relacional, união, interseção, produto cartesiano, projeção, e seleção são estendidas para tratar a informação parcial ou consultas vagas.

As principais características de uma linguagem de consulta baseada na álgebra relacional estendida é apresentada.

A manipulação que o modelo de Prade-Testemale realiza sobre a informação representada mediante distribuição de possibilidade também é objeto de estudo desta tese e foi incorporada ao modelo *Aliança*.

3.3 Modelo de Umano-Fukami

Um dos primeiros modelos de bancos de dados relacionais nebulosos foi apresentado por Umano e Fukami em [37] e [38].

No modelo de Umano-Fukami são utilizadas distribuições de possibilidade para modelar o conhecimento sobre a informação de forma similar ao que foi feito no modelo de Prade-Testemale.

Seja D o universo de discurso de $A(x)$ e $\Pi_{A(x)}(d)$ representa a possibilidade que $A(x)$ tome o valor u em D , então para os valores “desconhecidos e aplicáveis”, que denomina *unknown*, a mesma representação de Prade-Testemale é aplicada:

$$\text{Unknown: } \Pi_{A(x)}(d) = 1; \quad \forall d \in D \quad (3.3.1)$$

Para os valores “não aplicáveis” existe um caso especial de distribuição de possibilidade denominado “indefinido” *undefined* e se representa como:

$$\text{Undefined: } \Pi_{A(x)}(d) = 0; \quad \forall d \in D \quad (3.3.2)$$

Para representar a situação em que não se conhece nem se uma “ausência” de informação é “aplicável” ou não, empregam um valor especial denominado *NULL*:

$$\text{NULL: } \left\{ \frac{1}{\text{Unknown}}, \frac{1}{\text{Undefined}} \right\} \quad (3.3.3)$$

Para o restante dos casos de informações imprecisas o modelo de Umano-Fukami adota uma representação similar a do modelo Prade-Testemale.

Umano e Fukami estenderam seu modelo de base de dados nebulosos, ao que chamaram de *possibility-distribution-fuzzy-relational* (modelo relacional nebuloso de distribuições de possibilidade), para representar e manipular dados ambíguos. Este modelo permite armazenar distribuições de possibilidade nos valores dos atributos. Além do que, cada tupla de uma relação tem associada a ela uma distribuição de possibilidade no intervalo $[0,1]$, de forma que indica o grau de pertinência dessa tupla na relação. Ou seja, eles definiram uma relação difusa R , de m atributos, com uma função de pertinência μ_R :

$$\mu_R : P(U_1) \times P(U_2) \times \dots \times P(U_m) \rightarrow P([0, 1])$$

onde o símbolo \times denota o Produto Cartesiano, $P(U_j)$, com $j = 1, 2, \dots, m$, é a coleção de todas as distribuições de possibilidade no universo de discurso U , do j –ésimo atributo de R .

A função μ_R associa a cada tupla da relação R um valor de $P([0, 1])$, que é o conjunto de todas as distribuições de possibilidade no intervalo $[0, 1]$ e que será vista como um grau de pertinência dessa tupla em R .

Um efeito prático é como se a cada relação R fosse adicionado um atributo (μ_R) com o domínio em $P([0, 1])$. Naturalmente, pode ser armazenado um número entre 0 e 1 (por exemplo 0.8) como valor desse atributo, sendo este visto como uma distribuição de possibilidade ($\frac{1}{0.8}$, no exemplo). Igualmente é possível armazenar valores “crisp” em qualquer outro atributo.

Desta forma o modelo possibility-distribution-fuzzy-relational possui a vantagem de permitir dois tipos de ambigüidade:

1. **Ambigüidades nos valores dos atributos:** Permite que os valores que armazenam as relações não sejam valores exatos, podendo ser distribuições de possibilidade.
2. **Ambigüidade na associação de valores:** Associa a cada tupla um grau de pertinência à relação, que pode ser visto como um grau de verdade para indicar em que medida estão relacionados os atributos de cada tupla (através da relação).

Neste contexto são definidos os operadores básicos da Álgebra Relacional entre relações deste tipo: União, Diferença, Produto Cartesiano, Projeção e Seleção. Também define outros operadores não básicos: Interseção, Junção e Divisão.

O modelo *Aliança* incorporou as distribuições de possibilidades nomeadas por Prade e Testemale como *Undefined*, *Unknown* e *NULL*, mas não faz usos de graus de inclusão para as tuplas em suas relações.

3.4 Modelo de Zemankova-Kaendel

O objetivo do modelo de Zemankova-Kaendel, publicado em [39], é propor uma estrutura de trabalho para representar dados imprecisos e manipular incerteza ou imprecisão na linguagem de consulta.

O modelo se baseia em:

- Representação de informação imprecisa;
- Aplicação das medidas de possibilidade e certeza;
- Aproximações lingüísticas de termos nebulosos na linguagem de consulta;
- Desenvolvimento de operadores nebulosos de comparação;
- Processamento de consultas com conectores nebulosos e qualificadores verdade;
- Manipulação de valores NULL usando o conceito de valor possibilístico esperado;
- Modificação de definições de termos nebulosos de acordo com cada usuário.

Uma vez que as distribuições de possibilidades usadas devem estar normalizadas para que as funções de inclusão de conjuntos nebulosos estejam adequadas para a representação de dados uma medida de “certeza” foi introduzida.

Definição 3.4 Medida de Certeza *Seja F um subconjunto nebuloso de U caracterizado por sua função de inclusão μ_F , e Π_X uma distribuição de possibilidade associada com a variável X , sendo que toma seus valores em U . A medida de certeza de F , $c(F)$, é definida por*

$$\begin{aligned} Cert\{X \acute{e} F\} &\equiv c(F) \\ Cert\{X \acute{e} F\} &\equiv \max(0, \inf_{u \in U} \{\mu_F(u), \Pi_X(u)\}) \end{aligned} \tag{3.4.4}$$

3.4.1 Arquitetura

A organização do modelo de banco de dados nebulosos Zemankova-Kaendel pode ser dividida em:

- **Base de dados de valores (VDB)**: onde se organizam os dados de forma similar aos outros modelos possibilísticos,

- **Base de dados explicativa (EDB):** onde se armazenam as definições para os subconjuntos nebulosos e relações nebulosas,
- **Regras de tradução:** conjunto de regras que é empregado para a manipulação de adjetivos, etc.

3.4.2 Tipos de Dados

Os domínios podem ser dos seguintes tipos:

- Conjunto escalar discreto (ex. CORES = {vermelho, branco, azul });
- Conjunto numérico discreto ou contínuo;
- Intervalo unitário $[0, 1]$.

Os valores dos atributos podem ser:

1. Escalares simples ou números;
2. Uma seqüência de escalares ou números;
3. Distribuição de possibilidade de escalar ou domínios de valores numéricos;
4. Um número real de um intervalo unitário $[0, 1]$;
5. Um valor nulo.

O modelo de Zemankova-Kaendel além de fazer uso da Medida de Similaridade, apresentada por Zadeh em [3], também utiliza a Medida de Certeza apresentada pela Equação 3.4.4 e uma medida chamada de Relação de Proximidade definida a seguir:

Definição 3.5 Relação de Proximidade *Seja D_i um domínio numérico, e $x, y, z \in D_i$. Então $p(x, y) \in [0, 1]$ é uma relação de proximidade que é reflexiva e simétrica, com transitividade da forma*

$$p(x, z) \geq \max_{y \in D_i} \{p(x, y), p(y, z)\} \quad (3.4.5)$$

A forma da relação de proximidade geralmente usada é

$$p(x, y) = e^{-\beta|x-y|}, \text{ onde } \beta > 0 \quad (3.4.6)$$

Esta forma especifica graus iguais de proximidade para pontos igualmente distantes. Por esta razão, a forma da relação de proximidade apresentada na equação 3.4.6 é referida como **proximidade absoluta** neste modelo.

3.5 Modelo GEFRED

O modelo GEFRED (A GEneralized model Fuzzy RElational Databases), criado por Medina em [1] e aperfeiçoado por Galindo em [2], é definido como sendo uma fusão entre as diferentes tendências que existiam para abordar o problema da representação e tratamento da informação nebulosa mediante as Bases de Dados Nebulosas. Por esta razão, será discutido mais detalhadamente.

3.5.1 Arquitetura do Banco de Dados Relacional Difuso: O Servidor FSQ (Fuzzy SQL)

A arquitetura do BDRD (Banco de Dados Relacional Difuso) mostra seus elementos básicos e como se relacionam uns com os outros.

Implementação da BDRD: FIRST

O FIRST (**F**uzzy **I**nterface for **R**elational **S**ys**T**ems) é uma interface nebulosa para sistemas relacionais que utilizam o GEFRED como base teórica.

A Figura 3.1 mostra o Esquema Geral do FIRST. Como a idéia de partida é construí-lo sobre um banco de dados convencional, todas as etapas de desenvolvimentos tomam o gerenciador como o elemento principal, e é ele que vai controlar todos os pedidos.

Descrição dos módulos:

- **SGBDR** (Sistema de Gerenciamento de Banco de Dados Relacionais, *Relational DataBase Management System, RDBMS*): Todas as operações feitas

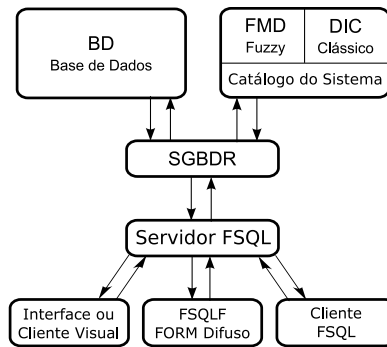


Figura 3.1: Esquema Geral do FIRST

para a extensão nebulosa são transformadas em pedidos do SGBDR anfitrião. Geralmente, todos os pedidos ao sistema se resolvem mediante seqüências SQL clássicas.

- **BD** (Base de Dados): Armazena, em formato relacional toda a informação que seja de interesse, similar a qualquer outra base de dados. A única diferença é que esta base de dados permitirá o armazenamento de informação nebulosa em suas tabelas.
- **FMB** (*Fuzzy Metaknowledge Base*, Base de Metaconhecimento Difuso): O “dicionário” ou “catálogo do sistema” de um SGBDR representa aquela parte do sistema que armazena informação sobre os dados reconhecidos na base de dados, assim como outros tipos de informações: Usuários, permissões, acessos, dados de controle... Dentro deste catálogo está incluída a FMB, que estende esta parte do sistema a fim de que possa recolher aquela informação necessária relacionada com a natureza imprecisa da nova coleção de dados a processar.
- **Servidor FSQL**: Seu objetivo é captar as sentenças em linguagem FSQL e traduzí-las a uma linguagem que possa ser entendida pelo SGBDR, ou seja, a linguagem SQL. Para efetuar essa tradução o SGBDR utilizará a informação armazenada na FMB.

- **Cliente FSQL:** Se trata de um programa que faz a interface entre o usuário (ou outro programa) e o Servidor FSQL.

3.5.2 Representação do Conhecimento Impreciso

Para os diferentes tipos de dados que constituem a definição de *domínio difuso generalizado*, apresentados na Tabela 3.1, são usados os seguintes critérios de representação:

Tabela 3.1: Tipos de Dados Representados pelo GEFRED

1.	Um escalar simples (Ex. Tamanho = Grande; representado mediante a distribuição de possibilidade $\frac{1}{Grande}$)
2.	Um número simples (Ex. Idade = 28; representado mediante a distribuição de possibilidade $\frac{1}{28}$)
3.	Um conjunto de possibilidades associadas a escalares (Ex. Aptidão = {Má, Boa}, se expressa $\{ \frac{1}{Má}, \frac{1}{Boa} \}$)
4.	Um conjunto de possibilidades associadas a números (Ex. Idade = {20, 21}, se expressa $\{ \frac{1}{20}, \frac{1}{21} \}$)
5.	Uma distribuição de possibilidades no domínio dos escalares (Ex. Aptidão = $\{ \frac{0.8}{Má}, \frac{1}{Boa} \}$)
6.	Uma distribuição de possibilidades no domínio dos números (Ex. Idade = $\{ \frac{0.4}{23}, \frac{1}{24}, \frac{0.8}{25} \}$)
7.	Um número Real $\in [0, 1]$ representando graus de satisfação (Ex. Qualidade = 0.8)
8.	Um valor desconhecido Unknown dado pela distribuição de possibilidade Unknown = $\{ \frac{1}{u} : u \in U \}$; sendo U o domínio considerado
9.	Um valor indefinido Undefined dado pela distribuição de possibilidade Undefined = $\{ \frac{0}{u} : u \in U \}$; sendo U o domínio considerado
10.	Um valor nulo NULL dado pela expressão NULL = $\{ \frac{1}{Unknown}, \frac{1}{Undefined} \}$

1. **Dados Precisos** (*crisp*, clássicos): É empregada a representação do SGBDR.
2. **Dados Imprecisos** (*fuzzy*, difusos): Os dados de natureza nebulosa suportados pelo GEFRED podem ser divididos em dois grupos com representações distintas para cada um deles:
 - *DADOS IMPRECISOS SOBRE REFERENCIAL ORDENADO* Este grupo de dados contém distribuições de possibilidade sobre domínios contínuos ou discretos sobre os quais existe uma relação de ordem. Foram adotadas as seguintes representações para este tipo de dados:
 - *Distribuição de Possibilidade Trapezoidal*
 - *Etiqueta Lingüística*
 - *Valores Aproximados*
 - *Intervalos de possibilidade*
 - *DADOS COM ANALOGIA SOBRE REFERENCIAL NÃO ORDENADO* Este grupo de dados está construído sobre domínios subjacentes discretos não ordenados onde se encontram definidas “*relações de semelhança*” ou similaridade entre os valores que os constituem. Os diferentes tipos que podem ser representados dentro deste grupo são:
 - *Escalares Simples* ($\{(1, d)\}$)
 - *Distribuição de Possibilidade sobre Escalares* ($\{(p_1, d_1), \dots, (p_n, d_n)\}$)

Por outro lado, existem outros 3 valores especiais que podem ser armazenados em qualquer um dos tipos *imprecisos* que acabaram de ser descritos. Esses 3 valores foram apresentados por Umano e Fukami [38].

- **UNKNOWN** (Desconhecido, mas aplicável)
- **UNDEFINED** (Não aplicável)
- **NULL** (Ignorância Absoluta)

3.5.3 Comparadores Nebulosos Generalizados

Os comparadores nebulosos generalizados permitem modelar uma ampla variedade de modalidades de comparação.

- Os dados com analogia sobre domínios discretos serão representados mediante “relações de semelhança”.

Uma “relação de semelhança” no contexto do GEFRED é um *comparador estendido* que vem representado por uma relação nebulosa binária que satisfaz as propriedades reflexiva e simétrica, ver Seção 2.4.1.1.

- **Igual a.** Este operador modela o conceito de igualdade para dados de natureza imprecisa. Formalmente a função de pertinência é dada por:

$$\mu_{igual}(\tilde{d}, \tilde{d}') = \sup_{(d, d') \in D \times D} \min\{p(d, d'), \pi_{\tilde{d}}(d), \pi_{\tilde{d}'}(d')\} \quad (3.5.7)$$

onde $p(d, d')$ é uma relação de semelhança e $\pi_{\tilde{d}}(d), \pi_{\tilde{d}'}(d')$ são as respectivas distribuições de possibilidade definidas sobre o domínio de discurso D .

- **Aproximadamente igual.** Este operador proporciona o grau em que os valores numéricos precisos são aproximadamente iguais. O cálculo é feito segundo a seguinte expressão:

$$\mu_{aprox.igual}(x, y) = \begin{cases} 0 & \text{se } |x - y| > \text{margem} \\ \frac{1 - |x - y|}{\text{margem}} & \text{se } |x - y| \leq \text{margem} \end{cases} \quad (3.5.8)$$

- **Maior ou igual.** Está definido sobre domínios ordenados. A função de pertinência deste operador vem dada pela relação nebulosa:

$$\mu_{\geq}(A, B) = \sup_{(x, y) \in X \times Y} \min\{\geq(x, y), \pi_A(x), \pi_B(y)\} \quad (3.5.9)$$

sendo A e B dados imprecisos de referencial ordenado ou dados numéricos precisos, $\pi_A(x), \pi_B(y)$ suas respectivas representações possibilísticas e \geq é o operador *maior ou igual* clássico dado por:

$$\geq(x, y) = \begin{cases} 0 & \text{se } x < y \\ 1 & \text{se } x \geq y \end{cases} \quad (3.5.10)$$

Este operador pode resolver as seguintes comparações

- Grau em que um número preciso é *maior ou igual* a uma distribuição de possibilidade.
 - Grau em que uma distribuição de possibilidade é *maior ou igual* que um número preciso.
 - Grau em que uma distribuição de possibilidade é *maior ou igual* a outra distribuição de possibilidade.
- *Menor ou igual.* Está definido sobre domínios ordenados. A função de pertinência deste operador vem dada pela relação nebulosa:

$$\mu_{\leq}(A, B) = \sup_{(x,y) \in X \times Y} \min\{\leq(x, y), \pi_A(x), \pi_B(y)\} \quad (3.5.11)$$

sendo A e B dados imprecisos de referencial ordenado ou dados numéricos precisos, $\pi_A(x), \pi_B(y)$ suas respectivas representações possibilísticas e \leq é o operador *menor ou igual* clássico dado por:

$$\leq(x, y) = \begin{cases} 0 & \text{se } x > y \\ 1 & \text{se } x \leq y \end{cases} \quad (3.5.12)$$

Este operador pode resolver comparações sobre os mesmos tipos que o operador *maior ou igual*.

- **Maior.** Está definido a partir do operador *menor ou igual*. Para isso calculamos o complemento deste operador:

$$\mu_{>}(A, B) = 1 - \mu_{\leq}(A, B) \quad (3.5.13)$$

- **Menor.** Está definido a partir do operador *maior ou igual*. Para isso calculamos o complemento deste operador:

$$\mu_{<}(A, B) = 1 - \mu_{\geq}(A, B) \quad (3.5.14)$$

3.5.4 Implementação do Conhecimento Impreciso na Base de Dados

Neste sistema existem 3 tipos de atributos suscetíveis ao tratamento impreciso. Se classificam segundo o tipo de seus domínios subjacentes.

1. **Atributos Nebulosos do Tipo 1** Atributos com “*dados precisos*” que possuem *etiquetas lingüísticas* definidas sobre eles. Levam associada uma informação adicional em forma de etiquetas lingüísticas. A Base de Metaconhecimento Difuso será a responsável pela representação destas etiquetas, também guardará informação sobre a natureza destes atributos.
2. **Atributos Nebulosos do Tipo 2** Atributos que podem guardar “*dados imprecisos sobre referencial ordenado*”. A Tabela 3.2 apresenta o sistema utilizado para representar os atributos deste tipo.

Tabela 3.2: Representação de Atributos do Tipo 2

Tipo de Dados	F_ TYPE	F1	F2	F3	F4
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[m, n]	5	m	NULL	NULL	n
APROX(d)	6	d	$d - margem$	$d + margem$	$margem$
TRAPEZOIDAL	7	α	$\beta - \alpha$	$\gamma - \delta$	δ

- **F_ TYPE**: Armazena o **tipo de valor** que corresponde ao dado que queremos armazenar, indicando sua representação.
- **F1, F2, F3 e F4**: Armazenam a descrição dos parâmetros que definem o dado e que depende do tipo de valor (**F_ TYPE**):
 - **UNKNOWN, UNDEFINED, NULL**: Estes 3 valores não necessitam de nenhum parâmetro.

- **CRISP**: Um valor do tipo crisp, necessita somente de um parâmetro, F1, onde se armazena o valor crisp em questão.
- **LABEL**: Igualmente, um valor do tipo etiqueta só precisa de um parâmetro para armazenar o indicador associado a etiqueta (FUZZY_ID). Este indicador é útil para poder acessar a FMB e obter a descrição associada a esta etiqueta.
- **INTERVALO**: Necessita dos valores extremos do intervalo $[m, n]$, que são armazenados em F1 e F4, respectivamente.
- **APROXIMADAMENTE**: Este valor só necessita um valor que é armazenado em F1 e que é o valor central da distribuição de possibilidade triangular, d . Para reduzir as operações (tanto matemáticas quanto de acesso a dados), os atributos F2, F3 e F4 são utilizados para armazenar os valores $d - margem$, $d + margem$ e $margem$, respectivamente.
- **TRAPEZIO**: Necessita de todos os quatro valores que identificam um trapézio: $[\alpha, \beta, \gamma, \delta]$. Em F2 e F3 se armazenam uma operações que simplificam as equações quando se opera com este tipo de dado.

3. **Atributos Nebulosos do Tipo 3**: Atributos sobre “domínio discreto não ordenado com analogia”. Estes atributos armazenam dados escalares ou distribuições de possibilidade sobre domínios escalares. Também aceitam dados do tipo UNKNOWN, UNDEFINED e NULL. Para atributos deste tipo é necessário armazenar na Base de Dados o tipo e a representação associada a cada dado. A Base do Metaconhecimento Difuso armazenará as *relações de semelhança* definidas sobre o domínio subjacente.

A Tabela 3.3 mostra a alternativa de implementação adotada para este tipo de atributos.

- **F_TYPE**: O **tipo de valor** que corresponde ao dado que será armazenado.

Tabela 3.3: Representação de Atributos do Tipo 3

Tipo de Dados	F_ TYPE	F_ P1	F1	...	F_ Pn	Fn
UNKNOWN	0	NULL	NULL	...	NULL	NULL
UNDEFINED	1	NULL	NULL	...	NULL	NULL
NULL	2	NULL	NULL	...	NULL	NULL
SIMPLE	3	1	d	...	NULL	NULL
DISTR. POS.	4	p_1	d_1	...	p_n	d_n

- Lista de n pares, com $n \geq 1$, do tipo (valor de possibilidade, etiqueta), $(F_P1, F1), \dots, (F_Pn, Fn)$: Nestes atributos se armazenam os dados da distribuição de possibilidade que desejamos guardar.

