PENERAPAN METODE LOAD-BALANCING CLUSTERS PADA DATABASE SERVER GUNA PENINGKATAN KINERJA PENGAKSESAN DATA

Halim Setya Muliyantoro
Program Studi Manajemen Informatika
Akademik Manajemen Informatika dan Komputer Bina Sarana Informatika Purwokerto
Jl. DR. Bunyamin No.106, Pabuaran Purwokerto
halim.hsm@bsi.ac.id

ABSTRACT

Information systems are complete and easy access is a necessity that a company owned.With the company's information system is its credibility to go public and to better known by the general public.In line with the development of information technology very rapidly, sometimes in actual access to the system failure can also occur due tothe failure happened server side. Failure caused by the server itself dies and there is no backup from another serverwhen the primary server directly replace dead or can also occur over load Access on the server so that the server can be crashed or down. One way to improve the quality of services will be increasingly large data is to divide the work load on the database server access in order that all requests fromusers can be served well. In this research design to optimize the performance of the database serverusing load balancing mechanisms clusters with MySQL Cluster technology. MySQL Cluster where there is a feature that can perform database replication and there is also a system that is able to cope with the failure of the database system itself.

***Keywords**: Load-balancing Cluster Server, Distribution Database, MySQLCluster.*

I. Pendahuluan

Komputer adalah bagian dari kehidupan kita yang tidak dapat kita pisahkan. Komputer merupakan suatu alat yang dapat membantu kita dalam menjalankan dan mempermudah tugas-tugas yang kita laksanakan. Komputer saat ini merupakan suatu alat yang multi fungsi. Komputer menjadi sebuah alat yang dapat memberikan kita informasi, komunikasi, multi media, pengolahan data, dan lain-lain.Saat ini komputer telah dijadikan teknologi pokok dalam pengolahan data dan penyajian informasi. Apalagi sejak aplikasi-aplikasi komputer berkembang dengan pesat, sehingga tercipta tehnik-tehnik penyajian informasi yang interaktif dan komunikatif.

penerapan *IT (information technologi-IT)* didalam suatu perusahaan atau organisasi adalah suatu hal keharusan yang harus difikirkan untuk dapat meningkatkan kinerja serta persaingan yang semakin kompetitif untuk mencapai suatu tujuan. Namun penerapan *IT* bukanlah suatu hal yang mudah karena mengingat selain mempersiapkan segala kebutuhan dengan matang dan juga membutuhkan biaya yang tidak sedikit.

Kehandalan pada sistem informasi dalam suatu lembaga atau perusahaan sangatlah penting untuk ditingkatkan dan dimaksimalkan agar investasi dan sumber daya yang terdapat di dalamnya dapat terolah dengan baik dan berkembang dan juga dapat menghemat biaya dan waktu. Layanan akan data juga ikut andil dalam menentukan kehandalan dalam pengaksesan sistem informasi. Data yang setiap waktunya akan terus bertambah dan semakin besar akan sangat menyulitkan apabila kita ingin menganalisa kesalahan yang ada dalam data tersebut. Untuk itu dibutuhkan suatu metode untuk mengurangi dan mengatasi pengontrolan data tersebut agar menghasilkan kinerja yang maksimal.

Berbagai cara dilakukan untuk memanage dan mengontrol dari mulai pengaksesan data, alur data, pengolahan data sampai dengan hasil yang akan dicapai. Pemilahan dan pengelompokkan data merupakan salah satu alternatif agar data yang semakin besar tersebut dapat terorganisir dengan baik.

Bina Sarana informatika merupakan lembaga pendidikan yang berbasiskan IT dalam menyediakan segala informasi bagi seluruh civitas akademik. Karena semakin bertambah dan banyaknya pengguna dalam hal pengaksesan informasi menimbulkan padatnya *trafficaccess* baik dalam sisi aplikasi terutama pada pengolahan data pada sisi *database.*

Terlebih lagi apabila pengaksesan aplikasi dilakukan secara *concurrent*dapat mengakibatkan *crash* baik pada sisi *server* aplikasi terutama *serverdatabase.*Bina Sarana Informatika menerapkan sebuah load balance pada dua buah web *server* dan dua buah *databaseserver*. Dan dari dua buah *databaseserver* tersebut diterapkan sebuah *failover* yang seakan akan *server* tersebut menjadi *single database server*. Yang apabila salah satu diantaranya terdapat gangguan dapat segera ditangani oleh *server* lain. Namun dari dua *database server* tersebut yang bekerja optimal hanya *serverprimery* saja sedangkan *server secondary* hanya menerima data dari proses yang dilakukan oleh *server primery* sehingga beban kerja yang diemban oleh *server primery* cukup berat dibandingkan *server secondary.* Maka dari itu untuk menjawab permasalahan ini maka penulis menerapkan suatu metode dan Salah satu metode yang digunakan dalam pengolahan data pada *serverdatabase* tersebut adalah menggunakan metode pengelompokkan data secara*Clustering*. *Load-balancing Clusters* merupakan salah satu metode pengelompokan data berdasarkan dari karakteristiknya dengan mendistribusikan beban pekerjaan secara merata melalui beberapa node yang bekerja di belakang (*back-end node*). Umumnya kluster ini akan dikonfigurasikan sedemikian rupa dengan beberapa *front-end load-balancing redundant*. Karena setiap elemen dalam sebuah *cluster load-balancing* menawarkan layanan penuh, maka dapat dikatakan bahwa komponen c*luster* tersebut merupakan sebuah *cluster* aktif /*cluster* HA aktif, yang bisa menerima semua permintaan yang diajukan oleh klien. Sehingga dapat diharapkan dengan adanya pengelompokan *clustering* tersebut kinerja pengolahan data yang dilakukan oleh *server database* tersebut dapat merata dan dapat terkontrol dengan baik dan yang dihasilkan juga dapat maksimal sehingga layanan akan data dalam sebuah sistem informasi dapat terpenuhi.

