JURNAL INFORMATIKA
Volume 5, Nomor 1, Maret 2026 ISSN 2809-3704

Studi Komparatif Kinerja Kueri Pencarian pada MySQL dan PostgreSQL berdasarkan Volume Data yang Berbeda

Faisal Masri Maulana[1]*, Defara Oda Rahma Aldora[2], Qonita Ghina Anbarputri[3], Aziz Effendi Lubis[4], A. Rosyan Assiry[5]

[1-5]Program Studi Sistem Informasi, Universitas Jambi, Indonesia

*Corresponding email: faisalmasrimaulana@gmail.com

ABSTRAK – Pertumbuhan data yang cepat dalam aplikasi dan sistem informasi menjadikan efisiensi dalam mencari data sebagai aspek penting untuk kinerja basis data. Pencarian teks yang kurang efisien dapat mengakibatkan lambatnya respons sistem dan berkurangnya pengalaman pengguna. MySQL dan PostgreSQL, yang merupakan basis data open-source yang sering digunakan, memiliki perbedaan dalam cara mereka memproses kueri pencarian. Penelitian ini bertujuan untuk melakukan perbandingan antara kedua basis data dalam hal kinerja saat menjalankan kueri pencarian teks dengan variasi dalam jumlah data. Pendekatan yang digunakan dalam penelitian ini adalah eksperimen perbandingan yang menitikberatkan pada pengukuran durasi pemrosesan kueri SELECT dalam empat situasi yang berbeda, yaitu sensitif terhadap huruf besar-kecil, tidak sensitif terhadap huruf besar-kecil, penerapan ekspresi reguler, dan pencarian fonetik dengan algoritma Soundex. Pengujian dilakukan dengan menggunakan data dummy yang berjumlah 100.000, 250.000, dan 500.000 records, dan semua memiliki struktur tabel serta skenario yang sama. Hasil dari pengujian ini menunjukkan bahwa PostgreSQL berfungsi lebih baik dalam pencarian case-sensitive dan regular expression di seluruh ukuran data. MySQL memberikan hasil yang lebih baik dalam pencarian case-insensitive pada data dengan ukuran kecil hingga menengah, sedangkan PostgreSQL menunjukkan konsistensi kinerja pada data besar. Dalam hal pencarian fonetik, MySQL lebih cepat dalam hal waktu eksekusi, tetapi PostgreSQL memberikan tingkat akurasi pencarian yang lebih tinggi.

Kata Kunci: MySQL, PostgreSQL, Kueri Pencarian, Pencocokan Pola, Kinerja Basis Data.

ABSTRACT – *The rapid growth of data in applications and information systems makes efficiency in data retrieval an important aspect for database performance. Inefficient text searching can lead to slow system response times and a reduced user experience. MySQL and PostgreSQL, which are commonly used open-source databases, have differences in how they process search queries. This research aims to compare the performance of the two databases when running text search queries with variations in data volume. The approach used in this study is a comparative experiment that focuses on measuring the query processing duration of SELECT queries in four different situations: case-sensitive, case-insensitive, regular expression application, and phonetic search using the Soundex algorithm. Testing was conducted using dummy data consisting of 100.000, 250.000, and 500.000 records, all with the same table structure and scenario. The results of this testing show that PostgreSQL performs better in case-sensitive and regular expression searches across all data sizes. MySQL provides better results in case-insensitive searches on small to medium-sized data, while PostgreSQL demonstrates consistent performance on large data. In terms of phonetic search, MySQL is faster in execution time, but PostgreSQL provides a higher level of search accuracy.*

