УДК 004.4
DOI: 10.18523/2617-3808.2024.7.11-17

Яремко С. А., Новак В. І.

СИСТЕМА ДЛЯ АВТОМАТИЗАЦІЇ ПРОЦЕСУ НОРМАЛІЗАЦІЇ БАЗ ДАНИХ

У статті розкрито поняття процесу нормалізації баз даних, проведено аналіз наявних інструментів для нормалізації, виділено основні їхні переваги та недоліки. Описано функціонал і реалізацію нової системи для автоматичної нормалізації структури бази даних з урахуванням недоліків наявних систем.

Ключові слова: реляційні бази даних, процес нормалізації, цілісність баз даних, функціональні залежності, алгоритм декомпозиції, властивість з'єднання без втрат, властивість збереження залежностей.

Вступ

У сучасному світі обсяг даних зростає з неймовірною швидкістю, і їхнє ефективне управління стає дедалі важливішим завданням для організацій різних галузей. Реляційні бази даних є одним із ключових інструментів для збереження даних для проєктів, де є важливою структурованість даних, забезпечення цілісності даних, а також підтримка складних запитів і транзакцій. Прикладами таких систем можуть слугувати фінансові системи, системи охорони здоров'я, системи електронної комерції, CRM-системи, ERP-системи та багато інших.

Правильне проєктування структури бази даних відіграє ключову роль у забезпеченні її ефективності, продуктивності та цілісності. Одним із основних методів досягнення цих цілей є нормалізація бази даних.

Процес нормалізації баз даних допомагає організувати дані так, щоби зменшити надлишковість даних, яка потенційно може призвести до виникнення аномалій оновлення, що своєю чергою спричиняє порушення цілісності даних. Цей процес сприяє зменшенню обсягу бази даних і витрат на зберігання даних, підвищенню швидкості виконання запитів і ефективності роботи бази даних.

Однак цей процес є складним завданням, що потребує глибоких знань і досвіду фахівців, тому є актуальним використання систем для автоматизації цього процесу. Сьогодні на ринку є недостатня кількість таких інструментів, тож у цій роботі запропоновано нову систему для реалізації процесу нормалізації бази даних.

Теоретичні аспекти нормалізації баз даних

Нормалізація — це формальний метод аналізу відношень на основі їхнього первинного ключа (або потенційних ключів) і наявних функціональних залежностей [1].

Процес нормалізації полягає у виконанні декомпозиції схем відношень відповідно до алгоритму декомпозиції на декілька відношень так, щоби кожне з результуючих відношень перебувало в певній нормальній формі. Своєю чергою, нормальна форма — це певний набір обмежень, і відношення перебуває в деякій нормальній формі, якщо задовольняє властивий їй набір обмежень. У теорії реляційних баз даних, як правило, виділяють таку послідовність нормальних форм: 1НФ, 2НФ, 3НФ, Нормальна форма Бойса-Кодда (НФБК), 4НФ, 5НФ [2].

Варто зазначити, що в процесі проєктування баз даних повністю усувати надлишковість даних непотрібно, оскільки при цьому неможливо буде підтримувати базу даних як єдине ціле, а нормальні форми є лише вказівками, а не абсолютними вимогами. Інколи постає необхідність відхилитися від них, щоб задовольнити практичні вимоги бізнесу. Є деякі випадки, коли навмисне порушення правил нормалізації є хорошою практикою.

© *Яремко С. А., Новак В. І.*, 2024

Базу даних вважають нормалізованою, якщо її відношення перебувають як мінімум у 3НФ чи НФБК.

Одним із алгоритмів, які використовують для процесу нормалізації, є алгоритм декомпозиції, що базується на теоремі Хіса і дає змогу проводити розбиття до 3НФ та НФБК [5]. Цей алгоритм було вибрано для реалізації процесу нормалізації у заданій системі.