II. Kajian Literatur

2.1. Sistem informasi

Sistem Informasi dapat diartikan sebagai suatu alat untuk menyajikan informasi dengan cara sedemikian rupa sehingga bermanfaat bagi penerimanya. Tujuannya adalah untuk menyajikan informasi guna pengambilan keputusan para perencanaan, pemrakarsaan, pengorganisasian, pengendalian kegiatan operasi subsistem suatu perusahaan, dan menyajikan sinergi organisasi pada proses. (Al Fatta, 2007)

Sistem Informasi yang berbasis komputer yang biasa disebut Sistem Informasi Manajemen dalam suatu organisasi terdiri dari komponen-komponen sebagai berikut: (Kadir, 2003)

1. Perangkat Keras
   Adalah perangkat keras komponen untuk melengkapi kegiatan masukan, proses, dan keluaran data.
2. Perangkat Lunak
   Perangkat lunak yaitu program dan instruksi yang diberikan ke komputer.
3. Database
   Database adalah kumpulan dari data yang saling berhubungan satu dengan lainnya, tersimpan di perangkat keras komputer dan digunakan perangkat lunak untuk memanipulasinya.
4. Telekomunikasi
   Yaitu komunikasi yang menghubungkan antara pengguna sistem dengan sistem komputer secara bersama-sama ke dalam suatu jaringan kerja.
5. Manusia
   Manusia merupakan personel dari sistem informasi, meliputi manajer, analis, programmer, dan operator, serta bertanggung jawab terhadap perawatan sistem

2.2. *Database*

Basis data atau *database* adalah kumpulan data yang secara logik berkaitan dalam merepresentasikan fenomena atau fakta secara terstruktur dalam *domain* tertentu untuk mendukung aplikasi pada sistem tertentu. Basis data mendekripsikan *state* organisasi atau perusahaan atau sistem dan merupakan komponen utama sistem informasi karena semua informasi untuk pengambilan keputusan berasal dari data pada *database*.(Hariyanto, 2004)

Menurut pengaksesannya, *database* dapat dibedakan menjadi empat jenis, yaitu basis data individual, basis data perusahaan, basis data terdistribusi dan *bank* data publik. (Williams dan sawyer dalam Kadir, 2003) :

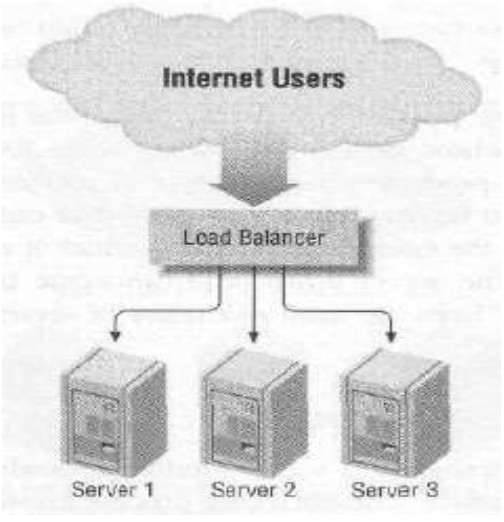
1. Basis data *Individual*
   Basis data individual adalah basis data yang digunakan oleh perseorangan. Biasanya basis data seperti ini banyak dijumpai dilingkupan personal komputer.
2. Basis data perusahaan
   Basis data perusahaan adalah basis data yang dimaksudkan untuk diakses oleh sejumlah pegawai dalam sebuah perusahaan dalam sebuah lokasi. Basis data seperti ini disimpan dalam sebuah server dan para

pemakai dapat mengakses dari masing-masing komputer yang berkedudukan sebagai *client*.

3. Basis data terdistribusi
   Adalah basis data yang disimpan pada sejumlah komputer yang terletak pada beberapa lokasi. Model seperti ini banyak digunakan pada perusahaan atau instansi yang memiliki sejumlah cabang yang berada diberbagai kota dan melayani transaksi data yang bersifat *online.*
4. *Bank* data publik
   Adalah jenis basis data yang dapat diakses oleh siapa saja (bersifat publik).

2.3. *Load Balancing*

Secara umum*load balancing* adalah pembagian beban kerja secara seimbang. Sedangkan *load balancing* dalam *computer internet working* adalah *Load balancing* adalah suatu metode untuk mendistribusikan beban kepada beberapa *host* sehingga beban kerja menjadi lebih ringan. Ini bertujuan agar waktu rata-rata mengerjakan tugas menjadi singkat dan dapat menaikkan utilitas prosesor.*Load balancing* adalah teknik untuk mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan *throughput*, memperkecil waktu tanggap dan menghindari*overload* pada salah satu jalur koneksi.



Gambar 1.*Server load balancing*
Sumber :O'Reilly & Associates, Inc

Algoritma umum yang digunakan pada standar kinerja *load balancing* adalah *round robin*, *threshold ratio*, *least connections*, *fastest response* dan *source*.

2.4. *Clustering*

Secara umum, salah satu karakteristik utama komputer *cluster* adalah konsep *single entity* dimana kumpulan banyak komputer yang menjadi komputer *cluster* dipandang sebagai satu kesatuan sistem tunggal.(Kamber, 2006) "*Cluster is collection of data objects that are similar to one another within the same cluster and are dissimilar to the object in the other cluster and the process of grouping a set of physical or abstract objects into classesof similar objects is called clustering ".*Suatu *clustering* merupakan suatu kelompok yang terdiri dari dua atau lebih yang ditugaskan secara khusus untuk menjalankan satu atau beberapa aplikasi yang dihubungkan dengan sedemikian rupa yang apabila terdapat kesalahan atau tidak berfungsi salah satu mesin, maka akan diambil alih atau digantikan oleh mesin yang lain secara otomatis.

1. *High Availability Clusters*

Tujuan dari *High-availability cluster* adalah untuk meningkatkan ketersediaan akan layanan yang disediakan oleh *cluster* tersebut. Pada umumnya tipe *clustering* ini sering disebut juga sebagai *Failover Cluster* . Umumnya *cluster* ini memiliki minimal dua *node* untuk melakukan *redudansi* data yang berguna untuk menghilangkan kegagalan disatu titik. HA Cluster dirancang untuk menjamin akses konstan untuk aplikasi layanan. Cluster ini dirancang untuk mempertahankan node berlebihan yang dapat bertindak sebagai sistem cadangan jika terjadi kegagalan. HA cluster bertujuan untuk memecahkan masalah yang timbul dari kegagalan mainframe dalam suatu perusahaan.

2. *Load Balancing Clusters*

Kategori *loadbalancing Clusters* jenis ini bekerja dengan cara melakukan proses penyampaian atau pendistribusian pembagian beban kerja dari data yang diproses secara merata melalui *node-node* yang bekerja berada di belakang *(back-end node*) sehingga semua operasi dapat berjalan dengan baik. Pada umumnya untuk dapat melakukan hal tersebut *cluster-cluster* yang termasuk dalam kategori ini dikonfigurasikan sedemikian rupa dengan beberapa *front-end load-balancing redundan*t. . Load-balancing cluster sangat berguna bagi mereka yang bekerja dengan anggaran TI yang terbatas. Mencurahkan beberapa node untuk mengelola alur kerja sebuah cluster memastikan bahwa kemampuan pemrosesan yang terbatas dapat dioptimalkan

3. *Compute Clusters*

Seringnya, penggunaan utama kluster komputer adalah untuk tujuan komputasi, ketimbang penanganan operasi yang berorientasi I/Oseperti layanan *Web* atau basis data. Contoh, sebuah *cluster* mungkin mendukung simulasi komputasional untuk perubahan cuaca atau tabrakan kendaraan.
Perbedaan utama untuk kategori ini dengan kategori lainnya adalah seberapa eratkah penggabungan antar *node*-nya. Sebagai contoh, sebuah tugas komputasi mungkin membutuhkan komunikasi yang
sering antar *node*--ini berarti bahwa kluster tersebut menggunakan sebuah jaringan terdedikasi yang sama, yang terletak di lokasi yang sangat berdekatan, dan mungkin juga merupakan *node-node* yang bersifat *homogen*. Desain kluster seperti ini, umumnya disebut juga sebagai *Beowulf Cluster*. Ada juga desain yang lain, yakni saat sebuah tugas komputasi hanya menggunakan satu atau beberapa *node* saja, dan membutuhkan komunikasi antar-*node* yang sangat sedikit atau tidak ada sama sekali.

