УДК 004.658:339.138 **doi: 10.32620/reks.2020.1.08**

Д. ТЕРЕНИК[1], Г. А. КУЧУК[2]

[1] ***Національний аерокосмічний університет імені М. Є. Жуковського «ХАІ», Україна***
[2]***Національний технічний університет «ХПІ», Україна***

ПОРІВНЯННЯ SQL І NOSQL БАЗ ДАНИХ НА ПРИКЛАДІ ПРОЕКТУВАННЯ АФФІЛЕЙТ РЕПОРТ СИСТЕМ

В даний час у зв'язку з бурхливим розвитком соціальних мереж і блогер культури в бізнесі намітилася тенденція використання аффілейт систем для просування свого продукту. Аффілейт репорт сервіс – це послуга, пропонована замовникам, які хочуть аналізувати дані про роботу партнерів в аффілейт системах. Дані системи використовуються керівниками і власниками бізнесу для аналізу даних електронної торгівлі і перетворення їх в дані про прибуток/витрати з метою коригування подальшого шляху розвитку свого бізнесу. Цей тип служби включає в себе зберігання даних для всіх пов'язаних підприємств, управління архівом даних, конверсії рекламних кампаній, відстеження тенденцій розвитку і багато іншого. Дані системи ґрунтуються на роботі з великими масивами даних, які необхідно коректно та безпечно зберігати і обробляти, використовуючи системи управління базою даних. Існує два основних напрямки: SQL і NoSQL, тобто реляційні та нереляційні бази даних. Відмінності між ними полягають в тому, як вони спроектовані, які типи даних підтримують, як зберігають інформацію, як підтримують захист інформації. Жорстка схеми реляційних баз даних дозволяє підтримувати безпеку і цілісність даних при їх зберіганні та зміні. Відсутність жорсткої схеми бази даних і в зв'язку з цим потреби при щонайменшій зміні концепції зберігання даних змінювати всю структуру таблиці, значно полегшують роботу з нереляційними базами даних і подальшим їх масштабуванням, однак має і свої недоліки. Важливо розуміти, що задачі бувають різні і методи їх вирішення також різні; вибір бази даних і системи управління базою даних є складною багатопараметричною задачею і є одним з найважливіших етапів при розробці подібних застосунків. Правильно обрана база даних дозволить зменшити грошові та часові витрати, пов'язані з розробкою програмного комплексу, а також полегшити підтримку системи у подальшому. Метою статті є проведення порівняння реляційних та нереляційних баз даних за різними показниками, що використовуються при проектуванні Аффілейт репорт систем. Зокрема, проведений аналіз ефективності за показником швидкодії різних операцій, на основі якого зроблені висновки щодо застосування тієї чи іншої бази даних.

***Ключові слова*:** *база даних; система управління базою даних; SQL; NoSQL; MongoDB; PostgreSQL; аффілейт маркетинг; партнерська програма; репорт система.*

Вступ

Аффілейт (affiliate) маркетинг або «Партнерський маркетинг» можна визначити як метод просування бізнесу в мережі веб-майстрами партнерами, в якому партнер отримує винагороду за кожного відвідувача, передплатника, покупця і/або продаж, здійснені завдяки його зусиллям [1].

Класифікувати партнерські програми можна залежно від того, за що вони платять гроші. Існує кілька схем, за якими можуть платити учасникам [2]:

– фіксована плата (англ. Flat Fee Advertising, FFA);
– оплата за продаж (англ. Cost Per Sale, CPS);
– плата за клік (англ. Pay Per Click, PPC);
– вартість за відвідувача (англ. Cost per Visit, CPV);
– оплата за перегляд (англ. Cost Per Impression, Pay Per Impression, PPI);
– оплата за дію (англ. Cost Per Action, Pay Per Action, PPA).

Для успішної реалізації партнерської програми організатор мусить точно відстежувати кількість покупок або інших дій користувачів, мати надійну систему нараховування та виплати комісії, не забувати про контроль та відстеження зловживань, а також постійно просувати свою партнерську програми у мережі Інтернет [3].

