КОМП'ЮТЕРНІ НАУКИ

DOI: 10.31319/2519-2884.45.2024.18
УДК 004.652.3

Ялова К.М., к. т. н., доцент, ORCID: 0000-0002-2687-5863, e-mail: yalovakateryna@gmail.com
Бабенко М.В., к.т.н., доцент, ORCID: 0000-0003-1013-9383, e-mail: mvbab@ukr.net
Журавель А.В., здобувач другого (магістерського) рівня, e-mail: andriy.juravel2000@gmail.com
Дніпровський державний технічний університет, м. Кам'янське

Yalova Kateryna, Candidate of Technical Sciences, Associate Professor, Head of the Department of the Systems software
Babenko Mykhailo, Candidate of Technical Sciences, Associate Professor, Associate Professor of the Department of the Systems software
Zhuravel Andrii, master'sdegree student, e-mail: andriy.juravel2000@gmail.com
Dniprovsky State Technical University, Kamianske

ШАБЛОННИЙ МЕТОД ДЛЯ СТВОРЕННЯ КОНСТРУКТОРА РЕЛЯЦІЙНИХ БАЗ ДАНИХ

У роботі представлено результати проєктування природномовного інтерфейсу для взаємодії з базою даних, розробленого як конструктор-генератор реляційних баз даних. За рахунок застосування шаблонного методу керований діалог із користувачем організовується у вигляді системи «питання-відповідь», де інформація від користувача вводиться природньою мовою. Вхідною інформацією для проектування архітектури реляційної бази даних є інформація щодо предметної області, для якої вона проєктується. Мета застосування конструктора — трансформувати текстовий опис предметної області до SQL-запитів, які, зрештою, застосовуються для програмного створення реляційної бази даних. У роботі наводяться математичні моделі предметної області та реляційної бази даних, а також визначено алгоритм конвертування знань про предметну область до структурних елементів бази даних. Результатом роботи конструктора є вихідні абстракції предметної області. На вимогу користувача вони можуть бути представлені у вигляді бізнес-правил предметної області, схеми реляційної бази даних, SQL-запитів на основі ключових слів Create і Insert або файлу бази даних MSSQLServer.

***Ключові слова**: природномовний інтерфейс для взаємодії з базою даних; реляційна база даних; конструктор запитів; шаблонний метод; обробка природньої мови.*

Results of the design process are presented in the paper. The paper presents the results of designing a natural language interface to database, developed as a constructor of relational databases. By using a template-based method, a guided dialogue with the user is organized in a «question-and-answer» format, where information is inputted by the user in natural language. The input information for designing the architecture of the relational database is based on the domain knowledge for which it is being designed. The purpose of the constructor using is to transform the textual description of the domain into SQL queries, which are ultimately used for the programmatic creation of a relational database. The paper provides mathematical models of the domain and the relational database, and defines an algorithm for converting domain knowledge into the structural elements of the database. The result of the constructor's operation is the output of domain abstractions. At the user's request, these abstractions can be presented as business rules for the domain, a relational database schema, SQL queries based on the Create and Insert keywords, or a MSSQLServer database file.

***Keywords**: natural language interface to database; relational database; queries constructor; template-based method; natural language processing.*

Постановка проблеми

Сучасні програмні застосунки, незалежно від області використання — чи то мобільний застосунок для обміну повідомленнями, чи то сервіси для управління компаніями — опрацьовують великий обсяг інформації, яку необхідно зберігати значні проміжки часу. Для зберігання, накопичення та аналізу даних програмних застосунків зазвичай використовують сховища даних. Вибір типу сховища даних здебільшого залежить від розміру програмного застосунку, його функціональності та галузі застосування [1]. Для інформаційних систем (ІС) та великих програмних застосунків обґрунтовано і доцільно використовувати реляційні бази даних (РБД) як сховища даних. РБД — це сховище даних, архітектура якої побудована на зв’язаних таблицях, а всі операції з даними — це операції обробки цих таблиць. Для створення, керування та обробки даних РБД використовують системи управління базами даних (СУБД) і спеціалізовані мови програмування для створення запитів, найчастіше — Structured Query Language (SQL).

Бурхливий розвиток технологій обробки природної мови (Natural Language Processing — NLP), створення та популяризація лінгвістичних моделей, підсилених штучним інтелектом (Artificial Inteligence-Powered Language Models), таких як: ChatGPT, Google Bard, Anthropic Claude, Microsoft Copilot стають науковим підґрунтям для пошуку програмних рішень щодо спрощення написання програмного коду, створення програмних застосунків, інтерфейсів, сховищ даних тощо. Іншими словами — створити програми, які б допомагали створювати інші програми. Прикладом впровадження NLP до процесу розробки програмного забезпечення є обробка текстових команд і їх перетворення на SQL-запити, що дозволяє спростити взаємодію між користувачами та РБД, замінивши необхідність конструювання SQL-запитів на механізм розпізнавання команд до РБД. Ідея використання природної мови замість команд SQL лягла в основу створення підходу «Природномовного інтерфейсу для взаємодії з базою даних» (Natural Language Interface to Database — NLIDB) [2]. Концепція NLIDB полягає в тому, щоб зробити сховища даних та обробку їх даних більш доступними для широкого кола нетехнічних користувачів, зменшити вимоги до рівня компетенцій та знань у галузі інформаційних технологій (ІТ) розробників БД [3]. Кінцевою метою NLIDB є організація спілкування користувачів із БД у діалоговому режимі природною мовою [4]. Розробка ІС та програмних застосунків із використанням NLIDB здійснюється для спрощення користувацьких діалогів та зменшення витрат часу на розробку й обробку даних БД. З одного боку, програмні засоби NLIDB використовують мову SQL, правила нормалізації даних та правила проектування РБД, а з іншого — комунікують із користувачами природною мовою для:

- виконання запитів створення, оновлення, видалення РБД її таблиць і даних мовою Data Definition Language (DDL);

- отримання актуальної, повної, достовірної інформації з РБД як результат виконання згенерованих SQL-запитів.

Аналіз останніх досліджень та публікації

Перші спроби створення NLIDB були здійснені в 60—70х роках, розробники засобів NLIDB намагалися розділити та проаналізувати вхідні текстові повідомлення від користувачів із подальшою конвертацією цього тексту в команди SQL [5]. Першою NLIDB вважається розробка В. Вудса (W. Woods) та Р. Каплана (R. Kaplan) під назвою Lunar, представленою в 1972 році. Наступною значущою системою NLIDB була CHAT-80, створена Ф. Перейра (F. Pereira) та Д. Уорреном (D. Warren), яка була представлена в 1980р. та дозволяла конвертувати англомовні текстові запити до запитів мовою Prolog для організації пошуку даних в БД. Сучасними програмними застосунками NLIDB є Google Cloud’s Big Query, Miscosoft PowerMiscosoft Power BI, Wolfram Alpha, Text-toSQL Models, Caius Query, Supa SQL [6].

До теперішнього часу науковці спрямовували свою роботу на підвищення якості і точності перекладу, оптимізацію користувацьких інтерфейсів для реалізації NLIDB в різних ПО. До найвидатніших науковців в галузі NLIDB відносять: Y. Choi, C. Manning, D.Chen, M. Lapata, P. Liang, S. Ieyer, L. Zettlemouyer, D. Yang та інших. В українському науковому просторі такі науковці як: О. Жолус, С. Мірошніченко, С. Ковальчук, В. Мельничук, Л. Анісімова, В. Іванчук присвятили свої роботи задачам NLP, зокрема NLDB [7—8].

Для реалізації NLIDB використовують два підходи:

1. Правила-орієнтований підхід (Rules-based approach — RBA) [9], мета якого сформувати і застосувати певний набір правил, шаблонів, алгоритмів для перекладу вхідного тексту до SQL-запитів.

2. Нейромережевий підхід (Neural approach — NA) [10], ключовою вимогою якого є застосування нейромережевих архітектур для перекладу текстових речень природної мови до SQL-запитів.

Шаблонний метод (Pattern-based method), який реалізує ідею застосування ключових слів та шаблонів, що дозволяють відповідати на складні запити природньою мовою, є фундаментальним методом RBA. Наукові роботи M. D. Agatonovic, H. Cunningham, Zheng [11], X. Xu, C. Liu, D. Song [12] присвячені розвитку принципів шаблонного методу, створення ефективних механізмів обробки визначених мовленнєвих патернів для ідентифікації команд. Різновидом шаблонного методу є метод синтаксичного розбору (Parsing-based method), а такі науковці як: F. Li, H.V. Jagadish, D. Saha, A. Floratou, K. Sankaranarayanan [13], A. Kopp, D. Orlovskyi, S. Orekhov [14] присвятили свої роботи реалізації ідеї створення та застосування дерева синтаксичного розбору для обробки вхідного тексту.

Незважаючи на суттєвий прогрес, різноманіття наукових досліджень та прикладних розробок, завдання розробки NLIDB залишається актуальним науково-практичним завданням. Обробка невизначеності, розуміння контексту і значна варіативність в поданні команд природною мовою впливають на якість конвертації тексту до програмних команд, що потребує подальших пошуків ефективних рішень та механізмів.

Формулювання мети дослідження

Численні дослідження обгрунтовують ефективність реалізації NLIDB для створення та обробки даних РБД [2—8]. Метою дослідження є проектування NLIDB програмного застосунку у вигляді конструктора-генератора РБД, який організовує діалог із користувачем — діючою особою ПО — таким чином, що в результаті цієї взаємодії проектується архітектура РБД. Знання користувача про ПО перекладаються на запити SQL на основі ключового слова Create для створення РБД та її таблиць. Для досягнення мети дослідження були успішно розв’язані наступні проєктні завдання:

1. Розробка архітектури конструктора-генератора РБД.
2. Створення шаблонів та моделі діалогу з користувачем як засіб отримання знань про ПО.
3. Побудова алгоритму трансформування вербального опису ПО у формалізовану архітектуру РБД.
4. Розробка шаблонних правил для перетворення мовленнєвих команд на SQL-запити.
5. Проєктування та програмна реалізація конструктора-генератора РБД.
6. Тестування та апробація запропонованих проєктних рішень.

