OPTIMASI QUERY UNTUK PENCARIAN DATA MENGGUNAKAN PENGURAIAN KALIMAT

Ardi Sanjaya
Teknik Informatika Universitas Nusantara PGRI Kediri
Jl K.H Ahmad Dahlan No.76 Mojoroto Kediri
Email : dersky@gmail.com

Abstrak

Efektifitas dalam pencarian data sangat mutlak diperlukan. Salah satu permasalahan yang dihadapi penulis dalam hal pencarian data pada database yaitu ketika mencari suatu data dengan kata kunci sebanyak 2 atau lebih (berupa kalimat) serta susunannya terbalik, maka jika kunci tersebut langsung digunakan sebagai pencarian maka cakupan pencarian akan menjadi terbatas dan tidak menemukan hasil. Penelitian ini mencoba melakukan optimasi query untuk pencarian data yaitu dengan menguraikan kalimat yang dijadikan acuan pencarian menjadi kata kunci dan di kombinasikan dengan operator OR pada syarat pencarian. Dengan penguraian kalimat dan kombinasi operator OR pada syarat pencarian menjadikan cakupan pencarian lebih luas. Berdasarkan hasil, diperoleh bahwa pencarian data menggunakan penguraian kalimat menjadi kata kunci sebagai syarat pencarian lebih bisa menemukan data yang dimaksud. Dan jika susunan kata pada kalimat yang dijadikan acuan pencarian terbalik maka masih mampu menemukan data yang dicari.

Saran untuk penelitian selanjutnya yaitu melengkapi proses pencarian dengan seleksi tingkat kemiripan dan hanya menampilkan hasil yang memiliki tingkat kemiripan yang besar sehingga diperoleh hasil yang lebih optimal.

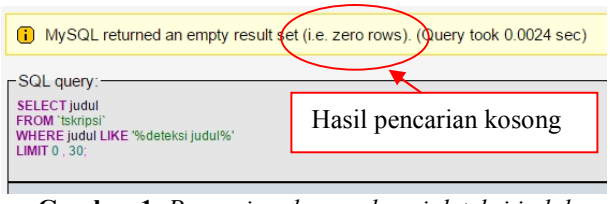
Kata kunci: *Optimasi, pencarian data, query.*

1. Pendahuluan

Pada perkembangan teknologi informasi, seperti pada aplikasi pengolahan *database* sangat harapkan hasil yang optimal terutama yang berkaitan dengan pencarian suatu data. Ada beragam cara untuk memaksimalkan pencarian data. Ganang Ardiy Tama (2015) dalam penelitiannya yang berjudul Optimasi Query Pada Sistem FAQ Di Suara Warga Universitas Negeri Semarang mengimplementasikan dan mengetahui model *query* dengan kinerja terbaik pada sistem FAQ di Suara Warga UNNES. Penelitian tersebut diujikan sebanyak 25 data Suara Warga UNNES dan 45 data dari inputan 9 responden mahasiswa UNNES pada 5 topik bahasan dengan 3 model *query*. Hasil pengujian yang dilakukan nilai *recall* 0.92, *precision* 0.41, *fmeasure* 0.56 dengan model *query* isi, nilai *recall* 0.86, *precision* 0.59, *fmeasure* 0.70 dengan model *query* isi dengan *feature selection*, nilai *recall* 0.88, *precision* 0.78, *fmeasure* 0.82 dengan model *query* judul. Dari hasil tersebut model *query* dari judul inputan memiliki nilai paling tinggi dari ke-3 pengujian tersebut [1]. Sementara Aripin (2010) dalam penelitiannya yang berjudul Meningkatkan Efektifitas Pengelolaan Database Dengan Optimasi SQL menyimpulkan bahwa perlu adanya cara untuk penulisan query yang efektif sehingga SQL dapat bekerja dengan lebih efektif dalam menghasilkan informasi diantaranya dengan *index*, menentukan tipe data, menghindari *field* bernilai null, query yang mudah terbaca, menghindari *select* *, membatasi perintah *ORDER BY*, penggunaan *WHERE* dalam *SELECT*, kecepatan akses operator, membatasi jumlah record, membatasi penggunaan *function*[2].

