Prosiding Nasional Rekayasa Teknologi Industri dan Informasi XIII Tahun 2018 (ReTII)
November 2018, pp. 58~62
ISSN: 1907-5995

Perancangan Dan Implementasi *Data Query Extractor* Dengan Remote Method-Invocation

Eddy Maryanto
Jurusan Teknik Informattika,Universitas Jenderal Soedirman Purwokerto
Korespondensi: eddy_maryanto@unsoed.ac.id

ABSTRAK

Basis data terdistribusi merupakan salah satu subjek penelitian yang menarik banyak minat peneliti. Basis data terdistribusi memiliki beberapa kelebihan antara lain reliabilitas dan availabilitas data. Masih banyak masalah yang berkaitan dengan basis data teristribusi yang dapat diteliti dan diselesaikan Pada penelitian ini dilakukan desain dan implementasi *query data extractor*. *Query data extractor* merupakan subsistem penting dari sebuah sistem basis data terdistribusi. Subsistem ini mengekstrak *query* yang diisu oleh aplikasi dari sebuah *node* untuk menghasilkan data yang diperlukan untuk pengambilan keputusan yang optimal pada proses fragmentasi dan alokasi.
Kata kunci: basis data terdistribusi, query data extractor, fragmentasi, alokasi

ABSTRACT

Distributed databases have been a research subject attracted many researcher to study. Distributed databases have many advantages as reliability and data availability. There are still many problems in this subject area needed to be solved through researches. In this research, query data extractor, a subsystem of distributed database system is designed and implemented. Query data extractor reside at the heart of the distributed database system. The main task of the query data extractor is to extract query data issued by applications sitting at a site to result data needed by optimizer subsystem in fragmentation and allocation processes.
Keyword : distributed databases, query data extractor, fragmentation, allocation

1. PENDAHULUAN

Teknologi basis data telah merubah paradigma pemrosesan data. Pada paradigma awal, setiap aplikasi mendefinisikan, memelihara, dan memiliki datanya sendiri-sendiri. Setelah dikembangkannya teknologi basis data, pendefinisisan dan pemeliharaan data dilakukan secara terpusat dan aplikasi-aplikasi berbagi data yang ada (*data sharing*). Dengan adanya teknologi basis data juga berdampak pada independensi data, dimana aplikasi-aplikasi tidak akan terpengaruh oleh adanya perubahan pisikal maupun logikal pada struktur data yang ada dan sebaliknya.

Sejalan dengan adanya kemajuan dalam teknologi jaringan komputer, teknologi sistem basis data telah dikembangkan ke arah teknologi sistem basis data terdistribusi atau *distributed database system* (DDBS) yang mana teknologi ini merupakan perpaduan dari dua teknologi yaitu teknologi basis data dan teknologi jaringan komputer. Dengan adanya teknologi sistem basis data terdistribusi, paradigma pengelolaan data telah mengalami pergeseran dari pengelolaan yang terpusat menjadi pengelolaan data yang terintegrasi (*integrated data management*) [2].

Sistem basis data terdistribusi menjanjikan kinerja yang lebih baik dibandingkan sistem basis data terpusat. Peningkatan kinerja meliputi reliabilitas sistem yang lebih tinggi, eksekusi *query* secara paralel, dan kemudahan dalam ekspansi sistem. Disamping menjanjikan kinerja yang lebih baik, basis data terdistribusi memunculkan beberapa masalah yang sampai saat ini belum mempunyai solusi yang memuaskan dan perlu pengkajian lebih lanjut. Permasalahan-permasalahan tersebut antara lain proses fragmentasi, replikasi, dan alokasi pada suatu infrastruktur jaringan komputer dengan topologi dan karakteristik tertentu [3].

