INSOLOGI: Jurnal Sains dan Teknologi
https://journal.literasisains.id/index.php/INSOLOGI
ISSN 2828-4984 (Media Online) | ISSN 2828-4992 (Media Cetak)
Vol. 3 No. 6 (Desember 2024) 663-673
DOI: 10.55123/insologi.v3i6.4596
Submitted: 14-11-2024 | Accepted: 30-11-2024 | Published: 25-12-2024

Perbandingan Performa Pencarian Data Berbasis Teks dengan Dan Tanpa Full-text Index pada Basis Data MySQL

Imam Thoib[1*], Beda Puspita Candra[2], Nafis Sururi[3], Danang Satya Nugraha[4]
[1,2,3,4]Program Studi Sistem Informasi, Institut Teknologi Mojosari, Nganjuk, Indonesia
Email: [1*]ithoib@itmnganjuk.ac.id, [2]bedapuspita@itmnganjuk.ac.id, [3]nafissururi@itmnganjuk.ac.id, [4]danangsatyan@itmnganjuk.ac.id

Abstract

The amount of text-based data generated today is increasing drastically, especially with the development of digital technology and the internet. With the rapid growth of data, one of the main challenges faced is how to store and search large amounts of text-based data efficiently. This study analyzes the effect of using Full-text index in MySQL on the performance of text-based data searches, especially on large datasets. The study was conducted using a marketplace product dataset containing 2.3 million data. Testing includes searching using one, two, and three-word keywords in tables with and without full-text index. The test results show that tables with full-text index have faster search times than tables without Full-text index. Statistical analysis using the t-test produces a p value <0.05, indicating a significant effect of using Full-text index on data search efficiency. These findings confirm that full-text index is an effective solution to improve the performance of large-scale text data searches.

Keywords: *Database, Mysql, Data Search, Full-text Search, Full-text Index.*

Abstrak

Jumlah data berbasis teks yang diproduksi saat ini meningkat secara drastis, terutama dengan berkembangnya teknologi digital dan internet. Dengan pertumbuhan data yang sangat cepat, salah satu tantangan utama yang dihadapi adalah bagaimana menyimpan dan mencari data berbasis teks dalam volume besar dengan efisien. Penelitian ini menganalisis pengaruh penggunaan *full-text index* pada MySQL terhadap kinerja pencarian data berbasis teks, khususnya pada dataset besar. Penelitian dilakukan menggunakan dataset produk *marketplace* berisi 2,3 juta data. Pengujian mencakup pencarian menggunakan kata kunci satu, dua, dan tiga kata pada tabel dengan dan tanpa *full-text index*. Hasil pengujian menunjukkan bahwa tabel dengan *full-text index* memiliki waktu pencarian lebih cepat dibandingkan tabel tanpa *full-text index*. Analisis statistik dengan uji *t-test* menghasilkan nilai $p < 0,05$, yang mengindikasikan adanya pengaruh signifikan dari penggunaan *full-text index* terhadap efisiensi pencarian data. Temuan ini menegaskan bahwa *full-text index* adalah solusi efektif untuk meningkatkan performa pencarian data teks dalam skala besar.

Kata Kunci: Basis Data, Mysql, Pencarian Data, *Text Search, Full-text Index.*

1. PENDAHULUAN

Dalam era transformasi digital, volume data yang dihasilkan oleh berbagai sistem informasi meningkat secara eksponensial. Hal ini mendorong kebutuhan akan teknik pengelolaan data yang efisien, terutama dalam aspek penyimpanan dan pencarian informasi (Dirgantara & Suryadarma, 2014; Yudistira, 2021). Salah satu tantangan utama adalah bagaimana menyediakan mekanisme pencarian teks yang cepat dan akurat pada basis data dengan jumlah data yang besar (Grivin Mokodaser & Dwijayanti, n.d.; Homepage, Putry Avrylya, & Susetyo, 2024). MySQL, sebagai salah satu sistem manajemen basis data relasional (RDBMS) yang populer, menyediakan fitur *full-text*

index untuk mendukung pencarian teks yang efisien (Šušter & Ranisavljević, 2023).

Full-text index adalah salah satu solusi yang dirancang untuk meningkatkan performa pencarian teks pada basis data (Salunke & Ouda, 2024). Indeks ini memungkinkan sistem untuk melakukan pencarian *full-text* dengan waktu pemrosesan yang jauh lebih singkat dibandingkan pencarian tanpa indeks (Madavi, Patel, Jain, & Kate, 2023). Penelitian sebelumnya menunjukkan bahwa pencarian berbasis indeks secara signifikan mempercepat waktu pencarian pada kumpulan data yang besar (Hajar, Utami, & Al Fatta, 2022). Namun, implementasi *full-text index* juga memiliki keterbatasan tertentu, seperti konsumsi ruang penyimpanan tambahan dan kompleksitas pemeliharaan (Fotopoulos, 2022).

