Penerapan metode *Data Mart Query* (DMQ) dalam *Distributed Database System*

Untung Rahardja[1]
Retantyo Wardoyo[2]
Shakinah Badar[3]

Email: untung@pribadiraharja.com; shakinah@pribadiraharja.com;

Diterima: 5 Agustus 2009 / Disetujui: 29 Agusutus 2009

ABSTRACT

Along with rapid development of network and communication technology, the proliferation of online information resources increases the importance of efficient and effective distributed searching in distributed database environment system. Information within a distributed database system allows users to communicate with each other in the same environment. However, with the escalating number of users of information technology in the same network, the system often responds slowly at times. In addition, because of large number of scattered database in a distributed database system, the query result degrades significantly in the occurrence of large-scale demand at each time data needs. This paper presents a solution to display data instantly by using Data Mart Query. In other words, Data Mart Query (DMQ) method works to simplify complex query manipulating table in the database and eventually creates a presentation table for final output. This paper identifies problems in a distributed database system especially display problem such as generating user's view. This paper extends to define DMQ, explain the architecture in detail, advantages and weaknesses of DMQ, the algorithm and benefits of this method. For implementation, the program listings displayed written ASP script and view the example using DMQ. DMQ methods is proven to give significant contribution in Distributed Database System as a solution that is needed by network users to display data instantly that is previously very slow and inefficient.

ABSTRAKSI

Seiring dengan kemajuan zaman, laju pertumbuhan IPTEK yang semakin pesat mendorong perkembangan teknologi jaringan. Suatu informasi disalurkan antara satu database dengan database yang lain memungkinkan untuk saling berhubungan dalam suatu sistem database yang terdistribusi. Namun dengan banyaknya pengguna teknologi informasi dalam suatu

1. **Dosen Jurusan Sistem Informasi, STMIK Raharja**
   Jl. Jend Sudirman No.40 Modern Cikokol-Tangerang Telp 5529692
2. **Dosen Fakultas Science dan Teknologi, Universitas Gadjah Mada**
   Sekip Utara Bls, Yogyakarta 55281
3. **Mahasiswa Jurusan Sistem Informasi, STMIK Raharja**
   Jl. Jend Sudirman No.40 Modern Cikokol-Tangerang Telp 5529692

jaringan, membuat suatu sistem kadang berjalan lambat. Selain itu banyaknya data yang tersebar dalam suatu sistem database yang terdistribusi, mengakibatkan terjadinya proses query besar-besaran pada saat setiap kali membutuhkan data. Dengan menggunakan Data Mart Query, memungkinkan sebuah display data dapat ditampilkan dengan cepat. Dengan kata lain, metode Data Mart Query dapat langsung menampilkan source code pada display dan proses query yang dikerjakan pada engine. Dalam artikel ini, diidentifikasikan masalah yang dihadapi dalam suatu sistem yang terdistribusi khususnya masalah dalam menampilkan view ke pengguna, definisi dari Data Mart Query tersebut, arsitekturnya dalam sebuah database, keuntungan dan kelemahan dari Data Mart Query, algoritma serta manfaat dari metode ini. Pada implementasinya, ditampilkan listing program yang ditulis menggunakan script ASP serta contoh view dengan menggunakan Data Mart Query. Kontribusi metode Data Mart Query dalam Distributed Database System merupakan suatu solusi yang sangat membantu kebutuhan user pada proses display data yang sebelumnya sangat lambat dan tidak efisien.

Kata kunci: Data Mart Query, Distributed Database

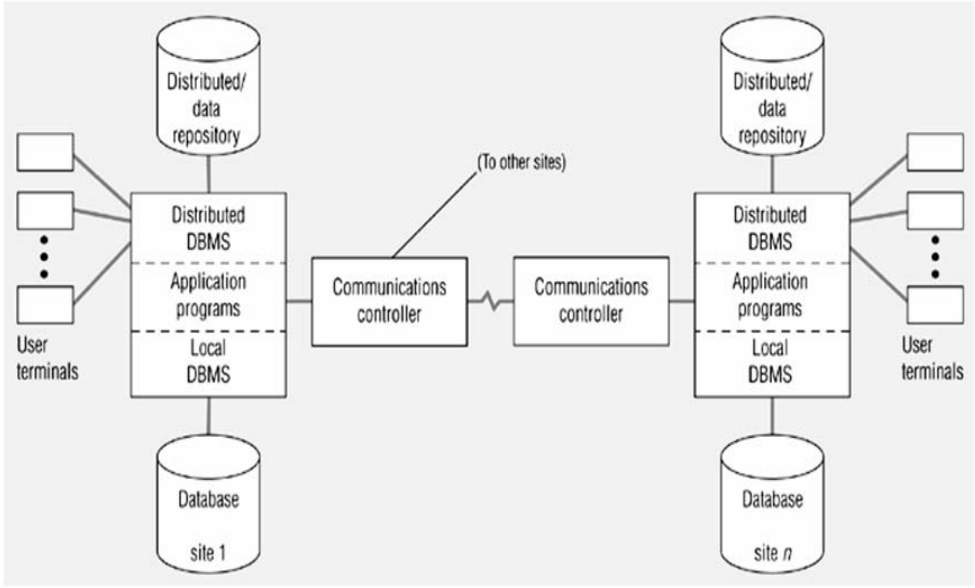
PENDAHULUAN

Perkembangan teknologi yang terus meningkat dengan cepat, mempengaruhi laju kebutuhan manusia atas informasi, terlebih disuatu organisasi atau perusahaan. Informasi terus mengalir dan jumlahnya semakin lama semakin meningkat seiring dengan jumlah permintaan, serta jumlah data yang semakin banyak. Selain itu penggunaan *database* dalam suatu perusahaan dan organisasipun semakin banyak terlebih dengan adanya sistem jaringan. *Database* dapat didistribusikan dari satu komputer ke komputer lain. Jumlah arus pemakaipun meningkat seiring besarnya organisasi atau perusahaan.

Organisasi maupun perusahaan membutuhkan sistem informasi untuk mengumpulkan, mengolah dan menyimpan data serta menyalurkan suatu informasi. Berkembangnya sistem informasi dari waktu ke waktu telah menghasilkan banyak informasi yang semakin kompleks. Kompleksnya informasi tersebut disebabkan oleh banyaknya permintaan, jumlah data serta tingkat iterasi perintah SQL dalam suatu program.

Pemanfaatan teknologi informasi oleh organisasi atau perusahaan secara garis besar bertujuan untuk memudahkan pelaksanaan proses bisnis dan meningkatkan kemampuan kompetitif. Melalui teknologi informasi, diharapkan proses bisnis perusahaan dapat dilaksanakan lebih mudah, cepat, efisien dan efektif. Penggunaan teknologi jaringan didalam suatu organisasi ataupun perusahaan menjadi hal yang biasa. Didalam suatu sistem jaringan sekarang ini banyak organisasi ataupun perusahaan yang telah menerapkan *database* terdistribusi untuk sistem *database*nya.

Database terdistribusi merupakan sebuah *database* yang berada dibawah kontrol DBMS sentral dimana tempat penyimpanan tidak terpusat ke suatu CPU tetapi mungkin disimpan di *multiple* komputer dalam lokasi fisik yang sama atau disebarkan melalui jaringan komputer yang saling terkoneksi.



