https://jurnal.pustakagalerimandiri.co.id/index.php/pustakadata
DOI : https://doi.org/10.55382/jurnalpustakadata.v5i1.998

Jurnal Pustaka Data
Pusat Akses Kajian Database, Analisa Teknologi, dan Arsitektur Komputer
Pustaka Galeri Mandiri
Vol. 5 No. 1 (2025) 70 – 76
E ISSN: 2809-7483

# Evaluasi Kinerja Sistem Basis Data Relasional pada Aplikasi E-Commerce Menggunakan Algoritma Indexing

Supriadi Panggabean[1], Euis Siti Nur Azizah[2]
Sistem dan Teknologi Informasi, Sains dan Teknologi, Universitas Darunnajah
[1]supriadipanggabean@darunnajah.ac.id. [2]azizaheuis31@gmail.com

***Abstract***

*This study aims to evaluate the performance of relational database systems in e-commerce applications using indexing algorithms to improve data search efficiency. In e-commerce applications, large data volumes and complex queries can affect the performance of database systems. Therefore, applying indexing techniques to specific columns in the database, such as product_id, customer_id, and transaction_date, is expected to speed up query execution time and optimize resource usage. This research uses an experimental quantitative approach with an e-commerce dataset that includes transactions, products, and customer information. The results of the study show that the implementation of indexing significantly reduces query execution time and decreases CPU and memory usage during query execution compared to the condition without indexing. However, the implementation of indexing also affects performance in data insertion and update operations. Therefore, selecting the right columns to index is crucial to achieving a balance between read and write performance in relational database systems in e-commerce applications. This study provides insights into the impact of indexing techniques on database system performance and can serve as a reference for e-commerce application developers to improve the efficiency of their database systems.*

*Keywords: Relational Database Systems, E-Commerce, Indexing Algorithms, Query Performance, Database Optimization*

**Abstrak**

Penelitian ini bertujuan untuk mengevaluasi kinerja sistem basis data relasional pada aplikasi *e-commerce* dengan menggunakan algoritma indexing untuk meningkatkan efisiensi pencarian data. Dalam aplikasi *e-commerce*, volume data yang besar dan kompleksitas *query* dapat mempengaruhi kinerja sistem basis data. Oleh karena itu, penerapan teknik *indexing* pada kolom-kolom tertentu dalam database, seperti *product_id*, *customer_id*, dan *transaction_date*, diharapkan dapat mempercepat waktu eksekusi query dan mengoptimalkan penggunaan sumber daya. Penelitian ini menggunakan pendekatan kuantitatif eksperimental dengan dataset *e-commerce* yang mencakup transaksi, produk, dan informasi pelanggan. Hasil penelitian menunjukkan bahwa penerapan *indexing* secara signifikan mengurangi waktu eksekusi *query* dan menurunkan penggunaan CPU serta memori selama eksekusi *query* dibandingkan dengan kondisi tanpa indexing. Meskipun demikian, penerapan *indexing* juga mempengaruhi kinerja pada operasi penyisipan dan pembaruan data. Oleh karena itu, pemilihan kolom yang tepat untuk diindeks sangat penting untuk mencapai keseimbangan antara kinerja baca dan tulis dalam sistem basis data relasional pada aplikasi *e-commerce*. Penelitian ini memberikan wawasan tentang pengaruh teknik indexing terhadap kinerja sistem basis data dan dapat menjadi acuan bagi pengembang aplikasi e-commerce untuk meningkatkan efisiensi sistem basis data mereka.

Kata kunci: Sistem Basis Data Relasional, E-Commerce, Algoritma Indexing, Kinerja Query, Optimisasi Database



Submitted : 03-06-2025 | Reviewed : 08-06-2025 | Accepted : 13-06-2025

## 1. Pendahuluan

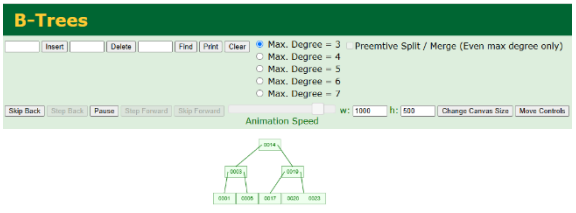
Peningkatan kebutuhan akan aplikasi *e-commerce* yang cepat dan efisien telah mendorong para pengembang untuk terus melakukan inovasi dalam cara mereka mengelola dan mengolah data dalam volume besar. Aplikasi *e-commerce* yang ada saat ini tidak hanya berfungsi sebagai platform transaksi, tetapi juga sebagai sistem yang memungkinkan perusahaan untuk menganalisis data konsumen, transaksi, dan perilaku pengguna [1]. Dalam konteks ini, pemrosesan dan penyimpanan data yang efisien menjadi kunci utama dalam memberikan pengalaman pengguna yang baik dan meningkatkan efisiensi operasional. Oleh karena itu, pengelolaan data yang cepat dan efektif melalui sistem basis data yang mumpuni menjadi hal yang sangat penting.