Dalam implementasinya, *full-text index* digunakan untuk melakukan pencarian teks pada kolom dengan tipe data tertentu, seperti *CHAR, VARCHAR, dan TEXT* (Chatziparaskevas, 2023). Pencarian teks ini relevan dalam berbagai aplikasi, termasuk pencarian dokumen, katalog produk, dan aplikasi berbasis data besar lainnya. Namun, dampak performa dari penggunaan *full-text index* dibandingkan pencarian teks tanpa indeks seringkali kurang mendapat perhatian dalam penelitian berbasis aplikasi nyata.

Perbandingan performa antara pencarian dengan dan tanpa indeks sangat bergantung pada ukuran dataset, pola pencarian, dan kompleksitas query. Studi menunjukkan bahwa pada dataset kecil, perbedaan performa mungkin tidak signifikan, tetapi pada dataset yang lebih besar, penggunaan indeks dapat mengurangi waktu pencarian yang signifikan (Hajar et al., 2022). Oleh karena itu, analisis lebih lanjut diperlukan untuk memahami kapan dan bagaimana *full-text index* dapat memberikan keuntungan maksimal.

Selain performa, faktor lain yang perlu dipertimbangkan adalah efisiensi sumber daya. *Full-text index* membutuhkan ruang penyimpanan tambahan (Maesaroh, Gunawan, Lestari, Tsaurie, & Fauji, 2022) untuk menyimpan struktur indeks, yang dapat memengaruhi kapasitas penyimpanan dan performa keseluruhan sistem. Studi lain mengemukakan bahwa konsumsi memori tambahan ini dapat menjadi penghambat pada sistem dengan sumber daya terbatas, terutama jika data sering diperbarui atau dimodifikasi (Hajar et al., 2022).

Konteks penggunaan *full-text index* dalam aplikasi nyata juga menjadi perhatian penting. Misalnya, dalam sistem pencarian berita atau dokumen hukum, *query* pencarian sering kali kompleks dan melibatkan operator logis seperti *AND, OR,* dan *NOT* (Chatziparaskevas, 2023). Penelitian menunjukkan bahwa *full-text index* mampu menangani *query* semacam ini dengan efisiensi yang tinggi, tetapi performanya dapat menurun pada *query* yang sangat spesifik atau dengan parameter pencarian yang berlebihan.

Namun, meskipun banyak penelitian menunjukkan keuntungan penggunaan *full-text index*, ada juga studi yang melaporkan tantangan dalam implementasinya. Sebagai contoh, penelitian lain mengindikasikan bahwa pada dataset dengan data yang sangat dinamis, proses pembaruan indeks dapat menjadi hambatan utama (Hajar et al., 2022). Hal ini disebabkan oleh kebutuhan untuk membangun ulang indeks setelah setiap pembaruan data, yang dapat meningkatkan latensi sistem secara keseluruhan (Fotopoulos, 2022).

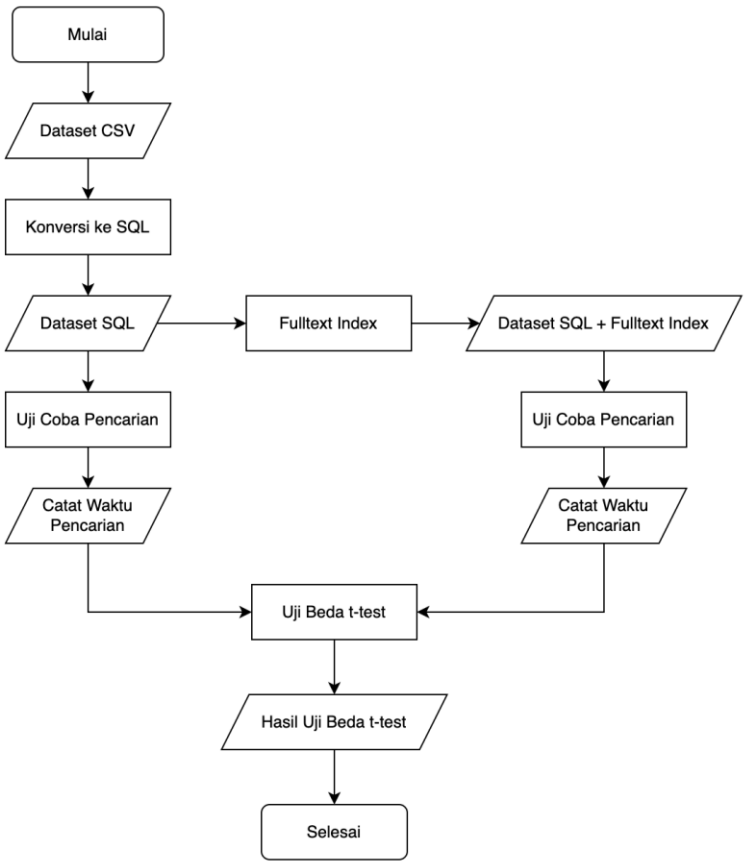
Di sisi lain, pencarian tanpa indeks, meskipun lebih lambat dalam beberapa kasus, memiliki keunggulan dalam hal kesederhanaan dan penggunaan sumber daya. Tanpa adanya indeks, *query* pencarian dilakukan langsung pada data mentah, yang dapat lebih fleksibel dalam konteks dataset yang kecil atau data yang sering berubah (Hajar et al., 2022). Namun, pendekatan ini kurang efektif pada dataset besar yang memerlukan waktu pemrosesan yang signifikan untuk setiap pencarian (Asyhari, Subchan Mauludin, &

Tengah, 2019).