Виклад основного матеріалу

Проектування РБД — це трудомісткий процес, що включає такі етапи: аналізу вимог та предметної області (ПО), вивчення даних, необхідних для збереження, їх структуризацію, нормалізацію і представлення у вигляді даних таблиць, а також відтворення зв’язків між ними. Кінцевий результат створення РБД у значній мірі залежить від досвіду та рівня знань проєктувальника. Цей процес регулюється набором принципів та правил, найбільш формалізованими з яких є правила нормалізації даних. Інші принципи описують узагальнені рекомендації щодо вирішення проблем надмірності, цілісності, захищеності та індексації даних. Найбільшою складністю процесу проектування архітектури РБД є перетворення знань про ПО в архітектуру РБД. Предметна область — це частина реальності, яка представляє собою набір пов’язаних між собою сутностей і бізнес-процесів. Математично ПО може бути описана кортежем:

$$DD = \langle E, R, Bp \rangle, \quad (1)$$

де $E=\{e_1,\ldots e_n\}$ — множина сутностей ПО; $R=\{R_1,\ldots R_n\}$ — множина зв’язків між сутностями, $Bp=\{bp_1,\ldots bp_n\}$ — множина бізнес-процесів, які відбуваються в ПО.

Кожна сутність ПО описується набором її кількісних і означних властивостей $P=\{p_1,\ldots p_n\}$:

$$E_i \subseteq p_1\times,\ldots,\times p_n. \quad (2)$$

Зв'язок R_{ij} між сутностями і та j визначає скільки екземплярів і-ї сутності відноситься до екземплярів j-ої, і описується як:

$$R_{ij} \subseteq p_i \times p_j . \quad (3)$$

Множина бізнес-процесів ПО Bp описується I вхідними і O вихідними потоками, L управліннями та M механізмами. Для бізнес-процесу BP_i вони описуються як:

- множина вхідних потоків $I_i=\{i_{i1},\dots i_{in}\}$;
- множина вихідних потоків $O_i=\{o_{i1},\dots o_{in}\}$;
- множина потоків управління $L_i=\{l_{i1},\dots l_{in}\}$, яка описує об'єкти, що регламентують та задають правила здійснення процесів в ПО;
- множина потоків-механізмів процесу $M_i=\{m_{i1},\dots m_{in}\}$, яка описує ресурси, що використовуються або завдяки яким здійснюються бізнес-процеси ПО.

Кожний бізнес-процес ПО може бути описаний як:

$$Bp_i = \langle I_i, O_i, L_i, M_i \rangle . \quad (4)$$

Множина бізнес-процесів описує правила та логіку взаємодії між сутностями ПО в ході їх функціонування.

У свою чергу, РБД описується набором наступних обов'язкових компонент:

- таблиця — набір рядків і стовпчиків, заповнених даними;
- атрибут — іменовані стовпчики таблиці;
- кортеж — рядок таблиці;
- домен — набір доступних значень для одного або декількох атрибутів;
- ключ — унікальний ідентифікатор, що визначає кожний рядок таблиці (первинний ключ) чи використовується для зв'язування таблиць (зовнішній ключ);
- зв'язок — механізм, що визначається своєю множинністю — кількістю кортежів дочірньої таблиці, що відносяться до одного кортежу батьківської.

В узагальненому вигляді математична модель РБД може бути представлена наступним чином:

$$RBD = \langle T, R \rangle , \quad (5)$$

де $T=\{T_1,\dots,T_n\}$ — множина таблиць РБД; $R=\{R_1,\dots,R_n\}$ — множина зв'язків між таблицями РБД. Зв'язки характеризуються множинністю один-до-одного (1:1) та один-до-багатьох (1:N).

Структура таблиць РБД може бути описана як:

$$T = \langle A, D, K \rangle , \quad (6)$$

де $A=\{a_1,\dots,a_i\}$ — множина атрибутів n-тої таблиці РБД; $D=\{d_1,\dots,d_j\}$ — множина доменів n-тої таблиці, яка містить всі значення для кожного і-того атрибуту таблиці; $K=\{k_1,\dots,k_m\}$ — множина кортежів n-тої таблиці, що поєднує всі значення атрибутів для m-го рядка таблиці. Дані таблиці РБД є декартовим добутком доменів кожного атрибуту і визначається як:

$$T \subseteq d_1 \times,\dots,\times d_n . \quad (7)$$

Для кожного і-того атрибута існує відповідний j-тий домен. Припустимо, що дані, які описуються і-тим атрибутом, визначаються як $t(a_i)$, тоді повинно виконуватися наступне обмеження: $t(a_i) \in d_j$.

Крім того, множина атрибутів A може бути представлена у вигляді:

$$A = \langle N, T, TA, TD \rangle , \quad (8)$$

де $N=\{n_1,\dots n_l\}$ — множина назв атрибутів для кожної n-тої таблиці; $TA=\{ta_1,ta_2\}$ — множина різновидів атрибутів, $ta_1=\{ta_{11}, ta_{12}\}$ — множина різновидів ключових атрибутів, що складається зі значень: первинний і зовнішній ключ, ta_2 — атрибут-значення; $TD=\{td_1,\dots,td_c\}$ — множина типів даних атрибутів для кожного n-тої таблиці РБД.