3.5.5 Implementação do Conhecimento Impreciso na Base de Metaconhecimento Difuso (FMB)

Como temos visto, existe certo tipo de informação sobre os atributos descritos, que precisa ser armazenada de uma forma acessível pelo sistema. A Base de Metaconhecimento Difuso, FMB, será a encarregada de organizar toda a informação relacionada com a natureza imprecisa destes atributos. No FIRST, a Base de Metaconhecimento Difuso é considerada como sendo uma extensão do Catálogo do sistema, assim a informação é organizada mediante o uso de tabelas ou relações, o que torna complexa sua criação e manutenção.

Tabelas da Base de Metaconhecimento Difuso

A organização das tabelas que constituem a Base de Metaconhecimento Difuso são apresentadas na Figura 3.2.

- Tabela **FUZZY_COL_LIST** (FCL): contém uma descrição daqueles atributos da base de dados que são suscetíveis de tratamento difuso.
- Tabela **FUZZY_OBJECT_LIST** (FOL): contém uma lista de objetos de tipo difuso que estão definidos nas colunas da base de dados. Reflete uma

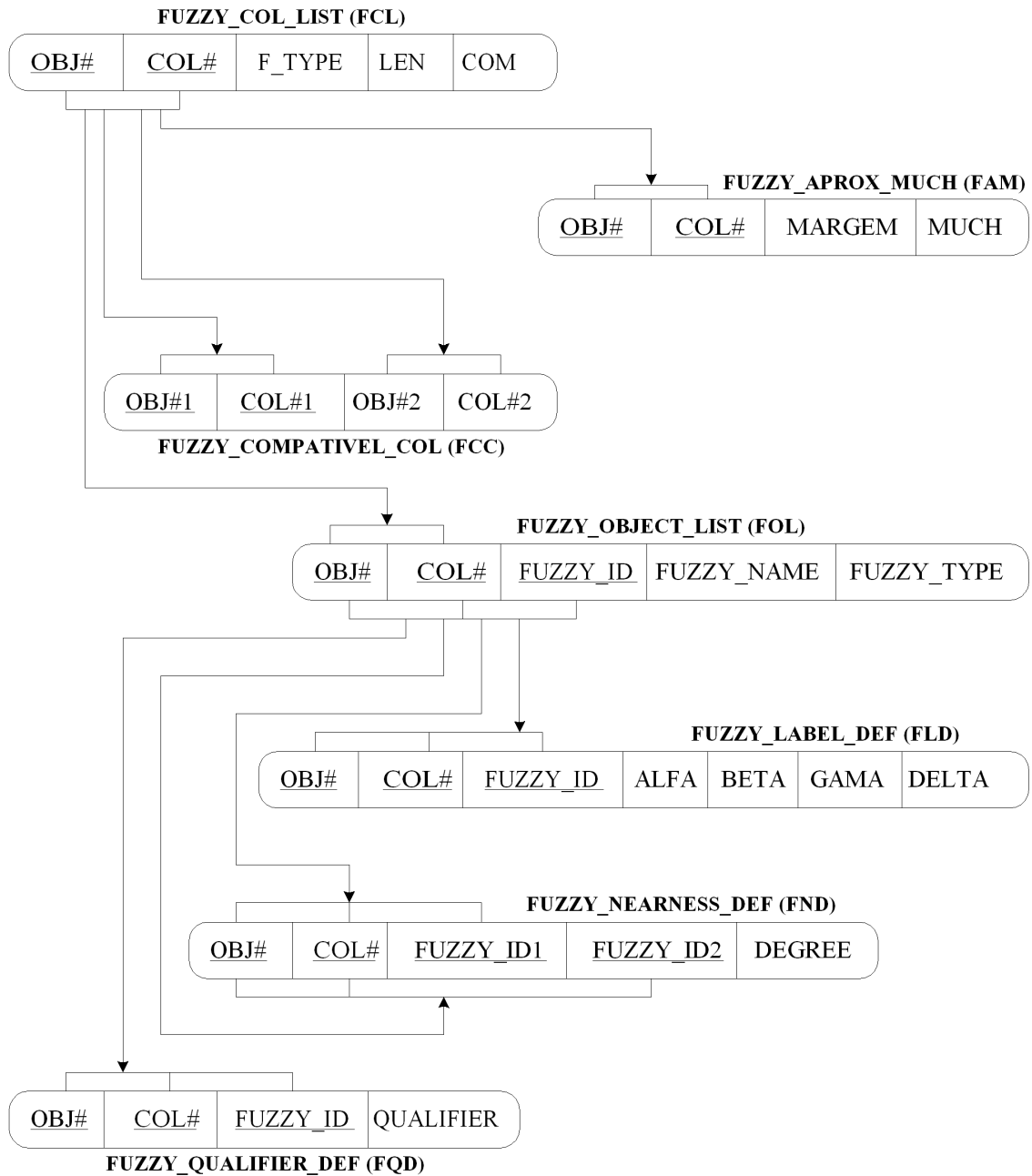


Figura 3.2: Esquema da Base do Metaconhecimento Difuso (FMB)

classificação destes objetos mediante o campo FUZZY_TYPE.

- Tabela **FUZZY_LABEL_DEF** (FLD): contém os pontos que determinam a distribuição de possibilidade trapezoidal correspondentes aos tipos de objetos 0, 3 e 4 do atributo FUZZY_TY-PE da tabela FUZZY_OBJECT_LIST. O

significado dessa distribuição depende, naturalmente, desse tipo.

- Tabela **FUZZY_APROX_MUCH** (FAM): armazena dados que são utilizados quando se trabalha com atributos difusos dos tipos 1 ou 2.
- Tabela **FUZZY_NEARNESS_DEF** (FND): representa a medida de proximidade, semelhança ou similaridade entre os diferentes valores do domínio permitidos sobre os campos do tipo 3.
- Tabela **FUZZY_COMPATIBLE_COL** (FCC): indica os atributos difusos do tipo 3 que são compatíveis com outros. Desta forma não é necessário definir as etiquetas e as relações de similaridade para cada um deles. Isso nos permite comparar os atributos difusos tipo 3 entre si, se são compatíveis, já que isto indica que ambos os atributos possuem o mesmo domínio.
- Tabela **FUZZY_QUALIFIER_DEF** (FQD): expressa o nível de cumprimento mínimo associado ao qualificador lingüístico [tabela **FUZZY_OBJECT_LIST** com **FUZZY_TYPE** igual a 2].

3.5.6 Exemplo de Implementação no FIRST da BD e FMB

Nesta parte será apresentado um exemplo de como se representa no banco de dados e na FMB uma tabela com todos os tipos nebulosos vistos e alguns não nebulosos. O exemplo é ilustrado por uma relação de empregados de uma determinada empresa que está apresentada na Tabela 3.4. Ao longo deste exemplo é mostrado como se implementa a informação que está armazenada nesta relação. É mostrada a estrutura interna que adotam os campos difusos no banco de dados e como se armazenam os dados relativos aos atributos nebulosos na FMB.

Tabela 3.4: Relação Empregados¹

Nome	Salário	Idade	Rendimento
N1	850	31	Bom
N2	1000	Adulto	Regular
N3	900	Jovem	Ruim
N4	210	Maduro	Excelente
N5	970	Jovem	Unknown
N6	1250	#30	Bom
N7	1050	[30, 35]	Regular
N8	1800	\$(22, 25, 33, 35]	Bom
N9	1050	Unknown	Regular

¹O símbolo # significa “aproximadamente”, $[n, m]$ é um intervalo e o valor $\$[\alpha, \beta, \gamma, \delta]$ é um trapézio possibilístico.

A Tabela 3.4 que apresenta a relação de empregados de uma dada empresa possui os seguintes atributos:

- **Nome:** Armazena o nome dos empregados. É um atributo “crisp” não difuso, do tipo texto. Este campo é a chave primária da tabela.
- **Salário:** Armazena o salário de cada empregado. É um atributo Tipo 1. Portanto, todos os valores que podem ser armazenados nesta coluna são do tipo “crisp”. Este atributo mantém informações na FMB. A Figura 3.3 mostra a definição das etiquetas lingüísticas definidas sobre este atributo. O valor da margem é de 150, para valores aproximados, e a distância mínima é 400, para considerar dois valores como sendo muito separados. Ambos valores são armazenados na Tabela 3.7 FUZZY_APPROX_MUCH.
- **Idade:** Armazena a idade de cada empregado. É um atributo difuso Tipo 2 e pode portanto armazenar todos os tipos de dados mostrados na Tabela 3.2. A Figura 3.4 mostra a definição das etiquetas lingüísticas definidas sobre este atributo. O valor da margem para valores aproximados é 5 e a

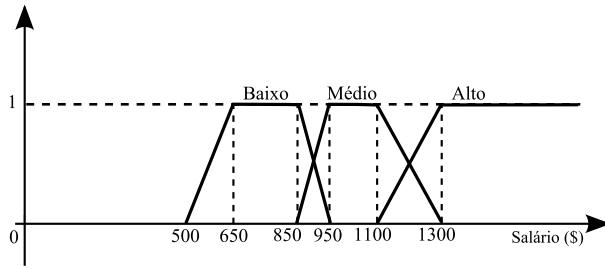


Figura 3.3: Definição de etiquetas sobre o atributo **Salário**

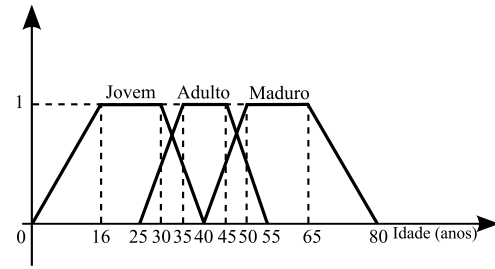


Figura 3.4: Definição de etiquetas sobre o atributo **Idade**

distância mínima para considerar dois valores como muito separados é 9. Ambos valores são armazenados em FUZZY_APPROX_MUCH, Tabela 3.7

- **Rendimento:** Armazena a qualificação de rendimento de cada empregado. É um atributo do Tipo 3 e pode, portanto, armazenar os tipos de dados que se mostram na Tabela 3.3. Para simplificar é assumido que a máxima longitude das distribuições de possibilidade para este atributo é 1. As etiquetas (escalares) que são definidas e sua relação de similaridade estão apresentadas na Tabela 3.5.

Tabela 3.5: Relação de Similaridade s_r sobre o atributo **Rendimento**

$s_r(d, d')$	Ruim	Regular	Bom	Excelente
Ruim	1	0.8	0.5	0.1
Regular	0.8	1	0.7	0.5
Bom	0.5	0.7	1	0.8
Excelente	0.1	0.5	0.8	1

Com esses dados, a tabela FUZZY_COL_LIST é apresentada na Tabela 3.6. Para um melhor entendimento, os números dos atributos OBJ# e COL# foram substituídos pelo nome dos objetos.

A TABELA 3.8, um exemplo de FUZZY_OBJECT_LIST, apresenta todas as etiquetas das Figuras 3.3 e 3.4 e da Tabela 3.5. As etiquetas trapezoidais re-

Tabela 3.6: Exemplo para a tabela FUZZY_COL_LIST

OBJ#	COL#	F_TYPE	LEN	COM
(Empregados)	(Salário)	1	NULL	'EMPREGADOS.SALÁRIO'
(Empregados)	(Idade)	2	NULL	'EMPREGADOS.IDADE'
(Empregados)	(Rendimento)	3	1	'EMPREGADOS.RENDIMENTO'

Tabela 3.7: Exemplo para a tabela FUZZY_APPROX_MUCH

OBJ#	COL#	MARGEM	MUCH
(Empregados)	(Salário)	150	400
(Empregados)	(Idade)	5	9

gistradas nesta tabela, com FUZZY_TYPE = 0, estão definidas em FUZZY_LABEL_DEF, Tabela 3.9.

Tabela 3.8: Exemplo para a tabela FUZZY_OBJECT_LIST

OBJ#	COL#	FUZZY_ID	FUZZY_NAME	FUZZY_TYPE
(Empregados)	(Salário)	0	'BAIXO'	0
(Empregados)	(Salário)	1	'MEDIO'	0
(Empregados)	(Salário)	2	'ALTO'	0
(Empregados)	(Idade)	0	'JOVEM'	0
(Empregados)	(Idade)	1	'ADULTO'	0
(Empregados)	(Idade)	2	'MADURO'	0
(Empregados)	(Rendimento)	0	'RUIM'	1
(Empregados)	(Rendimento)	1	'REGULAR'	1
(Empregados)	(Rendimento)	2	'BOM'	1
(Empregados)	(Rendimento)	3	'EXCELENTE'	1

A relação de similaridade sobre o atributo RENDIMENTO da Tabela 3.5, é armazenada na FMB em FUZZY_NEARNESS_DEF, Tabela 3.10.

Uma vez feita estas definições, a relação Empregados implementada no SGBD Oracle possui a estrutura interna mostrada na Tabela 3.11, sendo adaptada para

Tabela 3.9: Exemplo para a tabela FUZZY_LABEL_DEF

OBJ#	COL#	FUZZY_ID	ALFA	BETA	GAMA	DELTA
(Empregados)	(Salário)	0	500	650	850	950
(Empregados)	(Salário)	1	850	950	1100	1300
(Empregados)	(Salário)	2	1100	1300	9999	9999
(Empregados)	(Idade)	0	0	16	30	40
(Empregados)	(Idade)	1	25	35	45	55
(Empregados)	(Idade)	2	40	50	65	80

Tabela 3.10: Exemplo para a tabela FUZZY_NEARNESS_DEF

OBJ#	COL#	FUZZY_ID1	FUZZY_ID2	DEGREE
(Empregados)	(Rendimento)	0	1	0.8
(Empregados)	(Rendimento)	0	2	0.5
(Empregados)	(Rendimento)	0	3	0.1
(Empregados)	(Rendimento)	1	2	0.7
(Empregados)	(Rendimento)	1	3	0.5
(Empregados)	(Rendimento)	2	3	0.8

apresentar os atributos Idade e Rendimento, a implementação mostrada nas Tabelas 3.2 e 3.3, respectivamente.

Tabela 3.11: Representação Interna Real da relação **Empregados** no SGBD com a extensão do FIRST

NOME	SALÁRIO	IDADET	IDADE1	IDADE2	IDADE3	IDADE4	...
N1	850	3	31	NULL	NULL	NULL	...
N2	1000	4	1	NULL	NULL	NULL	...
N3	900	4	0	NULL	NULL	NULL	...
N4	210	4	2	NULL	NULL	NULL	...
N5	970	4	0	NULL	NULL	NULL	...
N6	1250	6	30	25	35	5	...
N7	1050	5	30	NULL	NULL	35	...
N8	1800	7	22	25	33	35	...
N9	1050	0	NULL	NULL	NULL	NULL	...

...	REND.T	REND.P1	REND.1
	3	1	2
	3	1	1
	3	1	0
	3	1	3
	0	NULL	NULL
	3	1	2
	3	1	1
	3	1	2
	3	1	1

Exemplo 3.1 *Um exemplo de consulta que mostra o grau de cumprimento (usando CDEG), usa uma constante do tipo trapezoidal e evita os valores UNKNOWN e apresentado a seguir:*

Observações a respeito deste exemplo:

1. O grau mínimo de cumprimento está estabelecido como sendo 0.75. Assim, na tabela resultante, a coluna CDEG(Habitantes) terá valores no intervalo $[0.75, 1]$.

```

SELECT  Cidade, CDEG(Habitantes)
FROM    População
WHERE   País = 'Espanha'
          AND Habitantes FGEQ  $[200, 350, 650, 800]$  0.75
          AND Habitantes IS NOT UNKNOWN;

```

2. Como o comparador FGEQ foi usado, os valores γ e δ do trapézio não serão usados, isto é, se o número de habitantes é crisp e excede 350, o grau de cumprimento será 1. Naturalmente se o número de habitantes é crisp e menor que 200 o grau será 0.
3. Se uma cidade espanhola possui no atributo difuso Habitantes a distribuição de possibilidade $[50, 150, 200, 350]$, seu grau de cumprimento da condição será 0.5 e não aparecerá no resultado final, já que o grau mínimo estabelecido é 0.75.

Um exemplo de aplicação deste modelo na área de Turismo pode ser visto em [19].

Por fazer uso da estrutura de tabelas para armazenar sua Base de Metaconhecimento Difuso, o modelo GEFRED apresenta-se complexo. A plena compreensão da organização do conhecimento torna-se um processo difícil até mesmo para usuários especializados. O modelo *Aliança*, estudado nesta tese, introduz um novo caminho para representar esta base de metainformação nebulosa organizada no formato XML, que simplifica as tarefas de gerenciamento de dados.

3.6 Trabalho de Turowski e Weng

Nesta seção e na próxima são apresentados trabalhos que tratam a representação de dados nebulosos em arquivos XML.

O trabalho de Turowski e Weng [20] apresenta uma forma de representar e processar informações nebulosas baseada em XML.

Estes pesquisadores constataram que apesar do uso da lógica nebulosa melhorar o processo de decisão das empresas, estas soluções ainda estão restritas a áreas especiais e raramente são integradas aos sistemas de negócios principais das empresas. Neste trabalho eles apresentaram a proposta de usar um caminho comum para representar informações nebulosas suavizando esta integração. Foi introduzida uma sintaxe formal para os tipos de dados nebulosos usados para armazenar informações nebulosas. Esta sintaxe tornou-se operativa através da definição de um documento apropriado (DTD – Document Type Definitions). Foi descrito como as informações, que serão descritas em DTDs, podem ser trocadas entre sistemas de aplicação através do XML (extensible markup language).

Observou-se que a colaboração entre aplicações nebulosas desenvolvidas usando diferentes ambientes ainda apresentam problemas de integração. Turowski e Weng propuseram um formato comum para troca de informações nebulosas com o objetivo de diminuir os problemas de integração.

A aproximação apresentada é baseada no uso da linguagem XML. O formato XML foi escolhido porque permite a criação livre de *tokens* e estruturação livre do documento. Para a definição de novos *tokens* (também chamados de *tags* em XML) e a customização da estruturas dos documentos, os DTDs foram utilizados. DTDs são parte do XML. Um DTD serve como um *template* que auxilia na explicação da sintaxe e do conteúdo do documento que está baseado em um DTD específico.

Foi proposto o encapsulamento das informações nebulosas em *tags* XML nomeadas de acordo com um conjunto padronizado. Com as informações nebulosas encapsuladas em *tags* XML padronizadas as mensagens que contenham informações nebulosas de outros sistemas de aplicações podem ser processadas e com isso ter seu conteúdo conhecido.

O exemplo da Figura 3.5 mostra como combinar um documento XML e uma folha de estilo para representar informações nebulosas em um navegador de Internet. O conjunto nebuloso é descrito em um documento XML. A forma de apresentação está definida em uma folha de estilo. O resultado é uma tabela com duas colunas e três linhas. Cada linha representa um ponto de um conjunto nebuloso

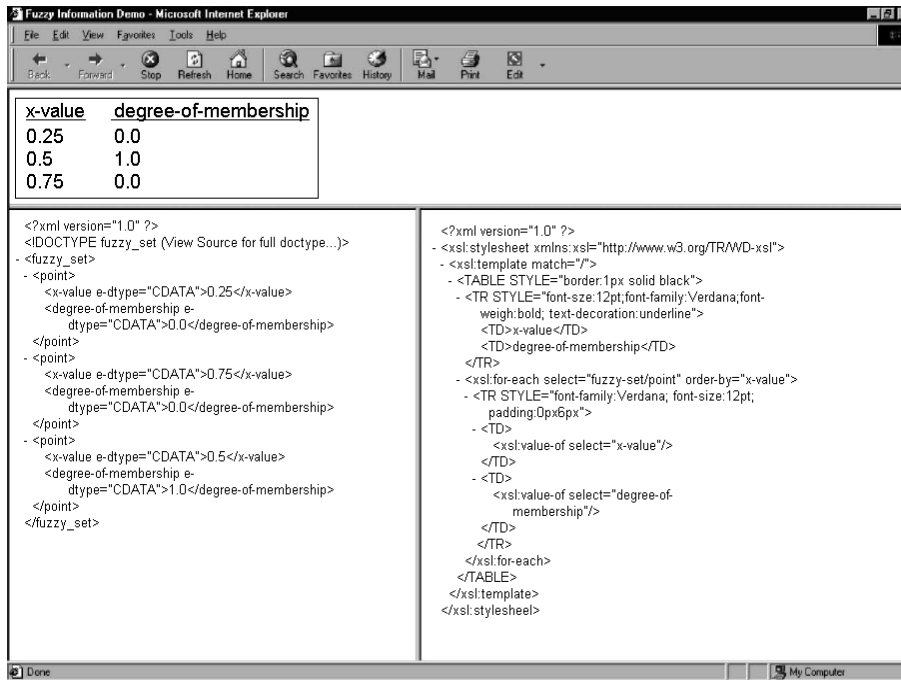


Figura 3.5: Exemplo de um conjunto nebuloso contínuo

contínuo.

3.6.1 A modelagem da informação nebulosa

O elemento base da informação nebulosa é o conjunto nebuloso descrito por sua função de inclusão. Existem dois tipos de conjuntos: os conjuntos discretos (para cada ponto é associado um único grau de inclusão) e os conjuntos contínuos (graus de inclusão são associados e calculados por interpolação).

Números nebulosos e intervalos nebulosos, que são necessários para operações aritméticas, são definidos baseados em conjuntos nebulosos contínuos, enquanto que inferências nebulosas são baseadas em conceitos lingüísticos.

Um controlador nebuloso suporta inferências nebulosas pela aplicação de uma base de regras. Esta base de regras contém regras no formato se-então. Os operadores nebulosos também são descritos na base de regras.

3.6.2 DTD

Para cada tipo de dado nebuloso foi definido um DTD que estabelece limitadores na estrutura lógica do documento XML. O que fornece um conjunto de regras para a estrutura de um documento.