Studi komparatif yang mendalam diperlukan untuk mengevaluasi efektivitas *full-text index* dalam berbagai skenario penggunaan. Analisis semacam ini dapat memberikan wawasan yang lebih baik tentang kapan sebaiknya menggunakan indeks dan kapan pencarian tanpa indeks lebih diutamakan. Penelitian ini bertujuan untuk menjawab pertanyaan tersebut dengan mengukur performa pencarian berbasis teks pada MySQL, baik dengan maupun tanpa *full-text index*, menggunakan dataset berukuran besar dan dengan berbagai variasi kata kunci.

2. METODOLOGI PENELITIAN

Langkah-langkah yang dilakukan dalam penelitian ini disajikan dalam alur penelitian pada gambar 1.



Gambar 1. Alur penelitian

2.1 Dataset

Untuk uji coba pencarian data berbasis teks, peneliti menggunakan dataset yang berisi 2,3 juta produk marketplace yang disediakan pada website involve.asia. Dataset ini dipilih karena di dalamnya terdapat banyak kolom dengan isi data mayoritas teks. Dataset ini awalnya disajikan dalam format CSV (*comma separated values*). Setelah itu, dataset CSV tersebut dikonversi ke dalam format yang kompatibel dengan sistem manajemen basis data (DBMS) MySQL untuk memanfaatkan fitur pencarian berbasis teks yang lebih efisien. Proses konversi ini mencakup pemetaan kolom-kolom dalam file CSV ke dalam struktur tabel yang sesuai dalam MySQL. Data yang telah berhasil dikonversi kemudian diimpor ke dalam sebuah tabel di MySQL, yang memiliki 22 kolom berbeda yang masing-masing menyimpan informasi terkait produk, seperti nama produk, deskripsi,

harga, kategori, dan lainnya. Gambar 2 menunjukkan struktur tabel secara rinci, yang menggambarkan bagaimana data terorganisasi dalam basis data dan siap untuk diproses lebih lanjut dalam uji coba pencarian teks.

#	Name	Type	Collation	Attributes	Null	Default
1	ID	int(11)			Yes	NULL
2	Name	text	utf8mb4_general_ci		Yes	NULL
3	Category	text	utf8mb4_general_ci		Yes	NULL
4	Description	text	utf8mb4_general_ci		Yes	NULL
5	ImageURL	text	utf8mb4_general_ci		Yes	NULL
6	Location	text	utf8mb4_general_ci		Yes	NULL
7	Shopname	text	utf8mb4_general_ci		Yes	NULL
8	ShopURL	text	utf8mb4_general_ci		Yes	NULL
9	ProductURL	text	utf8mb4_general_ci		Yes	NULL
10	Price	text	utf8mb4_general_ci		Yes	NULL
11	Condition	text	utf8mb4_general_ci		Yes	NULL
12	category_level_1	text	utf8mb4_general_ci		Yes	NULL
13	category_level_2	text	utf8mb4_general_ci		Yes	NULL
14	category_level_3	text	utf8mb4_general_ci		Yes	NULL
15	brand	text	utf8mb4_general_ci		Yes	NULL
16	courier	text	utf8mb4_general_ci		Yes	NULL
17	seller_rating	int(11)			Yes	NULL
18	CostCenterNumber	int(11)			Yes	NULL
19	bebasongkir	text	utf8mb4_general_ci		Yes	NULL
20	sold_count	int(11)			Yes	NULL
21	link_lite	text	utf8mb4_general_ci		Yes	NULL
22	shop_type	text	utf8mb4_general_ci		Yes	NULL

Gambar 2. Struktur tabel

Pada tahap selanjutnya, untuk keperluan eksperimen, tabel yang telah diimpor tersebut diduplikasi menjadi dua tabel dengan isi yang sama persis. Tabel pertama diberi nama produk1, yang tidak menggunakan *index*, sementara tabel kedua diberi nama produk2, yang memiliki kolom *Name* dan *Description* yang diberi *full-text index*. Pemilihan kolom *Name* dan *Description* untuk diberi *full-text index* didasarkan pada pertimbangan bahwa kolom-kolom tersebut menyimpan informasi penting berupa nama dan deskripsi produk, yang umumnya menjadi acuan utama dalam pencarian produk di platform marketplace. Nama produk dan deskripsi adalah elemen yang sering digunakan oleh pengguna untuk mencari produk yang relevan dengan kebutuhan mereka. Dengan menggunakan *full-text index* pada kolom ini, diharapkan proses pencarian dapat dilakukan dengan lebih cepat dan efisien, mengingat kedua kolom tersebut berfungsi sebagai titik fokus dalam pencarian berbasis teks.

```
CREATE FULLTEXT INDEX index_produk2
ON produk2(Name,Description);
```

Gambar 3. Perintah SQL untuk pembuatan *Full-text index*

Perintah SQL yang digunakan untuk membuat *full-text index* pada kolom *Name* dan *Description* pada tabel produk2 dapat dilihat pada gambar 3.