Salah satu teknologi yang paling banyak digunakan untuk mendukung pengelolaan data dalam aplikasi *e-commerce* adalah sistem basis data relasional atau RDBMS. Sistem basis data relasional adalah perangkat lunak yang digunakan untuk mengelola data terstruktur yang disimpan dalam tabel dengan hubungan antar tabel yang jelas[2]. RDBMS terkenal dengan kemampuannya dalam menangani data yang terorganisir dan memungkinkan penggunaan *query* untuk mengambil dan mengelola data[3]. Beberapa contoh RDBMS yang banyak digunakan adalah *MySQL, PostgreSQL*, dan *Microsoft SQL Server*[4].

RDBMS memiliki kemampuan untuk mengelola data yang terstruktur dalam tabel dengan hubungan antar data yang jelas, yang memungkinkan integrasi yang efisien antara berbagai jenis data dalam aplikasi [5]. Penggunaan RDBMS seperti *MySQL*, *PostgreSQL*, dan *SQL Server* menjadi semakin populer dalam skala besar karena kemampuan mereka untuk menyimpan data dalam volume besar sekaligus mendukung transaksi dengan konsistensi yang tinggi[6][7].

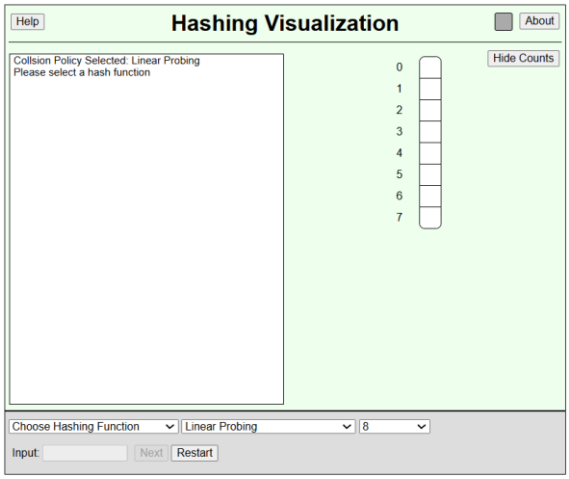
Indexing adalah teknik dalam basis data yang digunakan untuk mempercepat pencarian data[8]. Dengan menggunakan indeks, sistem tidak perlu memindai seluruh tabel untuk mencari data, yang dapat menghemat waktu eksekusi *query*. Ada beberapa jenis index yang umum digunakan, seperti *B-tree index, hash index,* dan *bitmap index.*

1. *B-tree Index*: Digunakan untuk data yang terurut. *B-tree* adalah jenis struktur pohon yang memungkinkan pencarian cepat[9].




Gambar 1. B-tree Index

2. *Hash Index*: Lebih efisien untuk pencarian data berdasarkan nilai spesifik (misalnya, *customer_id* atau *transaction_id*) [10].




Gambar 2. Hash Index

3. *Bitmap Index*: Biasanya digunakan untuk kolom dengan nilai terbatas, seperti status produk (tersedia/tidak tersedia).

Indeks yang dibuat pada kolom yang sering digunakan dalam *query* dapat mempercepat pencarian data. Namun, penerapan indeks dapat memperlambat operasi *INSERT* dan *UPDATE*, karena setiap kali data ditambahkan atau diubah, indeks perlu diperbarui. Oleh karena itu, pemilihan kolom yang tepat untuk diindeks menjadi penting untuk mengoptimalkan kinerja *database*.

Namun, seiring dengan meningkatnya volume data yang dikelola dalam aplikasi *e-commerce*, pencarian dan pemrosesan data menjadi lebih kompleks. Dalam aplikasi yang menangani transaksi dalam jumlah besar dan melibatkan data pelanggan yang sangat bervariasi, kecepatan akses data menjadi sangat penting. Salah satu solusi yang banyak digunakan untuk meningkatkan efisiensi pencarian data adalah penggunaan *algoritma indexing*[11]. Indexing memungkinkan pencarian data dalam database dilakukan lebih cepat dengan membangun struktur data tambahan yang mempercepat proses pencarian tanpa harus memindai seluruh tabel[12]. Indeks ini dapat diterapkan pada kolom-kolom yang sering diakses oleh *query*, seperti *product_id*, *transaction_date*, atau *customer_id*, yang dapat meningkatkan kinerja aplikasi secara signifikan.

