J-ICON, Vol. 9 No. 1, Maret 2021, pp. 1~10
DOI: 10.35508/jicon.v9i1.3598

ANALISIS MODEL RESTRUKTURISASI KUERI DALAM MENGOPTIMASI WAKTU RESPONS EKSEKUSI DATA PADA BASIS DATA RELASIONAL MYSQL PHP 7.2.27

Olivia Maria Inacio Tavares[1], Sartje Mala Rangkoly[2], Sarah Bunda Desi Bawan[3]
Khifni Beyk Ahmad[4], Ema Utami[5], Muhammad Syukri Mustafa[6]

[1,2,3,4,5]Program Studi Magister Teknik Informatika, Universitas AMIKOM Yogyakarta
[1]Email: olivia.1280@students.amikom.ac.id
[2]Email: sartje.1291@students.amikom.ac.id
[3]Email: sarah1276@students.amikom.ac.id
[4]Email: khifni.1298@students.amikom.ac.id
[5]Email: ema_u@amikom.ac.id
[6]STMIK Dipanegara Makassar, Jl. Perintis Kemerdekaan, Sulawesi Selatan
[6]Email: syukri@dipanegara.ac.id.

ABSTRAK

Basis data merupakan sekumpulan data pembentuk informasi yang bermanfaat bagi pengelolanya baik dalam suatu instansi maupun organisasi untuk memudahkan proses pengelolaan data agar lebih terjamin keefektifan, dan keamanannya. Basis data yang baik memiliki performa waktu respon yang cepat dan akurat dalam pemrosesan eksekusi data. Untuk dapat mengelola basis data dibutuhkan bahasa atau kueri khusus yang dikenal dengan *Structured Query Language* (SQL). Seiring meningkatnya kebutuhan akan informasi secara cepat tentunya dibutuhkan penanganan SQL yang tepat pula. Berdasarkan kendala tersebut maka akan dilakukan analisis pengujian untuk mengoptimasi waktu respons kueri pada *Database Management System* (DBMS) yang banyak dimanfaatkan saat ini yaitu basis data relasional *My Structured Query Language* (MySQL) dengan penggunaan PHP versi 7.2.27 yang mana akan diujikan pada Sistem Layanan Aspirasi dan Informasi Kelurahan Oebufu (SELMA) yang diakses secara langsung melalui panel *localhost* untuk pengujian dengan jumlah *dataset* sebanyak 1000 data. Adapun pengujian dikhususkan pada efektivitas penggunaan kueri SELECT dalam memanggil dan menampilkan data dengan merestrukturisasi penggunaan klausa. Penelitian ini bertujuan untuk membantu meningkatkan pengelolaan data dengan memberikan hasil analisis perbandingan waktu respons di antara model kueri umum dengan hasil kueri yang telah dioptimasi agar dapat ditentukan struktur kueri yang tepat dan efisien. Hasilnya terbukti dengan dilakukannya optimasi kueri didapatkan presentasi optimasi waktu yang signifikan yaitu sebesar 78% jauh lebih cepat dibandingkan dengan kueri awal. Penelitian ini menghasilkan kesimpulan bahwa optimasi berperan penting dalam mengefisiensikan waktu respons kueri basis data.
Kata kunci: basis data, optimasi, *structured query language*, MySQL, restrukturisasi kueri.

ABSTRACT

Database is a collection of data that forming an information that useful for managers in an organization to facilitate data management process to become more effective and safely guarantee. A good database has fast and accurate response time performance in data execution processing. To manage the database, it requires a special language or query known as Structure Query Language (SQL). Increase need for fast information, will need a proper SQL as well. Therefore, through this research, a test analysis will be used to optimize the query response time in database management system commonly used today, namely the My Structured Query Language (MySQL) using PHP version of 7.2.27 relational database which will be tested on the Oebufu Village Aspiration and Information Service System (SELMA) that acessed from localhost panels with a total of 1000 test data records. The test is devoted to the effectiveness of using SELECT queries in calling and displaying data through restructuring the use of clauses. This study aims to help improve data management by providing the results of the comparative analysis of response times between the general query model and the optimized query results in order to determine the correct and efficient query structure. The result is proven by doing query optimization, it is found that the presentation of time optimization is significant, which is 78% much faster than the initial query, through this test it can be concluded that optimization plays an important role in streamlining the response time of database queries.
Kata kunci: database, optimization, structured query language, MySQL, query restructurization.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

1. PENDAHULUAN

Saat ini penggunaan teknologi basis data sangat berperan penting dalam memudahkan proses pengelolaan data dengan beragam jumlah dan kebutuhan dari pengelolanya. Tidak sedikit organisasi yang telah memanfaatkan implementasi basis data dalam menyukseskan penerapan sistem informasi yang dijalankan. Hal ini dikarenakan pemanfaatan sistem informasi dewasa ini telah menjadi salah satu tuntutan utama untuk dipenuhi setiap lapisan organisasi maupun instansi dalam membantu memudahkan kewajiban operasionalnya [1]. Sistem Layanan Aspirasi dan Informasi Kelurahan Oebufu (SELMA) merupakan salah satu contoh implementasi pengembangan sistem informasi dan teknologi informasi (SI/TI) dalam instansi pemerintah yang menerapkan pemanfaatan teknologi basis data. Salah satu tujuan diterapkannya sistem informasi tersebut yaitu untuk memudahkan proses penyebaran infomasi secara cepat dan akurat kepada berbagai pihak yang membutuhkan [2].

Berdasarkan [3] basis data dapat dikatakan efektif bila mampu mengelola sekumpulan data dengan baik dilihat dari lama waktu respon transaksi data yang dibutuhkan serta mampu memberikan hasil pengelolaan yang akurat dan tepat sesuai kebutuhan pengguna. Seiring bertambahnya jumlah data dengan model formulasi kueri belum optimal, tentu mengakibatkan kinerja akses data menjadi lebih lambat dan kurang efektif. Apabila hal ini terjadi, secara otomatis dapat mengganggu arus lalu lintas transaksi data pada sistem informasi terkait. Dalam menanggulangi hal tersebut maka diperlukan optimalisasi kueri dengan membandingkan kecepatan respon eksekusi data menggunakan berangam model klausa yang tepat.