#	Name	Type	Collation	Attributes	Null	Default
1	ID	int(11)			Yes	NULL
2	Name	text	utf8mb4_general_ci		Yes	NULL
3	Category	text	utf8mb4_general_ci		Yes	NULL
4	Description	text	utf8mb4_general_ci		Yes	NULL
5	ImageURL	text	utf8mb4_general_ci		Yes	NULL
6	Location	text	utf8mb4_general_ci		Yes	NULL
7	Shopname	text	utf8mb4_general_ci		Yes	NULL
8	ShopURL	text	utf8mb4_general_ci		Yes	NULL
9	ProductURL	text	utf8mb4_general_ci		Yes	NULL
10	Price	text	utf8mb4_general_ci		Yes	NULL
11	Condition	text	utf8mb4_general_ci		Yes	NULL
12	category_level_1	text	utf8mb4_general_ci		Yes	NULL
13	category_level_2	text	utf8mb4_general_ci		Yes	NULL
14	category_level_3	text	utf8mb4_general_ci		Yes	NULL
15	brand	text	utf8mb4_general_ci		Yes	NULL
16	courier	text	utf8mb4_general_ci		Yes	NULL
17	seller_rating	int(11)			Yes	NULL
18	CostCenterNumber	int(11)			Yes	NULL
19	bebasongkir	text	utf8mb4_general_ci		Yes	NULL
20	sold_count	int(11)			Yes	NULL
21	link_lite	text	utf8mb4_general_ci		Yes	NULL
22	shop_type	text	utf8mb4_general_ci		Yes	NULL

Gambar 4. Struktur tabel produk2 dengan *Full-text index*

Struktur tabel produk1 yang tidak menggunakan *index* dapat dilihat pada gambar 2, yang menunjukkan bagaimana data tersimpan secara langsung dalam setiap kolom tanpa adanya optimasi untuk pencarian teks. Di sisi lain, struktur tabel produk2, yang sudah dilengkapi dengan *full-text index* pada kolom *Name* dan *Description*, dapat dilihat pada gambar 4. Pada gambar 4, terdapat perbedaan yang jelas dibandingkan dengan gambar 2, yaitu pada kolom *Name* dan *Description* yang dilengkapi dengan ikon kunci di sebelahnya. Ikon kunci ini menunjukkan bahwa kolom-kolom tersebut telah diindeks dengan *full-text index*.

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Rename Drop	index_produk2	FULLTEXT	No	No	Name			Yes	
					Description			Yes	

Create an index on 1 columns Go

Gambar 5. Daftar *index* pada tabel produk2

Perbedaan mendasar lainnya yang dapat dilihat pada struktur tabel produk2 terdapat pada bagian bawah, di mana terdapat informasi detil terkait *index* yang telah dibuat, seperti yang ditampilkan pada gambar 5. Pada bagian ini, dapat dilihat secara jelas berbagai informasi yang berkaitan dengan *index* yang telah diterapkan pada kolom *Name* dan *Description*. Informasi ini meliputi nama *index*, tipe *index*, serta kolom-kolom yang dijadikan *index*. Nama *index* biasanya mencerminkan identitas dari *index* tersebut, yang dalam hal ini dapat berupa nama yang spesifik seperti "*index_produk2*" yang telah dibuat untuk tabel produk2. Tipe *index* menunjukkan jenis *index* yang diterapkan, dalam hal ini adalah *full-text index*, yang dirancang khusus untuk pencarian teks dalam kolom yang memiliki banyak data berbasis teks.

2.2 Kata Kunci Pencarian

Untuk melakukan uji coba pencarian data pada basis data berbasis teks diperlukan berbagai variasi kata kunci. Kata kunci yang digunakan untuk uji coba dapat dilihat pada tabel 1. Kata kunci di bawah ini dipilih berdasarkan kata kunci yang relevan dengan dataset yang akan diuji.

Tabel 1. Kata kunci pencarian

1 kata	2 kata	3 kata
sepatu	sepatu wanita	sepatu wanita formal
kaos	kaos polos	kaos polos hitam
laptop	laptop toshiba	laptop toshiba satellite
speaker	speaker logitech	speaker logitech stereo
pisau	pisau blender	pisau blender philips
buku	buku anak	buku baca anak
bluetooth	bluetooth headset	bluetooth headset wireless
softcase	softcase iphone	softcase iphone 6s
meja	meja lipat	meja lipat portable
lipstik	lipstik merah	lipstik merah muda

Pada tabel 1 disebutkan ada 3 kelompok kata kunci yaitu 1 kata, 2 kata dan 3 kata. Setiap kelompok kata kunci terdapat 10 kata kunci di dalamnya yang akan digunakan untuk uji coba pencarian data.

