USULAN PENELITIAN MULA

BIDANG KELEMBAGAAN DAN PENGEMBANGAN SISTEM

IMPLEMENTASI *HIGH AVAILABILITY* PADA *DATABASE* (STUDI KASUS UNIVERSITAS TERBUKA)

Oleh:

Harris Rovandi, S.Kom

M. Novan Billiranto, S.Kom

PUSAT KOMPUTER

UNIVERSITAS TERBUKA

2014

LEMBAR PENGESAHAN
USULAN PENELITIAN MULA BIDANG ILMU KELEMBAGAAN DAN PENGEMBANGAN SISTEM INFORMASI
UNIVERSITAS TERBUKA

1.	a.	Judul Penelitian	:	Implementasi *High Availability* pada *Database*
	b.	Bidang Penelitian	:	Kelembagaan dan Pengembangan Sistem
	c.	Klasifikasi Penelitian	:	Penelitian Mula
2.	Ketua Peneliti			
	a.	Nama	:	Harris Rovandi, S.Kom
	b.	NIP	:	19780120 200312 1 003
	c.	Golongan	:	III
	d.	Jabatan Akademik	:	-
	e.	Program Studi	:	-
3.	Anggota Peneliti			
	a.	Jumlah Anggota	:	1 (satu) orang
	b.	Nama Anggota	:	M. Novan Billiranto, S.Kom
	c.	Program Studi	:	-
4.	a.	Periode Penelitian	:	April – September 2014
	b.	Lama Penelitian	:	6 (enam) bulan
5	Biaya Penelitian		:	Rp. 10.000.000,- (Sepuluh Juta Rupiah)
6.	Sumber Biaya		:	UT
7.	Pemanfaatan Hasil Pelatihan			

Mengetahui,		Ketua Peneliti,
Kepala Pusat Komputer		
Dra.Dyah Paminta Rahayu		Harris Rovandi
NIP. 19641208 199103 2 001		NIP. 19780120 200312 1 003
Menyetujui,		Menyetujui,
Ketua LPPM – UT		Kepala Pusat Keilmuan
Ir. Kristanti Ambar Puspitasari, M.Ed., Ph.D.		Dr. Herman, M.A.
NIP. 19610212 198603 2 001		NIP. 19560525 198603 1 004

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

RINGKASAN

Database adalah suatu kumpulan data yang disimpan secara sistematis di dalam komputer dan dapat dimanipulasi menggunakan aplikasi, Teknologi tersebut memungkinkan kita untuk mengolah data tanpa harus menggunakan waktu yang lama Universitas Terbuka sebagai institusi yang bergerak dibidang pendidikan telah menginvestasikan banyak tenaga, waktu, dan biaya dalam melakukan pengembangan dan pemeliharaan *database*. Tingginya kebutuhan dalam penggunaan *database* membuat tuntutan akan kebutuhan penyediakan basis data (*database*) yang dapat diakses terus secara aktif bekerja selama 24 jam x 7 hari. Tujuan dari penelitian ini untuk menghasilkan solusi database *High Availability*, *flexibility*, dan *scalability* sehingga dapat meminimalkan dampak gangguan pada *database*.Metode yang digunakan dalam penulisan ini adalah metode *research and development model* dengan pengujian yang meliputi *installation tes, basic sanity test, cluster fault test*, *load test, scalability test* dan *stability test*. Hasil yang dicapai pada penelitian ini adalah implementasi database *high availability* dengan *flexibility*, *scalability* dan kekurangannya. Dari implementasi yang dilakukan dapat disimpulkan Penerapan aplikasi Percona XtraDB Cluster sebagai solusi *High Avalability* database dapat meningkatkan ketersediaan database hingga 99% tanpa *down time.*

Kata Kunci:

Database, High Availability.