Penelitian ini bertujuan untuk membantu meningkatkan kinerja pengelolaan data pada SELMA dengan dilakukannya analisis optimalisasi model kueri pada model *Database Management System* (DBMS) yang akan diujikan yaitu pada basis data relasional *My Structured Query Language* (MySQL) dengan penggunan PHP versi 7.2.27. Manfaat yang dihasilkan melalui penelitian ini yaitu dapat menjadi dasar tolok ukur bagi pengembang atau pengelola sistem informasi lainnya untuk melakukan optimasi pada basis data yang dibangun, serta menjadi referensi pembanding model optimalisasi mana yang tepat, efisien dan efektif untuk diimplementasikan pada model basis data tersebut. Pengujian ini akan dilakukan dengan menggunakan data dari SELMA dengan jumlah data uji sebanyak 1000 *record*. Adapun pengujian dikhususkan pada salah satu model kueri *Data Manipulation Language* (DML) yang digunakan untuk memanggil dan menampilkan data dalam basis data yaitu dalam penggunaan kueri SELECT dengan merestrukturisasi penggunaan klausa yang paling optimal untuk meminimalkan waktu respons kueri. Diantaranya yaitu untuk penggunaan SELECT-SYMBOL OPERATOR, IN, INNER JOIN-WHERE, DISTINCT, EXISTS, NOT IN-LEFT JOIN, OR-UNION dan LIKE. Pengujian dilakukan sebanyak 3 kali dalam mengoptimasi kueri lalu dicatat waktu respons yang dihasilkan dan disajikan dalam bentuk tabel serta grafik agar dapat diketahui besaran waktu perbandingannya.

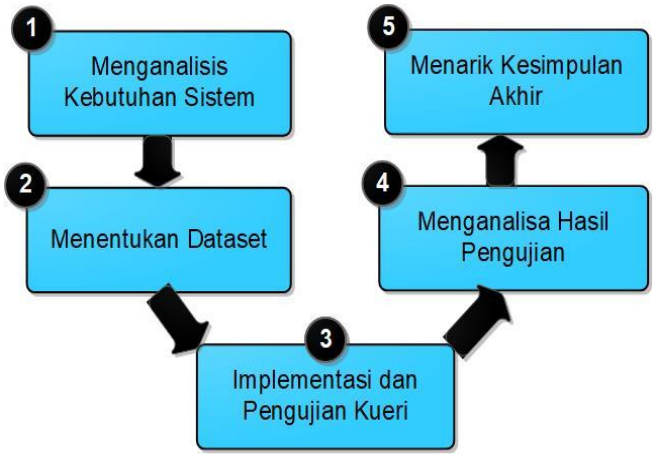
Proses optimalisasi kueri ini akan dilakukan dengan melihat model struktur kueri yang paling sesuai dan optimal dilihat dari kecepatan waktu eksekusi data yang dihasilkan menggunakan klausa-klausa yang akan diujikan pada fungsi SELECT tersebut. Yang mana salah satu bentuk optimasi kueri yang paling umum dilakukan yaitu melalui restrukturisasi kueri asli menjadi model kueri baru yang telah dioptimalisasi diharapkan dapat menghasilkan waktu respons transaksi data menjadi jauh lebih optimal dibandingkan dengan waktu respons yang sebelumnya [4]. Optimalisasi kueri saat ini telah menjadi parameter yang seharusnya dipenuhi dalam merancang dan mengoperasikan pengelolaan data melalui suatu model basis data [5], hal ini dapat dilihat dari banyaknya penelitian terdahulu yang membahas mengenai optimasi kueri dalam hubungannya untuk mempercepat proses transaksi data di antaranya yaitu: optimasi *query* sederhana guna kecepatan *query* pada *database server* [6], optimasi *query* pada *Human Resource Information System* (HRIS) di Universitas XYZ [7], optimasi basis data spasial untuk mengidentifikasi lampu APILL di Kabupaten Sleman [8], optimalisasi *query* MySQL menggunakan Index [9], analisa performansi *query* pada *database* Smell [10].

Berdasarkan penelitian-penelitian tersebut seluruhnya menyimpulkan bahwa optimasi kueri pada setiap basis data mampu meningkatkan efisiensi kecepatan waktu respons transaksi data. Pada umumnya penelitian-penelitian tersebut menggunakan metode optimasi penambahan index serta simbol operator dan yang umum dilakukan yaitu melalui restrukturisasi ulang susunan kueri agar menjadi lebih optimal, efektif dan efisien dibandingkan dengan kueri asal. Hal ini membuktikan bahwa masih terdapat banyak metode lannya yang dapat diujicobakan untuk mengoptimalisasikan kueri pada basis data. Selain itu pada penelitian ini optimasi akan dilakukan untuk model manajemen basis data MySQL dengan mengujikan hasil restrukturisasi kueri menggunakan 8 model klausa pendukung fungsi SELECT dengan susunan yang beragam dan berbeda dengan penelitian-penelitian sebelumnya agar dapat diperoleh hasil yang jauh lebih tepat dan akurat dengan menambahkan lebih banyak bentuk pengujian tentunya akan memberikan kebaharuan yang bernilai dalam penelitian terkait optimasi kueri basis data.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

2. MATERI DAN METODE

Dalam penelitian ini data yang dimanfaatkan terfokuskan pada 3 tabel dalam basis data aplikasi SELMA yang kemudian diuji dengan menggunakan ke 8 kombinasi klausa yang telah dikelompokkan. Penelitian ini melalui lima langkah dalam alur kerangka yang disusun diantaranya yaitu: menganalisis kebutuhan sistem dengan mengidentifikasi kebutuhan perangkat keras serta perangkat lunak yang digunakan dalam penelitian, lalu menentukan *dataset* penelitian yang diakses melalui *localhost* dimana *dataset* yang digunakan berjumlah 1000 data yang didapat dan diinputkan sebagai tambahan pada ketiga tabel uji yaitu tabel aparatur, tabel rt serta tabel rw yang untuk dapat melihat lama waktu respon kueri yang dihasilkan oleh sistem tersebut, selanjutnya yaitu melakukan implementasi dan pengujian terhadap kueri SELECT dengan memanfaatkan ke 8 model kombinasi klausa untuk melihat dan menentukan model restrukturisasi yang paling tepat digunakan dan mampu untuk mengefisiensikan lama waktu respon yang diperoleh dalam pemanggilan data pada sistem basis data tersebut, langkah yang terakhir yaitu menganalisis setiap hasil dari pengujian yang dihasilkan. Untuk jelasnya alur metode ditampilkan pada gambar 1 berikut.