Для того щоб упевнитись у правильності декомпозиції, потрібно перевірити, чи отримана декомпозиція задовольняє властивість з'єднання без втрат [5] і властивість збереження залежностей [6]. Властивість з'єднання без втрат гарантує, що будь-який екземпляр початкового відношення може бути відновлено за допомогою відповідних екземплярів декомпозованих відношень із використанням лише операцій природного з'єднання. Згаданий вище алгоритм декомпозиції гарантує, що результуюче розбиття задовольнятиме цю властивість. Властивість збереження залежностей гарантує, що обмеження, які накладали на початкове відношення, можна підтримувати, застосовуючи ті самі обмеження до кожного з дрібніших відношень.

Ця система на основі поданих на вхід атрибутів реляції та множини функціональних залежностей реалізує процес пошуку мінімальної множини функціональних залежностей, пошук квазі-ключів, зазначений алгоритм декомпозиції до 3НФ, а також явно проводить перевірку виконання властивості з'єднання без втрат і збереження залежностей.

Огляд наявних систем

На ринку нині є такі системи для нормалізації баз даних, кожна з яких має свої переваги й недоліки:

1. Functional Dependencies Checker [4]. Система реалізована у вигляді вебзастосунку з відкритим програмним кодом з використанням програмної платформи Node.js. Інтерфейс інструменту подано на рис. 1.

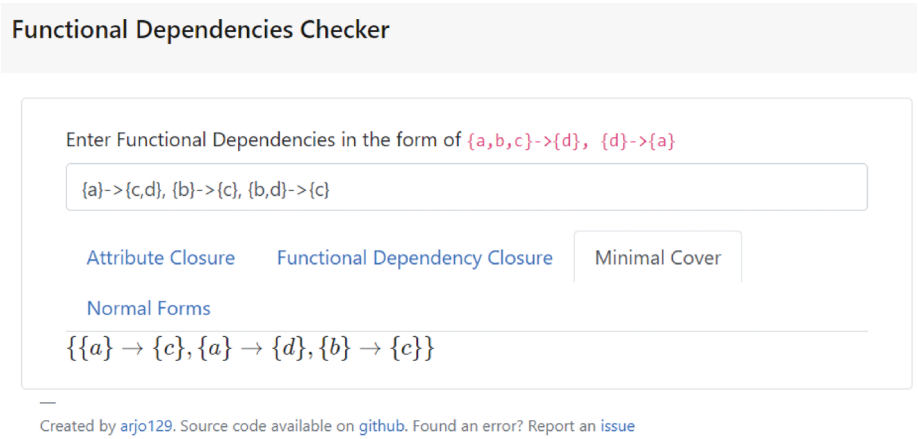

Рис. 1. Інтерфейс системи Functional Dependencies Checker

Цей застосунок дає можливість обчислити замикання усіх можливих комбінацій атрибутів відносно заданої множини функціональних залежностей, знайти потенційні та суперключі, побудувати замикання заданої множини функціональних залежностей, визначити мінімальну множину функціональних залежностей, а також перевірити, чи реляція перебуває у 2, 3 нормальних формах та НФБК. Проте у цьому застосунку не можна провести декомпозицію реляції до певної нормальної форми.

2. Relational Database Tools [7]. Систему реалізовано у вигляді вебзастосунку з відкритим програмним кодом з використанням мови програмування Java. Інтерфейс інструменту подано на рис. 2.

У цій системі можна обчислити замикання усіх можливих комбінацій атрибутів відносно заданої множини функціональних залежностей, знайти потенційні та суперключі, побудувати замикання заданої множини функціональних залежностей, визначити мінімальну множину функціональних залежностей, перевірити, чи ця реляція перебуває у 2, 3 нормальних формах, НФБК, а також 4 нормальній формі, надає можливість зведення реляції до 3 нормальної форми з виконанням властивостей з'єднання без втрат і збереження залежностей та НФБК без гарантії виконання властивості збереження залежностей. З-поміж недоліків системи можна виділити незручний інтерфейс користувача й відсутність зведення реляції до 4 нормальної форми.