Desain *cluster* ini, sering disebut sebagai "*Grid*". Beberapa *compute cluster* yang dihubungkan secara erat yang didesain sedemikian rupa, umumnya disebut dengan "*Supercomputing*". Beberapa perangkat lunak *Middleware* seperti MPI atau *Paraller Virtual Machine* (PVM) mengizinkan program *compute clustering* agar dapat dijalankan di dalam *cluster-cluster* tersebut.

2.5. *Database Cluster*

Database Clustering adalah kumpulan dari beberapa server yang berdiri sendiri dan kemudian bekerja sama sebagai suatu kesatuan sistem tunggal (Hodges, 2007).

1. *Shared Disk Clusters*

Arsitektur *shared disk clusters* menggunakan server-server *independent* dengan masukan *central input/output* (I/O) *devices* yang dapat diakses ke semua *node* di *cluster* dan berbagi sebuah sistem penyimpanan tunggal. Setiap server mempunyai prosesor dan memori sendiri, tetapi berbagi *disk resources*.Biasanya, arsitektur ini digunakan untuk berbagi penyimpanan *disk* untuk *file* dan *database*. Implementasi utama dari *shared-disk clustering* adalah bukan untuk *scalability. Shared-disk clustering* ini diimplementasikan untuk *availability* dan menambah *node* cadangan sebagai *failover node*

2. *Shared nothing cluster*

Sebuah arsitektur *cluster* Jaringan Komputer*shared nothing cluster*, tiap server dalam *cluster* menangani prosesor, memori, *storage, record locks* dan transaksi yang terpisah atau tidak memiliki node yang independen dan mandiri dan melakukan koordinasi dengan server lain melalui jaringan dengan menggunakan *high speed, low-latency interconnect technology* serta tidak menyediakan akses disk konkuren dari beberapa node, karena hanya satu node untuk perlu mengakses penyimpanan pada satu waktu.

2.6. *MySQL Cluster*

MySQL Cluster merupakan sebuah tipe basis data (database) yang dapat beroperasi dalam ukuran data yang relatif besar. *MySQL Cluster* adalah sebuah teknologi yang memungkinkan pengelompokan di memori *database* dalam sistem *shared-nothing*.

MySQL Cluster menggabungkan *MySQL Server* biasa dengan sebuah mesin penyimpanan *in-memory* terkluster yang dinamakan NDB. NDB berarti bagian dari suatu rangkaian yang dikhususkan sebagai mesin penyimpanan, sedangkan *MySQL Cluster* diartikan sebagai kombinasi atau gabungan dari *MySQL* dan mesin penyimpanan yang baru tersebut. NDB adalah sebuah mesin penyimpanan memory yang menawarkan ketersediaan yang tinggi dan fitur-fitur persistensi data. Mesin penyimpanan NDB dapat diatur dengan sebuah bidang *failover* dan pilihan-pilhan *load-balancing*, tetapi untuk memulai paling mudah dengan mesin penyimpanan pada level cluster. Sebuah *MySQL Cluster* terdiri dari sekumpulan komputer, masing-masing menjalankan sejumlah proses mencakup beberapa *MySQL server, node-node* penyimpanan untuk *cluster NDB*, *server-server* manajemen dan program-program pengakses data yang khusus. Sistem database ini terdiri dari beberapa *node* yang dapat didistribusikan ke beberapa perangkat keras dan ke beberapa wilayah/zona yang berbeda sekaligus untuk tetap menjaga ketersediaan data meskipun jaringan ataupun salah satu *node* sedang mengalami kegagalan.

III. Metode Penelitian

3.1. Studi Kepustakaan

Mempelajari literatur tentang teori dasar yang mendukung penelitian ini dengan bersumber pada buku mengenai jaringan dan *server,* jurnal-jurnal maupun artikel-artikel yang ada diinternet, catatan-catatan kuliah dan buku-buku lain yang ada kaitannya dengan mekanisme *load balancing clusters* dan *database server.*

3.2. Analisis dan Perancangan Sistem

Pada tahap ini dilakukan analisa kebutuhan sistem yang akan dibuat dan menjadi dasar untuk perancangan system.Dibutuhkannya pengetahuan tentang jaringanbaik perangkat keras maupun perangkat lunak khususnya untuk menentukan mekanisme sistem yang akan diterapkan pada *database server*dengan metode *load balancing clusters.*

3.3. Pengambilan data kinerja *databaseserver* sebelum penerapan mekanisme *load balancing clusters.*

Pada tahap pengambilan data ini, digunakanteknik pengukuran sebagai berikut:

1. Mengukur kinerja *databaseserver* berdasarkan beban rata-ratawaktu tunggu (*load average*).
2. Mengukur kinerja *databaseserver* berdasarkan penggunaan memori (memory usage) yang berjalan
3. Mengukur kinerja *databaseserver* berdasarkan jumlah processes yang diterima dari web *server* ke *databaseserver*

3.4. Implementasi

Pada tahapan ini dilakukan implementasi mekanisme *load balancing clusters*pada *databaseserver* sesuai dengan aspek-aspek yang telah dijelaskan sebelumnya. Selain itu, dilakukan analisis mengenai informasi-informasi yang didapatkan yang berhubungan dengan implementasi tersebut.