Exemplo 3.2 DTD de um conjunto nebuloso discreto

```
<!ELEMENT discrets.fuzzy_set (object, degree-of-membership) * />
<!ELEMENT object ANY>
<!ELEMENT degree-of-membership (#PCDATA) />
<!ATTLIST degree-of-membership e-dtype NMTOKEN #FIXED 'float' />
```

O DTD para um conjunto nebuloso discreto é apresentado no Exemplo 3.2. Na primeira linha é especificado o elemento raiz de um conjunto discreto. Ele consiste de zero ou mais pares de elementos ‘*object*’ (‘objeto’) e ‘*degree-of-membership*’ (‘graus de inclusão’). O *token* ‘*’ expressa a cardinalidade do elemento raiz. O conteúdo do elemento ‘*degree-of-membership*’ (‘grau de inclusão’) é seguido por (#PCDATA). A lista de atributos (ATTLIST) descreve os tipos de dados representados em mais detalhes. Neste caso, o tipo de dado do elemento ‘*degree-of-membership*’ é ‘*float*’.

O processador XML assegura o tratamento apropriado para este tipo de declaração de dado, uma vez que o processador de XML é um software que acessa o conteúdo e a estrutura de um documento XML.

Este trabalho está direcionado a troca de informações entre sistemas, que apesar de tratar informações nebulosas, estas se limitam a conjuntos nebulosos discretos e contínuos. Não são mostrados detalhes de armazenamento deste tipo de informação nos bancos de dados.

3.7 O mapeamento entre um banco de dados nebulosos e um arquivo XML nebuloso - Um Trabalho de Gauray e Alhajj

O Trabalho de Gauray e Alhajj [21] descreve como mapear dados de uma base de dados relacional em um documento XML.

Neste trabalho, Gauray e Alhajj apresentaram uma aproximação para incorporar dados nebulosos e imprecisos em documentos XML. Os caminhos utilizados para descrever a introdução da nebulosidade foram a teoria da possibilidade e as relações de similaridade. Foi mostrado como os dados são mapeados de um banco de dados relacional nebulosos em um documento XML, com o correspondente XML esquema (*schema*). Esta aproximação foi adicionada para distribuir dados armazenados em bancos de dados nebulosos pela web como documentos XML nebulosos.

Em geral, um documento XML permite manter dados exatos e precisos. Conseqüentemente, sem dar suporte para imprecisão e nebulosidade. Foram usados conjuntos nebulosos e distribuições de possibilidade para incorporar nebulosidade ao XML.

Especificamente, tentou-se identificar as entidades em XML que pudessem ter valores nebulosos ou que pudessem tratar a nebulosidade. Primeiramente foi analisada a estrutura de um documento XML para identificar os vários pontos que poderiam ser manipuladas usando nebulosidade; e então foram especificados os mecanismos para incorporar a nebulosidade.

Documentos XML são “*bem formados*”, de acordo com as recomendações XML, e podem (ou não) ser validados. Estes documentos apresentam uma estrutura lógica e uma estrutura física. Do ponto de vista da inclusão de dados nebulosos em documentos XML, a estrutura lógica é a chave da questão, uma vez que esta define o conteúdo (dados) de um documento XML. Considerando que elementos são a chave de um documento XML, o estudo foi focado nos elementos somente

desconsiderando os atributos. O XML esquema possui dois tipos de elementos: simples e complexos. Elementos simples possuem precisamente zero atributos e zero elementos em sua definição, enquanto que elementos complexos podem ter um ou mais atributos e/ou um ou mais elementos em sua definição. Os elementos complexos podem ser classificados como: a) elementos vazios, b) elementos que possuem somente outros elementos, c) elementos que contém somente texto e d) elementos que contém texto e elementos.

3.7.1 Mapeamento de Bancos de Dados Nebuloso Relacionais para arquivos XML Nebuloso

Para traduzir bancos de dados relacionais nebulosos em XML nebulosos, três interpretações de bancos de dados relacionais nebulosos foram utilizadas.

Os autores consideraram a primeira interpretação de banco de dados nebulosos como sendo aquela em que uma tupla pertence a uma relação com um certo grau. O algoritmo para o mapeamento da relação nebulosa para o XML nebuloso foi definido como segue. A Tabela 3.12 é usada como ilustração.

Tabela 3.12: Uma relação nebulosa ‘Likes’ com tuplas nebulosas

Name1	Name2	i
John	Ross	0.6
Tom	Amanda	0.3
Lisa	Joe	0.9
Rob	Emily	0.7

- Mapeamento do nome de cada relação para um elemento complexo no XML esquema com a definição de um simples sub-elemento para as tuplas.
- Na tupla tipo, criou-se um elemento <fuzzyDegree> como o primeiro elemento. Mapeou-se todos os outros atributos regulares como elementos subsequentes. Por exemplo:

Exemplo 3.3 XML nebuloso da Tabela 3.12 - 'Likes'

```
<xs:complexType name="LikesTupleType" >
  <xs:sequence>
    <xs:element name="fuzzyDegree", type="xs:decimal" />
    <xs:element name="Name1", type="xs:string" />
    <xs:element name="Name2", type="xs:string" />
  </xs:sequence>
</xs:complexType>
```

- Criou-se um elemento raiz que representasse o elemento do banco de dados incluindo todas as estruturas e seus sub-elementos.
- Usou-se <key> e <keyref> para incluir unicidade e integridade referencial.

Considerou-se que a segunda interpretação de um banco de dados nebulosos relacional é aquela em que os valores de atributos são nebulosos e a distribuição de possibilidade é associada ao atributo. Para este caso, o algoritmo de mapeamento foi definido como mostrado abaixo usando a Tabela 3.13 como ilustração.

Tabela 3.13: Uma relação nebulosa 'Employees' com atributos nebulosas

Emp_id	Name	Age	Salary
4867	Joe	30	High
5189	Tom	Young	About 4000
9876	John	About 28	4500
3189	Sara	Middle-aged	Average

- Mapeia-se cada nome de relação em um elemento do tipo complexo no esquema XML com uma definição simples para as tuplas.
- Mapeia-se todos os atributos como sub-elementos com a definição dos tipos das tuplas. Para cada atributo que possuir valores nebulosos definiu-se um elemento <fuzzyValue> com um grau como sub-elemento no elemento atributo. Como um exemplo, foi definido o tipo da tupla em *Employees* (Empregados) como segue:

Exemplo 3.4 XML nebuloso da Tabela 3.13 - ‘Employees’

```
<xs:complexType name="EmployeeTupletype" >
  <xs:sequence>
    <xs:element name="emp_id", type="xs:integer" />
    <xs:element name="name", type="xs:string" />
    <xs:element name="age" >
      <complexType>
        <xs:element name="fuzzyValue" />
        ...
        <xs:attribute name="degree" type="xs:decimal" />
        ...
      </xs:complexType>
    
```

Opcionalmente, também incluiu-se `<fuzzyDegree>` como um primeiro elemento com a definição do tipo da tupla para fundir a primeira com a segunda interpretação.

- Criou-se um elemento raiz que representa o elemento banco de dados e inclui todas as relações com seus sub-elementos.
- Usou-se `<key>` e `<keyref>` para incluir unicidade e integridade referencial.

Considerou-se que a terceira interpretação dos bancos de dados nebulosos relacional é aquela que a relação de similaridade introduz nebulosidade associando-se a cada um dos atributos nebulosos. O algoritmo para mapear este tipo de relação nebulosa em um XML nebuloso possui um passo extra e é apresentado a seguir.

Tabela 3.14: Uma relação nebulosa ‘Employees_Body’ com relação de similaridade para o atributo ‘Health’ dada na Tabela 3.15

Emp_id	Health	Height	Looks
4867	Good	6-3	Good
5189	Fair	5-11	Excellent
9876	Bad	5-8	Good

Tabela 3.15: Relação de similaridade associada ao atributo ‘Health’

	Bad	Fair	Good	Excellent
Bad	1	0.8	0.5	0.1
Fair	0.8	1	0.7	0.5
Good	0.5	0.7	1	0.8
Excellent	0.1	0.5	0.8	1

- Mapeou-se cada nome de relação em um elemento complexo no XML esquema com um sub-elemento para as tuplas.
- Definiu-se um sub-elemento `<SimilarityRelation Name="xyz">` com o elemento relação para cada relação do atributo similaridade. Pode ser possível que múltiplos atributos associem-se a mesma relação de similaridade. O seguinte exemplo mostra a definição de uma relação de similaridade no XML esquema.

Exemplo 3.5 XML nebuloso da Tabela 3.14 - ‘Employees_Body’

```

<xs:complexType name="BodyTupleType" >
...
<xs:element name="SimilarityRelation", minOccurs="0", maxOccurs="unbounded" />
...
<xs:attribute name="Name", type="xs:string" />
</xs:element>
...
</xs:complexType>

```

- Mapeou-se todos os atributos como sub-elementos com a definição do tipo de tupla. Para cada atributo inseriu-se um atributo “SimilarityRelationRef”, o valor que será o correspondente nome da relação de similaridade.

Exemplo 3.6 XML nebuloso da Tabela 3.15 - 'Health'

```
<xs:complexType name="BodyTupleType" >
  <xs:sequence>
    <xs:element name="health" />
    ...
    <xs:attribute name="SimilarityRelationRef", type="xs:string" />
  </xs:element>
</xs:element name="heigth" >
  ...
</xs:element>
...
</xs:complexType>
```

- Criou-se um elemento raiz que representa o elemento banco de dados e inclui todas as relações como sub-elementos.
- Usou-se <key> e <keyref> para incluir unicidade e integridade referencial

O trabalho de Gauray e Alhadj possui o objetivo de extrair os dados de um banco de dados relacional que armazena dados nebulosos e apresentá-los em formato XML para que estes possam ser interpretados por um outro sistema. Não apresenta detalhes da forma como estes dados nebulosos são armazenados em sua base de dados.

Capítulo 4

O Sistema de Gerenciamento de Banco de Dados Nebulosos *Aliança*

Neste capítulo é apresentada uma nova arquitetura para banco de dados nebulosos *Aliança*. São mostrados seus elementos básicos e como eles se relacionam.

4.1 Objetivo

Banco de dados nebulosos usualmente armazenam informações imprecisas que necessitam de metainformações a elas associadas, com o objetivo de adicionar contexto e semântica. A complexidade de se armazenar e tratar os conceitos nebulosos em estruturas de bancos de dados tradicionais foi a mais importante motivação desta pesquisa.

Esta pesquisa tem como objetivo principal apresentar uma nova proposta de arquitetura para banco de dados nebulosos que buscará introduzir uma forma diferenciada de representação para a base de metaconhecimento nebuloso, visando simplificar as questões de gerenciamento de dados nebulosos.

Ao longo da pesquisa, alguns objetivos foram traçados:

1. Tornar as consultas a Bancos de Dados mais próximas do raciocínio humano;
2. Ser acessível a qualquer tipo de usuário de banco de dados, especializado ou

- não;
3. Ser de fácil manutenção;
 4. Apresentar respostas satisfatórias para as consultas feitas;
 5. Apresentar graus de satisfação ao responder consultas feitas sobre condições ou atributos nebulosos;
 6. Proporcionar um tratamento coerente para os dados nebulosos respeitando sua natureza;
 7. Apresentar uma base de metaconhecimento nebuloso organizada de forma simplificada.

4.2 Arquitetura

A Figura 4.1 mostra a arquitetura geral do Banco de Dados Nebulosos - *Aliança* juntamente com os principais módulos do sistema:

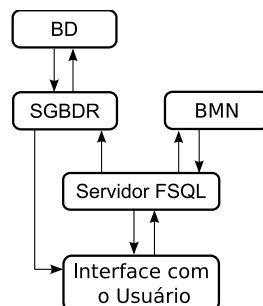


Figura 4.1: Arquitetura Geral do Banco de Dados Nebulosos *Aliança*

- **SGBDR** (Sistema de Gerenciamento de Banco de Dados Relacionais): Todas as operações nebulosas, ou que envolvem dados nebulosos, são transformadas pelo Servidor FSQ em operações clássicas e então realizadas pelo gerenciador principal. Diferentemente do FIRST (em GEFRED) [2], em

Aliança o SGBDR não possui ligação direta com a base do metaconhecimento nebuloso.

- **BD** (Banco de Dados): Armazena em tabelas as informações da mesma maneira que os bancos de dados tradicionais. Além disso, *Aliança* permite o armazenamento de dados nebulosos em suas tabelas. Esta informação nebulosa é armazenada usando um conjunto de atributos que definem todas as características relevantes dos dados nebulosos, que serão mais detalhados a diante.
- **BMN** (Base de Metaconhecimento Nebuloso): Armazena de forma organizada as informações adicionais sobre os dados nebulosos que estão no BD.
- **Servidor FSQL** (Servidor SQL Nebuloso): É o módulo principal do sistema, pois é o responsável pelo relacionamento entre o SGBD e a BMN. Seu principal objetivo é transformar as informações dadas em FSQL em SQL clássico para que o gerenciador possa processá-las e retornar uma resposta.
- **Interface com o usuário**: A interface com o usuário estabelece a comunicação entre os usuários e o Servidor FSQL.

4.2.1 Representação do Conhecimento Impreciso

Dentre os diferentes tipos de informação imprecisas que podem ser armazenadas em um banco de dados nebulosos foi definido que a nova arquitetura de banco de dados nebulosos *Aliança* aceitará 8 diferentes tipos de dados. Estes 8 tipos formam um conjunto rico que cobre os tipos de dados nebulosos considerados.

- **Dados Precisos** (*crisp*): São dados precisos comumente tratados pelos bancos de dados tradicionais, podendo ou não receber um tratamento nebuloso. Estes dados são classificados como sendo do **Tipo 0** e não necessitam que informações adicionais sejam armazenadas na Base de Metaconhecimento Nebuloso (BMN). Cadeias alfanuméricas, valores numéricos, datas são exemplos do usual formato dos dados precisos.

- **Unknown** (*desconhecido, mas aplicável*): Um atributo recebe o valor Unknown quando este pode tomar qualquer valor do domínio de discurso, mas não se pode afirmar exatamente qual é este valor. Dados deste tipo são denominados como sendo do **Tipo 1**. O tipo Unknown, apresentado pela Equação 3.3.1, é representado mediante uma distribuição de possibilidade, $\{\frac{1}{u}, \forall u \in U\}$, onde U é o domínio considerado. A Figura 4.2 mostra graficamente esta distribuição.

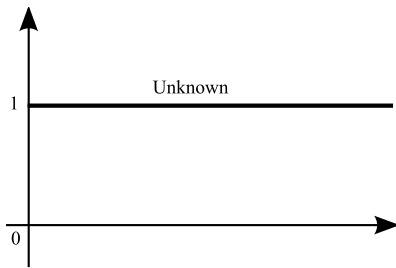


Figura 4.2: Distribuição de possibilidade para o tipo Unknown

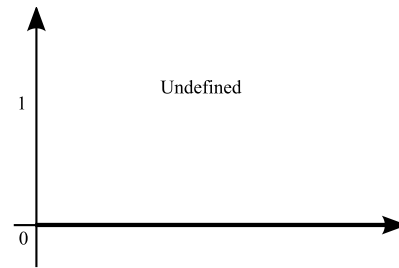


Figura 4.3: Distribuição de possibilidade para o tipo Undefined

- **Undefined** (*indefinido, não aplicável*): Um atributo recebe o valor Undefined quando nenhum dos valores do domínio sobre o qual está definido é aplicável. Dados deste tipo são denominados do **Tipo 2**. O tipo Undefined, apresentado pela Equação 3.3.2, é representado mediante uma distribuição de possibilidade, $\{\frac{0}{u}, \forall u \in U\}$, onde U é o domínio considerado. A Figura 4.3 mostra esta distribuição.
- **Null** (*ignorância absoluta*): Um atributo recebe o valor Null quando não há nenhuma informação, tanto quando não a conhecemos (Unknown) quanto não é aplicável (Undefined), como apresentado pela Equação 3.3.3. Dados deste tipo são denominados do **Tipo 3**.
- **Etiqueta Lingüística com Distribuição de Possibilidade**: Quando um atributo é associado a um valor nebuloso, ele recebe uma etiqueta lingüística com distribuição de possibilidade. Estes dados são classificados como sendo do **Tipo 4** e possuem associados a eles uma distribuição de possibilidade

trapezoidal armazenada na BMN. A Figura 4.4 mostra um exemplo de etiqueta lingüística com distribuição de possibilidade. Trapézios são muito usados em sistemas nebulosos para representar valores vagos.

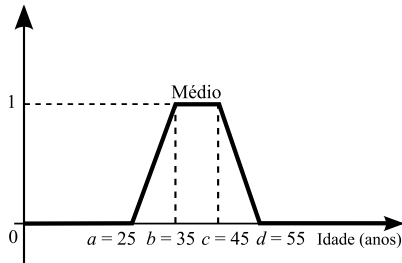


Figura 4.4: Um exemplo de um rótulo lingüístico para o conceito “Médio”

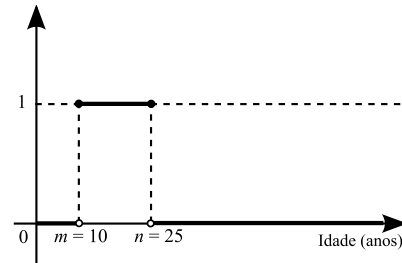


Figura 4.5: Um exemplo de intervalo de possibilidade

- **Intervalo de Possibilidade** $[m, n]$: Dados deste tipo são denominados como sendo do **Tipo 5** e possuem uma distribuição de possibilidade intervalar associada a ele. A Figura 4.5 mostra um exemplo de intervalo de possibilidade. Este tipo de dados necessitam que informações adicionais sejam armazenadas na BMN.
- **Valor Aproximado** (*aproximadamente d*): Dado um valor d , pertencente ao domínio, o conceito impreciso *aproximadamente d* é definido por uma distribuição de possibilidade triangular construída sobre d e uma *margem m* como mostra a Figura 4.6. Este é o **Tipo 6** e também necessitam que informações adicionais sejam armazenadas na BMN para definir a margem usada.
- **Etiqueta Lingüística com Similaridade**: É um tipo de dado que está construído sobre um domínio não ordenado. Neste domínio encontram-se definidos relacionamentos de similaridade (semelhança) entre as etiquetas lingüísticas que o constituem. Este relacionamento é representado por uma tabela que apresenta as relações diretas entre todos os pares de valores pertencentes ao domínio. Este é o **Tipo 7** e necessita que estas relações de

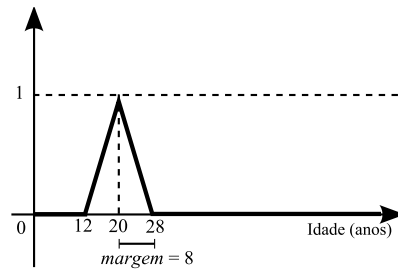


Figura 4.6: Um exemplo de valor aproximado

similaridade sejam armazenadas na BMN, como mostrado mais adiante na Tabela 4.3.

4.3 Base do Metaconhecimento Nebuloso

Como apresentado na seção 4.2.1, certos tipos de dados necessitam que algumas informações sobre eles sejam armazenadas para que possam ser tratados de maneira adequada.

A base de metaconhecimento nebuloso (BMN) contém, de maneira organizada, a informação adicional necessária requerida pelo sistema.

Diferentemente do que foi feito no FIRST (em [1] e [2]), o banco de dados *Aliança* não armazena sua BMN em tabelas ou relações dentro do próprio SGBD. A BMN no *Aliança* é descrita no formato XML. Este formato torna a BMN fácil de ser compreendida e mantida. Outra consequência é o fato de não ser necessário acessar complexas tabelas para recuperar informações descrevendo a semântica das variáveis nebulosas.

A informação armazenada para cada tipo, previamente apresentado na seção 4.2.1, é mostrada na Tabela 4.1.

Como pode ser observado, os tipos de dados 0, 1, 2 e 3 não armazenam informações adicionais na BMN.

Tabela 4.1: Informações armazenadas na BMN

Tipo de Dado	Identificador do Tipo	Informação Armazenada
Dados Precisos	0	Nenhuma
Unknown	1	Nenhuma
Undefined	2	Nenhuma
Null	3	Nenhuma
Etiquetas Lingüísticas com Distribuição de Possibilidade	4	Rótulos e suas características
Intervalo de Possibilidade	5	Mínimo e Máximo
Valor Aproximado	6	Margem
Etiqueta Lingüística com Similaridade	7	Pares (a, b) , Grau de similaridade entre a e b

4.3.1 Estrutura da BMN

A base do metaconhecimento nebuloso é onde toda informação adicional necessária para manipular as transações no banco de dados que envolvam dados nebulosos está armazenada. *Aliança*, diferentemente dos projetos anteriores, define para a BMN uma estrutura de diretórios, onde o diretório raiz é chamado pelo mesmo nome do banco de dados. Cada tabela do banco de dados contém, no diretório raiz, um subdiretório de mesmo nome. Este subdiretório contém um arquivo XML para cada atributo.

Será apresentada, através de um exemplo, a estrutura interna dos campos nebulosos e a nova maneira de armazenar a BMN do *Aliança*. O exemplo tratado é um banco de dados chamado *Antiquário* que contém uma relação que armazena informações sobre carros antigos. A Figura 4.7 mostra a estrutura de diretórios da BMN enquanto a Tabela 4.2 apresenta uma lista de carros antigos usada como exemplo.

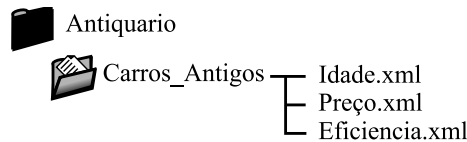


Figura 4.7: Estrutura da BMN

Tabela 4.2: Carros Antigos ¹

Id_Carro	Modelo	Preço	Idade	Eficiência
001	Alfa Romeo JK	#17500	34	Ruim
002	Alfa Romeo Convertible	35000	Antigo	Regular
003	Dodge Polara	[7000, 8000]	29	Ruim
004	Dodge Dart	Médio	#35	Excelente
005	Porsche Spyder 550	28000	Antigo	Unknown
006	Porsche Spyder 550	Alto	Unknown	Boa
007	Willys Gordini	Baixo	[38, 43]	Regular
008	Willys Bicuda	#6000	Médio	Ruim

¹ O símbolo # representa “aproximadamente” e $[m, n]$ é um intervalo e significa entre m e n

A descrição dos atributos da Tabela 4.2 é mostrada abaixo de acordo com o critério de classificação usado em *Aliança*, para os diferentes tipos nebulosos.