Gambar 1. Metode penelitian

1. Menganalisis Kebutuhan Sistem

Pada tahapan ini dilakukan analisis kebutuhan sistem di antaranya mengelompokkan perangkat keras serta perangkat lunak yang dipergunakan pada penelitian mengoptimasi waktu respons kueri. Berikut ditampilkan pada tabel 1 untuk perangkat keras dan tabel 2 untuk perangkat lunak yang dipergunakan dalam penelitian ini.

Tabel 1. Spesifikasi perangkat keras yang digunakan

No	Perangkat Keras	Version
1.	Memory	8,00 GB RAM
2.	Processor	Intel Core i7-7700HQ (2,80 GHz) DDR4
3.	Graphic	Intel ^{R} HD Graphics
4.	Harddisk	1 TB

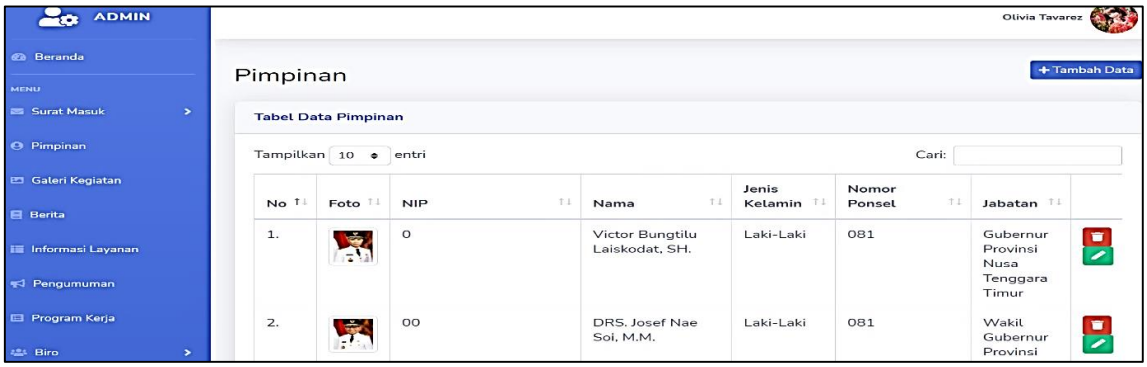
Tabel 2. Spesifikasi perangkat lunak yang digunakan

No	Perangkat Lunak	Version
1.	Microsoft Windows	10/64 bit OS
2.	MySQL Xampp	PHP 7.2.27.
3.	Aplikasi SELMA	Android & Panel Version

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

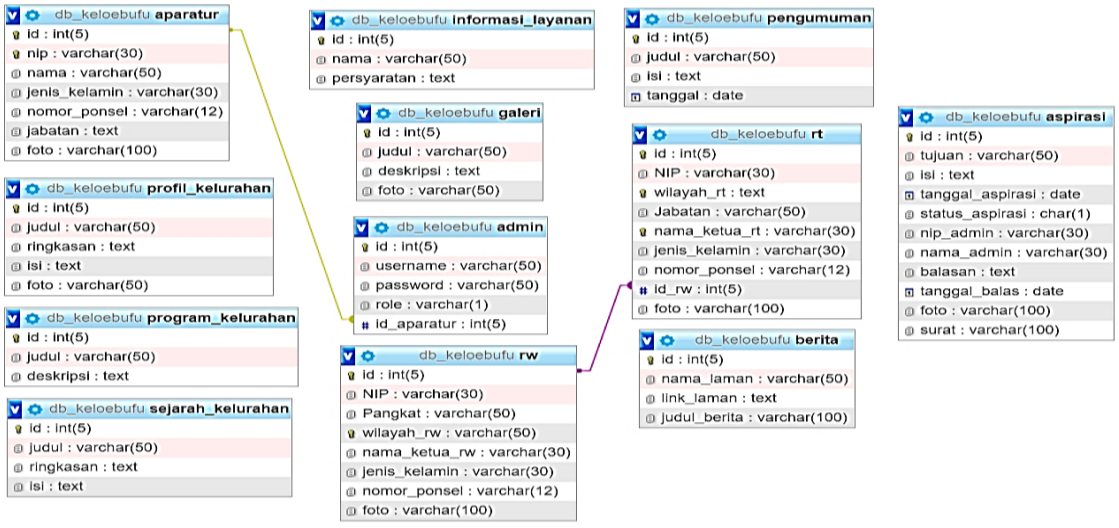
2. Menentukan *Dataset*

Penentuan *dataset* yang digunakan dalam studi kasus bersumber dari data-data yang tersimpan pada basis SELMA dalam bentuk. SQL yang di-*import* pada panel PHPMyAdmin versi 7.2.27 kemudian diujikan pada basis data MySQL. Tabel-tabel yang dilibatkan dalam pengujian adalah tabel aparatur, tabel rt serta tabel rw dengan kumpulan datanya berupa *text* dan *image*. Berikut untuk gambaran tampilan admin sistem aplikasi SELMA pada gambar 2 berikut.



Gambar 2. Aplikasi SELMA

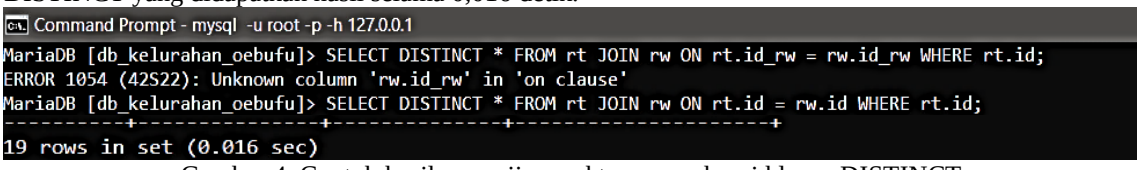
Selanjutnya hasil pengujian awal akan dicatat kecepatan waktu *respond* kueri yang dihasilkan,dan dianalisis perbandingan waktunya. Apabila kecepatan waktu respon yang didapatkan lebih singkat dibandingkan dengan model kueri awal maka model pengujian yang dilakukan optimal untuk digunakan. Adapun pengujian kueri yang dilakukan yaitu terkhusus pada optimalisasi fungsi SELECT untuk ketiga tabel uji pada sistem SELMA. Gambar 3 menampilkan keseluruhan relasi antar tabel pada sistem SELMA.