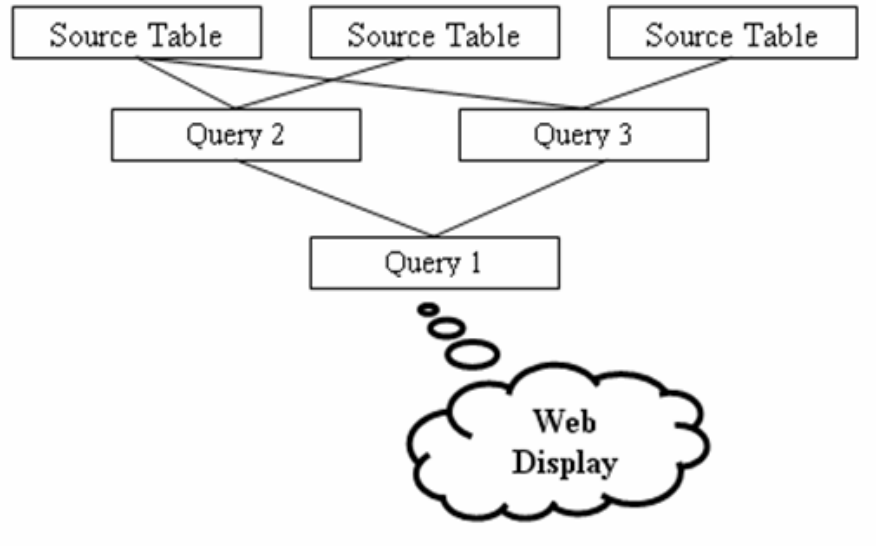
Gambar 1. Arsitektur *Distributed Database*

Gambar diatas, merupakan gambaran arsitektur *database* yang terdistribusi. Dimana didalam sistem *database* terdistribusi ini memungkinkan beberapa terminal terkoneksi dalam suatu sistem *database*. Dan masing-masing terminal ini bisa mengakses atau memperoleh data dari *database* baik yang ada dikomputer pusat maupun dikomputer lokal ataupun *database* yang satu dengan *database* yang lain. *Database* terdistribusi juga mempunyai keunggulan seperti dapat merefleksikan struktur organisasi, otonomi lokal. Kesalahan dalam satu fragmen tidak akan mempengaruhi *database* keseluruhan. Adanya *balancing database* didalam *server* dan sistem dapat dimodifikasi tanpa mempengaruhi modul lain.

Untuk itu, penyimpanan data pada tabel SQL *server* dalam sebuah *distributed database*, merupakan langkah praktis yang dilakukan banyak kalangan saat ini. Bagi instansi tertentu proses penyajian data haruslah cepat, misalnya pada google.com dimana penyajian data yang diinginkan oleh kosumen haruslah dengan cepat berada pada tampilan *display*, padahal tidak sedikit data yang harus dikeluarkan. Pada dasarnya penyajian data dari tabel memerlukan waktu proses yang sesuai dengan banyaknya data yang akan di tampilkan. Penggunaan cara konvensional pada dasarnya adalah cara praktis, karena tidak membutuhkan pengeditan bila data di *database* bertambah, namun apakah kecepatan tampilan akan lama bila data yang ditampilkan banyak.

PERMASALAHAN

Distributed database memang memiliki banyak keunggulan terlebih untuk struktur organisasi saat ini. Namun diantara keunggulan itu, *distributed database* juga memungkinkan suatu sistem menjadi lebih kompleks, karena banyaknya *database* yang tersebar dan jumlah data yang banyak dan terus meningkat didalam suatu organisasi maupun perusahaan. Jika suatu *database* memiliki sejumlah data yang tersimpan dengan banyak *query* dan tabel, suatu permintaan mengakibatkan proses pencarian data atau *source data* menjadi lambat. Selain itu banyaknya user yang dapat mengakses suatu tampilan *web* atau *Web display* suatu sistem informasi juga menjadi lambat.. Berikut ini disampaikan tampilan sumber data konvensional yang memiliki *query* bertingkat:



Gambar 2. *Source Data* Konvensional

Dari gambar diatas, kita bisa lihat bahwa untuk menghasilkan suatu tampilan pada *web display*, pada *source data* konvensional perlu dilakukan *query* bertingkat. *Source data* dilakukan mulai dari tabel yang satu kemudian tabel yang lain lalu ke *query* yang satu ke *query* yang lain. Bayangkan jika ada ratusan atau ribuan tabel dan *query* didalam suatu *database*, kemudian *database* itu terdistribusi sehingga terjadi hubungan antara *database* yang satu dengan yang lain. Berapa lama waktu yang dibutuhkan hanya untuk memberikan satu tampilan *web*?

Dari penjelasan di atas, dapat dirumuskan beberapa permasalahan yaitu sebagai berikut:

1. Apakah proses *view* karena *query* bertingkat pada *Database* terdistribusi sudah ada penelitiannya?
2. Berapa dampak dari lambatnya sebuah proses akibat *query* bertingkat?
3. Metode apa yang dapat digunakan untuk mempercepat proses display pada suatu sistem *database* terdistribusi?
4. Apa kelemahan dan kelebihannya dengan metode baru yang diusulkan ini?

LITERATURE RE*VIEW*

Banyak penelitian yang sebelumnya dilakukan mengenai *distributed database*. Dalam upaya pengembangan *distributed database* ini perlu dilakukan studi pustaka sebagai salah satu dari penerapan metode penelitian yang akan dilakukan. Diantaranya adalah mengidentifikasikan kesenjangan (*identify gaps*), menghindari pembuatan ulang (*reinventing the wheel*), mengidentifikasikan metode yang pernah dilakukan, meneruskan penelitian sebelumnya, serta mengetahui orang lain yang spesialisasi dan area penelitiannya sama dibidang ini. Beberapa *Literature review* tersebut adalah sebagai berikut :

1. Penelitian ini dilakukan oleh Jun Lin Lin dan Margaret H. Dunham dari Southerm Methodist University dan Mario A. Nascimento berjudul " *A Survey of Distributed Database Checkpointing*". Penelitian ini membahas mengenai *checkpointing* pada *database* terdistribusi dan pendekatan-pendekatan yang digunakan. Penelitian ini bermula dari adanya banyak survey yang dilakukan berkenaan dengan proses *recovery database*, dan banyak teknik yang diusulkan untuk mengatasinya. Dengan *distributed database checkpointing*, dapat mengurangi waktu proses *recovery* suatu kegagalan didalam *database* terdistribusi. *Checkpointing* dapat digambarkan sebagai suatu aktivitas menulis informasi ke penyimpanan yang stabil selama operasi normal dalam rangka mengurangi jumlah pekerjaan pada saat restart. Penelitian ini membantah bahwa sedikit batasan dan sedikit sumber daya menjadi masalah dalam pendekatan *database* terdistribusi, serta Membantah bahwa *checkpointing* hanya dapat digunakan untuk sistem distribusi yang *multidatabase*. Meskipun penelitian ini telah banyak dilakukan namun cukup rumit dalam implementasinya. Dengan penelitian ini kita dapat mengembangkan *database* terdistribusi dengan *checkpointing* untuk mempercepat proses *recovery database*[1].

2. Penelitian ini dilakukan oleh David J. DeWitt dari Universitas Wisconsin dan Jim Gray tahun 1992 berjudul "*Parallel Database Systems:The Future of High Performance Database Processing*". Penelitian ini dilakukan dengan Konsep *database* terdistribusi yang

merupakan *database* yang disimpan pada beberapa komputer yang terdistribusi satu sama lain. Pada penelitian ini, dijelaskan Sistem *database* paralel mulai menggantikan *Mainframe* komputer besar untuk pengolahan data dan transaksi tugas. Paralel *database* komputer memiliki arsitektur yang berkembang dari penggunaan perangkat lunak yang eksotik untuk perangkat keras yang paralel. Seperti kebanyakan aplikasi, user menginginkan *hardware* sistem *database* yang murah, cepat. Ini menyangkut tentang prosesor, memori dan disk. Akibatnya, konsep *hardware database* yang eksotis tidak sesuai untuk teknologi saat ini. Di lain sisi, ketersediaan *microprocessors* cepat, murah dan kecil menjadi paket standar murah tapi cepat sehingga menjadi *platform* yang ideal untuk sistem *database* paralel. Stonebraker mengusulkan rancangan sederhana untuk spektrum disain yaitu *shared memory, shared disk* dan *shared nothing*. Dan bahasa yang digunakan dalam *database* adalah SQL sesuai dengan standar ANSI dan ISO. Dengan penelitian ini, kita dapat mengembangkan sistem *database* agar dapat digunakan diberbagai ruang lingkup[2].

