Jurnal Dinamika Informatika
Volume 8, No 2, September 2019
ISSN 1978-1660 : 79-89
ISSN *online* 2549-8517

Optimasi *Query* Sistem Informasi Menggunakan *Stored Procedure MySQL*

Tri Hastono
Jurusan Informatika, Universitas PGRI Yogyakarta
trihastono@upy.ac.id

Intisari

Query digunakan untuk operasi-operasi basisdata pada sistem informasi yang menggunakan basis data sebagai penyimpanan datanya. Pentingnya *query* seharusnya menjadi perhatian utama pada tahap perancangan aplikasi. Umumnya perancang aplikasi mengabaikan hal tersebut, sehingga aplikasi yang dibangun akan memiliki kinerja buruk ketika data dalam basis data semakin besar. Penelitian ini memberikan solusi untuk menurunnya kinerja sistem informasi dengan cara optimasi *query* sistem informasi menggunakan *stored procedure*. Basis data pada penelitian ini dirancang menggunakan MySQL Server 5 dengan tabel sebanyak 12 tabel. Jenis *stored procedure* yang dipilih pada penelitian adalah *function*. *Function* yang dibuat ada 13 *function* dengan 12 *function* untuk operasi dasar tabel dan 1 *function* digunakan untuk menangkap status proses operasi basis data. Ada 2 hal yang dibandingkan pada penelitian ini, yaitu : baris kode *query* dan waktu eksekusi *query*. Hasil penelitian menunjukkan, baris kode query *native query* lebih unggul dibandingkan query yang dioptimasi. Untuk waktu pemrosesan query, *query* yang telah dioptimasilah yang paling unggul dengan 0.16219 detik untuk tambah data, 0,24705 detik untuk edit data, dan 0,13998 detik untuk hapus data. *Query* hasil optimasi pada penelitian ini sangat membantu mempertahankan kinerja sistem informasi.

Kata kunci—Optimasi, Query, Sistem Informasi, Stored Procedure

Abstract

Queries are used for operations on information systems that use a database as data storage. The importance of queries should be a major concern at the application design stage. Generally the application designer ignores it so applications that are made will have poor performance when the data in the database is getting bigger. This study provides a solution to the declining performance of information systems by optimizing query information systems using Stored procedures. The database in this study is designed by using MySQL Server 5 with 12 tables. The type of stored procedure chosen in this study is function. There are 13 functions created with 12 functions for basic table operations and 1 function is used to capture the status of the database operation processes. There are 2 things that are compared in this research : query code lines and query execution time. The results show, native query code lines are superior to optimized queries. For query processing time, the optimized query is the most superior with 0.16219 seconds for adding data, 0.24705 seconds for editing data, and 0.13998 seconds for deleting data. query optimization results in this study are very helpful in maintaining information system performance.

Keywords—Optimization, Query, Information Systems, Stored Procedure

Jurnal Dinamika Informatika
Volume 8, No 2, September 2019
ISSN 1978-1660 : 79-89
ISSN *online* 2549-8517

PENDAHULUAN

Latar Belakang

Diera digital sekarang ini informasi adalah suatu kebutuhan yang mendasar. Hausnya kita akan informasi terlihat dari perubahan perilaku keseharian kita. Dahulu ketika kita bangun tidur umumnya cuci muka atau gosok gigi, sekarang ini memegang perangkat selular ketika kita baru bangun tidur adalah kebiasaan baru kita. Ketergantungan kita pada perangkat seluler merupakan kebiasaan baru kita yang membuktikan bahwa kita sangat haus akan informasi [1,2,3].

Ketergantungan kita akan perangkat selular adalah sesuatu yang wajar, karena dengan perangkat selular yang kita miliki, kita bias melakukan banyak hal. Kita bias melakukan transaksi jual beli barang atau mengerjakan tugas kantor melalui perangkat selular yang kita miliki. Kita bias mendapatkan informasi dari dalam negeri atau belahan dunia yang lain dari perangkat selular yang kita miliki secara *real time* [1,2].