***Keywords*:** *MySQL, PostgreSQL, Search Query, Pattern Matching, Database Performance.*

PENDAHULUAN

Pada era digital seperti saat ini, peningkatan volume data pada aplikasi dan website menuntut sistem yang mampu menangani proses pencarian data secara cepat dan efisien. Proses pencarian data menjadi aspek penting karena berpengaruh langsung terhadap kinerja platform dan pengalaman pengguna. Untuk mendukung kebutuhan tersebut, *Database Management System* (DBMS)

berperan sebagai perangkat lunak yang mampu mengelola data serta mengatur akses dan pengambilan data sesuai kebutuhan pengguna [1]. Oleh karena itu, pemilihan DBMS yang tepat menjadi penting, karena DBMS merupkan bagian fundamental dalam pengembangan sebuah sistem informasi [2]. Di antara berbagai pilihan yang tersedia, MySQL dan PostgreSQL adalah dua sistem basis data *open-source* yang populer dan sering digunakan oleh pengembang sistem [3]. MySQL menempati peringkat dua sedangkan PostgreSQL menempati posisi empat pada pemeringkatan DBMS di seluruh dunia [4].

MySQL biasanya dipilih karena kinerjanya yang cepat, ringan, dan stabil di berbagai proyek web [5]. MySQL juga dikenal karena proses instalasi dan konfigurasinya yang relatif mudah, kemampuan baca-tulis yang cepat, serta komunitasnya yang luas. Karakteristiknya yang ringan membuat MySQL cocok diterapkan pada aplikasi web dan sistem yang didominasi oleh *read-heavy workloads*. Sementara itu, PostgreSQL berada di tingkat basis data objek-relasional yang lebih canggih, integritas data yang lebih kuat, dan kemampuan menangani kueri kompleks lebih efisien. PostgreSQL memberikan keunggulan tambahan karena mendukung tipe data yang lebih kompleks, tingkat kepatuhan terhadap standar SQL yang lebih baik, serta fitur-fitur canggih yang mendukung kebutuhan pengembangan aplikasi skala besar [6].

Salah satu operasi yang paling sering dilakukan dalam proses basis data adalah pencarian. Operasi ini dapat dilakukan dengan berbagai cara, dimulai dari pola sederhana yang dilakukan menggunakan LIKE *condition*, yaitu pencocokkan sebagian dari satu *string* dengan *string* lain dengan mencari bagian awal yang sesuai pola yang ditentukan. LIKE menilai kecocokan *string* berdasarkan karakter yang diatur dalam pola input [7]. Pencarian menggunakan LIKE bisa dilakukan dengan mencari *string* yang *case-sensitive* maupun *case-insensitive*. Pencarian juga dapat dilakukan dengan menggunakan *regular expression* (regex), yang merupakan notasi khusus yang dipakai untuk menemukan pola dalam teks, memungkinkan kita membuat pola kompleks untuk mencari kata atau karakter dengan ciri tertentu [8]. Terakhir, algoritma yang sering digunakan dalam proses pencarian data adalah *string matching*, yaitu algoritma yang berfungsi untuk menemukan pola berupa rangkaian karakter tertentu di dalam teks dengan ukuran data yang besar [9]. Pada algoritma ini terdapat jenis *phonetic string matching* yang termasuk metode *inexact string matching*. *Phonetic String Matching* merupakan metode pencocokan *string* yang didasarkan pada kesamaan cara pengucapan, meskipun penulisan kedua *string* yang dibandingkan berbeda [10].