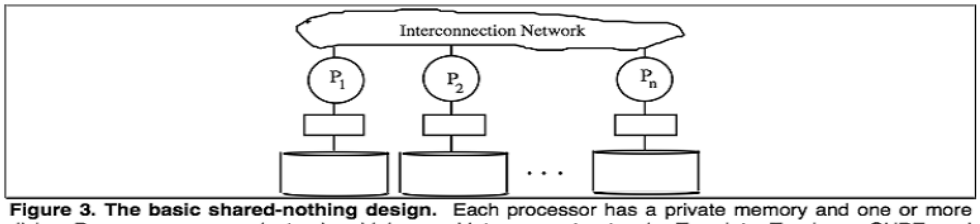


Figure 3. The basic shared-nothing design. Each processor has a private memory and one or more disks. Processors communicate via a high-speed interconnect network. Teradata, Tandem, nCUBE, and the newer VAXclusters typify this design.

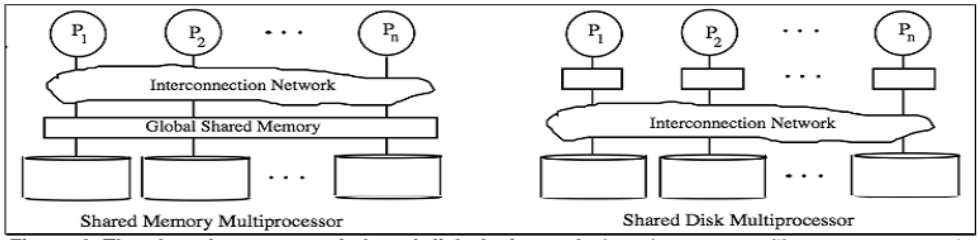
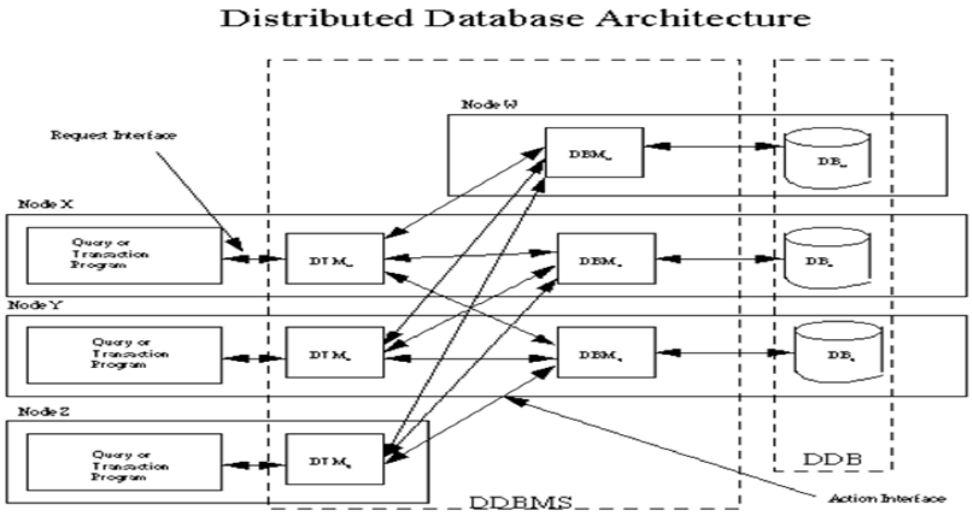


Figure 4. The shared-memory and shared-disk designs. A shared-memory multi-processor connects all processors to a globally shared memory. Multi-processor IBM/370, VAX, and Sequent computers are typical examples of shared-memory designs. Shared-disk systems give each processor a private memory, but all the processors can directly address all the disks. Digital's VAXcluster and IBM's Sysplex typify this design.

Gambar 3. *Desain Shared-Nothing, Shared-Memory* dan *Shared-Disk*

3. Penelitian ini dilakukan oleh Carolyn Mitchell dari Norfolk State University berjudul "*Components of a Distributed Database*" tahun 2004. Penelitian ini membahas tentang komponen-komponen didalam *database*. Salah satu komponen utama dalam DDBMS

adalah *Database Manager*. "Sebuah *Database Manager* adalah perangkat lunak yang bertanggung jawab untuk memproses segmen data yang didistribusikan. Komponen utama lainnya adalah *Query User Interface*, yang merupakan sebuah program klien yang bertindak sebagai sebuah antarmuka untuk Transaksi Manager yang terdistribusi.." Sebuah Transaksi Manager terdistribusi adalah program yang menterjemahkan permintaan dari pengguna dan mengkonversi mereka ke *query database* manager, yang biasanya didistribusikan. Sebuah sistem *database* yang terdistribusi terbuat dari kedua manajer yaitu *Database* Manager dan Transaksi Manager Terdistribusi[3].



Gambar 4. Arsitektur *Distributed Database* dan Komponennya

4. Penelitian yang dilakukan oleh Hamidah Ibrahim, "*Deriving Global Integritas Dan Local Rules For Distributed Database".* Fakultas Ilmu Komputer dan Teknologi Informasi Universitas Putra Malaysia, 43400 UPM Serdang. Ibrahim mengatakan bahwa tujuan terpenting didalam *database* sistem adalah menjamin konsistensi data, yang berarti bahwa data yang terdapat dalam *database* harus baik dan akurat. Didalam pelaksanaannya untuk menjaga konsistensi perubahan data sangat sulit, khususnya untuk didistribusikan dalam *database.* Dalam tulisan ini, menjelaskan sebuah algoritma penegakan aturan berdasarkan mekanisme untuk didistribusikan *database* yang bertujuan meminimalisir jumlah data yang harus ditransfer atau diakses diseluruh jaringan yang menjaga konsistensi dari *database* di satu situs, yaitu di situs mana pembaruan perlu dilakukan. Teknik ini disebut sebagai tes integritas generasi, yang berasal dari lokal dan global integritas, dan aturan yang telah efektif dapat mengurangi biaya kendala dalam memeriksa suatu data yang telah didistribusikan dalam lingkungan. Didalam penelitian

ini telah berhasil menghasilkan sebuah sistem sentralistik yang besar dengan tingkat kehandalan yang tinggi untuk integritas data[4].

5. Penelitian yang dilakukan oleh Steven P. Coy dari *University of Maryland* berjudul "*Security Implication of the Choice of Distributed Database Management System Model: Relational Vs Object Oriented"*. Penelitian ini menjelaskan bahwa keamanan data harus dibenahi ketika mengembangkan *database* dan diantaranya memilih antara *relational* dan *object oriented model*. Banyak faktor yang harus dipertimbangkan, terutama dari segi efektifitas dan efisiensi, juga apakah sekuritas dan integritas ini memakan sumber daya yang terlalu besar tidak semata mata fitur keamanan. Kedua pilihan ini akan mempengaruhi kekuatan dan kelemahan dari *database* tersebut. Untuk *centralized database* kedua model ini bisa dikatakan sama baiknya. Namun untuk *distributed database*, *relational model* lebih unggul dibidang sekuritas. Ini lebih banyak disebabkan karena *object oriented model database* masih kurang maturitasnya. Sehingga didalam lingkungan heterogenous, proses integritasnya masih menimbulkan banyak masalah. OODBMS tetap saja masih perlu perkembangan teknologi lebih lanjut, namun di lingkungan homogenous, OODBMS dapat menjadi pilihan yang baik[5].