Kemajuan teknologi sekarang ini tidak hanya membawa dampak positif, terdapat juga dampak negatif dari kemajuan teknologi. Salah satu contoh dampak negatif yang muncul dari kemajuan teknologi informasi adalah berita bohong atau *hoax*. Salah satu dampak positif yang nyata adalah kemudahan mendapatkan informasi sehingga memunculkan istilah "dunia berada digenggaman tangan kita"[4].

Informasi yang begitu banyak disekitar kita sebetulnya tersimpan dalam suatu tempat yang dinamakan basis data [4]. Basis data adalah suatu wadah untuk meletakkan tabel-tabel yang didibutuhkan pada sistem informasi. Data pada tabel sistem informasi tersebut pengolahaannya menggunakan *query*. *Query* dari sistem informasi menentukan seberapa bagus kinerja sistem informasi. Tahap perancangan query menentukan seberapa bagus *query* yang dihasilkan untuk sebuah sistem informasi [5,6].

Pada prakteknya, programmer jarang sekali memperhatikan efektifitas dan efiesiensi *query*. Pada database dengan data yang masih sedikit, belum terlihat mengenai kinerjanya sistem informasi tersebut. Tapi ketika data menjadi semakin

besar penurunan kinerja akan sangat terasa. Sistem informasi terlihat berat kerjanya, bahkan mungkin mengalami kegagalan dalam melakukan sebuah proses. Penurunan kinerja sistem informasi dapat diatasi dengan melakukan optimasi *query*.

Pada penelitian ini memberikan solusi mengenai optimasi query pada sistem informasi menggunakan *stored procedure*. Penelitian ini dilakukan pada SD Muhammadiyah Sidoarum, Gamping, Sleman, DIY. Konsep dari penelitian ini cukup sederhana yaitu semua proses operasi basisdata dilakukan query yang telah dioptimasi dibebankan pada sisi server. Adapun jenis *stores procedure* yang dipilih pada penelitian ini adalah *function*. *Function* yang dibangun sejumlah 13 *function* pada basis data yang dirancang dengan MySQL Server 5.0. 12 *function* yang dibangun berisi 3 proses, yaitu proses tambah, edit dan hapus data tabel. Sedangkan 1 *function* digunakan untuk memberikan status proses operasi basisdata. Tabel yang terdapat pada basis data sejumlah 12 tabel.

Studi Pustaka

Terdapat beberapa penelitian yang sudah pernah dilakukan oleh peneliti sebelumnya yang digunakan pada penelitian sebagai bahan acuan penelitian.

Penelitian yang diangkat oleh Suharti dan Yeyen Dwi Atma mengangkat tema mengengai trik sederhana untuk untuk optimalisasi query guna mendapatkan kinerja sistem informasi yang optimal. Sintaks *query* SQL yang akan dioptimalkan adalah sintaks SELECT *, DISTINCT, IN, INNER JOIN dalam klausa WHARE, EXISTS, DISTINCT, NOT IN dan LEFT JOIN. Hasil dari penelitian menunjukkan bahwa waktu eksekusi *query* yang telah dioptimalkan lebih sedikit dibandingkan dengan query aslinya [7].

Pada penelitian yang dilakukan oleh Yohanes Aryo Bismo Raharjo dkk membahas analisis kinerja optimalisasi query pada sintaks *cross join, natural join* dan *full outer join*. Terdapat 3 tahapan dalam optimasi *query*, yaitu tahap persiapan, tahap pengujian dan tahap analisis hasil pengujian. Tahap persiapan dimulai dari pengumpulan data dan perancangan basis data. Pada Tahap pengujian penelitian dilakukan optimasi ketiga sintaks *query*. Algoritma untuk optimasi pada penelitian adalah tahap pengujian adalah *Query Hash Join* dan *Query Nested Join*.