2.3 Uji Coba Pencarian Data

Pencarian data pada tabel produk1 dilakukan menggunakan operator '*LIKE'*, yang merupakan metode pencarian teks yang umum digunakan dalam SQL. Operator ini memungkinkan pencarian berdasarkan pola atau substring dalam kolom. Sementara itu, pada tabel produk2 yang telah diimplementasikan *full-text index*, pencarian data dilakukan dengan menggunakan operator '*MATCH AGAINST*' (Ferdian, Hermawan, & Edy, 2023). Operator ini dirancang khusus untuk pencarian teks yang lebih cepat dan efisien, memanfaatkan *full-text index* untuk mempercepat proses pencarian di kolom yang telah diindeks. Setiap pencarian data, baik pada tabel produk1 maupun produk2, dibatasi untuk menampilkan hanya maksimal 1000 baris data, menggunakan operator '*LIMIT'* dalam *query* SQL. Pembatasan ini dilakukan untuk menghindari beban yang berlebihan pada sistem, serta untuk memastikan hasil yang ditampilkan tetap relevan dan mudah diproses. Contoh *query* yang digunakan dalam eksperimen ini untuk kedua tabel dapat dilihat pada gambar 6 dan 7.

```
SELECT ID, Name, Price
FROM produk1
WHERE Name
LIKE '%sepatu%'
OR Description
LIKE '%sepatu%'
LIMIT 1000;
```

Gambar 6. Contoh *query* pencarian data dengan operator *LIKE*

```
SELECT ID, Name, Price
FROM produk2
WHERE
MATCH(Name,Description)
AGAINST('sepatu')
LIMIT 1000;
```

Gambar 7. Contoh *query* pencarian data dengan operator *MATCH AGAINST*

Query pencarian data dilakukan pada aplikasi *command prompt*, kemudian dicatat waktu yang diperlukan setiap pencarian data dalam satuan milisekon (ms).

2.4 Spesifikasi Perangkat Uji Coba

Untuk melakukan uji coba pencarian data, selain menyiapkan dataset yang relevan, juga diperlukan persiapan perangkat keras dan perangkat lunak yang memadai. Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam penelitian ini dapat mempengaruhi kinerja dan kecepatan dalam proses uji coba pencarian data. Adapun spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam penelitian ini dapat dilihat pada Tabel 2.

Tabel 2. Spesifikasi Perangkat Uji Coba

Processor	Intel Core i5-6300
RAM	8 GB
SSD	128 GB
Sistem Operasi	Windows 11
Versi MySQL	8.2

Kecepatan yang dihasilkan pada saat uji coba pencarian data tentunya akan berbeda jika digunakan perangkat dengan spesifikasi yang berbeda.

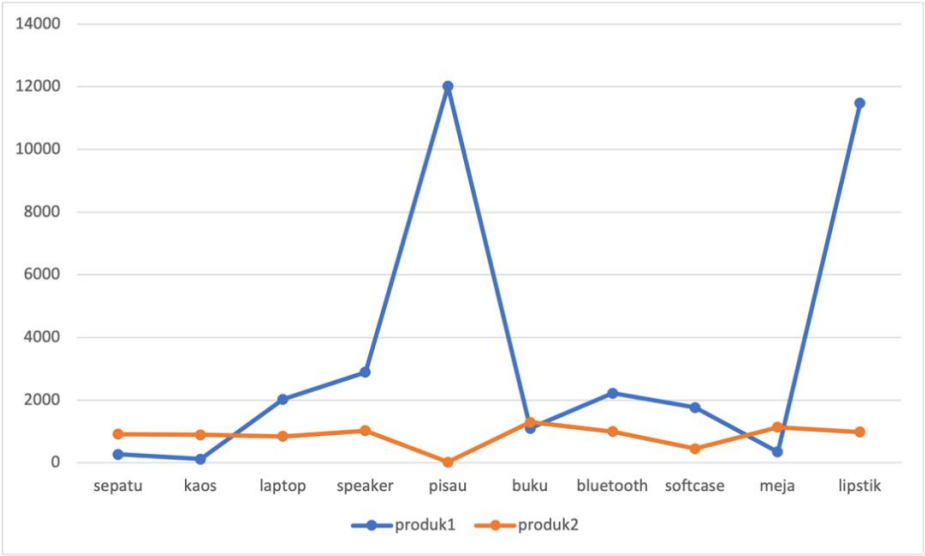
2.5 Uji Beda *T-test*

Untuk mengetahui pengaruh dari penggunaan *full-text index* pada pencarian data berbasis teks pada basis data MySQL, dilakukan uji beda *t-test*. Dengan uji beda *t-test* nantinya akan dapat diketahui signifikan atau tidaknya pengaruh dari penggunaan *full-text index*. Nilai α yang digunakan adalah 0.05 dengan tingkat kepercayaan 95%. Apabila hasil uji beda *t-test* $< \alpha$, maka pengaruh penggunaan *full-text index* adalah singinifikan, sebaliknya jika nilai uji beda *t-test* $> \alpha$ maka pengaruh penggunaan *full-text index* bisa dikatakan tidak signifikan.