3.5 Pengambilan data kinerja *databaseserver* setelah penerapan mekanisme *load balancing clusters.*

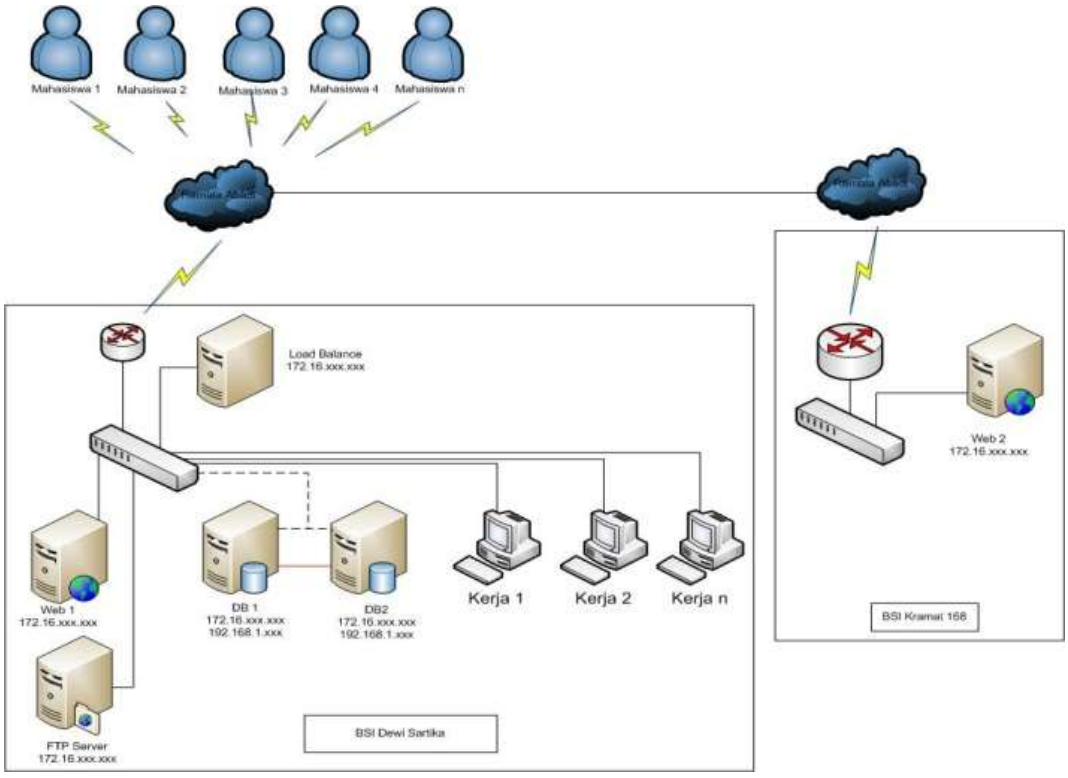
Pada tahap pengambilan data ini, digunakanteknik pengukuran sebagai berikut:

1. Mengukur kinerja *databaseserver* berdasarkan*MySQL Cluster Statistic*.
2. Mengukur kinerja *databaseserver* berdasarkan penggunaan memori (memory usage) yang berjalan
3. Mengukur kinerja *databaseserver* berdasarkan jumlah *Statistic Report* secara keseluruhan.

IV. Pembahasan

4.1. Analisis Jaringan Sebelum *Loadbalancing Clusters server*

Pada saat pelaksanaan Ujian secara Online, Bina Sarana Informatikamenggunakan *load balancing* pada *web server* dan *failover* pada *database server.* Namun belum menggunakan *database* yang terkluster pada sisi server. Berikut ini adalah gambar topologi jaringan Bina Sarana Informatika sebelum *loadbalancing cluster* diimplementasikan.

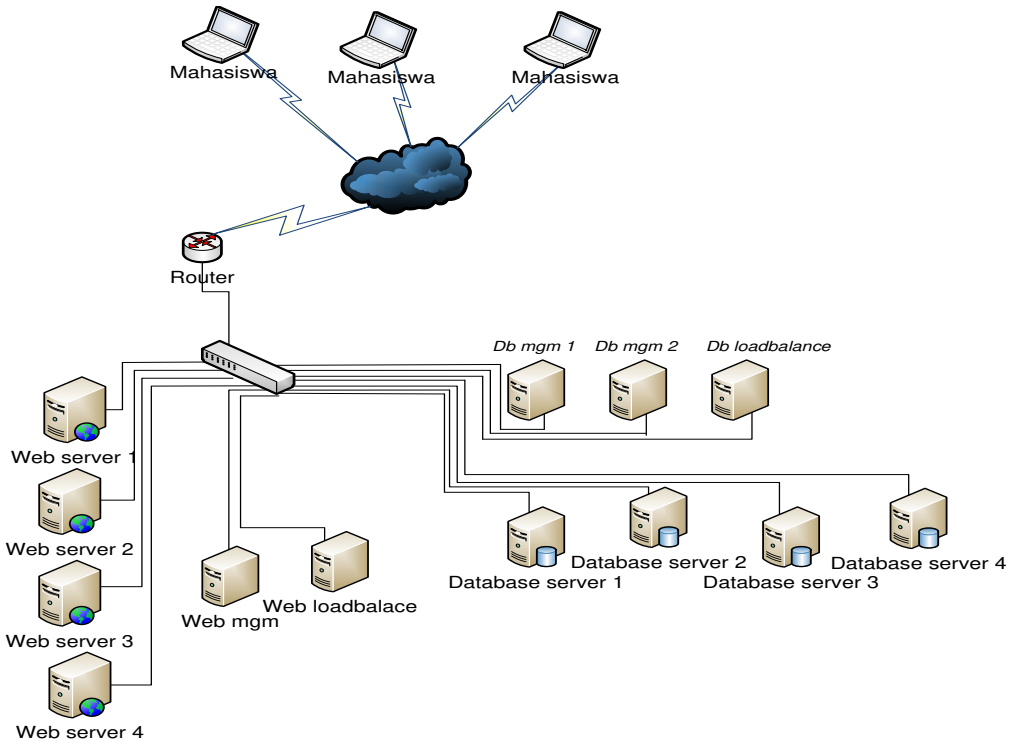


Gambar 2. Topologi Jaringan *Server*

Pada topologi ini mekanisme *service failover*tetap masih digunakan, namun penggunaannya *failover* hanya digunakan pada mesin *database*, hal ini dilakukan untuk memastikan ketersediaan *resource database* dari *content website* yang sudah dibuat.

4.2. Rancangan dan Prinsip Kerja*Load-BalancingClusters*

Berikut ini adalah gambar topologi jaringan Bina Sarana Informatika menggunakan *load balancing* pada *web server* dan *loadbalance clusters* pada *database server*:



Gambar 3. Topologi Jaringan *loadbalance clusters* pada *database server*

Pada gambar topologi diatas terlihat bahwa infrastruktur yang sudah diterapkan sebelumnya mengalami perubahan dan terdapat penambahan beberapa *server* untuk penggunaan baik pada *web server* maupun pada *databse server*. Dua buah *Server loadbalance* digunakan untuk melakukan *service* atau pelayanan terhadap user untuk mengakses data ke *web server* dan juga pengaksesan data dari *web server* ke *database server.*