6. Penelitian yang dilakukan oleh Stephane Gançarski, Claudia León, Hubert Naacke, Marta Rukoz and Pablo Santini yang berjudul "*Integrity Constraint Checking in Distributed Nested Transactions over a Database Cluster*" adalah sebuah solusi untuk memeriksa integritas dan kendala global dalam berhubungan multi *database* sistem. Penelitian ini juga menyajikan hasil eksperimental yang diperoleh atas solusi PC *cluster* dengan Oracle9i DBMS. Tujuan adalah melakukan eksperimentasi untuk mengukur waktu yang dihabiskan dalam memeriksa kendala global dalam sistem yang terdistribusi. Alhasil menunjukkan bahwa *overhead* berkurang hingga 50% dibandingkan dengan pemeriksaan integritas yang terpusat. Studi menunjukkan bahwa sistem berkemungkinan besar melanggar *referential integrity* dan *global conjunctive constraints.* Namun dengan cara *distributed nested transactions*, dengan adanya eksekusi dan parallelism, integritas dapat lebih terjamin[6].

7. Penelitian ini dilakukan oleh Allison L. Powell James C.dkk, Perancis Departemen Ilmu Komputer Universitas Virginia, berjudul berjudul *"The Impact of Database Selection on Distributed Searching".* Penelitian ini menjelaskan bahwa *distributed searching* terdiri dari 3 bagian yaitu *database selection, query processing*, dan *results merging*. Cukup beberapa *database* yang dijadikan *database* seleksi (tidak semuanya) dan performa akan meningkat cukup signifikan. Bila seleksi *database* dilakukan dengan baik, pencarian secara *distributed* akan berkinerja lebih baik dibandingkan pencarian secara sentralisasi.

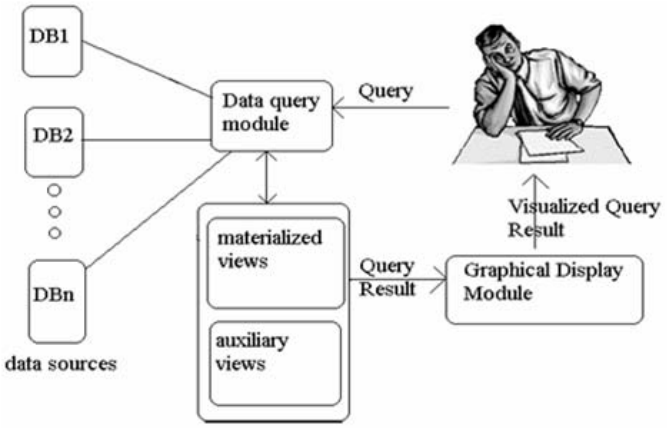
Pencarian *database* juga ditambahkan proses seleksi dan ranking sehingga secara potensial meningkatkan efektifitas pencarian data[7].

8. Penelitian ini dilakukan oleh Yin-Fu Huang dan HER JYH-CHEN (2001) dari Universitas Nasional Sains dan Teknologi Yunlin Taiwan, berjudul "*Fragment Allocation in Distributed Database Design*". Pada penelitian ini menjelaskan mengenai *Wild Area Network* (WAN), fragmen alokasi adalah isu utama dalam distribusi *database* desain karena kekhawatiran kinerja keseluruhan didistribusikan pada system *database*. Disini system yang diusulkan sederhana dan modelnya yang komprehensif mencerminkan aktivitas transaksi yang didistribusikan dalam *database*. Berdasarkan model dan informasi transaksi, dua bentuk algoritma dikembangkan untuk mendapatkan alokasi yang optimal seperti total biaya komunikasi yang sebisa mungkin diminimalkan. Hasilnya menunjukkan bahwa alokasi fragmentasi ditemukan dengan menggunakan algoritma yang tepat akan menjadi lebih optimal. Beberapa penelitian juga dilakukan untuk memastikan bahwa biaya rumus dapat benar-benar mencerminkan biaya komunikasi didunia nyata[8].

9. Penelitian ini dilakukan oleh Nadezhda Filipova dan Filcho Filipov (2008) dari *University of Economics*. Varna, Bul. Kniaz BorisI berjudul "*Development of database for distributed information measurement and control system*". Penelitian ini menjelaskan mengenai pengembangan *database* dari pengukuran informasi yang didistribusikan dan sistem kontrol yang menerapkan metode optik untuk plasma spectroscopy fisika dan penelitian atom collisions dan menyediakan akses untuk mendapat informasi dan sumber daya perangkat keras di jaringan Intranet/Internet, berdasarkan *database* pada sistem manajemen *database* Oracle9i. Perangkat lunak klien yang diwujudkan adalah dalam *Java Language*. Perangkat lunak ini dikembangkan dengan menggunakan model arsitektur, yang memisahkan aplikasi data dari komponen grafis presentasi dan masukan pengolahan logika. Berikut grafis presentasi telah dilaksanakan, pengukuran radiasi dari Spectra beam plasma dan benda, perangsangan fungsi *non-elastis collisions* dari berat partikel dan analisis data yang diperoleh dalam percobaan sebelumnya. Berikut grafis klien yang memiliki fungsi interaksi dengan *database browsing* informasi tentang percobaan dari jenis tertentu, pencarian data dengan berbagai kriteria, dan memasukkan informasi tentang percobaan sebelumnya[9].

10. Penelitian yang dilakukan oleh Lubomir Stanchev dari University of Waterloo tahun 2001 berjudul "*Semantic Data Control In Distributed Database Environment*". Penelitian ini menyatakan bahwa ada tiga tujuan utama dalam *semantic data control* yaitu: *view managemen, data security* dan *semantic integrity control.* Dalam sebuah relasi, fungsi-fungsi ini dapat mencapai keseragaman dengan menegakkan aturan-aturan manipulasi

kontrol data. Solusinya adalah dengan sentralisasi ataupun terdistribusi. Dua hal utama yang efisien untuk melakukan kontrol adalah definisi data dan penyimpanan aturan (situs pilihan) dan penegakan desain algoritma yang meminimalkan biaya komunikasi. Masalahnya adalah sulit, karena peningkatan fungsi (dan umum) cenderung meningkatkan komunikasi situs. Solusi untuk semantik data kontrol terdistribusi adalah eksistensi dari sentralisasi solusi. Masalahnya adalah sederhana jika aturan kontrol sepenuhnya direplikasi di semua situs dan sulit jika situs otonomi dipatenkan. Selain itu, khusus optimasi dapat dilakukan untuk meminimalkan biaya kontrol data tetapi dengan tambahan *overhead* seperti pengelolaan data *snapshot*. Dengan demikian, spesifikasi kontrol data terdistribusi harus disertakan pada desain *database* sehingga biaya kontrol update untuk program-program ini juga dipertimbangkan[10].



Gambar 5. *Data Visualization* dengan *materialized and auxiliary views*

Dari sepuluh *literature review* yang ada, telah banyak penelitian mengenai *checkpointing*, *parallel database system*, pembahasan *component database system*, juga mengenai *security.* Disamping itu juga ada pembahasan mengenai *nested transaction, distributed searching, view management* dan juga *fragment allocation*. Namun dapat disimpulkan pula bahwa belum ada peneliti yang secara khusus membahas atau mengatasi masalah proses *view* yang lambat akibat *query* bertingkat.

