Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет ННЦЗФН
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Розробка та дослідження методів підтримки технологій нечітких множин засобами реляційних систем
(тема)

Виконав:
студент 2 курсу, групи СШІзм-20-2
Клад А.К.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник к.т.н, доц. Шергін В.Л.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет ННЦЗФН
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Клад Антону Костянтиновичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та дослідження методів підтримки технологій нечітких множин засобами реляційних систем

затверджена наказом університету від 25 листопада 20 21 р. № 168
2. Термін подання студентом роботи до екзаменаційної комісії _____ _____ 20 21 р.
3. Вихідні дані до роботи Науково-технічні публікації, дані Інтернет та відомих наукових проектів, електронні документації, тестові набори даних

4. Перелік питань, що потрібно опрацювати в роботі аналіз предметної області, реляційна модель баз даних, теорія нечітких множин, операції реляційної алгебри засобами мови SQL, постановка задачі дослідження, математичний опис нечітких запитів до реляційних баз даних, поняття нечіткого запиту, функція приналежності, метод побудови нечітких запитів, імітаційне моделювання і рішення задач на тестових вибірках, дослідження працездатності та ефективності застосування нечітких запитів у реальних базах даних, аналіз отриманих результатів

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри)_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Основна частина	к.т.н., доцент, Шергін В. Л.		

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Отримання завдання на кваліфікаційну роботу	01.09.2021	виконано
2.	Аналіз завдання та пошук літератури за темою	02.09.2021–08.09.2021	виконано
3.	Опрацювання літератури та аналіз об'єкту	09.09.2021–15.09.2021	виконано
4.	Вибір програмних засобів для розробки системи	16.09.2021–18.09.2021	виконано
5.	Розробка програмного засобу	19.09.2021–05.11.2021	виконано
6.	Аналіз отриманих результатів	06.11.2021–16.11.2021	виконано
7.	Оформлювання пояснювальної записки	17.11.2021–24.11.2021	виконано
8.	Проходження нормоконтролю	25.11.2021–28.11.2021	виконано
9.	Оформлення презентаційних матеріалів	29.11.2021–01.12.2021	виконано
10.	Попередній захист	02.12.2021	виконано
11.	Представлення кваліфікаційної роботи		

Дата видачі завдання 1 вересня 20 21 р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 77 с., 24 рис., 17 табл., 3 дод., 26 джерел.

БАЗА ДАНИХ, ІНФОРМАЦІЙНА СИСТЕМА, КОРПОРАТИВНА ІНФОРМАЦІЙНА СИСТЕМА, НЕЧІТКИЙ ЛІНГВІСТИЧНИЙ ІНТЕРФЕЙС, ОПЕРАТИВНЕ ДЖЕРЕЛО ДАНИХ, СИСТЕМА УПРАВЛІННЯ БАЗАМИ ДАНИХ, СХОВИЩЕ ДАНИХ, STRUCTURED QUERY LANGUAGE.

Об'єктом дослідження є методи підтримки технологій нечітких множин засобами реляційних систем.

Метою роботи є проведення дослідження методів підтримки технологій нечітких множин в базах даних та розробка рекомендацій по написанню ефективних SQL запитів.

Методи дослідження: аналіз літератури та інтернет – джерел.

У даній роботі розглянуті проблеми та рішення в області підтримки нечітких множин у реляційних базах даних. Розглянуто основні проблеми перетворення запитів, проблеми семантичної і логічної оптимізації, проблеми вибору і оцінки вартості альтернативних планів виконання запитів. Виконано аналіз існуючих методів оптимізації виконання запитів. Детально розглянуто питання побудови послідовності з'єднання відносин в запитах реляційної бази даних. На практиці продемонстровані результати застосування деяких методів оптимізації.

РЕФЕРАТ

Пояснительная записка: 77 с., 24 рис., 17 табл., 3 прил., 26 источников.

БАЗА ДАННЫХ, ИНФОРМАЦИОННАЯ СИСТЕМА, КОРПОРАТИВНАЯ ИНФОРМАЦИОННАЯ СИСТЕМА, НЕЧЕТКИЙ ЛИНГВИСТИЧЕСКИЙ ИНТЕРФЕЙС, ОПЕРАТИВНЫЙ ИСТОЧНИК ДАННЫХ, СИСТЕМА УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ, ХРАНИЛИЩЕ ДАННЫХ, STRUCTURED QUERY LANGUAGE.

Объектом исследования являются методы поддержки технологий нечетких множеств средствами реляционных систем.

Целью работы является исследование методов поддержки технологий нечетких множеств в базах данных и разработка рекомендаций по написанию эффективных SQL запросов.

Методы исследования: анализ литературы и интернет – ресурсы.

В данной работе рассмотрены проблемы и решения в области поддержки нечетких множеств в реляционных базах данных. Рассмотрены основные проблемы преобразования запросов, проблемы семантической и логической оптимизации, проблемы выбора и оценка стоимости альтернативных планов выполнения запросов. Проведен анализ существующих методов оптимизации выполнения запросов. Подробно рассмотрен вопрос построения последовательности соединения отношений в запросах реляционной базы данных. На практике показаны результаты внедрения неких способов оптимизации.

ABSTRACT

Explanatory note: 77 p., 24 fig., 17 tabl., 3 ann., 26 sources.

DATABASE, INFORMATION SYSTEM, CORPORATE INFORMATION SYSTEM, FUZZY LINGUISTIC INTERFACE, OPERATIONAL DATA SOURCE, DATABASE MANAGEMENT SYSTEM, DATA STORE, STRUCTURED QUERY LANGUAGE.

The object of research is methods of support of fuzzy set technologies by means of relational systems.

The aim of the work is to conduct research on methods for supporting fuzzy set technologies in databases and to develop recommendations for writing efficient SQL queries.

Research methods: analysis of literature and Internet sources.

This paper considers problems and solutions in the field of fuzzy set support in relational databases. The main problems of query transformation, problems of semantic and logical optimization, problems of selection and estimation of cost of alternative plans of execution of queries are considered. The analysis of existing methods of optimization of execution of inquiries is executed. The question of construction of sequence of connection of relations in inquiries of a relational database is considered in detail. In practice, the results of the application of some optimization methods are demonstrated.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

БД – база даних;

ІС – інформаційна система;

КІС – корпоративна інформаційна система;

НЛІ – нечіткий лінгвістичний інтерфейс;

ОДД – оперативне джерело даних;

СД – сховище даних;

СУБД – система управління базами даних;

SQL – Structured Query Language – мова структурованих запитів.

ВСТУП

Математична теорія нечітких множин (fuzzy sets) і нечітка логіка (fuzzy logic) є узагальненнями класичної теорії множин і класичної формальної логіки. Дані поняття були вперше запропоновані американським вченим Лотфі Заде (Lotfi Zadeh) [1] в 1965 р Основною причиною появи нової теорії стало наявність нечітких і наближених міркувань при описі людиною процесів, систем, об'єктів [2].

Перш ніж нечіткий підхід до моделювання складних систем отримав визнання у всьому світі, пройшло не одне десятиліття з моменту зародження теорії нечітких множин. І на цьому шляху розвитку нечітких систем прийнято виділяти три періоди.

Нечітка логіка є багатозначною логікою, що дозволяє визначити проміжні значення для таких загальноприйнятих оцінок, як так/ні, істинно/помилково, чорне/біле і т.п. Вирази подібні до таких, як злегка тепло або досить холодно можна формулювати математично і обробляти на комп'ютерах.

Перший період (кінець 60-х-початок 70 років) характеризується розвитком теоретичного апарату нечітких множин (Л. Заде, Е. Мамдані, Беллмана). У другому періоді (70-80-ті роки) з'являються перші практичні результати в області нечіткого управління складними технічними системами (парогенератор з нечітким керуванням) [3]. Одночасно стало приділятися увага питанням побудови експертних систем, заснованих на нечіткій логіці, розробці нечітких контролерів. Нечіткі експертні системи для підтримки прийняття рішень знаходять широке застосування в медицині та економіці.

Нарешті, в третьому періоді, який триває з кінця 80-х років і триває в даний час, з'являються пакети програм для побудови нечітких експертних

систем, а області застосування нечіткої логіки помітно розширюються. Вона застосовується в автомобільній, аерокосмічній і транспортної промисловості, в області виробів побутової техніки, у сфері фінансів, аналізу і прийняття управлінських рішень та багатьох інших. Найголовнішим поняттям систем, заснованих на нечіткій логіки, є поняття нечіткої (під)множини.

Тріумфальний хід нечіткої логіки по світу почалося після докази в кінці 80-х Бартоломеєм Коско знаменитої теореми FAT (Fuzzy Approximation Theorem) [4], [5], [6]. У бізнесі і фінансах нечітка логіка отримала визнання після того як в 1988 році експертна система на основі нечітких правил для прогнозування фінансових індикаторів єдина передбачила біржовий крах [7]. І кількість успішних фазі-застосувань в даний час обчислюється тисячами.

У Японії дослідження у галузі нечіткої логіки отримали широку фінансову підтримку. У Європі та США зусилля були спрямовані на те, щоб скоротити величезний відрив японців. Так, наприклад, агентство космічних досліджень NASA стало використовувати нечітку логіку в маневрах стикування.

Більша частина даних, які обробляються в сучасних інформаційних системах, носять чіткий, числовий характер. Однак в запитах до баз даних, які намагається формулювати людина, часто присутні неточності і невизначеності. Тож не дивно, коли на запит в пошуковій системі Інтернету користувачеві видається безліч посилань на документи, упорядкованих за ступенем релевантності (або відповідності) запиту. Тому що текстової інформації від початку властива нечіткість і невизначеність, причинами якої є семантична неоднозначність мови, наявність синонімів і т.д.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Реляційна модель баз даних

Реляційна модель даних – логічна модель даних. Вперше була запропонована британським ученим співробітником компанії IBM Едгаром Франком Коддом (E. F. Codd) в 1970 році в статті «A Relational Model of Data for Large Shared Data Banks». В даний час ця модель є фактичним стандартом, на який орієнтуються практично всі сучасні комерційні СУБД. Термін «реляційний» означає, що теорія полягає в математичному понятті ставлення (relation).

У реляційній моделі (рисунок 1.1) досягається набагато більш високий рівень абстракції даних, ніж в ієрархічній або мережевій. У згаданій статті Е. Ф. Кодда стверджується, що «реляційна модель надає засоби опису даних на основі тільки їх природної структури, тобто без потреби введення якої додаткової структури для цілей машинного представлення» [8].

Іншими словами, подання даних не залежить від способу їх фізичної організації. Це забезпечується за рахунок використання математичної теорії відносин (сама назва «реляційна» походить від англійського relation – «відношення») [9], [10].

До складу реляційної моделі даних зазвичай включають теорію нормалізації. Крістофер Дейт визначив три складові частини реляційної моделі даних:

- структурна;
- маніпуляційна;
- цілісна.

Рисунок 1.1 – Приклад реляційної моделі баз даних

Структурна частина моделі визначає, що єдиною структурою даних є нормалізоване n-арное ставлення. Відносини зручно представляти у формі таблиць, де кожен рядок є кортеж, а кожен стовпець – атрибут, визначений на деякому домені (рисунок 1.2). Даний неформальний підхід до поняття відносини дає більш звичну для розробників і користувачів форму представлення, де реляційна база даних являє собою кінцевий набір таблиць.

Маніпуляційна частина моделі визначає два фундаментальних механізму маніпулювання даними – реляційна алгебра і реляційне числення. Основною функцією маніпуляційної частини реляційної моделі є забезпечення заходів реляційності будь-якої конкретної мови реляційних БД: мова називається реляційною, якщо вона має не меншу виразність і потужність, ніж реляційна алгебра (рисунок 1.3) або реляційне числення [11].

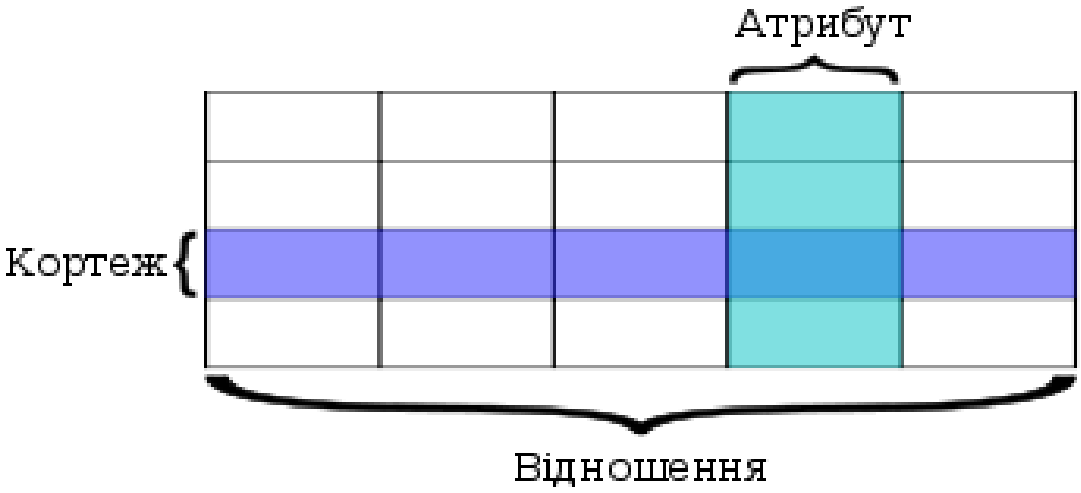

Рисунок 1.2 – Відношення (реляційна модель)

