OPTIMASI JOIN QUERY PADA BASIS DATA (STUDI KASUS : BASIS DATA SISTEM INFORMASI MANAJEMEN UNIVERSITAS COKROAMINOTO PALOPO)

Nahrun Hartono[1)],Wisnu Kurniadi[2)], Safwan Kasma[3)]

[1),3)]*Fakultas Teknik KomputerUniversitas Cokroaminoto Palopo,*
[2)] *Magister Teknik Informatika Universitas Islam Indonesia*
Email :[1]nahrunhartono@gmail.com, [2]wisnukurniadi27@gmail.com, [3] cawanksafwan@gmail.com

Abstrak

Sistem penyimpanan dan pengelolaan data dalam komputer disebut sebagai Sistem Manajemen Basis Data (SMBD) atau sering disebut dengan basis data. Basis data merupakan kumpulan data yang umumnya menggambarkan aktivitas aktivitas dan pelakunya dalam suatu organisasi. Pengolahan data pada basis data menggunakan bahas SQL (Structured English Query Language) atau biasa disebut dengan query, hasil dari pengolahan data dengan bahasa SQL kemudian ditampilkan melalui interface (antar muka) aplikasi agar lebih mudah dimengerti dan lebih informatif. Pada implementasinya query query yang menghasilkan sebuah informasi yang dibutuhkan disisipkan diantara bahasa pemrograman seperti PHP, HTML, Visual basic dan lain lain. Query SELECT merupakan query yang paling sering digunakan pada aplikasi atau sistem informasi, query SELECT biasanya diikuti dengan klausa JOIN atau biasa disebut dengan JOIN query. Optimalisasi dilakukan dengan mengubah JOIN query kedalam benuk query yang berbeda namun menghasilkan data yang sama. Query yang digunakan adalah query yang diambil dari source code Sistem Informasi Manajemen Universitas Cokroaminoto Palopo (SIMUNCP). Hasil pengujian menunjukkan perbedaan yang cukup signifikan antara query yang asli dengan query yang telah dioptimalisasi.

***Kata Kunci**: Optimalisasi, join query, basi data.*

1. Pendahuluan

Sistem penyimpanan dan pengelolaan data dalam komputer disebut sebagai Sistem Manajemen Basis Data (SMBD) atau sering disebut dengan basis data. Basis data merupakan kumpulan data yang umumnya menggambarkan aktivitasaktivitas dan pelakunya dalam suatu organisasi. Basis data berperan sebagai sumber informasi bagi sebuah sistem informasi, oleh karena itu basis data harus mampu memenuhi kebutuhan akan informasi bagi para penggunanya [1]. Tabel, kolom dan *record* merupakan elemen penting dalam basis data, ketiga elemen tersebut tidak dapat dipisahkan satu sama lain, dan tidak menutup kemungkinan dalam satu basis data terdapat banyak tabel, didalam tabel memiliki banyak kolom dan kolom-kolom tersebut berisi *record-record* (data). Pengolahan data pada basis data menggunakan bahas SQL *(Structured English Query Language)*atau biasa disebut dengan query, hasil dari pengolahan data dengan bahasa SQL kemudian ditampilkan melalui*interface*(antar muka) aplikasi agar lebih mudah dimengerti dan lebih informatif.Pada implementasinya *queryquery* yang menghasilkan sebuah informasi yang dibutuhkan disisipkan diantara bahasa pemrograman seperti PHP, HTML, Visual basic dan lain lain. Dapat disimpulkan bahwa basis data merupakan fondasi utama untuk sebuah aplikasi, karena basis data menyimpan semua data data yang dibutuhkan oleh pengguna.

