DOI: 10.36910/6775-2524-0560-2019-36-12
УДК 004.056
В.М. Мельник, А.І. Нагорнюк, К.Я Бортник
Луцький національний технічний університет

ДОСЛІДЖЕННЯ ШВИДКОДІЇ ВИКОНАННЯ ЗАПИТУ В СУБД MYSQL ТА MARIADB ЗАСОБАМИ PYTHON ТА PHP

Мельник В.М., Нагорнюк А.І., Бортник К.Я. Дослідження швидкодії виконання запиту в СУБД MySQL та MariaDB засобами Python та PHP. В даній роботі проведено дослідження швидкодії виконання запитів для баз даних MySQL та MariaDB з виконанням базових операцій у середовищах Python та PHP. Проведено теоретичний огляд основних характеристик вищезгаданих баз даних, а також здійснено огляд особливостей підключення та взаємодії з ними в мовах підтримки Python та PHP, а саме, організація роботи з об'єктами підключення та курсорами. Встановлено залежність між кількістю виконання запитів та їх швидкодією виконання.
Ключові слова: СУБД, MySQL, MariaDB, Python, PHP, швидкодія виконання, запит.

Мельник В.М., Нагорнюк А.И., Бортник Е.Я. ИССЛЕДОВАНИЕ БЫСТРОДЕЙСТВИЯ ВЫПОЛНЕНИЯ ЗАПРОСОВ В СУБД MYSQL И MARIADB СРЕДСТВАМИ PYTHON И PHP. В данной работе проведено исследование быстродействия выполнения запросов для баз данных MySQL и MariaDB с выполнением базовых операций в средах Python и PHP. Проведен теоретический обзор основных характеристик вышеуказанных баз данных, а также осуществлен обзор особенностей подключения и взаимодействия с ними в языках поддержки Python и PHP, а именно, организация работы с объектами подключения и курсорами. Установлена зависимость между количеством выполнения запросов и их быстродействием выполнения.
Ключевые слова: СУБД, MySQL, MariaDB, Python, PHP, быстродействие выполнения, запрос.

Melnyk V., Nahorniuk A., Bortnik K. RESEARCH OF PERFORMANCE REQUESTS IN MYSQL AND MARIADB DBMS SUPPORTED BY PYTHON AND PHP. In this paper, we investigate the performance of requests for MySQL and MariaDB databases with basic operations made in Python and PHP environments. A theoretical overview of the main characteristics for the databases described above was made, as well as an overview of connection features and interaction with them in Python and PHP support languages, namely, the organization of work with connection objects and cursors. The relationship between the number of requests and their speed of execution is established.
Keywords: DBMS, MySQL, MariaDB, Python, PHP, runtime speed, request.

1. Постановка проблеми. Вибір системи управління баз даних (СУБД) являє собою складне багатопараметричне завдання і є одним із важливих етапів в ході розробки додатків з базами даних. Обраний програмний продукт повинен задовольняти як поточним, так і майбутнім потребам підприємства. При цьому слід враховувати фінансові витрати на придбання необхідного обладнання, самої системи, розробку необхідного програмного забезпечення на її основі, а також навчання персоналу.

MySQL є однією з найбільш поширених СУБД у світі. Це програмне забезпечення безкоштовне і розповсюджується з відкритим вихідним кодом. Вона розроблена на C/C++ і є однією з найбільш популярних варіантів баз даних. СУБД була розроблена шведською компанією «MySQL AB» в 1995 році. У 2008 році MySQL AB була придбана компанією Sun Microsystems, а в 2010 році Oracle придбала Sun Microsystems. З тих пір MySQL підтримується і управляється навіть і Oracle [1, 2].

Під час придбання Sun Microsystems компанією Oracle деякі з старших інженерів, які працювали над розробкою MySQL, відчули, що існує конфлікт інтересів між MySQL і комерційної базою даних Oracle, тобто Oracle Database Server. В результаті, ці інженери створили форк бази коду MySQL і заснували власну організацію. Так народилася MariaDB. На сьогоднішній день обидві бази даних користуються великою популярністю і широко використовуються різними компаніями розробників [3, 4].

2. Аналіз існуючих способів взаємодії з базою даних. Для роботи з базами даних у Python можна використовувати різноманітні бібліотеки, специфічні та спеціалізовані для кожної конкретної БД. Наприклад, для MySQL – MySQLdb, pymysql, mysql, для Postgres – Psycopg2, pg8000, py-postgresql, PyGreSQL, ocpgdb, bpgsql [5]. Для з'єднання з СУБД MariaDB розробнику необхідно використовувати коннектор з MySQL. Оскільки підтримка SQLite є в стандартній бібліотеці Python, то встановлювати додаткові модулі не потрібно.