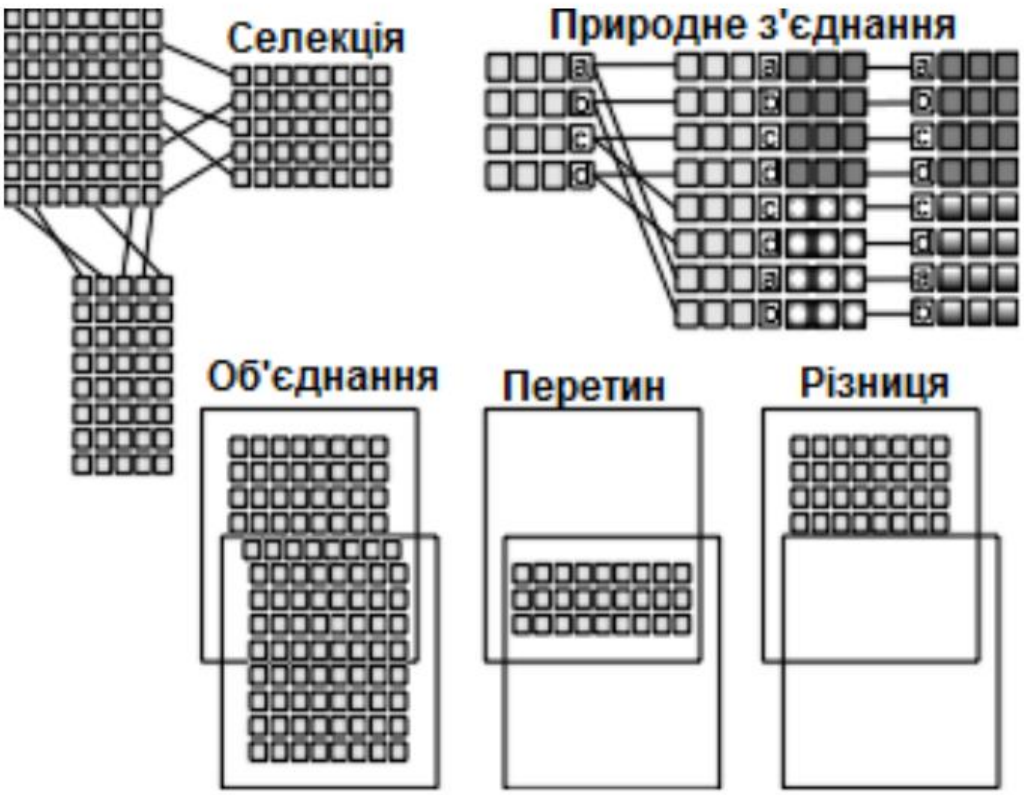

Рисунок 1.3 – Деякі операції реляційної алгебри

Цілісна частина моделі визначає вимоги цілісності сутностей і цілісності посилань. Перша вимога полягає в тому, що будь-який кортеж будь-якого відношення відмітний від будь-якого іншого кортежу цього відношення, тобто іншими словами, будь-яке відношення має володіти первинним ключем. Вимога цілісності по посиланнях, або вимога зовнішнього ключа полягає в тому, що для кожного значення зовнішнього ключа, що з'являється в посиланному відношенні, у відношенні, на яку веде посилання, повинен знайтися кортеж з таким же значенням первинного ключа, або значення зовнішнього ключа повинно бути невизначеним, тобто ні на що не вказувати (рисунок 1.4).

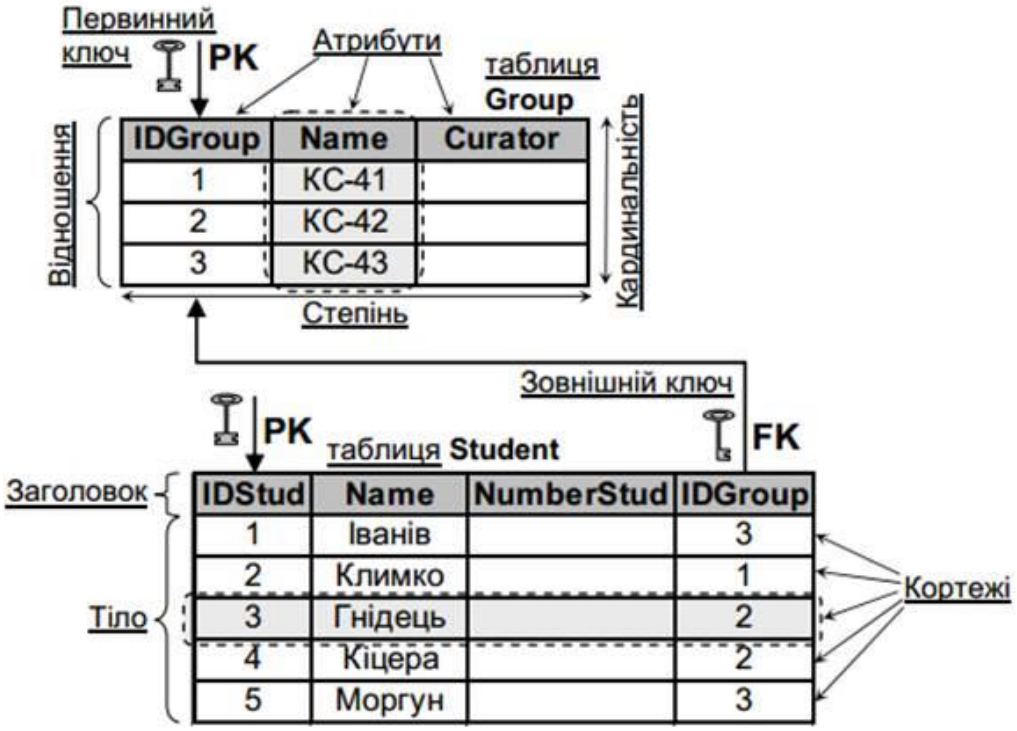

Рисунок 1.4 – Приклад побудування первинного та зовнішнього ключів

Можна провести аналогію між елементами реляційної моделі даних і елементами моделі «сутність-зв'язок». Реляційні відносини відповідають

наборам сутностей, а кортежі – сутностям. Тому, також як і в моделі «сутність-зв'язок» стовпці в таблиці, що представляє реляційне відношення, називають атрибутами.

Кожен атрибут визначений на домені, тому домен можна розглядати як безліч допустимих значень даного атрибуту. Кілька атрибутів одних відносин і навіть атрибути різних відносин можуть бути визначені на одному і тому ж домені.

Іменоване безліч пар «ім'я атрибута – ім'я домену» називається схемою відношення. Потужність цієї множини – називають ступенем чи «арністю» відносини. Набір іменованих схем відносин представляє із себе схему бази даних.

Атрибут, значення якого однозначно ідентифікує кортежі, називається ключовим (або просто ключем). У нашому випадку ключем є атрибут «Табельний номер», оскільки його значення унікально для кожного працівника підприємства. Якщо кортежі ідентифікуються тільки зчепленням значень декількох атрибутів, то говорять, що відношення має складовий ключ. Ставлення може містити кілька ключів. Завжди один із ключів оголошується первинним, його значення не можуть обновлятися. Всі інші ключі відносини називаються можливими ключами.

На відміну від ієрархічної і мережної моделей даних в реляційної відсутнє поняття групових відносин. Для відображення асоціацій між кортежами різних відносин використовується дублювання їх ключів.

1.2 Теорія нечітких множин

Основи теорії нечітких множин і нечіткої логіки були закладені наприкінці 1960-х років у працях відомого американського математика Лотфі Заде. Його праця «Fuzzy Sets», опублікована у 1965 р. в журналі «Information

and Control», стала поштовхом до розвитку нової математичної теорії. Він дав назву і новій галузі науки – «fuzzy sets» (fuzzy – нечіткий, розмитий, м'який) [12]. Основною причиною появи нової теорії стали нечіткі і наближені міркування, що використовувались для опису людиною процесів, систем, об'єктів. Математична теорія нечітких множин (fuzzy sets) і нечітка логіка (fuzzy logic) є узагальненнями класичної теорії множин і класичної формальної логіки [13].

Перш ніж нечіткий підхід до моделювання складних систем отримав визнання в усьому світі, минуло не одне десятиліття. Що ж запропонував Л. Заде? По-перше, він розширив класичне канторовське поняття множини, припустивши, що характеристична функція (функція належності елемента множині) може набувати будь-яких значень в інтервалі [0, 1], а не тільки значень 0 або 1. Такі множини він назвав нечіткими (рисунок 1.5) [14]. Л. Заде визначив також низку операцій із нечіткими множинами і запропонував узагальнення методів логічного висновку.

Ввівши згодом поняття лінгвістичної змінної і припустивши, що її значеннями (термами) є нечіткі множини, Л. Заде створив апарат для опису процесів інтелектуальної діяльності, включаючи нечіткість і невизначеність виразів (наприклад, високий, середній, незначний ризики).

Завданням нечітких множин є визначення належності деякого об'єкта чи елемента до заданої множини. Нехай E – деяка множина, а A – підмножина E, тобто A⊂E. Той факт, що елемент *x* множини E належить і множині A в теорії множин позначають так: $x \in \mathrm{A}$. Щоб виразити цю належність, можна скористатися й іншим поняттям – характеристичною функцією $\mu_{\mathrm{A}}(x)$, значення якої вказують, чи є (так або ні) *x* елементом A:

$$\mu_{\mathrm{A}}(x) = \begin{cases} 1, \text{якщо } x \in A, \\ 0, \text{якщо } x \notin A. \end{cases} \tag{1.1}$$

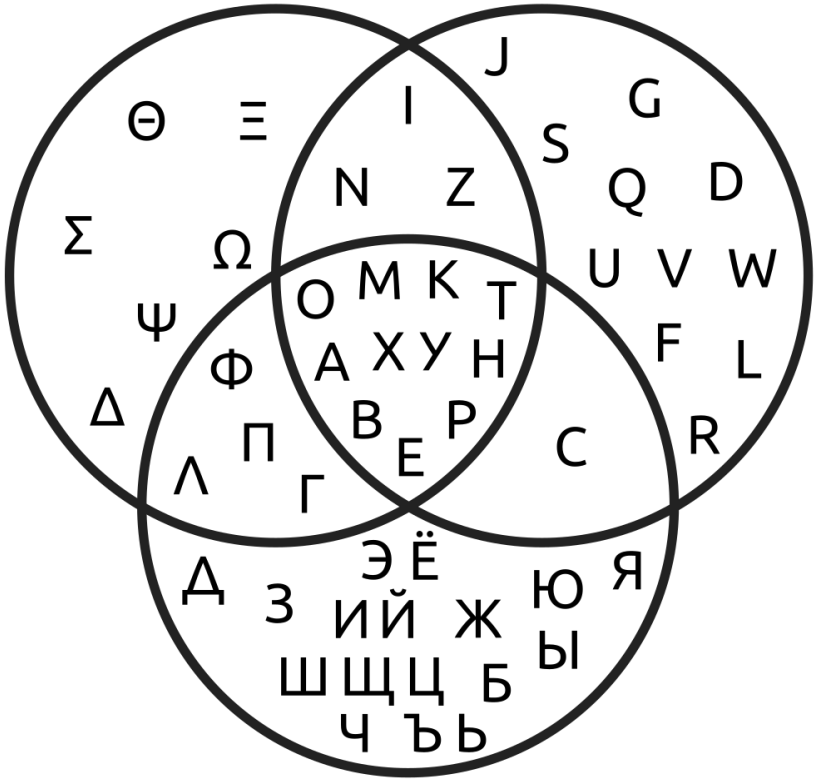

Рисунок 1.5 – Діаграма Венна, що демонструє множини великих літер грецької, латинської і російської мови

Згідно з теорією нечітких множин, характеристична функція належності може набувати будь-якого значення в інтервалі [0, 1], а не тільки два – 0 і 1. Відповідно до цього, елемент x_i множини E може не належати A($\mu_A(x) = 0$), бути елементом A невеликою мірою (значення $\mu_A(x)$ близьке до нуля), бути елементом A значною мірою ($\mu_A(x)$ близьке до 1) або бути елементом A($\mu_A(x) = 1$). Отже, поняття належності узагальнюється. Нечітку під множину A універсальної множини E позначають $A_н$ і визначають упорядкованими парами [15]:

$$A_H = \{(x, \mu_A(x)) | x \in E\}. \quad (1.2)$$

Характеристична функція належності (або просто функція належності) $\mu_{\text{A}}(x)$ набуває значень у деякій упорядкованій множині M (наприклад, M = [0, 1]). Ця функція належності вказує ступінь (або рівень) належності елемента x до підмножини A. Множину M називають множиною надійності. Якщо M = {0, 1}, то нечітку підмножину A можна розглядати як звичайну або чітку множину, функція належності якої набуває лише бінарних значень.

Тому основні операції над нечіткими множинами також являють собою узагальнення відповідних властивостей та операцій класичної теорії множин.

Рівність. Дві нечіткі множини $\widetilde{\text{A}}$ і $\widetilde{\text{B}}$, що задані на U, є рівними, якщо вони складаються з одних і тих же елементів для всіх $x \in U$ та $\mu_{\text{A}}(x) = \mu_{\text{B}}(x)$. Позначаються як $\widetilde{\text{A}} = \widetilde{\text{B}}$.

Доповнення. Нехай M = [0, 1], $\widetilde{\text{A}}$ і $\widetilde{\text{B}}$ – нечіткі множини, які доповнюють одна одну, якщо $\mu_{\text{A}}(x) = 1 - \mu_{\text{B}}(x)$ для всіх $x \in U$.

Перетин двох нечітких множин (нечітке «І»), що позначають A ∩ B – найбільша нечітка підмножина, яка знаходиться одночасно в A і B. Визначають так:

$$\mu_A(x) \wedge \mu_B(x) = \min\big(\mu_A(x), \mu_B(x)\big). \qquad (1.3)$$

Об'єднання двох нечітких множин (нечітке «АБО»), що позначають A ∪ B – найменша нечітка підмножина, яка включає як A, так і B, з функцією належності

$$\mu_A(x) \vee \mu_B(x) = \max\big(\mu_A(x), \mu_B(x)\big). \qquad (1.4)$$

Різниця двох нечітких множин A – B = A ∩ B з функцією належності

$$\mu_{A-B}(x) = \mu_{A\cap\overline{B}}(x) = \min\big(\mu_A(x), 1-\mu_B(x)\big). \quad (1.5)$$

Нехай M = [0,1] і A – нечітка множина з елементами x з універсальної множини E і множиною значень функцій належності M.

Величину $\sup\limits_{x\in E}\mu_A(x)$ називають висотою нечіткої множини A. Нечітка множина A є нормальною, якщо її висота дорівнює 1, тобто верхня межа її функції належності дорівнює 1.

Нечітка множина є порожньою, якщо $\forall x \in E \mu_{\mathrm{A}}(x) = 0$. Непорожню субнормальну множину можна нормалізувати за формулою

$$\mu_A(x) \coloneqq \frac{\mu_A(x)}{\sup\limits_{x\in E}\mu_A(x)}. \quad (1.6)$$

Розглянемо прямокутну систему координат, на осі ординат якої відкладено значення $\mu_{\mathrm{A}}(x)$, на осі абсцис – у довільному порядку розміщені елементи E. Якщо множина E за своєю природою впорядкована, то цей порядок бажано зберегти в розміщенні елементів на осі абсцис. Таке подання інформації демонструє прості операції над нечіткими множинами.

