УДК 61:007
DOI: http://dx.doi.org/10.11603/mie.1996-1960.2017.1.7670

ОСОБЛИВОСТІ РОЗРОБЛЕННЯ БАЗ ДАНИХ ДЛЯ МЕДИЧНИХ УСТАНОВ

В. З. Стецюк, А. Й. Савицький, Т. П. Іванова[1], Л. Ю. Бабінцева[2], Ю. О. Луговський, М. М. Лугін

Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
[1]Національна дитяча спеціалізована лікарня «ОХМАТДИТ»
[2]Національна медична академія післядипломної освіти імені П. Л. Шупика

В статті розглянуто проблеми розроблення баз даних для медичних установ в Україні та засоби їх вирішення. Основною умовою такого розроблення є можливість створення єдиного реєстру для медичних установ України для подальшого його впровадження, використання та аналізу результатів застосування. Представлено комплекс засобів, що рекомендовано вживати розробникам програмного забезпечення для покращення ефективності роботи персоналу установи, на прикладі НДСЛ «ОХМАТДИТ». Описуються підходи, що дозволяють розробнику зберігати більше даних та утворювати між ними логічні зв'язки для надання користувачу більшої свободи дій і можливостей при роботі з базою даних.

Ключові слова: ефективність роботи персоналу, комплекс засобів, бази даних, медична установа.

ОСОБЕННОСТИ РАЗРАБОТКИ БАЗ ДАННЫХ ДЛЯ МЕДИЦИНСКИХ УЧЕРЕЖДЕНИЙ

В. З. Стецюк, А. И. Савицкий, Т. П. Иванова[1], Л. Ю. Бабинцева[2], Ю. А. Луговской, М. М. Лугин

Национальный технический университет Украины «Киевский политехнический институт имени Игоря Сикорского»
[1]Национальная детская специализированная больница «ОХМАТДЕТ»
[2]Национальная медицинская академия последипломного образования имени П. Л. Шупика

В статье рассмотрены проблемы разработки баз данных для медицинских учреждений в Украине и средства их решения. Основным условием такой разработки является возможность создания единого реестра для медицинских учреждений Украины для дальнейшего его внедрения, использования и анализа результатов применения. Представлен комплекс средств, которые рекомендуется применять разработчикам программного обеспечения для повышения эффективности работы персонала учреждения, на примере НДСБ «ОХМАТДЕТ». Описываются подходы, позволяющие разработчику хранить больше данных и образовывать между ними логические связи для предоставления пользователю большей свободы действий и возможностей при работе с базой данных.

Ключевые слова: эффективность работы персонала, комплекс средств, базы данных, медицинское учреждение.

ESSENTIAL FEATURES OF MEDICAL DEPARTMENTS DATABASE DEVELOPMENT

V. Z. Stetsyuk, A. Yo. Savytskyi, T. P. Ivanova[1], L. Yu. Babintseva[2], Yu. O. Luhovskyi, M. M. Luhin

National Technical University of Ukraine «Igor Sikorsky Kyiv Polytechnic Institute»
[1]National children specialized hospital «OHMATDYT»
[2]Shupyk National Medical Academy of Postgraduate Education

Nowadays in medical departments, the main problem is to keep big amount of information and create logical connections between them to increase efficiency of medical personal. You need to store different amount of information in different columns and sometimes create new one, but without developer, it is a problem, because you need to edit structure of database.

To prevent this problem developers using software package that helps them to modify database schema. Moreover, the same tool medical personal could use as well.

First of all we need to understand what exactly you working with, what type of information will be worked on, also ensure on simple examples (the way it could be easy) what needs to be done. Next step consist of algorithm structure and base structure, it is the main part of development.

Packages that you need could be an assembly of several languages for example: C#, SQL, Python. Any type of connection that focused on this, called «NoSQL»

Main difference: NoSQL gives you more abilities than simple SQL, by using combination of languages that focused on result and efficiency.

Instead of most NoSQL databases it offers a concept of «eventual consistency» in which database changes are propagated to all nodes «eventually» (typically within milliseconds) so queries for data might not return updated data immediately or might result in reading data that is not accurate, a problem known as stale reads. Additionally, some NoSQL systems may exhibit lost writes and other forms of data loss. Fortunately, some NoSQL systems provide concepts such as write-ahead logging to avoid data loss. For distributed transaction processing across multiple databases, data consistency is an even bigger challenge that is difficult for both NoSQL and relational databases. Even current relational databases do not allow referential integrity constraints to span databases. There are few systems that maintain both ACID transactions and X/Open XA standards for distributed transaction processing.