Gambar 3. Relasi tabel pada basis data aplikasi SELMA

3. Implementasi dan Pengujian Kueri

Pada tahapan ini akan ditampilkan proses implementasi optimasi untuk setiap model penggunaan klausa pelengkap fungsi SELECT pada basis data MySQL beserta dengan lama waktu respon yang ditempuh, setiap model pengujian kueri akan dilakukan sebanyak 3 kali uji kemudian untuk hasil lama waktu respon akan dicatat dan ditampilkan dalam bentuk grafik perbandingan antara kueri awal dengan kueri hasil. Gambar 4 merupakan salah satu contoh pengujian yang dilakukan dalam pengujian kueri DISTINCT yang didapatkan hasil selama 0,016 detik.



Gambar 4. Contoh hasil pengujian waktu respon kueri klausa DISTINCT

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

Konsep umum yang dilakukan untuk menguji melalui beberapa langkah diantaranya yaitu:

1. Klausa SELECT-SYMBOL OPERATOR, dioptimasi dengan mengganti penggunaan simbol operator (*) dan menyebutkan seluruh *field* yang terdapat pada tabel uji. Contohnya, kueri awal: SELECT * FROM rt; bila dioptimasikan akan menjadi: SELECT rt.nip, nama_ketua_rt, nomor_ponsel FROM rt;.
2. Klausa IN, klausa ini akan lebih tepat digunakan pada saat ingin menampilkan data dengan menetapkan satu atau lebih ketentuan kriteria data yang ingin ditampilkan. Hasil yang didapatkan akan lebih optimal dibandingan dengan penggunaan kueri umum yang memanfaatkan klausa penghubung Contohnya, kueri Awal: SELECT aparatur.* FROM aparatur WHERE aparatur.id= '1' or aparatur.id= '81 '; bila dioptimasikan akan menjadi: SELECT aparatur.* FROM aparatur WHERE aparatur.id IN (1,81);.
3. Klausa INNER JOIN-WHERE, klausa ini digunakan dalam pemanggilan data di antara beberapa tabel yang yang memiliki nilai sama dan saling berhubungan diantara yang satu dengan lainnya. Penggunaan klausa INNER JOIN dapat bekerja lebih optimal bila menambahkan klausa WHERE susunan kuerinya sehingga pencarian data menjadi jauh lebih singkat dan tidak memakan banyak waktu. Contohnya, kueri awal: SELECT * FROM rt INNER JOIN rw ON rw.id_rw = rt.id_rw; bila dioptimasikan akan menjadi: SELECT * FROM rt INNER JOIN rw ON rw.id_rw = rt.id_rw WHERE rw.id_rw = rt.id_rw;.
4. Klausa DISTINCT, dimanfaatkan dalam pemanggilan data dengan menghilangkan kemungkinan duplikasi pada data yang ingin ditampilkan. Penggunaan klausa ini kurang efektif apabila dalam tabel tidak terdapat indeks Oleh sebab itu untuk tabel yang memiliki index dalam susunannya seperti tabel rt akan lebih efektif bila menggunakan kombinasi diantara klausa JOIN, ON dan WHERE untuk memanggil data yang diinginkan. Contohnya, kueri awal: SELECT DISTINCT * FROM rt JOIN rw ON rt.id_rw = rw.id_rw WHERE rt.id; bila dioptimasikan akan menjadi: SELECT * FROM rw JOIN rt ON rw.wilayah_rw = rw.wilayah_rw WHERE rt.wilayah_rt ='44';.
5. Klausa EXISTS, lebih optimal digunakan dalam pemanggilan data pada beberapa tabel untuk menentukan keberadaan data serta mampu menjalankan pencarian pada beberapa tabel. Contohnya, kueri awal: SELECT DISTINCT* FROM rt a, rw b WHERE (b.id_rw = a.id_rw); bila dioptimasikan menggunakan EXISTS akan menjadi: SELECT * FROM rt a WHERE EXISTS (SELECT 'A' FROM rw b WHERE b.id_rw = a.id_rw);.
6. Klausa NOT IN-LEFT JOIN, klausa NOT IN digunakan untuk menampilkan atau mengevaluasi data yang tidak terdapat dalam sebuah tabel. Penggunaan klausa negatif pada SQL mampu memakan lebih banyak waktu jika dibandingkan dengan klausa positif oleh sebab itu penggunaan NOT IN dapat digantikan fungsinya dengan klausa LEFT JOIN. Contohnya, kueri awal: SELECT * FROM aparatur WHERE id NOT IN (SELECT id FROM admin); bila dioptimasikan akan menjadi: SELECT aparatur.* FROM aparatur LEFT JOIN admin ON aparatur.NIP=admin.NIP WHERE.NIP IS NULL;.
7. Klausa OR-UNION, digunakan dalam menampilkan perbandingan di antara dua data pada beberapa tabel dengan kriteria yang telah ditemtukan terlebih dahulu. Penggunaan klausa UNION jauh lebih efektif untuk digunakan dibanding OR meskipun struktur kuerinya jauh lebih panjang. Contohnya, kueri awal: SELECT nip, nama, jabatan FROM aparatur WHERE SUBSTR (nip, 1, 9) = '196' OR SUBSTR (nip, 1, 9) = '197'; bila dioptimasikan akan menjadi: SELECT nip, nama, jabatan FROM aparatur WHERE SUBSTR (nip, 1, 9) = '196' UNION SELECT nip, nama, jabatan FROM aparatur WHERE SUBSTR (nip, 1, 9) = '197';.
8. Klausa LIKE, digunakan dalam menspesifikasikan kriteria data yang ingin dipanggil atau ditampilkan. Dalam pengujian ini klausa LIKE dioptimasi dengan mengganti penggunaannya dengan operator simbol (>=). Contohnya, kueri awal: SELECT nip, nama, jabatan FROM aparatur WHERE nip LIKE '196%'; bila dioptimasikan akan menjadi: SELECT nip, nama, jabatan FROM aparatur WHERE nip >= '196';.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

3. HASIL DAN PEMBAHASAN

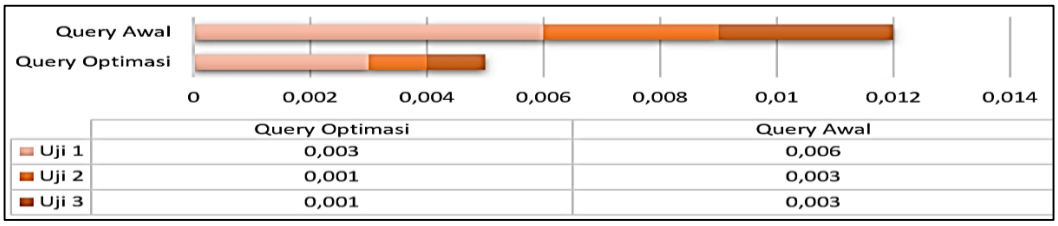
a. Optimasi Kueri SELECT Dengan Penggunaan Klausa SYMBOL OPERATOR

Penggunaan simbol operator (*) digantikan dengan menyebutkan satu per satu *field* dari tabel yang ingin ditampilkan. Pada pengujian ini ingin ditampilkan data yang terdapat dalam tabel rt.