BAB 1. PENDAHULUAN

1.1. Latar Belakang

Di zaman sekarang ini banyak sekali teknologi yang sudah berkembang khususnya di bidang data (*database*). *Database* adalah suatu kumpulan data yang disimpan secara sistematis di dalam komputer dan dapat dimanipulasi menggunakan aplikasi, teknologi ini memungkinkan kita untuk mengolah data tanpa harus menggunakan waktu yang lama. Penerapan teknologi ini akan meningkatkan proses kerja menjadi lebih efektif dan efisien. Akan tetapi dengan penggunaan database perusahaan akan dihadapkan pada resiko terjadinya gangguan pada *database* seperti terjadinya kehilangan data (*data loss*) dan juga data yang tidak dapat diakses (*downtime*).

Universitas Terbuka sebagai institusi yang bergerak dibidang pendidikan telah menginvestasikan banyak tenaga, waktu, dan biaya dalam melakukan pengembangan dan pemeliharaan *database*. Terdapat kurang lebih 23 database yang sudah berjalan sampai dengan saat ini (April 2014). Tingginya kebutuhan dalam penggunaan *database* membutuhkan investasi biaya yang besar dalam pembelian *software* dengan diiringi tuntutan akan kebutuhan penyediakan basis data (*database*) yang dapat diakses terus secara aktif bekerja selama 24 jam x 7 hari. Proses pemeliharaan data *backup* yang selama ini dijalankan dalam menyelesaikan masalah *downtime* belum memberikan solusi yang optimal, dikarenakan penggunaan *database* yang terdistribusi pada satu *single node* membutuhkan waktu yang lama dalam prosess pengembalian data (*recovery*).

Penerapan *High Availability* (HA) pada basis data adalah solusi jawaban dalam mengatasi masalah *downtime* yang terjadi pada Universitas Terbuka, *High Availability* (HA) arsitektur merupakan suatu pendekatan yang mendefinisikan komponen, modul atau pelaksanaan layanan sistem yang menjamin kinerja operasional yang optimal, bahkan pada saat beban tinggi. (Lewis, 1999) mengatakan *High availability* menjadi hal yang semakin penting karena kita sudah sangat tergantung kepada komputer di dalam kegiatan bisnis. Solusi penerapan *High avalaibility* dapat diterapkan dengan biaya yang murah tanpa menggunakan perangkat lunak yang mahal. Walaupun hubungan antara *high availability* dan biaya yang selalu bersifat linier (Marcus, 2003).

1.2. Rumusan Masalah

Dalam penelitian ini penulis mencoba merumuskan persoalan dalam bentuk pertanyaan :

1. Apakah Universitas Terbuka memerlukan penerapan teknologi *High Availability*?
2. Apakah keuntungan Universitas Terbuka menerapkan teknologi *High Availability*?
3. Apakah *database* pada lingkungan Universitas Terbuka mendukung penggunaan Sistem *High Availability* ?

1.3. Tujuan Penelitian

Secara umum tujuan penelitian ini untuk menghasilkan solusi *High Availability*, *flexibility*, dan *scalability* sehingga dapat meminimalkan dampak gangguan pada *database.* Namun Secara khusus penelitian ini bertujuan :

1. Menganalisis data hasil kajian untuk menghasilkan konfigurasi *database* sehingga layanan *High Availability*, *flexibility*, dan *scalability* dapat tercapai.
2. Merancang model *High Availability*, *flexibility*, dan *scalability* pada *database*.

1.4. Manfaat Penelitian

Manfaat yang diperoleh dari penelitian ini adalah :

1. Meminimalkan dampak negatif dari gangguan dan *downtime* pada *database server*.
2. Memelihara stabilitas dan integrasi data selama proses *recovery server database*.
3. Mencegah terjadinya kehilangan data pada saat *database* server mengalami kerusakan secara fisik.
4. Dapat memberi masukan, saran dan dasar pertimbangan kepada Universitas Terbuka (UT), khususnya Pusat Komputer untuk meningkatkan kualitas dari Sistem *database*.
5. Diperoleh konfigurasi dan metode *clustering* yang optimal untuk Sistem *database*.