Jurnal Dinamika Informatika
Volume 8, No 2, September 2019
ISSN 1978-1660 : 79-89
ISSN *online* 2549-8517

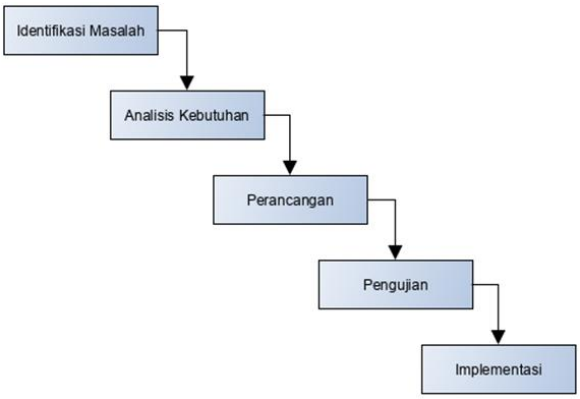
Pengujian data dilakukan pada baris data 1, 2, 4, 8,16, 32, 64, 128, 256, 512, 1024, 2048, 4096 dan 5000. Pada baris data 1, 4, 64, 128, 256, 512, 1024, 2048, 4096, dan 5000 menunjukkan *respons time* sintaks *natural join* lebih cepat dibanding sintaks *cross join* dan *full outer join*. Dan pada baris data 2, 8, 16, dan 32, sintaks *cross join* lebih unggul dibandingkan dari sintaks *natural join* dan *full outer join* [8].

Penelitian yang diangkat oleh Galih Setyo Wibowo dkk membahas mengenai optimasi *query* pada portal web lumbung Data Pendidikan Jawa Tengah. Data pada portal web Pendidikan Jawa Tengah mencapai 46.442.378 baris data dari 31 tabel. Untuk menggali informasi dari portal web tersebut tidak memungkinkan menggunakan query biasa, oleh karena itu diperlukan metode khusus untuk mengatasinya. Metode yang dipilih pada penelitian adalah algoritma Ingres. Hasil dari dari pengujian pertama mengalami penurunan kecepatan eksekusi sebesar 122%. Pada percobaan penelitian didapatkan kenaikanan kecepatan mencapai 60% [9].

Penelitian yang dilakukan oleh Melany Mustika Dewi dkk membahas mengenai perbandingan optimasi *query nasted join* dan *hash join* pada basis data yang dirancang menggunakan MySQL Server. Waktu eksekusi dari penggunaan teknik optimasi yang nantinya akan dilakukan analisa. Database pada penelitian adalah databse pegawai dengan jumlah tabel sebanyak 6 tabel. Hasil dari 3 kali pengujian didapatkan hasil optimasi *query nested join* lebih cepat dibandingkan *hash join* dengan rata-rata kecepatannya 0,172 per detik [10].

METODE PENELITIAN

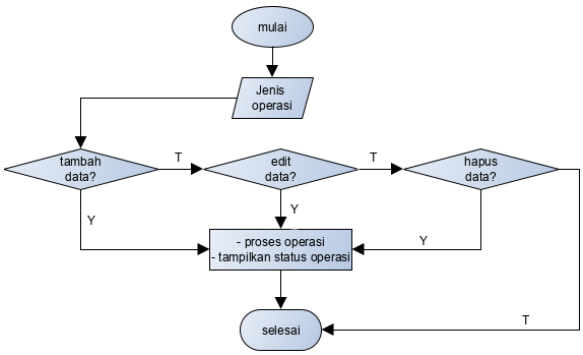
Penelitian ini berfokus pada optimasi *query* dari sistem informasi menggunakan *Stored Procedure*. Metode yang dipilih untuk melakukan pembuatan optimasi *query* sistem informasi keuangan adalah metode *Waterfall*. Terdapat beberapa tahapan pada penelitian, yaitu identifikasi masalah, analisis kebutuhan, perancangan, dan pengujian. Gambar 1 adalah gambar metode *waterfall*.