Permasalahan lainnya yang juga perlu dikaji adalah masalah yang terkait dengan impelementasi dari konsep atau model yang terkait dengan sistem basis data terdistribusi seperti mengimplementasi server data *query*, pemrosesan *joint-query*, dan *caching* hasil *query* intermediate maupun final yang efisien. Pada penelitian ini akan dikaji tentang permasalahan bagaimana membuat implementasi server data *query* yang efisien berbasis Java Remote-Method Invocation (RMI). RMI merupakan sebuah teknologi pemanggilan metode secara remote yang sangat berguna untuk implementasi sistem terdistribusi [1]

Penelitian ini bertujuan untuk:

(1) Membuat desain arsitektural dari "*Data Query Extractor*" untuk sebuah sistem basis data terdistribusi dengan menggunakan bahasa pemrograman Java

(2) Mengimplementasikan "*Data Query Extractor*" berdasarkan desain yang sudah dibuat dengan berbasiskan teknologi *Remote Method Invocation* (RMI).
(3) Melakukan pengujian terhadap subsistem "*Data Query Extractor*" hasil implementasi.

2. METODE PENELITIAN

Penelitian ini menurut rencana akan dilaksanakan selama enam bulan. Metode penelitian yang digunakan *action research*. Untuk menyelesaikan masalah yang dihadapi, langkah pertama yang dilakukan adalah membuat desain sistem basis data terdistribusi secara keseluruhan, lengkap dengan subsistem yang dibutuhkan. Selanjutnya ditentukan spesifikasi dari salah satu subsistem yaitu subsistem "*Data Query Extractor*". Berdasarkan spesifikasi yang sudah ditentukan, dibuatlah rancangan lebih detil untuk subsistem "*Data Query Extractor*". tersebut. Hasil perancangan disajikan dalam bentuk diagram blok. Pada tahapan selanjutnya, dilakukan implementasi dari rancangan "*Data Query Extractor*" yang sudah dibuat dan kemudian dilanjutkan dengan pengujian fungsionalitasnya. Metode pengembangan sistem yang digunakan adalah metode *Rapid Applicattion Development* (RAD). Jenis pengujian yang digunakan adalah *functional testing*.

3. HASIL DAN ANALISIS

3.1 Desain Arsitektur Sistem Basis Data Terditribusi

Proses penciptaan sebuah sistem terdistribusi melibatkan dua proses utama yaitu fragmentasi dan alokasi. Proses fragmentasi membutuhkan informasi yang berkaitan dengan aplikasi yaitu *query* dan informasi yang berkaitan dengan basis data yaitu skema basis data. Sedangkan proses alokasi memerlukan informasi yang berkaitan dengan aplikasi, basis data, sistem komputer setiap node, dan jaringan komunikasi data. Semua informasi yang dibutuhkan oleh proses fragmentasi dan alokasi seiring dengan perjalanan waktu tentunya dapat berubah, sehingga tingkat optimalitas sebuah sistem terdistribusipun dapat menurun. Berdasarkan fakta ini maka dibutuhkan sistem terdistribusi yang adaptif.

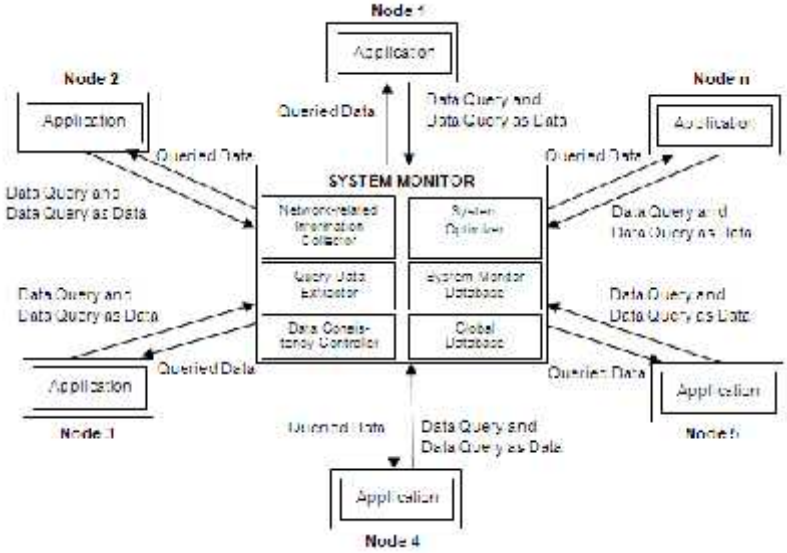
Pada sistem terdistribusi adaptif, proses fragmentasi dan alokasi dilakukan secara berulang dan otomatis apabila tingkat optimalitas sistem sudah menurun pada level tertentu yang disebabkan oleh perubahan yang terjadi. Oleh karena itu, pada sistem terdistribusi adaptif diperlukan sebuah ***System Monitor*** yang berfungsi memonitor tingkat optimalitas sistem, serta melakukan proses refragmentasi dan realokasi apabila penurunan tingkat optimalitas sistem sudah melampaui batas nilai yang ditentukan. Pada sistem terdistribusi adaptif, sistem berevolusi secara otomatis dari sebuah sistem yang pada saat awal merupakan sebuah sistem dengan basis data terpusat menjadi sistem dengan basis data yang terdistribusi. Penelitian ini merupakan penelitian awal dari serangkaian penelitian yang akan dilaksanakan dengan tujuan untuk menciptakan sistem basis data terdistribusi yang adaptif. Sistem basis data terdistribusi adaptif yang akan diciptakan berbasis arsitektur jaringan *peer-to-peer* (P2P) dan menerapkan prinsip *reusable application* yaitu multi DBMS.