Нехай A – нечіткий інтервал між 5 і 8, а B – нечітке число, близьке до 4 (рисунок 1.6 а, б) [16].

Нечітку множину між 5 і 8 І (AND) близько 4 (темна лінія) ілюструє рисунок 1.6 (в), нечітку множину між 5 і 8 АБО (OR) близько 4 – рисунок 1.6 (г) (темна лінія).

Для опису нечітких множин уводять поняття нечіткої і лінгвістичної змінних. Нечітку змінну описує набір <β, X, A>, де β – назва змінної, X – універсальна множина (область визначення β), A – нечітка множина на X, що описує обмеження на значення нечіткої змінної β.

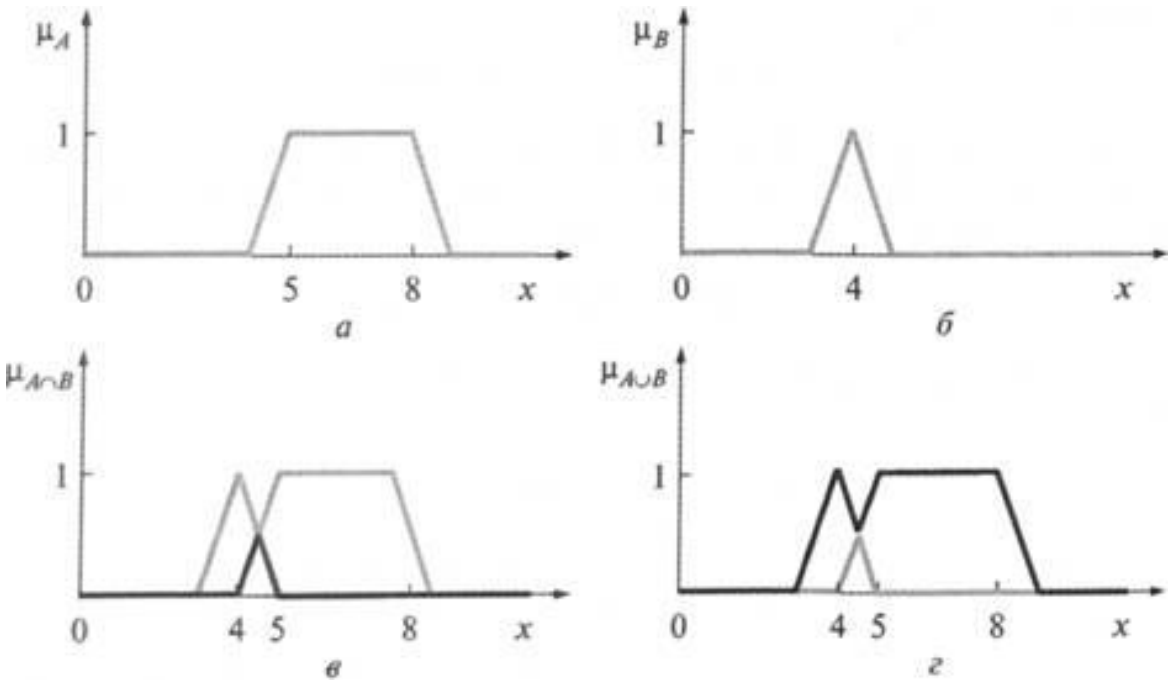

Рисунок 1.6 – Приклади нечітких множин (а, б), їх перетину (в) та об'єднання (г)

Значеннями лінгвістичної змінної можуть бути нечіткі змінні, тобто лінгвістична змінна знаходиться на вищому рівні, ніж нечітка змінна. Кожна лінгвістична змінна складається з: назви; множини своїх значень, що також називається базовою терм- множиною T. Елементи базової терм-множини є назвами нечітких змінних; універсальної множини X синтаксичного правила G, за яким генеруються нові терми із застосуванням слів природної або формальної мови; семантичного правила P, яке кожному значенню лінгвістичної змінної ставить у відповідність нечітка підмножина множини X.

Лінгвістичну змінну описує набір <β, T, X, G, M>, де:

– β – найменування лінгвістичної змінної;

– T – множина її значень (терм-множина), що є назвами нечітких змінних, областю визначення кожної з яких є

множина X; множину T називають базовою терм-множиною лінгвістичної змінної;

– G – синтаксична процедура, що дає змогу оперувати елементами терм-множини T, зокрема генерувати нові терми (значення);

– M – семантична процедура, що дає змогу перетворити кожне нове значення лінгвістичної змінної, утвореної процедурою G, на нечітку змінну, тобто сформувати відповідну нечітку множину.

1.3 SQL – Structured query language

За допомогою SQL ми можемо взаємозаміно використовувати мову запитів і програмування, тож багато розробників називають SQL своєрідною мовою програмування, оскільки SQL-двигун містить: компілятор, що компілює команди запитів у процедури та віртуальну машину, що виконує ці процедури [17]. Концепція двигуна SQL та виконання запиту роблять його мовою програмування.

SQL розшифровується як «Структурована мова запитів», яку можна називати або мовою програмування, або мовою запиту, головне призначення SQL – взаємодія з реляційною базою даних, в якій зберігаються дані в табличній формі. SQL може керувати великою кількістю даних, особливо якщо дані записуються одночасно і у нас є багато переходів над цими даними.

При всіх змінах SQL залишається найпоширенішим лінгвістичним засобом взаємодії прикладного програмного забезпечення з базами даних. У той самий час сучасні СУБД, і навіть інформаційні системи, використовують СУБД, надають користувачеві розвинені засоби візуального побудови запитів.

Коли користувач використовує SQL для управління даними, то отримує можливість створювати, отримувати, оновлювати та видаляти дані між

різними базами даних. Існують різні системи управління реляційними базами даних (СУБД), такі як FireBird, MySQL, SQLite, Postgres SQL [18].

1.4 Операції реляційної алгебри засобами мови SQL

Мова SQL стала фактично стандартною мовою доступу до баз даних. Всі СУБД, які претендують на назву «реляційні», реалізують той чи інший діалект SQL [19]. Жодна система не реалізує стандарт SQL в повному обсязі, у всіх діалектах мови є можливості, які не є стандартними. Кожен діалект – це надмножина деякої підмножини стандарту SQL [20].

Мова SQL оперує термінами, дещо відмінними від термінів реляційної теорії, наприклад, замість «відношень» використовуються «таблиці», замість «кортежів» – «рядки», замість «атрибутів» – «колонки» або «стовпці».

Оператор SELECT є фактично найважливішим для користувача і найскладнішим оператором SQL. Він призначений для вибірки даних з таблиць і реалізує одне з основних призначення бази даних – надавати інформацію користувачеві. Оператор SELECT завжди виконується над деякими таблицями, шо входять в базу даних.

Незважаючи на наявність діалектів і відмінностей у синтаксисі, здебільшого тексти SQL-запитів, що містять DDL і DML, можуть бути легко перенесені з однієї СУБД в іншу.

Незважаючи на наявність міжнародного стандарту ANSI SQL-92, багато розробників СУБД вносять зміни в мову SQL, що застосовується в СУБД, що розробляється, тим самим відступаючи від стандарту. Таким чином, з'являються специфічні для кожної конкретної СУБД діалекти мови SQL.

Результатом виконання оператора SELECT завжди є таблиця. За результатами дій оператор SELECT схожий на оператори реляційної алгебри (рисунок 1.7). Будь-який оператор реляційної алгебри може бути

виражений відповідним оператором SELECT. Складність оператора SELECT визначається тим, шо він містить в собі всі можливості реляційної алгебри, а також додаткові можливості, яких в реляційній алгебрі немає.

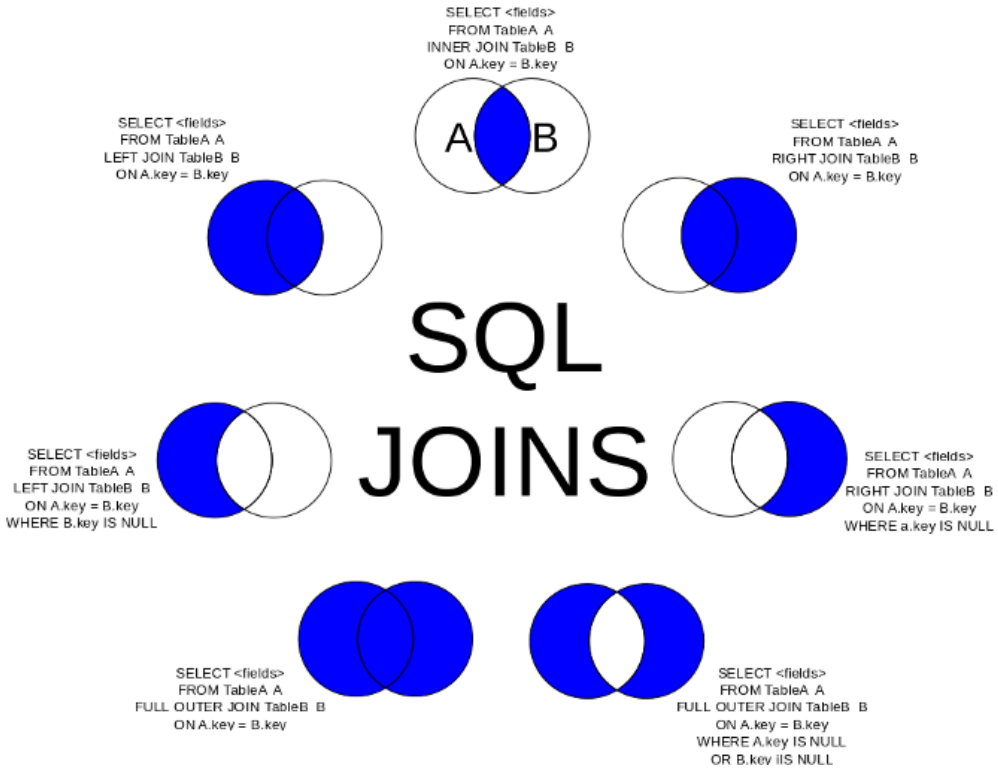

Рисунок 1.7 – Реалізація операцій реляційної алгебри засобами мови SQL

1.5 Постановка задачі дослідження

Як ми бачимо, щоб розробити та дослідити методи підтримки технології нечітких множин за допомогою реляційних систем виконаємо наступні дії. По перше сформулюємо математичний опис нечітких запитів до баз даних, формалізуємо лінгвістичні змінні, та функції приналежності. Проаналізуємо роботу методів нечітких запитів, на основі яких спроектуємо робочу модель, яка буде опрацьовувати дані в таблицях на підставі нечітких запитів.

2 МАТЕМАТИЧНИЙ ОПИС НЕЧІТКИХ ЗАПИТІВ ДО РЕЛЯЦІЙНИХ БАЗ ДАНИХ

Застосування апарату теорії нечітких систем в задачах пошуку інформації переживає в даний час період бурхливого зростання. Питання математичної коректності опису погано формалізується, слабо структурувати інформації та її обробки завжди знаходилися в полі зору фахівців з нечітким системам. Більш того, теорія нечітких множин принаймні на початковому етапі свого розвитку бачилася як засіб опису такої інформації. Не випадково першою нечіткої теорією була теорія лінгвістичної змінної Заде. Природним питанням в рамках обробки нечіткої інформації постало питання пошуку інформації в базах даних, опису об'єктів яких є лінгвістичні опису об'єктів предметної області.

Проблеми пошуку інформації в нечітких базах даних довгий час залишалася на периферії проведених досліджень. І тільки на початку 90-х років вони знову привернули значних інтерес як теоретиків, так і практиків.

2.1 Поняття нечіткого запиту

Більша частина даних, які обробляються в сучасних інформаційних системах, носять чіткий, числовий характер. Однак в запитах до реляційних баз даних, які намагається формулювати людина, часто присутні неточності і невизначеності.

Сучасні СУБД дозволяють працювати тільки з чіткими базами даних і чіткими запитами. В даний час існує проблема розробки математичного опису, методів і засобів обробки даних в умовах нечіткості і невизначеності.

Механізми нечітких запитів (fuzzy queries, flexible queries) [21] до реляційних баз даних базуються на теорії нечітких множин Заде, були вперше

запропоновані в 1984 році і згодом отримали розвиток в роботах Д. Дюбуа і Г. Прада [22].

Поняття нечіткість і невизначеність мають різний зміст.

Невизначеність виникає через неповноту знань, що відносяться до деякого події або наявності властивості будь-якого об'єкта. Нечіткість відноситься до способу опису самої події, властивості.

Нечіткі запити – перспективний напрямок в сучасних системах обробки інформації, вони з'явилися в зв'язку з необхідністю пом'якшити булеву логіку в реляційних базах даних. Даний інструмент дає можливість формулювати запити на природній мові, наприклад, «Отримати список працівників середнього віку із сумою продажів вище середньої порівняно з минулим періодом», або «Вивести список недорогих пропозицій про здавання житла близько до центру міста», що неможливо при використанні стандартного механізму запитів.

Нечіткі запити доцільно використовувати в областях, де здійснюється вибір інформації з бази даних з використанням якісних критеріїв і нечітко сформульованих умов. Наприклад, в прямому маркетингу, підборі об'єктів нерухомості, при виборі туристичних послуг, пошуку вакансій.

Механізм нечітких запитів до реляційних баз даних базується на теорії нечітких множин, яка була вперше запропонована Лотфі Заде [23], [24].

Під середовищем пошуку інформації будемо розуміти пару <запит, база даних>. Існує чотири варіанти нечіткості в середовищі пошуку (таблиця 2.1).

Чітка база даних – сукупність записів, значення атрибутів яких є або строкові, або чисельні значення, однозначно зрозумілі користувачами.

Під чітким запитом розуміється логічне висловлювання, терми якого виражаються звичайними засобами теорії нечітких множин і двозначної чіткої математичною логікою.

Таблиця 2.1 – Варіанти нечіткості в середовищі пошуку

№	Запит	База даних
1	чіткий	чітка
2	нечіткий	чітка
3	чіткий	нечітка
4	нечіткий	нечітка

Це означає, що можна або перерахувати значення ознак цікавлять нас об'єктів, або вказати межі зміни параметрів ознак і зв'язати дані пари <ознака-значення> логічними зв'язками.

Нехай, наприклад, з бази даних потрібно витягти наступну інформацію: «Отримати список молодих співробітників з невисокою заробітною платою», «Знайти пропозиції про здачу не дуже дорогого житла близько до центру міста».