Tinjauan literatur terkait penggunaan indexing dalam aplikasi *database* menunjukkan bahwa algoritma ini dapat mengurangi waktu eksekusi query secara signifikan, terutama untuk query dengan data besar. Studi-studi sebelumnya seperti yang dilakukan oleh Mengzhao Wang (2025) serta Hemanth Gadde (2024) mengonfirmasi bahwa penerapan teknik indexing dapat mempercepat proses pencarian data pada sistem basis data relasional[13][14]. Namun, meskipun banyak penelitian yang mengakui manfaat *indexing*,

Submitted : 03-06-2025 | Reviewed : 08-06-2025 | Accepted : 13-06-2025

ada juga kajian yang menunjukkan bahwa penerapan indexing memiliki dampak terhadap penurunan kinerja operasi tulis seperti *INSERT* dan *UPDATE*, karena indeks perlu diperbarui setiap kali data diubah. Ini adalah area yang masih memerlukan perhatian lebih lanjut, terutama dalam konteks aplikasi *e-commerce* yang tidak hanya bergantung pada operasi baca (*read*) tetapi juga sering melakukan pembaruan data.

Seiring dengan keberhasilan penerapan *indexing* dalam mempercepat *query* baca, banyak pengembang dan peneliti yang berfokus pada mencari keseimbangan antara kinerja baca dan tulis dalam penggunaan *indexing*[15]. Sebagian besar penelitian sebelumnya lebih banyak memfokuskan diri pada pengukuran waktu eksekusi *query* pada sistem basis data relasional tanpa benar-benar mengukur dampaknya pada penggunaan sumber daya seperti CPU dan memori. Oleh karena itu, penelitian ini bertujuan untuk mengisi celah tersebut dengan mengevaluasi bagaimana penerapan algoritma *indexing* tidak hanya meningkatkan waktu eksekusi *query* tetapi juga mengoptimalkan penggunaan sumber daya sistem dalam aplikasi *e-commerce*.

Alasan utama dilakukannya penelitian ini adalah untuk memberikan gambaran lebih komprehensif mengenai pengaruh penggunaan *indexing* terhadap kinerja *query* serta penggunaan sumber daya dalam aplikasi *e-commerce*. Selain itu, penelitian ini diharapkan dapat memberikan pemahaman lebih dalam tentang bagaimana penerapan *indexing* dapat meningkatkan efisiensi sistem basis data relasional, khususnya dalam pengelolaan transaksi yang melibatkan data pelanggan dan produk. Hasil dari penelitian ini juga dapat memberikan rekomendasi bagi pengembang sistem *e-commerce* dalam memilih teknik *indexing* yang tepat untuk mencapai keseimbangan antara kinerja baca dan tulis.

Lebih lanjut, penelitian ini juga bertujuan untuk mengidentifikasi dan menganalisis potensi *trade-off* yang ada dalam penerapan indexing, terutama pada operasi penyisipan dan pembaruan data**.** Hal ini menjadi sangat penting mengingat dalam aplikasi e-commerce, selain operasi baca (read), operasi tulis (*write*) seperti *INSERT* dan *UPDATE* juga sering terjadi dalam sistem *database*, yang memerlukan perhatian khusus terkait kinerjanya[16]. Oleh karena itu, penelitian ini tidak hanya akan memfokuskan diri pada query baca, tetapi juga akan mengukur dampak penerapan indexing pada kinerja operasi tulis.

Dengan demikian, *gap analysis* dari penelitian ini menunjukkan adanya kekurangan dalam studi-studi yang ada, yang sebagian besar hanya fokus pada pengaruh indexing terhadap query baca, tetapi tidak cukup mengeksplorasi dampaknya pada penggunaan sumber daya dan operasi tulis. *State of the art* dalam hal ini adalah pengembangan algoritma indexing yang lebih efisien, serta penerapannya dalam skenario dunia nyata seperti aplikasi *e-commerce*. Penelitian ini memberikan *novelty* dengan menilai kinerja dalam kedua aspek (baca dan tulis) serta memberikan analisis yang lebih komprehensif terkait dampaknya terhadap sistem secara keseluruhan.