3. HASIL DAN PEMBAHASAN

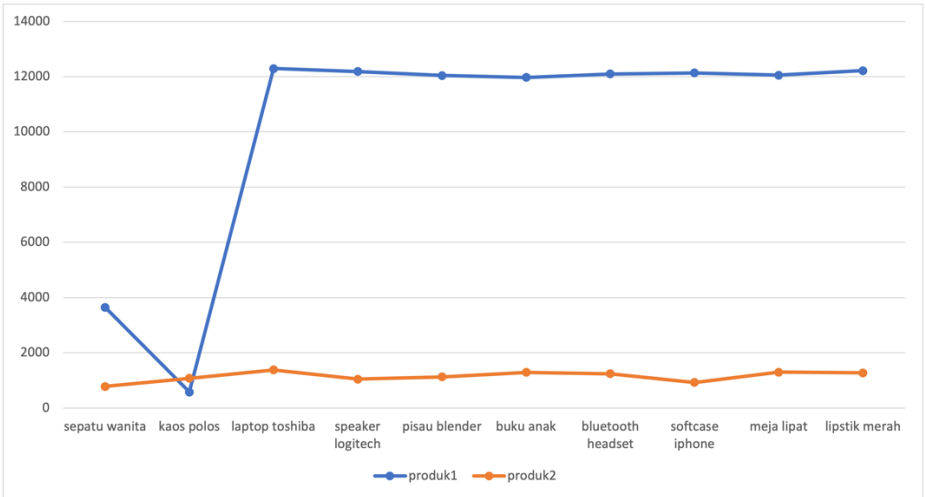
3.1 Hasil Uji Kecepatan Pencarian Data

Pada tahapan ini telah dilakukan uji coba pencarian data dengan kata kunci 1 kata, 2 kata dan 3 kata pada tabel produk1 dan produk2. Hasil uji coba disajikan pada gambar 8 s.d. 10 dengan mencatat waktu yang dibutuhkan dalam satuan milisekon (ms) untuk menjalankan *query* pencarian data untuk masing-masing kata kunci.



Gambar 8. Grafik kecepatan pencarian kata kunci satu kata

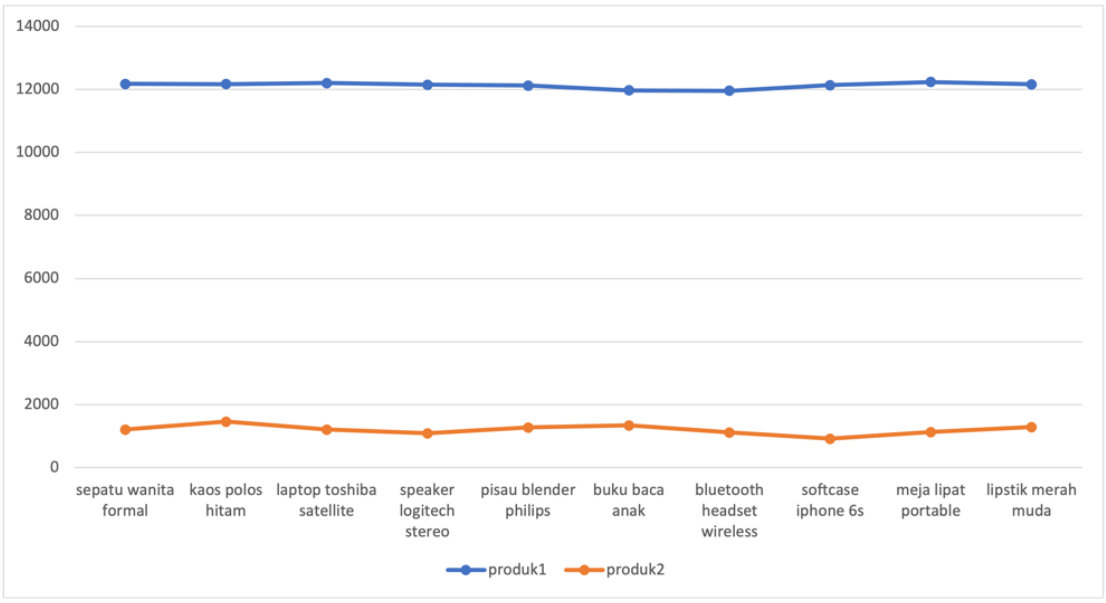
Hasil uji coba pencarian data menggunakan kata kunci 1 kata pada gambar 8 menunjukkan adanya peningkatan kecepatan pencarian data pada tabel produk2 untuk 7 kata kunci yaitu laptop, *speaker*, buku, *bluetooth*, *softcase* dan lipstik. Sedangkan 3 kata kunci lainnya lebih cepat pada tabel produk1 yang tidak menggunakan *full-text index*. Rata-rata kecepatan pencarian data menggunakan kata kunci 1 kata pada tabel produk1 adalah 3.421,2 ms, sedangkan pada tabel produk2 adalah 853,8 ms. Hal ini menunjukkan penggunaan *full-text index* pada tabel produk2 lebih cepat 4 kali dalam pencarian data berbasis teks menggukanan kata kunci 1 kata dibanding pencarian pada tabel produk1 yang tidak menggunakan *full-text index.*