Тут висловлювання «Молодий», «Невисока», «Не дуже дорогий», «Близько» мають розмитий, неточний характер, хоча заробітна плата визначена до гривні, а віддаленість квартири від центру – з точністю до кілометра.

2.2 Лінгвістична змінна

Причиною всьому є те, що в реальному житті ми оперуємо і розмірковуємо невизначеними, неточними категоріями. Такі запити неможливо виконати засобами мови SQL. І на допомогу приходить концепція нечітких запитів.

Нечіткий запит, на відміну від чіткого, може містити терми з нечіткими значеннями. Наприклад, значенням ознаки «Розмір» можуть бути «Великий»,

«Не великий і не маленький»; значенням ознаки «Ціна» – «Дешева», «Більш-менш дорога» і т.п.

Подібним чином чіткі бази даних відрізняються від нечітких: атрибути останніх можуть мати нечіткі значення.

Як видно з таблиці 2.1, найбільш загальною є ситуація 4. Ситуації 2 і 3 є її окремими випадками і, крім того, є симетричними. Ситуація 1 найбільш проста і добре вивчена. Практично всі бази даних належать саме цій ситуації. Продемонструємо обмеженість чітких запитів на наступних прикладах.

Приклад 1. Нехай потрібно отримати відомості про квартиру загальною площею 36 м^2, і вартістю не вище 1300 т.грн. Даний запит можна записати на мові SQL наступним чином:

```
select *from Flat where(Square=36 and Cost <=1300);
```

Квартира площею 37 м^2. і вартістю 1300 т.грн. не потрапить в результат запиту, хоча її характеристики майже задовольняють вимогам запиту.

Приклад 2. Нехай потрібно отримати відомості про менеджерів з продажу не старше 30 років, у яких сума річних угод перевищила 50 т.грн. за таким-то регіону. Даний запит можна записати на мові SQL наступним чином:

```
select FIO from Managers where (Managers.Age <= 30
AND Managers.Sum > 50000 AND Managers.RegionID = 4);
```

Менеджер з продажу 31 років з річною сумою продажів в 70 т.грн, або 29 років з сумою в 49,5 т.грн. не потраплять в результат запиту, хоча їх характеристики майже задовольняють вимогам запиту.

Нечіткі запити допомагають впоратися з подібними проблемами «зникнення» інформації.

Для реалізації таких запитів знадобиться формалізовано описати безлічі значення термів логічного виразу. Моделлю таких структур є лінгвістична змінна і функція приналежності.

Лінгвістичної змінної називається набір

$$\langle \beta, \mathfrak{X}, T, G, M \rangle, \quad (2.1)$$

де β – ім'я лінгвістичної змінної;

$\mathfrak{X}$ – базове безліч;

Т – безліч її значень (терм-множина), що представляють імена нечітких змінних, областю визначення яких є безліч;

G – синтаксична процедура, що дозволяє оперувати елементами терм-множини, зокрема, генерувати нові терми (значення);

М – семантична процедура дозволяє перетворювати нові значення лінгвістичної змінної, утвореної процедурою, в нечітку змінну, тобто сформувати відповідне нечітке безліч.

2.3 Функція приналежності

Функцією приналежності називається функція, яка дозволяє обчислити ступінь приналежності довільного елемента x множини $\mathfrak{X}$ до нечіткій множини F.

Далі з'являється проблема визначення функції приналежності. Це питання знаходиться за межами теорії нечітких множин і більше відноситься до теорії експертного оцінювання і методів обробки експертної інформації.

Розглянемо найбільш поширені способи генерації нових лінгвістичних термів на основі базового терм-множини. Це корисно для побудови різноманітних семантичних конструкцій, які посилюють або послаблюють висловлювання, наприклад: «дуже низька ціна», «приблизно середнього віку» і т.д. Для цього існують лінгвістичні модифікатори (linguistic hedges), які посилюють або послаблюють висловлювання. До підсилює відноситься модифікатор «Дуже» (very), до послаблює – «Більш-або-менш», або

«Приблизно», «Майже» (more-or-less), нечіткі множини яких описуються функціями належності виду:

$$\mu_{very}(X) = (\mu(X))^2, \tag{2.2}$$

$$\mu_{more-on-less}(X) = \sqrt{\mu(X)}. \tag{2.3}$$

Для прикладу формалізуємо нечітке поняття «Вік співробітника компанії» (рисунок 2.1). Це і буде назва відповідної лінгвістичної змінної. Задамо для неї область визначення X = [18; 70] і три лінгвістичних терма – «Молодий», «Середній», «Вище середнього». Останнє, що залишилося зробити – побудувати функції належності для кожного лінгвістичного терма. Виберемо трапецеїдальні функції приналежності з наступними координатами:

– «Молодий» = [18, 18, 28, 34],
– «Середній» = [28, 35, 45, 50],
– «Вище середнього» = [42, 53, 70, 70],

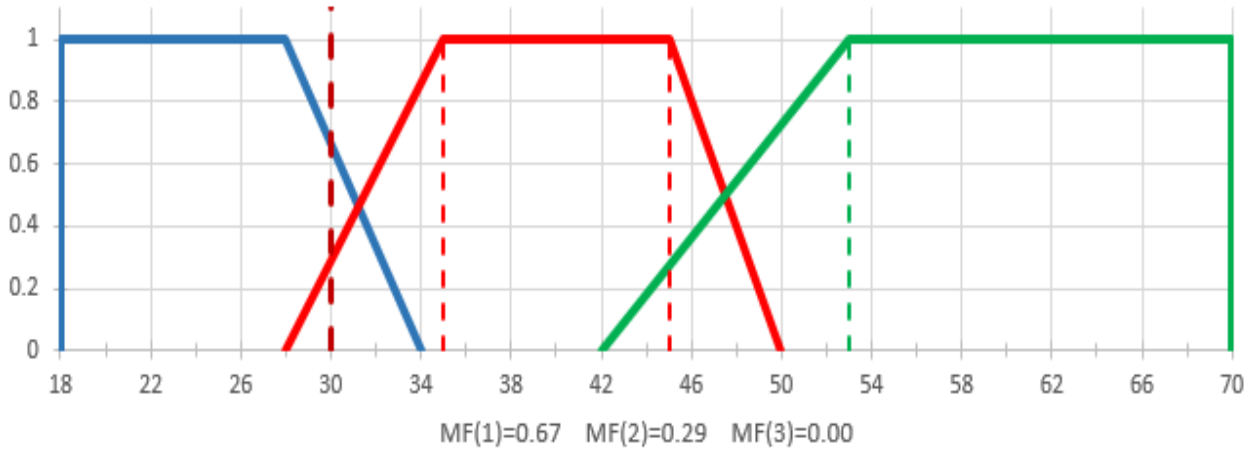

Рисунок 2.1 – Функції належності термів змінної «Вік співробітника компанії»

Основна вимога при побудові функцій належності – значення функцій приналежності повинно бути більше нуля хоча б для одного лінгвістичного терма. На закінчення визначимо операцію нечіткого заперечення (NOT):

$$\overline{\mu(X)} = 1 - \mu(X). \quad (2.4)$$

Наведених вище відомостей достатньо для побудови і виконання нечітких запитів.

Повернемося до прикладу з менеджерами з продажу. Для простоти припустимо, що вся необхідна інформація знаходяться в одній таблиці (таблиця 2.2) з наступними полями: ID – номер співробітника, AGE – вік і SUM – річна сума угод (рисунок 2.2).

Таблиця 2.2 – Приклад сум по співробітникам

ID	AGE	SUM
1	23	120 500
2	25	164 000
3	28	398 000
4	31	489 700
5	33	241 900

Лінгвістична змінна «Вік» була задана раніше. Визначимо ще одну лінгвістичну змінну для поля SUM з областю визначення X=[0; 600 000] і термами «Мала», «Середня» і «Велика» і аналогічно побудуємо для них функції приналежності:

– «Мала» = [0, 0, 0, 200 000];
– «Середня» = [90 000, 180 000, 265 000, 330 000];
– «Велика» = [300 000, 420 000, 600 000, 600 000].

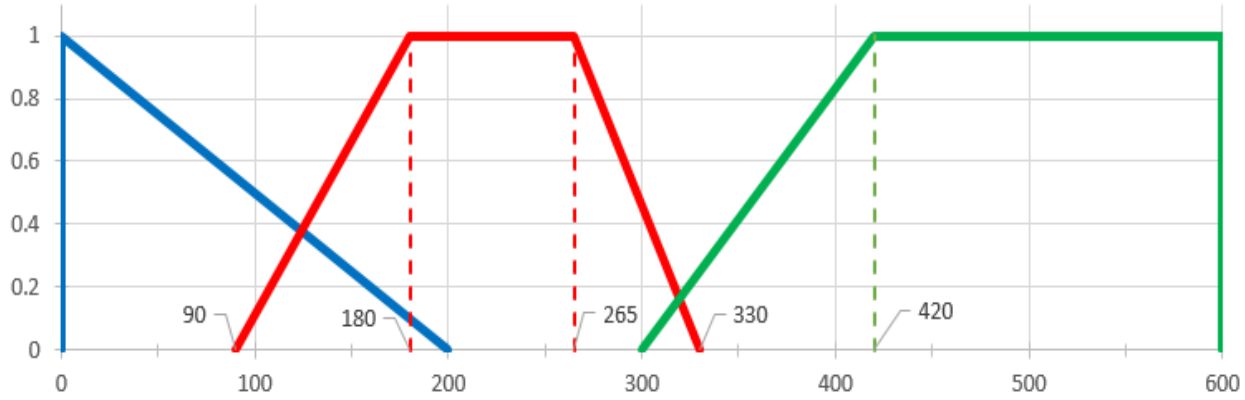

Рисунок 2.2 – Функції приналежності термів змінної «Річна сума угод»

До такої таблиці можна робити нечіткі запити. Наприклад, отримати список всіх молодих менеджерів з продажу з великою річною сумою угод, що на SQL-подібному синтаксисі запишеться так:

```
SELECT * from Managers where (Age = "Молодий"
AND Sum = "Велика");
```

Розрахувавши для кожного запису агреговане значення функції приналежності μ (за допомогою операції нечіткого «І»), отримаємо результат нечіткого запиту (таблиця 2.3).

Таблиця 2.3 – Результат приналежності μ

ID	AGE	SUM	MF
3	28	398 000	0,82
4	31	489 700	0,50

Записи 1,2,5 не були в результат запиту, тому що для них значення функції приналежності дорівнює нулю. Записів, точно задовольняють поставленому запитом, в таблиці не знайшлося. Менеджер з продажу 28 років і річною сумою 398 000 відповідає запиту з функцією приналежності 0,82. На

практиці зазвичай вводять порогове значення функції приналежності, при перевищенні якого записи включаються в результат нечіткого запиту. Аналогічний чіткий запит міг би бути сформульований, наприклад, так:

```
SELECT * from Managers where (Age <= 28 AND Sum> = 420000);
```

Його результат є порожнім. Однак якщо ми трохи розширимо рамки віку в запиті, то ризикуємо втратити інших співробітників з трохи більшим або меншим віком. Тому можна сказати, що нечіткі запити дозволяють розширити область пошуку відповідно до початково заданими людиною обмеженнями.

Використовуючи нечіткі модифікатори, можна формувати і більш складні запити (таблиця 2.4):

```
SELECT * from Managers where (Age = "Більш-або-менш Середній"
AND Sum = "Середня");
```

Таблиця 2.4 – Результат обчислення

ID	AGE	SUM	MF
5	33	251 900	0,85

Часто потрібно оперувати лінгвістичними змінними, а нечіткими аналогами точних значень. Для цього існує нечітке відношення «БЛИЗЬКО» (Наприклад, «Ціна близько 20»). Для реалізації подібних нечітких відносин аналогічно будується нечітка множина з відповідною функцією приналежності, але вже на деякому відносному інтервалі (наприклад, [-5; 5]) для уникнення залежності від контексту. При обчисленні функції приналежності нечіткого відносини «Близько Q» (Q – деяке чітке число) виробляють масштабування на відносний інтервал.

Проілюструємо вищесказане на прикладі таблиці з даними про цінні папери (таблиця 2.5). Нехай вона має в своєму складі наступні поля: PRICE (вартість цінного паперу), RATIO (відношення ціни до прибутку, price-

to-earnings ratio), AYIELD (усереднений дохід за останній квартал, average yield,%).

Таблиця 2.5 – Відношення ціни до прибутку

ID	PRICE	RATIO	AYIELD
1	260	11	15,0
2	380	5	7,0
3	810	6	10,0
4	110	9	14,0
5	420	10	16,0

Нехай потрібно знайти цінні папери для покупки не дорожче, з прибутковістю 15% і ставленням ціни прибутку 11. Це еквівалентно наступному SQL-запиту:

```
SELECT * from TABLE where ((PRICE<=150) and (RATIO=11)
and (AYIELD=15));
```

Результат такого запиту буде порожнім. Тоді сформулюємо цей же запит в нечіткому вигляді з використанням відносини "Близько" :

```
SELECT * from TABLE where ((PRICE = "Близько 150")
AND (RATIO = "Близько 11") AND (AYIELD = "Близько 15"));
```

Побудуємо нечітка множина для відносини в відносному інтервалі [-5; 5] (рисунок 2.3). Це буде трапеція з координатами [-2, -1, 1, 2].

Розрахуємо значення нечіткого запиту «Ціна близько 250» для ціни 380. Попередньо поставимо області визначення кожної лінгвістичної змінної: PRICE – [0; 1000], RATIO – [0; 20], AYIELD – [0; 20]. Значення 130 (отримане як різниця між 380 та 250) від масштабуємо на інтервал [-5; 5], отримаємо величину X=1,3 і MF(1,3)=0,7.