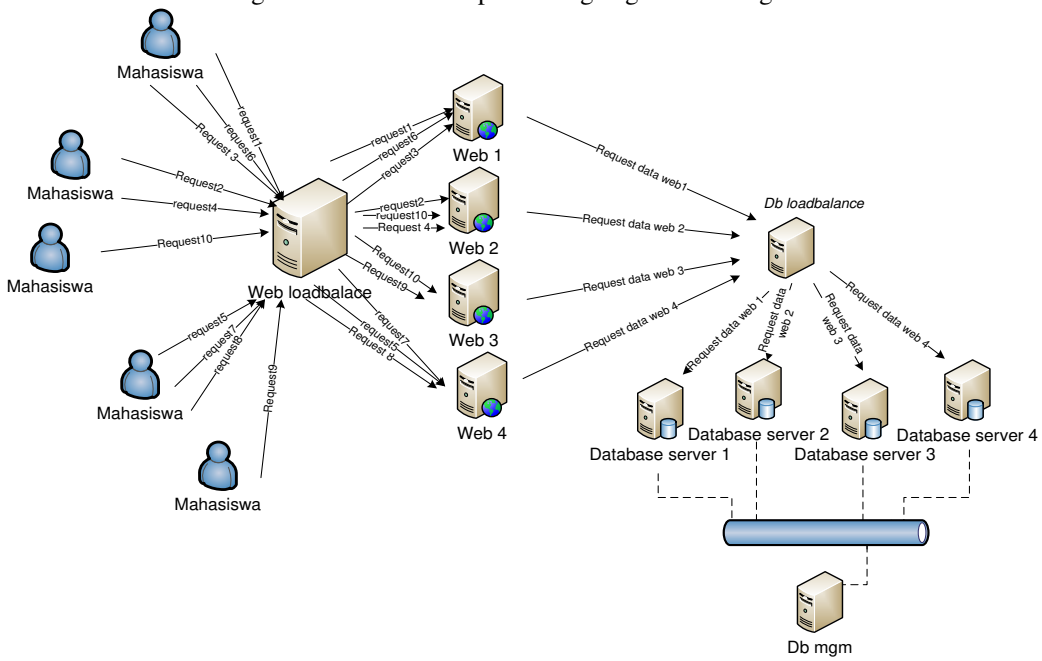
Penggunaan mekanisme *failover* pada *database server* dialihkan fungsinya dengan suatu mekanisme sistem yang baru yaitu *cluster*. Merujuk dari topologi diatas pada sisi *web server* terdapat penambahan perangkat (*device*) *server* sebanyak 2 buah perangkat *server* dari yang sudah ada. Dan pada sisi *database server* terdapat penambahan perangkat 2 buah mesin *database server,* 2 buah mesin *management server* dan 1 buah mesin *loadbalance.* Didalam topologi diatas penambahan perangkat *server* lebih diutamakan pada sisi *database server,* dimana *database server* menjadi unsur paling penting dalam setiap pelayanan yang akan dilakukan

Pada gambar topologi jaringan diatas, pengguna melakukan ujian secara *online* dengan mengakses *website* ujian. *Web Server* satu sampai dengan *web server* empat digunakan untuk menyimpan *content* isi dari *website* ujian *online.Server web Loadbalance* digunakan untuk membagi beban kerja atau permintaan yang datang dari pengguna agar beban kerja antara server tersebut menjadi seimbang. *Server web loadbalance* ini menggunakan *software Haproxy* yang akan melakukan pembagian beban berdasarkan banyaknya koneksi yang sedang dilayani oleh sebuah *server*. *Server* dengan pelayanan koneksi yang paling sedikit akan diberikan beban yang berikutnya akan masuk, dengan demikian beban *server* akan selalu seimbang.*Server web MGM* digunakan sebagai mesin *cluster* untuk *server-server web content* menggunakan *software glusterfs*yang digunakan sebagai mesin *cluster*, dimana seorang programmer tidak perlu memasukan

content web satu persatu ke mesin *web server*, akan tetapi cukup memasukan *content website* tersebut kedalam mesin *ftp server*, maka secara otomatis mesin *ftp* tersebut akan mereplikasi *content* tersebut ke empat mesin *web server* secara bersamaan.

Dari keempat *server-server web* tersebut, permintaan layanan akan diteruskan ke mesin-mesin *database server* dengan melewati mesin *database loadbalance* untuk pembagian beban kerja pada keempat mesin *database server*. Pada masing-masing *node database server* dikonfigurasi sedemikian rupa menggunakan *MySQL Cluster* dengan pengaturan pembagian *cluster*-nya dilakukan oleh mesin *Database MGM* (DBMGM). DBMGM difungsikan untuk melakukan kontrol aliran serta konektifitas dari *database*, selain itu juga digunakan untuk melakukan monitoring serta untuk mengetahui status kondisi dan juga nilai-nilai statistik kinerja dari masing-masing *database server* tersebut.

Sebagai ilustrasi untuk mengetahui bagaimana cara kerja *database server* dalam melayani peserta ujian *online* dijelaskan dengan gambar sebagai berikut:



Gambar 4. Mekanisme Kerja *Loadbalancing clusters*

Pada gambar mekanisme jaringan diatas, pengguna disini adalah mahasiswa, melakukan ujian secara *online* menggunakan fasilitas *wireless* yang yang telah disediakan dengan mengakses *website* ujian *online*. Jika seorang mahasiswa1 ingin membuka *website* ujian maka harus melakukan *request*1 kepada mesin *load balance*, Kemudian oleh *load balance request*1 tersebut dicatat sebagai *request*1 yang datang dari mahasiswa1 dan diarahkan kepada mesin *web server* 1. Selanjutnya *web server* 1 akan melakukan *request data web* 1 kepada mesin *load balance database,* kemudian oleh mesin *load balance database, request data web* 1 akan dicatat sebagai *request data web* 1 yang datang dari *web server* 1 atas *request* dari mahasiswa 1 dan diarahkan kepada mesin *database* 1 untuk dilakukan proses.

Selanjutnya ketika seorang mahasiswa2 ingin membuka sebuah *website* maka harus melakukan *request*2 kepada mesin *load balance*, oleh *load balance request*2 tersebut dicek terlebih dahulu, jika mahasiswa tersebut belum pernah melakukan *request*, maka *request*2 tersebut dicatat bahwa *request*2 datang dari mahasiswa2 dan diarahkan kepada mesin *web* 2 dan diteruskan kemesin *load balance databased*an diarahkan ke mesin *database* 2 untuk dilakukan proses.