Для організації партнерської програми компанії використовують спеціальне програмне забезпечення, що забезпечує [4]:

– реєстрацію партнерів і підтримку призначеного для користувача інтерфейсу;
– підрахунок відвідувачів, що прийшли по посиланнях з сайтів партнерів, реєстрацію їхніх покупок та інших дій на сайті організатора;
– підрахунок кількості показів банерів і текстових посилань організатора на сайтах партнерів;
– ведення рахунків партнерів і розрахунок винагороди;

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

– автоматичне відстежування зловживань з боку партнерів;

– збір і аналіз статистики як в автоматичному, так і в ручному режимі.

Метою статті є проведення порівняння реляційних та нереляційних баз даних за різними показниками, що використовуються при проектуванні Аффілейт репорт систем.

1. Класифікація баз даних

Розглянемо два основних напрямки: реляційні та нереляційні бази даних.

1.1. Нереляційні бази даних

У NoSQL базах даних (БД) не використовується ANSI SQL DML (англ. Data Manipulation Language – мова керування даними).

У NoSQL базах, на відміну від реляційних, структура даних не регламентована (або слабо типізована, якщо проводити аналогії з мовами програмування) – в окремому рядку або документі можна додати довільне поле без попереднього декларативного зміни структури всієї таблиці. Таким чином, якщо з'являється необхідність поміняти модель даних, то єдина достатня дія – відобразити зміну в коді програми. Однак, у неструктурованою схеми є свої недоліки: з'являються накладні витрати в коді програми при зміні моделі даних (додаткові перевірки наявності поля); виникають додаткові труднощі в розумінні і контролі структури даних при паралельній роботі з базою різних проектів [5].

Найбільш поширеним типом NoSQL баз даних є документні бази. В основі документних баз даних лежать документні сховища, які мають структуру дерева (іноді лісу).

У коді програми дані часто представлені як об'єкт або документ у форматі, подібному JSON, оскільки для розробників це є ефективною і інтуїтивною моделлю даних. Документні бази даних дозволяють розробникам зберігати і запитувати дані з бази даних за допомогою тієї ж документної моделі, яку вони використовують в коді програми [6].

Структура дерева починається з кореневого вузла і може містити кілька внутрішніх і листових вузлів. Листові вузли містять дані, які при додаванні документа заносяться в індекси, що дозволяє навіть при досить складній структурі знаходити місце (шлях) потрібних даних.

API (англ. application programming interface) для пошуку дозволяє знаходити за запитом документи і частини документів. На відміну від сховищ типу ключ-значення, вибірка за запитом до документного сховища може містити частини великої кількості документів без повного завантаження цих документів в оперативну пам'ять [7].

Документи можуть бути організовані (згруповані) в колекції. Їх можна вважати віддаленим аналогом таблиць реляційних баз даних, але колекції можуть містити інші колекції. Хоча документи в колекції можуть бути довільними, для більш ефективного індексування краще об'єднувати в колекцію документи зі схожою структурою.

Гнучкий, напівструктурований, ієрархічний характер документів і документних баз даних дозволяє їм розвиватися відповідно до потреб застосунків. Документна модель добре працює в каталогах, призначених для профіля користувача і системах управління контентом, де кожен документ є унікальним і змінюється з часом.

Приклади систем управління базами даних (СУБД) цього типу – CouchDB, Couchbase, MarkLogic, MongoDB, eXist, Berkeley DB.

У цій роботі розглядається MongoDB.

1.2. Реляційні бази даних

Реляційна база даних – це сукупність взаємопов'язаних таблиць, кожна з яких містить інформацію про об'єкти певного типу.

Рядок таблиці містить дані про один об'єкт, а стовпці таблиці описують різні характеристики цих об'єктів – атрибутів. Записи (рядки таблиці) мають однакову структуру – вони складаються з полів, що зберігають атрибути об'єкта. Кожне поле (стовпець) описує тільки одну характеристику об'єкта і має строго певний тип даних. Всі записи мають одні і ті ж поля, тільки в них відображаються різні інформаційні властивості об'єкта [8].