Kueri awal: SELECT* FROM rt;

Optimasi: SELECT rt.nip, nama_ketua_rt, nomor_ponsel FROM rt;

Gambar 5 menampilkan perbandingan waktu respon kueri SELECT untuk klausa simbol operator sebanyak 3 kali pengujian dapat dilihat terdapat perbandingan besaran waktu sebesar 0,007s jauh lebih cepat dibandingkan dengan kueri awal.



	Query Optimasi	Query Awal
Uji 1	0,003	0,006
Uji 2	0,001	0,003
Uji 3	0,001	0,003

Gambar 5. Grafik perbandingan waktu respon klausa SYMBOL OPERATOR

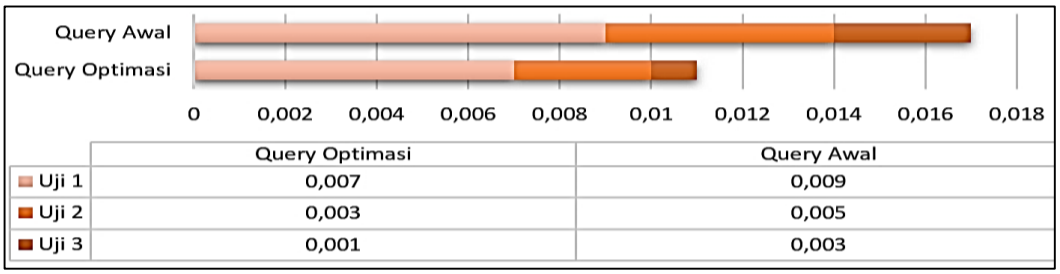
b. Optimasi Kueri SELECT Dengan Penggunaan Klausa IN

Penggunaan klausa IN jauh lebih optimal dalam menampilkan data dengan menetapkan satu atau lebih kriteria dibandingkan dengan memanfaatkan kueri umum. Kueri pengujiannya berikut:

Kueri awal: SELECT aparatur.* FROM aparatur WHERE aparatur. id= '1' or aparatur.id= '81 ';

Optimasi: SELECT aparatur.* FROM aparatur WHERE aparatur. id IN (1,81);

Gambar 6 menampilkan perbandingan waktu respon kueri SELECT untuk klausa IN sebanyak 3 kali pengujian dapat dilihat terdapat perbandingan besaran waktu sebesar 0,006s jauh lebih cepat diandingkan dengan kueri awal.



	Query Optimasi	Query Awal
Uji 1	0,007	0,009
Uji 2	0,003	0,005
Uji 3	0,001	0,003

Gambar 6. Grafik perbandingan waktu respon klausa IN

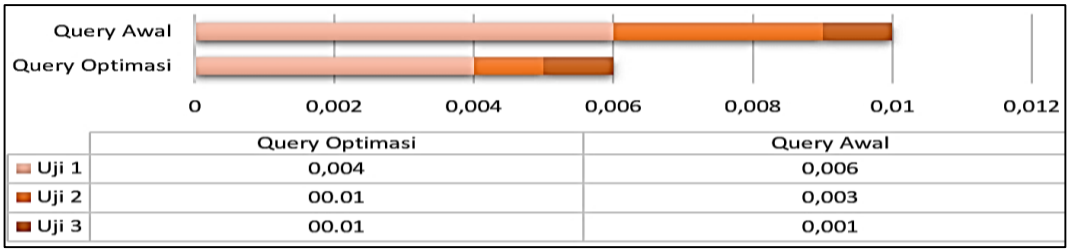
c. Optimasi Kueri SELECT Dengan Penggunaan Klausa INNER JOIN-WHERE

Penggunaan klausa INNER JOIN dapat bekerja lebih optimal bila menambahkan klausa WHERE susunan kuerinya sehingga pencarian data menjadi jauh lebih singkat dan tidak memakan banyak waktu. Pada pengujian ini ingin ditampilkan data tabel rt dan rw.

Kueri awal: SELECT* FROM rt INNER JOIN rw ON rw.id_rw = rt.id_rw;

Optimasi: SELECT* FROM rt INNER JOIN rw ON rw.id_rw = rt.id_rw WHERE rw.id_rw = rt.id_rw;

Gambar 7 menampilkan perbandingan waktu respon kueri SELECT untuk klausa INNER JOIN-WHERE sebanyak 3 kali pengujian dapat dilihat terdapat perbandingan besaran waktu sebesar 0,004s jauh lebih cepat diandingkan dengan kueri awal.



	Query Optimasi	Query Awal
Uji 1	0,004	0,006
Uji 2	00.01	0,003
Uji 3	00.01	0,001

Gambar 7. Grafik perbandingan waktu respon klausa INNER JOIN-WHERE

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

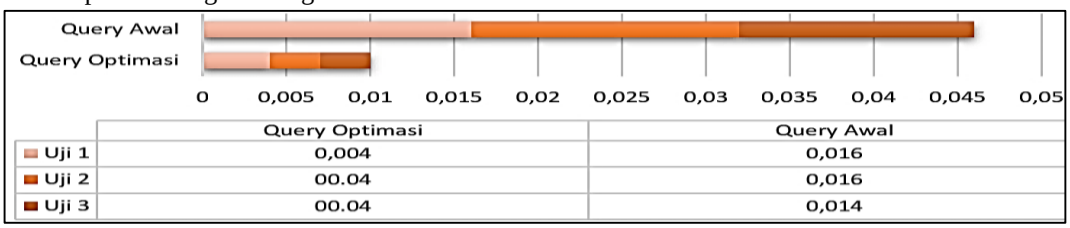
d. Optimasi Kueri SELECT Dengan Penggunaan Klausa DISTINCT

Penggunaan klausa DISTINCT dalam pemanggilan data lebih banyak memakan waktu oleh karena itu dalam pengujian ini klausa tersebut akan dihapus dan digantikan dengan klausa penghubung. Pada pengujian ini ingin ditampilkan data tabel rt dan rw.