Gambar 9. Grafik kecepatan pencarian kata kunci dua kata

Uji coba dilanjutkan dengan menggunakan kata kunci 2 kata, dari hasil uji coba yang disajikan pada gambar 9 menunjukkan adanya peningkatan kecepatan pencarian data pada tabel produk2 untuk 9 kata kunci yaitu sepatu wanita, laptop toshiba, *speaker logitech*, pisau *blender*, buku anak, *bluetooth headset*, *softcase* iphone, meja lipat dan lipstik merah. Sedangkan 1 kata kunci lainnya tidak mengalami peningkatan yaitu kata kunci kaos polos. Rata-rata kecepatan pencarian data menggunakan kata kunci 2 kata pada tabel produk1 adalah 10.122,7 ms, sedangkan pada tabel produk2 adalah 1.143,3 ms. Hal ini menunjukkan penggunaan *full-text index* pada tabel produk2 lebih cepat 9 kali dalam pencarian data berbasis teks menggukanan kata kunci 2 kata dibanding pencarian pada tabel produk1 yang tidak menggunakan *full-text index*.

Pada tahap terakhir dilakukan uji coba pencarian data dengan kata kunci 3 kata, dari hasil uji coba yang disajikan pada gambar 10 menunjukkan adanya peningkatan kecepatan pencarian data untuk semua kata kunci pada tabel produk2 yang menggunakan *full-text index*. Rata-rata kecepatan pencarian data menggunakan kata kunci 3 kata pada tabel produk1 adalah 12.124,4 ms, sedangkan pada tabel produk2 adalah 1.205 ms. Hal ini menunjukkan penggunaan *full-text index* pada tabel produk2 lebih cepat 10 kali dalam pencarian data berbasis teks menggukanan kata kunci 3 kata dibanding pencarian pada tabel produk1 yang tidak menggunakan *full-text index*.



Gambar 10. Grafik kecepatan pencarian kata kunci tiga kata

Dari semua uji coba pencarian data, baik menggunakan kata kunci 1 kata, 2 kata maupun 3 kata menunjukkan 25 kata kunci menunjukkan peningkatan waktu pencarian data ketika menerapkan *full-text index* pada tabel produk2. Sedangkan untuk 5 kata kunci lainnya tidak mengalami peningkatan dan lebih cepat ketika tanpa menggunakan *full-text index*. Hal ini menunjukkan bahwa penggunaan *full-text index* dapat meningkatkan performa pencarian data berbasis teks pada basis data dengan volume besar. Pencarian data menggunakan *full-text index* lebih cepat 8 kali dibanding pencarian data pada tabel tanpa *index*.

3.2 Hasil Uji Beda *T-test*

Pada tahapan ini telah dilakukan uji beda *t-test* pada waktu yang diperlukan untuk menjalankan *query* pencarian data pada tabel produk1 dan produk2 dengan kata kunci 1 kata, 2 kata dan 3 kata. Hasil uji beda *t-test* disajikan pada tabel 4.

Tabel 5. Hasil uji beda *t-test*

	produk1	**produk2**
Mean	8556,1	1067,366667
Variance	26263124,23	83757,13678
Observations	30	30
Pearson Correlation	0,302188514	
Hypothesized Mean Difference	0	
df	29	
t Stat	8,130566765855890	
P(T<=t) one-tail	0,000000002882334	
t Critical one-tail	1,699127026533500	
P(T<=t) two-tail	0,000000005764667	
t Critical two-tail	2,045229642132700	

Dari hasil uji beda *t-test* diperoleh nilai *t-test* kurang dari 0.05. Hal ini menunjukkan bahwa adanya pengaruh yang signifikan dalam pencarian data berbasis teks pada basis data MySQL ketika menggunakan *full-text index*.

Perbedaan performa pencarian dengan dan tanpa *full-text index* terletak pada cara data diolah dan efisiensi algoritma yang digunakan. Tanpa *full-text index*, basis data harus memindai seluruh tabel untuk menemukan kecocokan teks, yang sangat membebani sumber daya, terutama pada tabel besar. Pendekatan ini lambat karena menggunakan operasi pencocokan string linear, seperti *LIKE '%keyword%'*, yang memerlukan evaluasi per karakter pada setiap baris data. Sebaliknya, *full-text index* menggunakan struktur data khusus untuk mengindeks kata-kata dalam teks. Hal ini memungkinkan pencarian langsung ke lokasi data yang relevan tanpa memindai seluruh tabel, sehingga jauh lebih cepat dan hemat sumber daya.