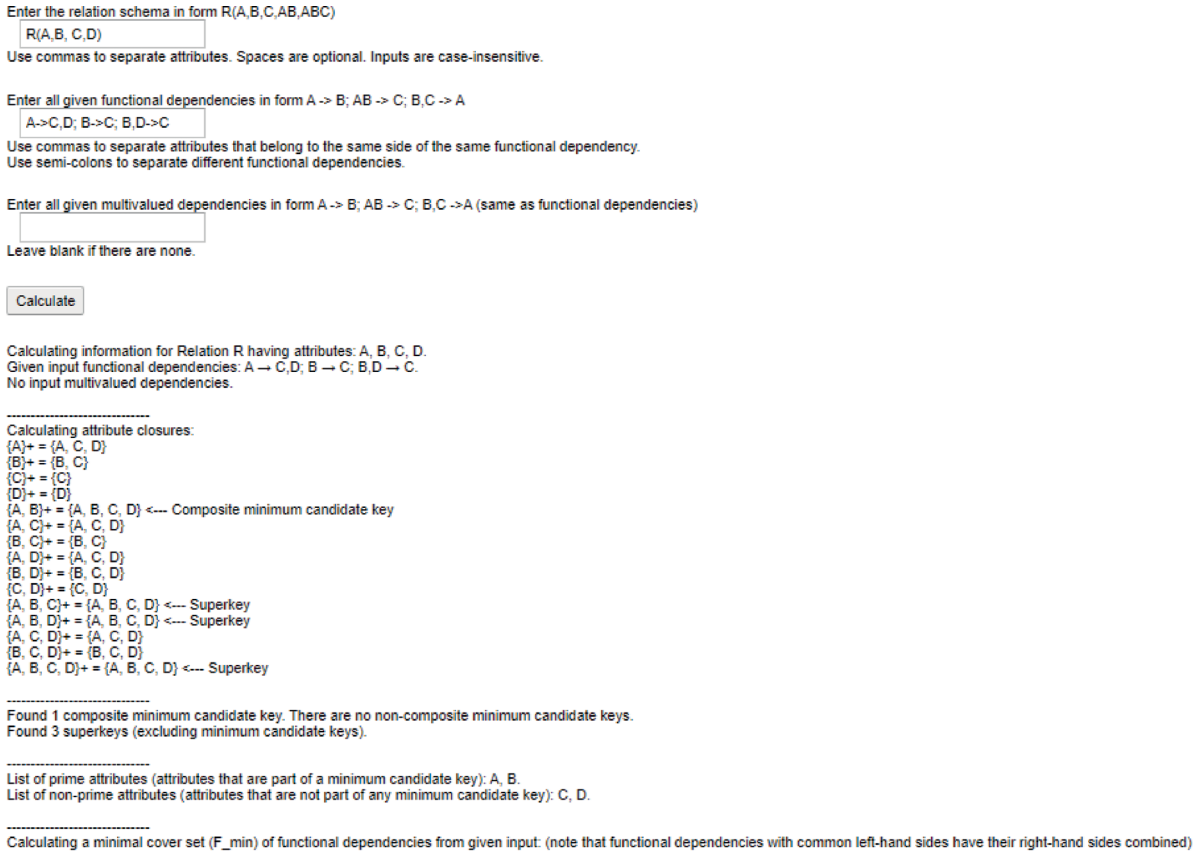

Рис. 2. Інтерфейс системи Relational Database Tools

3. Boyce-Codd relation solver [3]. Систему реалізовано у вигляді вебзастосунку з відкритим програмним кодом з використанням мови програмування JavaScript. Інтерфейс інструменту подано на рис. 3.