Усі модулі для взаємодії з базами банних у Python повинні відповідати стандарту PEP 249, у якому описані основні принципи створення API. Завдяки специфікації DB-API існує спільний інтерфейс для її вивчення. Перенесення коду для використання іншого продукту баз даних є набагато простішим і часто вимагає зміни лише декількох рядків. Цей API був визначений для заохочення подібності між модулями Python, які використовуються для доступу до баз даних. Поточна версія DB-API 2.0 (PEP 249) замінена старішою версією DB-API 1.0 (PEP 248) [6, 7].

Для підключення до бази даних перш за все необхідно виконати метод connect(), в який передати параметри підключення, такі як ім'я хоста, порту, назву (логін) користувача та його пароль. Після успішного отримання об'єкту Connection з нього необхідно отримати об'єкт Cursor. Цей об'єкт представляє курсор бази даних, який використовується для управління контекстом операцій. Курсори, створені з одного і того ж з'єднання, не є ізольованими, тобто будь-які зміни, зроблені курсором бази даних, негайно помітні іншими курсорами. Курсори, створені з різних з'єднань, можуть бути або не можуть бути ізольовані. Це залежить від того, як реалізована підтримка транзакцій (див. також методи .rollback() і .commit() з'єднання). Після отримання курсору можна проводити необхідні маніпуляції: виконання запитів, вибірку даних, і інше [6].

Для полегшення взаємодії з даними можна використовувати ORM. Об'єктно-реляційне відображення (ORM) – це метод, який дозволяє запитувати і маніпулювати даними з бази даних з використанням об'єктно-орієнтованої парадигми. Бібліотека ORM – це абсолютно звичайна бібліотека, написана на певній мові, яка інкапсулює код, необхідний для маніпулювання даними, завдяки чому немає необхідності використовувати SQL, а можна взаємодіяти з даними як з об'єктом. Приклади ORM – це Python: Django ORM, SQLAlchemy; PHP: Propel, Doctrine.

У PHP для взаємодії з БД також можуть використовуватись спеціалізовані бібліотеки для кожної конкретної бази даних. Наприклад, для MySQL, і, відповідно, для MariaDB – це бібліотеки mysql, mysqli та PDO. Бібліотека mysql заборонена для використання у PHP 5.5 і новіших його версій. Натомість слід використовувати mysqli та PDO [9].

PDO – це PHP Data Objects, або рівень абстракції доступу до бази даних. PDO пропонує уніфікований інтерфейс для доступу до багатьох різних баз даних, а також резюмує не тільки API бази даних, але й основні операції, які в іншому випадку доводиться повторювати сотні разів у кожному додатку. На відміну від mysql та mysqli, які є низькорівневими API, не призначеними для прямого використання, а лише як будівельний матеріал для вищого рівня абстракції), PDO вже є абстракцією. Справжніми перевагами PDO є: безпека (використання попередньо підготовлених запитів), зручність використання (багато допоміжних функцій для автоматизації рутинних операцій), повторне використання (уніфікований API для доступу до безлічі баз даних, від SQLite до Oracle).

Щоб отримати доступ до серверу баз даних через PDO, необхідно створити об'єкт. Для цього необхідно передати дані про сервер у вигляді DSN, а також ім'я користувача, пароль, і інші дані за необхідності. Після цього є можливість виконувати запити. Робити це можна різними способами – якщо запит не містить у своїй структурі змінних, можна використовувати метод PDO::query(). Якщо ж у запиті використовуються змінні, то необхідно використовувати метод prepare(), а опісля – використовувати метод execute(). Підстановка змінних у запити може здійснюватися за допомогою позиційних або іменованих псевдозмінних. Для встановлення відповідності між псевдозмінною і її значенням під час підстановки в запит можуть використовуватись масиви, асоціативні (іменовані) масиви, або методи bindValue() чи bindParam() [10].

3. Результати проведених досліджень. Під час проведення дослідження використовувались СУБД MySQL версії 8.0.16 та MariaDB версії 10.4.7, а також інструменти підтримки: Python 3.7.4 та PHP 7.2.10. Загальна структура запиту створення таблиці наступна:

```
CREATE TABLE IF NOT EXISTS research (
 id INT(7) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
 vv TINYINT);
```

Після створення таблиці виконувались операції вставки, вибірки, оновлення та видалення (CRUD) для 100, 500, 1 000, 5 000, 10 000, 50 000 та 10 000 записів. Вимірювання відбувалося за допомогою

визначення різниці між часовими мітками з використанням методу time.time() в Python та функції microtime() у PHP.

Нижче подано графік залежності часу виконання запитів для CRUD операцій від кількості записів для СУБД MySQL засобами Python (рис.1).