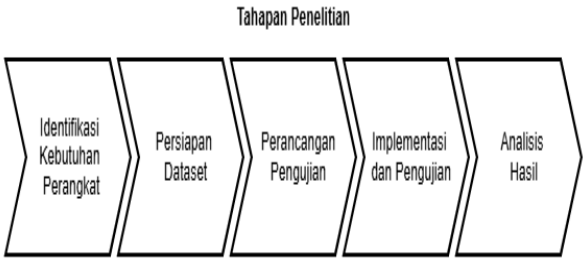
Pada penelitian terdahulu terkait perbandingan performa MySQL dan PostgreSQL, pengujian dilakukan menggunakan kueri SELECT, COUNT, dan JOIN, dimana hasilnya menunjukkan bahwa PostgreSQL memiliki performa *response time* yang lebih baik dibandingkan MySQL, sehingga dinilai lebih sesuai untuk pengelolaan database dengan skala data besar [11]. Penelitian lain juga membandingkan performa MySQL dan PostgreSQL melalui pengujian operasi CRUD (SELECT, INSERT, UPDATE, dan DELETE), dan menyimpulkan bahwa PostgreSQL secara umum unggul dalam kecepatan dan konsistensi waktu respons, terutama pada pengolahan data dalam jumlah besar [12]. Selain itu, penelitian yang melibatkan PostgreSQL, MySQL, dan MongoDB menyatakan bahwa PostgreSQL merupakan DBMS dengan performa paling unggul, serta membuktikan bahwa basis data relasional tidak selalu lebih lambat dibandingkan NoSQL, karena performa sangat bergantung pada skenario dan jenis kueri yang digunakan [13].

Berbeda dengan penelitian-penelitian tersebut yang berfokus pada kueri dasar dan operasi pengelolaan data, penelitian ini bertujuan untuk melakukan analisis komparatif performa MySQL dan PostgreSQL

pada kueri pencarian teks, meliputi *case-sensitive*, *case-insensitive*, *regex*, dan pencarian fonetik. Hasil dari penelitian ini diharapkan dapat memberikan panduan bagi developer dalam memilih DBMS yang paling optimal sesuai dengan karakteristik kebutuhan pencarian data teks.

METODE

Penelitian ini menggunakan metode penelitian komparatif yang dilakukan dengan membandingkan suatu variabel pada sampel yang berbeda untuk mengetahui ada atau tidaknya perbedaan yang dihasilkan dari perbandingan tersebut [14]. Variabel yang dibandingkan adalah performa antara MySQL dan PostgreSQL dalam mengeksekusi kueri SELECT pada skenario pencarian. Pengujian dilakukan dengan mengukur waktu eksekusi kueri SELECT pada masing-masing sistem basis data dengan beberapa skenario seperti LIKE *condition*, *regex*, dan *phonetic search* menggunakan dataset yang sama. Berikut merupakan tahap penelitian yang dilakukan.



Gambar 1. Tahapan Penelitian

2.1. Identifikasi Kebutuhan Perangkat

Tahap ini melakukan identifikasi perangkat lunak dan perangkat keras yang dibutuhkan untuk melakukan pengujian sesuai dengan spesifikasi yang dibutuhkan MySQL dan PostgreSQL. Tabel 1 merupakan spesifikasi dari *Software* yang digunakan untuk pengujian.

Tabel 1. Spesifikasi *Software* yang Digunakan

No.	Software	Versi
1	Microsoft Windows	Windows 10
2	PGAdmin	9.1.1
3	Laragon	6.0
4	DBeaver	25.3.0

Tabel 2 menunjukkan spesifikasi *Hardware* yang digunakan dalam pengujian.

Tabel 2. Spesifikasi *Hardware* yang Digunakan

No.	*Hardware*	Versi
1	RAM	8 GB
2	Prosesor	Intel Core i5-1235U (12th Gen)
3	Graphic	Intel UHD Graphics
4	Storage	SSD 477 GB

2.2. Persiapan Dataset

Dataset yang digunakan dalam penelitian ini menggunakan data *dummy* yaitu data buatan yang tidak merepresentasikan data asli, di mana nilai pada setiap atribut atau field bersifat fiktif dan digunakan hanya untuk keperluan pengujian, yang diinput atau diekstraksi menggunakan SSIS (*SQL Server Integration Service*) [15]. Dataset *dummy* ini digunakan untuk memberikan gambaran mengenai kecepatan eksekusi masing-masing database. Dataset disimpan dalam tiga tabel bernama dummy_100, dummy_250, dan dummy_500. Pengujian difokuskan pada satu kolom teks, yaitu target_pencarian. Kolom ini berisi berbagai data nama dengan karakteristik yang disesuaikan dengan kebutuhan skenario pengujian, yaitu:

1. Variasi Kapitalisasi Teks
   Dataset memuat data teks yang sama dengan perbedaan huruf besar dan kecil, seperti John atau john. Karakteristik ini digunakan untuk menguji perbedaan hasil dan performa antara pencarian *case-sensitive* dan *case-insensitive*.
2. Variasi Pola Karakter
   Dataset mengandung data teks dengan pola tertentu, seperti nama yang diawali angka atau mengandung angka, misalnya 123John atau John12. Pola ini digunakan untuk pencarian berbasis *regular expression* REGEX dalam mendeteksi pola teks tertentu.
3. Variasi Kemiripan Bunyi
   Dataset juga memuat nama dengan ejaan berbeda namun memiliki kemiripan bunyi, seperti John dan Jon. Karakteristik ini

digunakan untuk mendukung pengujian menggunakan fungsi SOUNDEX

Jumlah data yang digunakan sebanyak 100.000, 250.000, dan 500.000 baris data. Penelitian ini tidak menggunakan relasi antar tabel agar pengujian fokus pada performa pencarian string pada satu kolom teks.

2.3. Perancangan Pengujian

Tahap ini menjelaskan skenario pengujian yang dilakukan pada masing-masing basis data. Sebelum melakukan pengujian, dilakukan instalasi dan konfigurasi MySQL dan PostgreSQL pada perangkat uji. MySQL dijalankan menggunakan Laragon, sedangkan PostgreSQL dijalankan menggunakan pgAdmin. Setelah kedua basis data aktif, masing-masing baris data dihubungkan ke DBeaver sebagai *database client* untuk menjalankan kueri dan mencatat waktu eksekusi. Penggunaan DBeaver bertujuan untuk memastikan bahwa pengujian dilakukan menggunakan antarmuka yang sama sehingga hasil pengukuran tidak dipengaruhi oleh perbedaan alat pengujian.

Selanjutnya, dataset dummy yang telah dipersiapkan (dummy_100, dummy_250, dan dummy_500) diimpor ke masing-masing basis data dengan struktur tabel yang sama. Pengujian dilakukan dengan mengeksekusi kueri pencarian teks pada satu kolom, yaitu target_pencarian. Pada PostgreSQL, pencarian teks memanfaatkan operator ILIKE untuk pencarian yang *case-insensitive*, sedangkan pada MySQL digunakan keyword BINARY untuk pencarian *case-sensitive*. Selain itu, perbedaan performa juga diuji antara operator POSIX ~ dan standar REGEXP, karena masing-masing sistem basis data memiliki arsitektur dan filosofi pengembangan yang berbeda. Kueri yang digunakan berupa perintah SELECT dan diuji pada empat skenario pencarian teks, yaitu *pattern matching case-sensitive*, *pattern matching case-insensitive*, *pattern matching* berbasis *regex* dan pencarian fonetik menggunakan fungsi SOUNDEX. Pada PostgreSQL, fungsi SOUNDEX diaktifkan melalui ekstensi fuzzystrmatch. Dataset pertama yang digunakan pada penelitian ini yaitu dataset dummy_100 yang berisi 100.000 baris data seperti yang terlampir pada Tabel 3.

Tabel 3. Kueri SELECT pada dummy_100

Jenis Pengujian	MySQL	PostgreSQL
Pattern matching case-sensitive	SELECT * FROM dummy_100 WHERE target_pencarian LIKE BINARY 'John%';	SELECT * FROM dummy_100 WHERE target_pencarian LIKE 'John%';
Pattern matching case-insensitive	SELECT * FROM dummy_100 WHERE target_pencarian LIKE 'john%';	SELECT * FROM dummy_100 WHERE target_pencarian ILIKE 'john%';
Pattern matching berbasis regular expression (REGEX)	SELECT * FROM dummy_100 WHERE target_pencarian REGEXP '^[0-9]';	SELECT * FROM dummy_100 WHERE target_pencarian ~ '^[0-9]';
Phonetic search menggunakan fungsi SOUNDEX	SELECT * FROM dummy_100 WHERE SOUNDEX(target_pencarian) = SOUNDEX('John');	SELECT * FROM dummy_100 WHERE SOUNDEX(target_pencarian) = SOUNDEX('John');