Sistem basis data teristribusi adaptif terdiri dari beberapa komponen utama antara lain *system monitor*, *application*, dan DBMS. *System monitor* sendiri terdiri dari beberapa subsistem antara lain *query data extractor*, *network-related information collector*, *system optimizer*, *data consistency controller*, *global fragment data directory*, *system monitor database*, dan *global database*. *Global Database* ada pada saat awal sebelum terjadi proses fragmentasi dan alokasi. Desain arsitektural Sistem Basis Data Terditribusi Adaptif pada sebelum terjadi proses fragmentasi dan alokasi disajikan pada Gambar 1, sedangkan setelah terjadi proses fragmentasi dan alokasi disajikan pada Gambar 2. Perbedaan antara keduanya hanya terletak pada aliran data, dimana sebelum terjadi proses fragmentasi dan alokasi, semua *data query* (termasuk *data query as data*) mengalir dari semua node dimana aplikasi berada menuju node dimana *system monitor* berada, dan hasil *query* mengalir pada arah kebalikannya yaitu dari node dimana *system monitor* berada menuju node dimana aplikasi berada (Gambar 1).

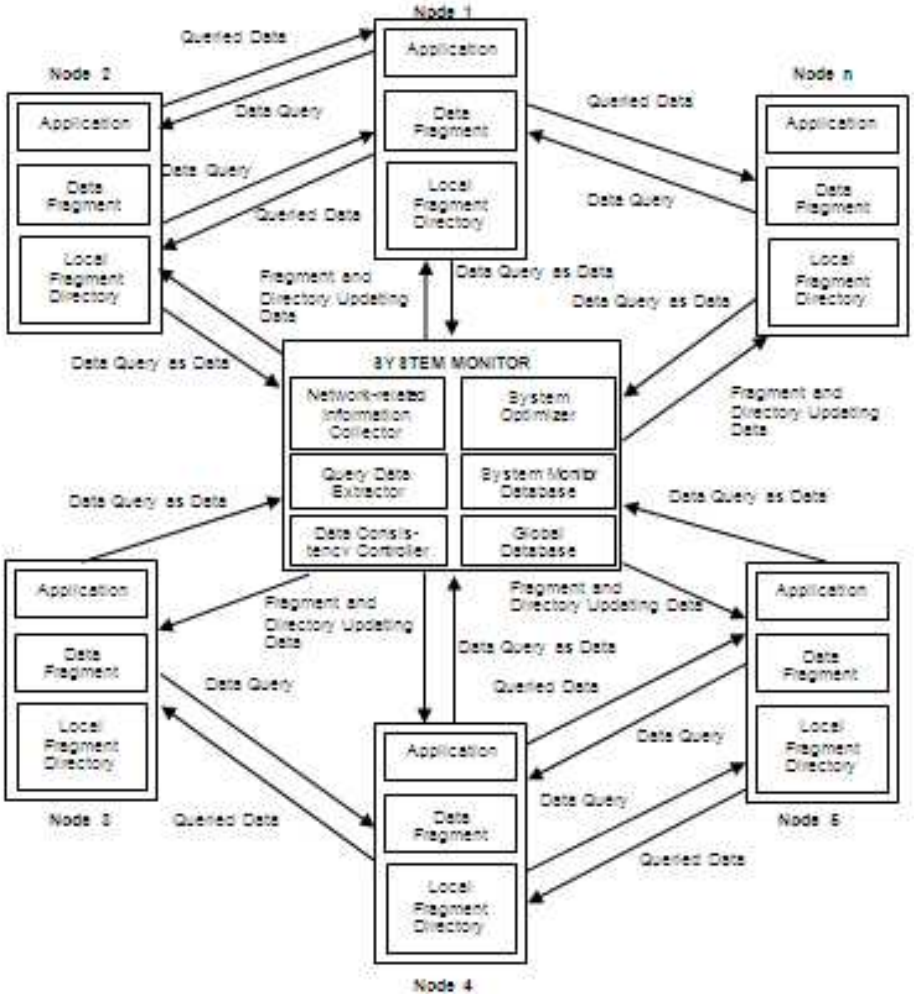
Setelah proses fragmentasi dan alokasi terjadi, aliran *data query* dan hasil *query* mengalami perubahan, dimana sekarang data query mengalir dari node dimana aplikasi berada menuju node dimana *data fragment* yang diinginkan berada, sedangkan aliran hasil *query* terjadi pada arah kebalikannya. Selain itu, setelah proses fragmentasi dan alokasi terjadi *global database* bisa tidak ada lagi, yang ada sekarang adalah *local database* atau *data fragment*, serta diciptakannya *local fragment directory* pada masing-masing node dimana aplikasi berada dan *global fragment directory* pada node dimana *system monitor* berada.