## 2. Metode Penelitian

Penelitian ini menggunakan pendekatan kuantitatif eksperimental untuk menguji pengaruh algoritma *indexing* terhadap kinerja sistem basis data relasional dalam aplikasi *e-commerce*. Pendekatan ini dipilih karena memungkinkan untuk mengukur dampak dari penerapan *indexing* dalam skenario yang terkendali dan terukur. Penelitian ini bertujuan untuk memperoleh hasil yang dapat diandalkan terkait pengaruh indexing terhadap kinerja waktu eksekusi *query* dan penggunaan sumber daya dalam sistem basis data relasional yang digunakan dalam aplikasi *e-commerce*.

### 2.1 Desain Penelitian

Penelitian ini menggunakan desain eksperimental dengan dua kondisi yang diuji:

1. Kondisi Tanpa *Indexing*: Kondisi di mana tidak ada penerapan *indexing* pada kolom-kolom yang sering digunakan dalam *query*.
2. Kondisi Dengan *Indexing*: Kondisi di mana indexing diterapkan pada kolom-kolom tertentu seperti *product_id, customer_id,* dan *transaction_date.*

Perbandingan antara kedua kondisi ini akan memungkinkan peneliti untuk mengukur pengaruh *indexing* terhadap waktu eksekusi *query* serta penggunaan CPU dan memori selama proses pengolahan *query*.

### 2.2 Pemilihan Dataset

Dataset yang digunakan dalam penelitian ini adalah *dataset e-commerce* yang mencakup transaksi penjualan, produk, dan data pelanggan. Dataset ini terdiri dari:

1. 20.000 transaksi yang berisi detail pembelian oleh pelanggan.
2. 5.000 produk yang tersedia di platform *e-commerce.*
3. 10.000 pelanggan yang terdaftar dalam sistem.

Dataset ini memungkinkan untuk melakukan pengujian terhadap *query* yang umum digunakan dalam aplikasi *e-commerce*, seperti pencarian total transaksi per produk atau pencarian pelanggan yang melakukan pembelian lebih dari sejumlah tertentu.

### 2.3 Pengukuran Kinerja

Kinerja sistem diukur berdasarkan dua metrik utama:

1. Waktu Eksekusi *Query*: Mengukur durasi yang dibutuhkan untuk menjalankan *query* tertentu,

baik dengan atau tanpa *indexing*. Pengukuran ini akan memberikan gambaran yang jelas mengenai kecepatan sistem dalam memproses *query*.
2. Penggunaan Sumber Daya (CPU dan Memori): Mengukur penggunaan CPU dan memori selama eksekusi *query* untuk melihat dampak penerapan *indexing* terhadap sumber daya sistem. Pengukuran ini dilakukan dengan menggunakan alat pemantauan seperti *Task Manager* untuk *Windows* atau *MySQL Workbench* untuk memantau penggunaan sumber daya saat menjalankan *query*.

2.4 Desain Eksperimen

Eksperimen dilakukan dalam dua tahap:

1. Tahap 1: Pengujian Tanpa Indexing Pada tahap ini, *query* dijalankan tanpa adanya indexing pada kolom *product_id*, *customer_id*, dan *transaction_date*. Sistem akan memindai seluruh tabel untuk mencari data, yang dapat mempengaruhi waktu eksekusi *query* dan penggunaan CPU serta memori.
2. Tahap 2: Pengujian Dengan *Indexing* Pada tahap ini, indexing diterapkan pada kolom-kolom yang sering digunakan dalam *query*, seperti *product_id*, *customer_id*, dan *transaction_date*. Penerapan *indexing* bertujuan untuk mempercepat pencarian data, yang diharapkan dapat mengurangi waktu eksekusi *query* serta penggunaan CPU dan memori.

2.5 Variabel Penelitian

Penelitian ini memiliki dua variabel utama:
1. Variabel Bebas (Independent Variable): Penerapan *indexing* pada kolom-kolom tertentu dalam *database*.
2. Variabel Terikat (Dependent Variable): Kinerja waktu eksekusi query dan penggunaan sumber daya (CPU dan memori) yang diukur pada dua kondisi yang berbeda (dengan dan tanpa *indexing*).