Tabel 4 menggunakan dataset kedua, yaitu dataset dummy_250, yang berisi 250.000 baris data, yang merepresentasikan skala data menengah. Skenario pengujian yang digunakan tetap sama dengan pengujian pada dataset sebelumnya, sehingga perbedaan waktu eksekusi yang diperoleh dapat

dianalisis sebagai dampak dari pertambahan volume data.

Tabel 4. Kueri SELECT pada dummy_250

Jenis Pengujian	MySQL	PostgreSQL
Pattern matching case-sensitive	SELECT * FROM dummy_250 WHERE target_pencarian LIKE BINARY 'John%';	SELECT * FROM dummy_250 WHERE target_pencarian LIKE 'John%';
Pattern matching case-insensitive	SELECT * FROM dummy_250 WHERE target_pencarian LIKE 'john%';	SELECT * FROM dummy_250 WHERE target_pencarian ILIKE 'john%';
Pattern matching berbasis regular expression (REGEX)	SELECT * FROM dummy_250 WHERE target_pencarian REGEXP '^[0-9]';	SELECT * FROM dummy_250 WHERE target_pencarian ~ '^[0-9]';
Phonetic search menggunakan fungsi SOUNDEX	SELECT * FROM dummy_250 WHERE SOUNDEX(target_pencarian) = SOUNDEX('John');	SELECT * FROM dummy_250 WHERE SOUNDEX(target_pencarian) = SOUNDEX('John');

Tabel 5 menggunakan dataset ketiga, yaitu dataset dummy_500, yang berisi 500.000 baris data, yang merepresentasikan skala data besar. Masih menggunakan skenario kueri yang sama seperti kedua dataset sebelumnya, sehingga perbedaan performa waktu eksekusi terlihat lebih signifikan, terutama pada pencarian berbasis regular expression dan phonetic search yang membutuhkan pemrosesan teks lebih kompleks.

Tabel 5. Kueri SELECT pada dummy_500

Jenis Pengujian	MySQL	PostgreSQL
Pattern matching case-sensitive	SELECT * FROM dummy_500 WHERE target_pencarian LIKE BINARY 'John%';	SELECT * FROM dummy_500 WHERE target_pencarian LIKE 'John%';
Pattern matching case-insensitive	SELECT * FROM dummy_500 WHERE target_pencarian LIKE 'john%';	SELECT * FROM dummy_500 WHERE target_pencarian ILIKE 'john%';
Pattern matching berbasis regular expression (REGEX)	SELECT * FROM dummy_500 WHERE target_pencarian REGEXP '^[0-9]';	SELECT * FROM dummy_500 WHERE target_pencarian ~ '^[0-9]';
Phonetic search menggunakan fungsi SOUNDEX	SELECT * FROM dummy_500 WHERE SOUNDEX(target_pencarian) = SOUNDEX('John');	SELECT * FROM dummy_500 WHERE SOUNDEX(target_pencarian) = SOUNDEX('John');

HASIL DAN PEMBAHASAN

Perbandingan kinerja MySQL dan PostgreSQL dilakukan berdasarkan waktu eksekusi kueri pencarian yang mencakup *case sensitive*, *case insensitive*, *regular expression*, dan *phonetic search* pada data sebanyak 100.000, 250.000, dan 500.000 *record*. Setiap skenario pengujian dijalankan sebanyak lima kali, dan hasil yang disajikan merupakan rata-rata

waktu eksekusi dalam milidetik (ms) dari seluruh percobaan.