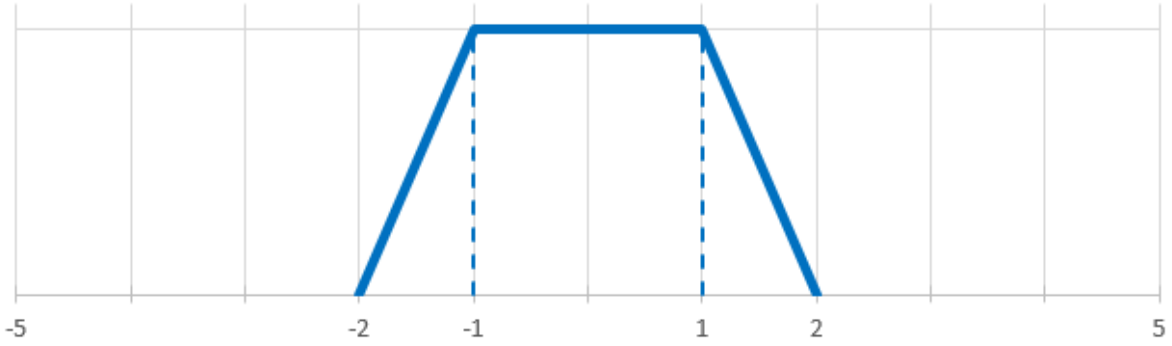

Рисунок 2.3 – Нечітка множина для відносини «Близько»

Застосувавши нечітке відношення БЛИЗЬКО до кожного поля PRICE, RATIO і AYIELD і розрахувавши агреговане значення функції приналежності за допомогою операції нечітке «І», отримаємо наступний результат запиту, який наведено в таблиці 2.6.

Таблиця 2.6 – Результат запиту «Близько 250»

ID	PRICE	RATIO	AYIELD	MF
1	260	11	15,0	1
4	110	9	14,0	0,9

Недоліком нечітких запитів є відносна суб'єктивність функцій приналежності.

2.4 Приклади лінгвістичних змінних, функцій приналежності і нечітких запитів

Розглянемо базу даних «Агентство нерухомості». Припустимо, що вся інформація знаходиться в одній таблиці (таблиця 2.7). Для реалізації нечітких запитів формалізуємо дві лінгвістичні змінні: «ціна» і «площа».

$$\langle \beta, \mathfrak{X}, T, G, M \rangle, \quad (2.5)$$

де β – «ціна»;

$\mathfrak{X}$ = [800; 1800];

T = {«дешева», «середня», «дорога»};

G – синтаксична процедура утворення нових термів;

M – процедура завдання на $\mathfrak{X}$ нечітких множин.

Синтаксична процедура утворення нових термів утворюється за допомогою зв'язок «І» , «АБО» і модифікаторів «НІ», «ДУЖЕ», «БІЛЬШ-МЕНШ», а процедура завдання на $\mathfrak{X}$ нечітких множин A="дешева", B= "середня" і C= "дорога".

Таблиця 2.7 – Данні «Агентство нерухомості»

ID	ADRES	ROOM	SQ	BALKON	COST
1	Lebeda, 54	1	31	0	1300
2	Markovsky, 57	1	24	0	950
3	Belinsky, 3	1	28	0	1380
4	Gorky, 38	1	28	0	1350
5	Gorky, 14	1	32	1	1550
6	Kopylova, 42	1	34	1	1520
7	Ketskhoveli, 57	1	32	1	1450
8	Red Army, 10	1	42	1	1750
9	Kopylova, 78	1	30	1	1630
10	Totmina, 5	1	34	1	1300
11	Kravchenko, 8	1	42	0	1480

Задамо функції приналежності для змінної «ціна» трапецієвидного виду (рисунок 2.4):

$$\mu_A(x) = \begin{cases} 1, x \in [800,950] \\ \frac{1150-x}{200}, x \in (950,1150], \\ 0, x > 1150 \end{cases} \quad (2.6)$$

$$\mu_B(x) = \begin{cases} 0, x < 900, x > 1500 \\ \frac{x-900}{250}, x \in [900,1150) \\ 1, x \in [1150,1300] \\ \frac{1500-x}{200}, x \in (1300,1500] \end{cases}, \quad (2.7)$$

$$\mu_C(x) = \begin{cases} 0, x < 1350 \\ \frac{x-1350}{150}, x \in [1350,1500). \\ 1, x \in [1500,1800] \end{cases} \quad (2.8)$$

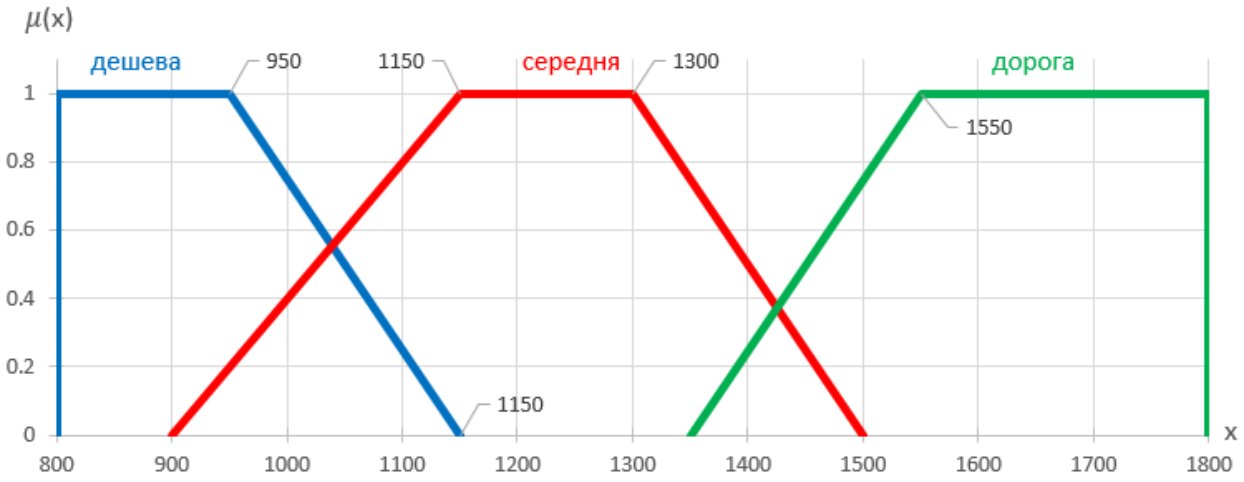

Рисунок 2.4 – Функції приналежності термів змінної «ціна»

Лінгвістична змінна «площа» має вигляд:

$$\langle \gamma, \mathcal{V}, T, G, M \rangle, \quad (2.9)$$

де γ – «площа»;

$\mathcal{V} = [20; 50]$;

$T =$ { «маленька», «середня», «велика» };

G – синтаксична процедура утворення нових термів;

M – процедура завдання на нечітких множин.

Задамо функції приналежності для змінної «площа» трапецієвидного виду (рисунок 2.5):

$$\mu_D(y) = \begin{cases} 1, y \in [2{,}24] \\ \frac{32-y}{8}, y \in (24, 32], \\ 0, y > 32 \end{cases} \quad (2.10)$$

$$\mu_E(y) = \begin{cases} 0, y < 26, y > 43 \\ \frac{x-33}{7}, y \in [26, 33) \\ 1, y \in [33, 38] \\ \frac{43-x}{5}, y \in (38, 43] \end{cases}, \quad (2.11)$$

$$\mu_F(y) = \begin{cases} 0, y < 37 \\ \frac{x-37}{5}, y \in [37, 42) \\ 1, y \in [42, 50] \end{cases}. \quad (2.12)$$

Обчисливши ступеня приналежності для кожної квартири, таблицю можна перетворити до наступного вигляду (таблиця 2.8).

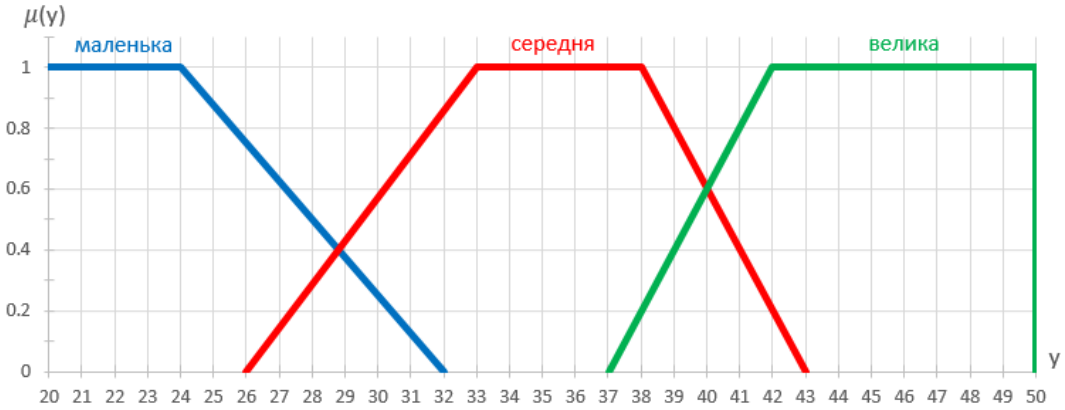

Рисунок 2.5 – Функції приналежності термів змінної «площа»

Таблиця 2.8 – Обчислення ступеня приналежності для кожної квартири

ID	ADRES	ROOM	BALKON	площа			ціна		
				μ_D	μ_E	μ_F	μ_A	μ_B	μ_C
1	Lebeda, 54	1	0	0,13	0,71	0	0	1	0
2	Markovsky, 57	1	0	1	0	0	1	0,2	0
3	Belinsky, 3	1	0	0,5	0,29	0	0	0,6	0,15
4	Gorky, 38	1	0	0,5	0,29	0	0	0,75	0
5	Gorky, 14	1	1	0	0,86	0	0	0	1
6	Kopylova, 42	1	1	0	1	0	0	0	0,85
7	Ketskhoveli, 57	1	1	0	0,86	0	0	0,25	0,5
8	Red Army, 10	1	1	0	0,2	1	0	0	1
9	Kopylova, 78	1	1	0,25	0,57	0	0	0	1
10	Totmina, 5	1	1	0	1	0	0	1	0
11	Kravchenko, 8	1	0	0	0,2	1	0	0,1	0,65

Побудуємо для неї чіткі і нечіткі запити, наприклад «Знайти однокімнатні квартири вартістю 1350 т.грн.» (таблиця 2.8).

```
SELECT * from Flat where(Room=1 and Cost=1350);
```

Таблиця 2.9 – Результат «Однокімнатні квартири вартістю 1350 т.грн.»

ID	ADRES	ROOM	BALKON	площа			ціна		
				μ_D	μ_E	μ_F	μ_A	μ_B	μ_C
4	Gorky, 38	1	0	0,33	0	0	0	1	0

Аналогічний нечіткий запит: «Знайти однокімнатні квартири середньо цінового діапазону» (таблиця 2.10).

```
SELECT *from Flat where(Room=1 and Cost=”середня”);
```

Таблиця 2.10 – Результат «Однокімнатні квартири середньо цінового діапазону»

ID	ADRES	ROOM	BALKON	площа			ціна		
				μ_D	μ_E	μ_F	μ_A	μ_B	μ_C
1	Lebeda, 54	1	0	0,13	0,71	0	0	1	0
10	Totmina, 5	1	1	0	1	0	0	1	0
4	Gorky, 38	1	0	0,5	0,29	0	0	0,75	0
3	Belinsky, 3	1	0	0,5	0,29	0	0	0,6	0,15
7	Ketskhoveli, 57	1	1	0	0,86	0	0	0,25	0,5
2	Markovsky, 57	1	0	1	0	0	1	0,2	0
11	Kravchenko, 8	1	0	0	0,2	1	0	0,1	0,65

А тепер ускладнений нечіткий запит: «Знайти однокімнатні квартири середньою площею з балконом і середньо ціновим діапазоном» (таблиця 2.11).

```
SELECT * from Flat where (Room = 1 and Balkony = 1
and Square = "середня" and cost = "середня");
```

Оскільки два параметра запиту визначені ступенем приналежності, то оператор AND визначається як декартовий добуток нечітких множин.

Декартовий добуток $A_1 \times A_2 \times ... \times A_n$ нечітких множин A_i визначається як нечітка множина A в $\mathcal{X}_i$, $i = 1, ..., n$ декартовом добутку $\mathcal{X} = \mathcal{X}_1 \times ... \times \mathcal{X}_n$ з функцією приналежності виду:

$$\mu_A(x) = \min\{\mu_{A_1}(x_1), \mu_{A_2}(x_2), \dots, \mu_{A_n}(x_n)\}, x = (x_1, \dots, x_{1n}) \in \mathfrak{X}, \quad (2.13)$$

Таблиця 2.11 – Результат «Однокімнатні квартири середньою площею з балконом і середньо ціновим діапазоном»

ID	ADRES	ROOM	BALKON	площа			ціна		
				μ_D	μ_E	μ_F	μ_A	μ_B	μ_C
10	Totmina, 5	1	1	0	1	0	0	1	0
7	Ketskhoveli, 57	1	1	0	0,86	0	0	0,25	0,5

Інструмент нечітких запитів дозволяє узгодити формальні критерії і неформальні вимоги до кола потенційних клієнтів і задавати інтервали вибору потенційних клієнтів як нечіткі множини. В такому випадку клієнти, які не задовольняють якомусь одному критерію, можуть бути обрані з бази даних, якщо вони мають хороші показники за іншими критеріями.

2.5 Нечіткі запити – перспективний напрямок

Нечіткі запити до баз даних (fuzzy queries) – перспективний напрямок в сучасних системах обробки інформації. Даний інструмент дає можливість формулювати запити на природній мові, наприклад: «Вивести список недорогих пропозицій про винаймання житла близько до центру міста», що неможливо при використанні стандартного механізму запитів. Для цієї мети розроблено нечітку реляційна алгебра і спеціальні розширення мов SQL для нечітких запитів. Велика частина досліджень в цій області належить західноєвропейським вченим Д. Дюбуа і Г. Праді.

Нечіткі асоціативні правила (fuzzy associative rules) – інструмент для вилучення з баз даних закономірностей, які формулюються у вигляді

лінгвістичних висловлювань. Тут введені спеціальні поняття нечіткої транзакції, підтримки і достовірності нечіткого асоціативного правила.

Сьогодні елементи нечіткої логіки можна знайти в десятках промислових виробів – від систем керування електропоїздами і бойовими вертольотами до пилососів і пральних машин. Рекламні кампанії багатьох фірм (переважно японських) підносять успіхи у використанні нечіткої логіки як особливу конкурентну перевагу. Без застосування нечіткої логіки немислимі сучасні ситуаційні центри керівників західних країн, у яких приймаються ключові політичні рішення і моделюються всілякі кризові ситуації.

3 ВТІЛЕННЯ РОБОЧОЇ МОДЕЛІ НЕЧІТКИХ ЗАПИТІВ ДО РЕЛЯЦІЙНОЇ БАЗИ ДАНИХ

3.1 Метод побудови нечітких запитів

Метод побудови нечітких запитів до реляційних баз даних для задач пошуку персональної інформації включає в себе наступні етапи:

– формалізація понять користувача;

– пошук інформації;

– аналіз результатів пошуку.