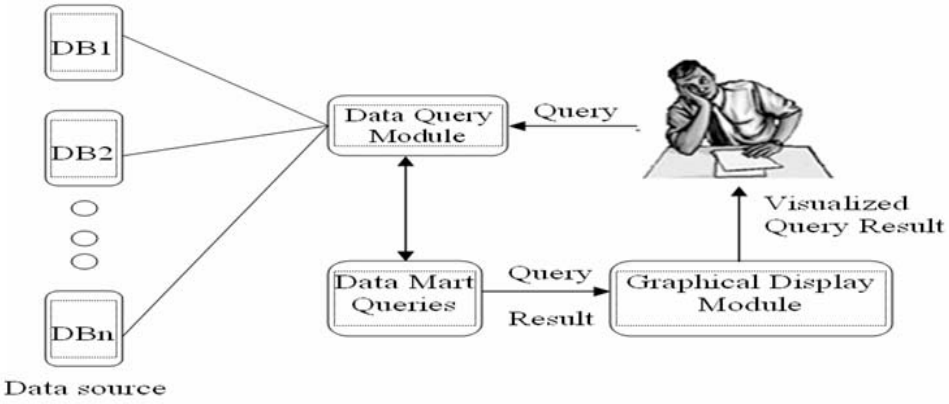
PEMECAHAN MASALAH

Untuk mengatasi berbagai masalah diatas, maka diperlukan proses yang cepat dan efisien dalam mengakses seluruh data yang banyak dan tidak teratur di *database*, terlebih untuk suatu sistem *database* yang terdistribusi. Saat ini programmer lebih memilih menggunakan Ms. Acces dan fungsi *query* untuk mengerjakan seluruh *script* perintah.

Alhasil proses *query* besar-besaran terjadi setiap membutuhkan data. Penggunaan SQL *server* bukanlah hal yang baru dalam hal ini, maka dari itu diusulkan untuk dibentuknya suatu system yang lebih menuju ke proses pada saat loading penyajian data dan memiliki kecepatan yang secara linear lebih cepat dibandingkan dengan cara konvensional. DMQ (*Data Mart Query*) merupakan metode yang menerapkan analogi "*Waste Space for Speed*". DMQ juga merupakan salah satu metode yang berbentuk terhadap pemisahan antara "*Engine*" dan "*Display*". Dengan kata lain metode DMQ dapat langsung menampilkan *source code* pada *display* dan proses *query* yang dikerjakan pada *engine*. Secara umum DMQ menghasilkan sebuah *display data* yang jauh lebih cepat dibandingkan dengan menggunakan metode umum, karena DMQ tidak melakukan proses lagi dalam menampilkan data. Dan akhirnya DMQ merupakan suatu solusi yang dapat membantu kebutuhan *user* pada proses *display data* yang sebelumnya sangat lambat dan tidak efesien.

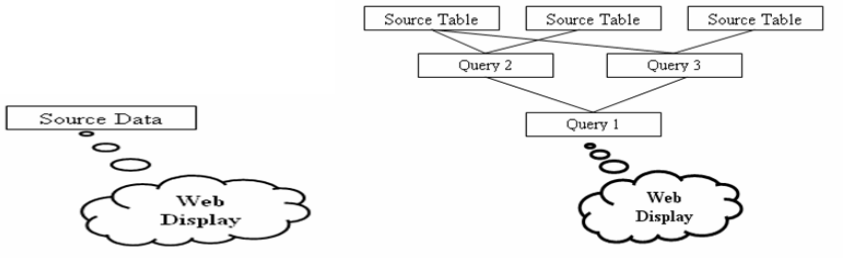
Pada *Data Mart Query* sumber data berasal dari tabel. Jadi pada proses DMQ ini, mengalokasikan seluruh data yang dipilih ke dalam suatu tabel. Sehingga *user* tidak perlu memikirkan pembuatan struktur tabel tujuan, yang perlu dipikirkan hanyalah dimana data tersebut berada. DMQ ini digunakan untuk menghindari penggunaan *Query* majemuk. DMQ akan mengorbankan besarnya kapasitas penyimpanan data (*space harddisc*) untuk meningkatkan kecepatan (*increase speed*). DMQ membutuhkan *triger update data* untuk menghasilkan data yang mutakhir.

Berikut ini adalah gambaran mengenai permintaan data dari *user*. Dimana seorang *user* melakukan permintaan akan suatu tampilan, kemudian *data query module* melakukan pencarian pada db1, db2 sampai dbn. Dengan menggunakan *Data Mart Query* atau DMQ dari *data query module* langsung menghasilkan *query* yang diinginkan dalam bentuk *graphical display module* yang dapat dilihat oleh *user*.



Gambar 6. *Data Visualization* dengan DMQ

Dengan *Data Mart Query* (DMQ) proses pencarian data lebih singkat, karena tidak seperti *source data* konvensional yang harus mencari dari tabel. *Data Mart Query* (DMQ) bisa memotong waktu proses karena proses pencarian data hanya ke satu tabel yang telah digabungkan.

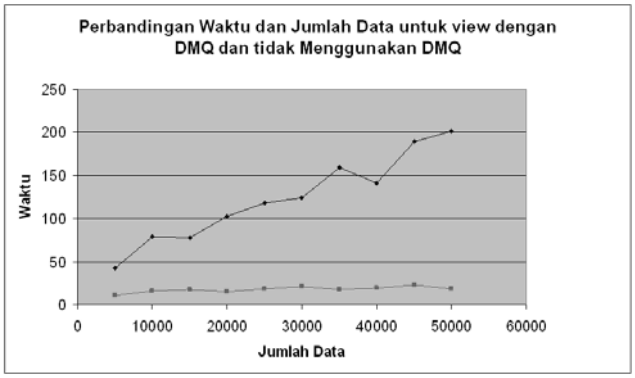


Gambar 7. Perbandingan *source data* konvensional dan *source data* dengan *Data Mart Query*

Jika dibandingkan maka tampilannya akan seperti gambar diatas. Dimana dengan DMQ bisa membuat tampilan *web* lebih cepat dilakukan karena tidak memerlukan proses pencarian yang rumit. Hal ini juga dapat dibuktikan pada grafik 1.

Keterangan :
= Tidak menggunakan DMQ
= Menggunakan DMQ

Grafik 1. Perbandingan Waktu dan Jumlah data



Pada grafik diatas, dapat dilihat bagaimana perbandingan waktu dan jumlah data untuk suatu *display web* dimana jumlah data untuk masing-masing grafik nilainya sama. Grafik diatas menjelaskan bahwa jika suatu *view* tidak menggunakan DMQ, maka grafiknya akan naik keatas, atau semakin besar jumlah data maka waktu prosesnya akan semakin lama. Namun sebaliknya jika menggunakan DMQ untuk *view*nya maka berapapun jumlah

datanya waktu yang dibutuhkan untuk proses *view*nya relatif konstan.

REGRESI LINEAR

Selain dibuktikan secara grafik, dapat juga dibuktikan secara eksponensial dengan persamaan Regresi Linear sebagai berikut:

$$Y' = a + bX$$

Dimana a = Y pintasan, (nilai Y' bila X = 0)

b = kemiringan dari garis regresi(kenaikan atau penurunan Y' untuk setiap perubahan satu-satuan X) atau koefisien regresi, yang mengukur besarnya pengaruh X terhadap Y kalau X naik satu unit

X = nilai tertentu dari variabel bebas

Y' = nilai yang diukur/ dihitung pada variabel tidak bebas

Nilai a dan b pada persamaan regresi dapat dihitung dengan humus dibawah ini:

a = é - X .....[11]

A. Perhitungan Regresi Linear untuk *view* tanpa menggunakan DMQ

Berikut ini adalah data yang diperoleh jika tidak menggunakan *Data Mart Query* :

Tabel 1. Data dan Perhitungan untuk *view* tanpa DMQ

Jumlah Data X	Waktu Y	x $(X - \overline{X})$	y $(Y - \bar{Y})$	xy	y^2	x^2
5000	43	-22500	-80.25	1811250	6480.25	506250000
10000	79	-17500	-44.5	778750	1980.25	306250000
15000	78	-12500	-45.5	568750	2070.25	156250000
20000	103	-7500	-20.5	153750	420.25	56250000
25000	118	-2500	-5.5	13750	30.25	6250000
30000	124	2500	0.5	1250	0.25	6250000
35000	159	7500	35.5	266250	1260.25	56250000
40000	141	12500	17.5	218750	306.25	156250000
45000	189	17500	65.5	1146250	4290.25	306250000
50000	201	22500	77.5	1743750	6006.25	506250000
275000	1235			6702500	22844.5	2062500000

$$\overline{X} = \frac{\sum x}{N} = \frac{275000}{10} = 27500$$

$$\bar{Y} = \frac{\sum y}{N} = \frac{1235}{10} = 123.5$$

Dari hasil perhitungan diatas, nilai a dan b dihitung sebagai berikut:

$$b = \frac{\sum xy}{\sum x^2} = \frac{6702500}{2062500000} = 0.0032$$

$$\begin{aligned} a &= \bar{Y} - b\,\overline{X} \\ &= 123.5 - 0.0035\ (27500) \\ &= 35.5 \end{aligned}$$

Sehingga, persamaan regresi yang memperlihatkan hubungan kedua *variabel* antara jumlah data dan waktu untuk suatu tampilan *view* adalah:

$$Y = 35.5 + 0.0032X$$

Jadi setiap kali jumlah data bertambah maka waktu prosesnya akan bertambah 0.0032 kali.

