Migrasi dan Optimalisasi Database Sistem Informasi Manajemen Universitas Cokroaminoto Palopo

Nahrun Hartono[1], Ema Utami[2], Armadyah Amborowati[3]
Program Pascasarjana Magister Teknik Informatika STMIK AMIKOM Yogyakarta.
Jl. Ring Road Utara Condong Catur, Sleman, Yogyakarta, Indonesia
E-mail: [1]nahrunhartono@gmail.com, [2]emma@nrar.net, [3]armadyah.a@amikom.ac.id

Masuk: 18 Februari 2016; Direvisi: 10 Maret 2016; Diterima: 10 Maret 2016

Abstract. *Information Management System of Cokroaminoto Palopo University (SIMUNCP) is a web application implemented on a Local Area Network (LAN). SIMUNCP uses MySQL as its database. The data is moved from the old database as a source to postgreeSQL as a target by migration. The migration is done because of lack of features on the old database that uses MySQL could not meet the needs of the organization. Before the migration, the first process is performed to evaluate the existing errors in the old database and the evaluation results are then used as a reference to design the new database. After the data migration is done the next process is measuring the quality of data on the new database. The quality of the data measured is an aspect of accuracy and nonduplicate aspect. Once that is done the next is to do is optimizing the query, Optimized query is a query that exists in the source code of application SIMUNCP.*
Keywords: *Migration, Database, Optimization*

Abstrak. *Sistem Informasi Manajemen Univeritas Cokroaminoto Palopo (SIMUNCP) merupakan aplikasi web yang diimplementasikan pada jaringan Local Area Network (LAN). SIMUNCP menggunakan MySQL sebagai basis datanya. Migrasi dilakukan dengan memindahkan data dari basis data lama yang menjadi sumber ke basis data postgreSQL sebagai basis data baru menjadi, hal ini dikarenakan minimnya fitur pada basis data lama yang menggunakan MySQL sehingga tidak mampu memenuhi kebutuhan organisasi. Sebelum dilakukan migrasi, yang dilakukan adalah mengevaluasi kesalahan-kesalahan yang ada pada basis data lama dan hasil evaluasi tersebut kemudian dijadikan acuan untuk merancang basis data baru. setelah migrasi data dilakukan selanjutnya adalah melakukan pengukuran kualitas data pada basis data baru, kualitas data yang diukur adalah aspek akurasi dan aspek nonduplikat, setelah itu dilakukan optimasi query, dimana query-query yang dioptimasi adalah query-query yang ada pada source code aplikasi SIMUNCP.*
Kata Kunci: *Migrasi, Basis data, Optimalisasi.*

1. Pendahuluan

Basis data sampai saat ini menjadi bagian penting bagi sistem informasi. Di era modern sekarang ini semua kebutuhan informasi manusia telah terdigitalisasi, manusia telah meminimalisir penggunaan kertas dan digantikan dengan data data digital yang disimpan pada komputer sebagai pusat data. Sistem penyimpanan dan pengelolaan data dalam komputer tersebut sering disebut sebagai basis data. Basis data merupakan kumpulan data yang umumnya menggambarkan aktivitas-aktivitas dan pelakunya dalam suatu organisasi. Basis data berperan sebagai sumber informasi bagi sebuah sisem informasi, basis data harus mampu memenuhi kebutuhan akan informasi bagi para penggunanya (Hartono, 2015). 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). Basis data dalam perancangannya memiliki aturan tersendiri dan menggunakan bahasa pemrograman tersendiri untuk mengelolanya. SQL atau disebut juga dengan SEQUEL

(*Structured English Query Language*) merupakan bahasa pemrograman yang memiliki tujuan khusus dan dirancang untuk mengelola data dalam Sistem Manajemen Basis Data (SMBD), atau untuk pengolahan aliran data dalam Sistem Manajemen Basis Data Relasional. SQL memiliki tiga bagian utama yaitu bahasa pemrograman untuk mendefinisikan data (*Data Definition Language*-DDL), untuk manipulasi dan akses data (*Data Manipulation Language*-DML) dan bagian yang digunakan untuk pengawasan/kontrol pemakai (*Data Control Language*) (Darmanto, 2015).

Terdapat banyak pilihan aplikasi SMBD, namun dalam pengembangan sistem informasi terkadang basis data tidak menjadi fokus perhatian. Seiring dengan berjalannya sistem informasi yang dikembangkan, maka sangan mungkin dilakukannya penambahan-penambahan fitur baik di level aplikasi maupun level basis data, sesuai dengan kebutuhan organisasi. Jika penambahan fitur tersebut di level basis data maka perlu dilakukan penyesuaian pada basis data tersebut. Untuk mengatasi hal tersebut menurut Li, dkk. (2009) dapat dilakukan dengan: (1) Melakukan penyesuaian antara *hardware* dan *software*, hal ini dilakukan untuk meningkatkan kinerja basis data. (2) Melakukan konfigurasi pada aplikasi basis data, ini dilakukan untuk meningkatkan *performance* dari basis data. (3) Mengoptimalkan perancangan basis data, hal ini dilakukan diawal pengembangan sistem informasi, namun yang menjadi persoalan adalah jika basis data telah dirancang, maka hal yang perlu dilakukan adalah dengan melakukan perbaikan-perbaikan pada basis data tersebut. (4) Melakukan optimalisasi pada level *query*, hal ini dilakukan untuk meningkatkan kinerja *query-query* yang diimplementasikan pada aplikasi.