Завдання трансформації знань про ПО в архітектуру РБД відбувається із використанням правил, представлених в табл. 1.

Таблиця 1. Правила трансформації ПО до РБД

Рівень ПО	Рівень РБД
Сутність	Таблиця
Властивість сутності	Атрибут
Екземпляр сутності ПО	Кортеж
Значення властивості сутності	Домен
Зв'язки між сутностями: Один до одного Один до багатьох ……багато до багатьох	Зв'язки між таблицями: 1:1 1:N N:N

Окрім правил, наведених у табл. 1, використовуються правила нормалізації даних за допомогою яких дані приводяться до відповідної нормальної форми (НФ), а саме:

- перша НФ встановлює такі критерії як: всі атрибути таблиць повинні бути атомарними (неподільними), назви атрибутів повинні бути унікальними в межах таблиці, таблиця не повинна зберігати дублікати кортежів, впорядкування атрибутів і доменів є незначущим;

- друга НФ є наступним кроком в процесі нормалізації даних і вимагає попереднього приведення даних до першої НФ. Основною вимогою другої НФ форми є вимога відсутності часткових залежностей в межах однієї таблиці: кожен неключовий атрибут повинен залежати від первинного ключа;

- третя НФ застосовується після приведення даних до другої НФ і визначає правила відсутності транзитивних зв'язків між неключовими атрибутами.

Процес нормалізації даних може бути продовжений до шостої НФ, однак застосування перших трьох НФ форм є достатнім для більшості практичних програмних застосунків [15].

Використання NLIDB усуває необхідність знання правил проєктування РБД і синтаксису мови SQL у користувача. Водночас побудова NLIDB вимагає перенесення процесу проєктування архітектури РБД на сторону програмного застосунку, залишаючи користувачеві лише подання інформації про ПО. Основна проблема реалізації NLIDB полягає в складності зменшення неоднозначності інформації, поданої користувачем природною мовою. Основною перевагою шаблонного методу є його простота і відсутність необхідності в створені складних механізмів синтаксичного аналізу, розбору та інтерпретації вхідного тексту. Узагальнений алгоритм застосування шаблонного методу складається з таких кроків:

1. Визначення шаблонів і шаблонних правил.

2. Синтаксичний розбір вхідного тексту і порівняння його із шаблоном.

3 Трансформування текстових команд у SQL-запити.

Архітектура NLIDB конструктора-генератора РБД складається з двох функціональних модулів:

1. Модуль діалогу з користувачем природною мовою;

2. Модуль генерації SQL-запитів.

Алгоритм діалогової взаємодії користувача з NLIDB конструктором описується такими кроками:

1. Внесення загальних даних про ПО. Користувач вводить інформацію, яка описує загальну структуру та контекст ПО.

2. Циклічне додавання даних про сутності, які необхідно зберегти в РБД. Для кожної сутності здійснюється повторюваний процес внесення інформації, враховуючи, що стартове значення керуючої змінної — це: $i=i_0$. Зміна індексу здійснюється за виразом $i_{n+1}=i_n+k$. Процес триває до виконання умови $i<i_m$, де k — це крок циклу, m — це кількість сутностей.

3. Циклічне додавання характеристик кожної і-тої виявленої сутності. Для кожної сутності здійснюється повторюваний процес додавання її властивостей. враховуючи, що початкове значення керуючої змінної $j=j_0$. Зміна індексу здійснюється за виразом $j_{n+1}=j_n+k$, де k — це крок циклу, p — це кількість властивостей і-тої сутності. Процес триває до виконання умови $j<j_p$ На

цьому кроці застосовуються правила нормалізації даних: вхідні дані приводяться до третьої НФ, додаються первинні ключі, а з підмножин атрибутів формуються нові таблиці.

4. Створення залежностей між таблицями. Вводиться інформація про зв’язки між сутностями ПО та визначається множинність цих зв'язків. Автоматично додаються зовнішні ключі. У разі потреби здійснюється перетворення зв’язку типу «багато-до-багатьох» у типи 1:N або 1:1 із додаткової службової таблиці в РБД.

5. Створення SQL-команд. Застосовується ключове слова Create для генерування команд створення РБД та її складових частин.

6. Генерування діаграми РБД, створення файлу РБД та SQL-запитів. Після завершення попередній кроків система пропонує користувачеві згенерувати діаграми РБД, або файл формату MSSQLServer.

7. Циклічне додавання значень атрибутів кожної і-тої виявленої сутності. На вимогу користувача РБД може бути заповнена даними, для цього використовується ключове слово Insert для створення SQL-команд додавання даних до відповідних таблиць РБД.

8. Повторення кроків 2–7 для корегування, уточнення, виправлення або модифікації первинного варіанту. Знайшовши помилки або неточності у вихідних даних, користувач може повторити кроки 2—7 для внесення корегувань.