Kueri awal: SELECT DISTINCT * FROM rt JOIN rw ON rt.id_rw = rw.id_rw WHERE rt.id;

Optimasi: SELECT * FROM rw JOIN rt ON rw.wilayah_rw = rw.wilayah_rw WHERE rt.wilayah_rt ='44';

Gambar 8 menampilkan perbandingan waktu respon kueri SELECT untuk klausa DISTINCT sebanyak 3 kali pengujian dapat dilihat terdapat perbandingan besaran waktu sebesar 0,034s jauh lebih cepat diandingkan dengan kueri awal.



Gambar 8. Grafik Perbandingan Waktu Respon Klausa DISTINCT

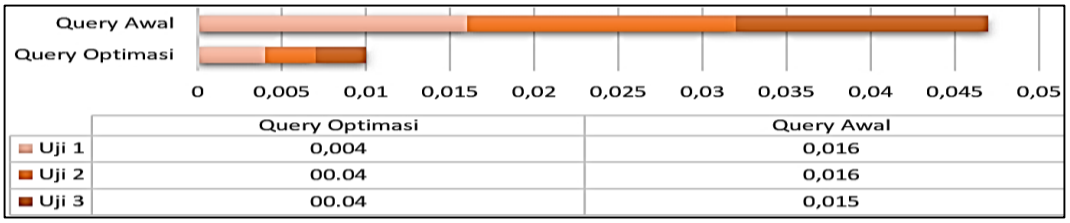
e. Optimasi Kueri SELECT Dengan Penggunaan Klausa EXISTS

Penggunaan klausa EXISTS lebih optimal untuk untuk menentukan keberadaan data serta mampu lebih cepat menjalankan pencarian pada beberapa *field* dalam tabel yang berbeda disbanding dengan klausa lain. Pada pengujian ini ingin ditampilkan data tabel rt dan rw.

Kueri Awal: SELECT DISTINCT * FROM rt a, rw b WHERE (b.id_rw = a.id_rw);

Optimasi: SELECT * FROM rt a WHERE EXISTS(SELECT 'A' FROM rw b WHERE b.id_rw = a.id_rw);

Gambar 9 menampilkan perbandingan waktu respon kueri SELECT untuk klausa EXISTS sebanyak 3 kali pengujian dapat dilihat terdapat perbandingan besaran waktu sebesar 0,035s jauh lebih cepat diandingkan dengan kueri awal.



Gambar 9. Grafik Perbandingan Waktu Respon Klausa EXISTS

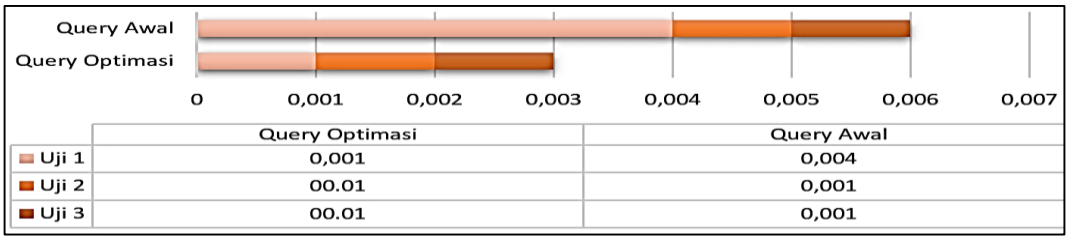
f. Optimasi Kueri SELECT Dengan Penggunaan Klausa NOT IN-LEFT JOIN

Klausa NOT IN digunakan untuk menampilkan data yang tidak terdapat dalam sebuah tabel. Penggunaan klausa *negative* pada SQL memakan lebih banyak waktu dibandingkan dengan positif oleh sebab itu NOT IN digantikan fungsinya dengan klausa LEFT JOIN.

Kueri awal: SELECT * FROM aparatur WHERE id NOT IN (SELECT id FROM admin);

Optimasi: SELECT aparatur.* FROM aparatur LEFT JOIN admin ON aparatur.NIP=admin.NIP WHERE admin.NIP IS NULL;

Gambar 10 menampilkan perbandingan waktu respon kueri SELECT untuk klausa NOT IN sebanyak 3 kali pengujian dapat dilihat terdapat perbandingan besaran waktu sebesar 0,003s jauh lebih cepat diandingkan dengan kueri awal.



Gambar 10. Grafik Perbandingan Waktu Respon Klausa NOT IN-LEFT JOIN

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

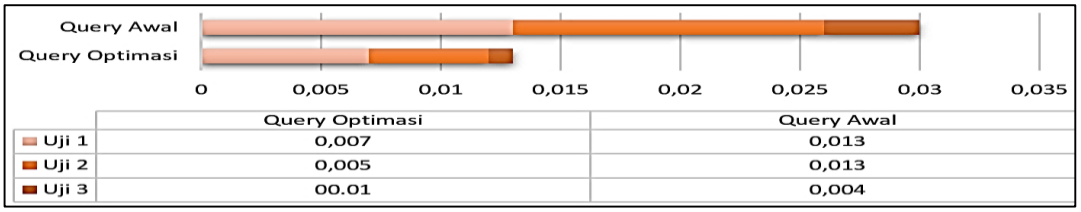
g. Optimasi Kueri SELECT Dengan Penggunaan Klausa OR-UNION

Klausa UNION digunakan dalam menampilkan perbandingan diantara dua data pada beberapa tabel dengan kriteria yang telah ditemtukan terlebih dahulu. Penggunaan klausa ini jauh lebih efektif untuk digunakan dibanding OR meskipun struktur kuerinya jauh lebih panjang.

Kueri Awal: SELECT nip, nama, jabatan FROM aparatur WHERE SUBSTR (nip, 1, 9) = '196' OR SUBSTR (nip, 1, 9) = '197';

Optimasi: SELECT nip, nama, jabatan FROM aparatur WHERE SUBSTR (nip, 1, 9) = '196' UNION SELECT nip, nama, jabatan FROM aparatur WHERE SUBSTR (nip, 1, 9) = '197';

Gambar 11 menampilkan perbandingan waktu respon kueri SELECT untuk klausa OR-UNION sebanyak 3 kali pengujian dapat dilihat terdapat perbandingan besaran waktu sebesar 0,017s jauh lebih cepat diandingkan dengan kueri awal.