3.1. *Case Sensitive*

Pengujian *Case Sensitive* dilakukan untuk mengukur kecepatan DBMS dalam melakukan pencarian dan pencocokan data yang memiliki huruf sama persis dengan kata yang diberikan.

Tabel 6. Uji *Case Sensitive*

DBMS	100.000	250.000	500.000
MySQL	31,92	62,44	115,4
PostgreSQL	15,47	32,3	66,02

Sumber: Hasil Uji

Berdasarkan Tabel 6, PostgreSQL menghasilkan waktu eksekusi yang lebih rendah dibandingkan MySQL pada seluruh skala data yang diuji.

3.2. *Case Insensitive*

Pada skenario ini, dilakukan pengujian untuk mengukur kecepatan DBMS dalam melakukan pencarian dan pencocokan data dengan mengabaikan perbedaan kapitalisasi karakter.

Tabel 7. Uji *Case Insensitive* (ms)

DBMS	100.000	250.000	500.000
MySQL	38,802	83,184	147,60
PostgreSQL	48,65	97,03	137,80

Sumber: Hasil Uji

Berdasarkan Tabel 7, pada skenario *case-insensitive*, MySQL menunjukkan waktu eksekusi yang lebih cepat dibandingkan PostgreSQL saat jumlah data 100.000 dan 250.000. Namun, pada jumlah data 500.000, PostgreSQL menghasilkan waktu eksekusi yang lebih rendah dibandingkan MySQL.

3.3. *Regular Expression* (*Regex*)

Pada skenario ini, dilakukan pengujian untuk mengukur kecepatan DBMS dalam melakukan pencarian *string* berbasis pola menggunakan metode pencocokan teks yang memanfaatkan aturan dan simbol khusus untuk merepresentasikan pola karakter tertentu.

Tabel 8. Uji *Regex*

DBMS	100.000	250.000	500.000
MySQL	54,82	110,40	206,00
PostgreSQL	25,00	67,91	97,39

Sumber: Hasil Uji

Berdasarkan Tabel 8, PostgreSQL menunjukkan kinerja yang lebih baik dibandingkan MySQL pada seluruh skala data dalam pengujian *regular expression*, yang mengindikasikan kemampuan pemrosesan pencarian berbasis pola yang lebih efisien.

3.4. Phonetic Search

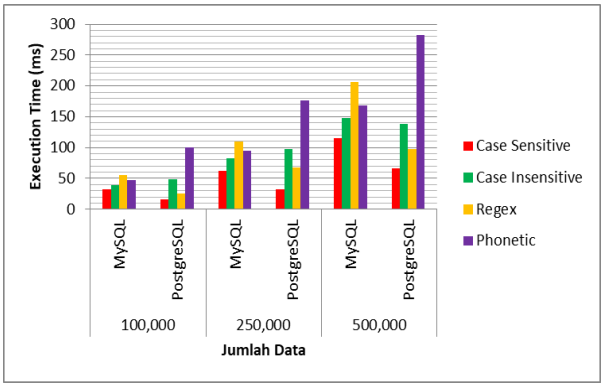
Skenario ini menggunakan algoritma Soundex untuk mencari data berdasarkan kemiripan bunyi atau pengucapan kata.

Tabel 9. Uji Phonetic Search

DBMS	100.000	250.000	500.000
MySQL	46,72	94,32	168,00
PostgreSQL	99,82	175,85	282,14

Sumber: Hasil Uji

Meskipun terlihat pada Tabel 9 MySQL mencatatkan waktu eksekusi yang lebih rendah, namun terdapat temuan penting mengenai output yang dihasilkan. Algoritma Soundex pada MySQL cenderung hanya mencocokkan kemiripan pada bagian awal kata, sehingga kurang presisi. Sebaliknya, PostgreSQL melakukan analisis fonetik yang lebih mendalam pada seluruh komponen bunyi kata, memberikan hasil yang lebih banyak dan dengan waktu eksekusi yang lebih lama. Untuk melihat performa MySQL dan PostgreSQL secara keseluruhan, hasil pengujian pada setiap skenario pencarian (*case sensitive*, *case insensitive*, *regular expression*, dan *phonetic search*) ditampilkan dalam bentuk grafik perbandingan rata-rata waktu eksekusi berdasarkan variasi jumlah data.