Sistem Informasi Manajemen Universitas Cokroaminoto Palopo (SIMUNCP) adalah sebuah aplikasi web yang diimplementasikan pada jaringan *Local Area Network* (LAN) dan menggunakan MySQL sebagai basis datanya. Pada awal perancangan SIMUNCP, basis data tidak menjadi perhatian khusus. Salah satu hal yang menjadi masalah pada basis data SIMUNCP yang digunakan saat ini adalah persoalan penambahan data dimana terkadang *user* salah melakukan penambahan data khususnya nim mahasiswa, sementara basis data yang digunakan saat ini yaitu MySQL tidak mendukung fitur *constraint check*. Fitur *constraint check* sendiri berfungsi untuk membatasi penambahan data pada basis data, *user* akan dipaksa memasukkan data sesuai dengan format/aturan pada *constraint check* yang telah dibuat. Sebagai contoh adalah penggunaan *constraint check* untuk membatasi penambahan nim mahasiswa, dimana aturan nim mahasiswa pada Universitas Cokroaminoto Palopo adalah sepuluh karakter angka, dengan menggunakan *constraint check* maka *user* akan secara tidak langsung dipaksa mengikuti aturan tersebut, jika *user* memasukkan data yang tidak sesuai dengan *constraint check* maka data tidak akan diterima. Selain itu pembatasan penambahan data juga dapat mencegah terjadinya eksploitasi antarmuka aplikasi ke suatu basis data.

Kesalahan dalam pembuatan program antarmuka yang melakukan komunikasi ke server basis data dapat menyebabkan munculnya kerawanan keamanan terhadap data yang disimpan. Injeksi SQL melalui program antarmuka dapat dilakukan jika terjadi kesalahan pemograman. Akar penyebab dari injeksi SQL adalah ketidakcukupan validasi masukan (Utami, 2014). Maka dari itu untuk mengatasi masalah keterbatasan fitur pada basis data SIMUNCP saat ini adalah dengan melakukan migrasi data ke basis data yang menyediakan fitur-fitur yang mumpuni. Migrasi data adalah proses memindahkan data dari satu lokasi, media penyimpanan atau sistem perangkat keras/perangkat lunak ke lokasi media penyimpanan atau sistem perangkat keras/perangkat lunak yang lain (Federal Student Aid, 2007). Salah satu cara memigrasi data adalah dengan teknik *CSV file*, dengan teknik ini migrasi data dapat dilakukan pada DBMS yang berbeda. Yang menjadi persoalan dalam melakukan migrasi adalah tidak hanya sekedar memindahkan data namun melakukan restrukturisasi untuk mengatasi kesalahan struktur basis data yang sudah ada (Muslih, dkk., 2015). Basis data SIMUNCP menggunakan MySQL dengan tipe *engine* MyISAM yang berarti pada basis data tersebut tidak terdapat *foreign key*, sehingga integritas data pada basis data tersebut tidak terjamin. Maka dari itu sebelum melakukan migrasi perlu dilakukan pengidentifikasian *constraint* pada basis data tersebut.

2. Tinjauan Pustaka

Ada beberapa definisi tentang migrasi data dari beberapa referensi. Migrasi data adalah proses memindahkan data dari satu lokasi, media penyimpanan atau sistem perangkat keras/perangkat lunak kelokasi media penyimpanan atau sistem perangkat keras/perangkat lunak yang lain, tahapannya adalah perencanaan migrasi data, analisis dan perancangan migrasi data, implementasi migrasi data dan penutupan migrasi data (Federal Student Aid, 2007). Pada referensi lain disebutkan pula bahwa migrasi data adalah proses atau teknik pemindahan data yang dilakukan dengan bantuan komputer di mana sistem yang lama mengalami perubahan baik dari tipe *storage*, format data maupun sistem pengolah data sedemikian rupa sehingga data dari sistem yang lama masih dapat digunakan pada sistem yang baru (Hidayat, dkk., 2015).

Beberapa penelitian terdahulu yang telah dilakukan yang berkaitan tentang migrasi diantaranya dilakukan oleh Putra & Wibawa (2014) yang membahas tentang proses migrasi data massal dari aplikasi yang dibangun dengan *Content Management System* (CMS) Zen Cart ke Prestashop dengan mengimplementasikan metode *interpretive transformer approach* pada tahap konversi data sebagai rangkaian proses *database reengineering*. Fachrurrozi (2013) membahas tentang perbaikan fungsi dari perangkat lunak dengan melakukan pendeteksian kesalahan pada basis data lama dengan *database smell*, melakukan rekayasa ulang basis data lama berdasarkan *database smell*, mengimplementasikan basis data baru dan melakukan pengujian. Ricky (2011) membahas tentang penggunaan Oracle SQL Developer sebagai alat bantu migrasi dan Oracle Golden Gate sebagai alat batu replikasi. Andoko (2010) membahas tentang melakukan migrasi dengan menggunakan SQL *Server Integration Service*. Bafadal (2012) menjelaskan tentang penggunaan aplikasi *converter database* sebagai alat bantu untuk melakukan migrasi dari struktur *database* yang berbeda yaitu interbase *database* ke postgreSQL *database* Hidayat, dkk. (2015) membahas tentang perbandingan waktu proses dari beberapa teknik migrasi data. Dijelaskan bahwa ada beberapa teknik dalam melakukan migrasi data namun tidak semua teknik tersebut dapat diterapkan pada basis data.

Sedangkan untuk penelitian terdahulu yang berkaitan tentang optimalisasi *query* diantaranya dilakukan oleh Pasnur (2013) yang membahas tentang menerapan teknik *JavaScript Object Notation* (JSON) sebagai format pertukaran data pada saat proses *query*. Syaifudin & Hartanto (2014) membahas tentang implementasi *Hybrid Online Analitycal processing* (HOLAP) dan penggunaan algoritma genetik untuk mengoptimalkan operasi *query* pada basis data terdistribusi.