Мета діалогу із користувачем в контексті NLIDB конструктора — це формалізація знань про ПО, переданих через користувача. В програмних застосунках, які взаємодіють із користувачами в діалоговому режимі суттєвою проблемою є забезпечення керованості діалогу, зменшення користувацьких помилок введення даних та уникнення тупикових ситуацій, коли застосунок не має алгоритмsd реакції на неочікувані дії користувача. Для спрощення процесу синтаксичного розбору вхідного тексту і формалізації знань про ПО був створений шаблон керуючих питань. Він застосовується для формування керованого діалогу із користувачем. В якості шаблонних правил використовується множина шаблонних відповідей і реакцій застосунку. Діалоговий режим NLIDB конструктора здійснюється в режимі «питання-відповідь», переходи між керуючими питаннями здійснюються в залежності від отриманих відповідей. Завдяки такому підходу NLIDB конструктор може ефективно взаємодіяти з користувачем, обробляти введені даних, зменшувати кількість помилок і непередбачених ситуацій під час діалогу із користувачем.

Створений шаблон керуючих запитань містить запитання трьох типів:

1. Контент-запитання Q_{con}. Результатом відповіді на ці запитання текст, введений з клавіатури. Відповіді дозволяють сформувати множину таблиць T, атрибутів A, кортежів K.

2. Класифікаційні запитання Q_{clas}. Відповіді на ці запитання дозволяють встановити тип зв’язку між сутностями, визначити його множинність, знайти кандидатів на зовнішні ключі, сформувати множину зв’язків R, привести дані до НФ.

3. Запитання про властивості Q_{char}. Такі питання націлені на отримання інформації про тип даних атрибутів сутностей, кількість необхідної пам'яті, обов’язковість їх заповнення. Відповіді на запитання про властивості дають змогу визначити кандидатів на первинні ключі, сформувати множину доменів D, множину типів атрибутів TA та множину типів даних TD.

Для побудови користувацького інтерфейсу використовувалися елементи керування, що замінюють ручне введення даних на вибір даних із запропонованих константних значень. На рис. 1 представлено одну з екранних форм створеного NLIDB конструктора РБД, де користувач відповідає на запитання для формування множини атрибутів таблиць РБД.

Користувач має можливість обрати рівень абстракції РБД як результат роботи конструктора. Кінцевим результатом роботи NLIDB конструктора може бути:

- список бізнес-правил ПО із вербальним визначенням об’єктів РБД, їх атрибутів, відносин та обмежень. Список бізнес-правил — абстракція найвищого рівня. На цьому етапі конструювання РБД користувач може визначити, на скільки адекватними є сформовані конструктором бізнес-правила; якщо виникає потреба, вони можуть бути виправлені, уточнені, додані або видалені;

- діаграма РБД. У цьому випадку користувачеві видається діаграма РБД, представлена у вигляді набору таблиць із зв’язками між ними (рис. 2). Якщо користувач не оголосив атрибут, який може однозначно ідентифікувати кожний запис таблиць РБД, то на стороні конструктора

здійснюється автоматичне додавання первинного ключа. Якщо користувачем визначив зв’язок між сутностями типу «багато-до-багатьох» та зазначив неможливість визначення ієрархічної залежності батько-нащадок, то на стороні конструктора додаються службові таблиці. На цьому етапі абстрагування даних користувач також має можливість внести зміни до опису ПО;