1.5 Ruang Lingkup Penelitian

Ruang lingkup penelitian meliputi batasan pada :

1. Definisi penelitian *High Availability* pada penelitian ini adalah database *cluster* ditambah dengan penggunaan *load balancer*.
2. Sistem Basis Data (*Database*) yang digunakan pada sistem *cluster* adalah Percona
3. Perangkat lunak yang digunakan sebagai *load balance* pada penelitian ini adalah HA Proxy.
4. Sistem *cluster* yang digunakan menggunakan varian Galera.

BAB 2. TINJAUAN PUSTAKA

2.1 *Database*

Menurut Chendramata (2009), *Database* adalah sebuah perangkat lunak yang dirancang dan diperuntukkan sebagai media untuk menyimpan data-data transaksi yang dihasilkan pada sebuah proses bisnis. *Database* minimal terdiri dari satu file yang cukup untuk dimanipulasi oleh komputer sedemikian rupa. Sedangkan menurut Nugroho (2005), *Database* adalah sebuah bentuk media yang digunakan untuk menyimpan sebuah data. *Database* dapat diilustrasikan seperti rumah atau gudang yang dijadikan tempat menyimpan berbagai macam barang (data).

Dari kedua pengertian tersebut dapat disimpulkan bahwa *database* merupakan perangkat lunak yang digunakan sebagai penyimpan data.

2.1.1 *Database* Terpusat

Database terpusat adalah *database* yang terletak dan dipelihara dalam satu lokasi, Salah satu keuntungan utama adalah bahwa semua data berada di satu tempat. Kelemahannya adalah kemacetan (*bottlenecks*) dapat terjadi.

2.1.2 *Database* Terdistribusi

Database terdistribusi bisa diartikan sebagai kumpulan dari data-data dengan berbagai bagian yang ditangani DBMS (Data Base Management System) secara terpisah dan berjalan pada sistem komputer. Pada *database* terdistribusi (distributed *database*), data disimpan pada beberapa tempat (site), memungkinkan transaksi untuk diproses pada banyak mesin, bukannya terbatas pada satu mesin. Berikut adalah proses pada *database* terdistribusi :

- Replikasi : Mencari data yang berubah pada *database* dengan menggunakan perangkat lunak tertentu. Setelah data yang berubah teridentifikasi maka proses replikasi akan membuat semua *database* menjadi sama. Proses replikasi dapat menjadi komplek, memakan waktu, dan sumber daya komputer berdasarkan ukuran dan jumlah *database*.
- Duplikasi : Mengidentifikasi satu *database* sebagai master dan kemudian menduplikat *database* tersebut. Proses duplikasi biasanya dilakukan pada waktu yang telah ditetapkan. Hal ini untuk memastikan bahwa disetiap lokasi yang didistribusikan memiliki data yang sama. Dalam duplikasi, pengguna hanya dapat merubah data pada *database* master.

2.1.3 *Database* Cluster

Database clustering adalah kumpulan dari beberapa server yang berdiri sendiri yang kemudian bekerja sama sebagai suatu sistem tunggal (Hodges,2007). Saat ini aplikasi *database* semakin berkembang, baik dalam hal kegunaan, ukuran, maupun kompleksitas. Hal ini secara langsung berdampak pada server *database* sebagai penyedia layanan terhadap akses *database*, konsekuensi dari semua itu adalah beban *database* server akan semakin bertambah berat dan mengakibatkan kurang optimalnya kinerja dari server tersebut. Oleh karena itu diperlukan perancangan yang tepat dan handal dalam membangun *database* server.

2.2 *High Available* (HA)

High Available (HA) adalah system cluter yang terdiri dari dua atau lebih mesin server atau yang biasa dikenal dengan node (Jagannatha, 1999). Setiap node pada cluster saling terhubung pada suatu network, setiap node pada cluster berkomunikasi dan menyampaikan informasi konek melalui heart-beat. Cluster juga memiliki penyimpanan (storage) untuk menyimpan data dan data tersebut terhubung pada jaringan public.