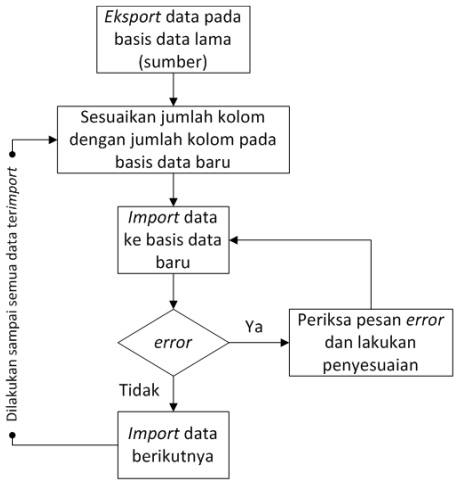
3. Metode Penelitian

Pada penelitian ini akan dilakukan perancangan basis data baru sebagai target migrasi, perancangan basis data baru ini dilakukan gana memperbaiki kesalahan yang ada pada basis data lama yang menjadi sumber migrasi, perbaikan dilakukan berdasarkan evaluasi pada basis data lama yang dilakukan sebelumnya. Seperti yang dijelaskan sebelumnya bahwa pada basis data SIMUNCP tidak terdapat *foreign key* sehingga sangat tidak mungkin untuk menggambarkan diagram alir data pada basis data tersebut. Untuk mengetahui relasi pada tabel di basis data tersebut maka hal yang perlu dilakukan adalah dengan mengidentifikasi *join query* pada *source code* aplikasi, dimana *join query* akan menunjukkan relasi pada tabel yang kemudian akan dijadikan acuan untuk merancang basis data baru target migrasi. Selain itu karena basis data SIMUNCP menggunakan MySQL dan tidak mendukung fitur *constraint check* maka sangat memungkinkan terjadinya kesalahan dalam penambahan data di basis data tersebut.

Penelitian ini adalah penelitian eksperimen, dimana setelah dilakukan migrasi penulis kemudian melakukan pengukuran kualitas data pada basis data baru, pengukuran kualitas basis data dilakukan dengan mengeksekusi *query-query* pada basis data baru. Adapun proses migrasi dari basis data lama ke basis data baru pada penelitian ini ditunjukkan pada Gambar 1.

Proses migrasi dilakukan dengan mengekstrak tabel dengan cara mengekspor setiap data yang ada pada tabel-tabel dibasis data lama ke dalam format *file* yang berekstensi *Comma Separated Value* (CSV), hasil ekstrak tersebut menghasilkan jumlah data dan jumlah kolom yang sama dengan yang ada pada tabel di basis data lama. Selanjutnya hasil ekstrak tersebut

dilakukan transformasi dengan menyesuaikan jumlah kolom pada tabel yang ada di basis data baru dengan menghapus kolom-kolom yang tidak ada dibasis data baru, hal ini dilakukan karena beberapa tabel pada basis data lama jumlah kolomnya berbeda dengan jumlah kolom yang ada pada basis data baru. Setelah itu dilakukan *load* dengan cara mengimpor data hasil transformasi tadi ke basis data baru, jika proses *load* tersebut terdapat *error* maka dilakukan penyesuaian data, jika tidak maka proses dilanjutkan ke tabel-tabel lainnya sampai semua *record* pada setiap tabel berhasil dipindahkan.

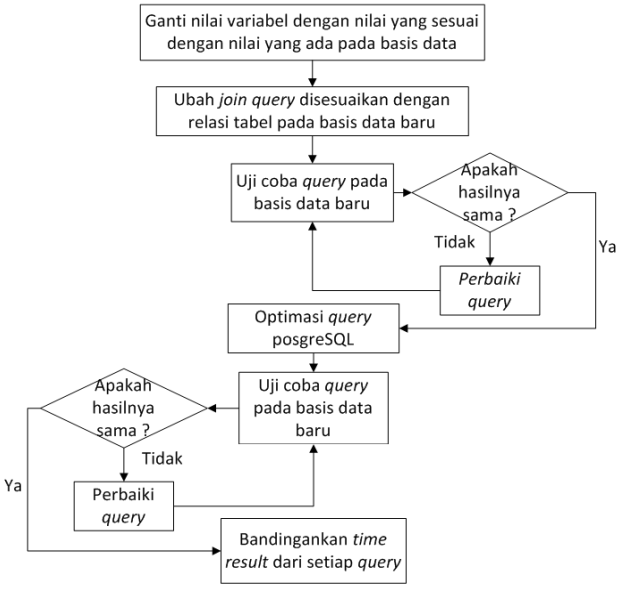


Gambar 1. Alur proses migrasi

Setelah migrasi dilakukan maka selanjutnya adalah melakukan pengukuran kualitas data. Kualitas data yang diukur adalah aspek akurasi yang berarti ukuran tingkat kecocokan data di basis data baru yang menjadi target migrasi dengan data di basis data lama yang menjadi sumber dan aspek nonduplikat yang berarti ukuran untuk memastikan bahwa terdapat hubungan *one to one* antara rekam dan objek dunia nyata atau kejadian nyata (Hegadi & Manjunath, 2013). Aspek akurasi sendiri terdapat dua bagian yang diukur yaitu akurasi baris dan akurasi kolom. Pengukuran kualitas data dilakukan dengan mengeksekusi *join query* pada basis data lama dan basis data baru yang diambil dari *source code* aplikasi SIMUNCP, hasil dari eksekusi tersebut akan menghasilkan jumlah *record* dan kolom, hasil tersebut kemudian dilakukan perbandingan antara basis data lama dengan basis data baru. Untuk aspek nonduplikat pengukurannya dilakukan dengan membandingkan hasil eksekusi *query* SELECT, yang dieksekusi pada basis data lama dan basis data baru. Setelah dilakukan pengukuran kualitas data, selanjutnya pada penelitian ini dilakukan optimalisasi *query*, dimana *query-query* yang dioptimasi adalah *join query* yang ada pada *source code* aplikasi SIMUNCP. Optimasi *query* dilakukan dengan pendekatan *heuristic* atau kadang juga disebut dengan *rule based*, teknik ini mengaplikasikan aturan *heuristic* untuk mempercepat proses *query*. Optimasi jenis ini mentransformasikan *query* dengan sejumlah aturan yang akan meningkatkan kinerja eksekusi (Darmanto, 2015). Adapun proses optimalisasi ditunjukkan pada Gambar 2.