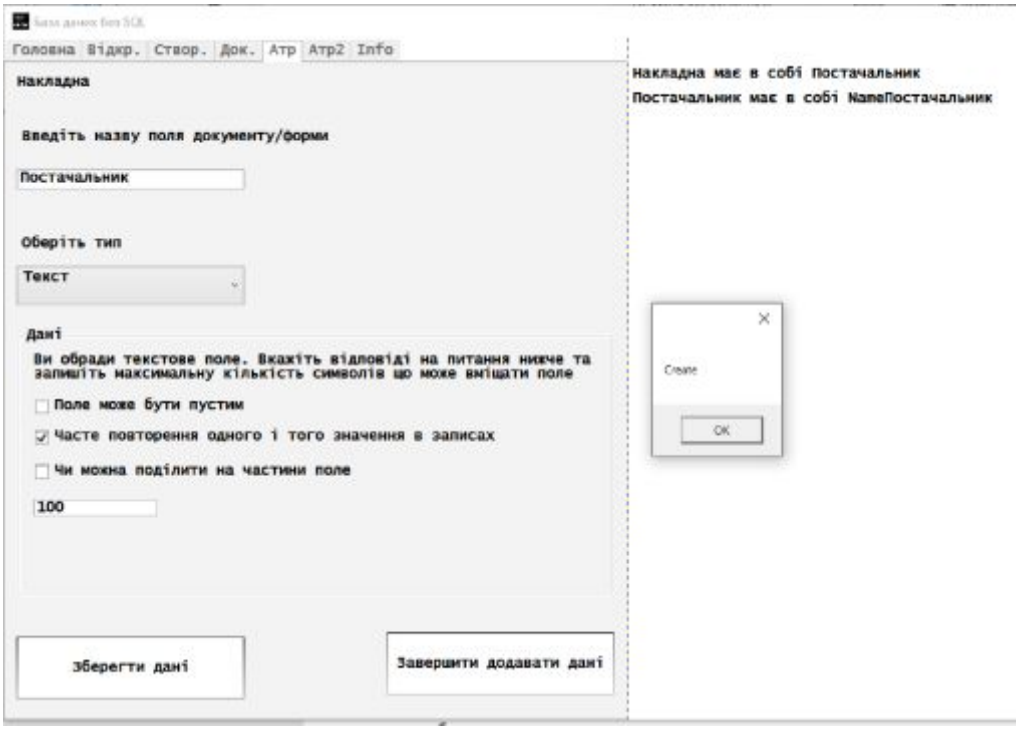

Рис. 1. Екранна форма діалогу з користувачем

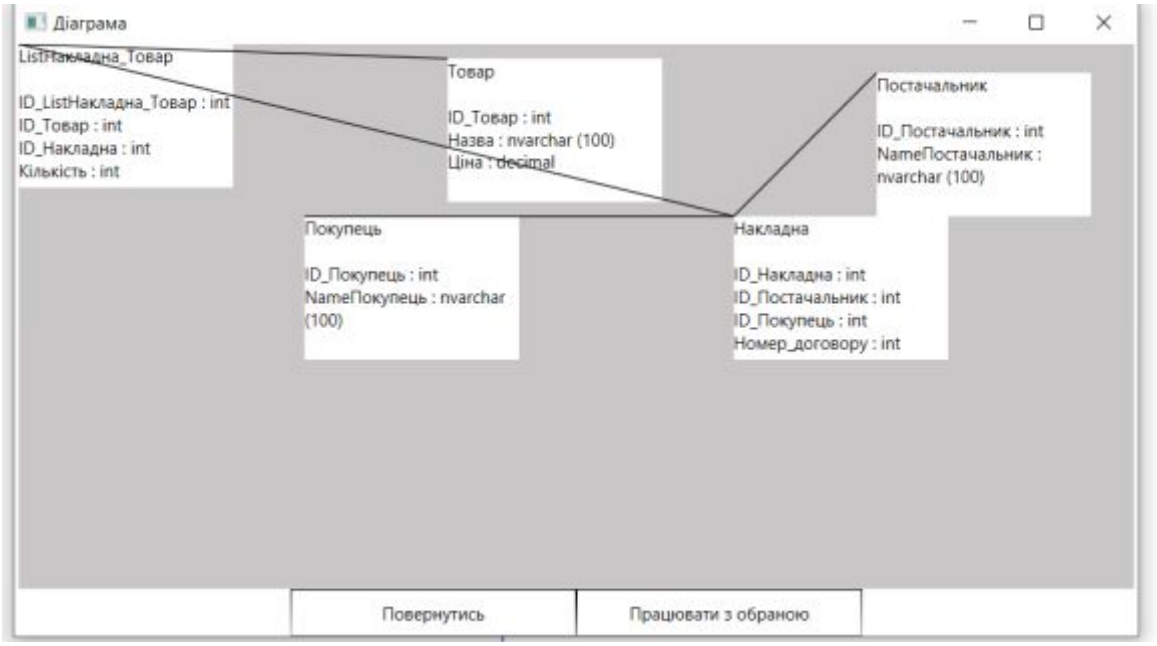

Рис. 2. Діаграма РБД, яка згенерована конструктором

- сформовані SQL-запити, які можна зберегти до файлу запитів. SQL-запити будуються на основі ключового слова Create, узагальненого формату:

CREATE DATABASE<назва_РБД>;
CREATE TABLE <назва_таблиці>
(<назва_атрибута1><тип_даних_атрибута><обмеження>,
…
(<назва_атрибутаN><тип_даних_атрибута><обмеження>)

- .mdf файл РБД. Останнім рівнем абстракції є фізична модель РБД, створена для використання в межах обраної системи управління базами даних (СУБД). Конструктор використовує сформовані SQL-запити і виконує їх за допомогою СУБД MSSQL Server. Файл РБД, створений MSSQL Server, має розширення .mdf і є готовим для подальшого використання. Застосування саме цієї СУБД не є обмеженням для конструктора, оскільки згенеровані SQL-запити є універсальними і можуть бути використані для створення фізичної моделі бази даних у різних СУБД.

На рис. 2 зображено приклад діаграми, згенерованої розробленим конструктором для документу «Накладна на відпуск товаро-матеріальних цінностей».

Останнім кроком у діалозі є заповнення створеної РБД даними, що здійснюється на вимогу користувача. У цьому випадку діалог із користувачем продовжується для отримання даних щодо множини значень кожного атрибуту кожної таблиці РБД. Введена текстова інформація використовується для генерування SQL-команд із ключовим словом Insert формату:

INSERT INTO <назва_таблиці> (<назва_атрибуту1>, ..., <назва_атрибутуN>)
VALUES (<значення1>, ..., (<значенняN>);

Запити на додавання даних також є доступними користувачам для збереження.