Етап формалізації понять користувача дозволяє ввести новий пошуковий ознака і безліч його значень. Відповідно до Заде [24], під змістом деякого поняття розуміється його обсяг, тобто безліч реальних об'єктів із зазначенням ступеня їх приналежності до поняття.

Іншими словами, завдання сенсу поняття еквівалентно завданням його функції приладдя. При цьому в якості універсальної множини виступає безліч значень деякого атрибута бази даних (домен).

Існує багато різних методів завдання функцій приналежності.

Вибір того чи іншого методу залежить від завдання, існуючої ситуації (наприклад, наявності експертів) та інших параметрів. Відзначимо, що при побудові функцій належності ми можемо побудувати оптимальне безліч значень ознаки з точки зору користувача – безліч, в рамках якого невизначеність при формуванні запиту буде мінімальна.

Отже, результат етапу формалізації понять користувача визначає набір значень ознаки із зазначенням функцій приналежності, їм відповідних. Ці функції приналежності визначені на доменах відповідних атрибутів бази даних. Етап пошуку інформації за нечіткими запитам реалізує наступний алгоритм пошуку інформації за нечіткими поняттями користувача.

Крок 1. Вибирається перший запис в базі даних.

Крок 2. Береться перша пошукова ознака, сформульована в запиті, і її значення.

Крок 3. За ознакою вибирається відповідний атрибут бази даних.

Крок 4. Обчислюється значення функції приналежності, відповідної значенням ознаки, в точці, що відповідає значенню аналізованого атрибута в найденому записі бази даних. Отримане значення функції приналежності запам'ятовується.

Крок 5. Береться наступна пошукова ознака. Кроки 3 і 4 повторюються до тих пір, поки не закінчаться пошукові ознаки. В результаті ми маємо набір значень функцій приналежності значень атрибутів аналізованої записи нечіткого запита.

Крок 6. На базі отриманого в результаті виконання кроку 5 набору значень обчислюється узагальнена оцінка приналежності аналізовані записи запитом. Зокрема, це може бути мінімальне значення із заданого набору, добуток елементів набору і т.п. Отримане узагальнене значення запам'ятовується в робочому полі бази даних.

Крок 7. Вибирається наступний запис в базі даних і повторюється крок 2. Повторення відбувається до тих пір, поки не переберуться всі записи в базі даних.

Таким чином, результат пошуку по нечіткому запиту – упорядкування записів в базі даних за ступенем їх відповідності даним запитом від 1 (повна відповідність) до 0 (повна невідповідність).

Етап аналізу результатів пошуку призначений для уточнення результатів пошуку інформації. Допускаючи невизначеність на вході, ми повинні надати засоби її «зняття» для конкретного користувача на «виході».

Функції приналежності, побудовані в блоці формалізації, відображають або узагальнену думку експертів, або думку якого-небудь експерта в залежно

від обраного методу побудови функції приналежності. Якщо думка користувача і думка експерта (експертів) не збігаються, необхідно надати механізм «індивідуалізації» функцій приналежності. Якщо користувач не задоволений результатом пошуку, він може сформулювати корекцію запиту за конкретною ознакою у вигляді направлення модифікації і модифікатора. Напрямок модифікації вказує полюс ознаки, модифікатор – його «силу». У напрямку модифікації обчислюється напрямок зсуву функцій приналежності використовуваних понять, по модифікатору – крок зсуву. Крок зсуву може обчислюватися різними способами, і це питання вимагає окремого дослідження. Також крок залежить від кількості об'єктів, що знаходяться між існуючою точкою і кінцем універсуму. Якщо таких об'єктів багато – крок вибирається досить великим, якщо мало – досить маленьким. Для отриманих в результаті модифікації функцій приналежності знову повторюється етап пошуку інформації. Таким чином, користувач може «уточнювати» результати пошуку інформації.

Метод побудови нечітких запитів до реляційних баз даних для задач пошуку персональної інформації повинен бути реалізований за допомогою нечіткого лінгвістичного інтерфейсу (НЛІ). Нечіткий лінгвістичний інтерфейс (НЛІ) має 3 блоки:

– блок формалізації понять користувача;

– блок пошуку інформації;

– блок аналізу результатів пошуку.

НЛІ дозволяє:

– вводити користувачеві поняття, що не містяться в описах об'єктів бази даних;

– визначати «сенс» введених понять як нечітких підмножин доменів атрибутів бази даних;

– здійснювати пошук інформації по введеним поняттям;

– уточнювати результати пошуку інформації за рахунок вказівки напряму модифікації і модифікатора сенсу використовуваних понять.

При застосуванні НЛІ користувач отримує «лінгвістичну оболонку» бази даних. Наявність різних класів користувачів, що мають різні функції приналежності, дозволяє створювати і зберігати кілька таких оболонок однієї і тієї ж бази даних, або кілька різних «поглядів» на одну і ту ж інформацію. Цей підхід дозволить значно підвищити ефективність використання існуючих баз даних.

3.2 Навчання SQL обчислювати приналежності термів

За допомогою реляційної бази даних Firebird 3.0 [26] створимо прототип. Спочатку зробимо таблицю, в якій зберігатимемо усі терміни (таблиця 3.1).

Таблиця 3.1 – Зберігання нечітких множин у таблиці «AFFILIATION»

ID	KEYS	NAME	FA	FB	FC	FD

За допомогою значення KEYS будемо групувати множини за належністю до певної властивості чи критерію. А значення полів FA, FB, FC, FD будуть відповідати координатам трапеції (рисунок 3.1).

Реляційна база даних Firebird 3.0 дозволяє створювати процедури і функції. Скористаємось функціоналом та створимо функцію обчислення приналежності значення до шуканого нечіткого подання значення (лістинг 3.1).

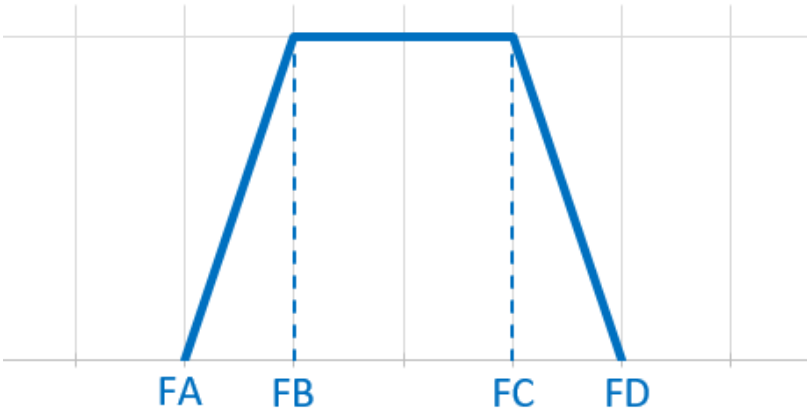

Рисунок 3.1 – Координати нечіткої множини

Лістинг 3.1 – Програмний код функції приналежності

```
create or alter function MF_X (
    KEYS VARCHAR_10_DOMAIN,
    NAME VARCHAR_30_DOMAIN,
    X SUMM_DOMAIN)
returns SUMM_DOMAIN
as
declare variable FF SUMM_DOMAIN;
declare variable FA SUMM_DOMAIN;
declare variable FB SUMM_DOMAIN;
declare variable FC SUMM_DOMAIN;
declare variable FD SUMM_DOMAIN;
begin
  fa=0; fb=0; fc=0; fd=0; ff=0;
  select first 1 fa,fb,fc,fd
    from AFFILIATION a where a.keys=:KEYS and a.name=:NAME
    into :fa,:fb,:fc,:fd;
  if (x<fa) then
    begin
      ff=0;
    end else
```

Продовження лістингу 3.1

```
  if ((x>=fa)and(x<fb)) then
    begin
      if ((fb-fa)<>0) then ff=(x-fa)/(fb-fa);
    end else
  if ((x>=fb)and(x<=fc)) then
    begin
      ff=1;
    end else
  if ((x>fc)and(x<=fd)) then
    begin
      if ((fd-fc)<>0) then ff=(fd-x)/(fd-fc);
    end else
  if (x>fd) then
    begin
      ff=0;
    end
  return ff;
end
```

Вхідні данні KEYS – значення групи множин, щоб виключити непорозуміння, який набір понять використовується для необхідного нечіткого поняття. Наприклад «Висока» може як ціна, так і популярність, тому кожен набір нечітких понять повинні бути згруповані окремо. Значення NAME служить для вказівки самого нечіткого поняття, а X для передачі конкретного значення критерію, для якого потрібно обчислити чи ставитися це до шуканого нечіткого поняття. Результат відносин від 0 до 1 у змінній ff є результатом роботи функції. Але цього недостатньо, бо треба брати до уваги множини, які розташовані поруч і обраховувати переважання їх над поточною, як зображено на рисунку 3.2 .

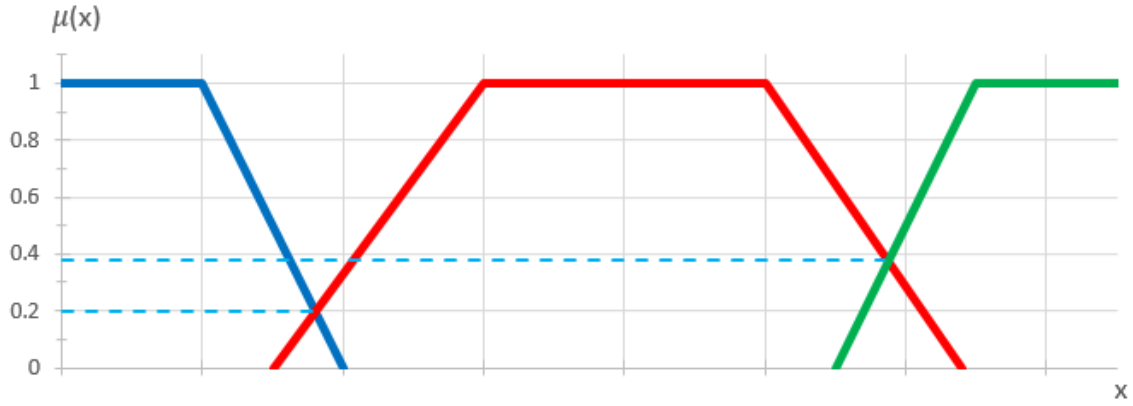

Рисунок 3.2 – Перетин множин

Для вирішення цього питання створимо процедуру SORT_MF, яка виведе всі нечіткі терміни та їх значення приналежності до цього параметра (лістинг 3.2).

Лістинг 3.2 – Процедура виведення $\mu(x)$ усіх термів шуканого значення

```
create or alter procedure SORT_MF (
    KEYS VARCHAR_10_DOMAIN,
    PNAME VARCHAR_30_DOMAIN,
    X SUMM_DOMAIN)
returns (
    NAME VARCHAR_30_DOMAIN,
    OSN INTEGER_DOMAIN,
    Y SUMM_DOMAIN)
as
begin
  for select name from AFFILIATION a
    where a.keys=:KEYS
    into :name do
    begin
      osn=0;
      if (pNAME=name) then osn=1;
```

Продовження лістингу 3.2

```
      Y=MF_X(:KEYS,:name,:X);
      suspend;
    end
end
```

Маючи перелік значень зробимо функцію, яка визначає за старшинством приналежність шуканого терма к значенню (лістинг 3.3).

Лістинг 3.3 – Процедура виведення $\mu(x)$ усіх термів шуканого значення

```
create or alter function MF (
    KEYS VARCHAR_10_DOMAIN,
    NAME VARCHAR_30_DOMAIN,
    X SUMM_DOMAIN)
returns boolean
as
declare variable FF SUMM_DOMAIN;
declare variable OSN INTEGER_DOMAIN;
begin
  ff=0;
  osn=0;
  SELECT first 1 Y,osn
    FROM SORT_MF(:KEYS,:NAME,:X)
    order by Y desc, osn desc
    into :ff,:osn;
  if (osn=0) then ff=0;
  ---
  if (ff>0)
    then return true;
    else return false;
end
```

Тепер за допомогою цієї функції можна здійснювати нечіткі запити.

3.3 Імітаційне моделювання і рішення задач на тестових вибірках

Для того, щоб переконатись в працездатності нашого прототипу, візьмемо таблицю 2.7 та перенесемо ці значення у таблицю FLAT, та заведемо нечіткі множини стосовно ціни та площі.

Значення нечітких множин наведені у таблиці 3.2. Функція MF використовує процедуру сортування, підставивши значення наприклад «ціна 1035» побачимо значення приналежності (рисунок 3.3).

```
SELECT Y,osn,name FROM SORT_MF('Cost', 'середня', 1035)
order by Y desc, osn desc;
```

Таблиця 3.2 – Значення AFFILIATION

ID	KEYS	NAME	FA	FB	FC	FD
1	Cost	дешева	800.00	800.00	950.00	1050.00
2	Cost	середня	1000.00	1150.00	1350.00	1550.00
3	Cost	дорога	1400.00	1500.00	1800.00	1800.00
4	sq	маленька	20.00	20.00	24.00	30.00
5	sq	середня	29.00	33.00	38.00	40.00
6	sq	велика	39.00	42.00	50.00	50.00

Перевіримо ситуацію, коли значення належить обом нечітким виразам, наприклад «ціна 1030».

```
SELECT Y,osn,name FROM SORT_MF('Cost', 'середня', 1039)
order by Y desc, osn desc;
```

На рисунку 3.4 бачимо, наша процедура відпрацювала правильно.