У реляційній базі даних кожна таблиця повинна мати первинний ключ – поле або комбінацію полів, які однозначно ідентифікують кожен рядок таблиці [9]. Якщо ключ складається з кількох полів, він називається складеним. Ключ повинен бути унікальним і однозначно визначати запис. За значенням ключа можна відшукати єдиний запис. Ключі служать також для впорядкування інформації в базі даних.

Реляційні таблиці можуть бути пов'язані один з одним, отже, дані можуть вилучатись одночасно з декількох таблиць. Зв'язок кожної пари таблиць забезпечується при наявності в них однакових стовпців.

Таблиці реляційної бази даних повинні відповідати вимогам нормалізації відносин.

Кінцевою метою нормалізації є зменшення потенційної суперечливості збереженої в базі даних інформації [10].

Приклади СУБД цього типу – MS SQL, MySQL, Oracle, PostgreSQL

У цій роботі розглядається PostgreSQL.

2. Архітектура аффілейт бази даних

Аффілейт база даних на PostgreSQL оперує такими сутностями :

- партнер (User);
- партнерський купон (Coupon);
- замовлення (Order).

Кожний користувач має декілька партнерських купонів, які можуть використовуватися у багатьох різних замовленнях (зв'язок один до багатьох). UML діаграму спроектованої Аффілейт бази даних наведено на рис. 1.

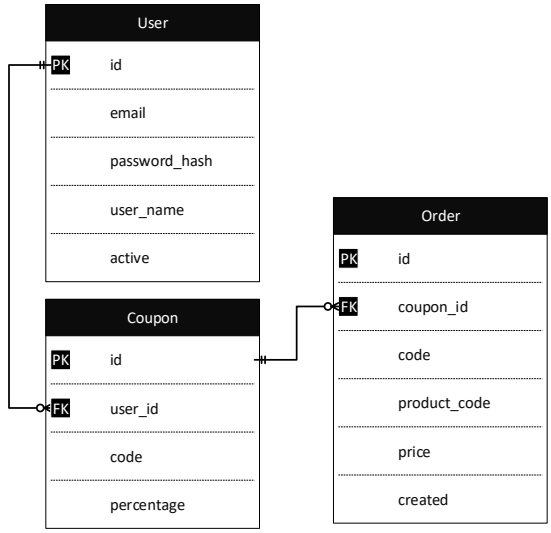

Рис. 1. UML діаграма PostgreSQL аффілейт бази даних

У MongoDB аффілейт базі даних вся інформація зберігається у одній колекції (Users), кожний документ якої зберігає всю інформацію про конкретного партнера (ідентифікаційні дані партнера, інформацію про опрацьовані замовлення з використаними купонами).

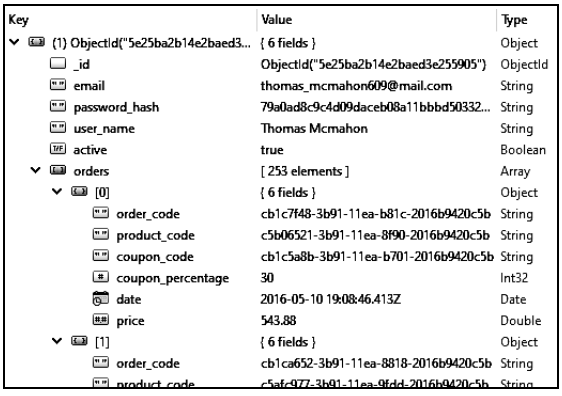

Рис. 2. Структура аффілейт бази даних у MongoDB

3. Розробка CRUD запитів

Для дослідження ефективності роботи баз даних було розроблено скрипти додавання, запиту, зміни та видалення даних.

Кожен скрипт замірює тривалість виконання операції с базою даних. Час виконання операцій має бути якомога меншим.