Query-query yang dioptimasi adalah *join query* yang terdapat pada *source code* aplikasi SIMUNCP, karena terjadi perubahan struktur pada basis data baru maka *query-query* tersebut perlu dilakukan penyesuaian. Berdasarkan hasil identifikasi *join query* terdapat 51 *join query* dan pada beberapa *query* tersebut menggunakan variabel, maka untuk memastikan hasil dari setiap ekesekusi *query* tersebut perlu dilakukan mengganti nilai variabel tersebut dengan data yang sesuai pada basis data. *Query-query* tersebut akan dieksekusi pada basis data baru dan memastikan informasi yang dihasilkan dari *query* tersebut sama dengan *query* yang asli yang ada pada *source code* aplkasi. Setiap eksekusi *query* akan menghasilkan *time result*, pengujian dilakukan sebanyak lima kali pada setiap *query* dan diambil nilai rata rata *time result* dari setiap *query* tersebut. Setiap *query* yang dieksekusi menghasilkan jumlah *record* yang berbeda-beda,

maka dari itu dilakukan pengelompokkan *query* berdasarkan jumlah *record* yang dihasilkan yaitu kelompok satuan, kelompok puluhan, kelompok ratusan, kelompok ribuan dan kelompok puluhan ribu.



Gambar 2. Alur proses optimalisasi *query*

4. Hasil dan Pembahasan

Perancangan basis data baru sebagai target migrasi dilakukan berdasarkan hasil dari evaluasi kesalahan-kesalahan yang ada pada basis data lama yang menjadi sumber migrasi, setelah itu dilakukan migrasi dari basis data lama ke basis data baru. Berdasarkan hasil analisis pada basis data yang digunakan SIMUNCP saat ini, dapat diketahui bahwa terdapat 105 tabel pada basis data tersebut, *disk space* yang digunakan yaitu sebesar 57.95 *megabyte* dan tidak terdapat *foreign key* pada setiap tabel sehingga sangat sulit untuk menggambarkan *Entity Relationship Diagram* (ERD) pada basis data tersebut.

Seperti yang dejelaskan sebelumnya bahwa MySQL tidak mendukung fitur *constraint check* untuk membatasi penambahan data pada kolom, sehingga *user* bisa saja menambahkan data yang tidak sesuai dengan ketentuan. Kode 1 menunjukkan contoh penambahan data pada tabel mhsw di basis data lama, dimana pada kolom nim dimasukkan nilai yang tidak sesuai dengan aturan nim pada Universitas Cokroaminoto Palopo yaitu sepuluh karakter angka, dimana nim yang dimasukkan adalah nim dengan karakter abjad/huruf, hasilnya adalah data tersebut dapat ditambahkan. Pada basis data baru kesalahan penambahan tersebut dapat diminimalisir dengan menambahkan *check constraint*, seperti ditunjukkan pada Kode 2 dimana jika nim yang dimasukkan adalah karakter abjad/huruf maka data akan ditolak dan data tidak dimasukkan kedalam tabel karena nilai yang dimasukkan tidak sesuai dengan aturan *constraint check* yang telah ditentukan.

```
mysql> INSERT INTO mhsw VALUES ('100004', '', '', '', 'NIMABCDEFG', '', '', '', '', '',
 '', 'NAHRUN HARTONO', 'nahrunhartono@gmail.com', 'L', 'Makassar', 'now()', '', '', '',
 '', '', '90234', '', '085932423670', '', 3, '', '', 'Indonesia', 'N', 'N', 'N', '', ''
, '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', 0.00,
 '', '', '', 'TI', 'A', 'REG', 31, '', '', 0, '', '20131', '', '', 'N', '', '', '', '',
 '', '', '', '', 13, '', '', '', '19', '', 'N', 0, 'N', '', 'now()', 'now()', 30, 3.00,
 '', 11, '', 'now()', '', '', '', 'N', '', '', '', '', '', '', '3.00', '', '', '', '',
'3.00', '', '', '', '', '3.00', '', '', 'now()', '', '', '', '', '', '', '', '', '', ''
, '', '', '', '', 'now()', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', 'now
()', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''
);
Query OK, 1 row affected, 18 warnings (0.00 sec)
```

Kode 1. *Query insert* pada basis data lama dengan nim salah

```
simuncppsql=# INSERT INTO mhsw VALUES ('100004','','','','NIMABCDEFG','','','','',now()
,'NAHRUN HARTONO','nahrunhartono@gmail.com','L','Makassar',now(),'','','','','90234'
,'085932423670','3','','','Indonesia','N','N','N','','','','TI','A','REG','31','20131',
'N','','13',19,'N','N',now(),now(),now(),'30','3.00','','11','','','N','','','','','3.0
0','','','','','3.00','','','','','3.00','','',now(),'','','','','','','','','','','','
','','','',now(),'','','','','','','','','','','','','','','',now(),'','','','','','','','','','
','','','',now(),'','0','B','','','');
ERROR:  new row for relation "mhsw" violates check constraint "check_nim"
```