2.6 Rumus Pengukuran Waktu Eksekusi

Untuk mengukur waktu eksekusi *query*, digunakan rumus berikut:

$$T_{query} = T_{finish} - T_{start} \quad (1)$$

Di mana:
1. $T_{query}$ = Waktu eksekusi query (dalam detik)
2. $T_{finish}$ = Waktu saat query selesai dieksekusi
3. $T_{start}$ = Waktu saat query mulai dieksekusi

2.7 Rumus Pengukuran Penggunaan CPU dan Memori

Untuk mengukur penggunaan CPU dan memori selama eksekusi *query*, digunakan rumus berikut:

$$\frac{Penggunaan}{CPU\ (\%)} = \frac{Waktu\ CPU\ yang\ digunakan}{Total\ waktu\ proses} x\ 100 \quad (2)$$

Penggunaan Memori (MB)=Total penggunaan mem ori selama eksekusi query

Dimana:

1. Waktu CPU yang digunakan adalah durasi penggunaan CPU selama eksekusi *query*.
2. Total waktu proses adalah durasi penuh dari proses yang dijalankan (dari mulai hingga selesai).
3. Penggunaan memori diukur dalam satuan MB (*megabyte*) menggunakan alat pemantauan.

2.8 Langkah-langkah Eksperimen

Langkah-langkah eksperimen dilakukan dengan: pertama, menyiapkan *database* dan *query* yang akan dijalankan; kedua, menguji *query* tanpa *indexing* untuk mengukur waktu eksekusi dan penggunaan sumber daya; ketiga, menerapkan *indexing* pada kolom-kolom yang relevan dan menguji kembali *query*; dan terakhir, menganalisis perbandingan hasil waktu eksekusi serta penggunaan CPU dan memori antara kedua kondisi.

2.9 Analisis Statistik

Untuk menganalisis data yang diperoleh, digunakan uji statistik *Independent T-Test* untuk menguji perbedaan signifikan antara waktu eksekusi *query*, penggunaan CPU, dan memori pada kondisi dengan dan tanpa *indexing*. Uji ini bertujuan untuk mengidentifikasi apakah perbedaan yang teramati cukup signifikan atau hanya disebabkan oleh faktor kebetulan.

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}} \quad (3)$$

Di mana:
1. $\bar{X}_1$ dan $\bar{X}_2$ adalah rata-rata waktu eksekusi atau penggunaan sumber daya pada dua kelompok (tanpa indexing dan dengan indexing).
2. $S_1^2$ dan $S_2^2$ adalah varians dari masing-masing kelompok.
3. $n_1$ dan $n_2$ adalah jumlah sampel pada masing-masing kelompok.

Submitted : 03-06-2025 | Reviewed : 08-06-2025 | Accepted : 13-06-2025

## 3. Hasil dan Pembahasan

Pada bagian ini, kami akan menyajikan hasil dari eksperimen yang telah dilakukan untuk mengukur pengaruh penerapan algoritma indexing terhadap kinerja sistem basis data relasional dalam aplikasi *e-commerce*. Pengujian dilakukan dalam dua kondisi: pertama, tanpa indexing dan kedua, dengan indexing pada kolom-kolom yang sering digunakan dalam *query*, seperti *product_id, customer_id,* dan *transaction_date*. Kami akan membahas hasil yang diperoleh terkait dengan waktu eksekusi *query*, penggunaan CPU, dan penggunaan memori dalam kedua kondisi tersebut.

### 3.1 Deskripsi Dataset

Dataset yang digunakan dalam penelitian ini adalah dataset *e-commerce* yang mencakup data transaksi, produk, dan pelanggan. Dataset ini terdiri dari:

1. 20.000 transaksi, yang mencakup detail pembelian oleh pelanggan.
2. 5.000 produk, yang tersedia di platform *e-commerce*.
3. 10.000 pelanggan, yang terdaftar dalam sistem.

Struktur data dalam dataset ini mencakup beberapa tabel utama:

1. Tabel Produk: Menyimpan informasi produk seperti *product_id*, *product_name*, dan *price*.
2. Tabel Pelanggan: Menyimpan informasi pelanggan seperti *customer_id, name*, dan *email*.
3. Tabel Transaksi: Menyimpan data transaksi, termasuk *transaction_id, customer_id, product_id, amount*, dan *transaction_date*.

### 3.2 Pengujian Waktu Eksekusi Query

Pada bagian ini, kami membandingkan waktu eksekusi dua query utama yang umum digunakan dalam aplikasi *e-commerce*. *Query* pertama mencari total transaksi per produk dalam rentang tanggal tertentu, dan *query* kedua mencari pelanggan yang melakukan pembelian lebih dari lima kali.