3.1. Операція додавання даних

Особливістю додавання даних в реляційні бази даних (PostgreSQL) є необхідність запам'ятовування ідентифікаторів об'єктів (PRIMARY KEY), для використання їх згодом в якості зовнішніх ключів (FOREIGN KEY) [11].

Особливістю додавання даних в документні бази даних (MongoDB) є дублювання даних (в нашому випадку – інформація про купон коди) [12].

3.2. Операції запиту даних

У MongoDB, в нашому випадку, так як колекція (Users) містить документи з масивом вкладених документів (Orders), при агрегації даних використовується команда $unwind, що дозволяє розгорнути поля-підмасиви шляхом дублювання батьківської сутності для кожного поля такого підмасива.

Якщо мова йде про масиви об'єктів, то для кожного розгорнутого рядка під ім'ям підмасива буде знаходиться один вкладений об'єкт [13, 14].

Для дослідження ефективності роботи баз даних будемо запитувати такі дані:

– список замовлень за купон кодом (orders_by_coupon); вибираємо всі замовлення з заданим купон кодом;

– топ продажів (top_bestsellers). Підсумовуємо ціни всіх проданих продуктів користувача. Сортуємо від більшого до меншого; вибираємо певну кількість (top_limit) перших записів;

– список купонів використовуваних користувачем (user_coupons); вибираємо всі купони, які використовує користувач із заданою електронною поштою (user_email).

3.3. Операції зміни даних

Для дослідження ефективності роботи баз даних будемо змінювати такі дані:

– нова ціна замовлення (set_order_price); встановити нову ціну продажу замовлення за кодом замовлення;

– нова процентна ставка купона (set_coupon_percentage); встановити нову процентну ставку купона за кодом купона.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

3.4. Операції видалення даних

Особливістю додавання даних в реляційні бази даних (PostgreSQL) є їх каскадний характер – при видаленні купона видаляться усі замовлення, пов'язані з цим купоном. Для даних, що зберігаються у нереляційних базах даних, видалення відбувається значно складніше.

Для дослідження ефективності роботи баз даних будемо видаляти такі дані:

– видалити замовлення за купон кодом (delete_orders_by_coupon); видалити всі замовлення що використовують певний купон код;

– видалити користувача із заданою електронною поштою (delete_user_by_email).

4. Результати експериментів

Запустимо багаторазово скрипти додавання, зміни, видалення та запиту даних для MongoDB та PostgreSQL баз даних, та порівняємо швидкість їх виконання. У табл. 1 наведені усереднені показники часу виконання скриптів.

Таблиця 1

Результати запуску скриптів (середні значення)

Скрипт	MongoDB	PostgreSQL
Додавання даних	0.34095	12.00763
Запросити топ продажів	0.06652	0.05794
Запросити список купонів використовуваних користувачем	0.01362	0.00478
Запросити список замовлень за купон кодом	0.19709	0.01172
Встановити нову процентну ставку купона	0.02600	0.00120
Встановити нову ціну замовлення	0.01256	0.00099
Встановити флаг активності користувача	0.00099	0.00096
Видалити замовлення за купон кодом	0.03255	0.00181
Видалити користувача за електронною поштою	0.00160	0.07179
Видалити купон та замовлення за кодом купона	0.02877	0.01465

Згруповані за типом операцій (додавання, зміни, видалення та запиту даних) результати порівняння швидкодії виконання скриптів для аналізуємих PostgreSQL та MongoDB баз даних наведено на рис. 3 – 6.