Kode 2. *Query insert* pada basis data baru dengan nim salah

Hasil evaluasi selanjutnya adalah diketahui bahwa basis data lama tidak terdapat *foreign key* sehingga sangat memungkinkan terjadinya kesalahan dalam integritas referensial dan tidak terdapat *check constraint* sehingga kesalahan dalam penambahan juga sangat mungkin terjadi, basis data baru yang dirancang sebagai target migrasi dapat meminimalisir kesalahan-kesalahan tersebut. Sebagai contoh, diketahui bahwa kolom kodebiaya pada tabel biaya2 mereferensi kolom kode pada tabel biaya, yang berarti jika dilakukan penambahan data pada pada tabel biaya2 nilai data pada kolom kodebiaya harus sama dengan nilai data pada kolom kode ditabel biaya, namun karena pada basis data lama tidak memiliki *foreign key* sehingga data referensi pada tabel yang berelasi bisa berbeda seperti ditunjukkan pada Kode 3. Kesalahan tersebut merupakan kesalahan itegritas referensial yang disebabkan tidak adanya *foreign key* pada tabel yang berelasi, maka dari itu pada basis data baru ditambahkan *constraint foreign key* pada kolom kodebiaya ditabel biaya2 dan mereferensi kolom kode ditabel biaya sehingga jika dilakukan penambahan data pada tabel biaya2 nilai pada kolom kodebiaya harus sama dengan nilai pada kolom kode pada tabel biaya, seperti yang ditunjukkan pada Kode 4.

```
mysql> INSERT INTO biaya2
    -> (id, kodebiaya, kodejurusan, kodeprogram, nama, kali, jenisbiaya,
    -> denda, spp, otomatis, status, statusawal, statuspotongan, currency,
    -> jumlah, pakaiscript, namascript, notactive)
    -> VALUES ('200', 'KODE', 'TI', 'REG', 'KKN', '1', '25', 'N',
    -> '1', 'Y', 'A', 'B', 'B1', 'Rp.', '125000', 'N', '', 'N');
Query OK, 1 row affected (0.00 sec)
```

Kode 3. *Query insert* pada basis data lama

```
simuncppsql=# INSERT INTO Biaya2
simuncppsql-# (id, kodebiaya, kodejurusan, nama, kali,
simuncppsql(# jenisbiaya, denda, spp, otomatis, status, statusawal,
simuncppsql(# statuspotongan,  jumlah, pakaiscript, namascript, notactive)
simuncppsql-# VALUES ('200','KODE','TI','KKN','1','25','N','-1','Y','A',
simuncppsql(# 'B','B1','125000', 'N','','N');
ERROR:  insert or update on table "biaya2" violates foreign key constraint "biaya2_kodebiaya_fkey"
DETAIL:  Key (kodebiaya)=(KODE) is not present in table "biaya".
simuncppsql=#
```

Kode 4. *Query insert* pada basis data baru

Hasil dari evaluasi kesalahan-kesalahan pada basis data lama kemudian menjadi acuan untuk merancang basis data baru sebagai target migrasi. Adapun perbedaan struktur tabel pada basis data baru dengan basis data lama secara umum ditunjukkan pada Tabel 1.

Tabel 1. Perbedaan umum struktur basis data lama dengan basis data baru

No	Perbedaan	Basis Data Baru	Basis Data Lama	Keterangan
1	Penggunaan *Primary key*	Ya	Ya	14 tabel pada basis data lama tidak memiliki primary key
2	Penggunaan *Foreign key*	Ya	Tidak	
3	Penggunaan *Constraint check*	Ya	Tidak	
4	Penggunaan *Auto Increment*	Ya	Ya	
5	Jumlah Tabel	94	105	Beberapa tabel tidak digunakan/tidak dirancangan pada basis data baru, karena tidak disebutkan di sourc code aplikasi
6	Jumlah Kolom	1132	2104	Beberapa tabel tidak digunakan/tidak dirancangan pada basis data baru, karena tidak disebutkan di source code aplikasi dan memiliki informasi yang sama pada tabel yang saling berelasi

Basis data baru sebagai target migrasi dirancang dengan menggunakan postgreSQL, hal ini dikarenakan postgreSQL merupakn aplikasi basis data yang *open source* dan memiliki fitur yang dibutuhkan. Selanjutnya adalah mengukur kualitas basis data pada basis data baru. Hasil pengukuran untuk aspek nonduplikat dilakukan dengan membandingkan hasil dari basis data lama dengan basis data baru, sedangkan untuk aspek akurasi hanya dilakukan pada basis data baru. Hal ini disebabkan karena tidak adanya pembanding akurasi pada basis data lama, sedangkan pada basis data baru dapat diukur tingkat akurasinya dengan menjadikan basis data lama sebagai pembanding. *Query* pengujian aspek akurasi akan berbeda antara basis data lama dengan basis data baru, hal ini disebabkan karena adanya perubahan struktur pada basis data yang baru, namun perubahan *query* tersebut tidak akan mempengaruhi informasi yang dihasilkan. Kode 5 menujukkan salah satu contoh *query* yang akan dieksekusi pada basis data lama dan Kode 6 merupakan perubahan dari *query* tersebut yang akan dieksekusi pada basis data baru.

```
SELECT  j.Tunda AS tndj,j.AlasanTunda AS alsj,k.KodeMK AS Kode,k.NamaMK AS
        MK,k.SKS,j.KodeFakultas,k.Bobot,CONCAT(ds.Name, ', ', ds.Gelar) AS
        DS,k.Nilai,k.GradeNilai,k.Tunda AS tndk,k.AlasanTunda AS alsk,
        k.Setara,k.NotActive
FROM krs k
        LEFT OUTER JOIN jadwal j ON k.IDJadwal=j.ID LEFT OUTER JOIN dosen ds
        ON k.IDDosen=ds.ID
ORDER BY k.KodeMK
```