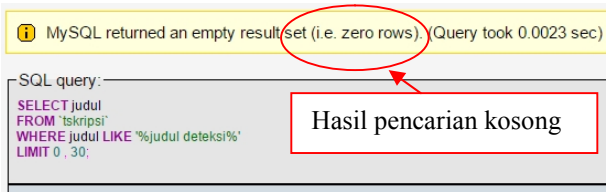
Salah satu permasalahan yang dihadapi penulis saat ini adalah terkait dengan masalah pencarian data. Ketika mencari suatu data dengan kata kunci sebanyak 2 kata atau lebih (berupa kalimat) dan kata tersebut susunannya terbalik, jika hanya menggunakan *query* biasa atau kunci pencarian tersebut langsung digunakan sebagai syarat pencarian (*where*) maka cakupan pencarian menjadi terbatas dan bisa jadi tidak akan mendapatkan hasil yang maksimal bahkan hasil pencarian kosong.

Misalnya bermaksud mencari suatu judul "Aplikasi Deteksi Kemiripan Judul" sementara kata kunci pencarian yang yang diingat adalah "Deteksi Judul" dan menggunakan *query* "select judul from tskripsi where judul LIKE '%deteksi judul%' " maka akan didapat hasil kosong. Simulasi tersebut disajikan pada gambar 1 berikut :



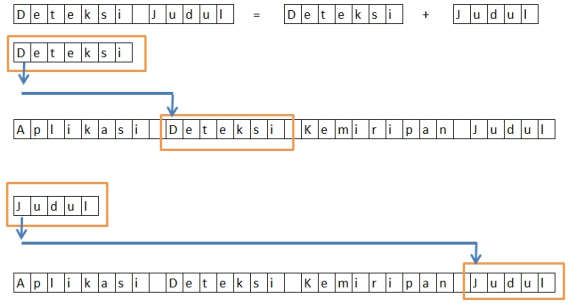
Gambar 1. *Pencarian dengan kunci deteksi judul*

Apabila kunci pencarian dibalik, susunan pencarian menjadi "select judul from tskripsi where judul LIKE '%judul deteksi%' ", ternyata masih belum mampu menemukan juga, hasilnya disajikan pada gambar 2 berikut :

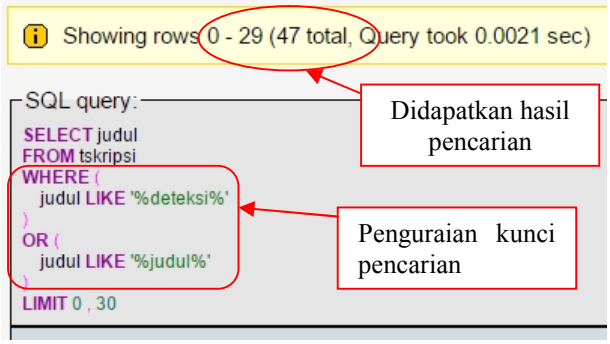


Gambar 2. *Pencarian dengan kunci judul deteksi*

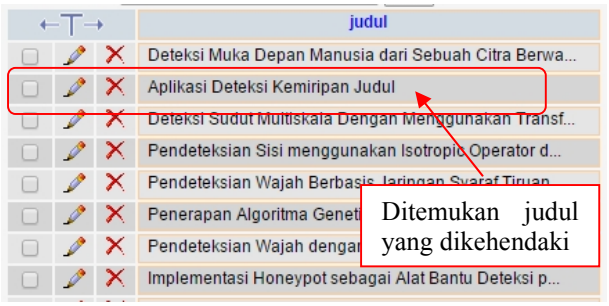
Gambaran dari penelitian ini adalah untuk mengoptimalkan pencarian data yaitu dengan cara mengurai kunci pencarian menjadi kata yang terpisah dan dijadikan acuan dengan kombinasi operator OR untuk syarat (*where*) pada *query* pencarian. Sehingga didapat *query* baru "select judul from tskripsi where (judul LIKE '%deteksi%') OR (judul LIKE '%judul%'))". Ilustrasi optimasi pencarian disajikan pada gambar 3, gambaran penggunaan *query*-nya disajikan pada gambar 4 dan hasil pencarian disajikan pada gambar 5.