- **Id_Carro:** É a chave primária da tabela do tipo número inteiro serial preenchido automaticamente pelo SGBD.
- **Modelo:** O modelo do carro. Campo texto do tipo preciso.
- **Preço:** O preço do carro. Atributo sujeito a tratamento impreciso, assim pode ser preenchido com valores dos Tipos 0, 1, 2, 3, 4, 5 e 6. Este atributo necessita que informações adicionais sejam armazenadas na BMN. O valor da *margem* (Tipo 6) foi definido, para este exemplo, como sendo 1000. As definições das etiquetas lingüísticas que usam distribuições de possibilidade (Tipo 4) são mostradas na Figura 4.8.

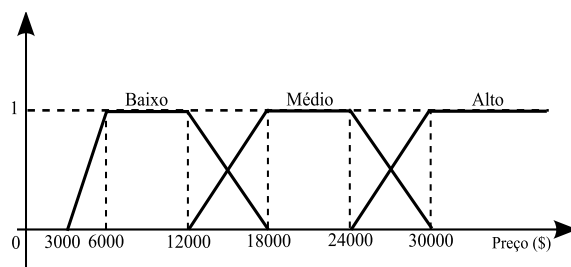


Figura 4.8: Definição das possíveis etiquetas usadas para o atributo **Preço**

- **Idade:** Idade do carro. É também um atributo que armazena dados nebulosos, assim os Tipos 0, 1, 2, 3, 4, 5 e 6 podem ser usados. O valor da *margin* (Tipo 6) foi definido como 5. As etiquetas lingüísticas baseadas em possibilidades são apresentadas na Figura 4.9.

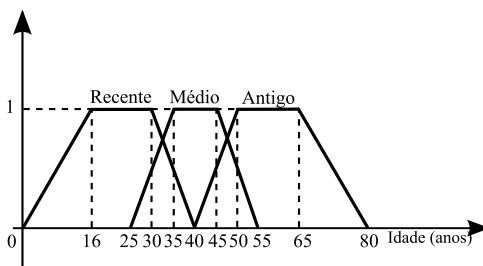


Figura 4.9: Definição das possíveis etiquetas usadas para o atributo **Idade**

- **Eficiência:** Armazena a eficiência de cada carro quando seus desempenhos são comparados. É um outro atributo que pode ser tratado como uma quantidade nebulosa. Diferentemente do atributo anterior este é baseado em relações de similaridade, assim apenas valores dos Tipos 1, 2, 3 e 7 podem ser armazenados. As etiquetas lingüísticas baseadas em similaridade e as possíveis similaridades entre seus pares, para o atributo eficiência, são apresentadas na Tabela 4.3.

Tabela 4.3: Relação de Similaridade s_r definida sobre o atributo **Eficiência**

$s_r(d, d')$	Ruim	Regular	Boa	Excelente
Ruim	1	0.8	0.5	0.1
Regular	0.8	1	0.7	0.5
Boa	0.5	0.7	1	0.8
Excelente	0.1	0.5	0.8	1

4.3.2 Descrição dos Arquivos XML

Como visto na seção anterior, os atributos **Idade** e **Preço** são definidos aceitando valores do Tipo 0, 1, 2, 3, 4, 5 e 6 e necessitam que informações adicionais sejam armazenadas na BMN em seus respectivos XML, ver Tabela 4.1.

O arquivo **Idade.xml** é dividido em seções internas (*tags*) e cada seção define algumas características do atributo. O arquivo **Idade.xml** é apresentado a seguir:

Exemplo 4.1 Arquivo Idade.xml

```

<? XML VERSION = "1.0" >

<IDADE>
  <DOMINIO A=0 B=110 />
  <TIPO T=4>
    <ROTULOS>
      <RECENTE A=0 B=16 C=30 D=40 />
      <MEDIO A=25 B=35 C=45 D=55 />
      <ANTIGO A=40 B=50 C=65 D=80 />
    </ROTULOS>
  </TIPO>
  <TIPO T=5>
    <INTERVALO MIN=1 MAX=5 />
  </TIPO>
  <TIPO T=6>
    <MARGEM M=5>
  </TIPO>
</IDADE>

```

* Tag <DOMINIO A=0 B=110 />: Apresenta o domínio $[A, B]$ sobre o qual o atributo nebuloso é descrito.

- * Tag <TIPO T=4>: Apresenta as características que representam as *distribuições de possibilidade* para cada rótulo (Tipo 4), ver Seção 4.2.1.
 - Tag <ROTULOS>: Esta subseção define todos os rótulos usados para este atributo, ver Figura 4.9, onde A, B, C e D são respectivamente *a*, *b*, *c* e *d* (Figura 4.4).
 - . Tag <RECENTE A=0 B=16 C=30 D=40 />: Apresenta os parâmetros que descrevem a distribuição trapezoidal RECENTE.
 - . Tag <MEDIO A=25 B=35 C=45 D=55 />: Apresenta os parâmetros que descrevem a distribuição trapezoidal MEDIO.
 - . Tag <ANTIGO A=40 B=50 C=65 D=80 />: Apresenta os parâmetros que descrevem a distribuição trapezoidal ANTIGO.

- * Tag <TIPO T=5>: Apresenta as características da *distribuição de possibilidade intervalar* (Tipo 5), ver Seção 4.2.1.
 - Tag <INTERVALO MIN=1 MAX=5 />: Esta tag estabelece os tamanhos mínimo e máximo para o intervalo.

- * Tag <TIPO T=6>: Apresenta as características requeridas para representar um *valor aproximado* (Tipo 6), ver Seção 4.2.1.
 - Tag <MARGEM M=5>: A tag define para valores aproximados a margem M. Neste caso definiu-se M como sendo 5.

Da mesma maneira, o arquivo **Preço.xml** armazena as informações adicionais do atributo **Preço**.

Exemplo 4.2 *Arquivo Preço.xml*

```
<? XML VERSION = "1.0" >

<PREÇO>
  <DOMINIO A=500 B=100000 />
  <TIPO T=4>
    <ROTULOS>
      <BAIXO A=3000 B=6000 C=12000 D=18000 />
      <MEDIO A=12000 B=18000 C=24000 D=30000 />
      <ALTO A=24000 B=30000 C=50000 D=100000 />
    </ROTULOS>
  </TIPO>
  <TIPO T=5>
    <INTERVALO MIN=500 MAX=3000 />
  </TIPO>
  <TIPO T=6>
    <MARGEM M=1000>
  </TIPO>
</PREÇO>
```

Para que fosse possível apresentar um exemplo de dados nebulosos do Tipo 7 o atributo **Eficiência** foi considerado como sendo definido sobre um domínio baseado em similaridade onde a imprecisão dos valores dos atributos estão primariamente em seus significados, ver Seção 2.4.1.1. Podendo assim somente ser preenchido com dados dos Tipos 1, 2, 3 ou 7. Portanto, o arquivo XML deve ser chamado de **Eficiencia.xml** e contém informações apresentadas no exemplo 4.3.

Exemplo 4.3 Arquivo Eficiência.xml

```
<? XML VERSION = "1.0" >
<EFICIENCIA>
  <DOMINIO A=ruim B=regular C=boa D=excelente />
  <TIPO T=7>
    <ROTULOS>
      <RUIM ruim=1 regular=0.8 boa=0.5 excelente=0.1 />
      <REGULAR ruim=0.8 regular=1 boa=0.7 excelente=0.5 />
      <BOA ruim=0.5 regular=0.7 boa=1 excelente=0.8 />
      <EXCELENTE ruim=0.1 regular=0.5 boa=0.8 excelente=1 />
    </ROTULOS>
  </TIPO>
</EFICIENCIA>
```

A tag DOMINIO apresenta o domínio discreto sobre o qual o atributo nebuloso **Eficiência** é descrito. Note que o domínio é composto de um conjunto de rótulos.

* Tag <DOMINIO A=ruim B=regular C=boa D=excelente />: Apresenta o domínio discreto sobre o qual o atributo nebuloso é descrito.

Como o atributo **Eficiência** é baseado em similaridades é necessário descrever cada relacionamento.

* Tag <TIPO T=7>: Apresenta as características necessárias para representar a *etiqueta lingüística baseada em similaridade*.

- Tag <ROTULOS>: Apresenta o grau de similaridade entre os rótulos.

- . Tag <RUIM ruim=1 regular=0.8 boa=0.5 excelente=0.1 />.
- . Tag <REGULAR ruim=0.8 regular=1 boa=0.7 excelente=0.5 />.
- . Tag <BOA ruim=0.5 regular=0.7 boa=1 excelente=0.8 />.
- . Tag <EXCELENTE ruim=0.1 regular=0.5 boa=0.8 excelente=1 />.

4.3.3 Representação Interna da Base de Dados

A Tabela 4.4 apresenta a real estrutura interna da Tabela 4.2. É importante notar que essa representação interna é transparente para o usuário.

Tabela 4.4: Representação interna da relação **Carros Antigos** no SGBD proposto *Aliança*

Id	Modelo	Preço	PreçoT	Preço1	Preço2	...
001	Alfa Romeo JK	#17500	6	17500	1000	
002	Alfa Romeo Conv.	35000	0	35000,00	-	
003	Dodge Polara	[7000, 8000]	5	7000	8000	
004	Dodge Dart	\$Médio	4	-	-	
005	Porsche Spyder	28000,00	0	28000,00	-	
006	Porsche Spyder	\$Alto	4	-	-	
007	Willys Gordini	\$Baixo	4	-	-	
008	Willys Bicuda	#6000	6	6000	1000	

...	Idade	IdadeT	Idade1	Idade2	Eficiencia	EficienciaT
	34	0	34	-	\$\$Ruim	7
	\$Antigo	4	-	-	\$\$Regular	7
	29	0	29	-	\$\$Ruim	7
	#35	6	35	5	\$\$Excelente	7
	\$Antigo	4	-	-	Unknown	1
	Unknown	1	-	-	\$\$Boa	7
	[38, 43]	5	38	43	\$\$Regular	7
	\$Medio	4	-	-	\$\$Ruim	7

Para o melhor entendimento dessa estrutura interna e seus atributos complementares, o exemplo considerado será novamente o atributo **Idade**.

O atributo **Idade** foi definido aceitando valores dos Tipos 0, 1, 2, 3, 4, 5 e 6, como pode ser visto nas seções anteriores. Para que os dados nebulosos pudessem ser tratados de maneira coerente, três novos atributos foram adicionados à Tabela 4.2:

- **IdadeT**: define o tipo do dado nebuloso armazenado podendo assumir um dos valores {0, 1, 2, 3, 4, 5, 6}.
- **Idade1** e **Idade2**: apresentam informações complementares de acordo com o

tipo armazenado. Uma descrição mais detalhada será apresentada na Tabela 4.5.

Tabela 4.5: Descrição dos novos atributos usados representação interna dos valores nebulosos

Idade	IdadeT	Idade1	Idade2	<i>Descrição</i>
30	0	30	-	30 é um valor preciso, IdadeT = 0 indica que o dado é do Tipo 0 Idade1 = 30 armazena o próprio valor preciso 30
Unknown	1	-	-	O rótulo Unknown é um atributo do Tipo 1, assim IdadeT = 1
Undefined	2	-	-	O rótulo Undefined é um atributo do Tipo 2, assim IdadeT = 2
Null	3	-	-	O rótulo Null é um atributo do Tipo 3, assim IdadeT = 3
\$Antigo	4	-	-	O rótulo \$Antigo é uma função trapezoidal, IdadeT = 4 indica que o dado é do Tipo 4
[30,45]	5	30	45	O valor [30, 45] é do tipo intervalo IdadeT = 5 que indica que o dado é do Tipo 5 Idade1 = 30 e Idade2 = 45 indicam os valores extremos do intervalo
#25	6	25	5	O valor #25 é do tipo valor aproximado, IdadeT = 6 indica que o dado é do Tipo 6, Idade1 = 25 indica que o valor central é 25 Idade2 = 5 indica que o tamanho da margem é 5

Como o atributo nebuloso **Eficiência** é definido baseado em similaridade, ocorrerá a adição de apenas um atributo extra chamado **EficienciaT**. Este novo atributo se faz necessário para a identificação do tipo do dado nebuloso, uma vez que os Tipos 1, 2, 3 e 7 não requerem informações extras dentro da base de dados.

Os novos atributos são usados quando uma consulta nebulosa é transformada em uma consulta clássica.

O uso de uma estrutura de diretórios e arquivos em XML facilita a criação e a manutenção de bancos de dados nebulosos quando comparados com o modelo anterior GEFRED descrito em [1] e [2]. O modelo GEFRED usa uma série de tabelas localizadas junto ao banco de dados para representar sua base de metac conhecimento nebuloso o que dificulta muito a criação de novos bancos de dados e as manutenções destes.

4.4 Comparadores Nebulosos

Os comparadores nebulosos apresentados nesta seção são aqueles escolhidos para serem utilizados pela linguagem FSQL.

Dos tipos de dados nebulosos suportados por *Aliança*, apresentados na Seção 4.2.1, somente ao Tipo 7 - *Etiquetas Lingüísticas com Similaridade* não se aplicam todos os comparadores nebulosos. Por ser um tipo descrito sobre domínio baseado em similaridade apenas o operador igualdade é considerado. Além disso, dados do Tipo 7 podem apenas comparar-se com elementos do mesmo Tipo 7.

A seguir serão descritas as representações adotadas para os diferentes comparadores nebulosos utilizados pelo banco de dados *Aliança*:

- **Possivelmente igual a (FEQ):** Este operador modela o conceito de *possivelmente igual* para dados de natureza nebulosa. Sua função de pertinência, para obtermos o grau em que X é possivelmente igual a Y, é formalmente dada por (ver Definição 2.22):

$$\mu_{FEQ}(X, Y) = \sup_{x_i \in \Omega} [\min(X(x_i), Y(x_i))] \quad (4.4.1)$$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6.

A Figura 4.10 apresenta graficamente um exemplo de como é obtido o grau de igualdade entre duas etiquetas lingüísticas (X) e (Y).

Para se obter o grau de igualdade entre dados do tipo 7 basta obter o grau de similaridade entre eles consultando a matriz de similaridade, ver Seção

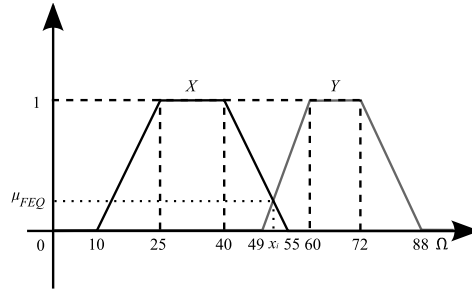


Figura 4.10: Um exemplo do operador FEQ

2.4.1.1.

$$\mu_{FEQ}(a, b) = s_r(a, b) \quad (4.4.2)$$

sendo a e b dados do tipo 7, e $s_r(a, b)$ o grau de similaridade entre eles.

- **Possivelmente maior ou igual que (FGEQ):** Este operador modela o conceito de *possivelmente maior ou igual* para dados de natureza nebulosa. Sua função de pertinência, para obtermos o grau em que X é possivelmente maior ou igual a Y , é formalmente dada por:

$$\mu_{FGEQ}(X, Y) = \sup_{x_i \in \Omega} [\min (X(x_i), \geq (Y(x_i)))] \quad (4.4.3)$$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6 e \geq é o operador “*maior ou igual*” estendido ($\geq (Y)$, apresentado na Seção 2.1.3.7).

A Figura 4.11 apresenta um valor aproximado (X) e uma etiqueta lingüística (Y) usados como exemplo. Enquanto a Figura 4.12 apresenta graficamente como é obtido o grau em que o valor aproximado (X) é *possivelmente maior ou igual* à etiqueta lingüística (Y).

- **Possivelmente menor ou igual que (FLEQ):** Este operador modela o conceito de *possivelmente menor ou igual* para dados de natureza nebulosa. Sua função de pertinência, para obtermos o grau em que X é possivelmente menor ou igual a Y , é formalmente dada por:

$$\mu_{FLEQ}(X, Y) = \sup_{x_i \in \Omega} [\min (X(x_i), \leq (Y(x_i)))] \quad (4.4.4)$$

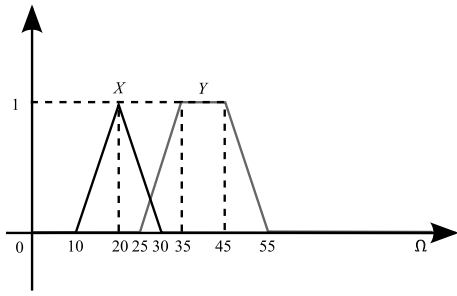


Figura 4.11: Valor aproximado X e etiqueta lingüística Y

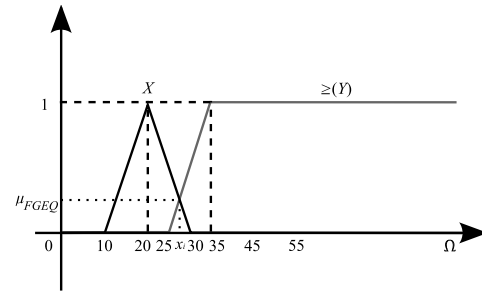


Figura 4.12: $\mu_{FGEQ}(X, Y)$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6 e \leq é o operador “menor ou igual” estendido ($\leq(Y)$), apresentado na Seção 2.1.3.7).

A Figura 4.13 apresenta um valor preciso (X) e um intervalo de possibilidade (Y) usados como exemplo. Enquanto a Figura 4.14 apresenta graficamente como é obtido o grau em que o valor de dado preciso (X) é *possivelmente menor ou igual* ao intervalo de possibilidade (Y).

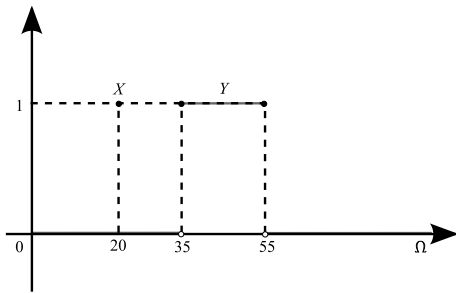


Figura 4.13: Valor preciso X e intervalo de possibilidade Y

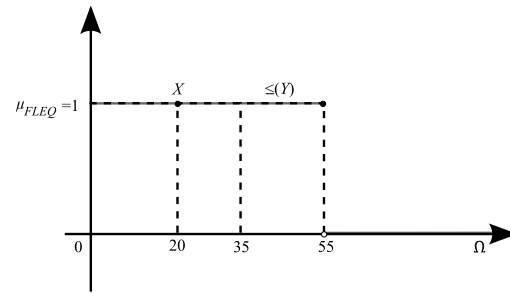


Figura 4.14: $\mu_{FLEQ}(X, Y)$

- **Possivelmente maior que (FGT):** Este operador modela o conceito de *possivelmente maior* para dados de natureza nebulosa, ver Seção 2.1.3.7). Sua função de pertinência, para obtermos o grau em que X é possivelmente maior que Y , é formalmente dada por:

$$\mu_{FGT}(X, Y) = 1 - \mu_{FLEQ}(X, Y) \quad (4.4.5)$$

sendo FLEQ o operador estendido “*menor ou igual a*”.

- **Possivelmente menor que (FLT):** Este operador modela o conceito de *possivelmente menor* para dados de natureza nebulosa, ver Seção 2.1.3.7). Sua função de pertinência, para obtermos o grau em que X é possivelmente menor que Y, é formalmente dada por:

$$\mu_{FLT}(X, Y) = 1 - \mu_{FGEQ}(X, Y) \quad (4.4.6)$$

sendo FGEQ o operador estendido “*maior ou igual a*”.

- **Possivelmente muito maior que (MGT):** Este operador modela o conceito de *possivelmente muito maior que* para dados de natureza nebulosa. Sua função de pertinência, para obtermos o grau em que X é possivelmente muito maior que Y, é formalmente dada por:

$$\mu_{MGT}(X, Y) = \sup_{x_i \in \Omega} [\min (X(x_i), >_{muito} (Y(x_i)))] \quad (4.4.7)$$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6 e $>_{muito}$ é o operador “*muito maior que*”, apresentado na Seção 2.1.3.7).

A Figura 4.15 apresenta um valor aproximado (X) e uma etiqueta lingüística (Y) usados como exemplo. Enquanto a Figura 4.16 apresenta graficamente como é obtido o grau em que o valor aproximado (X) é *possivelmente muito maior que* a etiqueta lingüística (Y).

- **Possivelmente muito menor que (MLT):** Este operador modela o conceito de *possivelmente muito menor que* para dados de natureza nebulosa. Sua função de pertinência, para obtermos o grau em que X é possivelmente muito menor que Y, é formalmente dada por:

$$\mu_{MLT}(X, Y) = \sup_{x_i \in \Omega} [\min (X(x_i), <_{muito} (Y(x_i)))] \quad (4.4.8)$$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6 e $<_{muito}$ é o operador “*muito menor que*”, apresentado na Seção 2.1.3.7).

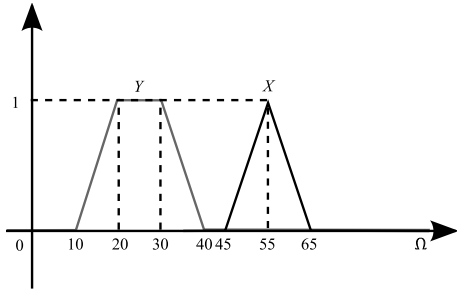


Figura 4.15: Valor aproximado X e etiqueta lingüística Y

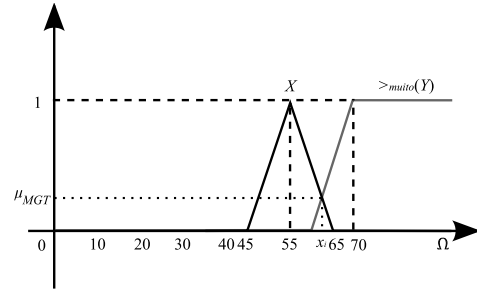


Figura 4.16: $\mu_{MGT}(X, Y)$

A Figura 4.17 apresenta um intervalo de possibilidades (X) e uma etiqueta lingüística (Y) usados como exemplo. Enquanto a Figura 4.18 apresenta graficamente como é obtido o grau em que o intervalo de possibilidades (X) é *possivelmente muito menor* que a etiqueta lingüística (Y).