Висновки

Досягнення в сфері NLP використовуються для створення програмних застосунків, які передбачають взаємодію з користувачем природною мовою. Найпопулярнішими і найбільш відомими застосунками NLP є віртуальні помічники мобільних телефонів, такі як SIRI або ChatGPT, який розпізнає і генерує текст відповідно до вхідних текстових запитів. Основна мета дослідження, результати якого представлені в цій роботі, полягала в застосуванні методів NLP, зокрема NLIDB, для створення програмного застосунку — конструктора РБД, де в діалоговому режимі з користувачем здійснюється створення РБД та її складових. Розробка РБД є складним, багатоетапним і слабоформалізованим завданням, а недоліки в проектуванні архітектури РБД впливають на якість її функціонування. Застосування програмних засобів автоматичного або автоматизованого проектування РБД може підвищити якість як процесу проектування РБД, так і його кінцевого результату.

У роботі представлено результати проєктування та програмної реалізації NLIDB конструктора-генератора РБД. Для побудови NLIDB застосунку використовувався шаблонний метод, який описує принципи обробки текстової інформації, поданої природною мовою із використанням визначених патернів та шаблонних правил. Для побудови діалогу із користувачем у форматі «питання-відповідь» був створений шаблон питань та алгоритм керування переходами між питаннями. В результаті здійснення керованого діалогу з користувачем знання про ПО трансформуються в складові частини РБД. На кожному етапі проектування РБД користувачеві надається можливість вносити корегування в опис ПО.

Кінцевий ефект від використання NLIDB конструктора-генератора РБД полягає в:

- спрощення процесу проєктування РБД;
- зменшенні вимог до знань розробників РБД щодо теорії баз даних та мови SQL;
- автоматизованому конвертуванні знань про ПО у файл РБД через згенеровані SQL-запити;
- автоматизації нормалізації даних.

Напрямом подальшого дослідження є застосування NLIDB для створення програмного засобу, який трансформує текстові повідомлення в SQL-запити для вибірки даних з РБД на основі ключового слова Select. Розроблений конструктор-генератор РБД може бути застосований як навчальний засіб для підсилення процесу набуття практичних навичок під час проведення лабораторних занять з дисципліни бази даних.

Список використаної літератури

1. Codd E. F. A relational model of data for large shared data banks. *Communications of the ACM*. 2021. Vol. 13(6). P. 377–387.
2. Giordani A., Moschitti A. Translating questions to SQL queries with generative parsers discriminatively reranked. *Data & Knowledge Engineering*. 2015. Vol. 95. P. 189–197.
3. Yaghmazadeh N., Wang Y., Dillig I., Dillig T. SQLizer: Query synthesis from natural language. *ACM on Programming Languages*. 2015. Vol. 1. P. 63–69.
4. Li F., Jagadish H. V. Constructing an interactive natural language interface for relational databases. VLDB Endowment. 2018. Vol. 8(1). P.73–84.
5. Saha A., Florencio D., Cid-Fuentes J. Pigeon: An intuitive SQL optimizer. *International Conference on Management of Data* , 2016. P. 587–602.
6. Zhong V., Xiong C., Socher R. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *arXiv preprint*. 2017. Vol. 1709.00103. P.1–13.
7. Пашкова Н. В. Застосування методів машинного навчання для покращення роботи систем природньо-мовного інтерфейсу до баз даних. *Вісник Національного університету «Львівська політехніка»*. 2018. №895. С. 112–117.
8. Шевченко Д. О., Ковальчук В. П. Система пошуку інформації в базах даних на основі запитів природної мови. *Сучасні інформаційні системи*. 2019. № 3(2). С. 12–18.
9. Yalova K., Yashyna K., Sqlem A.-B. Natural Language Interface to Database Approach in the Task of Relational Databases Design. *CEUR*. 2023. Vol. 3373, P. 320–331.
10. Iacob R. C, Brad F., Apostol E.S., Truica C.O., Hosu I.A. Neural approaches for natural language interfaces to databases: a survey, The 28th International Conference on Computational Linguistics, Barcelona, Spain, 2020, pp. 381–395.
11. Zheng W., Cheng H., Zou L., Yu J., Zhao K. Natural language question/answering: let users talk with the knowledge graph. *The Conference on Information and Knowledge Management*, Singapore, Singapore, 2017, pp. 217–226.
12. Xu X., Liu C., Song D. Sqlnet: Generating structured queries from natural language without reinforcement learning. *Commutating and language*. 2017. Vol. 11 (2017). P. 1–13.
13. Saha D., Floratou A., Sankaranarayanan K., Minhas F., Mittal A. R., Ozcan F. ATHENA: an ontology-driven system for natural language querying over relational data stores. *VLDB Endowment*. 2016. Vol.9(12). P. 1209–1220.
14. Kopp A.M., Orlovskyi D. L., Orekhov S. V. An approach and software prototype for translation of natural language business rules into database structure. *The 5th International Conference on Computational Linguistics and Intelligent Systems*, CEUR WS, Lviv, Ukraine, 2021, pp. 122–129.
15. Amato N. Mastering database normalization: A comprehensive exploration of normal forms. URL:https://www.researchgate.net/publication/374509386_Mastering_database_normalization_A_comprehensive_exploration_of_normal_forms (дата звернення: 16.08.2024).