Gambar 3. *Ilustrasi optimasi pencarian*



Gambar 4. *Gambaran penggunaan query*



Gambar 5. *Hasil pencarian*

1.1 Rumusan Permasalahan

Berdasarkan uraian diatas, maka dapat dirumuskan permasalahan yaitu bagaimana menghasilkan *query* untuk pencarian data dengan menggunakan kunci pencarian yang diurai susunannya sebagai syarat pencarian.

1.2 Batasan Permasalahan

Pada penelitian ini mencakup beberapa batasan yaitu :

1. Data uji berupa judul skripsi yang didapat dari hasil pencarian di *internet* sebanyak 1930 judul.
2. Pembanding pengujian menggunakan *query* bersyarat kata kunci pencarian yang tidak diurai.
3. Diimplementasikan menggunakan bahasa pemrograman PHP dan *database* MySQL.

1.3 Tujuan Penelitian

Pada penelitian ini mencoba menghasilkan *query* untuk mengoptimalkan pencarian data menggunakan penguraian kalimat menjadi bagian kata yang dijadikan syarat pencarian (*where*) dengan kombinasi operator OR. Sehingga jumlah syarat pencarian sebanyak jumlah kata dari hasil penguraian kalimat yang digunakan sebagai pencarian. Penelitian ini diimplementasikan pada pencarian judul skripsi.

1.4 Landasan Teori

1.4.1 Data

Data adalah sesuatu yang belum mempunyai arti bagi penerimanya dan masih memerlukan adanya suatu pengolahan. Data bisa berujut suatu keadaan, gambar, suara, huruf, angka, matematika, bahasa ataupun simbol-simbol lainnya yang bisa kita gunakan sebagai bahan untuk melihat lingkungan, obyek, kejadian ataupun suatu konsep[3].

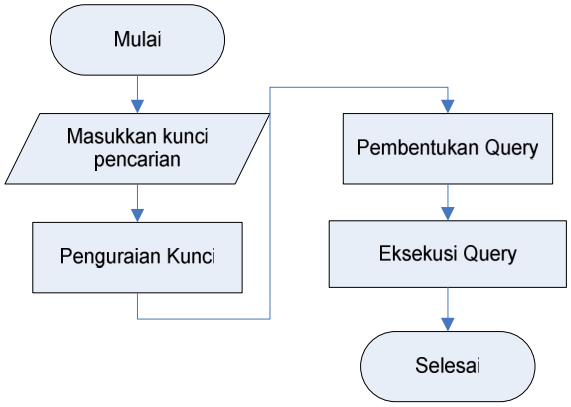
1.4.2 Query

Pengertian *query* adalah suatu kemampuan untuk menampilkan data dari *database* untuk diolah lebih lanjut yang biasanya diambil dari tabel-tabel dalam *database*. Pengertian *query* yang lain adalah pertanyaan (*question*) atau permintaan (order) informasi tertentu daru sebuah *database* yang tertulis dalam format tertentu. *Query* dapat didefinisikan sebagai perintah-perintah untuk mengakses data pada *database.*

2. Pembahasan

2.1 Alur

Alur optimasi query untuk pencarian data menggunakan penguraian kalimat dimulai dari input kunci pencarian. Kunci pencarian bisa berupa susunan 2 kata atau bisa berupa kalimat. Kemudian inputan kunci pencarian diurai menjadi kata yang terpisah. Tahap selanjutnya yaitu penyusunan *query*. Terakhir *query* yang sudah tersusun dieksekusi. Alur optimasi *query* disajikan pada gambar 2.1 berikut :