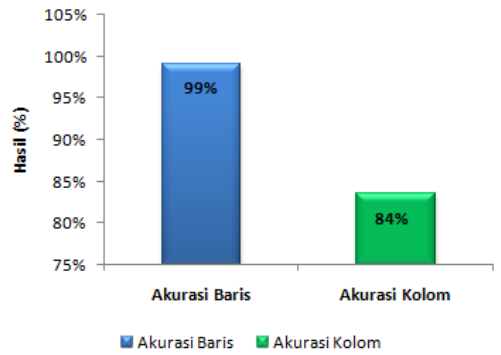
Kode 5. *Query* pengujian kualitas data aspek akurasi pada basis data lama

```
SELECT  j.Tunda AS tndj,j.AlasanTunda AS alsj,mk.Kode AS Kode, mk.Nama_indonesia AS
        MK,mk.SKS,jur.kodefakultas,n.Bobot,CONCAT(ds.Name, ', ', ds.Gelar) AS
        DS,k.Nilai,k.GradeNilai,k.Tunda AS tndk,k.AlasanTunda AS alsk,k.Setara,
        k.NotActive
FROM krs k
        LEFT OUTER JOIN matakuliah mk ON k.idmk=mk.id
        LEFT OUTER JOIN nilai n ON k.gradenilai=n.id
        LEFT OUTER JOIN jadwal j ON k.IDJadwal=j.ID
        LEFT OUTER JOIN jurusan jur ON jur.kode=j.kodejurusan
        LEFT OUTER JOIN dosen ds ON k.IDDosen=ds.ID
ORDER BY mk.Kode
```

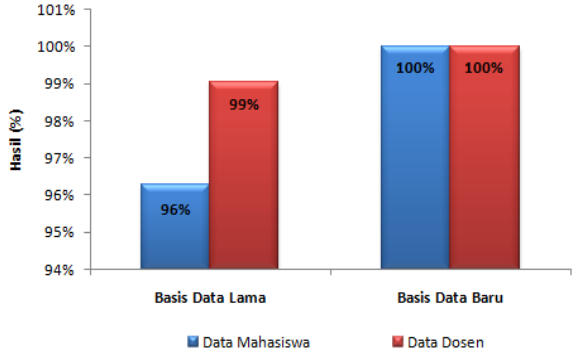
Kode 6. *Query* pengujian kualitas data aspek akurasi pada basis data lama

Pengukuran kualitas data aspek akurasi dilakukan dengan mengeksekusi *join query*, dimana terdapat 51 *query* yang dieksekusi pada basis data lama dan basis data baru. Hasil pengujian kualitas data aspek akurasi baris dan akurasi kolom ditunjukkan pada Gambar 3. Pada Gambar 3 dapat dilihat bahwa hasil pengukuran kualitas data untuk aspek akurasi baris senilai 0,99 atau dengan nilai persentase 99%, hasil pengukuran untuk akurasi baris tidak mencapai 100% disebabkan karena pada saat proses migrasi data yang dilakukan terdapat *error* yang mengharuskan dilakukannya perubahan data dan untuk akurasi kolom dihasilkan nilai 0,84 atau dengan nilai persentase 84%, hasil pengukuran tidak mencapai 100% pada akurasi kolom disebabkan karena beberapa kolom pada basis data sumber tidak digunakan berdasarkan hasil identifikasi *query* dan juga terdepat beberapa kolom yang menghasilkan informasi yang sama pada tabel yang berelasi, sehingga kolom-kolom tersebut pada basis data baru tidak dirancang.

Pengukuran kualitas data aspek nonduplikat dilakukan pada data mahasiswa yang terdapat pada tabel mhsw dan data dosen yang terdapat pada tabel dosen, yang seharusnya data mahasiswa dan data dosen tidak terjadi duplikasi. Untuk mengukur aspek nonduplikat pada data tersebut dilakukan dengan mengeksekusi *query*, adapun *query* yang digunakan untuk pengujian kualitas data aspek nonduplikat yaitu dengan mengeksekusi *query* SELECT tanpa klausa GROUP BY dan *query* SELECT dengan klausa GROUP BY. Hasil pengukuran kualitas data aspek nonduplikat ditunjukkan pada Gambar 4.



Gambar 3. Hasil pengukuran kualitas data aspek akurasi



Gambar 4. Hasil pengujian kualitas data aspek nonduplikat

Hasil pengujian kualitas data pada basis data target untuk aspek nonduplikat dihasilkan sangat baik yaitu dengan nilai persentase 100% seperti yang ditunjukkan pada Gambar 4. Hal ini berarti setelah migrasi data dilakukan tidak ada lagi data mahasiswa dan data dosen yang terduplikasi. Setelah hasil kualitas data pada basis data baru diketahui, selanjutnya adalah melakukan optimalisasi *query*. Dimana *query-query* yang dioptimasi adalah *query join* yang terdapat pada *source code* aplikasi. Kode 7 merupakan salah satu contoh *query* yang tidak teroptimasi dan Kode 8 merupakan *query* yang telah dioptimasi.

```
SELECT m.NIM, m.Name, b1.Currency,
       b1.Kurs, b1.Jumlah, bm.Catatan
FROM bayar2 b2
       LEFT OUTER JOIN bayar b1 on b2.BayarID=b1.ID
       LEFT OUTER JOIN mhsw m ON b1.NIM=m.NIM
       LEFT OUTER JOIN biayamhsw bm ON b2.BiayaID=bm.ID
       LEFT OUTER JOIN biaya2 bia2 ON bm.idbiaya2=bia2.id
WHERE bia2.Nama='$_SESSION[NamaBiaya]'
ORDER BY m.NIM
```