Gambar 1 Citra Metode waterfall

Tahap identifikasi masalah dilakukan dengan cara observasi dan wawancara dilokasi penelitian. Temuan yang ada adalah aplikasi pelaporan keuangan terlihat stabil. Jumlah data pada basisdata aplikasi tergolong lumyan banyak. Kondisi lain yang dihadapi adalah variasi nominal pembayaran. Hasil proses observasi dan wawancara tersebutlah yang digunakan sebagai dasar perancangan optimasi *query* guna mendapatkan kinerja yang efektif, efisien dan stabil.

Ada 2 proses pada tahap analisis kebutuhan yaitu : analisis kebutuhan pengguna dan analisis kebutuhan sistem. Hasil dari analisis kebutuhan pengguna dan kebutuhan sistem digunakan sebagai gambaran awal mengenai basisdata dari aplikasi. Dari basisdata tersebut dilakukan pengonsepan dari optimasi query agar memenuhi kebutuhan pengguna. *Flowchart* dari optimasi *query* dapat dilihat pada gambar 2.



Gambar 2 Citra Flowchart optimasi query

Jurnal Dinamika Informatika
Volume 8, No 2, September 2019
ISSN 1978-1660 : 79-89
ISSN *online* 2549-8517

Tahap pengujian adalah sebuah tahap dalam penelitian yang difungsikan untuk melakukan pengujian basisdata serta *query* yang telah dioptimasi. Pengujian dilakukan pada operasi CRUD (*Create*, *Read, Update and Delete*) dengan tabel yang ditentukan. Tahap implementasi pada penelitian berisi kombinasi basis data dan query yang telah dioptimasi pada Bahasa pemrograman yang dipilih.

HASIL DAN PEMBAHASAN

Hasil dan Pembahasan

Pada penelitian ini basisdata yang dibuat menggunakan MySQL Server 5.0. Adapun tabel yang ada dalam basis data sejumlah 12 tabel. Tabel-tabel tersebut adalah tabel Admin, Alamat, Detil_inkam, Dtl_jpos, Dtlklssiswa, Inkam, Jenis_pos, Kelas, Ortu, Outkam, pos, dan siswa. *Query* operasi *CRUD* pada tabel inilah yang yang dilakukan optimasi *query*.

Function yang dibangun pada penelitian ini berisi 3 fungsi, yaitu : tambah data, update data, dan delete data. Fungsi tersebut adalah fungsi standar pada sistem informasi sehingga function yang dibangun dipenelitian ini hampir sama untuk setiap tabel dalam basisdata. Gambar 3 dibawah ini adalah gambar salah satu function untuk fungsi dari operasi CRUD penelitian.

```
delimiter //
create function f_pos(st varchar(1),n varchar(200),ket varchar(35),id varchar(20)) returns varchar(30)
begin
    declare ada tinyint;
    set ada=(select count(*)from pos where n_pos=n and keterangan=ket);

    if st='t' then
        if ada=0 then
        begin
            insert into pos(n_pos,keterangan) values (n,ket);
            return (select f_sucer('t'));
        end;
        else return (select f_sucer('t'));
        end if;
    elseif st='e' then
    begin
        if ada=0 then
        begin
            update pos set n_pos=n,keterangan=ket where id_pos=id;
            return (select f_sucer('e'));
        end;
        end if;
        return (select f_sucer('e'));
    end;
    elseif st='h' then
    begin
        delete from pos where id_pos=id;
        return (select f_sucer('h'));
    end;
    end if;
end//
delimiter ;
```

Gambar 3 Citra *function* untuk fungsi dari operasi CRUD tabel

Dari gambar 3 diatas terlihat jelas jika *function f_pos* digunakan untuk melakukan 3 proses. Proses tersebut adalah proses tambah data, update data, dan hapus data. Ke 3 proses tersebut tergantung dari isi parameter *st* dari *function f_pos*. Jika isi dari parameter *st* adalah '*t*', maka proses yang dijalankan adalah tambah data pada tabel. Dan jika isi dari parameter *st* adalah '*e*', maka proses yang dijalankan adalah *update* data tabel. Sebaliknya jika isi dari parameter *st* adalah '*h*', maka proses yang dijalankan adalah hapus tabel.