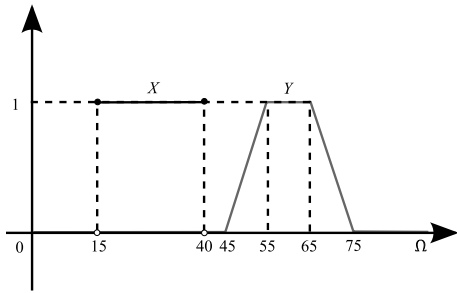


Figura 4.17: Intervalo de possibilidade X e etiqueta lingüística Y

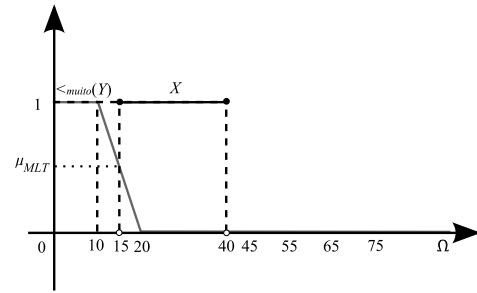


Figura 4.18: $\mu_{MLT}(X, Y)$

- **Necessariamente igual a ($NFEQ$):** Este operador modela o conceito de “*necessariamente igual a*” para dados de natureza nebulosa, sendo mais restritivo que o operador “*possivelmente igual a*”. Sua função de pertinência, para obtermos o grau em que X é necessariamente igual a Y , é formalmente dada por (ver Definição 2.23):

$$\mu_{NFEQ}(X, Y) = \inf_{x_i \in \Omega} [\max(1 - X(x_i), Y(x_i))] \quad (4.4.9)$$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6.

A Figura 4.19 apresenta um valor aproximado (X) e um intervalo de possibilidades (Y) usados como exemplo. Enquanto a Figura 4.20 apresenta graficamente como é obtido o grau em que o valor aproximado (X) é *necessariamente igual* ao intervalo de possibilidades (Y).

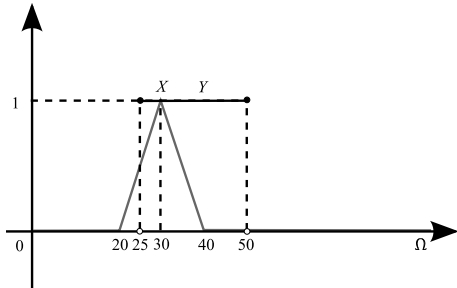


Figura 4.19: Valor aproximado X e intervalo de possibilidade Y

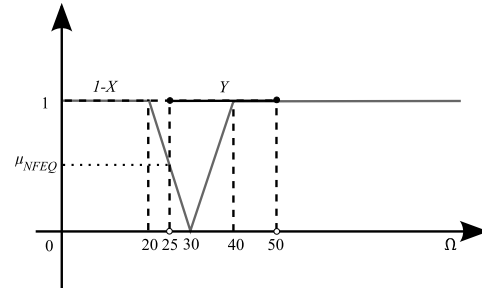


Figura 4.20: $\mu_{NFEQ}(X, Y)$

- **Necessariamente maior ou igual a (NFGT):** Este operador modela o conceito de “*necessariamente maior ou igual a*” para dados de natureza nebulosa. A função de pertinência, para obtermos o grau em que X é necessariamente maior ou igual a Y , é formalmente dada por:

$$\mu_{NFGT}(X, Y) = \inf_{x_i \in \Omega} [\max(1 - X(x_i), \geq(Y(x_i)))] \quad (4.4.10)$$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6 e \geq é o operador “*maior ou igual*” estendido ($\geq(X)$), apresentado na Seção 2.1.3.7).

A Figura 4.21 apresenta um intervalo de possibilidades (X) e uma etiqueta lingüística (Y) usados como exemplo. Enquanto a Figura 4.22 apresenta graficamente como é obtido o grau em que o intervalo de possibilidades (X) é *necessariamente maior ou igual a* etiqueta lingüística (Y).

- **Necessariamente menor ou igual a (NFLT):** Este operador modela o conceito de “*necessariamente menor ou igual a*” para dados de natureza

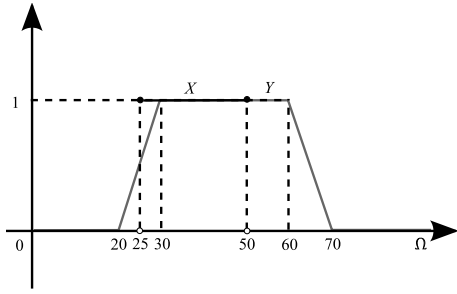


Figura 4.21: Intervalo de possibilidade X e etiqueta lingüística Y

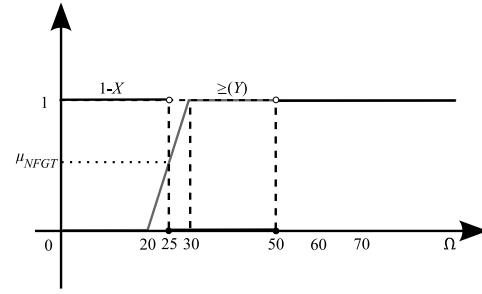


Figura 4.22: $\mu_{NFGT}(X, Y)$

nebulosa. A função de pertinência, para obtermos o grau em que X é necessariamente menor ou igual a Y , é formalmente dada por:

$$\mu_{NFLT}(X, Y) = \inf_{x_i \in \Omega} [\max(1 - X(x_i), \leq(Y(x_i)))] \quad (4.4.11)$$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6 e \leq é o operador “menor ou igual” estendido ($\leq(X)$), apresentado na Seção 2.1.3.7).

A Figura 4.23 apresenta uma etiqueta lingüística (X) e um valor preciso (Y) usados como exemplo. Enquanto a Figura 4.24 apresenta graficamente como é obtido o grau em que a etiqueta lingüística (X) é *necessariamente menor ou igual ao* valor preciso (Y).

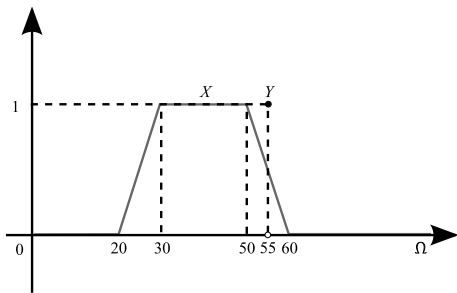


Figura 4.23: Etiqueta lingüística X e valor preciso Y

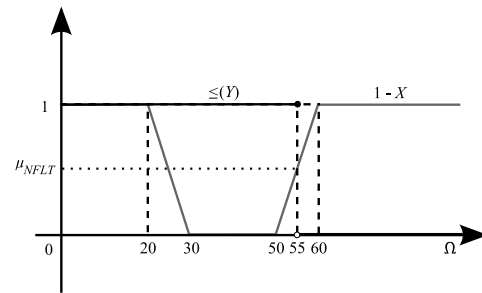


Figura 4.24: $\mu_{NFLT}(X, Y)$

- **Necessariamente maior que (NFGT):** Este operador modela o conceito de *necessariamente maior* para dados de natureza nebulosa. Sua função de

pertinência, para obtermos o grau em que X é necessariamente maior que Y , é formalmente dada por:

$$\mu_{NFGT}(X, Y) = 1 - \mu_{NFLEQ}(X, Y) \quad (4.4.12)$$

sendo NFLEQ o operador estendido “*necessariamente menor ou igual que*”.

- **Necessariamente menor que (NFLT):** Este operador modela o conceito de *necessariamente menor* para dados de natureza nebulosa. Sua função de pertinência, para obtermos o grau em que X é necessariamente menor que Y , é formalmente dada por:

$$\mu_{NFLT}(X, Y) = 1 - \mu_{NFGEQ}(X, Y) \quad (4.4.13)$$

sendo NFGEQ o operador estendido “*necessariamente maior ou igual que*”.

- **Necessariamente muito maior que (NMGT):** Este operador modela o conceito de *necessariamente muito maior que* para dados de natureza nebulosa. Sua função de pertinência, para obtermos o grau em que X é necessariamente muito maior que Y , é formalmente dada por:

$$\mu_{NMGT}(X, Y) = \inf_{x_i \in \Omega} [\max(1 - X(x_i), >_{muito}(Y(x_i)))] \quad (4.4.14)$$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6 e $>_{muito}$ é o operador “*muito maior que*” apresentado na Seção 2.1.3.7.

A Figura 4.25 apresenta um valor aproximado (X) e uma etiqueta lingüística (Y) usados como exemplo. Enquanto a Figura 4.26 apresenta graficamente como é obtido o grau em que o valor aproximado (X) é *necessariamente muito maior que* a etiqueta lingüística (Y).

- **Necessariamente muito menor que (NMLT):** Este operador modela o conceito de *necessariamente muito menor que* para dados de natureza nebulosa. Sua função de pertinência, para obtermos o grau em que X é necessariamente muito menor que Y , é formalmente dada por:

$$\mu_{NMLT}(X, Y) = \inf_{x_i \in \Omega} [\max(1 - X(x_i), <_{muito}(Y(x_i)))] \quad (4.4.15)$$

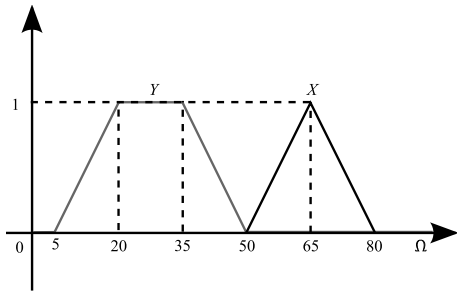


Figura 4.25: Valor aproximado X e etiqueta lingüística Y

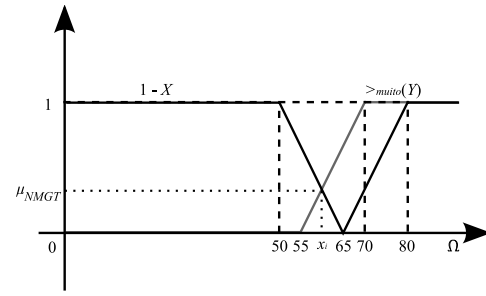


Figura 4.26: $\mu_{NMGT}(X, Y)$

sendo que Ω denota o universo de discurso dos dados imprecisos X e Y que podem ser dos tipos 0 ao 6 e $<_{muito}$ é o operador “*muito menor que*”, apresentado na Seção 2.1.3.7).

A Figura 4.27 apresenta um intervalo de possibilidades (X) e uma etiqueta lingüística (Y) usados como exemplo. Enquanto a Figura 4.28 apresenta graficamente como é obtido o grau em que o intervalo de possibilidades (X) é *necessariamente muito menor que* a etiqueta lingüística (Y).

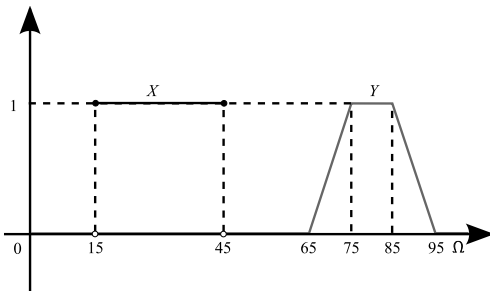


Figura 4.27: Intervalo de possibilidade X e etiqueta lingüística Y

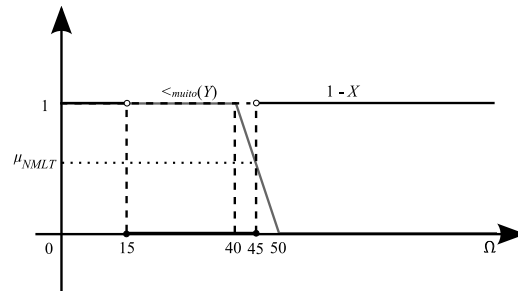


Figura 4.28: $\mu_{NMLT}(X, Y)$

4.5 Servidor FSQL

O servidor FSQL é o principal módulo do sistema, porque é o responsável pelo relacionamento entre o SGBD e a BMN. Sua função é transformar uma consulta FSQL em uma consulta SQL clássica.

4.5.1 FSQL - Fuzzy SQL

Foram definidas para o FSQL (*Fuzzy Structured Query Language*) novas ferramentas com o objetivo de facilitar a manipulação das informações imprecisas. São elas:

- **Etiquetas Lingüísticas:** Quando um atributo é preenchido por uma etiqueta lingüística, esta etiqueta é precedida por um símbolo que a torna fácil de ser identificada. Como existem dois tipos de etiquetas lingüísticas, as que apresentam uma distribuição de possibilidade são precedidas pelo símbolo \$ enquanto que as que representam relações de similaridade são precedidas pelo símbolo \$\$.
- **Intervalo de Possibilidades** $[m, n]$: são representados por dois valores numéricos m e n entre chaves $[m, n]$.
- **Valor Aproximado:** são precedidos pelo símbolo #. Por exemplo, #10 que significa “aproximadamente 10”
- **Comparadores Nebulosos:** além dos operadores de comparação clássicos ($>$, $<$, $=$, ...), o FSQL proposto possui comparadores nebulosos (FEQ, FGEQ, NFEQ, NFLEQ, ...), apresentados na Seção 4.4.
- **Grau de Aceitação:** Para cada condição nebulosa, usada na consulta, um grau de aceitação α pode ser definido. Este grau aparece após a apresentação da condição na consulta.

Exemplo 4.4 *Selecione os modelos dos carros que possuem preço alto com um grau maior ou igual a 0.8.*


```
SELECT Modelo
FROM Carros_Antigos
WHERE Preço FEQ Alto 0.8
```

Exemplo 4.5 *Selecione modelos e preços dos carros cuja idade está possivelmente entre aproximadamente 20 anos e antigo, com um grau pelo menos igual a 0.7.*

```
SELECT Modelo, Preço
FROM Carros_Antigos
WHERE Idade FGEQ #20 AND Idade FLEQ Antigo 0.7
```

4.5.2 Tratamento Especial dado às Etiquetas Lingüísticas com Distribuições de Possibilidade

As *Etiquetas Lingüísticas baseadas em Distribuições de Possibilidade* (Tipo 4) quando comparadas entre si recebem um tratamento diferenciado.

Este tratamento especial é a possibilidade de armazenar, em sua BMN, uma matriz de semelhança para cada um dos comparadores nebulosos *possivelmente igual a*, *possivelmente maior ou igual a*, *possivelmente menor ou igual a*, *possivelmente maior que*, *possivelmente menor que*, *necessariamente igual a*, *necessariamente maior ou igual a*, *necessariamente menor ou igual a*, *necessariamente maior que* e *necessariamente menor que*.

O que facilita os processos de consultas, uma vez que uma simples busca de um valor numa matriz substituiria as diversas operações matemáticas que seriam realizadas no momento de execução da consulta sempre que duas etiquetas lingüísticas baseadas em possibilidade fossem comparadas.

Estas *matrizes de semelhança com operações estendidas* serão apresentadas através de um exemplo.

Exemplo 4.6 *Considerando o atributo Preço, que aceita dados do Tipo 4, per-*

tencente à Tabela 4.2 de Carros Antigos. E suas etiquetas lingüísticas apresentadas na Figura 4.8.

1. *Matriz de Semelhança com Operação Estendida - $(FEQ(d, d'))$ Possivelmente igual a*: A Tabela 4.6 apresenta as relações de semelhança sobre o comparador nebuloso “ $(FEQ(d, d'))$ - Possivelmente igual a” entre as etiquetas do atributo **Preço**.

Tabela 4.6: Relação de Semelhança sobre o comparador nebuloso “ $(FEQ(d, d'))$ - Possivelmente igual a”

$FEQ(d, d')$	Baixo	Médio	Alto
Baixo	1	0.5	0
Médio	0.5	1	0.5
Alto	0	0.5	1

2. *Matriz de Semelhança com Operação Estendida - $(FGEQ(d, d'))$ Possivelmente maior ou igual a*: A Tabela 4.7 apresenta as relações de “ $(FGEQ(d, d'))$ - Possivelmente maior ou igual a” entre as etiquetas do atributo **Preço**.

Tabela 4.7: Relação de Semelhança sobre o comparador nebuloso “ $(FGEQ(d, d'))$ - Possivelmente maior ou igual a”

$FGEQ(d, d')$	Baixo	Médio	Alto
Baixo	1	0.5	0
Médio	1	1	0.5
Alto	1	1	1

Esta Relação de Semelhança apresentada na Tabela 4.7 não é simétrica, logo apenas pode ser lida linha a linha. Por exemplo, na primeira linha temos que Baixo é \geq Alto com grau 0, e na última linha que Alto é \geq Baixo com grau 1.

Com as matrizes de semelhança definidas, o arquivo XML da BMN para o atributo **Preço** deve ser complementado.

Este mesmo raciocínio pode ser estendido para os demais comparadores nebulosos.

Esta é uma maneira simplificada de representar as informações que estão implicitamente apresentadas na definição das etiquetas lingüísticas do Tipo 4 (Função de Inclusão Trapezoidal). Lembrando que a maioria das matrizes de semelhança não é simétrica, e portanto podem apenas ser consideradas as relações representadas linha a linha.

*Arquivo **Preço.xml** com o tratamento especial dado ao operador **FEQ**.*

```
<? XML VERSION = "1.0" >

<PREÇO>
  <DOMINIO A=500 B=4000 />
  <TIPO T=4>
    <ROTULOS>
      <BAIXO A=3000 B=6000 C=12000 D=18000 />
      <MEDIO A=12000 B=18000 C=24000 D=30000 />
      <ALTO A=24000 B=30000 C=50000 D=100000 />
    </ROTULOS>
    <FEQ>
      <BAIXO BAIXO=1 MEDIO=0.5 ALTO=0 />
      <MEDIO BAIXO=0.5 MEDIO=1 ALTO=0.5 />
      <ALTO BAIXO=0 MEDIO=0.5 ALTO=1 />
    </FEQ>
    <FGEQ>
      <BAIXO BAIXO=1 MEDIO=0.5 ALTO=0 />
      <MEDIO BAIXO=1 MEDIO=1 ALTO=0.5 />
      <ALTO BAIXO=1 MEDIO=1 ALTO=1 />
    </FGEQ>
  </TIPO>
  <TIPO T=5>
    <INTERVALO MIN=10 MAX=200 />
  </TIPO>
  <TIPO T=6>
    <MARGEM M=150>
  </TIPO>
</PREÇO>
```

4.6 Modelo de Dados

Quando se fala em Banco de Dados, espera-se que exista um modelo que represente-o, uma vez que, um **Modelo** é a representação abstrata e simplificada de um sistema real, com a qual se pode explicar ou testar o seu comportamento, como um todo ou em partes, ver [40].

Nesta seção um Modelo para a representação do Banco de Dados Nebulosos *Aliança* será proposto utilizando uma extensão da UML (Unified Modeling Language), chamada de FUML (Linguagem de Modelagem Unificada Fuzzy).

4.6.1 FUML - Fuzzy UML

A intenção aqui não é apresentar uma representação nebulosa para todos os tipos de Diagramas da UML, ver [41], mas sim apenas para o Diagrama de Classes. Uma vez que dentre os diversos diagramas da UML, o Diagrama de Classes foi intencionalmente projetado para ser uma extensão do Modelo Entidade-Relacionamento, podendo então ser utilizado para modelar a estrutura lógica das tabelas que compõem o banco de dados.

Na literatura, existem propostas de Diagramas UML para representar dados nebulosos, como Ma [42] e Tseng e Chen [43].

Em [42], para a modelagem de Bancos de Dados Nebulosos, Ma utilizou o modelo Entidade-Relacionamento (Modelo ER) concebido por Peter Chen em 1976 [44] e também o Diagrama de Classes (UML), ver [41].

Em ambos os trabalhos, [42] e [43], destacam-se as entidades nebulosas (onde suas tuplas apresentavam um certo grau de pertinência nas tabelas) e seus relacionamentos nebulosos. No trabalho de Tseng e Chen [43], é apresentada uma aproximação para mapear as consultas em linguagem natural com semântica fuzzy em SQL padrão através de representações de diagrama de classes UML. Cada uma das propostas são específicas e atendem aos estudos de seus autores. Assim, foi necessário desenvolver um Diagrama de Classes Nebulosas próprio para o Banco de Dados Nebulosos *Aliança*.

Em *Aliança*, foi definido que todas as tabelas teriam apenas chaves primárias *crisp*, evitando assim redundância entre tuplas de uma mesma tabela. Com isto, são considerados apenas relacionamentos tradicionais entre tabelas. A necessidade de uma representação especial aparece na representação de atributos que podem receber valores nebulosos e suas correspondentes metainformações.

4.6.2 Diagrama de Classes Nebulosas

Como tem sido feito neste trabalho o Diagrama de Classes Nebulosas também será descrito através de um exemplo. O exemplo considerado será um banco de dados chamado *Antiquário* que contém as seguintes relações:

- **Tabela Carros Antigos:** Apresentada anteriormente, Tabela 4.2, armazena informações sobre carros antigos. Esta tabela possui os atributos **Idade**, **Preço** e **Eficiência** que aceitam tratamentos nebulosos, portanto possui uma BMN de mesmo nome associada a ela. A Figura 4.7 mostra a estrutura de diretórios BMN e os arquivos XML dos atributos nebulosos **Idade**, **Preço** e **Eficiência** foram mostrados anteriormente pelos exemplos 4.1, 4.3 e 4.8.
- **Tabela Proprietário:** É uma tabela tradicional que armazena informações sobre os proprietários dos veículos antigos e possui atributos que não aceitam tratamentos nebulosos.
- **Tabela Endereço:** É uma tabela tradicional que armazena informações sobre o endereço dos proprietários dos veículos antigos e possui atributos que não aceitam tratamentos nebulosos.

O Diagrama de Classes Nebulosas construído para ser o modelo de dados do Banco de Dados Nebulosos *Antiquário*, que segue a arquitetura *Aliança*, é apresentado pelo Figura 4.29.