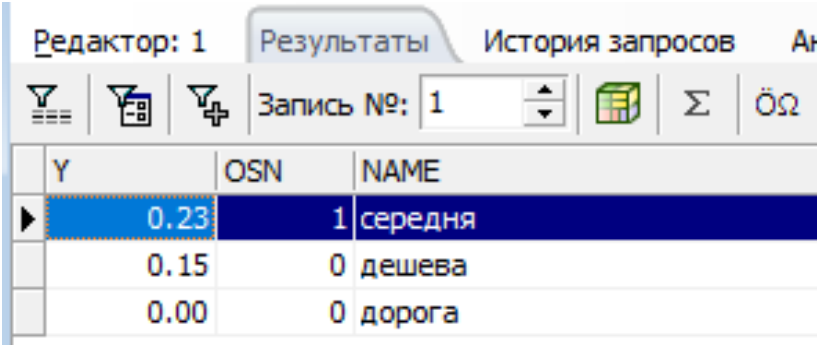

Рисунок 3.3 – Значення приналежності до ціни 1035

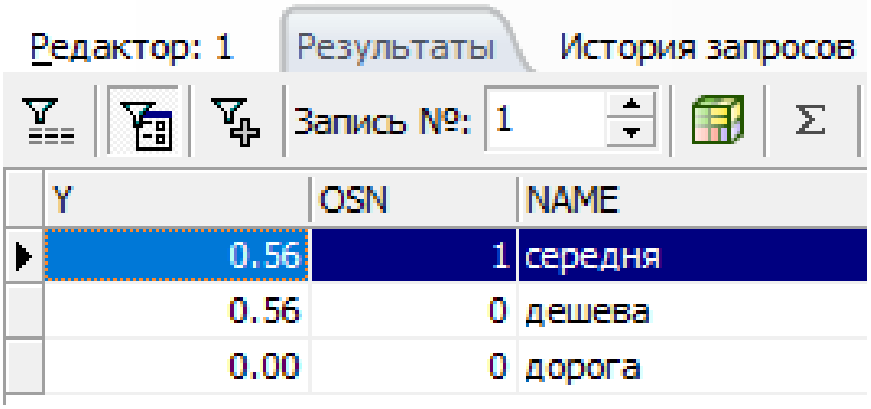

Рисунок 3.4 – Значення приналежності до ціни 1039

Тепер розглянемо приклад «ціна 1020».

```
SELECT Y,osn,name FROM SORT_MF('Cost', 'середня', 1020)
order by Y desc, osn desc;
```

Результат обчислення «ціна 1020» бачимо на рисунку 3.5 який нам показує, що значення 1020 відноситься як до середньої, так і до дешевої ціни, але дешева ціна переважає. Бачичи дані результати, можемо перейти до запитів вибірок даних (рисунок 3.6).

```
SELECT * from flat where mf('Cost','середня',flat.cost);
```

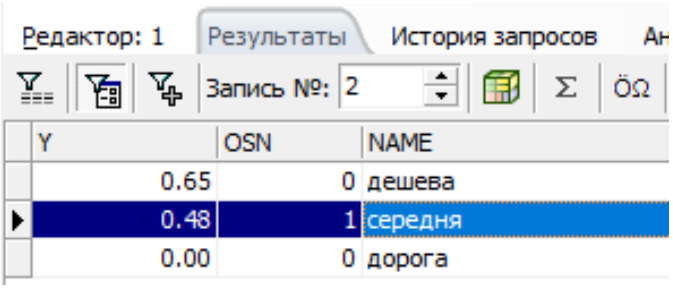

Рисунок 3.5 – Значення приналежності до ціни 1020

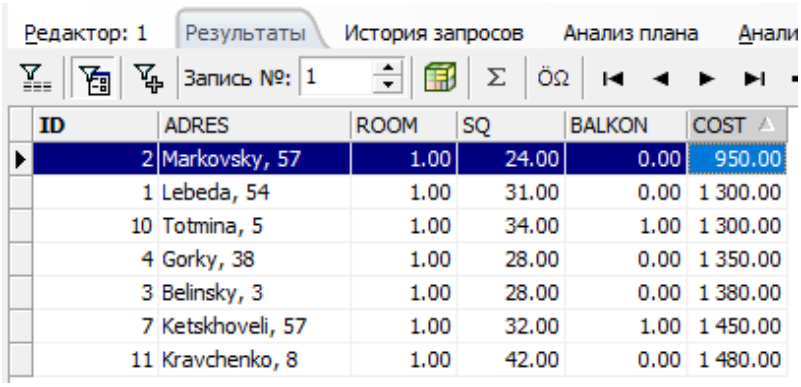

Рисунок 3.6 – Вибірка усіх квартир середньої цінової категорії

Тепер ускладнимо завдання та організуємо вибірку за двома поняттями: середня ціна та середня площа (рисунок 3.7).

```
SELECT * from flat where mf('Cost','середня',flat.cost)
and mf('sq','середня',flat.sq);
```

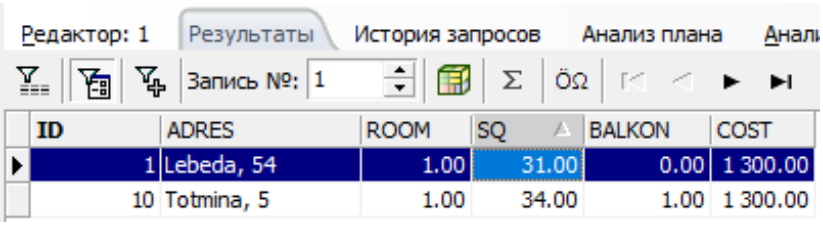

Рисунок 3.7 – Вибірка усіх квартир середньої цінової категорії із середньою площею

3.4 Дослідження працездатності та ефективності застосування нечітких запитів у реальних базах даних

Для цього візьмемо Dataset з відкритих даних «Melbourne Housing Snapshot» [27] з онлайн ресурсу Kaggle. Це знімок набору даних, створеного Тоні Піно. Його вилучено з загальнодоступних результатів, які щотижня публікуються на Domain.com.au. Набір даних включає адресу, тип нерухомості, передмістя, спосіб продажу, кімнати, ціну, агента з нерухомості, дату продажу та відстань від центру міста (таблиця 3.3).

Таблиця 3.3 – Структура Dataset-а

№	Стовпчик	Розмірність	Примітка
1	2	3	4
1	Suburb	String	Передмістя
2	Address	String	Адреса
3	Rooms	Integer	Кількість кімнат
4	Type	String	Тип нерухомості
5	Price	Decimal	Ціна
6	Method	String	Метод
7	SellerG	String	Агент з нерухомості
8	Date	Date	Дата продажу
9	Distance	Decimal	Відстань від центру
10	Postcode	Decimal	Поштовий індекс
11	Bedroom2	Integer	Кількість спалень (з іншого джерела)
12	Bathroom	Integer	Кількість санвузлів
13	Car	Integer	Кількість місць для автомобілів
14	Landsize	Decimal	Розмір землі
15	BuildingArea	Decimal	Площа забудови

Продовження таблиці 3.3

1	2	3	4
16	YearBuilt	Integer	Рік побудови
17	CouncilArea	String	Керівна рада району
18	Lattitude	Decimal	Широта
19	Longtitude	Decimal	Довгота
20	Regionname	String	Назва регіону
21	Propertycount	Decimal	Кількість об’єктів нерухомості, які існують у передмісті.

У нашому розпорядку 13378 унікальних записів (рисунок 3.8).

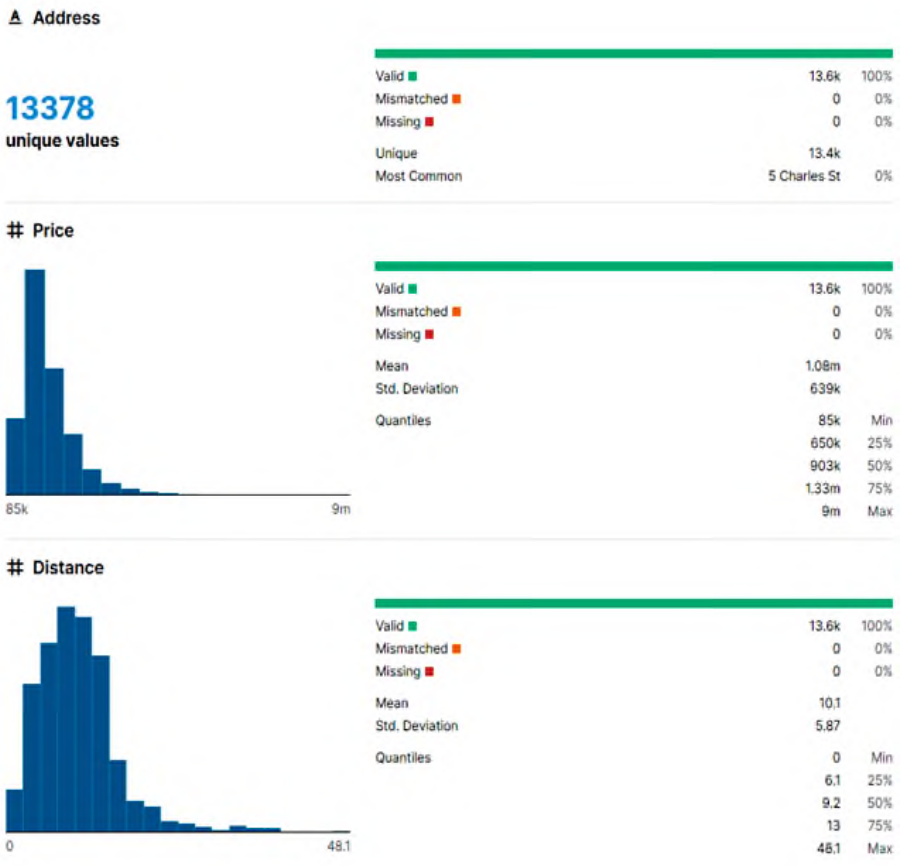

Рисунок 3.8 – Основні значення цікавих полів для тестування

Створимо лінгвістичну змінну ціни у вигляді класичного розподілу на дешеві, середні та дорогі (рисунок 3.9).

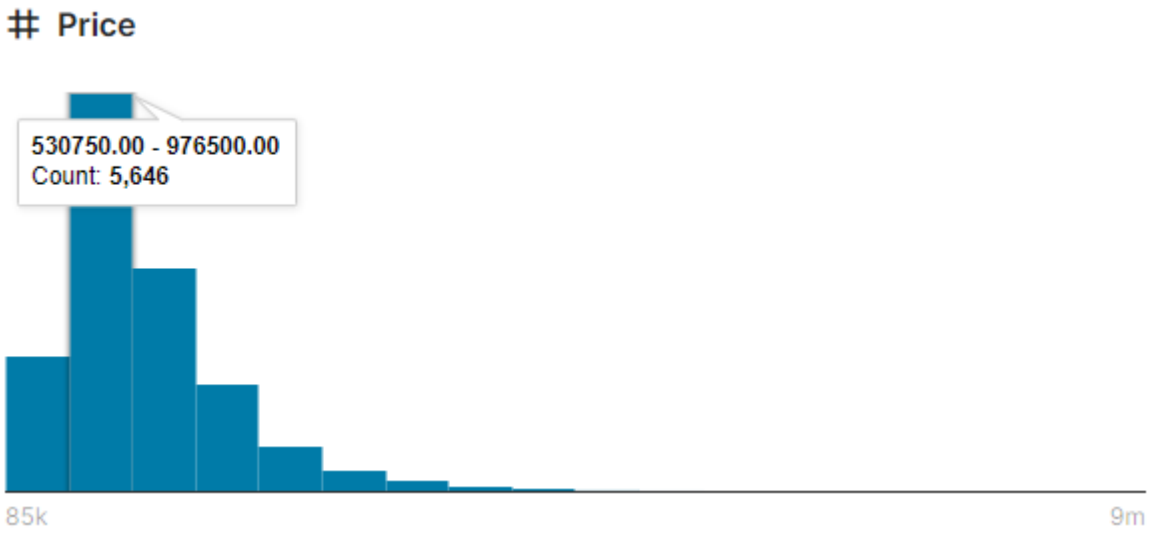

Рисунок 3.9 – Середнє значення ціни

Скориставшись середнім значенням рисунку 3.9 побудуємо лінгвістичні змінні ціни.

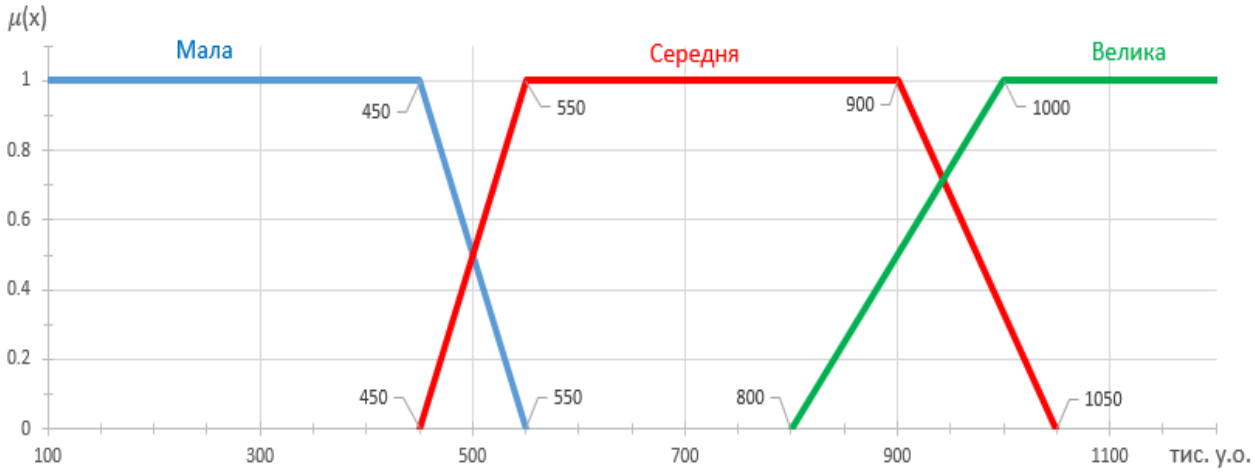

Рисунок 3.10 – Лінгвістичні змінні ціни

Спочатку виконаємо простий запит, без нечітких понять, використовуючи «з» та «по» наведений у лістингу 3.4. Щоб отримати загальну картину, виберемо одразу всі три інтервали та оцінимо скільки рядків потраплять у декілька інтервалів.

Лістинг 3.4 – Запит звичайної вибірки ціни

```
SELECT count(*),
(select count(*) from MELB_DATA a where a.price<=550000),
(select count(*) from MELB_DATA b where b.price>=450000 and
b.price<=1050000),
(select count(*) from MELB_DATA c where c.price>=800000)
from MELB_DATA;
```

Після цього скористаємося функцією нечіткої вибірки, поданий у лістингу 3.5.

Лістинг 3.5 – Запит нечіткої вибірки ціни

```
SELECT count(*),
(select count(*) from MELB_DATA a where mf('Cost','дешева',
a.PRICE)),
(select count(*) from MELB_DATA b where mf('Cost','середня',
b.PRICE)),
(select count(*) from MELB_DATA c where mf('Cost','дорога',
c.PRICE))
from MELB_DATA;
```

Отриманий результат бачимо у таблиці 3.4, який свідчить, що звичайний підхід має багато зайвих рядків під час вибірки.