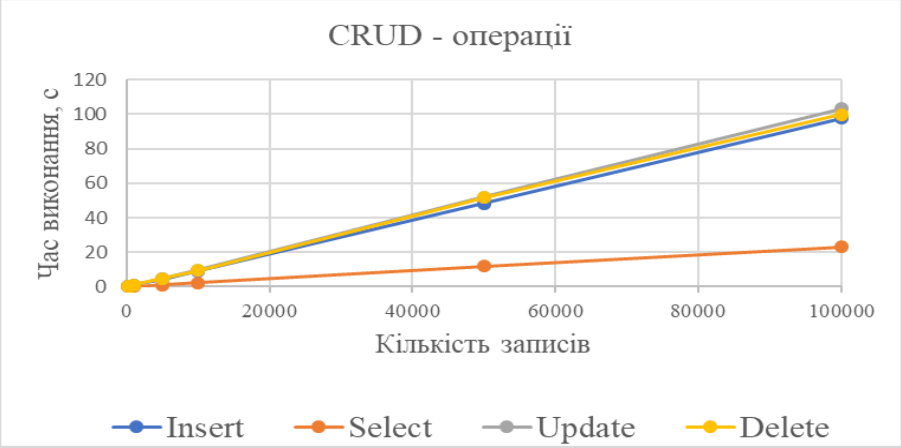

Рис. 1 – Графік залежності часу виконання запитів для CRUD операцій від кількості записів для СУБД MySQL засобами Python

Як видно з графіка, залежність є рівномірно зростаючою прямою лінійною. Зі зростанням кількості записів час їх виконання зростає лінійно. Операції вставки, оновлення та видалення виконуються майже однакову кількість часу. Однак операція вибірки є значно швидшою, майже в п'ять разів.

		Python, MySQL, середній час виконання, с			
		Вставка	Вибірка	Оновлення	Видалення
Кількість записів	100	0.1118	0.027801	0.099199	0.097
	500	0.559	0.1294	0.5786	0.5712
	1 000	1.171199	0.266201	1.2036	1.2444
	5 000	5.9728	1.461001	6.280401	5.970599
	10 000	12.006601	2.428001	12.112601	12.353397

Таблиця 1. Результати проведеного дослідження виконання запитів в базі даних MySQL з підтримки Python

		PHP, MySQL, середній час виконання, с			
		Вставка	Вибірка	Оновлення	Видалення
Кількість записів	100	0.120259	0.023962	0.108073	0.123157
	500	0.443965	0.090917	0.482407	0.483698
	1 000	1.016007	0.186314	1.016624	1.072325
	5 000	5.796535	1.206885	6.191491	6.160956
	10 000	12.169063	2.157948	12.281779	12.234494

Таблиця 2. Результати проведеного дослідження виконання запитів в базі даних MySQL з підтримки PHP

		Python, MariaDB, середній час виконання, с			
		Вставка	Вибірка	Оновлення	Видалення
Кількість записів	100	0.055	0.021	0.048339	0.050925
	500	0.232986	0.100601	0.248224	0.249581
	1 000	0.460396	0.198199	0.501002	0.497153
	5 000	2.629197	1.201602	2.742204	2.744992
	10 000	5.158813	2.2126	5.218996	5.2186

Таблиця 3. Результати проведеного дослідження виконання запитів в базі даних MariaDB з підтримки Python

		PHP, MariaDB, середній час виконання, с			
		Вставка	Вибірка	Оновлення	Видалення
Кількість записів	100	0.057982	0.023017	0.056827	0.05867
	500	0.257298	0.103248	0.263533	0.260386
	1 000	0.550662	0.221751	0.537945	0.546798
	5 000	2.825545	1.048985	2.740266	2.665675
	10 000	5.176147	1.942884	5.04902	4.888232

Таблиця 3. Результати проведеного дослідження виконання запитів в базі даних MariaDB з підтримки PHP

Результати дослідження, наведені у таблицях 1-4, демонструють залежність часу виконання запитів від типу бази даних та мови реалізації підключення. Підключення у Python було здійснене за допомогою бібліотеки pyodbc, а для PHP – стандартними засобами PDO.