B. Perhitungan Regresi Linear untuk *view* dengan menggunakan DMQ
Berikut ini adalah data yang diperoleh jika menggunakan *Data Mart Query* :

Tabel 2. Data dan perhitungan untuk *view* dengan DMQ

Jumlah Data X	Waktu Y	x $(X - \overline{X})$	y $(Y - \bar{Y})$	xy	y^2	x^2
5000	11	-22500	-7	157500	49	506250000
10000	16	-17500	-2	35000	4	306250000
15000	18	-12500	0	0	0	156250000
20000	15	-7500	-3	22500	9	56250000
25000	19	-2500	1	-2500	1	6250000
30000	21	2500	3	7500	9	6250000
35000	18	7500	0	0	0	56250000
40000	20	12500	2	25000	4	156250000
45000	23	17500	5	87500	25	306250000
50000	19	22500	1	22500	102	506250000
275000	180			355000	203	2062500000

Dari hasil perhitungan diatas, berdasarkan cara yang sama dihasilkan persamaan berikut ini:

$$Y = 15.25 + 0.0001X$$

Jadi setiap kali jumlah data bertambah maka waktu prosesnya akan bertambah 0.0001 kali.

Berdasarkan kedua perhitungan regresi diatas, dapat dibuktikan bahwa b_{DMQ} tidak signifikan.

$$b_{DMQ} : b_n = 0.0001 : 0.0032$$
$$b_{DMQ} : b_n \approx 1:32$$

Ini menunjukkan bahwa b_{DMQ} tidak signifikan dibandingkan b_n. dengan demikian regresi untuk non DMQ tetap :

$$y = 35.5 + 0.0032x$$

sedangkan regresi DMQ menjadi :

$$y = 15.25.$$

KORELASI LINEAR

Istilah korelasi menunjuk pada konsep saling hubungan diantara beberapa variabel. Dalam bentuknya yang kompleks, korelasi melibatkan banyak variabel sekaligus. Namun dalam pembahasan ini saya mengambil dua variabel yaitu X untuk jumlah data dan Y untuk waktu. Salah satu formula untuk menghitung besarnya koefisien korelasi antara dua variabel yang masing-masing berskala interval telah dirumuskan oleh ahli statistika dan disebut formula korelasi *product-moment Pearson*. Rumusannya adalah sebagai berikut:

Untuk *view* tanpa menggunakan DMQ semakin besar jumlah data cenderung diikuti oleh semakin besarnya waktu proses dan sebalik nya semakin kecil jumlah data maka semakin kecil pula waktu proses yang dibutuhkan. Tentu perubahan pada variabel X tidak diikuti oleh perubahan pada variabel Y secara mutlak. Ada sedikit variasi yang memperlihatkan besarnya perubahan pada X tidak selalu diikuti secara proporsional oleh perubahan pada Y. Hal ini menunjukkan adanya indikasi hubungan yang tidak sempurna antara dua variabel dan hal ini memang menjadi karakteristik variabel non fisik. Hubungan yang sempurna sekali hanya dapat diteemui pada variabel-variabel ilmu eksakta.
Korelasi dinyatakan dalam angka yang disebut koefisien korelasi dan diberi symbol $r_{xy.}$ Koefisien korelasi mengandung dua makna, yaitu kuat lemahnya hubungan dan arah hubungan antar variabel.

Kuat lemahnya hubungan antara dua variabel diperlihatkan oleh besarnya harga mutlak koefisien korelasi yang bergerak antara 0 sampai dengan 1. Semakin mendekati angka 0 berarti hubungan semakin lemah dan semakin koefisien mendekati angka 1 berarti hubungan semakin kuat.
Dari data pada Tabel 1. dapat dihitung korelasi linear antara jumlah data dan waktu proses untuk *view* tanpa menggunakan DMQ sebagai berikut:

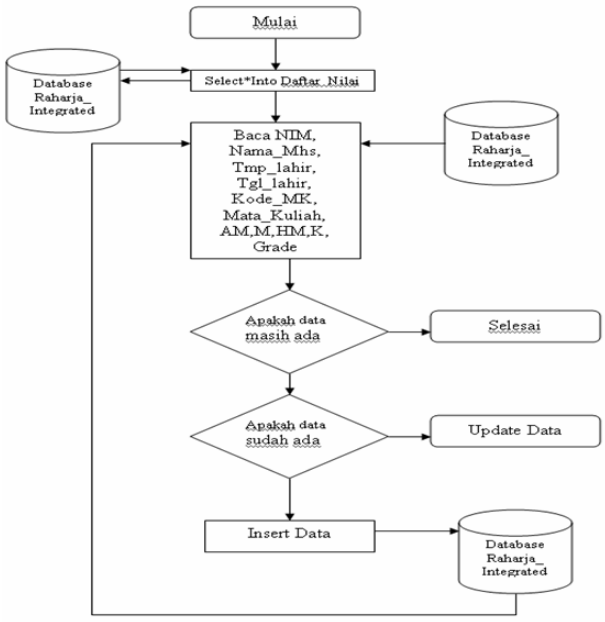
= 0.97

Sedangkan korelasi linear antara jumlah data dan waktu proses untuk *view* dengan menggunakan DMQ berdasarkan data pada tabel 2. adalah sebagai berikut:

= 0.55

Tingginya koefisien ini diartikan sebagai adanya hubungan yang kuat antara jumlah data dengan waktu. Tanda positif pada koefisien korelasi tersebut menunjukkan bahwa Jumlah data yang semakin tinggi memiliki waktu proses yang semakin tinggi.