Skema sistem yang disajikan pada Gambar 1 merepresentasikan sistem aktual, sedangkan skema yang disajikan pada Gambar 2 merepresentasikan sebuah ilustrasi sistem, dimana jumlah *site* atau *node* pada realitanya bisa lebih banyak daripada yang ada pada gambar tersebut dan pola alokasi *fragment* pada realitanya juga bisa berbeda dengan yang ada pada gambar tersebut sehingga pola aliran *data query* dan *queried data* pada realitanya juga dapat berbeda dengan yang ada pada gambar tersebut. Pola aliran *data*

query as data yaitu aliran *data query* yang diperlakukan sebagai data untuk keperluan analisis optimalitas atau efisiensi sistem oleh *sistem monitor* baik sebelum maupun sesudah terjadinya fragmentasi dan alokasi adalah tetap yaitu dari masing-masing *site*/*node* menuju *site*/*node* dimana *system monitor* berada.



Gambar 1. Struktur Sistem sebelum Fragmentasi dan Alokasi

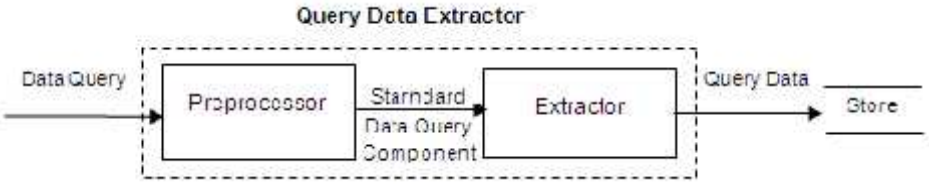


Gambar 2. Struktur Sistem setelah Fragmentasi dan Alokasi

3.2 Desain Arsitektur Subsistem *Query Data Extractor*

Menginggat keterbatasan sumberdaya terutama waktu, maka pada penelitian ini hanya akan dirancang dan diimplementasikan subsistem *Query Data Server* atau *Query Data Extractor* yang merupakan salah satu subsistem dari *System Monitor*. *Query Data Extractor* berfungsi untuk mengekstrak informasi dari *query data* yang dikirim oleh site/node yang dibutuhkan oleh subsistem *System Optimizer* untuk menentukan tingkat efisiensi sistem saat ini, dan juga untuk melakukan fragmentasi/refragmentasi serta alokasi/realokasi apabila diperlukan.