Sebuah sistem cluster dapat menahan kegagalan perangkat keran dan perangkat lunak pada salah satu node dengan waktu *downtime* yang sangat kecil. Cluster di konfigurasi untuk mendukung high availability dan up time pada server. HA cluster terdiri dari beberapa komponen hardware dan software yang rumit. Dimana komponen software nya adalah operating system, volume manager, dan database software.

2.3 Percona XtraDB Cluster

Percona XtraDB Cluster adalah solusi *open source* untuk sebuah sistem database cluster MySQL yang bersifat aktif/aktif dalam meningkatkan kinerja *high availability* dan high scalability sebuah database. Percona XtraDB Cluster juga sudah mengintegrasikan Percona Server dan Percona XtraBackup pada library Galera MySQL. Percona XtraDB cluster merupakan solusi dalam satu paket yang memungkinkan database menjadi sebuah system cluster yang high availability dengan biaya yang murah. Percona XtraDB Cluster telah di download lebih dari 150.000 sejak awal kali diluncurkan pada April 2012.

2.4 Load Balancing

Secara umum pengertian *load balancing* adalah pembagian seimbang. Sedangkan pada computer *load balancing* adalah proses distribusi beban terhadap sebuah service yang ada pada sekumpulan server atau perangkat jaringan.

Terdapat beberapa alas an utama menggunakan *Load Balancing* pada website atau aplikasi berbasis web, diantaranya adalah:

1. Waktu respons, Salah satu manfaat terbesar adalah untuk meningkatkan kecepatan akses website saat dibuka. Dengan dua atau lebih server yang saling berbagi beban lalu lintas web, masing-masing akan berjalan lebih cepat karena beban tidak berada pada 1 server saja.
2. Redudansi, Dengan *load balancing*, akan mewarisi sedikit redudansi sebagai contoh, jika website kita berjalan seimbang di 3 server dan salah satu *server* bermasalah, maka dua server yang lain akan dapat terus berjalan.

2.5 HA Proxy

HAProxy adalah software open source TCP/HTTP yang berfungsi sebagai penyeimbang beban atau yang lebih dikenal dengan istilah (*load balancer*), biasanya digunakan untuk meningkatkan kinerja situs web dan *service* dengan menyebarkan permintaan dari user ke beberapa server . Namanya adalah singkatan dari *High Availability* Proxy. Software ubu ini ditulis dalam bahasa C dan memiliki reputasi untuk menjadi cepat, efisien (dalam penggunaan prosesor dan memori) dan stabil.

HAProxy digunakan oleh sejumlah situs high-profile termasuk Stack Overflow, Reddit, Tumblr, dan Twitter dan digunakan dalam produk OpsWorks dari Amazon Web Services.

BAB 3. METODE DAN PERANCANGAN PENELITIAN

Bab ini berisi penjelasan-penjelasan mengenai metodelogi penelitian, tahapan penelitian, waktu tempat penelitian yang akan dilakukan, serta penjelasan mengenai Teknik dan Interpretasi hasil penelitian ini.

3.1 Metode Penelitian

Dilihat dari sifatnya penelitian ini termasuk kedalam metode penelitian pengembangan (*research and development model*) untuk menghasilkan sebuah Model *High Availability, flexibility,* dan *scalability* pada sistem *Database* di Universiatas Terbuka.

3.2 Langkah–langkah Penelitian

Tahap yang akan dilakukan dalam membuat penelitian ini adalah sebagai berikut:

1. Tahap persiapan yang meliputi:
   - Mengidentifikasikan kebutuhan.
   - Perumusan masalah.
   - melakukan kajian literatur.
2. Tahap Pengumpulan dan Pengelolahan data.
   - Perancangan arsitektur *High Availability* pada sistem *database*
   - Simulasi pengujian *High Availability* dengan menggunakan beberapa metodelogi diantara lain adalah: *Installation test*, *basic sanity test*, *cluster fault injection test*, *Load test, scalability test*, dan *stability test*.
3. Tahap Laporan.
   - Pembuatan laporan hasil dari penelitian
   - Rekomendasi berdasarkan penelitian ini

3.3 Subjek Penelitian

Subjek pada penelitian ini adalah rancangan *high availability* database yang akan melewati tahapan simulasi pengujian hingga pada akhirnya dapat di terapkan dan digunakan pada operasional Univeritas Terbuka.