4. KESIMPULAN

Berdasarkan hasil penelitian dan serangkaian uji coba yang telah dilakukan, dapat disimpulkan bahwa penggunaan *full-text index* pada basis data MySQL untuk pencarian data berbasis teks menunjukkan peningkatan signifikan dalam kecepatan pencarian data. Hasil uji beda *t-test* secara keseluruhan menunjukkan adanya pengaruh yang signifikan terhadap kinerja pencarian data ketika menggunakan *full-text index*, yang tercermin pada nilai *p* yang kurang dari 0,05. Temuan ini membuktikan bahwa *full-text index* dapat secara efektif meningkatkan efisiensi pencarian pada dataset besar, sehingga menjadi solusi yang tepat untuk mengatasi tantangan dalam pencarian data berbasis teks.

Dalam penelitian ini, mode pencarian data berbasis teks yang digunakan adalah mode bawaan MySQL, yaitu *Natural Language Mode*. Oleh karena itu, peneliti menyarankan agar penelitian selanjutnya mempertimbangkan untuk menguji mode pencarian lainnya, seperti *Boolean Mode* dan *Query Expansion Mode*. Penggunaan mode-mode ini dengan berbagai variasi *modifier* dapat memberikan wawasan lebih lanjut mengenai perbedaan kinerja pencarian dan memungkinkan penyesuaian metode pencarian yang lebih sesuai dengan kebutuhan spesifik pengguna. Dengan demikian, penelitian lebih lanjut diharapkan dapat memperluas pemahaman tentang optimasi pencarian teks dalam basis data MySQL dan meningkatkan aplikabilitasnya dalam berbagai konteks.

REFERENCES

Asyhari, I., Subchan Mauludin, M., & Tengah, J. M. (2019). IMPLEMENTASI FULL TEXT SEARCH PADA SISTEM INFORMASI PERPUSTAKAAN MENGGUNAKAN LARAVEL, *1*(1), 1–9. Retrieved from www.elastic.co

Chatziparaskevas, M. (2023). *Full-text indexing in open source DBMS*. University of Macedonia.

Dirgantara, U., & Suryadarma, M. (2014). Revolusi Industri 4.0: Big Data, Implementasi Pada Berbagai Sektor Industri (Bagian 2). *Jurnal Sistem Informasi Universitas Suryadarma*, *10*(1). https://doi.org/10.35968/jsi.v10i1.991

Ferdian, S., Hermawan, A., & Edy, E. (2023). Designing a Chatbot Based on Full-Text Search and 3D Modelling as a Promotional Media. *JUITA : Jurnal Informatika*, *11*(1), 19. https://doi.org/10.30595/juita.v11i1.15237

Fotopoulos, G. (2022). *Comparison of full-text search performance in relational and non-relational database systems*. UNIVERSITY OF THE PELOPONNESE & NCSR "DEMOCRITOS."

Grivin Mokodaser, W., & Dwijayanti, M. (n.d.). Implementasi Metode Indexing dan Penggunaan Subquery untuk Optimalisasi Database Rawat Jalan Rumah Sakit Menggunakan Mysql. *Cogito Smart Journal |*, *8*(2).

Hajar, A., Utami, E., & Al Fatta, H. (2022). Penggunaan Fulltext Indexing Untuk Meningkatkan Efisiensi Pencarian Data Pada Basis Data MYSQL Use of Fulltext Indexing to Improve Data Search Efficiency in MYSQL Databases, *12*(2). https://doi.org/10.30700/jst.v12i2.1264

Homepage, J., Putry Avrylya, T., & Susetyo, Y. A. (2024). Perbandingan Response Time Pencarian Menggunakan Text Indexing pada MongoDB dan ArangoDB Berbasis Web, *4*(3), 777–785.

Madavi, C., Patel, C., Jain, V., & Kate, P. V. (2023). MySQL Select Query Optimization Using Self-Join. *International Journal of Research Publication and Reviews*, *4*(4), 2564–2571. https://doi.org/10.55248/gengpi.234.4.36035

Maesaroh, S., Gunawan, H., Lestari, A., Tsaurie, M. S. A., & Fauji, M. (2022). Query Optimization In MySQL Database Using Index. *International Journal of Cyber and IT Service Management*, *2*(2), 104–110. https://doi.org/10.34306/ijcitsm.v2i2.84

Salunke, S. V., & Ouda, A. (2024). A Performance Benchmark for the PostgreSQL and MySQL Databases. *Future Internet*, *16*(10). https://doi.org/10.3390/fi16100382

Šušter, I., & Ranisavljević, T. (2023). Optimization of MySQL database. *Journal of Process Management and New Technologies*, *11*(1–2), 141–151. https://doi.org/10.5937/jouproman2301141q

Yudistira, N. (2021). Peran Big Data dan Deep Learning untuk Menyelesaikan Permasalahan Secara Komprehensif. *EXPERT: Jurnal Manajemen Sistem Informasi Dan Teknologi*, *11*(2), 78. https://doi.org/10.36448/expert.v11i2.2063