Sistem informasi dipengaruhi oleh tiga hal atau biasa disebut dengan *triad factor*, yaitu : 1) Desain basis data dan implementasinya, 2) Desain aplikasi dan

implementasinya, 3) Prosedur administratif[2]. Dari tiga faktor penentu *performance* sistem informasi, basis data merupakan faktor utama yang menetukan *performace* sistem informasi. Jika berbicara tentang *performace* sistem informasi maka tidak terlepas dari kecepatan akses data, kebutuhan akan akses data yang cepat serta fleksibel merupakan hal yang sangat penting.Dalam perancangan basis data untuk mendapatkan struktur basis data yang baik maka dapat dilakukan normalisasi pada setiap tabel tabel yang ada. Normalisasi merupakan proses pengelompokan elemen data menjadi tabel yang menunjukkan entitas sekaligus relasinya [3]. Normalisasi dilakuakn untuk mencegah terjadinya *anomalyinsert*, *anomalydelete*, *anomaliupdate*. Namun perlu dikatahui bahwa semakin normal sebuah tabel pada basis data maka akan mempengaruhi kecepatan akses data yang tentu saja akan mempengaruhi *performance* sistem informasi.

James Martin (1977) mendefinisikan basis data sebagai sekumpulan data yang saling terhubung (*interrelated data*) yang disimpan secara bersama sama pada suatu media tanpa ada kerangkapan data dan kalau pun ada kerangkapan data maka kerangkapan tersebut harus seminimal mungkin dan terkontrol (*controlled redudancy*) [4]. Data disimpan dengan cara cara tertentu sehingga proses penambahan, pengambilan, modifikasi data dapat dilakukan dengan mudah dan terkontrol. Data dapat digunakan satu atau lebih program aplikasi secara optimal yang berarti data tidak memiliki ketergantungan terhadap program aplikasi yang akan menggunakannya [4]. Definisi lain tentang basis data dijelaskan oleh C.J. Date (1995), basis data sebagai beberapa kumpulan data yang akan tetap tersimpan dan akan digunakan oleh sistem sistem aplikasi yang diberikan oleh organisasi [5].

Menurut James Martin (1975) yang disarikan oleh Suwanto Raharjo, dkk (2007) dalam penelitiannya menjelaskan bahwa ada dua tujuan dari basis data yaitu tujuan primer dan tujuan sekunder [6]. Tujuan primer dari basis data meliputi penggunaan basis data oleh banyak pengguna, menjaga investasi intelektual, penekanan biaya, menghilangkan poliferasi, kinerja (*performance*), kejelasan (*clarity*), kemudahan pemakaian, fleksibilitas (*flexibility*), kebutuhan data yang tidak terantisipasi dapat dipenuhi dengan cepat, perubahan yang mudah, akurasi (*accuracy*), Serta Ketersediaan (*availability*). Sedangkan tujuan sekunder dari basis data meliputi kebebasan data secara logik (*logical data independency*), kecepatan akses, kecepatan pencarian, standarisasi data, tersedianya kamus data, antar muka program tingkat tinggi, bahasa *end user*, integritas (*integrity*), kecepatan pemulihan kembali dari kerusakan (*fast recovery from failure*), kemudahan untuk pengaturan (*tunning*), perancangan dan pengawasan alat alat, serta pengorganisasian kembali atau migrasi data dapat dilakukan secara otomatis [6].
Tabel, kolom dan *record* merupakan elemen penting dalam basis data, ketiga elemen tersebut tidak dapat dipisahkan satu sama lain, dan tidak menutup kemungkinan dalam satu basis data terdapat banyak tabel, didalam tabel memiliki bayak kolom dan kolom-kolom tersebut berisi *record-record* (data).Record (data) pada basis data diakses dengan perintah SQL, perintah atau operasi yang paling sering digunakan adalah operasi *SELECT*. Operasi *SELECT*akan menampilkan data data pada tabel yang diinginkan oleh pengguna. Operasi *SELECT* juga dapat digabungkan dengan beberapa klausa salah satunya adalah *JOIN*. Klausa *JOIN* merupakan operasi yang digunakan dalam perintah *SELECT* untuk mendapatkan data gabungan dari dua tabel atau lebih.