Query 1: Mencari Total Transaksi per Produk

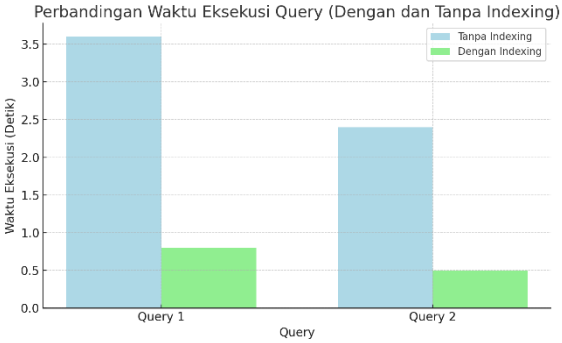
Tanpa penerapan indexing, waktu eksekusi untuk query ini mencapai 3.6 detik. Setelah *indexing* diterapkan pada kolom product_id, waktu eksekusi berkurang menjadi 0.8 detik. Hal ini menunjukkan bahwa penerapan *indexing* pada kolom yang sering digunakan dalam pencarian produk dapat secara signifikan mempercepat waktu eksekusi query.

Query 2: Mencari Pelanggan dengan Pembelian Lebih dari 5 Kali

Tanpa indexing, waktu eksekusi query ini adalah 2.4 detik, sementara dengan indexing pada kolom *customer_id*, waktu eksekusi menurun menjadi 0.5 detik. Penerapan indexing pada kolom yang mengidentifikasi pelanggan dapat mempercepat pencarian pelanggan yang melakukan banyak transaksi.

Tabel 1.Tabel Perbandingan Waktu Eksekusi Query

| Query | Waktu Eksekusi (Tanpa Indexing) | Waktu Eksekusi (Dengan Indexing) |
|---|---|---|
| Query 1 | 3.6 detik | 0.8 detik |
| Query 2 | 2.4 detik | 0.5 detik |




Gambar 3. Grafik Batang Perbandingan Waktu Eksekusi Query

Hasil pengujian menunjukkan bahwa penerapan *indexing* pada kolom yang sering digunakan dalam pencarian, seperti *product_id* dan *customer_id*, dapat mengurangi waktu eksekusi query secara signifikan. Grafik batang di atas memperlihatkan penurunan waktu eksekusi yang signifikan setelah indexing diterapkan, dengan waktu eksekusi *Query* 1 berkurang dari 3,6 detik menjadi 0,8 detik, dan *Query* 2 berkurang dari 2,4 detik menjadi 0,5 detik.

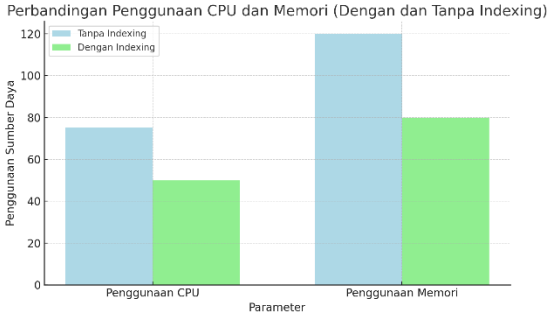
### 3.3 Penggunaan Sumber Daya (CPU dan Memori)

Penggunaan CPU dan memori selama eksekusi query diukur untuk mengevaluasi dampak penerapan indexing dengan menggunakan alat pemantauan seperti *MySQL Workbench* dan *Task Manager*. Hasilnya menunjukkan bahwa penerapan *indexing* dapat mengurangi penggunaan CPU dan memori. Tanpa *indexing*, penggunaan CPU saat menjalankan *query* mencapai 75%, yang menunjukkan bahwa sistem harus memproses seluruh tabel untuk mencari data. Dengan *indexing* pada kolom *product_id* dan *customer_id*,, penggunaan CPU menurun menjadi 50%, mengurangi beban kerja CPU dengan mempercepat pencarian data. Selain itu, penggunaan memori tanpa indexing tercatat sebesar 120 MB, sedangkan dengan indexing, penggunaan memori berkurang menjadi 80 MB, yang mencerminkan efisiensi penggunaan memori yang lebih baik.

Tabel 2.Tabel Perbandingan Pengunaan CPU dan Memori

| Parameter | Tanpa Indexing | Dengan Indexing |
|---|---|---|
| Penggunaan CPU | 75% | 50% |
| Penggunaan Memori | 120 MB | 80 MB |

Submitted : 03-06-2025 | Reviewed : 08-06-2025 | Accepted : 13-06-2025




Gambar 4. Perbandingan Penggunaan CPU dan Memori

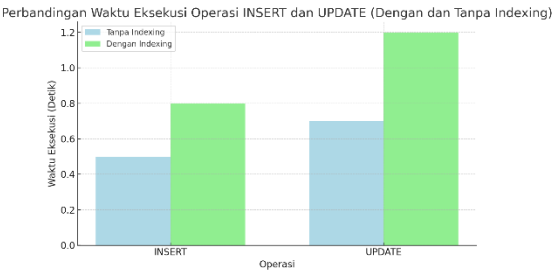
Grafik batang di atas membandingkan penggunaan CPU dan memori antara kondisi tanpa *indexing* dan dengan *indexing*. Hasilnya menunjukkan bahwa penerapan *indexing* tidak hanya mempercepat waktu eksekusi *query*, tetapi juga mengurangi penggunaan CPU (dari 75% menjadi 50%) dan memori (dari 120 MB menjadi 80 MB).