Key words: staff performance, development package, database, medical facility.

Вступ. Відомо, що функціональна база даних NoSQL — це механізм зберігання та оброблення даних. На відміну від реляційних баз даних NoSQL не використовує для своєї роботи таблиці та відношення між ними. Подібні бази даних існували вже в другій половині 1960-х років, але тоді вони ще не здобули гучне ім'я «NoSQL» (отримане після сплеску популярності на початку XXI століття, що було спричинено потребами у Web 2.0 компаній, таких як «Facebook», «Google» та «Amazon»). NoSQL бази даних все більше і більше використовують в завданнях із застосуванням «big-data». Зауважимо, що такі системи також називають «Not only SQL» (англ. — не тільки SQL) для підкреслення того, що вони можуть підтримувати SQL-подібну структуру та мову запитів [5, 6].

Мотиви такого підходу включають простоту дизайну схеми бази даних, значно спрощують горизонтальне масштабування на кластери машин (що є проблемою для реляційних баз даних), тонкий контроль доступу. Структури даних у NoSQL (до прикладу: ключ-значення, граф, документ) є відмінними від тих, які використовуються за замовчуванням у реляційних базах, що робить деякі операції над даними значно швидшими на NoSQL. Точна відповідність застосування NoSQL-бази даних залежить від проблем, що необхідно вирішити. Іноді структури даних, використовувані в NoSQL-базах, можуть розглядатись як більш гнучкі, ніж таблиці реляційних моделей [1, 2, 4].

Мета дослідження: спрощення роботи медичного персоналу з базами даних.

Матеріал і методи дослідження. Для вирішення завдання розробниками запропоновано використовувати пакети MongoDB, Visual Studio C#. Основою NoSQL є неструктурованість схеми, це означає, що дані можливо зберігати в будь-якому вигляді та динамічно додавати стовпці в бази даних за необхідності; також рядки, що не використовуються, навіть не потрібно видаляти — вони просто не виводяться. Подібні можливості дозволяють скоротити час розробки, надають змогу змінювати бази даних без наслідків. Якщо порівняти з SQL-базою, то при зміні самої бази, тобто при додаванні нових стовпців, необхідно переносити старі дані, що є проблемою при великому обсязі

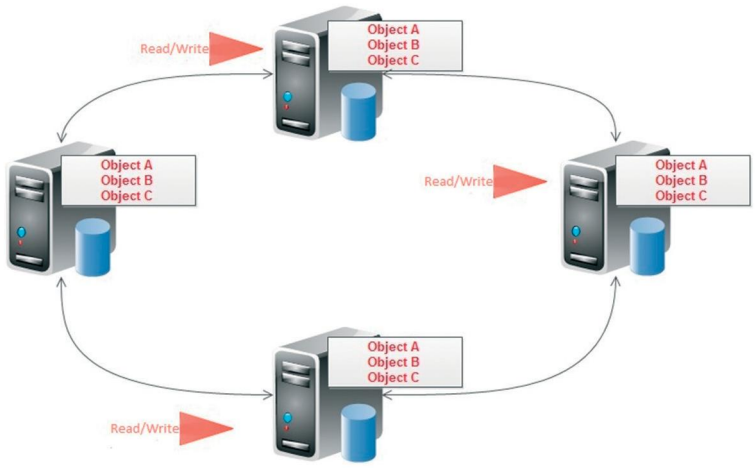

Рис. 1. Master-slave реплікація

даних, оскільки може виникнути необхідність додати до кожного рядка новий запис, враховуючи обмеження цілісності бази. NoSQL хоча і не вирішує першу проблему, але надає змогу простіше обробляти необхідні дані.

Результати та їх обговорення. Програмний комплекс, що використовує описані вище підходи, надає можливість набагато простіше обробляти великий обсяг даних. Яскравим прикладом може служити база, що містить у собі перелік послуг, а, в свою чергу, послуга містить набір матеріалів (фотоплівка для рентгенологічного дослідження, миючий засіб, лікарські засоби тощо), та має можливість враховувати час та оплату праці персоналу. Тобто в такій базі утворюється зв'язок один до багатьох декілька разів, також є можливість додавати специфічні дані. Хоча, слід зауважити, що при наповненні такої бази даних існує велика вірогідність помилки.