Gambar 6. *Alur proses optimasi query*

2.2 Penguraian Kalimat (kunci pencarian)

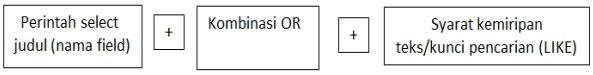
Penguraian kalimat menjadi bagian kata untuk kunci pencarian menggunakan spasi sebagai ciri untuk proses pemisahan/penguraian. Mula-mula ditetapkan variabel aw dan ak dengan nilai nol. Mulai dari *index* awal dan dengan penambahan pergeseran pergerakan sebesar 1, jika didapati spasi maka nilai variabel ak diubah menjadi nilai *index* yang sekarang. Lalu mengambil nilai *substring* dari nilai awal aw dan nilai akhir ak dan disimpan pada variabel kata dengan tipe array. Diulang seterusnya sampai mencapai index akhir dari kalimat. Berikut potongan kode program untuk mengurai kalimat :

```
$aw=0; $ak=0; $in=0;
for ($i=0;$i<=strlen($cari);$i++){
  if (substr($cari,$i,1)==' ')
  {
    $ak=$i;
    $kata[$in]=substr($cari,$aw,$ak-
$aw);
    $in=$in + 1;
    $aw=$ak;
   }
}
$kata[$in]=substr($cari,$aw+1,strlen(
$cari)-($aw+1));
```

Gambar 7. *Ilustrasi penguraian kalimat*

2.3 Penyusunan Query

Setelah penguraian kalimat, selanjutnya penyusunan *query*. Struktur *query* dapat digambarkan sebagai berikut :



Gambar 8. *Sturktur query untuk optimasi*

Perintah *select* judul digunakan untuk memilih kolom dengan nama judul yang berisi data judul skripsi dari tabel tbskripsi. Agar cakupan syarat pencarian lebih luas maka digunakan operator OR untuk menggabungkan syarat pencarian berdasarkan hasil penguraian kalimat. Adapun perintah untuk menyusun *query* adalah sebagai berikut :

```
$sql='select t.judul as judul from
tskripsi t where ';
if ($j==1) { $qr="judul LIKE
'%".$cari."%'"; $sql=$sql.$qr;}
else
{ $qr="(judul LIKE '%".$kata[0]."%')
";
for ($i=1;$i<=$in;$i++)
{ $qr=$qr." or (judul LIKE
'%".$kata[$i]."%') "; }
$sql=$sql.$qr.' group by judul';}
```

Tabel 1.*Struktur tabel tskripsi*

No	Nama Kolom	Tipe Data	Ket
1	Id	integer	Autoincrement
2	Judul	varchar	254

2.4 Pengujian dan Hasil

Dalam melakukan pengujian, penulis membuat 3 skenario uji coba dimana tiap skenario mengambil 5 contoh pencarian dengan mengambil beberapa kata acak dari judul yang hendak dicari. Berikut kombinasi acak untuk pengujian :

Tabel 2. *Pengacakan kunci pencarian*

No	Judul Yang Akan Dicari	Kombinasi acak
1	Identifikasi Personal Berdasar Bentuk Tangan	1. Identifikasi personal 2. Identifikasi tangan 3. Identifikasi bentuk tangan 4. Tangan personal 5. Berdasar personal
2	Pemrograman Assembly Pada UC 89C51 Sebagai Pengendali Robot Berjalan	1. Pengendali Robot Berjalan 2. Pemrograman Robot Berjalan 3. Assembly Robot Berjalan 4. 89C51 Assembly 5. Program Pengendali Robot
3	Simulasi Program Untuk Pengaturan Traffic Light Berbasis Fuzzy Logic	1. Pengaturan Traffic Light 2. Fuzzy Logic 3. Logic Fuzzy 4. Simulasi Traffic Light 5. Simulasi Berbasis Fuzzy Logic

Tabel 3. *Hasil pencarian skenario 1*

No	Kunci Pencarian	Hasil Pencarian (Judul)	
		Tanpa Optimasi	Dengan Optimasi
1	Identifikasi personal	2	26
2	Identifikasi tangan	0	24
3	Identifikasi bentuk tangan	0	37
4	Tangan personal	0	16
5	Berdasar personal	0	35