Selain 3 proses, function *f_pos* menjalankan *function f_sucer*. *Function f_sucer* adalah *function* yang dibuat untuk menangkap status proses. Gambar 4 adalah gambar *function f_sucer*.

```
4   delimiter //
5   create function f_sucer(proses varchar(30)) returns varchar(30)
6   begin
7       if row_count()=1 then
8       return concat(row_count(),'_',proses,'_OK');
9       elseif row_count()=0 then
10      return concat(row_count(),'_',proses,'_FAIL');
11      else
12      return concat(row_count(),'_',proses,'_twin');
13      end if;
14  end//
15  delimiter ;
```

Gambar 4 Citra *Function f_sucer*

Function f_sucer yang dirancang memanfaatkan fungsi tercadang dari MySQL. Fungsi tercadang tersebut adalah fungsi *row_count()*. Setiap *query* yang dijalankan pada MySQL akan memberikan status proses. Status nilai yang dikembalikan pada setiap eksekusi *query* dalam angka (0,1,-1). Jika *query* yang dijalankan gagal, status proses bernilai 0. Dan jika *query* yang dijalankan berhasil, status proses bernilai 1. Status -1 hanya digunakan jika proses tersebut gagal dan mengandung kesalahan fatal, seperti duplikat data.

Dengan memanfaatkan fungsi tercadang *row_count()* yang dikombinasikan dengan informasi tertentu, sehingga status proses menjadi lebih informatif. Pada penelitian ini, fungsi *row_count()* dikombinasikan dengan isi dari parameter *proses* dan deskripsi kesalahan ('*_OK*','*_FAIL*','*_twin*'). Ketika *function f_sucer*

Jurnal Dinamika Informatika
Volume 8, No 2, September 2019
ISSN 1978-1660 : 79-89
ISSN *online* 2549-8517

dijalankan pada proses edit data dan ternyata berhasil maka nilai kembalian dari *function f_sucer* menjadi '*1_e_OK*', sebaliknya jika gagal maka hasilnya adalah '*0_e_FAIL*'. Dari nilai kembalian dari *function f_sucer* tersebut mempermudah pemrogram aplikasi dalam perancangan.

Pengujian

Pengujian dari *stores procedure* yang dibuat pada penelitian ini dilakukan pada 2 hal, yaitu : waktu eksekusi dan jumlah baris *query*. Untuk waktu eksekusi dipilihlah tabel alamat karena tabel alamat datanya yang paling banyak. Baris data pada tabel alamat sejumlah 82502 baris. Pada pengujian, setiap operasi CRUD dilakukan masing 5 kali proses pada setiap fungsi CRUD. Waktu eksekusi tersebut dicatat kemudian dihitung rata-ratanya. Hasil dari pengujian waktu yang diperlukan untuk eksekusi disajikan pada tabel 1.

Tabel 1 Perbandingan waktu eksekusi native query dan query yang dioptimasi

Query	Waktu Proses (detik)		
	Tambah data	Edit data	Hapus data
Native query	0,16334	0,24747	0,14077
Query yang telah dioptimasi	0,16219	0,24705	0,13998

Sedangkan untuk perbandingan baris baris *native query* dan *query* yang telah dioptimasi disajikan pada tabel 2.