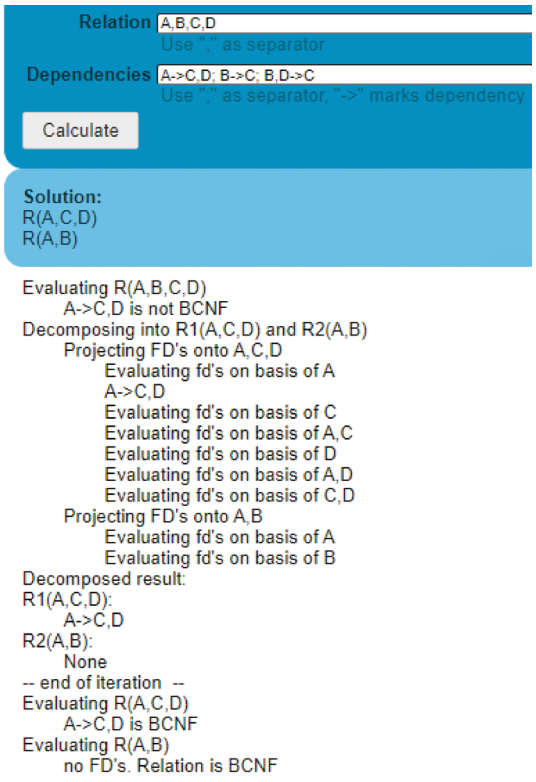

Рис. 3. Інтерфейс застосунку Boyce-Codd relation solver [3]

Функціонал системи дає змогу здійснити лише декомпозицію реляції до НФБК. Усі інші функції, що є доступними у системах, описаних вище, не передбачено.

Реалізація логіки системи

Для реалізації серверної частини системи було використано програмну платформу Node.js.

Основна частина коду, яка створює необхідні для процесу нормалізації об'єкти, складається з декількох класів: Relation, Dependency, Data та NormalizationData. Кожен клас призначений для оброблення певної частини представлення схеми бази даних і операцій, які мають бути виконані в процесі нормалізації.

Серверна частина системи використовує маршрутизатори фреймворку Express.js для ефективного управління HTTP-запитами.

Маршрутизатор CalculationRouter, створений як окремий клас, використовується для групування всіх маршрутів запитів, пов'язаних з обчисленнями нормалізації. Цей маршрутизатор прив'язується до маршруту «/calc».

Кожен маршрут у CalculationRouter налаштовано на оброблення POST-запитів для різних типів операцій нормалізації:

- /min — обчислення мінімальної множини функціональних залежностей;
- /nf — проведення декомпозиції до 3НФ;
- /lossless — перевірка властивості з'єднання без втрат;
- /preserve — перевірка властивості збереження залежностей.

Ці маршрути обробляються відповідними методами в CalculationController, які інкапсулюють логіку для кожного типу обчислень. Сам клас CalculationController містить методи, які безпосередньо взаємодіють із логікою нормалізації, інкапсульованою у класі NormalizedData.

Кожен метод працює за схожим шаблоном:

- отримуються необхідні дані з тіла запиту;
- створюються екземпляри класів Relation і Dependency для кожної реляції і залежності, заданих у запиті;
- створюється екземпляр NormalizedData зі створеними в попередньому пункті реляціями і залежностями;
- виконуються обчислення процесу нормалізації за допомогою методів класу NormalizedData;
- результати переводяться у формат, придатний для відповіді, і вони надсилаються на клієнтську частину для відображення.

```
findMinimalCover() {
    this.#singleAttributeRightPart()
    this.data.dependencies = this.#removeDuplicates(this.data.dependencies)
    this.#removingRedundantDeps()
    this.#removingRedundantLeftDeps()
    while (this.#changedLeft) {
        this.#removingRedundantDeps()
        this.#changedLeft = this.#checkMultipleLefts(this.data.dependencies)
        if (this.#changedLeft) this.#removingRedundantLeftDeps()
    }
}
```

Рис. 4. Метод знаходження мінімальної множини функціональних залежностей

```
get3NF() {
    const minimalCover = this.data.dependencies;
    const attributes = this.#getAllAttrs();
    const quasikey = this.getQuasikey();

    if (!quasikey) {
        console.log("Error: Quasikey not found.");
        return;
    }

    const relations3NF = this.#decomposeTo3NF(minimalCover, attributes, quasikey);
    this.data.relations = relations3NF

}
```

Рис. 5. Метод декомпозиції реляції до 3НФ

Також було проведено автоматизоване тестування системи, що передбачає unit-тестування з використанням фреймворку для тестування JavaScript Jest та ручне тестування коректності роботи застосунку.

Реалізація інтерфейсу користувача

Для реалізації клієнтської частини системи було використано відкриту JavaScript бібліотеку для створення інтерфейсів користувача React.js.