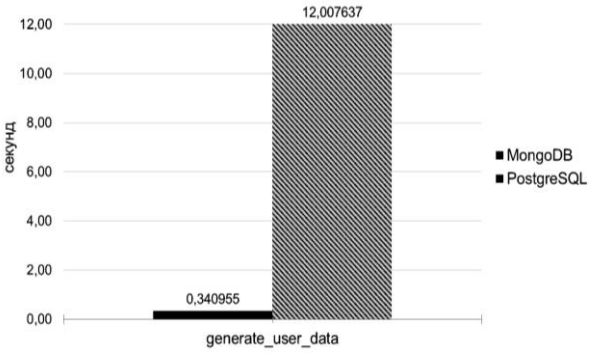

Рис. 3. Порівняння швидкодії операцій додавання даних для PostgreSQL та MongoDB

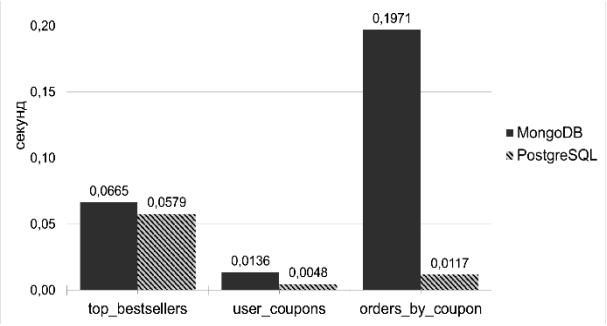

Рис. 4. Порівняння швидкодії операцій запиту даних для PostgreSQL та MongoDB

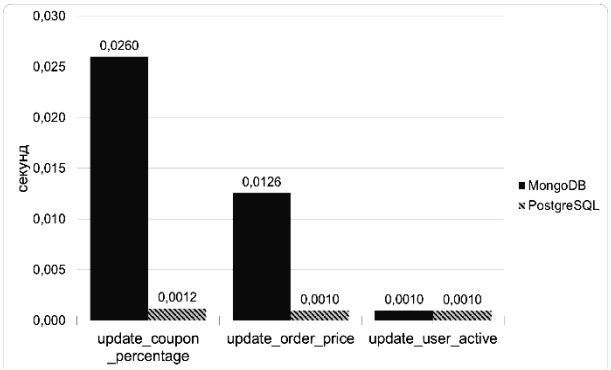

Рис. 5. Порівняння швидкодії операцій зміни даних для PostgreSQL та MongoDB

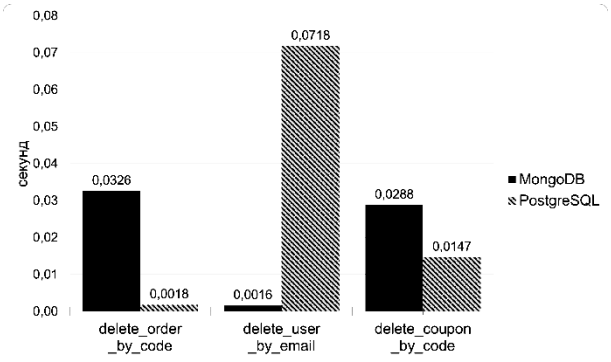

Рис. 6. Порівняння швидкодії операцій видалення даних для PostgreSQL та MongoDB

5 Аналіз отриманих результатів

Для коректного аналізу отриманих результатів необхідно враховувати важливість виконання кожного типу CRUD операцій для досліджуваної системи. На основі важливості конкретного типу операції можна визначити частоту виклику цієї операції. Для проведення подальших досліджень пропонуються наступні значення частот використання типів операцій (табл. 2). Дані значення залежать від аналізованого програмного комплексу, в нашому випадку це аффілейт репорт системи.

Таблиця 2

Підрахунок балів

Скрипт	Частота	MongoDB (бали)	PostgreSQL (бали)
Додавання даних	0,205	2,05	0
Запросити топ продажів	0,135	0	1,35
Запросити список купонів використовуваних користувачем	0,135	0	1,35
Запросити список замовлень за купон кодом	0,135	0	1,35
Встановити нову процентну ставку купона	0,08	0	0,8
Встановити Нову ціну замовлення	0,08	0	0,8
Встановити флаг активності користувача	0,08	0	0,8
Видалити замовлення за купон кодом	0,05	0	0,5
Видалити користувача за електронною поштою	0,05	0,5	0
Видалити купон та замовлення за кодом купона	0,05	0	0,5
Сума	1	2,55	7,45