3.4 Tempat Penelitian dan Waktu Penelitian

Penelitian ini akan dilakukan di Pusat Komputer khusunya pada ruang pengembangan Universitas Terbuka, Jalan Cabe Raya, Pamulang, Tangerang Selatan 15418, Telp. 021-7490941

3.5 Waktu Penelitian

No	Jenis Kegiatan	Bulan Ke					
		3	4	5	6	7	8
1	Persiapan						
2	Perancangan arsitektur HA						
3	Simulasi arsitektur HA						
4	Analisis arsitektur HA						
5	Penyusunan Laporan						
6	Publikasi Ilmiah						

3.6 Teknik Pengumpulan Data

Untuk melakukan simulasi pengujian atau pengumulan data pada *High Availability database* digunakan beberapa metodelogi pengujian diantaranya adalah : *Installation test*, *basic sanity test*, *cluster fault injection test*, *Load test*, *Configuration test* dan *stability test*.

a. *Installation test*

   Instalation test adalah pengujian yang dilakukan untuk memastikan bahwa produk percona DB dengan mudah diinstal pada server.

b. *Basic sanity test*

   Basic Sanity test adalah pengujian untuk memastikan bahwa produk percona DB secara mendasar telah memenuhi fungsi-fungsi dasar dari produknya.

c. *Cluster fault injection test*

   Pengujian yang dilakukan untuk memastikan proses yang akan berlangsung jika terjadi kesalahan pada *hardware* atau *software* di salah satu *node database*.

d. *Load test*

   Melakukan pengujian untuk memastikan tingkah laku cluster jika dibebani dengan beban kerja yang berat.

e. *Scalability test*

Scalability test dilakukan untuk mengethui performa cluster pada sistem database.

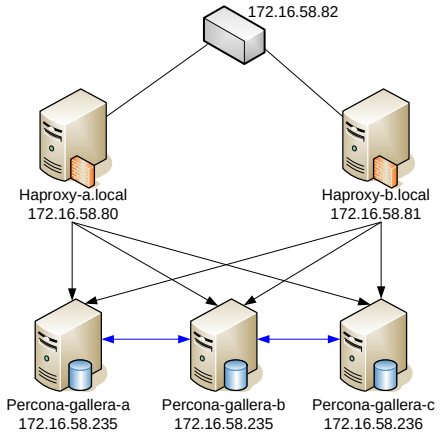
f. *Stability test*

Melakukan pengecekan stabilitas system dengan mengan menjalankan sistem dalam waktu tertentu.

3.7 Perancangan Arsitektur High Availability Database

Perancangan arsitektur *High availabity* pada penelitian ini dijalankan pada server virtualisasi yang terdiri dari 3 server database dan 2 Server HA Proxy sebagai load balancer. Adapun gambaran rancangan arsitektur HA yang akan diimplementasi adalah sebagai berikut:

Node /IP	Clone IP	Fungsi	Port yang digunakan	Keterangan
1 / 58.22	58.235	Node-a-1	3306	DBCluster Master-master Pecona. Cluster Registrasi
	58.236	Node-a-2	3306	
2 / 58.23	58.237	Node-a-3	3306	
	58.80	Haproxy-a	3006,80	Master HA-Proxy
	58.81	Haproxy-b	3006,80	Slave HA-Proxy
	58.82		3006,80	IP Main HA-Proxy



Gambar 1 Arsitektur yang digunakan pada Penelitian

3.8 Analisis Data serta Interpretasi Hasil Analisis

Untuk analisis data digunakan data yang didapat dari hasil pengujian berdasarkan pada metodelogi dan teknik pengumpulan data. Proses pengumpulan data dibantu dengan menggunakan aplikasi monitoring database yang kemudian akan dianalisa untuk menjadi sebuah informasi/kesimpulan.dari hasil pengujian.