Gambar 11. Grafik Perbandingan Waktu Respon Klausa OR-UNION

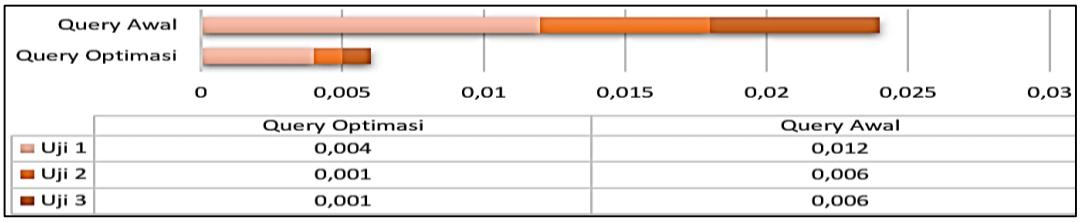
h. Optimasi Kueri SELECT Dengan Penggunaan Klausa LIKE

Klausa LIKE digunakan dalam menspesifikasikan kriteria data yang ingin dipanggil atau ditampilkan dalam pengujian ini klausa LIKE dioptimasi dengan mengganti penggunaannya dengan operator simbol (>=).

Kueri awal: SELECT nip, nama, jabatan FROM aparatur WHERE nip LIKE '196%';

Optimasi: SELECT nip, nama, jabatan FROM aparatur WHERE nip >= '196';

Gambar 12 menampilkan perbandingan waktu respon kueri SELECT untuk klausa NOT IN sebanyak 3 kali pengujian dapat dilihat terdapat perbandingan besaran waktu sebesar 0,018s jauh lebih cepat diandingkan dengan kueri awal.



Gambar 12. Grafik Perbandingan Waktu Respon Klausa LIKE

Tabel 3 menunjukkan rangkuman lama waktu respon dalam membandingkan model restrukturisasi kueri MySQL untuk ke 8 kelompok klausa yang diuji. Hasil pengujian yang tercatat dimulai dari waku respon kueri awal dan waktu respon setelah menggunakan kueri optimasi kemudian dihitung hasil selisih waktu didapat serta persentasi optimasi kueri seperti yang ditampilkan pada tabel 3.

Tabel 3. Pemetaan hasil pengujian optimasi klausa

No.	Pengujian Model Restrukturisasi Kueri	Total Keseluruhan Waktu Uji Coba (s)			Optimasi Kueri (%)
		Kueri Awal	Kueri Optimasi	Selisih	
1.	Klausa SELECT- SYMBOL OPERATOR	0,012	0,005	0,007	58%
2.	Klausa IN	0,017	0,011	0,006	35%
3.	Klausa INNER JOIN-WHERE	0,010	0,006	0,004	40%
4.	Klausa DISTINCT	0,046	0,012	0,034	73%
5.	Klausa EXISTS	0,047	0,012	0,035	74%
6.	Klausa NOT IN-LEFT JOIN	0,006	0,003	0,003	50%
7.	Klausa OR-UNION	0,030	0,013	0,017	56%
8.	Klausa LIKE	0,024	0,006	0,018	75%
		***RATA-RATA PRESENTASE OPTIMASI (%)**			78%

*Rata-rata pengurangan waktu didapat dari jumlah seluruh presentase kueri dibagi dengan 8 model pengujian klausa SQL dan dihasilkan sebesar 78% dari seluruh pengujian yang dioptimasi.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

Berdasarkan tabel 3 dapat disimpulkan secara jelas dan menyeluruh bahwa dengan dilakukannya optimasi kueri dalam sistem SELMA pada basis data MySQL mampu mengurangi waktu respon dalam proses pemanggilan data dengan menggunakan pemilihan klausa yang paling tepat. Luaran dari setiap eksekusi kueri yang dioptimasi sama dengan kueri awal namun kueri yang dioptimasi menunjukan waktu respon yang secara signifikan lebih cepat. Klausa yang umum dipergunakan seperti pada contoh model kueri awal tentunya perlu untuk dilakukan optimasi. Model kueri yang dioptimasi mampu menghasilkan lama waktu respon yang jauh lebih unggul dan efisien mulai dari penggunaan kueri SELECT dengan klausa SELECT-SYMBOL OPERATOR (SELECT*), klausa IN, klausa INNER JOIN-WHERE, klausa DISTINCT, klausa EXISTS, klausa NOT IN-LEFT JOIN, klausa OR-UNION, dan klausa LIKE dengan pengujian yang dilakukan pada 1000 *record* data mampu mengurangi lama waktu respons dengan rata-rata sebesar 78% pengurangan lama waktu respon.

4. KESIMPULAN DAN SARAN

Berdasarkan hasil pada tabel 3 yang ditampilkan, maka peneliti menguraikan kesimpulan yang dapat ditarik dalam tema mengoptimasikan kueri basis data MySQL diantaranya yaitu:

1. Penggunaan Klausa SELECT-SYMBOL OPERATOR akan lebih optimal digunakan jika disebutkan setiap *field* pada tabel yang ingin ditampilkan dibanding dengan memanggil keseluruhan isi tabel karena tidak efisien waktu dan data yang ditampilkan tidak spesifik sesuai dengan kebutuhan pengguna. Berdasarkan hasil pengujian pada klausa ini dengan disebutkan setiap field yang ingin ditampilkan berhasil mengurangi lama waktu respon sebesar 0,007s.
2. Penggunaan klausa IN mampu untuk mengoptimalkan proses pemanggilan data berdasarkan nilai *index.* Dengan memanfaatkan klausa ini maka data yang ingin ditampilkan akan diurutkan secara otomatis sesuai dengan daftar *index.*
3. Penggunaan klausa INNER JOIN-WHERE berdasarkan hasil pengujian akan jauh lebih optimal apabila kedua klausa tersebut digunakan dalam 1 struktur kueri karena mampu mengurangi waktu respon sebanyak 40% dibandingkan jika INNER JOIN digunakan sendiri.
4. Proses eksekusi penggunaan klausa DISTINCT cenderung jauh lebih lambat dilihat berdasarkan hasil pengujian sebesar 0,034s besaran selisih lama waktu yang dihasilkan, hal ini dikarenakan klausa tersebut melakukan proses pencarian atau eksekusi data dengan memilih seluruh kolom dalam setiap tabel.
5. Klausa NOT IN-LEFT JOIN mampu menampilkan data yang serupa dengan pilihan kolom dan tabel yang ditentukan. Adapun untuk eksekusi klausa NOT IN berdasarkan pengujian yang dilakukan tidak begitu optimal dan efisien karena bekerja dengan membaca seluruh data yang ada dalam tabel-tabel sebelumnya.
6. Penggunaan klausa OR-UNION digunakan untuk menampilkan pilihan data yang sebelumnya telah didefinisikan terlebih dahulu.
7. Penggunaan LIKE akan bekerja dengan *full table scanning* sehingga memakan banyak waktu.