THE TEMPLATE-BASED METHOD FOR DESIGNING A RELATIONAL DATABASE CONSTRUCTOR

Abstract

The concept of using natural language instead of SQL commands laid the foundation for the development of the Natural Language Interface to Database (NLIDB) approach. Despite significant progress and a variety of research and applied developments, the task of developing NLIDB remains an ongoing scientific and practical challenge. Issues such as handling ambiguity, understanding con-

text, and the significant variability in natural language command expression impact the quality of text-to-command conversion, necessitating further exploration of effective solutions and mechanisms.

Numerous studies have demonstrated the effectiveness of implementing NLIDB (Natural Language Interface to Database) in the creation and management of relational databases (RDBs). The objective of this study is to design an NLIDB software application in the form of an RDB designer, which facilitates a dialogue with the user — an expert in the subject area — in such a way that this interaction results in the design of the RDB architecture. The knowledge provided by the user about the subject area is translated into SQL queries using the «Create» keyword to generate the database and its corresponding tables. To achieve the research objectives, we successfully addressed the tasks of developing the architecture of the RDB designer, creating dialogue templates and a user interaction model, and constructing an algorithm that transforms the verbal description of the subject area into a formalized database architecture.

The architecture of the RDB constructor developed using NLIDB consists of two primary modules: a natural language dialogue module and an SQL query generation module. The guided dialogue with the user begins with the input of general data about the subject area and culminates in the output of either a list of business rules, a database diagram, SQL queries, or an .mdf file, depending on the user's requirements.

The direction of further research is the use of NLIDB to create a software tool that transforms text messages into SQL queries to extract data from a database based on the Select keyword. The developed database constructor can be used as an educational tool to enhance the process of acquiring practical skills during laboratory classes in the databases course.

References

[1] Codd, E. F. (2021). A relational model of data for large shared data banks. Communications of the ACM, 13(6), P. 377–387.

[2] Giordani, A., & Moschitti, A. (2015). Translating questions to SQL queries with generative parsers discriminatively reranked. Data & Knowledge Engineering, 95, P. 189–197.

[3] Yaghmazadeh, N., Wang, Y., Dillig, I., & Dillig, T. (2017). SQLizer: Query synthesis from natural language. Proceedings of the ACM on Programming Languages, 1(OOPSLA), 63.

[4] Li, F., & Jagadish, H. V. (2014). Constructing an interactive natural language interface for relational databases. Proceedings of the VLDB Endowment, 8(1), P. 73–84.

[5] Saha, A., Florencio, D., & Cid-Fuentes, J. (2016). Pigeon: An intuitive SQL optimizer. Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16), P. 587–602.

[6] Zhong, V., Xiong, C., & Socher, R. (2017). Seq2SQL: Generating structured queries from natural language using reinforcement learning. arXiv preprint arXiv:1709.00103.

[7] Pashkova, N. V. (2018). Application of machine learning methods to improve the performance of natural language interface to databases. Bulletin of the National University Lviv Polytechnic., 895, P. 112–117.

[8] Shevchenko, D.O., Kovalchuk, V.P. (2019). Information retrieval dystem in databases based on natural language queries. Modern information systems, 3(2), P. 12–18.

[9] Yalova, K., Yashyna, K., Sqlem, A.-B. (2023) Natural Language Interface to Database Approach in the Task of Relational Databases Design, CEUR, 3373, P. 320–331.

[10] Iacob, R. C., Brad, F., Apostol, E.S., Truica, C.O., Hosu, I.A. (2020) Neural approaches for natural language interfaces to databases: a survey, in: Proceedings of the 28th International Conference on Computational Linguistics, ICCL '20, International Committee on Computational Linguistics, Barcelona, Spain, 2020, pp. 381–395.

[11] Zheng, W., Cheng, H., Zou, L., Yu, L., Zhao, K. (2017) Natural language question/answering: let users talk with the knowledge graph, in: Proceedings of the Conference on Information and Knowledge Management, CIKM '17, Association for Computing Machinery, Singapore, Singapore, 2017, pp. 217–226.

[12] Xu, X., Liu, C., Song D. (2017) Sqlnet: Generating structured queries from natural language without reinforcement learning, Commutating and language, 11, P. 1–13.
[13] Saha, D., Floratou, A., Sankaranarayanan, K., Minhas, F., Mittal, A., Ozcan, R. F. (2016) ATHENA: an ontology-driven system for natural language querying over relational data stores, VLDB Endowment, 9 (12), P. 1209–1220.
[14] Kopp, A.M., Orlovskyi, D.L., Orekhov, S.V. (2021) An approach and software prototype for translation of natural language business rules into database structure, in: Proceedings of the 5th Internation Conference on Computational Linguistics and Intelligent Systems, COLINS ’21, CEUR WS, Lviv, Ukraine, 2021.
[15] Amato, N. (2023) Mastering database normalization: A comprehensive exploration of normal forms URL: https://www.researchgate.net/publication/374509386_Mastering_database_normalization_A_comprehensive_exploration_of_normal_forms

Надійшла до редколегії 11.09.2024