BAB 4. HASIL DAN EVALUASI PENELITIAN

Bab ini berisi mengenai hasil pengujian yang dilakukan dari implementasi *High availability database* berdasarkan metode-metode yang sudah dijelaskan pada bab sebelumnya

4.1 Laporan Hasil Penelitian High Availability Database

4.1.1 Laporan instalation test

Dalam tahap ini peneliti melakukan proses instalasi berdasarkan pada beberapa sumber referensi dan literature yang ada untuk memastikan bahwa produk percona DB dengan mudah dapat dinstal pada server (http://www.olindata.com/blog/2014/07/percona-xtradb-cluster-56-setup) . Dari referensi tersebut hasil tahapan untuk melakukan intalasi PerconaDB adalah sebagai berikut:

a. Melakukan Instalasi Sistem operasi berbasis Open source Linux (Centos).
b. Melakukan setting IP Address untuk menghubungkan semua node menjadi 1 cluster database.
c. Melakukan proses update sistem operasi, hal ini dilakukan untuk memastikan bahwa software di OS sudah dalam kondisi yang terupdate.
d. Menghapus instalasi software mysql database yang ada di server.
e. Melakukan instalasi software percona Xtra DB Cluster.
f. Konfigurasi file /etc/my.cnf untuk menjalankan aplikasi database Xtra DB Cluster.
g. Mematikan service firewall dan selinux pada OS (dalam beberapa kasus dalam installation test hal ini bisa menyebabkan masing-masing node tidak dapat terkoneksi pada satu cluster.
h. Mematikan firewall ipv6.
i. Mengaktifkan layanan NTP.
j. Mendaftarkan layanan NTP untuk booting awal ketika server dihidupkan
k. Konfigurasi NTP client untuk menyamakan waktu pada node, dari test installation node cluster akan tidak terhubung jika perbedaan waktu pada satu node dengan node yang lain terlalu jauh.
l. Mengaktifkan layanan mysql pada booting awal proses.
m. Menjalankan proses bootstrap pada salah satu node cluster.

Secara umum proses tes installasi tidaklah menyulitkan dikarenakan dukungan dokumentasi dan literatur yang cukup lengkap didapat di Internet.

4.1.2 Basic sanity test

Peneliti dalam test ini melakukan pengujian untuk memastikan bahwa produk percona DB secara mendasar telah memenuhi fungsi-fungsi dasar dari produk yang ditawarkan adapun daftar-daftar layanan yang diberikan oleh percona (http://www.percona.com/software/percona-xtradb-cluster) adalah sebagai berikut:

No	Layanan Percona	Status uji coba
1	High Availability	OK
2	Multi master replication	OK
3	Parallel replication	OK
4	Automatic node provisioning	OK

Tabel 1 Daftar Sanity test

a. Layanan High Availability

   Layanan high availability database pada produk percona sudah dilakukan pengujian oleh peneliti, salah satu pengujian adalah dengan melakukan pengujian pada sekema *fault injection*. Layanan database tetap bias menerima proses input dan query meskipun terdapat salah satu node yang *fail* atau gagal.

b. Layanan Multi Master Replication

   Layanan dukungan untuk multi master replication yang diberikan produk percona diuji oleh peneliti dengan melakukan input pada semua node yang terdapat pada 1 cluster. Dari hasil pengujian yang dilakukan semua node dalam cluster tersebut dapat menerima input dan bersifat aktif-aktif.