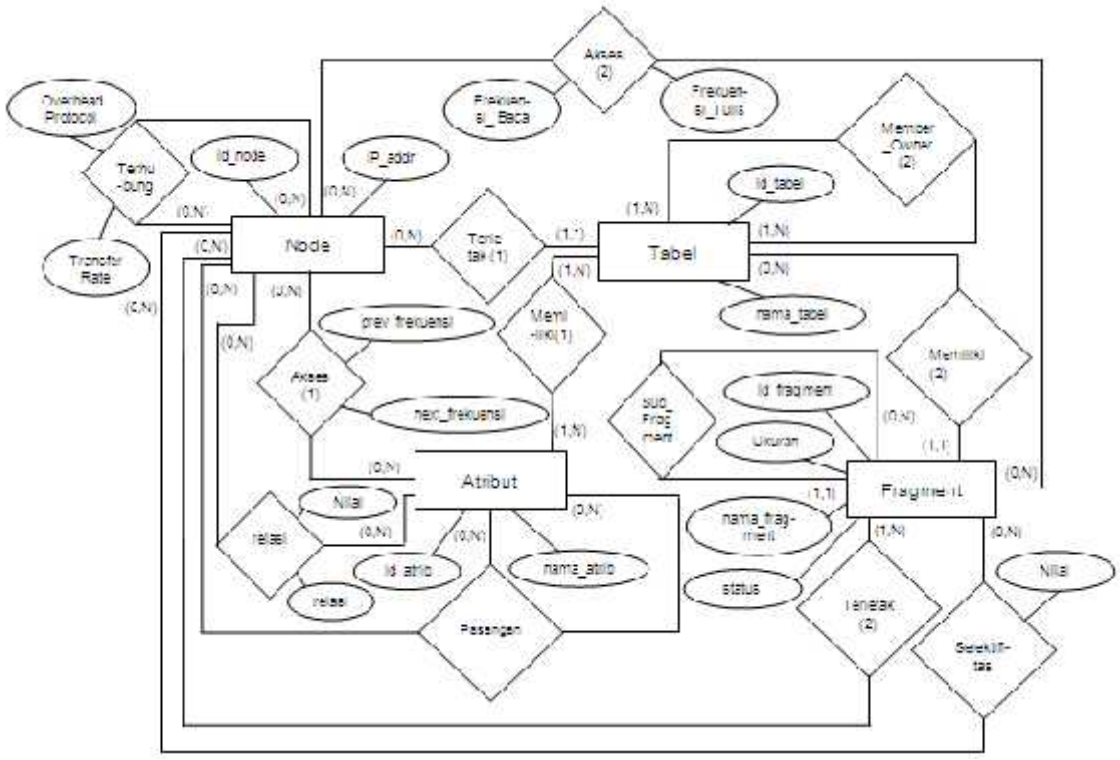
Subsistem *Query Data Extractor* sendiri terdiri dari dua bagian yaitu *tokenizer* dan *data storing manager* sebagaimana disajikan pada Gambar 3. Tugas dari *tokenizer* adalah melakukan pemilahan bagian yang dibutuhkan dari *data query as data* dengan bagian lainnya. Data yang dihasilkan oleh *tokenizer* selanjutnya akan diklasifikasi sebelum disimpan kedalam Basis Data Sistem Monitor (*System Monitor Database*) oleh Manajer Penyimpanan Data (*Data Storing Manager*).



Gambar 3. Query Data Extractor

3.3 Desain Konseptual Basis Data Sistem Basis Data Terdistribusi

Berdasarkan analisis kebutuhan data untuk sistem monitor basis data terdistribusi dibuatlah desain konseptual basis data untuk sistem ini sebagaimana disajikan pada Gambar 4.