Маємо 10 скриптів (кожен виконує певну CRUD операцію), з частотою виклику кожного. При порівнянні часу виконання кожного скрипта будемо присвоювати бал тій базі даних, у якій цей час виявиться меншим (максимальна швидкодія). Максимальна кількість можливо отриманих балів – 10 (оскільки розглядається 10 скриптів). Для кожної операції отримані бали будемо множити на частоту виклику цієї операції, таким чином буде враховуватися важливість виконання операції для роботи досліджуваного програмного продукту в цілому. Підсумуємо бали, отримані кожною базою після проведення дослідів, і визначимо, вибір якої бази даних є найбільш виправданим у використанні для побудови аффілейт репорт систем.

Отримані результати представлені в табл. 2.

Висновки

Проаналізувавши дані з табл. 2 можна зробити висновок, що для розглянутої структури даних аффілейт репорт системи PostgreSQL база даних є більш ефективною за показником швидкодії операцій (7,45 бали проти 2,55, рис. 7).

Рис. 7. Порівняння кількості набраних балів для PostgreSQL та MongoDB

Додавання даних є значно повільнішим для PostgreSQL, бо необхідно забезпечувати реляційні зв'язки.

Запит та зміна даних повільніші для MongoDB, бо необхідно розгортати поля-підмасиви ($unwind).

Видалення користувача виконується швидше для MongoDB, бо вся інформація зберігаються в одному документі і немає необхідності видаляти пов’язані об’єкти (як для реляційних даних). Але видалення вкладених даних у MongoDB відбувається повільніше ніж у PostgreSQL (завдяки функціональності каскадних змін у реляційних базах).

Внутрішній устрій різних баз даних впливає на особливості роботи з ними.

В якійсь ситуації документні бази даних покращують продуктивність вашого застосунка, наприклад, якщо треба зберігати неструктуровані дані.

В іншій же ситуації краще буде використовувати традиційні реляційні бази даних. Крім того, можна використовувати змішаний підхід: зберігати різні типи даних в різних базах даних.

У майбутньому також необхідно проаналізувати:

– розмір витраченої пам'яті для збереження необхідних даних;

– складність супроводу (внесення змін у готову систему).

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

Література

1. Brown, Bruce C. The Complete Guide to Affiliate Marketing on the Web: How to Use and Profit from Affiliate Marketing Program [Text] / Bruce C. Brown. – Atlantic Publishing Company, 2008). – 384 p.

2. Prussakov, E. Affiliate Program Management: An Hour a Day [Text] / E. Prussakov. – Sybex, 2011. – 460 p.

3. Ulaner, Kevin. Affiliate Marketing: The Beginner's Step by Step Guide to Making Money Online with Affiliate Marketing [Text] / K. Ulaner. – CreateSpace, 2017. – 62 p.

4. Singh, Surabhi. Driving Traffic and Customer Activity Through Affiliate Marketing [Text] / Surabhi Singh. – IGI Global, 2017. – 233 p.

5. McCreary, Dan. Making Sense of NoSQL: A guide for managers and the rest of us [Text] / Dan McCreary, Ann Kelly. – Manning Publications, 2013. – 312 p.

6. Tiwari, Shashank. Professional NoSQL [Text] / Shashank Tiwari. – Packt Publishing, 2011. – 384 p.

7. , Fowler, Adam. NoSQL For Dummies / Adam Fowler. – Dummies Tech; 2015. – 456 p.

8. Hernandez, Michael J. Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design [Text] / Michael J. Hernandez. – Addison-Wesley Professional; 2003. – 611 p.

9. PostgreSQL Documentation Quick Links [Electronic resource]. – Access mode: https://www. postgresql.org/docs. – 22.10.2019.

10. Hannah, Crossing. PostgreSQL Administration 9. Recipe Book [Text] / H. Crossing, S. Riggs. – Litres, 2019. – 351 p.

11. Sweet, Justin. Legal Aspects of Architecture, Engineering & the Construction Process [Text] / Justin Sweet, Marc M. Schneier. – Cengage Learning, 2008. – 769 p.