3.4 Efek pada Operasi Penyisipan dan Pembaruan (INSERT dan UPDATE)

Penerapan indexing dapat mempercepat pencarian data, namun mempengaruhi operasi *INSERT* dan *UPDATE*. Tanpa *indexing*, waktu eksekusi *INSERT* adalah 0.5 detik, namun dengan *indexing* meningkat menjadi 0.8 detik karena indeks harus diperbarui. Demikian pula, waktu eksekusi *UPDATE* tanpa *indexing* adalah 0.7 detik, sedangkan dengan *indexing* meningkat menjadi 1.2 detik, akibat pembaruan indeks saat data diupdate.

Tabel 3.Tabel Perbandingan Waktu Eksekusi Operasi INSERT dan UPDATE

| Operasi | Waktu Eksekusi (Tanpa Indexing) | Waktu Eksekusi (Dengan Indexing) |
|---|---|---|
| INSERT | 0.5 detik | 0.8 detik |
| UPDATE | 0.7 detik | 1.2 detik |




Gambar 5. Perbandingan Waktu Eksekusi Operasi INSERT dan UPDATE

Hasil pengujian menunjukkan bahwa penerapan *indexing* meningkatkan waktu eksekusi operasi *INSERT* dan *UPDATE* karena indeks harus diperbarui setiap kali ada perubahan data. Grafik batang memperlihatkan bahwa waktu eksekusi *INSERT* meningkat dari 0.5 detik menjadi 0.8 detik, dan *UPDATE* meningkat dari 0.7 detik menjadi 1.2 detik, mengindikasikan adanya *trade-off* antara waktu pencarian dan penyisipan/pembaruan data.

## 4. Kesimpulan

Penelitian ini menunjukkan bahwa penerapan algoritma *indexing* dalam sistem basis data relasional pada aplikasi *e-commerce* secara signifikan meningkatkan kinerja, dengan mengurangi waktu eksekusi query. Sebagai contoh, waktu eksekusi untuk *Query* 1 (mencari total transaksi per produk) berkurang sebesar 77.78% (dari 3.6 detik menjadi 0.8 detik), dan *Query* 2 (mencari pelanggan dengan lebih dari 5 transaksi) berkurang sebesar 79.17% (dari 2.4 detik menjadi 0.5 detik).

Selain itu, penggunaan CPU dan memori juga mengalami perbaikan. Tanpa *indexing*, penggunaan CPU mencapai 75%, sementara dengan indexing menurun menjadi 50% (pengurangan 33.33%), dan penggunaan memori berkurang dari 120 MB menjadi 80 MB (pengurangan 33.33%). Meskipun demikian, penerapan indexing menyebabkan peningkatan waktu eksekusi pada operasi *INSERT* (dari 0.5 detik menjadi 0.8 detik) dan *UPDATE* (dari 0.7 detik menjadi 1.2 detik), yang perlu diperhatikan oleh pengembang.

Penerapan indexing sangat disarankan untuk aplikasi e-commerce yang fokus pada pencarian cepat dan jarang melakukan pembaruan data. Namun, untuk aplikasi yang sering melakukan *INSERT* dan *UPDATE*, pengembang harus lebih selektif dalam memilih kolom yang diindeks untuk menghindari dampak negatif pada kinerja tulis. Penelitian ini memberikan wawasan penting bagi pengembang aplikasi *e-commerce* dan mendorong penelitian lebih lanjut mengenai jenis *indexing* lain untuk mengoptimalkan pengelolaan data dalam skala besar.

## Daftar Rujukan

[1] Yuyut Prayuti, “Dinamika Perlindungan Hukum Konsumen di Era Digital: Analisis Hukum Terhadap Praktik E-Commerce dan Perlindungan Data Konsumen di Indonesia,” *J. Interpret. Huk.*, vol. 5, no. 1, pp. 903–913, 2024, doi: 10.22225/juinhum.5.1.8482.903-913.