Тому для стабільної роботи системи необхідно враховувати, що подібні бази даних не мають обмеження цілісності та при неправильному наповненні або видаленні в базі можуть залишитись непотрібні дані, котрі можуть спричиняти помилки, як в роботі системи, так і в роботі медичного персоналу. Відповідно створення бази даних вимагає від розробника розуміння того де може статися збій системи та як запобігти появі помилок. Також розробнику слід використовувати особливі підходи, що нададуть системі максимальної надійності, швидкості та довговічності, особливо при роботі з даними великих обсягів, зі складною структурою та ієрархією. Отже, основні плюси запропонованого підходу:

- застосування різних типів сховищ;
- можливість розробки схеми бази даних;
- лінійна масштабованість (додавання процесорів збільшує продуктивність роботи);
- скорочення часу розробки;
- швидкість: навіть при великій кількості даних користувач не буде відчувати різницю.

Також необхідно дотримуватись існуючих режимів роботи.

Реплікація — копіювання даних на інші вузли при оновленні. Дозволяє як домогтися більшої масштабованості, так і підвищити доступність та збереження даних. Прийнято поділяти на два види: master-slave (рис. 1) і peer-to-peer (рис. 2).

Перший тип передбачає гарну масштабованість на читання (може відбуватися з будь-якого вузла), але немасштабований запис потрапляє тільки в майстер-вузол. Також є тонкощі із забезпеченням постійної доступності (в разі падіння майстра або вручну, або автоматично на його місце призначається один із решти вузлів). Для другого типу реплікації передбачається, що всі вузли рівні та можуть обслуговувати як запити на читання, так і на запис.

Логічне запитання, яке з'являється в такій ситуації: а що робити системам, які класично висувають високі вимоги до надійності обробки операцій (транзакцій) і в той же час мають потребу в високій швидкості обробки. На сьогодні, зазвичай, застосовують велику кількість підходів, що надають замовнику використовувати програмний продукт ще на етапі розробки. Так і в нашій ситуації необхідно врахувати, що в кожному відділені медичної установи існує своя специфіка. Тобто одні роблять

Рис. 2. Peer-to-peer реплікація

акцент на великий обсяг даних, а інші — на велику кількість зв'язків, в яких зберігається не тільки текстова інформація, а й зображення, тому важливо дороблятиси систему та оптимізувати її вже під конкретне завдання, оскільки створити систему та зробити її максимально простою та універсальною дуже складно. Розглядаючи готові рішення, слід зважувати на ціну, адже, навіть готові програмні продукти потрібно налаштовувати на виконання певних завдань, а в подальшому виділяти кошти та час на підтримку їх функціонування.

Повертаючись до роботи з NoSQL зазначимо, що ці системи можливо оновлювати набагато швидше та дешевше через меншу кількість вимог до розробника та кінцевого продукту. Прикладом впровадження можуть слугувати результати отримані в НДСЛ «ОХМАТДИТ». Зауважимо на низку переваг:

- надійність роботи (навіть при оновленні одного з компонентів інші знаходяться в робочому стані);
- простота застосування (персонал лікарні витрачає менше часу на те, щоб навчитися працювати з системою; система забезпечує зручний вигляд даних, що дозволяє застосовувати її у будь-якому підрозділі лікарні від простої реєстрації пацієнта до складних процедур оброблення даних, ведення статистики чи систем, що можуть надавати рекомендації як лікарю, так і пацієнту);
- автоматизованість (дозволяє технічній підтримці простіше та швидше обслуговувати клієнтів, вирішувати питання при збої системи, або навіть робити все це в автоматичному режимі, надаючи адміністратору тільки звіт про свої дії, де і коли стався збій, які заходи вжила сама система, щоб не втратити працездатність, і навіть рекомендації щодо заходів, які потрібно вжити для подолання проблеми);
- швидкість доступу та гнучкість (підтримується принцип лінійної масштабованості: чим більше комп'ютерів на відділення, тим швидше система працює — це дозволяє за необхідності перерозподіляти ресурси між ними, тобто якщо на одне з відділень впаде дуже велике навантаження, а система, наприклад, буде включати в себе функцію розподілу пацієнтів на групи за певним критерієм — діагнозом, віком, тощо — то в ненавантаженому відділенні просто необхідно буде переналаштувати систему під потреби навантаженого відділення, зрозуміло за умови, що вони знаходяться в одній мережі);
- необмеженість (є можливість додавати нові таблиці без порушень цілісності, також це корисно якщо один запис повинен включати в себе відношення до декількох інших, наприклад: один пацієнт — декілька лікарів). В реляційних базах даних цей зв'язок називається «один до багатьох» і потребує створення проміжної таблиці (рис. 3).