A Tabela Carros Antigos está representada por uma classe que possui um esteriótipo $\langle\langle fuzzy \rangle\rangle$. Este esteriótipo indica que a esta tabela possui atributos nebulosos.

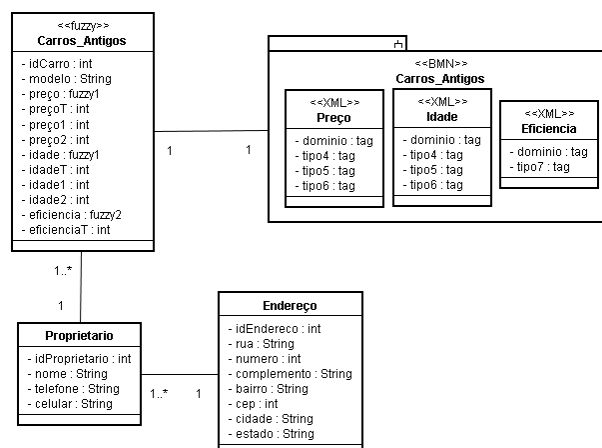


Figura 4.29: Modelo FUMML do Diagrama de Classes

Os atributos **Preço**, **Idade** e **Eficiência** possuem informações nebulosas associadas a eles na BMN através dos arquivos XML, e assim foram classificados como sendo do Tipo *fuzzy1* e *fuzzy2*. O que torna *fuzzy1* e *fuzzy2* diferentes são, na realidade, as tags que estão presentes nos arquivos XML associados. Atributos do Tipo *fuzzy1* são aqueles que aceitam dados dos Tipos Nebulosos de 0 a 6, enquanto que os atributos do Tipo *fuzzy2* são aqueles que aceitam os Tipos Nebulosos 0, 1, 2 e 7 (Etiquetas Lingüísticas com Similaridade), descritos em detalhes na Seção 4.2.1.

Toda tabela que apresenta o esteriótipo *<< fuzzy >>* possuirá arquivos XML na BMN associados a seus atributos nebulosos. Estes arquivos XML ficam armazenados em um diretório. Este diretório aparece no modelo representado por um subsistema.

As classes Proprietário e Endereço representam as tabelas Proprietário e Endereço que fazem parte do Banco de Dados *Antiquário* e são consideradas tabelas *crisp* por não possuírem atributos nebulosos, e portanto não necessitam de nenhum esteriótipo.

4.7 Implementação do Sistema *Aliança*

A fim de comprovar a viabilidade da nova arquitetura de banco de dados nebulosos *Aliança* um protótipo foi construído em uma parceria com o aluno de graduação Rafael Cavalcanti, ver [45].

Nesta seção será descrita a implementação deste protótipo do Sistema *Aliança*, juntamente com uma breve descrição das tecnologias utilizadas.

4.7.1 Arquitetura

A nova arquitetura foi apresentada pela Figura 4.1 e teve seus Módulos **SGBDR** (Sistema de Gerenciamento de Banco de Dados Relacionais), **BD** (Banco de Dados), **BMN** (Base de Metaconhecimento Nebuloso), **Servidor FSQL** (Servidor SQL Nebuloso) e **Interface com o usuário** descritos na Seção 4.2.

De forma semelhante, porém enfatizando a implementação, a Figura 4.30 mostra a arquitetura geral do Banco de Dados Nebulosos - *Aliança* juntamente com os principais módulos do sistema.

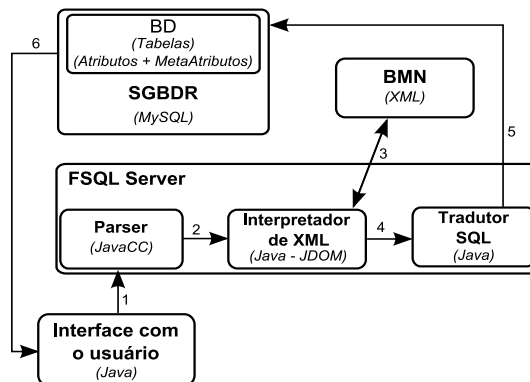


Figura 4.30: Arquitetura Geral do Banco de Dados Nebulosos *Aliança* - ênfase Implementação

O funcionamento do sistema *Aliança* pode ser explicado seguindo os 6 passos indicados na Figura 4.30.

1. O Módulo de *Interface com o usuário* recebe uma consulta elaborada em

- FSQL e a envia ao Servidor FSQL;
2. O Módulo *Parser* no Servidor FSQL recebe a consulta em FSQL e efetua uma análise léxica e sintática nesta consulta. Tal análise constata se a consulta foi corretamente elaborada e, em caso positivo, identifica os *tokens* existentes;
 3. Os *tokens* identificados disparam o Módulo *Interpretador de XML*, que acessa o Módulo *BMN* com o intuito de buscar as metainformações relativas aos *tokens*;
 4. As metainformações obtidas são enviadas ao Módulo *Tradutor SQL* que monta a consulta SQL clássica equivalente à consulta em FSQL submetida ao sistema;
 5. A consulta em SQL clássico é enviada ao SGBDR, que a executa e obtém o resultado;
 6. O resultado obtido é então apresentado pelo Módulo *Interface com o Usuário*.

4.7.2 Tecnologias Utilizadas

O Sistema de Gerenciamento de Banco de Dados Nebulosos *Aliança* foi desenvolvido usando as tecnologias apresentadas a seguir.

4.7.2.1 A Linguagem Java

Java é uma linguagem de programação orientada a objeto desenvolvida na década de 90. Desde seu lançamento, em maio de 1995, a plataforma Java foi adotada mais rapidamente do que qualquer outra linguagem de programação na história da computação. Java tornou-se popular pelo seu uso na Internet e hoje possui seu ambiente de execução presente em web browsers, mainframes, SOs, celulares, palmtops e cartões inteligentes, entre outros.

A linguagem Java foi projetada tendo em vista os seguintes objetivos:

- Orientação a objeto - Baseado no modelo de Smalltalk e Simula67;

- Portabilidade - Independência de plataforma - “write once run anywhere”;
- Recursos de Rede - Possui extensa biblioteca de rotinas que facilitam a cooperação com protocolos TCP/IP, como HTTP e FTP;
- Segurança - Pode executar programas via rede com restrições de execução;
- Bytecode interpretado, ao invés de compilado. Programas Java não são traduzidos para a linguagem de máquina como outras linguagens estaticamente compiladas e sim para uma representação intermediária, chamada de bytecodes. Os bytecodes são interpretados pela máquina virtual Java (JVM - Java Virtual Machine).

Além disso, podem-se destacar outras vantagens apresentadas pela linguagem:

- Sintaxe similar a Linguagem C/C++ e principalmente, a C#.
- Facilidades de Internacionalização - Suporta nativamente caracteres Unicode;
- Simplicidade na especificação, tanto da linguagem como do “ambiente” de execução (JVM);
- É distribuída com um vasto conjunto de bibliotecas (ou APIs);
- Possui facilidades para criação de programas distribuídos e multitarefa (múltiplas linhas de execução num mesmo programa);
- Desalocação de memória automática por processo de coletor de lixo;
- Carga Dinâmica de Código - Programas em Java são formados por uma coleção de classes armazenadas independentemente e que podem ser carregadas no momento de utilização.

Muitas fontes bibliográficas contendo maiores detalhes sobre a linguagem Java podem ser encontradas, como por exemplo [46], [47] e [48].

4.7.2.2 JavaCC

JavaCC (Java Compiler-Compiler) é uma ferramenta geradora de analisadores sintáticos que produz código Java, ver [49]. Constitui uma ferramenta poderosa, flexível e simples de usar, que gera um código compilado fácil de ser lido pelo programador.

JavaCC permite que uma determinada linguagem seja definida de maneira simples. Como saída produz o código-fonte de algumas classes Java que implementam os analisadores léxico e sintático para aquela linguagem. Apresenta também maneiras de incluir, junto à definição da linguagem, código Java para, por exemplo, construir-se a árvore de derivação do programa analisado.

JavaCC aproveita todas as características da linguagem Java para prover facilidades na construção de analisadores sintáticos. Principalmente o fato de Java ser orientada a objetos torna o uso de JavaCC proveitoso, facilitando a geração e adaptação dos códigos que avaliam os nós da árvore sintática. Uma outra característica da linguagem Java, o tratamento de exceções, torna o gerenciamento de erros sintáticos de mais fácil implementação e leitura.

O JavaCC define uma linguagem própria para descrição, em um único arquivo, do analisador léxico e do analisador sintático. No caso do analisador léxico, esta linguagem permite que cada *token* seja definido na forma de uma expressão regular.

As declarações para o analisador sintático correspondem às produções da gramática que se deseja implementar. Seguindo a filosofia da análise descendente recursiva, as declarações dos não terminais são parecidas com a declaração de um método. A diferença é que o corpo do não terminal possui as produções descritas através de seqüências de *tokens* e estruturas de repetição, escolhas ou opcionais.

É possível definir Análise Léxica e Sintática como

- **Análise léxica** é o processo de analisar a entrada de linhas de caracteres (tal como o código-fonte de um programa de computador) e produzir uma seqüência de símbolos chamado “símbolos léxicos” (*lexical tokens*), ou somente “símbolos” (*tokens*), que podem ser manipulados mais facilmente por

um *parser* (leitor de saída).

A Análise Léxica é a forma de verificar determinado alfabeto. Quando analisamos uma palavra, podemos definir através da análise léxica se existe ou não algum caractere que não faz parte do nosso alfabeto, ou um alfabeto inventado por nós. O analisador léxico é a primeira etapa de um compilador, logo após virá a análise sintática.

- **Análise sintática** (também conhecido pelo termo em inglês *parsing*) é o processo de analisar uma seqüência de entrada (lida de um arquivo de computador ou do teclado, por exemplo) para determinar sua estrutura gramatical segundo uma determinada gramática formal. Essa análise faz parte de um compilador, junto com a análise léxica e análise semântica.

A análise sintática transforma um texto na entrada em uma estrutura de dados, em geral uma árvore, o que é conveniente para processamento posterior e captura a hierarquia implícita desta entrada. Através da análise léxica é obtido um grupo de *tokens*, para que o analisador sintático use um conjunto de regras para construir uma árvore sintática da estrutura.

Em termos práticos, pode também ser usada para decompor um texto em unidades estruturais para serem organizadas dentro de um bloco, por exemplo.

4.7.2.3 API JDOM

JDOM (Java Document Object Model) é uma biblioteca cujo objetivo é processar arquivos em XML, ver [50].

JDOM é específico para a plataforma JAVA. A API usa a linguagem Java tornando os valores do texto sempre acessíveis como String. JDOM também faz uso das classes de coleção da plataforma Java 2, como List e Iterator, disponibilizando um ambiente rico para programadores familiarizados com a linguagem Java.

Não há hierarquia. No JDOM, um elemento XML é uma instância de Element, um atributo XML é uma instância de Attribute, e um documento XML é ele

mesmo uma instância de Document. Porque todos estes componentes representam conceitos diferentes em XML, e sempre existem referências a eles e nunca existirá um nó não especificado.

Devido ao fato de objetos JDOM serem instâncias diretas de classes como Document, Element, e Attribute, criar uma instância é tão fácil quanto usar o novo operador na linguagem Java. O que significa que não há interfaces para configurar – JDOM está pronto para ser usado direto do jar.

4.7.2.4 MySQL

MySQL foi o banco de dados escolhido por ser completo, robusto, rápido e ser de fácil acesso pela linguagem Java. Uma de suas peculiaridades são suas licenças para uso gratuito, tanto para fins estudantis como para realização de negócios.

O MySQL se tornou o mais popular banco de dados open source do mundo porque possui consistência, alta performance, confiabilidade e é de fácil uso.

O MySQL funciona em mais de 20 plataformas, incluindo Linux, Windows, HP-UX, AIX, Netware, dando a ele flexibilidade e controle. Maiores detalhes em [51].

4.7.3 Exemplos

Os exemplos apresentados aqui irão considerar a Tabela 4.2, sua representação interna Tabela 4.4, e a BMN contendo os arquivos Idade.xml, Preço.xml e Eficiencia.xml, apresentados nos exemplos 4.1, 4.8, 4.3 respectivamente.

Exemplo 4.7 *Selecione o modelo e o preço dos carros antigos que possuem preço alto (com grau 0.8).*

Os passos, apresentados na Figura 4.30, serão seguidos.

1. O Módulo de *Interface com o usuário* recebe uma consulta elaborada em FSQL e a envia ao Servidor FSQL;

```

SELECT    Id_Carro, Modelo, Preço
FROM      Carros_Antigos
WHERE     Preço FEQ Alto 0.8

```

2. O Módulo *Parser* no Servidor FSQL recebe a consulta em FSQL e efetua uma análise léxica e sintática nesta consulta. Tal análise constata se a consulta foi corretamente elaborada e, em caso positivo, identifica os *tokens* existentes;

```

SELECT    Id_carro, Modelo, Preço
FROM      Carros_Antigos
WHERE     Preço FEQ Alto 0.8

```

As palavras SELECT, FROM e WHERE foram identificadas como sendo palavras reservadas e assim os atributos e tabelas envolvidas também foram reconhecidos. O operador FEQ (Fuzzy Equal), após identificado, sofreu um tratamento especial, pois foi preciso identificar o dado nebuloso que veio em seqüência (Etiqueta - Alto) juntamente com seu tipo (Tipo 4). O grau de satisfação da consulta foi identificado (0.8).

3. Os *tokens* identificados dispararam o Módulo *Interpretador de XML*, que acessa o Módulo *BMN* com o intuito de buscar as metainformações relativas aos *tokens*;

```

SELECT    Id_carro, Modelo, Preço
FROM      Carros_Antigos      grau
WHERE     Preço FEQ Alto 0.8

```

Alto = <ALTO A=1200 B=1300 C=1800 D=4000 />

Os parâmetros que definem a Etiqueta Lingüística Alto com função de inclusão trapezoidal são encontrados (ALTO A=1200 B=1300 C=1800 D=4000).

4. As metainformações obtidas são enviadas ao Módulo *Tradutor SQL* que monta a consulta SQL clássica equivalente à consulta em FSQL submetida ao sistema;

Afim de tornar o processo mais eficiente, visões são criadas para cada um dos Tipos de Dados Nebulosos.

```
CREATE VIEW Visao0 AS
  SELECT *
  FROM Carros_Antigos
  WHERE Preçot = 0
CREATE VIEW Visao1 AS
  SELECT *
  FROM Carros_Antigos
  WHERE Preçot = 1
CREATE VIEW Visao4 AS
  SELECT *
  FROM Carros_Antigos
  WHERE Preçot = 4
CREATE VIEW Visao5 AS
  SELECT *
  FROM Carros_Antigos
  WHERE Preçot = 5
CREATE VIEW Visao6 AS
  SELECT *
  FROM Carros_Antigos
  WHERE Preçot = 6
```

Com o tipo nebuloso da Etiqueta Lingüística Alto identificado como sendo Tipo 4, as classes e métodos que tratam da comparação dos demais Tipos com o Tipo 4 são chamados e um SQL clássico equivalente é montado.

A comparação do Tipo 4 com ele mesmo foi tratada como mostrado na seção 4.5.2 e algumas características foram acrescentadas no arquivo **Preço.xml**.

Para a montagem do SQL clássico, apenas a Etiqueta Lingüística “Alto” foi considerada, uma vez que as demais etiquetas não atendem a consulta devido ao grau de igualdade com a Etiqueta “Alto” ser menor que 0.8.

Tipo 0 com o Tipo 4	SELECT	Id_carro, Modelo, Preço, 1 as grau
	FROM	Visao0
	WHERE	B <= Preço1 AND Preço1 <= C
Tipo 0 com o Tipo 4	UNION	
	SELECT	Id_carro, Modelo, Preço, ((Preço1 - A)/(B-A)) as grau
	FROM	Visao0
Tipo 0 com o Tipo 4	WHERE	A <= Preço1 AND Preço1 <= B AND ((Preço1 - A)/(B-A)) >= 0.8
	UNION	
	SELECT	Id_carro, Modelo, Preço, ((D - Preço1)/(D-C)) as grau
Tipo 0 com o Tipo 4	FROM	Visao0
	WHERE	Preço1 >= C AND Preço1 <= D AND ((D - Preço1)/(D-C)) >= 0.8
	UNION	
Tipo 1 com o Tipo 4	SELECT	Id_carro, Modelo, Preço, 1 as grau
	FROM	Visao1
	UNION	
Tipo 4 com o Tipo 4	SELECT	Id_carro, Modelo, Preço, 1 as grau
	FROM	Visao4
	WHERE	Preço = '\$Alto'
Tipo 4 com o Tipo 4	UNION	
	SELECT	Id_carro, Modelo, Preço, 1 as grau
	FROM	Visao5
Tipo 5 com o Tipo 4	WHERE	Preço2 >= B AND Preço1 < C
	UNION	
	SELECT	Id_carro, Modelo, Preço, ((Preço2 - A)/(B-A)) as grau
Tipo 5 com o Tipo 4	FROM	Visao5
	WHERE	A <= Preço2 AND Preço2 < B AND ((Preço2 - A)/(B-A)) >= 0.8
	UNION	
Tipo 5 com o Tipo 4	SELECT	Id_carro, Modelo, Preço, ((D - Preço1)/(D-C)) as grau
	FROM	Visao5
	WHERE	Preço2 >= C AND Preço1 < D AND ((D - Preço1)/(D-C)) >= 0.8
Tipo 5 com o Tipo 4	UNION	
	SELECT	Id_carro, Modelo, Preço, 1 as grau
	FROM	Visao6
Tipo 6 com o Tipo 4	WHERE	B <= Preço1 AND Preço1 <= C
	UNION	
	SELECT	Id_carro, Modelo, Preço, ((Preço1 + Preço2 - A)/(B - A + Preço2)) as grau
Tipo 6 com o Tipo 4	FROM	Visao6
	WHERE	A < (Preço1 + Preço2) AND Preço1 < B AND ((Preço1 + Preço2 - A)/(B - A + Preço2)) >= 0.8
	UNION	
Tipo 6 com o Tipo 4	SELECT	Id_carro, Modelo, Preço, ((D - Preço1 + Preço2)/(Preço2 + D - C)) as grau
	FROM	Visao6
	WHERE	Preço1 > C AND (Preço1 + Preço2) <= D AND ((D - Preço1 + Preço2)/(Preço2 + D - C)) >= 0.8

5. A consulta em SQL clássico é enviada ao SGBDR, que a executa e obtém o resultado;

Tabela 4.8: Carros Antigos - Resultado da Consulta

Id_Carro	Modelo	Preço
002	Alfa Romeo Convertible	35000
006	Porsche Spyder 550	Alto

O carro de com Id.Carro = 005, por exemplo, não aparece na resposta porque se iguala a Etiqueta “Alto” com o grau 0.67 e a consulta solicita grau pelo menos igual a 0.8.

6. O resultado obtido é então apresentado pelo Módulo *Interface com o Usuário*.

Exemplo 4.8 *Selecione o modelo e o preço dos carros antigos que possuem preço alto (com grau 0.8) e eficiência regular (com grau 0.5).*

Diferentemente do exemplo 4.7, este exemplo trata de uma consulta efetuada em uma tabela formada por duas condições nebulosas. O que torna a análise um pouco mais complexa.

1. O Módulo de **Interface com o usuário** recebe uma consulta elaborada em FSQL e a envia ao Servidor FSQL; Assim como apresentado no exemplo anterior

```
SELECT  Id_Carro, Modelo, Preço
FROM    Carros_Antigos
WHERE   Preço FEQ Alto 0.8 AND Eficiencia FEQ Regular 0.5
```

2. O Módulo **Parser** no Servidor FSQL recebe a consulta em FSQL e efetua uma análise léxica e sintática nesta consulta. Tal análise constata se a consulta foi corretamente elaborada e, em caso positivo, identifica os *tokens* existentes;


```

SELECT   Id_Carro, Modelo, Preço
FROM     Carros_Antigos
WHERE    Preço FEQ Alto 0.8 AND Eficiencia FEQ Regular 0.5

```

A palavra AND foi identificado e com isso a consulta foi quebrada em duas subconsultas.

```

SELECT   Id_Carro, Modelo, Preço
FROM     Carros_Antigos
WHERE    Preço FEQ Alto 0.8

```

Junção Natural

```

SELECT   Id_Carro, Modelo, Preço
FROM     Carros_Antigos
WHERE    Eficiencia FEQ Regular 0.5

```

Para cada consulta as palavras SELECT, FROM e WHERE foram identificadas como sendo palavras reservadas e assim os atributos envolvidos também foram reconhecidos.

3. Daqui por diante, o processo ocorre da mesma forma que apresentado no exemplo 4.7 para cada consulta separadamente. Ao final uma junção natural é executada e o resultado apresentado.

Este raciocínio se estende para n condições nebulosas em um consulta para uma ou mais tabelas.

Assim, para uma consulta nebulosa com n condições teremos n subconsultas e $n - 1$ **junções naturais** ao final quando se tratar de AND e $n - 1$ **uniões** quando se tratar de OR.

O protótipo construído por Rafael [45] efetua consultas utilizando o operador estendido FEQ (*Fuzzy Equal*) contemplando no máximo duas condições nebulosas. Comprovando a viabilidade da nova arquitetura para bancos de dados nebulosos *Aliança*.

Capítulo 5

Conclusões e Trabalhos Futuros

Os resultados apresentados nesta tese podem ser resumidos nos seguintes pontos:

- Os tipos de dados com que o Banco de Dados Nebulosos *Aliança* opera é bastante extenso, oferecendo amplas possibilidades para a representação e manipulação de dados imprecisos.
- Organiza de forma consistente a informação nebulosa.
- Apresenta uma BMN (Base de Metaconhecimento Nebuloso) organizada, de fácil compreensão e manutenção. O que simplifica a representação e o armazenamento do conhecimento nebuloso em bancos de dados, eliminando complicadas tabelas usadas em sistemas pesquisados previamente.
- Como o uso de XML está se tornando muito comum em diversas áreas, o fato de *Aliança* organizar sua BMN em arquivos XML o torna de fácil disseminação para uso e aplicação.
- O Modelo *Aliança* apresenta grande flexibilidade para o tratamento da informação imprecisa. Tornando as consultas mais próximas ao raciocínio humano.
- É factível a construção de sistemas relacionais de bancos de dados nebulosos baseados neste modelo, o que foi comprovado pela construção do protótipo.