Середнє значення відстані до центру зображено на рисунку 3.11.

Таблиця 3.4 – Результат отримання ціни обома способами

Кількість рядків	Усього	1. Дешева	2. Середня	3. Дорога	1+2+3	Дубляжі	% помилки
Чіткий підхід	13580	2148	7224	6314	15686	2106	15,51
Нечіткий підхід	13580	1598	5718	6343	13659	79	0,58

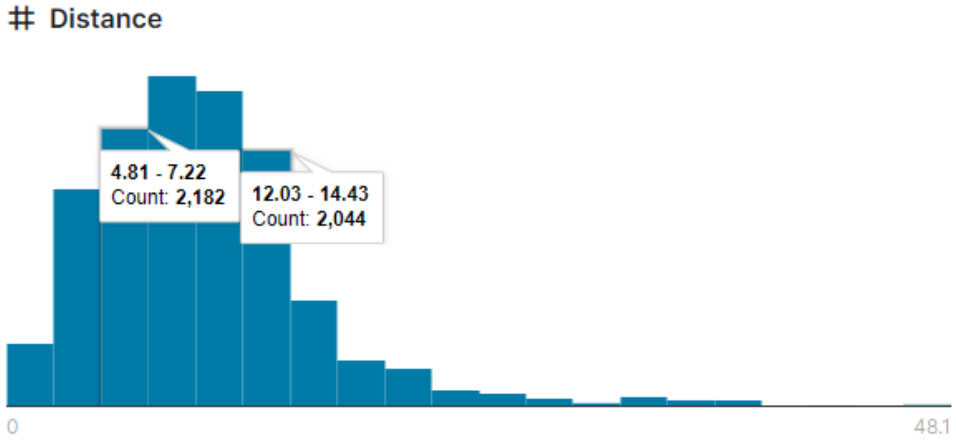

Рисунок 3.11 – Середнє значення відстані до центру

Тепер проведемо теж саме, тільки з відстанню до центру міста. Розкладемо відстань теж на три простих нечітких змінних, результат бачимо на рисунку 3.12.

Рисунок 3.12 – Лінгвістичні змінні відстані

Спочатку виконаємо простий запит, без нечітких понять, використовуючи «з» та «по» наведений у лістингу 3.6. Щоб отримати загальну картину, виберемо одразу всі три інтервали та оцінимо скільки рядків потраплять у декілька інтервалів.

Лістинг 3.6 – Запит звичайної вибірки відстані

```
SELECT count(*),
(select count(*) from MELB_DATA a where a.distance<=6),
(select count(*) from MELB_DATA b where b.distance>=5 and
b.distance<=12),
(select count(*) from MELB_DATA c where c.distance>=11)
from MELB_DATA;
```

Після цього скористаємося функцією нечіткої вибірки, поданий у лістингу 3.7.

Лістинг 3.7 – Запит нечіткої вибірки відстані

```
SELECT count(*),
(select count(*) from MELB_DATA a where mf('dst','центр',
a.distance)),
(select count(*) from MELB_DATA b where mf('dst','середина',
b.distance)),
(select count(*) from MELB_DATA c where mf('dst','околиця',
c.distance))
from MELB_DATA;
```

Отриманий результат бачимо у таблиці 3.5, який свідчить, що звичайний підхід має багато зайвих рядків під час вибірки.

Таблиця 3.5 – Результат отримання ціни обома способами

Кількість рядків	Усього	1. Центр	2. Середина	3. Околиця	1+2+3	Дубляжі	% помилки
Чіткий підхід	13580	3260	7278	5550	16088	2508	18,47
Нечіткий підхід	13580	2884	6393	4486	13763	183	1,35

Як бачимо, картина повторюється. Тепер змоделюємо запит, який використовує відразу два поняття. Лістинг 3.8 демонструє запит, який перебирає усі комбінації ціни та відстані і обчислює кількість рядків обома способами. Якщо взяти за ідеал нечіткий підхід, то отримаємо відсоток відхилення звичайного методу до загальної кількості рядків. Виконаємо його та отримаємо результати у таблиці 3.6.

Лістинг 3.8 – Запит нечіткої вибірки відстані

```
execute block
returns(
N1 VARCHAR_30_DOMAIN,
N2 VARCHAR_30_DOMAIN,
s1 summ_domain,
s2 summ_domain,
sr summ_domain)
as
declare variable total summ_domain;
declare variable FA1 summ_domain;
declare variable FD1 summ_domain;
declare variable FA2 summ_domain;
declare variable FD2 summ_domain;
begin
  select count(*) from MELB_DATA into :total;
  ---
```

Продовження лістингу 3.8

```
  for select NAME,FA,FD from affiliation where KEYS='dst'
    into :n1,:fa1,:fd1 do
      for select NAME,FA,FD from affiliation
        where KEYS='Cost'
        into :n2,:fa2,:fd2 do
        begin
          s1=0;
          s2=0;
          sr=0;
          select count(*) from MELB_DATA a where
            a.distance>=:fa1 and a.distance<=:fd1
            and a.price>=:fa2 and a.price<=:fd2
            into :s1;
          select count(*) from MELB_DATA a where
            mf('dst',:n1,a.distance) and
            mf('Cost',:n2,a.price)
            into :s2;
          sr=(s1-s2)*100.00/total;
          suspend;
        end
end
```

Таблиця 3.6 – Результат за двома поняттями

Відстань	Ціна	Чіткий підхід	Нечіткий підхід	% помилки
1	2	3	4	5
центр	дешева	586	415	1,26
центр	середня	1406	922	3,56
центр	дорога	2135	1565	4,20
середина	дешева	1083	737	2,55

Продовження таблиці 3.6

1	2	3	4	5
середина	середня	3547	2435	8,19
середина	дорога	4747	3253	11,00
околиця	дешева	869	466	2,97
околиця	середня	3469	2419	7,73
околиця	дорога	2966	1630	9,84

3.5 Аналіз отриманих результатів

За допомогою потужної мови програмування SQL у реляційних базах даних можна спроектувати перетворювач запитів із семантичного (нечіткого) подання питання до чітко сформульованого запиту для бази даних.

Як бачимо в прототип можна закласти безліч нечітких понять, що належать до конкретної якості або критерію, щоб полегшити структуру бази даних.

Можна для кожної властивості створювати окрему таблицю і в ній розміщувати значення лінгвістичних змінних. Але це не буде мати оптимізований вигляд. Тому в даному прототипі для оптимізації використано дуже просте рішення, а саме використання ключа «KEYS» який спростив спосіб заощадження всіляких лінгвістичних змінних. Також при тестуванні чітко проглядається при конвертуванні нечіткого поняття у чітке взаємозв'язок із низкою сформованими поняттями, щоб не перетнутися з іншими.

Провівши аналіз між класичним способом вибірки інтервалів значень та використовуючи нечітку логіку спостерігаємо значне покращення одержуваних результатів. Проаналізувавши роботу нечіткої логіки на 13378 унікальних рядків, вже зменшили з 18%-15% дубляжів рядків до 1,5%-0,5%,

що свідчить про позитивний вплив для використання в реляційних системах нечітку логіку.

Також можна спостерігати, що якщо ввести так званий пріоритет на нечіткі значення одного поняття, то в точках перетину понять буде переважати тільки старший, тим самим виключивши помилкові спрацьовування та призведе результат до 0% похибки.

Аналізуючи роботу бачимо, що нечіткою логікою властива відсутність суворих стандартів. Найчастіше вона застосовується в експертних системах, нейронних мережах та системах штучного інтелекту. Замість традиційних значень Істина та Брехня в нечіткій логіці використовується ширший діапазон значень, серед яких Істина, Брехня, Можливо, Іноді, Не пам'ятаю (Як би Так, Чому б і Ні, Ще не вирішив…). Нечітка логіка просто незамінна в тих випадках, коли на поставлене запитання немає чіткої відповіді (так чи ні; 0 або 1) або наперед невідомі всі можливі ситуації. Наприклад, у нечіткій логіці висловлювання виду «X є велике число» інтерпретується як таке, що має неточне значення, що характеризується деяким нечітким безліччю. Оскільки люди рідко бачать навколишній світ лише у чорно-білому кольорі, виникає потреба у використанні нечіткої логіки»

ВИСНОВКИ

В роботі представлені результати, що є відповідно до поставленої мети рішенням актуальної задачі підтримки технологій нечітких множин засобами реляційних систем.

Проведені дослідження дозволили зробити наступні висновки.

1. Проаналізовано стан проблеми використання нечіткої логіки в перспективних напрямках сучасних систем обробки інформації. Проаналізовано стан сучасної теорії нечіткої логіки, яка підходить для розв’язання завдань пов’язаних з реляційними базами даних.

2. Відповідно до поставленої мети було встановлено основні етапи аналізу та створення прототипу функції нечітких запитів до бази даних.

3. Запропонована діюча модель роботи нечітких запитів на мові програмування SQL у середовищі реляційної бази даних Firebird. Створена модель, яка може містити безліч груп нечітких лінгвістичних змінних.

4. Була вирішена задача по зменшенню кількості помилкових вибірок за рахунок врахування понять, що знаходяться поруч.

5. Запропонована система дозволяє вирішувати задачі, які мають нечітко поставлене завдання. Найчастіше це дуже потрібно в експертних системах, нейронних мережах та системах штучного інтелекту.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Заде Л. Понятие лингвистической переменной и его применение к принятию приближенных решений. М.: Мир, 1976.

2. Дюбуа Д., Прад Г. Теория возможностей. Приложения к представлению знаний в информатике. М.: Радио и связь, 1990.

3. Масалович А. Нечеткая логика в бизнесе и финансах. URL: http://www.tora-centre.ru/library/fuzzy/fuzzy-.htm (дата звернення 10.10.2021)

4. Dubois D., Prade H. Using Fuzzy Sets in Database Systems: Why and How? *Proc. of 1996 Workshop on Flexible Query-Answering systems (FQAS'96)*, Denmark, May 22-24, 1996. P. 89–103.

5. Cordon O., Herrera F., A General study on genetic fuzzy systems. *Genetic Algorithms in engineering and computer science*. 1995. P. 33–57.

6. Kosko B. Fuzzy systems as universal approximators. *IEEE Transactions on Computers*. vol. 43. No. 11. 1994. P. 1329–1333.

7. Смолко Д.С., Черноруцкий И.Г. Система поддержки принятия решения для портфеля ценных бумаг. Сборник докладов I Международной конференции по мягким вычислениям и измерениям (SCM-98). Санкт-Петербург, 1998. Том 2. С. 231–234.

8. Пасічник В.В., Резніченко В.А. Організація баз даних та знань. К.: BHV, 2006. 386 с.

9. Філатов В. О. Мультіагентні технології інтеграції гетерогенних інформаційних систем і розподілених баз даних : автореф. дис. … д-ра техн. наук : 05.13.06. Харків, 2005. 32 с.

10. Филатов В. А., Семенец Р. В. Методы и средства проектирования информационных систем и распределенных баз данных. Вестник Херсонского национального технического университета. 2007. No 4(27). С. 203–207.

11. Фісун М.Т., Дворецький М.Л., Дворецька С.В. Побудова моделей для оптимізації структури бази даних вузла у корпоративних інформаційних системах. Інформаційні технології та комп'ютерна інженерія: Міжнародний науково-технічний журнал Вінницького національно-технічного університету. vol 48, № 2. 2020. С. 52–60.

12. Ribeiro R.A., Moreira A.M. Fuzzy Query Interface for a Business Database. *International Journal of Human-Computers Studies*. Vol. 58. 2003. P. 363–391.

13. Filatov V. Fuzzy models presentation and realization by means of relational systems. *Econtechmod : an international quarterly journal on economics in technology, new technologies and modelling processes*. Lublin ; Rzeszow, 2014. Vol.(3). № 3. P. 99–102.

14. Круглов В.В., Дли М.И. Интеллектуальные информационные системы: компьютерная поддержка систем нечеткой логики и нечеткого вывода. М.: Физматлит, 2002.

15. Savchuk T., Kozachuk A. Development of cloud application efficiency evaluation criterion. *EasternEuropean Journal of Enterprise Technologies*. 2015. № 2 (77). P. 20–26.

16. Круглов В.В., Дли М.И., Голунов Р.Ю. Нечѐткая логика и искусственные нейронные сети. М.: Физматлит, 2001. 224 с.

17. Гринченко Н.Н. Проектирование баз данных. СУБД Microsoft Access. М.: Горячая Линия Телеком, 2004. 240 с.

18. Ковязин А.Н., Востриков С.М. Архитектура, администрирование и разработка приложений баз данных в InterBase/FireBird/Yaffil. М.: Кудиц-образ; Издание 4-е, 2006. 496 с.

19. Дейт, К. Дж. SQL и реляционная теория. Как грамотно писать код на SQL. М.: Символ-плюс, 2017. 480 с.

20. Оппель Э. Дж. SQL. Полное руководство. М.: Диалектика / Вильямс, 2016. 902 с.

21. Леоленков А.В. Нечеткое моделирование в среде MATLAB и fuzzyTECH. СПб., 2003.

22. Schmidhuber, Jürgen. Habilitation thesis: System modeling and optimization, 1993. 150 с.

23. Заде Л.. Принятие решений в расплывчатых условиях // В сб.: Вопросы анализа и процедуры принятия решений / Под ред. И. Ф. Шахнова, с предисл. Г. С. Поспелова. М.: Мир, 1976. С. 172–215.

24. Заде Л. Понятие лингвистической переменной и его применение к принятию приближенных решений: Пер. с англ. Н. И. Ринго под ред. Н. Н. Моисеева и С. А. Орловского. М.: Мир, 1976. 165 с.

25. Бондарь, Александр InterBase и Firebird. Практическое руководство для умных пользователей и начинающих разработчиков. М.: БХВ-Петербург, 2012. 592 с.

26. Симонов Денис. Руководство по языку SQL СУБД Firebird 3.0: Firebird 3.0.4 IBSurgeon, 2018. 719 с. URL: https://firebirdsql.org/file/documentation/reference_manuals/firebird-language-reference-30-rus.pdf (дата звернення 01.10.2021)

27. Melbourne Housing Snapshot. Kaggle. URL: https://www.kaggle.com/dansbecker/melbourne-housing-snapshot/ version/5 (дата звернення 01.10.2021)