12. Mancas, C. Conceptual Data Modeling and Database Design: A Fully Algorithmic Approach, Volume 1: The Shortest Advisable Path [Text] / C. Mancas. – CRC Press, 2016. – 698 p.

13. Aggregation Pipeline Stages. $unwind (aggregation) [Electronic resource]. – Access mode: https://docs.mongodb.com/manual/reference/operator/ aggregation/unwind. – 22.10.2019.

14. Taulli, T. How to Create the Next Facebook: Seeing Your Startup Through, from Idea to IPO [Text] / T. Taulli. – Apress, 2012. – 208 p.

References

1. Bruce, C. Brown. *The Complete Guide to Affiliate Marketing on the Web: How to Use and Profit from Affiliate Marketing Program*. Atlantic Publishing Company, 2008. 384 p.

2. Prussakov, Evgenii. *Affiliate Program Management: An Hour a Day*. Sybex, 2011. 460 p.

3. Ulaner, Kevin. *Affiliate Marketing: The Beginner's Step by Step Guide to Making Money Online with Affiliate Marketing*. CreateSpace, 2017. 62 p.

4. Singh, Surabhi. *Driving Traffic and Customer Activity Through Affiliate Marketing*. IGI Global, 2017. 233 p.

5. McCreary, Dan., Kelly, Ann. *Making Sense of NoSQL: A guide for managers and the rest of us*. Manning Publications, 2013. 312 p.

6. Tiwari, Shashank. *Professional NoSQL*. Packt Publishing, 2011. 384 p.

7. Fowler, Adam. *NoSQL For Dummies*. Dummies Tech, 2015, 456 p.

8. Hernandez, Michael J. *Database Design for Mere Mortals: A Hands-On Guide to Relational Database Design*. Addison-Wesley Professional, 2003, 611 p.

9. *PostgreSQL Documentation, Quick Links*. Available at: https://www.postgresql.org/docs/ (accessed 22.10.2019)

10. Hannah, Crossing, Simon, Riggs. *Crossing. PostgreSQL Administration 9. Recipe Book*, Litres, 2019. 351 p.

11. Sweet, Justin., Schneier, Marc M. *Legal Aspects of Architecture, Engineering & the Construction Process*, Cengage Learning, 2008, 769 p.

12. Mancas, C. *Conceptual Data Modeling and Database Design: A Fully Algorithmic Approach, Volume 1: The Shortest Advisable Path*. CRC Press, 2016. 698 p.

13. *Aggregation Pipeline Stages. $unwind (aggregation)*. Available at: https://docs.mongodb.com/ manual/reference/operator/aggregation/unwind (accessed 22.10.2019).

14. Taulli, Tom. *How to Create the Next Facebook: Seeing Your Startup Through, from Idea to IPO*. Apress, 2012. 208 p.

Надійшла до редакції 5.01.2020, розглянута на редколегії 20.01.2020

СРАВНЕНИЕ SQL И NOSQL БАЗ ДАННЫХ НА ПРИМЕРЕ ПРОЕКТИРОВАНИЕ АФФИЛЕЙТ РЕПОРТ СИСТЕМ

Д. Тереник, Г. А. Кучук

В настоящее время в связи с бурным развитием социальных сетей и блогер культуры в бизнесе наметилась тенденция использования аффилейт систем для продвижения своего продукта. Аффилейт репорт сервис – это услуга, предлагаемая заказчикам, которые хотят анализировать данные о работе партнеров в аффилейт системах. Данные системы используются руководителями и владельцами бизнеса для анализа данных электронной торговли и преобразования их в данные о прибыли/расходах для корректировки дальней-