Trabalhos Futuros

- O protótipo construído possui o intuito de comprovar a viabilidade da nova Arquitetura proposta. Para se obter um sistema completo como forma de produto serão necessários maiores investimentos.
- Um projeto para a construção de um sistema completo baseado na nova arquitetura apresentada neste trabalho terá início em 2009. Uma interface amigável para a construção dos XML contidos na BMN está prevista.
- A teoria estudada para a construção da nova arquitetura *Aliança* pode ser ainda estendida para envolver as DDL (Linguagem de Definição de Dados) e comandos não básicos da DML (Linguagem de Manipulação de Dados) como por exemplo a Divisão.
- A nova arquitetura proposta não está restrita ao modelo relacional e pode ser estendida a outros tipos de bancos de dados como aqueles orientados a objetos. Isto se deve ao fato da BMN não estar armazenadas em tabelas e sim em documentos XML externos ao Banco de Dados Relacional. Algum estudo seria necessário, mas comparando os dois caminhos para a implementação do banco de dados, relacional e orientado a objetos, é possível observar que, enquanto o relacional é baseado em *tuplas*, o modelo OO está diretamente relacionado a *objetos* e suas persistências. Atributos nebulosos seriam necessários também para os objetos de forma similar ao apresentado para o modelo relacional. Como a OQL (Object Query Language) é similar à SQL (Structure Query Language), a transformação de uma consulta nebulosa em uma consulta para objetos seguiria a mesma metodologia apresentada.

Referências Bibliográficas

- [1] MEDINA, J. M., *Bases de Datos Relacionales Difusas. Modelo Teórico y Aspectos de su Implementación*, Tese de D.Sc., Universidade de Granada, Granada, Andaluzia, Espanha, 1994
- [2] GALINDO, J. G., *Tratamiento de la Imprecisión en Bases de Datos Relacionales: Extensión del Modelo y Adaptación de los SGBD Actuales*, Tese de D.Sc., Universidade de Granada, Granada, Andaluzia, Espanha, 1999
- [3] ZADEH, L. A., “Similarity Relations and Fuzzy Orderings”, *Information Sciences*, v.3, n.2, pp. 177-200, 1971
- [4] ZADEH, L. A., “Fuzzy Sets as a Basis for a Theory of Possibility”, *Fuzzy Sets and Systems*, v.1, pp. 3-28, 1978
- [5] ZADEH, L. A., “Fuzzy Sets”, *Information Sciences*, v.8, n.3, pp. 338-353, 1965
- [6] FÁVARO, S., FILHO, O., K., *Noções de Lógica e Matemática Básica*, 1 ed. , Editora Ciência Moderna, 2005
- [7] SOUZA, J. N., *Lógica para Ciência da Computação*, 1 ed. , Editora Campus/Elsevier, 2002
- [8] JANG, R. J., SUN, C., MIZUTANI, E., *Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence*, USA, Editora Prentice-Hall, 1997

- [9] YEN, J.; LANGARI, R.; *Fuzzy Logic - Intelligence, Control, and Information*, 1. ed. , Editora Prentice-Hall, 1999
- [10] NGUYEN, H. T., WALKER, E. A. *A First Course in Fuzzy Logic*, USA, 3 ed. , Editora Chapman & Hall/CRC, 2005
- [11] BERGMANN, M., *An Introduction to Many-Valued and Fuzzy Logic: Semantics, Algebras, and Derivation Systems*, USA, 1. ed. , Editora Cambridge University Press, 2008
- [12] CODD, E. F., “Relational Databases: A Practical Foundation for Productivity”, *Comm. ACM*, v.25, n.2, pp. 109-117, 1982
- [13] KORTH, H. F., SILBERSCHATZ, A., *Sistema de Bancos de Dados*, 2ed. rev. São Paulo, Editora McGraw-Hill Ltda, 1995
- [14] CODD, E. F., “A Relational Model of Data for Large Shared Data Banks”, *Communications of the ACM*, v.13, n.6, pp. 377-387, 1970
- [15] YOUSSEFI, K., WHYTE, N., UBELL, M., et al., *INGRES Reference Manual* 6ed. University of California, Berkeley, California, Electronics Research Lab., 1977
- [16] ASTRAHAN, M. M., BLASGEN, M. W., CHAMBERLIN, D. D., et al., “System R: Relational Approach to Database Management”, *ACM Transactions Database Systems*, v.1, n.2, pp. 97-137, 1976
- [17] CHEN, P.P., “The Entity-Relationship Model: Toward a Unified View of Data”, *ACM Transactions Database Systems*, v.1, n.1, pp. 9-36, 1976
- [18] HAROLD, E. R., *XML Bible*, Foster City, Editora IDG Books Worldwide, 1999
- [19] GALINDO, J. G., ARANDA, M. C., CARO, J. L., GUEVARA, A., AGUAYO, A. “Applying Fuzzy Databases and FSQL to the Management of Rural Accommodation”, *Turism Management*, v.24, pp. 457-463, 2003

- [20] TUROWSKI, K., WENG, U., “Representing and processing fuzzy information - an XML-based approach”, *Knowledge-Based Systems*, v.15, n.1-2, pp. 67-75, 2002
- [21] GAURAV, A., ALHAJJ, R., “Incorporating Fuzziness in XML and Mapping Fuzzy Relational Data into Fuzzy XML”, *In: Symposium on Applied Computing ACM*, pp. 456-460, Dijon, France, 2006
- [22] TASHIRO, H., OHKI, N., NOMURA, T., et al. “Managing Subjective Information in Fuzzy Database System”, *IN ACM Conference on Computer Science*, pp. 156-161, New York, NY, USA, 1993
- [23] YAZICI, A., SOYSAL, A., BUCLES, B. P., et al., “Uncertainty in a Nested Relational Database Model”, *Data & Knowledge Engineering*, v.30, pp. 275-301, 1999
- [24] CHAUDHRY, N., MOYNE, J., RUNDENSTEINER, E. A., “An Extended Database Design Methodology for Uncertain Data Management”, *Information Sciences*, v.121, pp. 83-112, 1999
- [25] KACPRZYK, J., ZADROZNY, S., “Computing with words in intelligent database querying: standalone and Internet-based applications”, *Information Sciences*, v.134, pp. 71-109, 2001
- [26] YANG, Q., ZHANG, W., LIU, C., et al., “Efficient Processing of Nested Fuzzy SQL Queries in a Fuzzy Database”, *IEEE Transactions on Knowledge and Data Engineering*, v.13, n.6, pp. 884-901 2001
- [27] YAGER, R. R. “Querying Databases Containing Multivalued Attributes Using Veristic Variables”, *Fuzzy Sets and Systems*, v.129, pp. 163-185, 2002
- [28] RASCHIA, G., MOUADDIB, N., “SAINTETIQ: a Fuzzy Set-based Approach to Database Summarization”, *Fuzzy Sets and Systems*, v.129, pp. 137-162, 2002

- [29] CODD, E. F., “Extending the Database Relational Model to Capture More Meaning”, *ACM Transactions on Database Systems*, v.4, n.4, pp. 262-296, 1979
- [30] CODD, E. F., “Missing Information (Applicable and Inapplicable) in Relational Databases”, *ACM SIGMOD Record*, v.15, n.4, pp. 53-78, 1986
- [31] CODD, E. F., “More Commentary on Missing Information in Relational Databases (Applicable and Inapplicable Information)”, *ACM SIGMOD Record*, v.16, n.1, pp. 42-50, 1987
- [32] BUCKLES, B. P., PETRY, F. E., “A Fuzzy Representation of Data for Relational Databases”, *Fuzzy Sets and Systems*, v.7, n.3, pp. 213-226, 1982
- [33] BUCKLES, B. P., PETRY, F. E., “Extending the Fuzzy Databases with Fuzzy Numbers”, *Information Sciences*, v.34, pp. 145-155, 1984
- [34] BUCKLES, B. P.; PETRY, F. E.; “A Domain Calculus For Fuzzy Relational”, *Fuzzy Sets and Systems*, 29, pp. 327-340, 1989
- [35] BUCKLES, B. P.; PETRY, F. E.; *Fuzzy Databases in New Era*, ACM, 0-89791-658-1, pp. 497-502 (1995)
- [36] PRADE, H., TESTEMALE, C., “Generalizing Database Relational Algebra for the Treatment of Incomplete or Uncertain Information and Vague Queries”, *Information Sciences*, v.34, n.2, pp. 115-143, 1984
- [37] UMANO, M.; FUKAMI, S.; MIZUMOTO, M.; TANAKA, K.; “Retrieval Processing from Fuzzy Databases”, *Technical Reports of IECE of Japan*, v.80, n.204, pp. 45-54, 1980
- [38] UMANO, M., FUKAMI, S., “Fuzzy Relational Algebra for Possibility-Distribution-Fuzzy-Relational Model of Fuzzy Data”, *Journal of Intelligent Information Systems*, v.3, n.1, pp. 7-28, 1994

- [39] ZEMANKOVA, M., KANDEL, A., “Implementing Imprecision in Information Systems”, *Information Sciences*, v.37, n.1-3, pp. 107-141, 1985
- [40] ELMASRI, R. E., NAVATHE, S., *Sistemas de Banco de Dados*, 4 ed. , Editora Addison-Wesley, 2005
- [41] GUEDES, G. T. A., *UML - Uma Abordagem Prática*, 2 ed. São Paulo, Novatec Editora, 2006
- [42] MA, Z., *Fuzzy Database Modeling with XML*, Advances in Database Systems Volume 29, Universite de Sherbrooke - Canada, Editora Springer, 2005
- [43] TSENG F. S. C., CHEN C., “Extending the UML concepts to transform natural language queries with fuzzy semantics into SQL*”, *Information and Software Technology*, v.48, n.9, pp. 901–914, 2006
- [44] CHEN P. P., “The entity-relationship model—toward a unified view of data”, *ACM Transactions on Database Systems*, v.1, n.1, p.9-37, 1976
- [45] CAVALCANTI, R. S. T., *Implementando um Sistema de Gerenciamento de Banco de Dados Nebulosos*, Projeto Final de Curso, DCC/UFRJ, Rio de Janeiro, RJ, Brasil, 2007
- [46] DEITEL, H. M., DEITEL, P. J., *Java Como Programar*, 6 ed. , Editora Prentice-Hall, 2005
- [47] SIERRA, K., BATES, B., *Java - Use a Cabeça*, 2 ed. , Editora Alta Books, 2007
- [48] SIERRA, K., BATES, B., *SCJP: Certificação Sun para Programador Java 5 - Guia de Estudo*, 2 ed. , Editora Alta Books, 2006
- [49] DELAMARO, M. E., *Como Construir um Compilador: Utilizando Ferramentas Java*, 1 ed. São Paulo, Novatec Editora, 2004
- [50] HAROLD, E. R., *Processing XML with Java: A Guide to SAX, DOM, JDOM, JAXP, and TrAX*, Editora Person, 2002

- [51] RANGEL, A., *MYSQL: Projeto, Modelagem e Desenvolvimento de Bancos de Dados*, 1 ed. , AltaBooks, 2004

Apêndice A

Publicação

Apresentação do artigo *Aliança: A proposal for a fuzzy database architecture incorporating XML* que será publicado em janeiro de 2009 em FUZZY SETS AND SYSTEMS - An International Journal in Information Science and Engineering Official Publication of the International Fuzzy Systems Association (IFSA).



Aliança: A proposal for a fuzzy database architecture incorporating XML

R.D. Rodrigues*, A.J.O. Cruz, R.T. Cavalcante

*Instituto de Matemática, Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro, CCMN - NCE, 21945-970
Rio de Janeiro, RJ, Brazil*

Received 18 May 2007; received in revised form 8 May 2008; accepted 6 June 2008
Available online 12 June 2008

Abstract

This paper presents a new fuzzy database architecture called *Aliança*. Fuzzy databases usually store information and its associated meta-information in order to add context to the whole range of different types that can be stored in these databases. However, the fuzzy database systems became very complex and difficult to maintain because fuzzy concepts are very difficult to represent using traditional database representation forms. *Aliança*, however, introduces a new way to represent this fuzzy meta-information base, which greatly simplifies data management tasks. The main advantages of these new representation are the easyness of understanding, implementation, use and support. This new meta-information base organizes information using the XML format, which adds the extra advantage of portability. FSQL assumes a conventional database and adds external tools to handle fuzzy data and is based on similarity relations and the theory of possibility introduced by Zadeh.

© 2008 Elsevier B.V. All rights reserved.

Keywords: Fuzzy databases; Information retrieval; Fuzzy meta-knowledge base; XML; Fuzzy database architecture

Contents

1. Introduction	270
2. Architecture overview	270
3. Representation of Vague Knowledge in <i>Aliança</i>	271
4. Fuzzy meta-knowledge base	273
4.1. Structure of the FMB	274
5. FSQL Server	278
5.1. FSQL—fuzzy SQL	278
6. Conclusions	279
References	279

* Corresponding author. Tel.: +55 21 27043272.

E-mail address: raquel.defelippo@gmail.com (R.D. Rodrigues).

1. Introduction

Traditional databases were designed for efficient storage, safe modification and correct recuperation of large samples of precisely defined data. These databases try to model real world data using few and precise structures, but unfortunately we are surrounded by uncertainty and imprecise information. Human beings manipulate very well that kind of information and, in fact, frequently we take decisions based on these pieces of information. We reason with vague terms such as “The price is high” or “The car is going too fast”. These facts are true (or false) only to some extent and computers may not process them. Fuzzy logic can be applied to extend computer capacities and provide them with tools to deal with such kind of information.

Some of the most important and common operations in databases are queries for information. For example, one might ask “*Who are the people who are more than 20 and less than 30 years old?*”. In many occasions it would be more important to the solution of a problem, or reasonable given the information available, if we could ask the same question as a human would do: “*Who are the young people?*”. The idea of “*young*” hints that the user is not asking for persons 90 years old and this may not be important or the exact age is not available for all people. The label “*young*” in a traditional database would create problems because usually only precise numerical dates are considered. So, this simple piece of information would need special pre-processing depending on the type of the answers required. Besides, the simple use of the word “*young*” will not improve the usability of the database because the semantic of the word “*young*” is not clear from the information stored in the database. So, we would not know if a person, who is 25 years old, is young or not.

In order to allow treatment of this kind of information in an efficient way, fuzzy databases were developed. These new types of databases can store, handle and respond to queries about vague and precise information in a very flexible way.

Fuzzy logic techniques have been applied to database and information retrieval areas for years. However, one of the first proposals for treatment of imprecise information on databases was presented by Codd [7] and further developed in [9,10], but the model did not use fuzzy logic. It proposed the use of the value NULL to indicate that an attribute can be any value of the domain. The model presented by Buckles–Petry [2,3] used the similarity measure defined by Zadeh [35]. The proposal of Prade–Testemale [22] went further, allowing attributes to receive fuzzy values. Attributes of precise and partial (imprecise and unknown) values were represented by the possibility distribution proposed by Zadeh [35].

One of the first relational database models was presented by Umano and Fukami [28]. This proposal also used possibility distributions to represent knowledge about the information in a way similar to the Prade–Testemale model. The model presented by Zemankova–Kaendel [37] created a structure to represent imprecise information and to manipulate uncertainty or vagueness in the query language. This proposal uses a new measure called *certainty measure*. The model GEFRED (a GEneralized model Fuzzy RELational Databases) described in [20] and improved by Galindo [11] is a fusion of main proposals that existed at the time to manipulate and store imprecise information. Therefore, this model [13,12,21] presents the advantage of incorporating all these previous ideas within its framework.

Turowski and Weng [25] presented a formal syntax, based on DTDs, to describe fuzzy data types used by business systems. The work of Gaurav and Alhadjj [14] describes how to map data from a fuzzy relational database into an XML document. Their goal was limited to publishing fuzzy data on the Web as XML documents.

This work presents a new fuzzy database architecture called *Aliança* (Alliance). This name represents the fact that the system is the union of fuzzy logic techniques, a database relational management system and a fuzzy meta-knowledge database defined in XML [15], in order to handle and represent vague information.

This paper is organized as follows: the architecture of new fuzzy relational database model *Aliança* is presented in Section 2. The representation of Vague Knowledge in *Aliança* is presented in Section 3. The fuzzy meta-knowledge base (FMB) is presented in Section 4 with an example. Section 5 presents the FSQL (*fuzzy structured query language*) Server. And finally, Section 6 presents the conclusions.

2. Architecture overview

In this section, we present a new fuzzy database architecture *Aliança* and discuss its basic elements and their relationships. The main objective of this proposal is to store and handle efficiently imprecise information using a

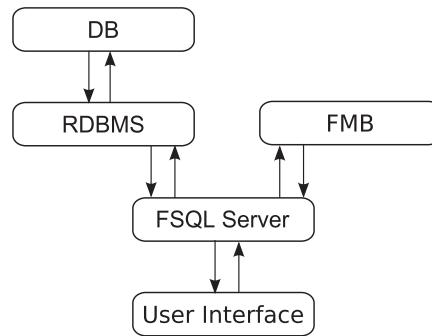


Fig. 1. General fuzzy database architecture *Aliança*.

classic database management system. The system uses a modified SQL language in order to allow the handling of the information stored in the database. This SQL version used in *Aliança* is called FSQL.

Fig. 1 shows the general architecture of the fuzzy database architecture *Aliança*. The main modules of the system are:

- **RDBMS** (relational database management system): This is a traditional relational database manager. Therefore, all fuzzy operations, or the ones that involve fuzzy data, are translated by the FSQL Server module into classical operations, so that the RDBMS can process them. Differently from previous proposals, for example, FIRST (in GEFRED) [11], in *Aliança* the RDBMS does not have any direct relationship with the FMB.
- **DB** (database): The information about the application is stored in database tables like in all traditional relational database. However, *Aliança* allows the storage of data about fuzzy information in its tables. This extra information is stored using a set of hidden attributes that define all relevant characteristics of the fuzzy data. This information is stored side by side with the traditional data formats and it is transparent to the end user.
- **FMB** (fuzzy meta-knowledge base): The information necessary to define and describe data of fuzzy nature is stored in the FMB. This information is organized in XML format and only the FSQL Server can access it.
- **FSQL Server**: FSQL Server is the main part of the system, because it is responsible for the relationship between the FMB and the RDBMS. One of its objectives is to transform FSQL information in traditional SQL information in order to permit the database management to process it and return an answer. This server was developed in Java.
- **User's Interface**: User's Interface is a program that allows the communication between the users and the FSQL Server.

It is important to note that the proposed architecture is not restricted to the relational model and can be easily extended to an object-oriented database. This is mainly due to the fact that the FMB is not stored on tables but on a text based XML document that is on a level external to the relational database.

Comparing the two ways used to implement databases, relational and object-oriented, we observe that, while the relational model is based on tuples, the OO model deals directly with objects and their persistency. Therefore, it would be simple to allow objects to receive a treatment similar to the one presented here. In addition to the fuzzy attribute of the object it would be necessary to add another two attributes in a very similar way as it will be discussed for relational databases in Section 4.1.

Besides that the structure of the Object Query Language (OQL) is similar to the structure of SQL and, therefore, the transformation of a fuzzy query into an object query would follow the same methodology presented in Section 5.1.

3. Representation of Vague Knowledge in *Aliança*

In this section we present the different types of information that can be stored in the database and its forms of representation. *Aliança* accepts eight different types of data. This is a very rich set of types and cover a wide range of data types. Each one of these types receives a numeric label from 0 to 7.

- **Crisp data**: Crisp data are the usual precise data handled by the traditional databases. Crisp data may get the same treatment as the imprecise data. This kind of data is classified as *type 0* and it does not need any additional information added to the FMB. Strings, real and natural numbers, dates are examples of usual crisp data formats.

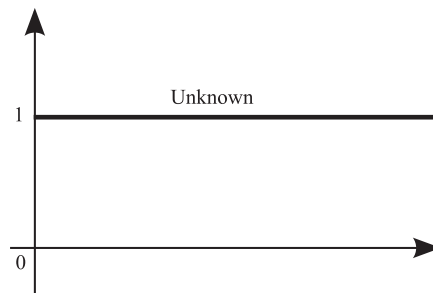


Fig. 2. Possibility distribution for a type Unknown.

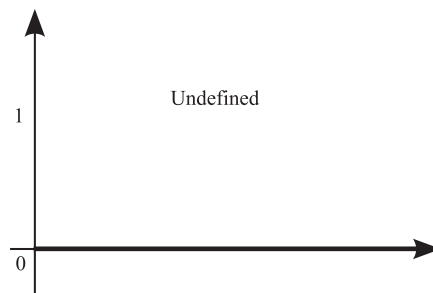


Fig. 3. Possibility distribution for a type Undefined.

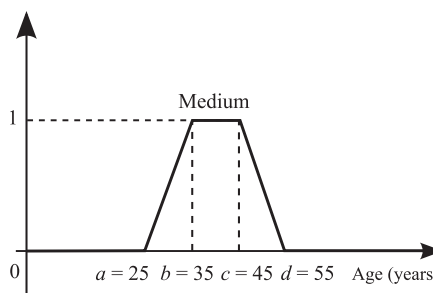


Fig. 4. An example of linguistic label for the concept “Medium”.