MERANCANG PROGRAM MELALUI FLOWCHART



Gambar 8. Flowchart Daftar Nilai IPK

LISTING PROGRAM

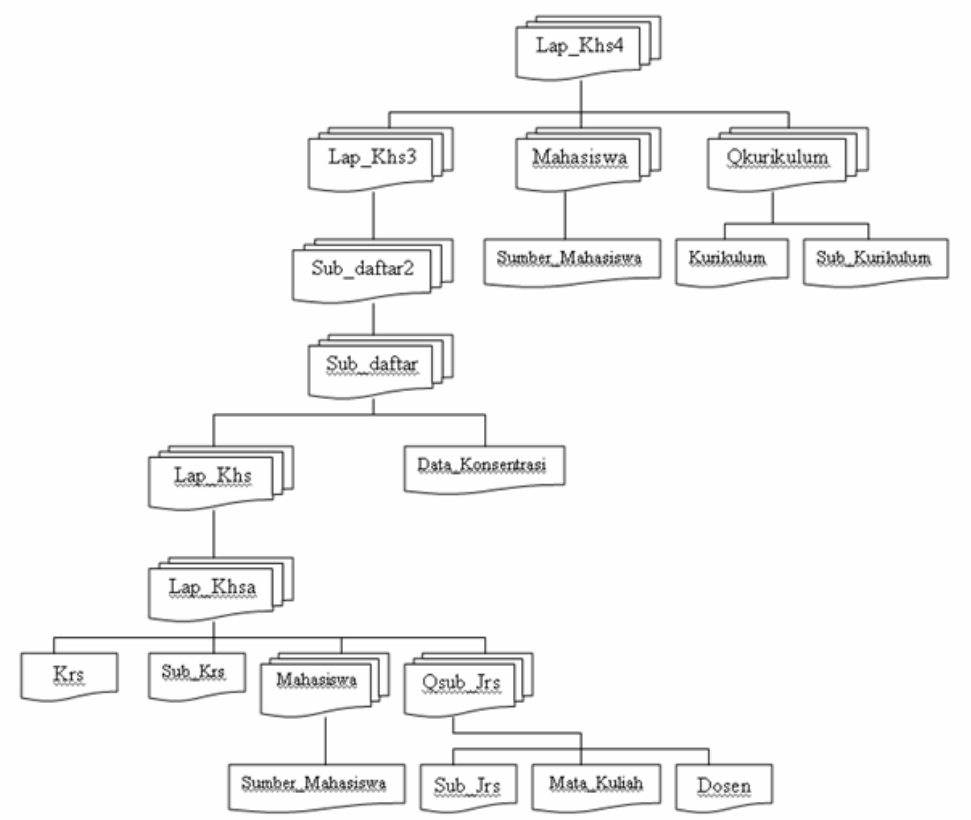
Daftar nilai IPK merupakan sebuah program yang menggunakan metode DMQ (*Data Mart Query*), sehingga listing program yang akan ditampilkan yaitu listing program untuk *update* daftar nilai IPK. Berikut listing programnya :

```
<%
dim conn
set conn=server.CreateObject("ADODB.Connection")
conn.open "PROVIDER=MSDASQL;DRIVER={SQL
SERVER};SERVER=rec;DATABASE=raharja_integrated;"
%>
<% dim strsql1,rs1
strsql1="drop table Daftar_Nilai"
set rs1=conn.execute(strsql1) %>
<% dim strsql2,rs2
strsql2="select * INTO Daftar_Nilai from Lap_KHS4"
set rs2=conn.execute(strsql2) %>
<% response.redirect ("default.asp") %>
```

Gambar 9. Listing program update daftar nilai IPK

IMPLEMENTASI

Konsep *Data Mart Query* (DMQ) telah diimplementasikan pada Perguruan Tinggi Raharja dalam membuat tampilan Daftar Nilai IPK (Indeks Prestasi Kumulatif Kumulatif). IPK tersebut merupakan rata-rata dari IPS (Indeks Prestasi Sementara). IPK adalah sistem penilaian yang disiapkan untuk mengukur dan mengetahui tingkat kemampuan mahasiswa selama melakukan perkuliahan. Dengan Penggunaan DMQ didalam pembuatan daftar nilai IPK, membuat tampilannya dengan cepat dapat diakses.



Gambar 10. Struktur *Query* didalam *database* Raharja_Integrated

ALGORITMA Lap_khs4 :

SELECT dbo.Lap_Khs3.NIM, dbo.Lap_Khs3.Kode_MK, dbo.Lap_Khs3.Mata_Kuliah, dbo.Lap_Khs3.Sks, dbo.Lap_Khs3.Grade, dbo.Lap_Khs3.AM, dbo.Lap_Khs3.K, dbo.Lap_Khs3.M, dbo.QKurikulum.Kelompok, dbo.QKurikulum.Kajur FROM dbo.Lap_Khs3 INNER JOIN dbo.Mahasiswa ON dbo.Lap_Khs3.NIM = dbo.Mahasiswa.NIM INNER JOIN dbo.QKurikulum ON dbo.Mahasiswa.Jenjang = dbo.QKurikulum.Jenjang AND dbo.Mahasiswa.Jurusan = dbo.QKurikulum.Jurusan AND dbo.Mahasiswa.Konsentrasi = dbo.QKurikulum.Konsentrasi AND dbo.Lap_Khs3.Kode_MK = dbo.QKurikulum.Kode

TAMPILAN LAYAR

Tampilan layar (*interface*) Panel Pimpinan telah terintegrasi dengan beberapa sistem informasi seperti *Raharja Multimedia Edutainment* (RME), *Absensi On-line* (AO), dan

Student Information Services (SIS). Adapun *interface – interface* tersebut terdiri dari :

a. Tampilan Utama Panel Pimpinan

Pada tampilan ini kita bisa melihat keseluruhan Mahasiswa yang ada pada Perguruan Tinggi Raharja beserta (Indeks Mutu Kumulatif) IMK dan (Indeks Prestasi Kumulatif) IPKnya. Pada tampilan ini juga terdapat jumlah mahasiswa laki-laki dan perempuan, Status mahasiswa beserta jumlahnya, mahasiswa yang berhak mengikuti UTS dan UAS, Top 10 best dan *Top 10 Worst* IPK dan IMK, serta Rata-rata IPK baik untuk mahasiswa aktif maupun lulusan Perguruan Tinggi Raharja.

Gambar 11. Tampilan Utama Panel Pimpinan

Pada kolom disebelah kiri ataupun atas, ketika akan di klik *link-link*nya, maka akan terbuka sebuah URL pada kolom sebelah kanan. Dalam gambar diatas terdapat *link* Asdir. Ketika *link* tersebut di klik, maka akan terbuka sebuah URL yang berisi seluruh Top 100 Mahasiswa dengan status Aktif yang diurutkan secara *descending*. URL tersebut memiliki *interface* seperti gambar dibawah ini:

< **Asdir** > Top 100 Mahasiswa Dengan Status: Aktif (Sort by IPK: Descending) Show All »

NO	NIM	NAMA	IMK	IPK
100	0711459145	Eko Prasetiyani	92	3.92
99	0614457689	Shakinah Badar	96	3.91
98	0822362010	Rihlah Saidah	95	3.90
97	0833462296	Ageng Setiani Rafika	98	3.90
96	0812461530	Ari Yanti Wardani	97	3.88
95	0823360959	Nanik Kurniasari	97	3.88
94	0631456226	Anil Ram	92	3.87
93	0812461303	Rummina Pratiwi	85	3.87
92	0713459014	Tri Wulansari	93	3.86
91	0812461175	Eka Inayatulloh	97	3.85
90	0811361601	Dirga Nopia Alyasah	95	3.84
89	0812461668	Nico Sahrial	97	3.84
88	0621455916	Panji Alam Setiazi	77	3.84
87	0833461506	Bangkit Mulia Ramadhan	96	3.83
86	0631457667	Pajar Saputra	86	3.82
85	0614457558	Andrew Tirta	87	3.82
84	0811461102	Aghnia Sabila	92	3.81
83	0813462176	Ita Anggraeni	96	3.81
82	0722457902	Anselmus Dilang	92	3.81

Gambar 12. Tampilan Top 100 Mahasiswa denagn status aktif

Pada tampilan diatas, terdapat NIM, Nama Mahasiswa, serta IPK dan IMKnya. Untuk dapat melihat secara detail daftar nilai IPK seorang mahasiswa, silahkan klik nilai IPK mahasiswa tersebut.

b. Tampilan daftar nilai IPK pada Panel Pimpinan

Berbeda dengan *interface* sebelumnya, *interface* IPK pada panel pimpinan ini khusus menggambarkan nilai Indeks Prestasi Kumulatif (IPK) setiap mahasiswa. IPK merupakan nilai rata–rata dari keseluruhan nilai yang diperoleh pada seluruh semester yang telah dijalankan oleh setiap mahasiswa. IPK tersebut dikemas dalam sebuah "Daftar Nilai IPK" yang formatnya dapat dilihat pada gambar dibawah ini:

PERGURUAN TINGGI SWASTA : STMIK RAHARJA
JURUSAN : SISTEM INFORMASI
KONSENTRASI : KOMPUTER AKUNTANSI
JENJANG PENDIDIKAN : STRATA 1