2. Pembahasan

Sistem Informasi Manajemen Universitas Cokroaminoto Palopo (SIMUNCP) merupakan sebuah aplikasi berbasis *web*, aplikasi tersebut menggunakan MySQL sebagai sistem manajemen basis datanya. SIMUNCP diimplementasikan pada sebuah jaringan *LocalAreaNetwork* (LAN) yang hanya bisa diakses pada lingkungan kampus Universitas Cokroaminoto Palopo. Berdasarkan hasil observasi pada aplikasi SIMUNCP diketahui bahwa terdapat 51*querySELECT* yang menggunakan klausa *JOIN*atau dapat disebut dengan *joinquery*, beberapa diantaranya menggunakan klausa *WHERE* yang diikuti

dengan variabel maka dari itu agar klausa tersebut dapat dieksekusi dan menghasilkan data maka setiap variabel yang ada pada *queryquery* tersebut perlu diganti nilainya dengan nilai yang sesuai, perubahan nilai variabel pada *query* dapat dilihat pada gambar 1.

```
1  SELECT
2    k.*, m. NAME,
3    m.KodeProgram,
4    m.KodeJurusan,
5    jur.Nama_IndONesia AS JUR
6  FROM
7    khs k
8  LEFT OUTER JOIN mhsw m ON k.NIM = m.NIM
9  LEFT OUTER JOIN jurusan jur ON m.KodeJurusan = jur.Kode
10 LEFT OUTER JOIN tahun t ON k.tahun = t.id
11 WHERE
12   k.SKS > 0
13 AND t.kode = '$thn'
14 AND t.KodeProgram = '$kdp'
15 ORDER BY
16   m.KodeJurusan,
17   m.NIM
```

Gambar 1. *Query Dengan Nilai Variabel*

Pada gambar1dapat lihat bahwa *query* tersebut memiliki nilai variabel pada baris kode 13 dan 14 yaitu variabel $thn dan variabel $kdp. jika *query* tersebut dieksekusi pada basis data tentu tidak akan menghasilkan data yang diinginkan sebab tidak terdapat data dengan nilai tersebut. Maka dari itu nilai variabel tersebut perlu diganti agar ketika dieksekusi dapat menghasilkan data, untuk variabel $thn diganti dengan nilai 20131 sedangkan untuk variabel $kdp diganti dengan nilai TI. Hal yang sama juga dilakukan pada *queryquery* lain yang memiliki nilai variabel.

Setiap *joinquery* yang dieksekusi akan menghasilkan jumlah data yang berbeda beda untuk itu pada penelitian ini dilakukan pengelompokan *joinquery* berdasarkan jumlah data yang dihasilkan yaitu kelompok satuan, kelompok puluhan, kelopok ratusan, kelompok ribuan dan kelompok puluhan ribu. Adapun jumlah *joinquery* yang dieksekusi dari masing masing pengelompokan tersebut dapat dilihat pada tabel 1.

Tabel 1. Pengelompokan join query

No	Kelompok	Jumlah Join Query
1	Satuan	19
2	Puluhan	15
3	Ratusan	6
4	Ribuan	6
5	Puluhan Ribu	5

Tabel 1 menunjukkan pengolompokkan *query* berdasarkan jumlah data yang dihasilkan, dapat dilihat bahwa jumlah *query* pada setiap kelompok berbeda. Untuk kelompok satuan terdapat 19 *query*, kelompok puluhan terdapat 15 *query*, kelompok ratusan terdapat 6 *query*, kelompok ribuan terdapat 6*query*dan kelompok puluhan ribu sebanyak 5 *query*. Setiap *queryquery* tesebut kemudian dieksekusi pada basis data baru dan diambil nilai rata rata *time result* dari lima kali. Masing masing *query* yang ada merupakan *query* yang diambil dari *sourcecode* SIMUNCP, *queryquery* tersebut kemudian dilakukan optimalisasi dengan mengubah *joinquery* tersebut kebentuk yang

berbeda adapun contoh *query* pada aplikasi SIMUNCP dapat dilihat pada gambar 1 sedangkan perubahan *query* ditunjukkan pada gambar 2.

```
SELECT
  k.*, m. NAME,
  m.KodeProgram,
  m.KodeJurusan,
  jur.Nama_IndONesia AS JUR
FROM
  khs k
LEFT OUTER JOIN mhsw m ON k.NIM = m.NIM
LEFT OUTER JOIN jurusan jur ON m.KodeJurusan = jur.Kode
LEFT OUTER JOIN tahun t ON k.tahun = t.id
WHERE
  k.SKS > 0
AND t.kode = '20131'
AND t.KodeProgram = 'TI'
ORDER BY
  m.KodeJurusan,
  m.NIM
```