- *Unknown (but applicable)*: An attribute gets the value Unknown when it may receive any value from its domain, but it is not possible to define exactly which value. This kind of data is classified as *type 1*. The type Unknown is represented using the possibility distribution, $\{1/u, \forall u \in U\}$, where U is the domain. Fig. 2 shows this distribution.
- *Undefined (not applicable)*: An attribute gets the value Undefined when none of the values from the domain is applicable. This kind of data is classified as *type 2*. The type Undefined is represented using the possibility distribution, $\{0/u, \forall u \in U\}$, where U is the domain. Fig. 3 shows this distribution.
- *Null (absolute ignorance)*: An attribute gets the value Null when no information about it is available, either when we do not know (Unknown) or when it is not applicable (Undefined). This kind of data is of *type 3*.
- *Linguistic label with a possibility distribution*: When an attribute is associated to a vague value it receives a linguistic label with a possibility distribution. This kind of data is of *type 4* and it has an associated trapezoidal possibility distribution whose definition is stored in the FMB. Fig. 4 shows an example of a linguistic label. Trapezes are very used in fuzzy systems to represent vague values.
- *Possibility interval $[m, n]$* : This kind of data is of *type 5* and it is associated to an interval possibility distribution. Fig. 5 shows an example of possibility interval. This kind of data also needs additional data stored in the FMB.
- *Approximate value (approximately d)*: If the value d is in the domain, the vague concept *approximately d* is defined by a triangular possibility distribution defined around d with a *margin a* as shown in Fig. 6. This is the *type 6* kind of data and it also needs additional data stored in the FMB to define the margin used.

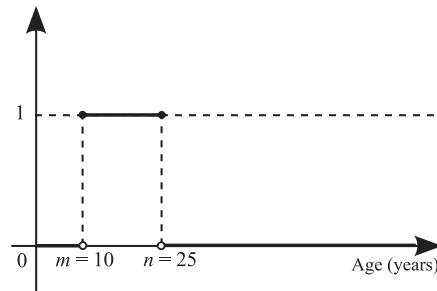


Fig. 5. An example of possibility interval.

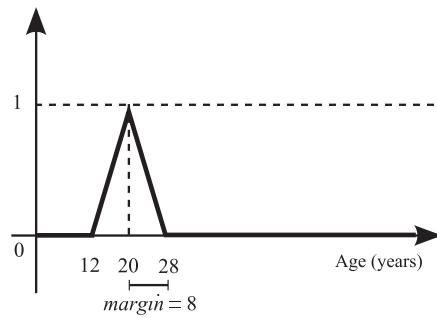


Fig. 6. An example of approximate value.

Table 1
Information stored in the FMB

Type of data	Type	Information stored
Crisp data	0	None
Unknown	1	None
Undefined	2	None
Null	3	None
Linguistic label with possibility distribution	4	Labels and their defining characteristics
Possibility interval	5	Minimum and maximum
Approximate value	6	Margin
Linguistic label with similarity	7	Pairs (a, b) , degree of similarity between a and b

- Linguistic label with similarity:** This kind of data is defined on a nonordered domain. In this domain a relationship of similarity between the linguistic labels is defined. The relation is represented by a table showing the strength of the relations between all pairs of values belonging to the domain. This is the *type 7* kind of data and needs additional data to be stored in the FMB, as shown in Table 3.

4. Fuzzy meta-knowledge base

As we have seen in Section 3, some data types need additional information in order to be correctly manipulated. The FMB contains an efficient and organized way the necessary additional information required by the system.

Differently from what it was proposed in FIRST (in GEFRED by Medina [20], Galindo [11]), the database *Aliança* does not store its fuzzy meta-knowledge using tables and relations within the database. The FMB in *Aliança* is described in an XML format. This format makes this process easier for understanding and maintenance. The information stored for each type presented previously is shown in Table 1. As can be seen from the table, data types 0, 1, 2 and 3 do not store any additional information in the FMB.

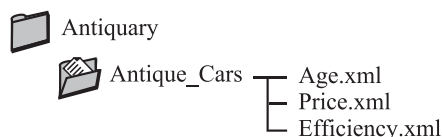


Fig. 7. FMB structure.

Table 2
List of antique cars

Id_car	Model	Price	Age	Efficiency
001	Alfa Romeo JK	17 500.00	34	Bad
002	Alfa Romeo Convertible	35 000.00	Antique	Regular
003	Dodge Polara	7 000.00	29	Bad
004	Dodge Dart	15 000.00	#35	Excellent
005	Porsche Spyder 550	28 000.00	Antique	Unknown
006	Porsche Spyder 550	33 000.00	Unknown	Good
007	Willys Gordini	10 000.00	[38,43]	Regular
008	Willys Bicuda	6 000.00	Medium	Bad

The symbol # means “approximately” and $[m, n]$ is an interval and means between m and n .

4.1. Structure of the FMB

The FMB is where all additional information necessary to handle the database transactions is stored. *Aliança*, differently from previous designs, defines for the FMB a directory structure, where the root directory is named after the database. Each database table contains, in the root directory, a subdirectory for each table. This subdirectory contains one XML file for each attribute.

We will describe, through an example, the internal structure of the fuzzy fields and the new way that data related to the fuzzy attributes are stored in the FMB of *Aliança*. The example is a database called *Antiquarius* that stores a relation containing information about antique cars. Fig. 7 shows the FMB directory structure while Table 2 lists the cars used as examples.

The description of the attributes of Table 2 is shown below according to the classification criteria used in *Aliança*, for the different fuzzy types.

- *Car_Id*: It is an integer serial numeric field automatically completed by the SGDB and it is the table primary key.
- *Model*: The model of each car. It is a text field of crisp type.
- *Price*: It is the price of each car. It is an attribute amenable to fuzzy treatment, so it can be filled with values of the type presented in Table 1. This attribute needs extra information stored in the BMN. The value of the *margin* (type 6) was defined as 1000. The definitions of the linguistic labels that use possibility distributions (type 4) are shown in Fig. 8.
- *Age*: Keeps the age of each car. It is, also, an attribute that can store fuzzy data, therefore all types defined in Table 1 can be used. The value of the margin was defined as 5. The linguistic labels are presented in Fig. 9.
- *Efficiency*: Stores the efficiency of each car when their performances are compared. It is another attribute that can be treated as a fuzzy quantity. Differently from the previous attributes, only similarity relations, which are type 7 values, can be stored. The linguistic labels and the similarities between all possible pairs of labels, for the attribute efficiency, are presented in Table 3.

The attributes Age and Price are defined over an ordered domain, therefore they can be filled with values of types 0, 1, 2, 3, 4, 5 and 6 from Table 1. In order to discuss the contents of the XML files we will consider the contents of the file Age.xml that is shown in Example 1.

It is important to note that tags that belong to the application domain, for instance the label RECENT, are variable. Tags that define the data structure, for instance DOMAIN and TYPE, are fixed.

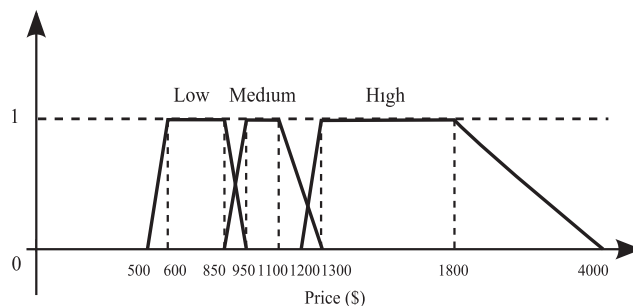


Fig. 8. Definition of the possible labels used for the attribute *Price*.

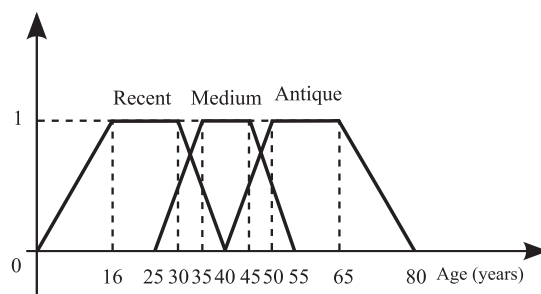


Fig. 9. Definition of the labels for the attribute *Age*.

Table 3
Similarity relation s_r defined over the attribute *Efficiency*

$s_r(d, d')$	Bad	Regular	Good	Excellent
Bad	1	0.8	0.5	0.1
Regular	0.8	1	0.7	0.5
Good	0.5	0.7	1	0.8
Excellent	0.1	0.5	0.8	1

Example 1. File *Age.xml*

```

<? XML VERSION = "1.0" >
<AGE>
  <DOMAIN A=0 B=110 />
  <TYPE T=4>
    <LABELS>
      <RECENT A=0 B=16 C=30 D=40 />
      <MEDIUM A=25 B=35 C=45 D=55 />
      <ANTIQUA A=40 B=50 C=65 D=80 />
    </LABELS>
  </TYPE>
  <TYPE T=5>
    <INTERVAL MIN=1 MAX=5 />
  </TYPE>
  <TYPE T=6>
    <MARGIN M=5>
  </TYPE>
</AGE>
    
```

The file Age.xml is divided into sections and each section defines some characteristics of the attribute, for example, its domain.

- * Tag <DOMAIN A=0 B=110 />: Presents the domain $[A, B]$ over which the fuzzy attribute is described.
- * Tag <TYPE T=4>: This section of the file defines the characteristics that represent the possibility distributions (type 4) of each label.
 - Tag <LABELS>: This subsection defines all the labels used by this attribute, see Fig. 9, where A, B, C and D, respectively, are a, b, c and d (Fig. 4).
 - . Tag <RECENT A=0 B=16 C=30 D=40 />: Presents the parameters that describe the trapezoidal distribution RECENT.
 - . Tag <MEDIUM A=25 B=35 C=45 D=55 />: Presents the parameters that describe the trapezoidal distribution MEDIUM.
 - . Tag <ANTIQUE A=40 B=50 C=65 D=80 />: Presents the parameters that describe the trapezoidal distribution ANTIQUE.
- * Tag <TYPE T=5>: This section presents the characteristics of the *possibility intervals*.
 - Tag <INTERVAL MIN=1 MAX=5 />: This tag establishes the possible minimum and maximum sizes for the interval.
- * Tag <TYPE T=6>: Presents the required characteristics to represent an *approximate value*.
 - Tag <MARGIN M=5>: The tag defines for approximate values the margin size M as 5.

In a very similar way, the file Price.xml stores the additional information of the attribute Price. Although this attribute (see Table 2) presents only crisp data, it has additional information in the FMB in order to make the vague treatment possible.

Example 2. File Price.xml

```
<? XML VERSION = "1.0">
<PRICE>
  <DOMAIN A=500 B=4000 />
  <TYPE T=4>
    <LABELS>
      <LOW A=500 B=600 C=850 D=950 />
      <MEDIUM A=850 B=950 C=1100 D=1300 />
      <HIGH A=1200 B=1300 C=1800 D=4000 />
    </LABELS>
  </TYPE>
  <TYPE T=5>
    <INTERVAL MIN=10 MAX=200 />
  </TYPE>
  <TYPE T=6>
    <MARGIN M=150>
  </TYPE>
</PRICE>
```

In order to present an example of type 7 XML file let us consider the attribute Efficiency which is defined over an unordered domain and can only be filled with data of this type. As explained before the XML file must be called Efficiency.xml and contains the information presented in Example 3.

As in the previous example, the tag DOMAIN presents the discrete domain over which the fuzzy attribute is described. Note that the domain is composed of a set of labels.

- * Tag <DOMAIN A=bad B=regular C=good D=excellent />: Presents the discrete domain over which the fuzzy attribute is described.

Table 4

Internal representation of the relation *Antique cars* in the SGBD with the proposed extension for the *Aliança*

Id	Name	Price	PriceT	Price1	Price2	Age	AgeT	Age1	Age2	Efficiency	EfficT
001	Alfa Romeo JK	17500.00	0	17500.00	–	34	0	34	–	\$\$Bad	7
002	Alfa Romeo Conv.	35000.00	0	35000.00	–	\$Antique	4	–	–	\$\$Regular	7
003	Dodge Polara	7000.00	0	7000.00	–	29	0	29	–	\$\$Bad	7
004	Dodge Dart	15000.00	0	15000.00	–	#35	6	35	5	\$\$Excellent	7
005	Porsche Spyder	28000.00	0	28000.00	–	\$Antique	4	–	–	Unknown	1
006	Porsche Spyder	33000.00	0	33000.00	–	Unknown	1	–	–	\$\$Good	7
007	Willys Gordini	10000.00	0	10000.00	–	[38,43]	5	38	43	\$\$Regula	7
008	Willys Bicuda	6000.00	0	6000.00	–	\$Medium	4	–	–	\$\$Bad	7

Example 3. File *Efficiency.xml*

```

<?XML VERSION = "1.0">
<EFFICIENCY>
  <DOMAIN A=bad B=regular C=good D=excellent />
  <TYPE T=7>
    <LABELS>
      <BAD bad=1 regular=0.8 good=0.5 excellent=0.1 />
      <REGULAR bad=0.8 regular=1 good=0.7 excellent=0.5 />
      <GOOD bad=0.5 regular=0.7 good=1 excellent=0.8 />
      <EXCELLENT bad=0.1 regular=0.5 good=0.8 excellent=1 />
    </LABELS>
  </TYPE>
</EFFICIENCY>

```

Since the attribute Efficiency is based on similarities it is necessary to describe the strength of each relation.

- * Tag <TYPE T=7>: Presents the characteristics necessary to represent a *linguistic label based on similarities*.
- Tag <LABELS>: Presents the similarity degrees among the labels.
 - . Tag <BAD bad=1.0 regular=0.8 good=0.5 excellent=0.1 />.
 - . Tag <REGULAR bad=0.8 regular=1.0 good=0.7 excellent=0.5 />.
 - . Tag <GOOD bad=0.5 regular=0.7 good=1.0 excellent=0.8 />.
 - . Tag <EXCELLENT bad=0.1 regular=0.5 good=0.8 excellent=1.0 />.

Considering these definitions just discussed it is possible to present in Table 4 the real internal structure of Table 2. It is important to note that this internal representation is transparent to the user. In order to explain these new attributes let us again consider the fuzzy attribute Age as an example. Since Age is defined over an ordered domain it can be filled with values of types 0, 1, 2, 3, 4, 5 and 6 from Table 1. In order to take into account all these types the three new attributes are added to the database: AgeT, Age1 and Age2. AgeT defines the type of the stored data. Age1 and Age2 will receive information according to the type stored. For example, a data type of type 6 (Approximate value) requires one extra value, a margin.

Let us now consider the fuzzy attribute Efficiency. Since this attribute is defined over unordered domain it can only be filled with types 1, 2, 3 and 7 attributes. This kind of attribute will imply in the addition of only one extra column called EfficiencyT.

All these new attributes are used when fuzzy queries are used. A fuzzy query is translated into a classical query based on the new attributes.

The use of a structure of directories and XML files facilitates the creation and maintenance of fuzzy databases when compared to the model GEFRED described in [20,11]. This model uses a series of tables within the database manager that are difficult to create and maintain.

Table 5
Description of the new attributes used in the internal representation of fuzzy values

Age	AgeT	Age1	Age2	Description
30	0	30	–	30 is a crisp value, therefore AgeT = 0, and Age1 = 30 which indicates the type receives 0 and Age = 30
Unknown	1	–	–	The label Unknown is a type 1 attribute
Undefined	2	–	–	The label Undefined is a type 2, therefore IdadeT = 2
Null	3	–	–	The label Null is a type 2 attribute
\$Antique	4	–	–	The label \$Antique is a trapezoidal function, therefore AgeT = 4 and the remaining information is stored in the file Age.xml
[30, 45]	5	30	45	The value [30, 45] is of an interval type, therefore AgeT = 5 and Age1 = 30 and Age2 = 45
#25	6	25	5	The value 25 id of type Approximate, therefore AgeT = 6, Age1 = 25 and Age2 = margin size

5. FSQL Server

The FSQL Server is the system principal module, because it is responsible for the relationship between the RDBMS and FMB. One of its objectives is to transform the information produced by the FSQL into a classical SQL.

The server was written in Java and uses JavaCC [17] to generate the FSQL parser. It performs both syntactic and lexical analysis on the submitted query. After separating the text into tokens, the FSQL identifies the fuzzy data type that is right after the fuzzy comparator. It then searches on the BMN for the information required to build the classical SQL query. The resulting query compares the given fuzzy type to all types stored in the database. It is important to note that this query is composed of n queries, where n is the number of different types shown in Section 3. Besides this, each of these queries can be expanded according to the membership functions used to represent the fuzzy values.

5.1. FSQL—fuzzy SQL

FSQL assumes a conventional database and adds external tools to handle fuzzy data. This approach is also used by Bosc et al. [1], Galindo [11], Kacprzyk and Zadrozny [18]. The FSQL introduces new features to facilitate the handling of imprecise information. FSQL is based on similarity relations [35] and the theory of possibility introduced by Zadeh [36].

- *Linguistic labels*: When an attribute receives a linguistic label, this label is preceded by a symbol that makes its identification easy. There are two linguistic labels: linguistic labels with possibility distributions and linguistic labels with similarity relations. The linguistic label with similarity is preceded by the symbol \$ and the labels with similarity relations by \$\$.
- *Possibility interval* $[m, n]$: They are represented by two numerical values m and n between brackets $[m, n]$.
- *Approximate value*: They are preceded by the symbol #. For example, #10 expresses “approximately 10”.
- *Fuzzy comparators*: Besides the classical comparison operators ($>$, $<$, $=$, \dots), the proposed FSQL has the following fuzzy comparators (FEQ—possibly fuzzy equal, FGEQ—possibly fuzzy greater than, NFEQ—necessarily fuzzy equal, NFLEQ—necessarily fuzzy less or equal than, etc.). For example:

$$\mu_{\text{FEQ}}(X, Y) = \sup_{x_i \in \Omega} [\min(X(x_i), Y(x_i))], \quad (1)$$

where Ω is the universe of discourse of both fuzzy data X and Y .

$$\mu_{\text{NFEQ}}(X, Y) = \inf_{x_i \in \Omega} [\max(1 - X(x_i), Y(x_i))], \quad (2)$$

where Ω is the universe of discourse of both fuzzy data X and Y .

- *Acceptation degree*: For every fuzzy condition used in a query, a degree similar to the alpha cut can be defined. This degree appears between parenthesis and after the condition.

Example 4. Find all expensive cars (with degree 0.8).

```
SELECT Model
FROM Antique_Car
WHERE Price FEQ High (0.8)
```

Example 5. Select models and prices for cars whose age is possibly between approximately 20 years old and age antique; use a degree of at least 0.7.

```
SELECT Name, Age
FROM Employee
WHERE Age FGEQ #20 AND Age FLEQ Antique (0.7)
```

6. Conclusions

This work presented and discussed the main characteristics of the new fuzzy database *Aliança*, which includes the main features needed to treat information of a wide range of possibilities. Particularly, imprecise information is well supported. Despite the fact that it included this range of data types, the additional complexity is lower than the previous proposals due to the fact that it uses XML to represent the meta-knowledge. An additional advantage is the use of XML to represent the fuzzy meta-knowledge which makes it easy to maintain and understand the structure of imprecise information. The fuzzy database architecture *Aliança* approximates the interaction with databases to the usual way human beings reason.

References

- [1] P. Bosc, L. Duval, O. Pivert, An initial approach to the evaluation of possibilistic queries addressed to possibilistic databases, *Fuzzy Sets and Systems* 140 (2003) 151–166.
- [2] B.P. Buckles, F.E. Petry, A fuzzy representation of data for relational databases, *Fuzzy Sets and Systems* 7 (1982) 213–226.
- [3] B.P. Buckles, F.E. Petry, Extending the fuzzy databases with fuzzy numbers, *Inform. Sci.* 34 (1984) 145–155.
- [7] E.F. Codd, Extending the database relational model to capture more meaning, *ACM Trans. Database Systems* 4 (1979) 262–296.
- [9] E.F. Codd, Missing information (applicable and inapplicable) in relational databases, *ACM SIGMOD Record* 15 (4) (1986).
- [10] E.F. Codd, More commentary on missing information in relational databases, *ACM SIGMOD Record* EFC-6 (1986).
- [11] J.G. Galindo, Tratamiento de la Imprecisión en Bases de Datos Relacionales: Extensión del Modelo y Adaptación de los SGBD Actuales, Tese Doutoral, Universidade de Granada, Espanha, 1999.
- [12] J.G. Galindo, M.C. Aranda, J.L. Caro, A. Guevara, A. Aguayo, Applying fuzzy databases and FSQL to the management of rural accommodation, *Tourism Management* 24 (2003) 457–463.
- [13] J.G. Galindo, J.M. Medina, M.C. Aranda-Garrido, Fuzzy division in fuzzy relational databases: an approach, *Fuzzy Sets and Systems* 121 (2001) 471–490.
- [14] A. Gaurav, R. Alhadj, Incorporating Fuzziness in XML and Mapping Fuzzy Relational Data into Fuzzy XML, ACM, New York, 2006, pp. 456–460, 1-59593-108-2/06/0004.
- [15] E.R. Harold, XML Bible, IDG Books Worldwide, Inc., 1999.
- [17] Java.net Web Site (<https://javacc.dev.java.net/>).
- [18] J. Kacprzyk, S. Zadrozny, Computing with words in intelligent database querying: standalone and Internet-based applications, *Inform. Sci.* 134 (2001) 71–109.
- [20] J.M. Medina, Bases de Datos Relacionales Difusas. Modelo Teórico y Aspectos de su Implementación, Tese Doutoral, Universidade de Granada, Espanha, 1994.
- [21] J.M. Medina, M.A. Vila, J.C. Cubero, O. Pons, Towards the implementation of a generalized fuzzy relational database model, *Fuzzy Sets and Systems* 75 (1995) 273–289.
- [22] H. Prade, C. Testemale, Generalizing database relational algebra for the treatment of incomplete or uncertain information and vague queries, *Inform. Sci.* 34 (1984) 115–143.
- [25] K. Turowski, U. Weng, Representing and processing fuzzy information—an XML-based approach, *Knowledge-Based System* 15 (2002) 67–75.
- [28] M. Umamo, S. Fukami, Fuzzy relational algebra for possibility-distribution-fuzzy-relational model of fuzzy data, *J. Intelligent Inform. Systems* 3 (1994) 7–28.
- [35] L.A. Zadeh, Similarity relations and fuzzy orderings, *Inform. Sci.* 3 (2) (1971) 177–200.
- [36] L.A. Zadeh, Fuzzy sets as a basis for a theory of possibility, *Fuzzy Sets and Systems* 1 (1978) 3–28.
- [37] M. Zemankova, A. Kandel, Implementing imprecision in information systems, *Inform. Sci.* 37 (1985) 107–141.