DAFTAR NILAI

NAMA MAHASISWA : SHAKINAH BADAR
TEMPAT, TANGGAL LAHIR : TANGERANG, 2-Februari-1988
NOMOR INDUK MAHASISWA : 0614457689

No	Kode Mkl	Mata Kuliah	Prestasi KHS				No	Kode Mkl	Mata Kuliah	Prestasi KHS			
			HM	AM	K	M				HM	AM	K	M
MPK							**MKB**						
1	BI101	Bahasa Inggris I	A+	4.00	3	12.00	1	AK105	Akuntansi I	A	4.00	3	12.00
2	BI102	Bahasa Inggris II	A+	4.00	3	12.00	2	EK131	Matematika Ekonomi	A+	4.00	3	12.00
3	BI103	Bahasa Indonesia	B+	3.30	2	6.60	3	JR111	Jaringan Komputer	A	4.00	3	12.00
4	UM100	Pancasila	A	4.00	2	8.00	4	LA103	Logika dan Algoritma	A	4.00	3	12.00
5	UM110	Agama	B+	3.30	2	6.60	5	MT104	Aljabar Linier II	A+	4.00	2	8.00
6	UM140	Kewarganegaraan	A	4.00	2	8.00	6	MT185	Manajemen Pemasaran	A	4.00	2	8.00
MKK							7	PA101	Pengantar Teknologi Informasi	A	4.00	2	8.00
1	HP100	Hukum Paten Dan Merk	A+	4.00	2	8.00	8	PA115	Paket Program Niaga	A	4.00	3	12.00
2	MJ100	Pengantar Manajemen	A	4.00	2	8.00	9	PR183	Perancangan Homepage	A-	3.70	3	11.10
3	MJ110	Manajemen Proyek	A+	4.00	2	8.00	10	SI131	Sistem Operasi	A	4.00	3	12.00
4	MJ170	Metode Penelitian	A	4.00	2	8.00	11	SI139	Perancangan Basis Data	A+	4.00	3	12.00
5	MJ175	Manajemen Perkantoran	A	4.00	2	8.00	12	SI142	Data Warehouse	A	4.00	3	12.00
6	MT103	Aljabar Linier I	A	4.00	2	8.00	13	SI152	Komunikasi Data	A	4.00	2	8.00
7	MT111	Kalkulus I	A	4.00	2	8.00	14	SI161	Pemrograman Terstruktur	A	4.00	3	12.00
8	MT112	Kalkulus II	A+	4.00	2	8.00	15	SI171	Struktur Data	A	4.00	3	12.00
9	MT121	Statistik Deskriptif	A+	4.00	2	8.00	16	SI200	Keamanan Komputer	A-	3.70	2	7.40
10	MT131	Statistik Probabilitas	A-	3.70	3	11.10	17	SI330	Analisis Proses Bisnis	A	4.00	2	8.00
11	SI103	Analisa Sistem Informasi	A	4.00	3	12.00	18	SK140	Sistem Terdistribusi	A-	3.70	2	7.40
12	SI111	Perancangan Sistem Informasi	A+	4.00	3	12.00	19	TI220	Interaksi Manusia Dan Komputer	B+	3.30	2	6.60
13	SI138	Sistem Basis Data	A-	3.70	3	11.10	**MPB**						
14	SI221	Testing dan Implementasi	A+	4.00	3	12.00	1	AK106	Akuntansi II	A	4.00	3	12.00
15	SI320	Pengetahuan Bisnis	A+	4.00	2	8.00	2	AK107	Akuntansi III	A	4.00	3	12.00
16	SI340	Manajemen Sains	A+	4.00	3	12.00	3	AK133	Akuntansi Berbasis Komputer	A-	3.70	3	11.10
17	SI350	Konsep Sistem Informasi	A	4.00	2	8.00	4	AK161	Pemeriksaan Akuntansi	A	4.00	3	12.00
MBB							5	TI171	Organisasi Komputer I	A	4.00	2	8.00
							6	TI172	Organisasi Komputer II	B+	3.30	2	6.60
							7	TI999	Independent Study	A+	4.00	2	8.00

KETERANGAN		JUMLAH	121	473.6
HM	Huruf Mutu	IPK	3.91	
AM	Angka Mutu	JUMLAH KREDIT KOMULATIF	121	
K	Kredit			
M	Mutu		Tangerang, 7-Mei-2009	

DINA FITRIA MURAD, S.KOM.

Gambar 13. Tampilan Daftar Nilai IPK

Untuk memberikan tampilan daftar nilai IPK diatas, banyak tabel dan *Query* yang digunakan. Sehingga jika menggunakan *source data* secara konvensional membutuhkan waktu yang lama. Namun tampilan daftar nilai IPK diatas dibuat dengan menggunakan *Data Mart Query*, sehingga pada saat membuka halaman ini, tidak membutuhkan waktu yang lama.

KESIMPULAN

Berdasarkan uraian diatas, disimpulkan bahwa *Data Mart Query* (DMQ) merupakan metode yang tepat untuk mempercepat waktu proses pada suatu sistem informasi dengan *database* yang terdistribusi. DMQ ini digunakan untuk menghindari penggunaan *Query* majemuk. Dengan demikian DMQ akan mengorbankan besarnya kapasitas penyimpanan data (*space harddisc*) untuk meningkatkan kecepatan (*increase speed*) dalam pengaksesan. Hal ini pun telah dibuktikan baik secara logik, secara grafik dengan perhitungan regresi linear dan korelasi linear dan juga melalui implementasi.

DAFTAR PUSTAKA

1. Allison L. Powell, James C. French, Jamie Callan, Margaret Connell and Charles L. Viles (2000). *The Impact of Database Selection on Distributed Searching. 23rd ACM SIGIR Conference on Information Retrieval (SIGIR'00)*, pages 232-239.

2. DeWitt. D.J., Gray.J. *Parallel Database Systems* (1992). *The Future of High Performance Database Processing.* San Francisco: Computer Sciences Department, University of Wisconsin.

3. Filipova Nadezhda dan Filipov Filcho (2008). *Development Of Database For Distributed Information Measurement And Control System* University of Economics. Varna, Bul. Kniaz Boris I.

4. Hamidah Ibrahim (2001). *Deriving Global And Local Integrity Rules For A Distributed Database.* Departement of Computer Science Faculty of Computer Science and Information Technology, University Putra Malaysia 43400 UPM Serdang.

5. Huang Yin-Fu dan JYH-CHEN HER (2001). *Fragment Allocation in Distributed Database Design*. Nasional Yunlin Universitas Sains dan Teknologi Yunlin. Taiwan 640, R.O.C.

6. Lin. J. L., Dunham M. H. and Nascimento M. A. (1997) *A Survey of Distributed Database Checkpointing.* Texas: Department of computer science and engineering, Shoutern Methodist University.

7. Mitchell Carolyn (2004). *Component of a distributed database.* Department of Computer science, Norfolk state University.

8. Stanchev Lubomir (2001). *Semantic Data Control In Distributed Database Environment*. University of Waterloo.

9. Stephane Gangarski, Claudia Leon, Hurbert Naacke, Marta Rukoz and Pablo Santini (2006). *Integrity Constraint Checking In Distributed Nested Transactions Over A Database Clustur.* Laboratorie the Information Paris 6. University Pierre et Marie Curie 8 rue du Capitaine Scott, 75015, Paris. Centro de Computacion Paralela Y Distribuida, Universidad Central de Venezuela. Apdo. 47002, Los Chaguaramos, 1041 A, Caracas, Venezuela.

10. Steven P Coy (2008). *Security Implications of the Choice of Distributed Database Management System Model: Relational Vs Object Oriented*. University of Maryland.

11. Supranto(2000). *Statistik Teori dan Aplikasi*, Erlangga.