На головній сторінці подано опис можливостей системи, а саме: можливість знаходження мінімальної множини функціональних залежностей, зведення реляції до 3НФ, явна перевірка виконання властивості з'єднання без втрат та збереження залежностей (рис. 6). При натисканні кнопки «Алгоритм» можна побачити опис використаних алгоритмів для реалізації процесу нормалізації (рис. 7).

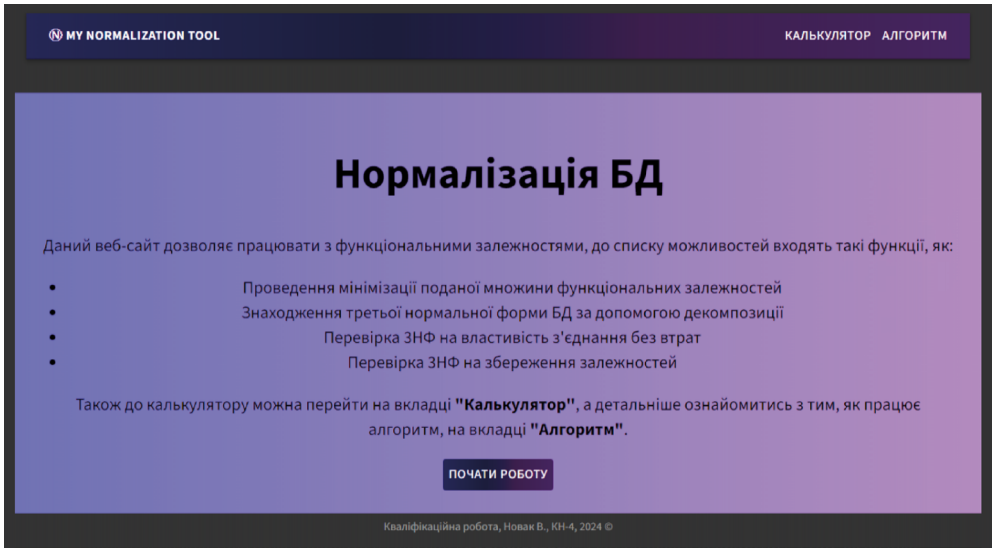

Рис. 6. Головна сторінка клієнтської частини застосунку

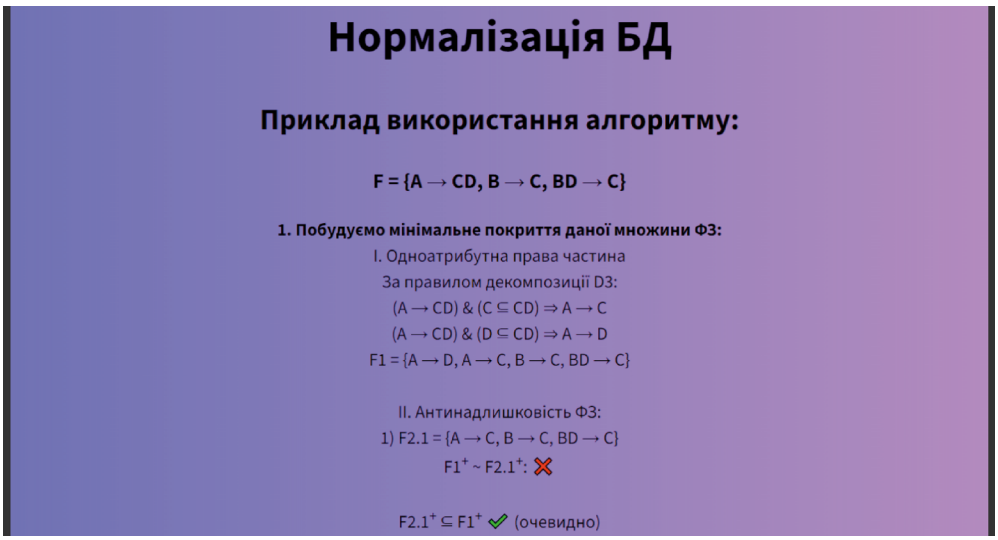

Рис. 7. Опис алгоритмів

При переході на сторінку обчислень «Калькулятор» з'являється набір текстових полів, де перше поле призначене для введення усіх атрибутів реляції, а усі інші пари текстових полів — для введення функціональних залежностей (рис. 8). Клієнт має змогу виконати певну операцію, використовуючи відповідні кнопки на інтерфейсі користувача (рис. 9).