Tabel 2 Perbandingan baris native query dan query yang dioptimasi

Query	Baris query		
	Tambah data	Edit data	Hapus data
Native query	1	1	1
Query yang telah dioptimasi	Diatas 2	Diatas 2	Diatas 2

KESIMPULAN

Pada penelitian ini ada 2 hal yang diuji, yaitu : waktu eksekusi *query* dan baris kode *query*. Hasil penelitian menunjukkan, baris kode query native query lebih unggul dibandingkan query yang dioptimasi. Untuk waktu pemrosesan query, query yang telah dioptimasilah yang paling unggul dengan 0.16219 detik untuk tambah data, 0,24705 detik untuk edit data, dan 0,13998 detik untuk hapus data.

Meskipun *query* yang dioptimasi berjalan bagus dari sisi waktu eksekusi, bukan berarti tidak memiliki kekurangan. Karena melakukan optimasi adalah sesuatu hal yang rumit, maka memerlukan kecermatan dan ketilitian yang tinggi serta waktu yang cukup banyak. Tetapi secara umum *query* yang telah dioptimasi pada penelitian ini sangat membantu dalam menjaga kinerja dari system informasi.

SARAN

Keberhasilan dalam melakukan optimasi *query* dipengaruhi banyak hal. Mulai dari tipe data sampai dengan normalisasi tabel. Yang tidak kalah pentingnya adalah pemilihan metode atau algoritma yang tepat sangat menentukann hasil dari optimasi.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih pihak-pihak yang secara langsung maupun tidak langsung memberikan dukungan, sehingga penelitian ini dapat terselesaikan.

Jurnal Dinamika Informatika
Volume 8, No 2, September 2019
ISSN 1978-1660 : 79-89
ISSN *online* 2549-8517

DAFTAR PUSTAKA

[1] Ngafifi, M., 2014, Kemajuan Teknologi Dan Pola Hidup Manusia Dalam Perspektif Sosial Budaya, *Jurnal Pembangunan Pendidikan: Fondasi dan Aplikasi*, vol 2, hal 33-47.

[2] Nasution, Z., 2011, Konsekuensi Sosial Media Teknologi Komunikasi Bagi Masyarakat, *Journal Simbolika*, vol 1, hal 37-41.

[3] Setiawan, D., 2018, Dampak Perkembangan Teknologi Informasi dan Komunikasi Terhadap Budaya Impact of Information Technology Development and Communication on Culture, *Journal Reformasi*, vol 1, hal 62-72.

[4] Irawan, A., Hasna, A., Pahlevi, R., 2016, "Sistem Informasi Perdagangan Pada Pt Yoltan Sari Menggunakan Php Berbasis Web", *Journal Positif*, vol 11, hal 8-15.

[5] Indrajani, I., 2018, *Database Design All in One Theory, Practice and Case Study*, Jakarta, DKI: Elex Media Komputindo.

[6] Sianipar, R. H., 2015, *Pemrograman Database Menggunakan MySQL*, Andi Publisher, Sleman, Daerah Istimewa Yogyakarta.

[7] Suhartati, S., Atma, Y. D., 2017, "Optimasi Query Sederhana Guna Kecepatan Query Pada Database Server," Metik Journal, vol. 1, no. 1, pp. 13-17.

[8] Bismo Raharjo, Y. A., Mantriwira, D., and Sumanto, F., 2018, "Analisis Kinerja Optimasi Query Cross Join,Natural JoinDan Full Outer Join," CSRID Journal, vol. 9, no. 2, pp. 106-115.

[9] Wibowo, G. S., Susanto, A., I. U. W.M., 2018, "Optimasi Query Menggunakan Algoritma Ingres Pada Portal Lumbung Data Pendidikan Jawa Tengah" Jurnal Maklumatika, vol. 5, no. 1, pp. 1-12.

[10] Dewi, M. M., & Rezeki, N. (2017). Analisis Perbandingan Optimasi Query Nasted Join dan Hash Join pada MySQL Server. CSRID Journal, 9(1), 31-42.