Gambar 2. Komparasi MySql dan PostgreSql

Berdasarkan grafik pada Gambar 2 tersebut, terlihat bahwa peningkatan jumlah data berdampak pada meningkatnya waktu eksekusi pada kedua DBMS di seluruh skenario pengujian. Pola performa yang ditunjukkan masing-masing DBMS berbeda pada tiap jenis pencarian, di mana PostgreSQL cenderung menunjukkan keunggulan pada hampir seluruh jenis pencarian dengan *execution time* yang lebih rendah pada jumlah data yang besar, sementara MySQL memiliki *execution time* yang lebih lambat dibandingkan PostgreSQL.

KESIMPULAN

Berdasarkan hasil penelitian, dapat disimpulkan bahwa MySQL dan PostgreSQL memiliki perbedaan kinerja dalam menangani berbagai jenis kueri pencarian. PostgreSQL menunjukkan waktu eksekusi yang lebih baik pada pencarian *case sensitive* dan *regular expression* di seluruh skala data, sehingga lebih efektif untuk pencarian string eksak dan berbasis pola. Pada pencarian *case insensitive*, MySQL memiliki kinerja yang lebih cepat pada jumlah data yang lebih kecil, namun pada data yang lebih besar PostgreSQL menunjukkan performa yang lebih stabil. Sementara itu, pada pencarian fonetik, MySQL memberikan waktu eksekusi yang lebih singkat, tetapi PostgreSQL mampu menghasilkan hasil pencarian yang lebih akurat.

Penelitian ini masih memiliki beberapa keterbatasan, sehingga disarankan bagi peneliti selanjutnya untuk memperluas cakupan skenario pengujian dengan menggunakan volume data yang lebih masif guna melihat batas maksimal efisiensi dari kedua DBMS. Selain itu, disarankan pula untuk melakukan analisis mendalam mengenai pengaruh penerapan berbagai jenis indeks (seperti B-Tree, GIN, atau GiST) serta optimasi konfigurasi server terhadap waktu respons kueri pencarian. Penelitian mendatang juga dapat mempertimbangkan pengujian pada lingkungan multi-user atau concurrency untuk mensimulasikan beban kerja aplikasi di dunia nyata secara lebih akurat.

REFERENSI

[1] Widodo, "DATABASE MANAJEMEN SYSTEM (DBMS) BASIC CONCEPTS," *Idaaratul 'Ulum(Jurnal Prodi MPI)*, vol. 6, no. 1, pp. 32–50, 2024, doi: https://doi.org/10.70688/idaarotululum.v6i01.386.

[2] M. Aslyza, "MANAJEMEN DATA BERBASIS DATABASE: SOLUSI UNTUK PENYIMPANAN DAN AKSES DATA YANG LEBIH EFISIEN," *J. Ilm. Nusantara(JINU)*, vol. 2, no. 3, pp. 909–917, 2025, doi: https://doi.org/10.61722/jinu.v2i3.4900.

[3] K. Y. Santika, E. Triandini, and P. D. W. Ayu, "Systematic Literature Review untuk Identifikasi Pemilihan Database Management System dalam Pengembangan Sistem," in *SEMINAR NASIONAL CORISINDO*, 2023.

[4] Red Gate Software Ltd, "DB-Engines Ranking," db-engines.com. [Online]. Available: https://db-engines.com/en/ranking

[5] F. A. N. Dhiana, Supriyono, A. Y. Badharudin, and A. Fauzan, "Implementasi Sistem Pengelolaan Program Mentoring Mahasiswa Baru Menggunakan Metode Prototyping," *Bull. Comput. Sci. Res.*, vol. 5, no. 4, pp. 287–297, 2025, doi: 10.47065/bulletincsr.v5i4.538.