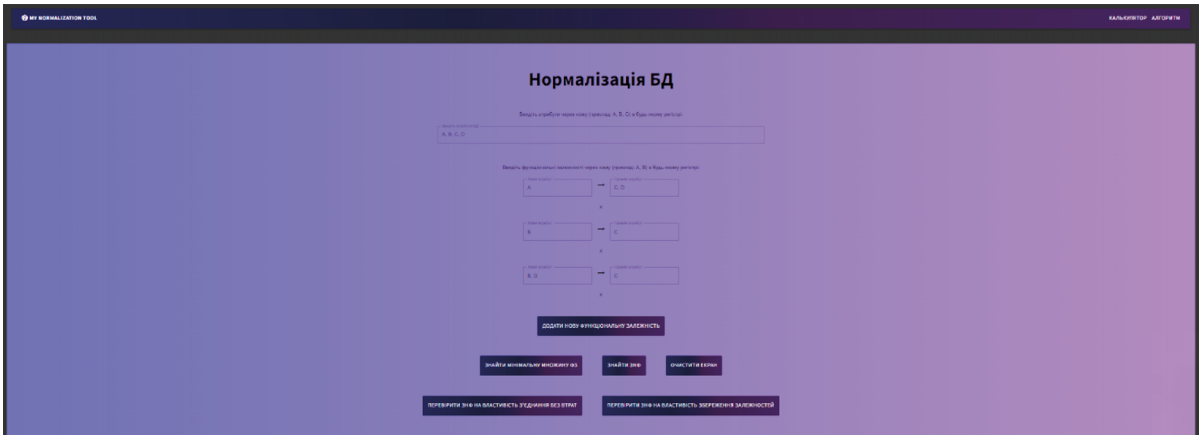

Рис. 8. Сторінка калькулятора

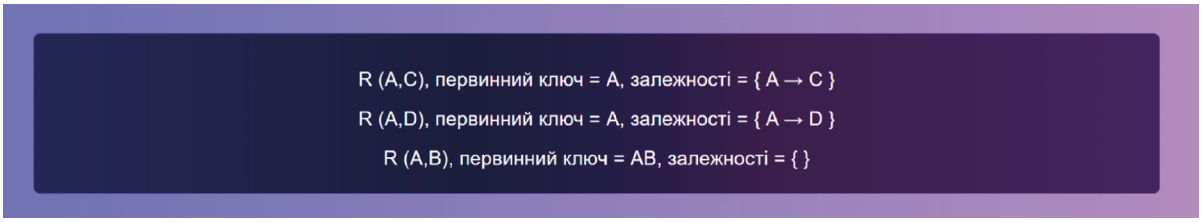

Рис. 9. Зведення прикладу рис. 8. до 3НФ

Висновок

У статті описано основні теоретичні аспекти процесу нормалізації баз даних, розглянуто наявні інструменти, виділено їхні основні переваги й недоліки. Також на прикладах надано опис функціоналу нової системи для нормалізації баз даних, яка враховує деякі недоліки наявних систем.

Ця система не лише є актуальною для розробників баз даних, аналітиків та інших фахівців, що працюють із великими обсягами даних, забезпечуючи швидке та ефективне досягнення оптимальної структури бази даних, спрощуючи підтримку та розширення бази даних, а й може слугувати навчальним інструментом для студентів і новачків у галузі інформаційних технологій, допомагаючи їм зрозуміти процеси нормалізації через можливість розв'язання практичних прикладів та ознайомлення з використаними алгоритмами.

Подальшими кроками покращення системи є реалізація декомпозиції реляції до НФБК та 4НФ.