Gambar 2. *Query Sebelum Perubahan*

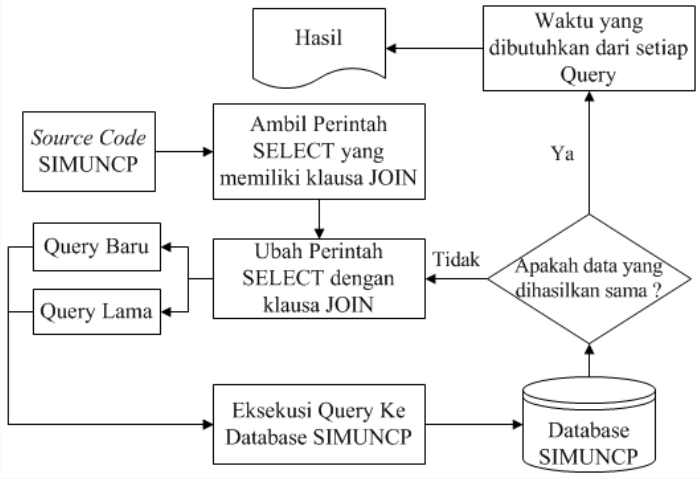
```
SELECT
  k.*, m. NAME,
  m.KodeProgram,
  m.KodeJurusan,
  jur.Nama_Indonesia AS JUR
FROM
  khs k,
  mhsw m,
  jurusan jur,
  tahun t
WHERE
  k.SKS > 0
AND t.kode = '20131'
AND t.KodeProgram = 'TI'
AND k.NIM = m.NIM
AND m.KodeJurusan = jur.Kode
AND k.tahun = t.id
ORDER BY
  m.KodeJurusan,
  m.NIM
```

Gambar 3. *Query Setelah Perubahan*

Gambar 2 menunjukkan *query* yang sebenarnya yang terdapat pada *sourcecode* SIMUNCP, sedangkan gambar3 merupakan *query* setelah perubahan. Dapat dilihat perbedaan dari kedua *query* tersebut, *query* yang ada pada *sourcecode* SIMUNCP yang ditunjukkan pada gambar 1 menggunakan perintah *SELECT* pada satu tabel kemudian dilakukan *JOIN* dengan tabel yang lain yang saling berelasi yang kemudian kolom yang berelasi dihubungkan dengan klausa *ON*. Sedangkan *query* perubahan yang ditunjukkan pada gambar 3 menggunakan perintah *SELECT* pada tabel tabel yang ingin ditampilkan datanya kemudian diikuti dengan klausa *WHERE* dan klausa *AND* untuk menghubungkan kolom kolom yang saling berelasi pada setiap tabel. Kedua *query* tersebut akan menghasilkan data yang sama jika dieksekusi.

Penelitian ini merupakan penelitian eksperimen, yang dilakukan adalah melakuakan pengujian dari setiap *query* yang ada. Data yang digunakan adalah data data yang ada pada basis data Sistem Informasi Manajemen Universitas Cokroaminoto Palopo

(SIMUNCP), pada basis data tersebut akan dilakukan pengujian dengan mengeksekusi *querySELECT* yang memiliki klausa *JOIN*. Adapun alur penelitian yang dilakukan dapat dilihat pada gambar 4.