c. Layanan Parallel replication

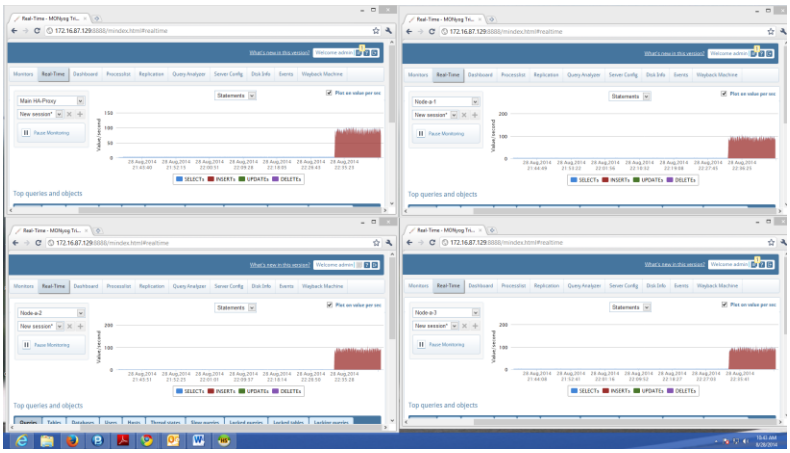
   Dukungan percona pada parallel replication dapat dilihat pada pengujian scalability test. Hasil yang di peroleh pada semua node meiliki nilai yang sama identik walaupun 1 atau 2 node pada *cluster* tersebut mengalami kegagalan sistem.

d. Layanan Automatic node provisioning

   Automatic node provisioning pada percona database dijalankan ketika salah satu node dalam satu cluster dihidupkan dari sebuah kegagalan sistem, automatic node provisioning juga akan otomatis berjalan ketika node baru dimasukkan pada group cluster.

4.1.3 Cluster fault injection test

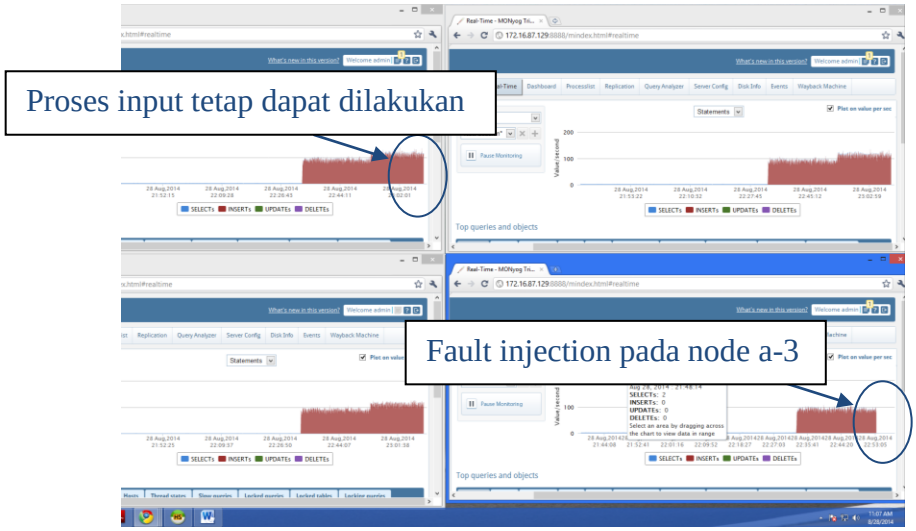
Dalam uji coba *Cluster fault injection* peneliti melakukan uji coba dengan melakukan penginputan data sebanyak 999.999 record pada *load balancer* dan *node* pada salah satu *cluster* disimulasikan akan dimatikan dalam waktu tertentu (*fault injection*). Pengujian diawali dengan peginputan data sebanyak 999.999 record ke server *load balancer* secara automatic .