Kode 7. *Query* yang tidak teroptimasi

```
SELECT m.NIM, m.Name, b1.Currency,
       b1.Kurs, b1.Jumlah, bm.Catatan
FROM bayar2 b2, bayar b1, mhsw m, biayamhsw bm, biaya2 bia2
WHERE bia2.Nama='$_SESSION[NamaBiaya]' AND b2.BayarID=b1.ID AND b1.NIM=m.NIM AND
b2.BiayaID=bm.ID AND bm.idbiaya2=bia2.id
ORDER BY m.NIM
```

Kode 8. *Query* yang teroptimasi

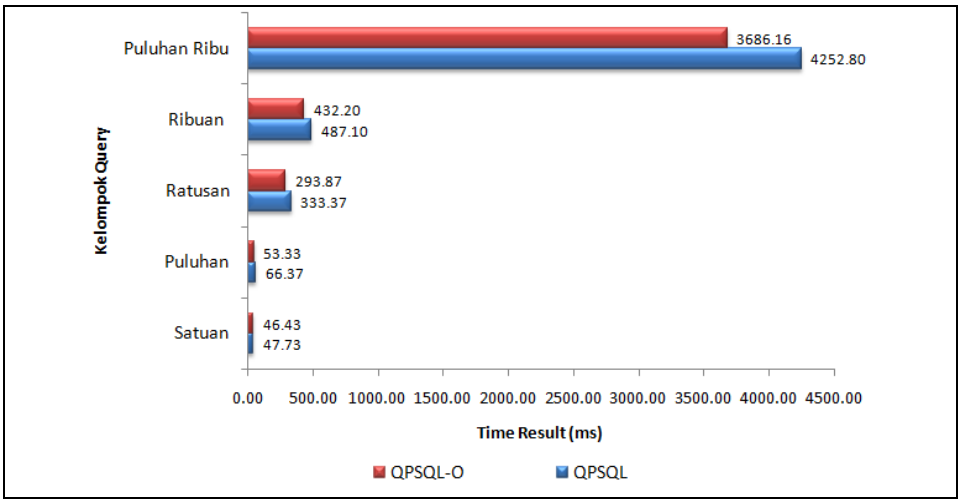
Untuk pengujian optimalisasi *query* terdapat 51 *query join*, dimana *query-query* tersebut akan dieksekusi pada basis data baru yang telah dirancang sebelumnya, setiap hasil eksekusi akan menghasilkan *time result* dalam satuan *mili second* (ms), kemudian dilakukan perbandingan antara *query* yang belum dioptimasi dengan *query* telah dioptimasi. Adapun hasil pengujian sebelum optimasi dan sesudah optimasi ditunjukkan pada Tabel 2.

Tabel 2. Hasil pengujian optimasi *query*

Query	Kelompok Berdasarkan Jumlah Data yang dihasilkan				
	Satuan	Puluhan	Ratusan	Ribuan	Puluhan Ribu
QPSQL	47.73	66.37	333.37	487.10	4252.80
QPSQL-O	46.43	53.33	293.87	432.20	3686.16

Tabel 2 merupakan hasil pengujian optimasi *query* dalam satuan *mili second* (ms), dimana terdapat lima kelompok data berdasarkan jumlah data yang dihasilkan pada setiap *query*. QPSQL merupakan *query* yang dieksekusi pada basis data baru yang belum teroptimasi dan QPSQL-O merupakan *query* yang dieksekusi pada basis data baru yang telah dioptimasi. Dari hasil pengujian dihasilkan *query* yang telah dioptimasi lebih baik dibandingkan *query*

sebelum optimasi, dimana untuk *query* kelompok satuan dihasilkan 47,73 ms sebelum optimasi dan setelah dioptimasi dihasilkan 46,43 ms, untuk *query* kelompok puluhan dihasilkan 66,37 ms sebelum optimasi dan setelah dilakukan optimasi dihasilkan 53,33 ms, untuk *query* kelompok ratusan dihasilkan 333,37 ms sebelum dioptimasi dan setelah dioptimasi 293,87 ms, untuk *query* kelompok ribuan dihasilkan 487,10 ms dan setelah dilakukan optimasi dihasilkan 432,20 ms, sedangkan untuk *query* kelompok puluhan ribu dihasilkan 4252,80 ms dan setelah dilakukan optimasi dihasilkan 3686,16 ms. Grafik hasil pegujian *query* setelah optimasi dan sebelum optimasi ditunjukkan pada Gambar 5.



Gambar 5. Grafik hasil pengujian optimasi *query*

Dari hasil pengujian dihasilkan rata-rata *time result* dari kelima kelompok *query* pengujian untuk *query* sebelum optimasi yaitu 1037,47 ms dan *query* setelah optimasi yaitu 902,40 ms. Maka dapat disimpulkan bahwa *query* setelah dilakukan optimasi dan dieksekusi pada basis data baru lebih cepat dibandingkan dengan *query* sebelum dilakukan optimasi yang dieksekusi pada basis data yang sama.