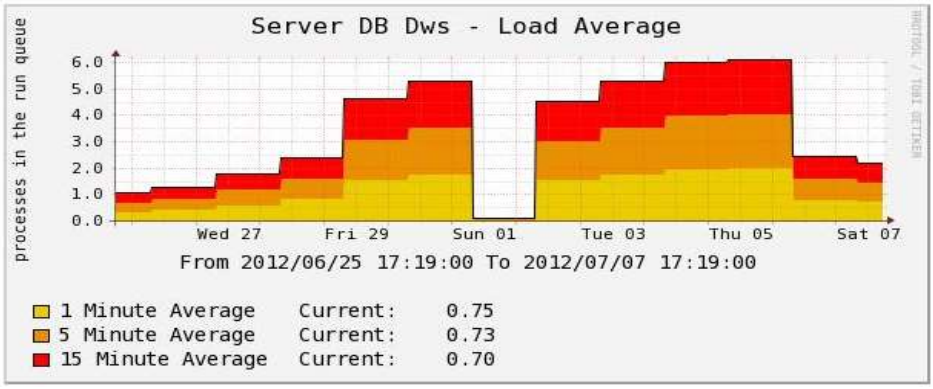
Selanjutnya ketika *request*3 di lakukan oleh mahasiswa 1, maka *load balance*akan mengecek terlebih dahulu, apakah mahasiwa tersebut pernah melakukan *request* atau belum. Jika pernah melakukan *request*, maka *request* tersebut akan diarahkan ke mesin *web* yang sama. Dalam hal ini mahasiswa1 melakukan *request*3, maka *load balance* akan mengecek apakah mahasiswa yg bersangkutan pernah melakukan *request* atau tidak, karena mahasiswa 1 pernah melakukan *request* dan di arahkan pada mesin *web*1, maka *request*3 pun di arahkan ke mesin *web*1, dan Selanjutnya *web server* 1 akan melakukan *request data web* 3 kepada mesin *load balance database,* kemudian oleh mesin *load balance database,*

request data web 3 akan dicatat sebagai *request data web* 3 yang datang dari *web server* 1 atas *request* dari mahasiswa 1 dan diarahkan kepada mesin *database* 1.

Begitupun selanjutnya, ketika *request*4 di lakukan oleh mahasiswa 2, maka *load balance* akan mengecek terlebih dahulu, apakah mahasiswa tersebut pernah melakukan *request* atau belum. Jika pernah melakukan *request*, maka *request* tersebut akan diarahkan ke mesin *web* yang sama. Dalam hal ini mahasiswa 2 melakukan *request* 4, maka *load balance* akan mengecek apakah mahasiswa yg bersangkutan pernah melakukan *request* atau tidak, karena mahasiswa 2 pernah melakukan *request* dan di arahkan pada mesin *web*2, maka *request*4 pun di arahkan ke mesin *web*2, dan Selanjutnya *web server* 2 akan melakukan *request data web* 4 kepada mesin *load balance database,* kemudian oleh mesin *load balance database, request data web* 4 akan dicatat sebagai *request data web* 4 yang datang dari *web server* 2 atas *request* dari mahasiswa 2 dan diarahkan kepada mesin *database* 2. Hal ini pun berlaku untuk *request-request* selanjutnya yang akan dilakukan oleh *user-user* lainnya yang akan mengakses *web* dan *database.*

4.3. Hasil Kinerja *Server* Sebelum Penerapan Mekanisme *Load Balancing Clusters*

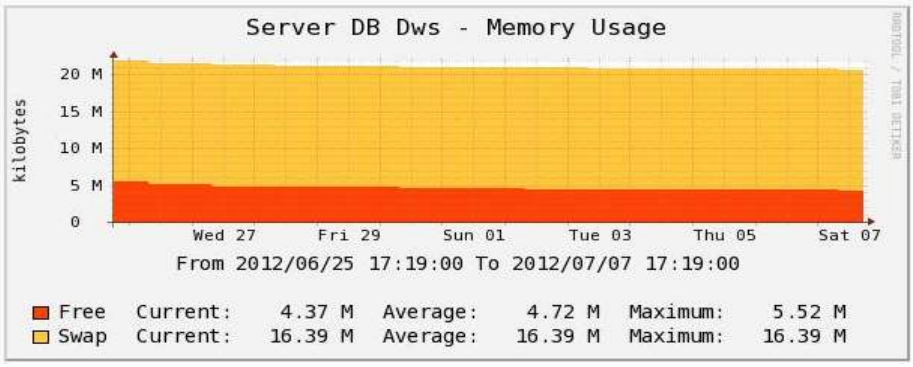
1. *Load Average*



Gambar 5.*Load Average* sebelum mekanisme *Load-balancing Clusters*

Dari grafik terlihat bahwa penggunaan waktu tunggu yang diperlukan seluruh mahasiswa untuk mengakses web pada saat ujian akhir semester setelah penerapan*failover* pada *database server* dengan nilai *load average* untuk 1 menit rata-rata adalah 0,75, untuk 5 menit rata-rata adalah 0,73 dan untuk 15 menit rata-rata adalah hanya 0,70. Kondisi ini masih cukup baik untuk jumlah kondisi data pada saat tersebut.

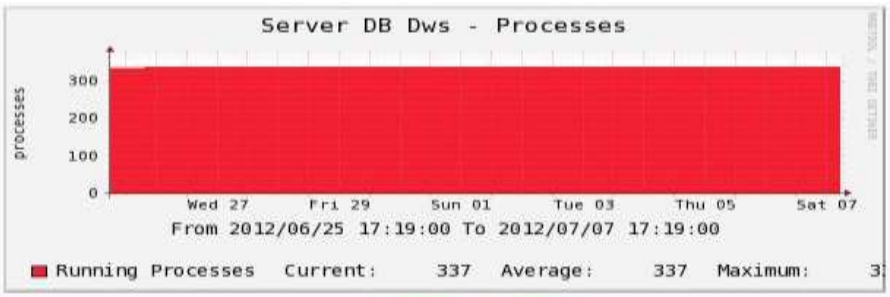
2. *MemoryUsage*



Gambar 6.*Memory Usagesebelum mekanisme Load-balancing Clusters*

Dari grafik terlihat bahwa penggunaan *memory swap* rata-rata 16,39 *Giga Byte,*sementara memori fisik yang dimiliki oleh *webserver* tersebut adalah 16*Giga Byte*, sehingga memori dapat berjalan di bawah dari kuota maksimum.

3. *Processes*

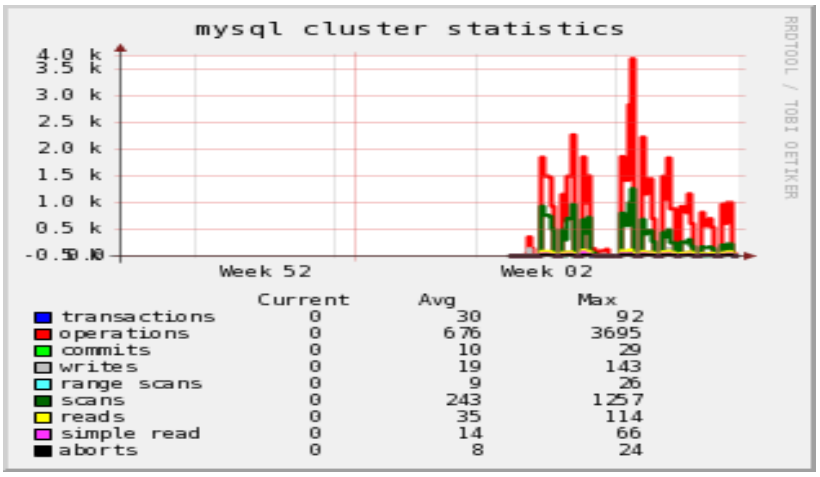


Gambar 7.*Processes*sebelum mekanisme *Load-balancing Clusters*

Dari grafik terlihat bahwa jumlah rata-rata *processes* yang dikerjakan oleh *database server* adalah 337 dengan maksimum *processes* adalah 337. Namun demikian, dalam hal ini masih menunjukkan bahwa *processes* tersebut sudah paling maksimum untuk jumlah pemrosessan data pada saat itu.