Для наочності аналізу даних побудовано графік (рис. 2), який демонструє наведені значення в таблицях

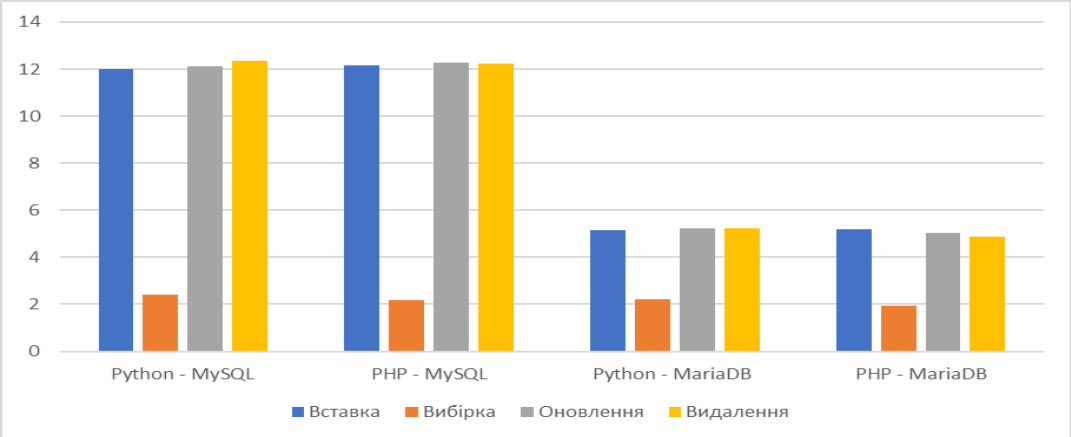

Рис. 2 – Результати досліджень для різних операцій виконання в середовищах Python та PHP

Як видно з графіка, швидкість виконання запитів вибірки фактично не залежить ні від типу бази даних, ні від способу підключення до неї. Також немає суттєвої різниці між способами підключення в межах однієї бази даних. Запити в середовищах Python та PHP виконуються майже однакову кількість часу.

Варто зауважити також подібну швидкодію операцій вставки, оновлення та видалення. Особливо помітне прискорення, майже в шість раз, спостерігається для операції вибірки. Що ж до порівняльної характеристики швидкодії СУБД, у даному випадку з графіка можемо спостерігати значне скорочення затрачуваного часу на виконання запитів бази даних MariaDB у порівнянні з MySQL – приблизно у 2,5 рази.

4. Висновки. У даній роботі було розглянуто основні особливості підключення та взаємодії з поширеними СУБД MySQL та MariaDB і визначено основні принципи побудови з'єднань з базою даних у середовищах Python та PHP. Було проведено дослідження, метою якого було визначити залежність часу виконання CRUD операцій у різних середовищах від кількості запитів до бази даних відповідного типу. Виявлено прямолінійну залежність між кількістю операцій з базою даних та часом їх виконання. Здійснено порівняння результатів швидкодії виконання базових операцій у різних базах даних за наявності різноманітних умов.

Reference

1. Jesper Wisborg Krogh. MySQL Connector/Python Revealed / Jesper Wisborg Krogh // MySQL Connector/Python Revealed: SQL and NoSQL Data Storage Using MySQL for Python Programmers. 1st ed. Edition 2018 Apress 540 c
2. Russell J. T. Dyer. Learning MySQL and MariaDB: Heading in the Right Direction with MySQL and MariaDB / Russell J. T. Dyer // Learning MySQL and MariaDB: Heading in the Right Direction with MySQL and MariaDB 1 edition 2015 O'Reilly Media 408 c
3. Charles Bell. Expert MySQL / Charles Bell // Expert MySQL. 2nd ed. edition 2012 Apress 640 c
4. Federico Razzoli. MariaDB Essentials / Federico Razzoli, Emilien Kenler // MariaDB Essentials: Quickly get up to speed with MariaDB-the leading, drop-in replacement for MySQL, through this practical tutorial. 1 edition 2015 Packt Publishing 208 c
5. Adrian W. West. Practical PHP 7, MySQL 8, and MariaDB Website Databases / Adrian W. West, Steve Prettyman // Practical PHP 7, MySQL 8, and MariaDB Website Databases: A Simplified Approach to Developing Database-Driven Websites . 2nd ed. edition 2018 Apress 568 c
6. MySQL official site. URL:https://www.mysql.com/
7. MySQL – Wikipedia. URL:https://en.wikipedia.org/wiki/MySQL
8. MariaDB official site. URL:https://mariadb.org/
9. MariaDB – Wikipedia. URL:https://en.wikipedia.org/wiki/MariaDB
10. Python PostgreSQL Tutorial Using Psycopg2 URL:https://pynative.com/python-postgresql-tutorial/
11. PEP 249 -- Python Database API Specification v2.0 URL:https://www.python.org/dev/peps/pep-0249/
12. Use cases of the DB API for a PostgreSQL database // Python wiki. URL:https://wiki.python.org/moin/UsingDbApiWithPostgres#python-db-api
13. Object-relational mapping – Wikipedia. URL:https://en.wikipedia.org/wiki/Object-relational_mapping
14. PHP official documentation. URL:https://www.php.net/manual/en/mysql.php
15. (The only proper) PDO tutorial. URL:https://phpdelusions.net/pdo