5. Kesimpulan

Proses migrasi data pada basis data SIMUNCP dilakukan dengan melakukan ekspor data dari basis data lama dan diimpor ke basis data baru yang merupakan perbaikan dari basis data lama. Pada beberapa aplikasi basis data tersedia fitur untuk melakukan migrasi, namun SIMUNCP menggunakan MySQL sebagai basis datanya dan tidak menyediakan fitur tersebut. Dari hasil migrasi data yang dilakukan dihasilkan kualitas data pada basis data baru adalah baik, terutama pada aspek nonduplikat dengan nilai persentase 100% sedangkan untuk aspek akurasi senilai 92%, nilai ini tidak mencapai 100% disebabkan karena beberapa faktor antara lain adalah disebabkan karena terdapat data yang tidak memiliki nilai referensi pada tabel yang saling berelasi sehingga data tersebut tidak dimasukkan, selain itu terdapat perbedaan jumlah kolom pada beberapa tabel antara basis data baru dengan basis data lama ini disebabkan karena pada basis data baru tabel-tabel yang saling berelasi memiliki kolom yang menghasilkan informasi sama sehingga kolom tersebut dihapus atau tidak dirancang.

Setelah pengukuran kualitas data dilakukan kemudian dilakukan optimalisasi pada *query-query* yang ada pada *source code* aplikasi, *query-query* yang dioptimasi adalah join *query* yang digunakan pada aplikasi dan *query-query* tersebut dikelompokkan berdasarkan jumlah data yang dihasilkan, terdapat lima kelompok *query* dan dari hasil pengujian dari kelima kelompok *query* tersebut dihasilkan *query* yang dioptimasi lebih cepat dengan rata-rata *time result* yaitu 902,40 ms dibandingkan dengan *query* yang belum dioptimasi dengan rata-rata *time result* yaitu 1037,47 ms.

6. Saran

Dari keseluruhan pemaparan yang dijelaskan sebelumnya pada penelitian ini dapat disimpulkan bahwa pada penelitian ini dilakukan migrasi data dengan melakukan ekspor dan impor, kemudian dilakukan pengukuran kualitas data pada basis data baru kualitas data yang diukur adalah aspek akurasi dan nonduplikat, sedangkan aspek aspek lain seperti integritas turunan, kelengkapan data dan konsistensi data belum dilakukan pengujian, serta pengujian untuk optimalisasi *query* dilakukan berdasarkan *time result* dari setiap *query* yang dieksekusi. Sedangkan untuk mengujian berdasarkan *cost* dari dari *query* belum dilakukan pengujian, hal ini merupakan keterbatasan pengetahuan penulis dan keterbatasan waktu sehingga penelitian ini masih terdapat banyak kekurangan, penulis berharap penelitian ini menjadi acuan untuk penelitian-penelitian selanjutnya yang menyangkut tentang migrasi dan optimalisasi basis data.

Referensi

Andoko, B. S. (2010). Migrasi Database dengan Menggunakan SSIS (SQL Server Integration Service) di PT XYZ Indonesia. *Jurnal Teknologi Informasi , 1 (1),* 36-48.

Bafadal, M. M. (2012). Migrasi Basisdata pada Sistem Informasi Akademik Universitas Tanjungpura. *Jurnal SISFOTENIKA , 2* (2), 71-80.

Darmanto, E. (2015). Analisa Optimalisas Bahasa SQL Berdasarkan Relational Algebra pada Kasus Rekapitulasi Mahasiswa Layak Wisuda. *Jurnal SIMETRIS , 6* (2), 405-414.

Fachrurrozi, M. (2013). Peningkatan Fungsionalitas Perangkat Lunak Melalui Restrukturisasi Data (Studi Kasus: Sistem Informasi Akademik Fakultas Ilmu Komputer UNSRI). *Jurnal GENERIC , 4* (1), 33-38.

Federal Student Aid. (2007). *Data Migration Roadmap : A Best Practice Summary.* (Online), (http://federalstudentaid.ed.gov/static/gw/docs/ciolibrary/ECONOPS_Docs/DataMigrationRoadmap.pdf)

Hartono, N. (2015). Kentungan Penggunaan External Function pada Database PostgreSQL. *Jurnal CSRID , 7* (1), 39-47.

Hegadi, R., & Manjunath, T. N. (2013). Data Quality Assessment Model for Data Migration Business Enterprise. *International Journal of Engineering and Technology , 5* (1), 101-109.

Hidayat, W., Aldhi, M. D., & Ananda, D. (2015). Teknik Migrasi Data Lintas DBMS Dengan Menggunakan Metadata. *Informatika, Telekomunikasi dan Elektronika , 7* (2), 137-142.

Li, Q., Honglin, X., & Yan, G. (2009). Research on Performance Optimization and Implementation of Oracle Database. *Proceedings of the 3rd international conference on Intelligent information technology application, 3,* 520-523.

Muslih, Rahmawan, E., & Nurhendratno, S. S. (2015). Desain pola Struktur Mapping Schema untuk Sinkronisasi dan Integrasi Multidatabase Terdistribusi Dalam Mengelola Data Epidemiologi. *Techno. Com , 14* (1), 62-71.

Pasnur, P. (2013). Optimalisasi Query Data Dengan JavaScript Object Notation (JSON) Pada Aplikasi Penerimaan Mahasiswa Baru Online STMIK AKBA. *Jurnal Inspiration, 3* (1), 47-52.

Putra, D. H., & Wibawa, H. A. (2014). Imlementasi Interpretive Transformer Approach dalam Migrasi Data Sebagai Rangkaian Database Reengineering. *Jurnal Masyarakat Informatika, 5* (9), 53-61.

Ricky, M. Y. (2011). Aplikasi Migrasi Database dengan Menggunakan Database dan Replikasi Bi-Directional. *Jurnal COMTECH, 2* (2), 788-797.

Syaifudin, R., Selo, & Hartanto, R. (2014). Review: Implementasi Holap Untuk Optimasi Query Sistem Basis Data Terdistribusi Dengan Pendekatan Algoritma Genetik. *Jurnal Sistem Informasi Bisnis, 1* (3), 164-171.

Utami, E. (2014). The Advantages Of Using Check Constraints In The Academic Database Tables. *Journal of Software, 9* (2), 382-388.