Gambar 4. *Alur penelitian*

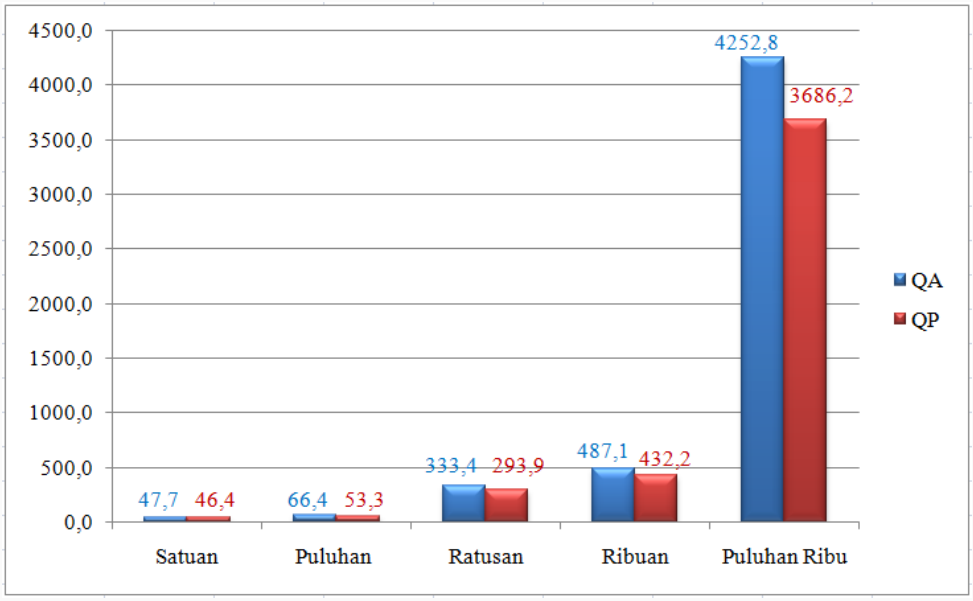
Gambar 4 menujukkan alur penelitian, yang pertama kali dilakukan adalah mengambil *querySELECT* yang memiliki klausa *JOIN*, *queryquery* tersebut diambil dari *sourcecode* SIMUNCP. Selanjutnya *queryquery* tersebut diubah dan dieksekusi pada basis data SIMUNCP, namun sebelumnya dilakukan pengecekan apakah *query* tersebut menghasilkan data yang sama dengan *query* sebelum diubah. Setiap *query* dieksekusi sebanyak lima kali dan akan dicatat waktu proses ekesekusinya kemudian diambil nilai rata rata *timeresult* dari eksekusi *query* tersebut. Agar mendapatkan hasil yang maksimal maka setealh melakukan pengujian eksekusi pada satu *query*dilakukan *restart* pada PC (*PersonalComputer*). Seperti yang disebutkan sebelumnya bahawa terdapat 51 query yang dialukan uji coba yang dikelompokkan menjadi enam kelompok query berdasarkan data yang dihasilkan. Hasil dari pengujian setiap *query* kemudian dirata ratakan berdasarkan kelompok *query* masing masing. Adapun hasil pengujian yang telah dilakukan ditunjukkan pada tabel 2.

Tabel 2. Hasil pengujian *query* berdasarkan kelompok

Kelompok Query	Total Rata Rata Time Result (ms)	
	QA	QP
Satuan	47,7	46,4
Puluhan	66,4	53,3
Ratusan	333,4	293,9
Ribuan	487,1	432,2
Puluhan Ribu	4252,8	3686,2

Tabel 2 merupakan total rata rata hasil pengujjian dari setiap kelompok *query*, pada kelompok satuan dengan jumlah *query* sebanyak 19 *query* total rata rata yang diperoleh untuk *query* asli adalah sebesar 47.7 *mili second* (ms) sedangkan total rata rata yang

diperoleh setelah dilakukan perubahan query adalah sebesar 64.4 *mili second* (ms), untuk kelompok *query*kelompok puluhan dengan jumlah *query*sebanyak15 *query* total rata rata yang diperoleh untuk *query* asli adalah sebesar 66,4 *milisecond* (ms) sedangkan total rata rata setelah dilakukan perubahan *query* adalah sebesar 53,3 *mili second* (ms), untuk kelompok *query* ratusan dengan jumlah *query* sebanyak 6 *query*total rata rata yang diperoleh untuk *query* asli adalah sebesar 333,4 *mili second* (ms) setelah dilakukan perubahan *query* dihasilkan total rata rata sebesar 293,9 *mili second* (ms), untuk kelompok *query*ribuan denagn jumlah *query* sebanyak 6 *query* total rata rata yang dihasilkan untuk *query* asli adalah sebesar 487,1 *mili second* (ms) setelah dilakukan perubahan *query* dihasikan total rata rata sebesar 432,2 *mili second* (ms), untuk *query* puluhan ribu dengan jumlah *query* sebanyak 5 *query* total rata rata yang dihasilkan untuk *query* asli adalah sebesar 4252,8 *milisecond* (ms) setelah dilakukan perubahan dihasilkan total rata rata sebesar 3686,2 *mili second* (ms). Adapun grafik hasil pengujian dari keseluruhan kelompok *query* dapat dilihat pada gambar 5.