4.4. Hasil Kinerja *Server* Setelah Penerapan Mekanisme *Load Balancing Clusters*

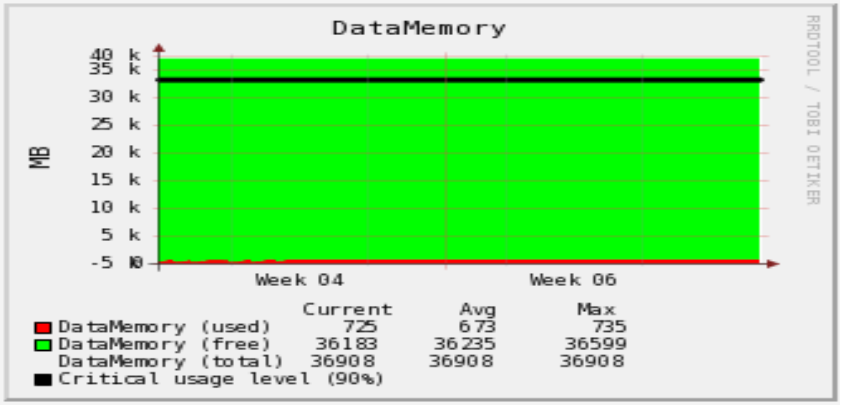
1. *MySQL Cluster Statistic*



Gambar 8. *MySQL Cluster Statistic*setelah mekanisme *Load-balancing Clusters*

Terlihat bahwa nilai rata-rata transaksi yang terjadi dengan nilai rata-ratanya adalah 30 dengan nilai maksimal 92. Begitu juga dengan nilai rata-rata *operation* dengan jumlah 676 dari nilai maksimalnya adalah 3695. Ini membuktikan bahwa arus transaksi yang terjadi pada mesin *database* sudah lebih baik dari sebelumnya.

2. *MemoryUsage*

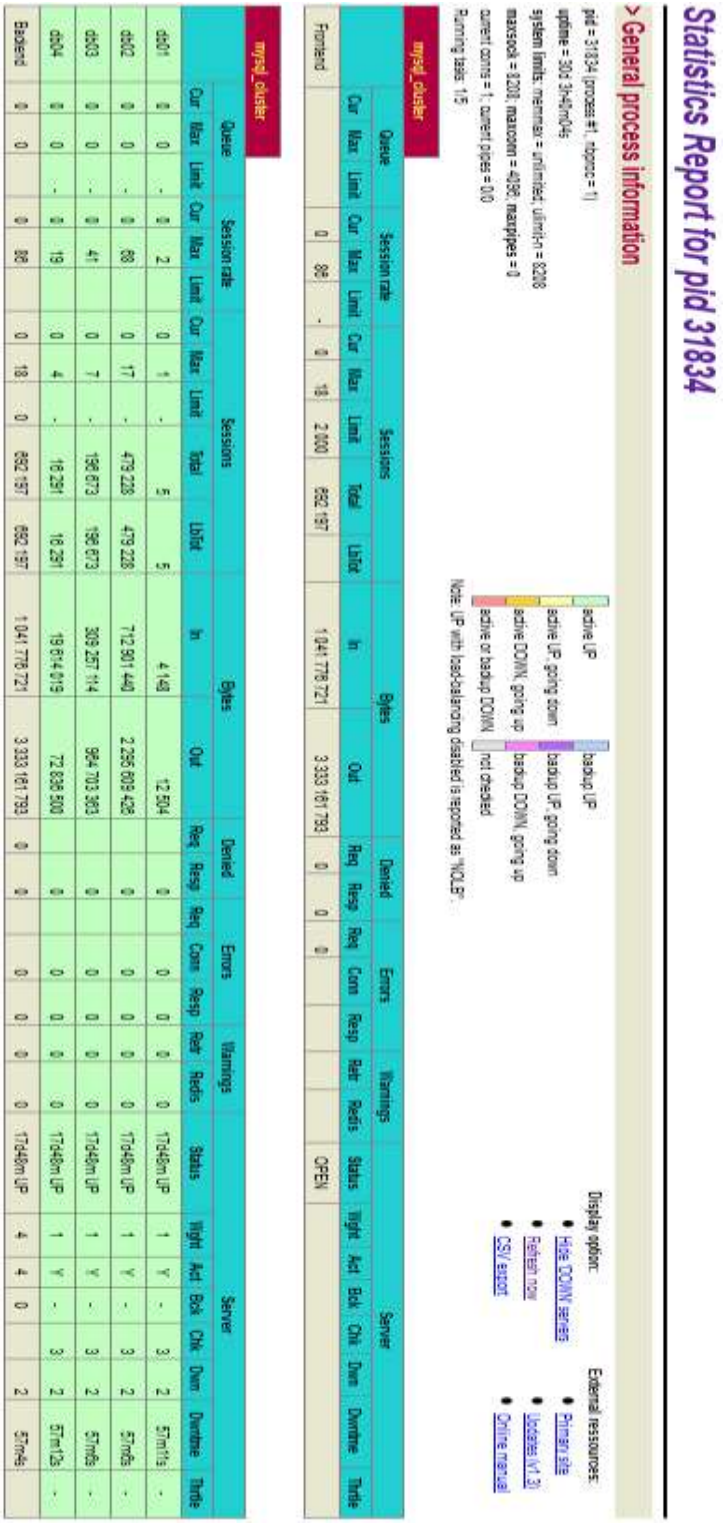


Gambar 9.*Memory Usage*setelah mekanisme *Load-balancing Clusters*

Dari grafik terlihat bahwa penggunaan *memory swap*yang terpakai cukup rendah. Hal ini dapat dilihat bahwa rata-rata penggunaan data memori tersebut adalah 673*Mega Byte*dari

nilai maksimal yang disediakan 735 *Mega Byte*, artinya memori berjalan masih dibawah kuota maksimum, sementara memori fisik yang dimiliki oleh*server* tersebut adalah16 Giga Byte. Berdasarkan kondisi tersebut maka dapat dikatakan bahwa kinerja *databaseserver* dengan *IP address* 172.16.xxx.xxx menjadi lebih baik jika dibandingkan dengan kinerja *databaseserver* sebelumnya.