Список літератури

1. Бардус І. О. Бази даних у схемах (на основі фундаменталізованого підходу) : навч. посіб. / І. О. Бардус, М. І. Лазарєв, А. О. Ніценко. — Харків : Вид-во «Діса плюс», 2017. — 133 с.
2. Гайджари В. І. Основи проектування та використання баз даних : навч. посіб. / В. І. Гайджари, О. А. Дацюк. — 2-ге вид., виправл. і доповн. — Київ : ІВЦ «Видавництво "Політехніка"», ТОВ «Фірма "Періодика"», 2004. — 256 с.
3. Boyce-Cood relation solver [Electronic resource]. — Mode of access: https://gamgi.github.io/js-bcnf/.
4. Functional Dependencies Checker [Electronic resource]. — Mode of access: https://arjo129.github.io/functionalDependencyCalculator/.
5. Garcia-Molina H. Database systems. The complete book / H. Garcia-Molina, J. D. Ullman, J. Widom. — Upper Saddle River, New Jersy : Person Prentice Hall, Person Education, Inc., 2009, 2002.
6. Ramakrishnan, R. Database management systems / R. Ramakrishnan, J. Gehrke. — Singapore : McGraw-Hill Higher Education, 2003.
7. Relational Database Tools [Electronic resource]. — Mode of access: http://raymondcho.net/RelationalDatabaseTools/RelationalDatabaseTools.html.

References

Bardus, I. O., Lazariev, M. I., & Nitsenko, A. O. (2017). *Bazy danykh u skhemakh (na osnovi fundamentalizovanoho pidkhodu)*. Disa plius.
BCNF-js | Boyce-Codd relation solver. (n. d.). gamgi.github.io. https://gamgi.github.io/js-bcnf/.
BCNF-js | Boyce-Codd relation solver. (n. d.). gamgi.github.io. https://gamgi.github.io/js-bcnf/.
Functional Dependencies Checker. (n. d.). Arjo's Blog. https://arjo129.github.io/functionalDependencyCalculator/.
Garcia-Molina, H., Ullman, J. D., & Widom, J. (2009). *Database systems, The complete book*. Person Prentice Hall, Person Education, Inc.
Haidzhary, V. I., & Datsiuk, O. A. (2004). *Osnovy proektuvannia ta vykorystannia baz danykh*. Vydavnytstvo "Politekhnika", TOV "Firma 'Periodyka'".
Ramakrishnan, R., & Gehrke, J. (2003). *Database management systems*. McGraw-Hill Higher Education.
Relational Database Tools. (n. d.). Raymondcho.net. http://raymondcho.net/RelationalDatabaseTools/RelationalDatabaseTools.html.

S. Yaremko, V. Novak

SYSTEM FOR AUTOMATING THE DATABASE NORMALIZATION PROCESS

As data volumes grow rapidly, efficient database management has become critical for organizations. Relational databases play an essential role in ensuring data integrity, enabling complex queries, and supporting various applications, including financial, healthcare, e-commerce, and CRM systems.

Database normalization, a key technique for structuring data and reducing redundancy, improves database efficiency and performance. However, the normalization process can be complex and demands expert knowledge. The article outlines the theoretical foundations of normalization, explaining various normal forms, including 1NF, 2NF, 3NF, and Boyce-Codd Normal Form (BCNF). It emphasizes that while normalization is essential, eliminating redundancy entirely is impractical when maintaining database cohesion.

The proposed system automates normalization using an algorithm based on Heath's theorem, which guarantees a lossless decomposition and dependency preservation. The system can identify minimal sets of functional dependencies, search for quasi-keys, and perform decompositions up to 3NF, ensuring that the database meets lossless join and dependency preservation requirements.

The authors compare the new system with existing tools, highlighting key advantages such as its user-friendly interface and comprehensive functionality, including decomposition capabilities and result integrity verification. The system is designed with Node.js for the backend and React.js for the user interface, providing a web-based platform for database normalization.

The article also explores potential use cases, noting that the system is beneficial for database developers, analysts, and students learning about database management. It simplifies the normalization process, making it faster and more user-friendly. The authors conclude by discussing future improvements, including support for BCNF and 4NF decompositions.

This system offers a practical solution for addressing database normalization challenges, reducing process complexity while enhancing data integrity and performance.

Keywords: relational databases, normalization process, database integrity, functional dependencies, decomposition algorithm, lossless join property, dependency preservation property.

Матеріал надійшов 30.09.2024

Creative Commons Attribution 4.0 International License (CC BY 4.0)