шего пути развития своего бизнеса. Этот тип службы включает в себя хранение данных для всех связанных предприятий, управление архивом данных, конверсию рекламных кампаний, отслеживание тенденций развития и многое другое. Данные системы основаны на работе с большими массивами данных, которые необходимо корректно и безопасно хранить и обрабатывать, используя системы управления базой данных. Существует два основных направления: SQL и NoSQL, реляционные и нереляционные базы данных. Различия между ними заключаются в том, как они спроектированы, какие типы данных поддерживают, как хранят информацию, как поддерживают защиту информации. Жесткая схемы реляционных баз данных позволяет поддерживать безопасность и целостность данных при их хранении и изменении. Отсутствие жесткой схемы базы данных и в связи с этим потребности при минимальном изменении концепции хранения данных менять всю структуру таблицы, значительно облегчают работу с нереляционными базами данных и последующую их поддержку, но имеет и свои недостатки. Важно понимать, что задачи бывают разные и методы их решения также различны; выбор базы данных и системы управления базой данных представляет собой сложную многопараметрическую задачу и является одним из важнейших этапов при разработке подобных приложений. Правильно выбранная база данных позволит уменьшить денежные и временные затраты, связанные с разработкой программного комплекса, а также облегчить поддержку системы в дальнейшем. Целью статьи является проведение сравнения реляционных и нереляционных баз данных по различным показателям, используемых при проектировании Аффилейт репорт систем. В частности, проведен анализ эффективности по показателю быстродействия различных операций, на основе которого сделаны выводы по применению той или иной базы данных.

Ключевые слова: база данных; система управления базой данных; SQL; NoSQL; MongoDB; PostgreSQL; аффилейт маркетинг; партнерская программа; репорт система.

SQL & NOSQL DATABASE COMPARISON BY CASE DESIGNING AFFILIATE SYSTEM REPORT

D. Terenyk, H. Kuchuk

Nowadays, due to the rapid development of social networks and the blogger culture, there is a tendency to use affiliate systems to promote their product. The Affiliate Reporting Service is a service offered to customers who want to analyze the affiliate systems' performance data. These systems are used by business executives and business owners to analyze ecommerce data and convert it into profit/expense data to adjust their business path further. This type of service includes data storage for all affiliates, data archive management, conversion of advertising campaigns, trend tracking, and more. These systems are based on large data sets that need to be stored correctly and safely stored and processed using database management systems. There are two major direction: SQL and NoSQL, relational and non-relational databases. The differences between them are how they are designed, what types of data they support, how they store information, how they support information security. A rigid relational database schema helps maintain the security and integrity of data when stored and modified. The lack of a rigid database schema and the need to change the entire structure of the table with a minimal change in the storage concept, make it easier to work with non-relational databases and subsequently support them, but it also has its disadvantages. It is important to understand that the tasks are different and the methods for solving them are also different; Choosing a database and database management system is a complex multi-parameter task and is one of the most important steps in developing such applications. Properly selected database will reduce the monetary and time costs associated with the development of the software, as well as facilitate system support in the future. The purpose of the article is to compare relational and non-relational databases by different metrics used in Affiliate Reporting Systems Design. In particular, a performance analysis was conducted on the performance of various operations, on the basis of which conclusions were drawn about the use of a particular database.

Keywords: database; database management system; SQL; NoSQL; MongoDB; PostgreSQL; affiliate marketing; reporting system.

Тереник Дмитро – магістрант кафедри комп'ютерних систем, мереж та кібербезпеки, Національний аерокосмічний університет імені М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Кучук Георгій Анатолійович – д-р техн. наук, професор, професор кафедри обчислювальної техніки та програмування, Національний технічний університет «Харківський політехнічний інститут», Харків, Україна.

Terenyk Dmitry – Master Student of Computer Systems, Networks and Cyber security Department, National Aviation University "Kharkiv Aviation Institute", Kharkiv, Ukraine, e-mail: dm.terenik@gmail.com.

Kuchuk Heorhii – Doctor of Technical Sciences, Professor, Professor of Computer Engineering and Programming Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine, e-mail: kuchuk56@ukr.net, ORCID Author ID: 0000-0002-2862-438X; Scopus Author ID: 6507706464, https://scholar.google.com.ua/citations?hl=ru&user=gHejYRUAAAAJ.