Saran yang perlu diperhatikan oleh para pengembang maupun *database engineer* untuk setiap model pengujian yang dilakukan di antaranya sebagai berikut:

1. Dalam penggunaan klausa SELECT-SYMBOL OPERATOR (SELECT*) sebaiknya harus selalu diperhatikan apa saja data-data yang ingin dipanggil atau ditampilkan pada basis data. Meskipun dari segi struktur kueri jauh lebih pendek atau singkat serta mudah dalam penggunaanya namun akan lebih optimal dan efisien apabila pencarian lebih difilter atau dikhususkan untuk pilihan kolom data yang diinginkan.
2. Untuk menggunakan klausa IN harus diperhatikan tipe data yang digunakan adalah dalam bentuk konstanta agar kueri lebih mudah dan dapat dieksekusi secara optimal.
3. Penggunaan klausa DISTINCT sebaiknya dihindari dalam pemanfaatan fungsi SELECT dikarenakan klausa DISTINCT biasanya dipergunakan dalam operasi untuk menghapuskan kueri terduplikasi. Apabila ingin menggunakan subkueri untuk menampilkan seluruh kolom pada tabel, sebaiknya memanfaatkan penggunaan klausa EXISTS.
4. Penggunaan klausa NOT IN akan lebih efisien jika digantikan dengan memanfaatkan klausa LEFT JOIN yang membaca sasaran data sesuai dengan yang diinginkan.
5. Pada kueri ada baiknya untuk menghindari penggunaan klausa OR dikarenakan waktu eksekusi klausa ini cenderung membutuhkan waktu yang lebih lama jika dibandingkan dengan klausa UNION. Apabila memungkinkan sebaiknya manfaatkan penggunaan klausa UNION yang setara.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)

6. Penggunaan klausa LIKE dapat menjadi lebih optimal apabila diintegrasikan dengan penggunaan ekspresi atau operator tambahan untuk mendukung spesifik data yang ingin ditampilkan.
7. Untuk penelitian selanjutnya dapat memberikan model pengujian optimasi kueri dengan menambahkan lebih banyak metode optimasi, menggunakan lebih banyak contoh klausa ataupun menggunakan alat bantu *tuning SQL* untuk lebih mengakuratkan hasil pengujian optimasi yang dihasilkan.

DAFTAR PUSTAKA

[1] O.M.I Tavares., Rangkoly S.M., Sarah B. D., Utami Ema., Mustafa, "Analisis Perbandingan Performansi Waktu Respons Kueri," *Jurnal Teknologi Informasi.*, vol. 4, no. 2, hal. 303–313, 2020, [Daring]. Tersedia pada: http://jurnal.una.ac.id/index.php/jurti/article/view/1695.

[2] O.M.I. Tavares., "Analisis dan Perancangan Layanan Aspirasi Dan Informasi Pada Kelurahan Oebufu," *Jurnal Teknologi Terpadu.*, vol. 5, no. 2, hal. 303–313, 2019, [Daring]. Tersedia pada: https://journal.nurulfikri.ac.id/index.php/jtt/article/download/226/157/777.

[3] R. Gunawan, "Pengukuran Query Respon Time pada NoSQL Database Berbasis Document Stored," *Jurnal Siliwangi.*, vol. 4, no. 2, hal. 100–103, 2018, [Daring]. Tersedia pada: http://jurnal.unsil.ac.id/index.php/jssainstek/article/view/609.

[4] M. Saputro, "Pengoptimalisasian Query Pada Study Kasus Sistem Informasi Penjualan Kue Di Toko Wien," *STMIK AKAKOM Yogyakarta*, vol. 489, no. 20. eprints.akakom., Yogyakarta, hal. 313–335, 2018, [Daring]. Tersedia pada: https://eprints.akakom.ac.id/6885/.

[5] P. Noviyanti, A. Deolika, S. Hartinah, C. A. Haris, T. Maryana, dan N. D. Sari, "Perbandingan Query Response Time pada Model Query View dan Cross Product," *e-Jurnal JUSITI.*, vol. 7, no. 2, hal. 131-141, 2018, [Daring]. Tersedia pada: https://ejurnal.dipanegara.ac.id/index.php/jusiti/article/view/248.

[6] S. Suhartati dan Y. Dwi Atma, "Optimasi Query Sederhana Guna Kecepatan Query Pada Database Server," *Metik Journal.*, vol. 1, no. 1, hal. 13–17, 2017, [Daring]. Tersedia pada: http://jurnal.stmikbpn.ac.id/index.php/metik1/article/view/5.

[7] H. Siswanto, T. Andi, dan K. Kusrini, "Optimasi Query Pada Human Resource Information System (HRIS) di Universitas XYZ," *JMAI (Jurnal Multimedia)., Artif. Intell.*, vol. 2, no. 1, hal. 1–6, 2018, doi: 10.26486/jmai.v2i1.53.

[8] I. Wasiso, A. Nugroho, D. Yulianto, dan K. Kusrini, "Optimasi Basis Data Spasial Untuk Mengidentifikasi Lampu Apill Di Kabupaten Sleman," *RESEARCH: Computer, Information System & Technology Management.*, vol. 3, hal. 31, 2020, doi: 10.25273/research.v3i1.5799.

[9] R. Pamungkas, "Optimalisasi Query Dalam Basis Data My SQL Menggunakan Index," *RESEARCH : Computer, Information System & Technology Management.*, vol. 1, no. 1, hal. 27, 2018, doi: 10.25273/research.v1i1.2453.

[10] J. H. Lubis, "Analisa Performansi Query Pada Database Smell," *Jurnal Manajemen dan Informasi Pelita Nusantara.*, vol. 21, no. 1, hal. 42–49, 2017, [Daring]. Tersedia pada: http://e-jurnal.pelitanusantara.ac.id/index.php/mantik/article/view/188/105.

ISSN: 2337-7631 (printed)
ISSN: 2654-4091 (Online)