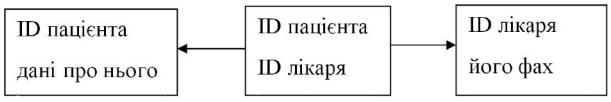

Рис. 3. Приклад зв'язку через проміжну таблицю

NoSQL дозволяє напряму надати відношення від пацієнта до лікаря через відсутність обмеження цілісності.

З мінусів системи можливо відмітити тільки те, що завдяки зберіганню баз в унікальній формі їх оброблення та форматування можуть зайняти більше часу, а також викликати необхідність створити правила чи рекомендації щодо наповнення. Від-

сутність останніх може призвести до стихійності зберігання даних та звести нанівець швидкість доступу, оскільки працювати з ними буде дуже складно.

Висновки-рекомендації. Впровадження запропонованого підходу має багато позитивних аспектів для НДСЛ «ОХМАТДИТ». Проте потребує зваженості та серйозного опрацювання через унікальність побудови системи на основі графів і мінімум обмежень. Слід ще до початку розроблення чітко встановити вимоги як до вигляду самої бази даних, так і до особливостей її роботи, щоб уникнути ситуації, коли кінцевий продукт не задовольняє технічним (не вистачає ресурсів для роботи) та функціональним (система не виконує всіх необхідних функцій) вимогам.

Висновки.

1. При розробці певного автоматизованого робочого місця або простої бази даних необхідно точно розуміти, який продукт необхідно отримати.
2. Сьогодні застосування NoSQL є інноваційним. Проте використання такого підходу є доцільним, якщо йдеться про задум на майбутнє та підтримку роботи з уже готовими рішеннями.

Література.

1. Сравнение NoSQL систем управления базами данных / Devacademy. – Режим доступу: http://devacademy.ru/posts/nosql/.
2. Садаладж П. Дж. NoSQL: новая методология разработки нереляционных баз данных / Садаладж П. Дж., Фаулер М. – М. : Вильямс, 2013. – 192 с.
3. Brewer E. A. A certain freedom: thoughts on the CAP theorem / Brewer E. A. // Proceeding of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing (Zurich, July 25–28, 2010). – N. Y., 2010. – P. 335–336.
4. NoSQL. – Режим доступу: http://nosql-database.org/.
5. Tiwari S. Professional NoSQL / Tiwari S. – Birmingham, 2011. – P. 4–11.
6. Vaish G. Getting Started with NoSQL / Vaish G. – Birmingham, 2013. – 142 p.

References.

1. Sravnenie NoSQL sistem upravleniya bazami dannykh [Comparison of NoSQL database management systems]. (n. d.). Retrieved from Devacademy website, http://devacademy.ru/posts/nosql/
2. Fowler, M., & Sadalage, P. J. (2014). NoSQL: novaya metodologiya razrabotki nerelyatsionnykh baz dannykh [NoSQL Distilled]. Moscow: Williams.
3. Brewer, E. A. (2010). A certain freedom: thoughts on the CAP theorem. In Proceeding of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing (Zurich, July 25–28, 2010) (pp. 335–336). New York: ACM. doi:10.1145/1835698.1835701.
4. NoSQL. (2017, March 5). In Wikipedia: The free encyclopedia. Retrieved March 7, 2017, from https://ru.wikipedia.org/wiki/NoSQL
5. Tiwari, S. (2011). NoSQL: what it is and why you need it. In S. Tiwari. Professional NoSQL. Birmingham: Packt.
6. Vaish, G. (2013). What NoSQL is and what it is not. In G. Vaish Getting Started with NoSQL. Birmingham: Packt.