Tabel 4. *Hasil pencarian skenario 2*

No	Kunci Pencarian	Hasil Pencarian (Judul)	
		Tanpa Optimasi	Dengan Optimasi
1	Pengendali Robot Berjalan	1	16
2	Pemrograman Robot Berjalan	0	21
3	Assembly Robot Berjalan	0	5
4	89C51 Assembly	0	1
5	Program Pengendali Robot	0	51

Tabel 5. *Hasil pencarian skenario 3*

No	Kunci Pencarian	Hasil Pencarian (Judul)	
		Tanpa Optimasi	Dengan Optimasi
1	Pengaturan Traffic Light	1	7
2	Fuzzy Logic	10	93
3	Logic Fuzzy	0	88
4	Simulasi Traffic Light	0	57
5	Simulasi Berbasis Fuzzy Logic	0	547

Tabel 6. *Waktu pencarian skenario 1*

No	Kunci Pencarian	Waktu Pencarian (detik)	
		Tanpa Optimasi	Dengan Optimasi
1	Identifikasi personal	0.0071	0.0047
2	Identifikasi tangan	0.0026	0.0044
3	Identifikasi bentuk tangan	0.0026	0.0052
4	Tangan personal	0.0028	0.0043
5	Berdasar personal	0.0023	0.0029

Tabel 7. *Waktu pencarian skenario 2*

No	Kunci Pencarian	Waktu Pencarian (detik)	
		Tanpa Optimasi	Dengan Optimasi
1	Pengendali Robot Berjalan	0.0025	0.0059
2	Pemrograman Robot Berjalan	0.0025	0.0065
3	Assembly Robot Berjalan	0.0031	0.0069
4	89C51 Assembly	0.0021	0.0045
5	Program Pengendali Robot	0.0024	0.0041

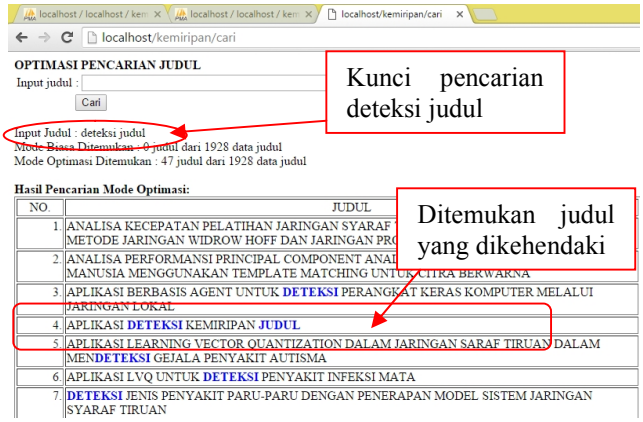
Tabel 8. *Waktu pencarian skenario 3*

No	Kunci Pencarian	Waktu Pencarian (detik)	
		Tanpa Optimasi	Dengan Optimasi
1	Pengaturan Traffic Light	0.0026	0.0067
2	Fuzzy Logic	0.0023	0.0026
3	Logic Fuzzy	0.0024	0.0031
4	Simulasi Traffic Light	0.0026	0.0035
5	Simulasi Berbasis Fuzzy Logic	0.0026	0.0006