3. *Statistic Report* Keseluruhan



Gambar 10.*Statistic Report setelah mekanisme Load-balancing Clusters*

Terlihat pada gambar statistik tersebut diatas bahwa semua *serverdatabase* dapat melayani seluruh permintaan atau *request* dari peserta yang mengakses web ujian. Dari gambar tersebut juga dapat dilihat ke empat *serverdatabase* bekerja dengan baik.

Berikut ini akan ditampilkan tabel perbedaan dari infrastruktur yang sebelum menggunakan *loadbalancing clusters* dan sesudah menggunakan *loadbalancing clusters* pada *database server* dengan tolak ukur berdasarkan hasil statistik yang didapatkan seperti diatas

Tabel 1. Sebelum dan sesudah pemakaian *loadbalancing clusters* pada *server*

Sebelum mengunakan *loadbalancing Clusters server*	**Sesudah mengunakan *loadbalancing Clusters server***
1. Database server masih bersifat *single database*	1. Database server sudah menggunakan *Database* yang terkluster yang bersifat *multi server*
2. *Resource hardware* yang tersedia masih sedikit dan kesulitan jika akan dilakukan penambahan	2. *Resource hardware* yang disediakan cukup besar dan mudah jika ingin dilakukan penambahan
3. Jika terjadi proses data yang sedikit besar server mudah *crash* dan backup hanya satu buah server	3. Jika terjadi proses data yang server tidak mudah *crash* dan backup bisa lebih dari satu buah server dan dapat ditambah
4. Dari hasil statistik yang didapat, performa server masih dikatakan cukup baik bila proses data yang sedikit	4. Performa server masih stabil apabila proses data yang terjadi cukup banyak.
5. *Server database* menggunakan aplikasi *failover* dengan dua buah server untuk proses pengolahan dan backup data	5. *Server database* menggunakan aplikasi *High Availability Clusters* dengan lebih dari dua buah server untuk proses pengolahan dan backup data dan kemudahan untuk ditambahkan sesuai dengan kebutuhan
6. Pemrosesan data hanya dilakukan pada satu buah server (*single server)*	6. Pemrosesan data dapat dilakukan pada banyak server

Berdasarkan hasil pengujian diatas mengenai rancangan dan optimalisasi kinerja *databaseserver* menggunakan metode *loadbalancing clusters*, grafik serta tabel hasil pengukuran statistik pada kinerja *server database*, dapat disimpulkan bahwa :

1. Pada pengamatan menggunakan *clusters statistic* dapat dianalisa bahwa ketika sebelum menerapkan mekanisme *clusters,* penggunaan kinerja *server database* bekerja dengan hampir batas maksimum dengan jumlah pemrosesan data yang belum sebanyak pada saat penerapan mekanisme *cluster* pada *databaseserver*. Dengan pengolahan data yang lebih banyak dari sebelumnya dengan menerapkan metode *clusters* bekerja dengan masih jauh dibawah batas maksimum, yang artinya bahwa seluruh pelayanan akan permintaan data dapat ditangani dengan baik.
2. Pada pengamatan grafik dan tabel data merupakan data yang diambil secara keseluruhan dengan perincian per-tanggal dan total keseluruhan. Terlihat bahwa jumlah dari keseluruhan data yang disediakan secara umum hampir sebanding dengan data yang masuk. Adanya perbedaan jumlah data yang masuk dikarena adanya banyak faktor yang menentukan dari timbulnya perbedaan tersebut, kebanyakan faktor dari luar seperti dari sisi peserta yang ikut ujian dikarenakan tidak hadir
atau telat, faktor alam (hujan dan banjir), kondisi jaringan, serta faktor-faktor lain yang menentukan bisa atau tidaknya akses ke *server*, diantaranya :
   a. Kondisi komputer *client* jika terdapat *virus, spyware, adware* atau *trojan* dapat menyebabkan lambatnya koneksi pada jaringan, karena tanpa diketahui program perusak tersebut dapat menggunakan koneksi internet dan mengakibatkan jaringan internet menjadi lambat.
   b. Adanya gangguan dari sisi *provider* penyedia *internet* (ISP) sehingga terjadi gangguan bahkan terputusnya jaringan secara masal dalam suatu area atau kampus sehingga mengakibatkan lambatnya atau terputusnya jaringan ke *server* sehingga permintaan ataupun *request*

dari peserta tidak dapat dilayani dengan baik.

V. Kesimpulan

Berdasarkan hasil yang didapatkan dari pembahasan yang telah dilakukan dalam penelitian ini maka didapatkan kesimpulan Perancangan optimalisasi *database* yang diterapkan kepada *database server* menggunakan mekanisme *loadbalancing clusters* ini yaitu dengan mengelompokkan paket data yang masuk berdasarkan dari permintaan atau request dari pengguna melalui *web*. *Request* yang datang dari pengguna yang masuk melewati pengalamatan ke mesin *loadbalance* pada sisi *web server* akan diarahkan ke *node-node* web *server* masing-masing dan *request* tersebut akan diteruskan ke *node-node database server* melawati mesin *loadbalance database* sebagai pembagi beban dan diatur oleh *management server* (MGM *server*) untuk melakukan pengontrolan permintaan serta proses yang dilakukan pada masing-masing *node database*. Sehingga kinerja dapat maksimal dan pemrosesan dapat dilakukan dengan cepat. Berdasarkan dari hasil analisa yang telah digambarkan dengan grafik dan tabel data setelah penerapan mekanisme *loadbalancing clusters* pada *database server* diatas,terlihat bahwa mekanisme tersebut terbukti berjalan dengan baik untuk melayani kebutuhan data yang besar diluar dari faktor-faktor lain yang mempengaruhinya..

Daftar Pustaka

Al Fatta, Hanif. 2007. Analisis dan Perancangan Sistem Informasi: Andi Offset. Yogyakarta.

Bourke, Tony. 2001. "*Server Load Balancing*". O'Reilly & Associaties, Inc : United States of America.

Han, Jiawei dan Kamber, Micheline. 2006. Data Mining Concept and Techniques, Second Edition. Morgan kaufmann : San Francisco. ISBN: 978-1-55860-3-384-451.

Hariyanto, Bambang. 2004. Sistem Manajemen Basis Data. Informatika : Bandung.

Hodges, R. 2007. Database High Availability and Scalability. CTO Continuent, Inc.

Introducing Severalnines ClusterControl. 2011. White Paper by Severalnines AB. http://www.severalnines.com/sites/default/files/docs/ (diakses pada 03 Januari 2013).

Kadir, Abdul. 2003. Pengenalan Sistem Informasi. Andi Offset : Yogyakarta.

MySQL Cluster 7.0 & 7.1: Architecture and New Features. 2010. A MySQL Technical White Paper by Oracle.