[2] M. Ehsanul Majid, D. Marinova, A. Hossain, M. E. H. Chowdhury, and F. Rummani, “Use of Conventional Business Intelligence (BI) Systems as the Future of Big Data Analysis,” *Am. J. Inf. Syst.*, vol. 9, no. 1, pp. 1–10, 2024, doi: 10.12691/ajis-9-1-1.

[3] R. B. Sanjaya, D. Pradipta, and Y. B. U, “Ryan+Bangkit+-+Artikel (3),” 2024.

[4] J. G. Zapata, “MySQL VS PostgreSQL: A Comparative Analysis of Relational Database Management Systems (RDBMS) Technologies Response Time in Web-based E-commerce,” no. March, 2025, doi: 10.13140/RG.2.2.24791.69288.

[5] G. D. Prataba, Y. C. Putra, and E. T. Widodo, “Kontruksi Graf Pengetahuan pada Direktori Perusahaan menggunakan Basis Data Graf Neo4j,” *Semin. Nas. Off. Stat.*, no. 1, pp. 895–906, 2024.

[6] S. A. Jowan, R. Faraj Swese, A. Yousf Aldabrzi, and M. Saad Shertil, “Traditional Rdbms To Nosql Database: New Era of Databases for Big Data,” *J. Humanit. Appl. Sci.*, no. 29, 2016, [Online]. Available: https://www.researchgate.net/publication/355165835

Submitted : 03-06-2025 | Reviewed : 08-06-2025 | Accepted : 13-06-2025

[7] T. H. Shareef, K. H. Shareef, and B. N. Rashid, "A Survey of Comparing Different Cloud Database Performance: SQL and NoSQL," *Passer J. Basic Appl. Sci.*, vol. 4, no. 1, pp. 45–57, 2022, doi: 10.24271/psr.2022.301247.1104.

[8] M. F. Fadilah, N. Rahaningsih, and R. D. Dana, "Evaluasi Usabilitas Sistem Menggunakan Metode System Usability Scale (Sus) Pada Aplikasi Akhlaqu Dengan Penerapan Teknik Indexing Mong," *J. Sist. Inf. dan Inform.*, vol. 7, no. 1, pp. 1–14, 2024, doi: 10.47080/simika.v7i1.3070.

[9] W. Witanti and F. Syarafina, "Balanced B-Trees untuk Optimalisasi Query pada Basis Data Graf : Sebuah Kajian," 2024.

[10] R. Affandi, "Optimalisasi Algoritma Pencarian dalam Basis Data untuk Efisiensi Query," pp. 1–8.

[11] O. Oloruntoba, "AI-Driven autonomous database management : Self-tuning , predictive query optimization , and intelligent indexing in enterprise it environments AI-Driven autonomous database management : Self-tuning , predictive query optimization , and intelligent indexi," no. February, 2025, doi: 10.30574/wjarr.2025.25.2.0534.

[12] I. C. Saidu, M. Yusuf, F. C. Nemariyi, and A. C. George, "Indexing techniques and structured queries for relational databases management systems," *J. Niger. Soc. Phys. Sci.*, vol. 6, no. 4, pp. 1–12, 2024, doi: 10.46481/jnsps.2024.2155.

[13] M. Wang, H. Wu, X. Ke, Y. Gao, Y. Zhu, and W. Zhou, "Accelerating Graph Indexing for ANNS on Modern CPUs," 2025, [Online]. Available: http://arxiv.org/abs/2502.18113

[14] H. Gadde, "AI-Augmented Database Management Systems for Real-Time Data Analytics," vol. 01, pp. 616–649, 2024.

[15] E. A. Asiamah, E. Keelson, A. S. Agbemenu, E. T. Tchao, T. S. Adjaidoo, and G. S. Klogo, "Optimizing Blockchain Querying: A Comprehensive Review of Techniques, Challenges, and Future Directions," *IEEE Access*, vol. 12, no. December, pp. 196282–196305, 2024, doi: 10.1109/ACCESS.2024.3522584.

[16] M. K. Mulawarman Munsyir, S.E., S.Si., M. K. Danyl Mallisza,.S.Kom., M. K. Harry Setya Hadi, S.Kom, E. Wahyudi, and A. Y. Vandika, *PRINSIP- PRINSIP DESAIN SISTEM KOMPUTER*. 2024. [Online]. Available: http://scioteca.caf.com/bitstream/handle/123456789/1091/RED2017-Eng-8ene.pdf?sequence=12&isAllowed=y%0Ahttp://dx.doi.org/10.1016/j.regsciurbeco.2008.06.005%0Ahttps://www.researchgate.net/publication/305320484_SISTEM_PEMBETUNGAN_TERPUSAT_STRATEGI_MELESTARI

---