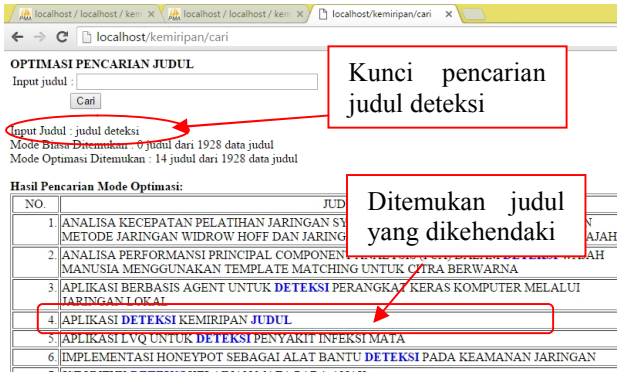
Pada skenario 1 uji coba 1 jika 2 kata kunci pencarian berurutan maka dengan *query* tanpa optimasi masih bisa mengenali/menemukan, namun pada uji coba ke 2 sampai dengan ke 5 apabila kata kunci pencarian tidak berurutan atau diacak maka dengan *query* tanpa optimasi tidak bisa menemukan data dan pada *query* dengan optimasi bisa menemukan. Pada kunci pencarian yang susunannya diacak dan menggunakan *query* yang telah dioptimasi, didapatkan hasil bahwa mampu menemukan judul yang dimaksud dan memperluas cakupan pencarian.

Demikian juga pada skenario 2 dan 3 apabila kata kunci susunannya tidak berurutan/diacak maka pada *query* tanpa optimasi sulit menemukan data dan pada *query* dengan optimasi mampu menemukan data.

Terkait dengan waktu proses eksekusi query sampai dengan menemukan hasil, berdasarkan tabel 6, 7 dan 8 dapat diketahui bahwa selisih antara eksekusi *query* tanpa optimasi dan *query* dengan optimasi tidak terlalu jauh. Dari pencarian terhadap 1930 data, rata-rata proses eksekusi masih berkisar dibawah 1 detik. Berdasarkan analisa, hal tersebut juga berpengaruh terhadap letak data di *database*. Apabila data yang dicari berada di awal, maka waktu pencarian juga bisa dipastikan akan lebih singkat daripada data yang terletak pada akhir *database*.



Gambar 9. *Hasil tampilan skenario 1 uji coba ke 1*



Gambar 10. *Hasil tampilan skenario 1 uji coba ke 2*

3. Kesimpulan

Berdasarkan uraian diatas dapat disimpulkan bahwa *query* dengan optimasi mampu menemukan data dan memperluas cakupan pencarian meskipun kata kunci pencarian letaknya tidak berurutan atau acak. Hal tersebut dikarenakan fungsi kombinasi operator OR yang dijadikan syarat pencarian berdasar kalimat yang diurai.

Pada proses pencarian *query* dengan optimasi membutuhkan waktu reltif lebih lama yaitu rata-rata selisih 0.0015 detik dari proses pencarian menggunakan *query* biasa.

Saran untuk penelitian selanjutnya adalah melengkapi proses pencarian dengan seleksi tingkat kemiripan dari kata pencarian terhadap hasil data yang ditemukan dan menampilkannya dengan batasan seperti misal menampilkan hasil pencarian 5 teratas yang memiliki kemiripan terbesar sehingga dapat diperolah hasil yang lebih optimal.

Daftar Pustaka

[1] Tama, G A, *Optimasi Model Query Pada Sistem FAQ Di Suara Warga Universitas Negeri Semarang*, http://lib.unnes.ac.id/20570/1/5302411232-S.pdf, diakses 7 Juli 2015.

[2] Aripin, *Meningkatkan Efektifitas Pengelolaan Database Dengan Optimasi Query, http://dinus.ac.id/wbsc/assets/dokumen/majalah/MENINGKATKAN_EFEKTIFITAS_PENGELOLAAN_DATABASE_DENGAN_OPTIMASI_SQL.pdf,* diakses 7 Juki 2015

[3] http://kuliah.dinus.ac.id/edi-nur/sb1-7.html, diakses 23 Oktober 2015

Biodata Penulis

Ardi Sanjaya **,** pernah bekerja di PT. Advanced Interconnect Technologies Batam tahun 2000-2006 sebagai Sealing Metal QUAD & Laser Mark Process Engineer. Memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika STT Cahaya Surya Kediri, lulus tahun 2010. Memperoleh gelar Magister Komputer (M.Kom) Program Pasca Sarjana Magister Teknik Informatika STMIK AMIKOM Yogyakarta, lulus tahun 2013. Saat ini menjadi Dosen di Universitas Nusantara PGRI Kediri.

ISSN : 2302-3805