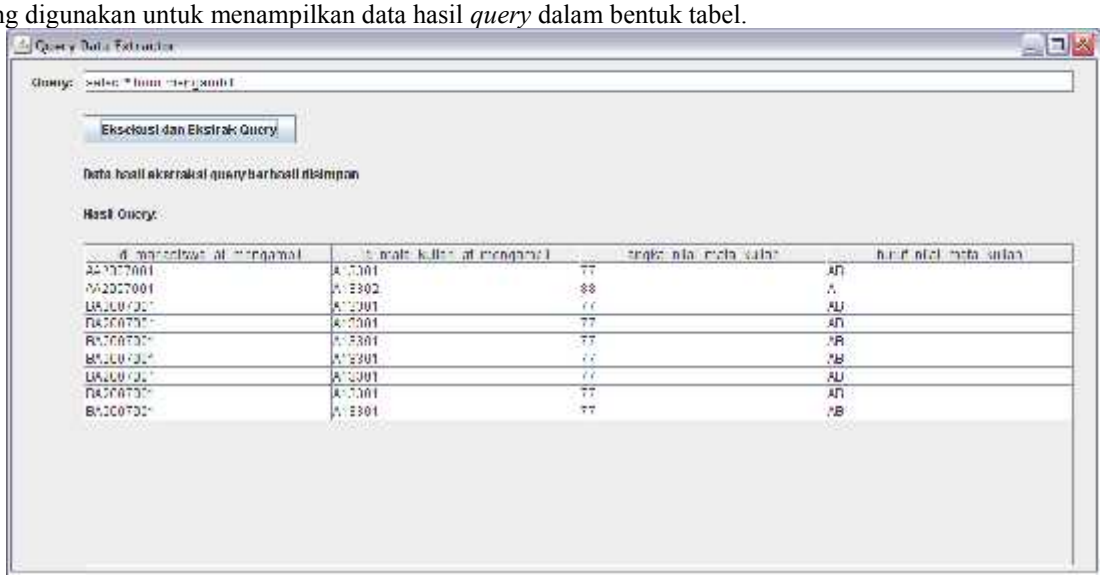


Gambar 4. Desain Konseptual Basis Data

3.4 Desain Antarmuka Subsistem Query Data Extractor

Desain antarmuka pengguna untuk subsistem query data extractor disajikan pada Gambar 5. Pada bagian atas terdapat sebuah *text field* untuk memasukkan SQL *query*. Pada bagian bawah *text field* ini terdapat sebuah komponen *button*. Pada komponen *button* ini ditambahkan sebuah *event* yang akan

dieksekusi pada saat *button* ini diklik. Pada *event* ini ditambahkan kode program untuk mengeksekusi dan mengekstrak *query* yang dimasukkan via *text field*. Pada bagian bawah *button* terdapat dua buah label, yang mana label atas digunakan untuk menampilkan notifikasi berhasil tidaknya proses penyimpanan data yang diperoleh dari hasil ekstraksi *query*, sedangkan label dibawahnya digunakan untuk menampilkan notifikasi terkait dengan hasil dari eksekusi *query*. Pada bagian bawah dari GUI sistem ini, terdapat sebuah *scoll pane* yang digunakan untuk menampilkan data hasil *query* dalam bentuk tabel.



Gambar 5. Tampilan Antarmuka Subsistem Query Data Extractor

4. KESIMPULAN

Data Query Extractor (DQE) yang dirancang dan diimplementasi telah dapat digunakan untuk mengekstrak query baca maupun tulis yang berbasis tiga perintah dasar yaitu *select*, *update*, dan *delete*, termasuk *joint selection*. DQE dapat dengan mudah dikembangkan sehingga dapat memproses *variant query* lainnya yang lebih kompleks yang melibatkan lebih banyak clausa seperti having, group by, dan lain-lain.

Aspek lain dari sistem basis data terdistribusi yang peneliti rencanakan untuk diteliti adalah desain dan implementasi *data query extractor* yang berbasis *Common Object Request Broker Architecture* (CORBA), juga desain dan implementasi subsistem *optimizer*.

UCAPAN TERIMAKASIH

Pada kesempatan ini pula, tim peneliti mengucapkan terimakasih yang sebesar-besarnya kepada Pimpinan UNSOED melalui LPPM yang telah memberikan dukungan dana, sehingga penellitian ini dapat terlaksana. Ucapan terimakasih juga peneliti sampaikan kepada tim reviewer yang senantiasa bekerja keras dalam penyeleksian proposal penelitian dalam upaya peningkatan kualitas penelitian secara berkelanjutan.

DAFTAR PUSTAKA

[1] Graba, J., 2007. An Introduction to Network Programming with Java. Revised Edition. Springer. New York. USA.

[2] Özsu, M.T. and Patrick Valduriez, 2010. Principles of Distributed Database Systems. Third Edition. Springer. New York. USA.

[3] Patel, A., Rakshitkumar Hirapara, and Vivekkumar Dhamecha, 2014. A Review on Fragmentation Techniques in Distributed Database, International Journal of Modern Trends in Engineering and Research (IJMTER), Volume 01, Issue 03, [September - 2014].