[6] J. V. Dachi and J. Suhada,

"Implementasi Basis Data Sederhana Menggunakan MySQL / PostgreSQL," *Router J. Tek. Inform. dan Terap.*, vol. 3, no. 2, pp. 197–207, 2025, doi: https://doi.org/10.62951/router.v3i2.621.

[7] F. P. Utama, R. M. H. Nurhadi, D. Fitria, and M. P. Ramadhan, "STRING MATCHING DENGAN METODE SEQUENTIAL SEARCHING DAN KONDISI LIKE PADA PENCARIAN JUDUL SKRIPSI," *J. Rekursif*, vol. 9, no. 1, pp. 43–47, 2021, doi: https://doi.org/10.33369/rekursif.v9i1.14315.

[8] J. M. Bintang, M. F. Ashshidiq, and H. F. Dzakhwan, "Penerapan Algoritma String Matching dan Regular Expression pada Aplikasi Kamus Besar Bahasa Indonesia (KBBI)," *BIOS J. Teknol. Inf. dan Rekayasa Komput.*, vol. 4, no. 1, pp. 34–41, 2023, doi: https://doi.org/10.37148/bios.v4i1.57.

[9] A. F. Siregar, "Perancang Aplikasi Ensiklopedia Budaya Indonesia Dengan Menerapkan Algoritma Crochemore-Perrin Untuk Pencarian Teks," *J. Glob. Tecnol. Comput.*, vol. 2, no. 1, pp. 16–23, 2022, [Online]. Available: https://doi.org/10.47065/jogtc.v2i1.2462

[10] Muntahanah and Y. Darnita, "Aplikasi Pengarsipan Dokumen Menggunakan Metode String Matching (Studi Kasus Perpustakaan SMP Negeri 5 Seluma)," *J. Inform. UPGRIS*, vol. 5, no. 1, 2019, doi: https://doi.org/10.26877/jiu.v5i1.2895.

[11] A. D. Praba and M. Safitri, "Studi Perbandingan Performansi antara Mysql dan Postgresql," *J. KHATULISTIWA Inform.*, vol. 8, no. 2, pp. 88–93, 2020, doi: https://doi.org/10.31294/jki.v8i2.8851.

[12] S. H. Ahsa, M. B. Syahputra, A. F. F. M. Putri, and A. A. Prasetyo, "ANALISIS PERBANDINGAN PERFORMA ANTARA MySQL dan PostgreSQL," in *Prosiding Seminar Nasional Teknologi dan Sistem Informasi (SITASI)*, 2023, pp. 6–7.

[13] Y. Y. Putra, O. Purwaningrum, and R. H. Winata, "PERBANDINGAN PERFORMA RESPON WAKTU KUERI MYSQL, POSTGRESQL, DAN MONGODB," *J. Sist. Inf. Dan Bisnis Cerdas*, vol. 15, no. 1, pp. 39–48, 2022, doi: https://doi.org/10.33005/sibc.v15i1.2749.

[14] Asmini, Ismawati, and A. Hendrawan, "STUDI KOMPARATIF TINGKAT PENDAPATAN PELAKU BISNIS YANG MENGGUNAKAN STRATEGI PEMASARAN OFFLINE DAN ONLINE (Studi Pada Pelaku Bisnis Thrifting Di Pasar Utan)," *SAMALEWA (Jurnal Ris. dan Kaji. Manajemen)*, vol. 4, no. 2, pp. 293–302, 2024, doi: https://doi.org/10.58406/samalewa.v4i2.1824.

[15] D. S. Budi and H. Syahrial, "Pengoptimalan Performa Database Pada Proses Transformasi Data Pada SQL Server," *Technomedia J.*, vol. 8, no. 3, pp. 78–90, 2024.