Gambar 5. *Grafik Hasil Pengujian Query*

Pada aplikasi atau sistem informasi *querySELECT* merupakan *query* yang paling sering digunakan, karena *query* tersebut merupakan *query* untuk menampilkan data data yang ada pada kolom kolom dari setiap tabel. Penggunaan *joinquery* diperuntukaan ketika data yang ingin ditampilkan adalah data data yang terdapat pada kolom yang tersebar dibeberapa tabel. Hasil pengujian yang telah dilakukan pada 51 *query* secara grafik dapat dilihat pada gambar 5. Grafik tersebut menunjukkan perbedaan waktu eksekusi (*time result*) dari setiap kelompok*query*. Dari grafik tersebut dapat dilihat bahwa selisih tertinggi dari *timeresult* ditunjukkan pada *query* yang menghasilkan jumlah data puluhan ribu.

3. Kesimpulan

Secara keseluruhan berdasarkan apa yang telah dipaparkan sebelumnya maka dapat disimpulkan bahwa penelitan ini hanya melakukan optimalisasi pada *joinquery* yang digunakan pada aplikasi Sistem Manajemen Informasi Universitas Cokroaminoto Palopo (SIMUNCP), sedangkan untuk jenis *query* yang lain tidak dilakukan pengujian. Terdapat 51 *query* yang dilakukan uji coba, berdasarkan hasil pengujian yang telah dilakukan *timeresult* hasil optimasi lebih kecil dibanding dengan *query* sebenarnya yang terdapat pada SIMUNCP dan perbedaan *timeresult* yang signifikan ditunjukkan pada

query dengan kelompok puluhan ribu yaitu dengan selisih 566,6 *milisecond* (ms).Tidak dapat dipungkiri bahwa penelitian ini masih jauh dari sempurna hal ini merupakan keterbatasan penulis baik keterbatasan pengetahun dan keterbatasan waktu, maka dari itu penulis berharap penelitian ini dapat dijadikan acuan bagi penelitian yang akan datang dalam hal optimalisasi*query*.

DAFTAR PUSTAKA

[1] Hartono, N. (2015). Kentungan Penggunaan External Function pada Database PostgreSQL. *Jurnal CSRID, 7* (1), 39-47.
[2] Rob, P., & Coronel, C. (2009). *Database Sistem: Design, Implenetation, and Management.* Course Technology.
[3] Utami, E., & Raharjo, S. (2006). *RDBMS dengan PostgreSQL di GNU/Linux.* Yogyakarta: Andi.
[4] Martin, J. (1977). *Computer Database Organization. Part I.* New Jersey: Prentice Hall PTR.
[5] Date, C. J. (1995). *An Introduction To Database Sistem.* Addison Wesley publ.
[6] Raharjo, S., Sutanta, E., & Utami, E. (2007). Analisis aspek-aspek kualitas schema database (studi kasus pada database akademik ista yogyakarta). *Seminar Nasional Teknologi* (pp. D1 -D13). Yogyakarta: SNT Amikom.

Biodata Penulis

Nahrun Hartono, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika Universitas Cokroaminoto Palopo. Kemudian memperoleh gelar Magister Komputer (M.Kom), Teknik Informatika STMIK AMIKOMYogyakarta.

Wisnu Kurniadi, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika Universitas Cokroaminoto Palopo.

Safwan Kasma, memperoleh gelar Sarjana Komputer (S.Kom), Jurusan Teknik Informatika Universitas Cokroaminoto Palopo. Kemudian memperoleh gelar Magister Pendidikan(M.Pd), Pendidikan Teknik Komputer UNM Makassar pada tahun 2016.