Gambar 2 Proses input data sebelum fault injection

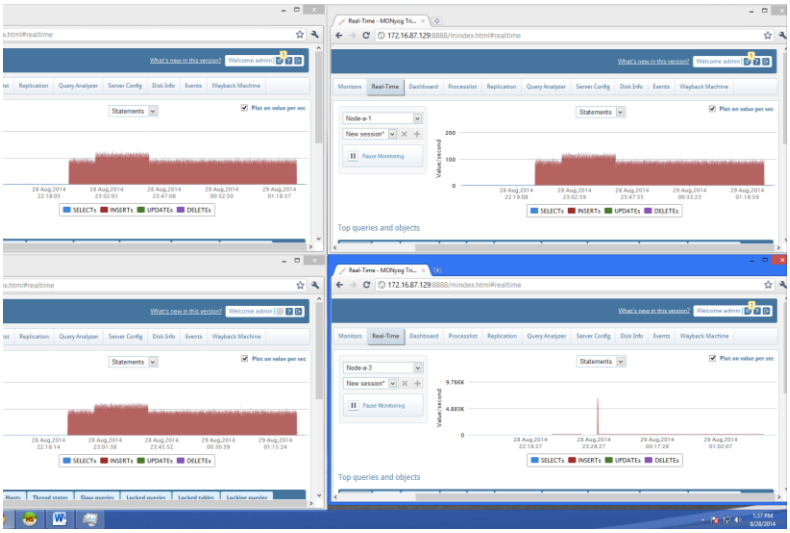
Pada gambar tampak bahwa proses input data yang dilakukan oleh client melalui IP Load balancer dapat berjalan dengan normal dengan kecepatan input rata-rata kurang lebih sebesar 102 write/second.

Setelah itu dilakukan pengujian *fault injection* pada node a-3 yang mengalami gangguan LAN NIC yang rusak. User tetap melakukan penginputan data pada HA Proxy (*Load Balancer*) dan proses tersebut tetap dapat berjalan meskipun database pada salah satu node (node-a-3) sedang mengalami gangguan

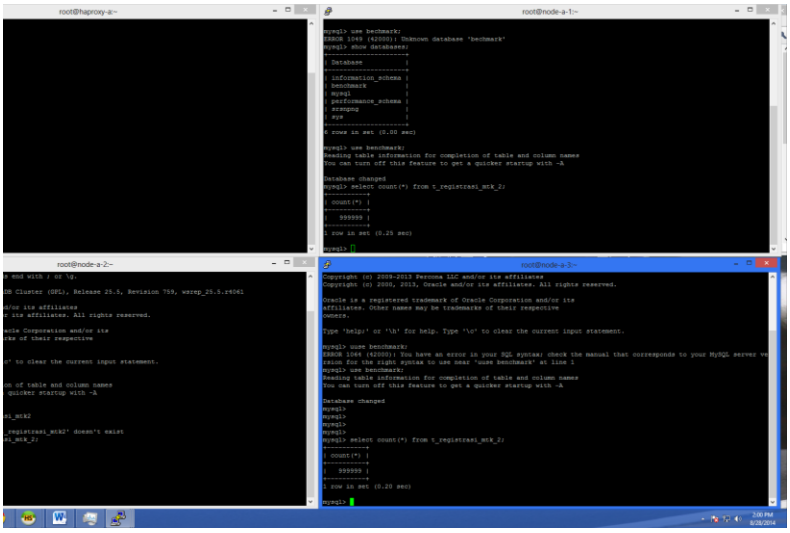


Gambar 3 Proses fault injection pada node-a-3

Kemudian dilakukan penggantian NIC Lancard selama10 menit berlangsung maka node-a-3 dihidupkan kembali dan proses pengcopy-an data terjadi pada node a-3 untuk menyamakan data yang tertingal pada nodea-3 dengan data pada node yang lain. Proses tersebut akan menaikkan kecepatan write speed pada node a-3 untuk menyamakan datanya dengan yang lain. Setelah data pada node-a-3 sama dengan data pada grup *cluster* maka proses penginputan data pada node-a-3 akan kembali normal seperti node-node yang lainnya. Berikut adalah gambar pada proses setelah dilakukan *fault injection* pada node-a-3.



Gambar 4 Proses copy data setelah dilakukan fault injection



Gambar 5 Hasil data akhir pada node-a-3

4.1.4 Scalability Test

Pengujian *Scalability test* dilakukan untuk mengetahui tingkah laku dan performa *cluster* database percona jika ditambahkan node pada clusternya. Pada pengujian ini dilakukan insert

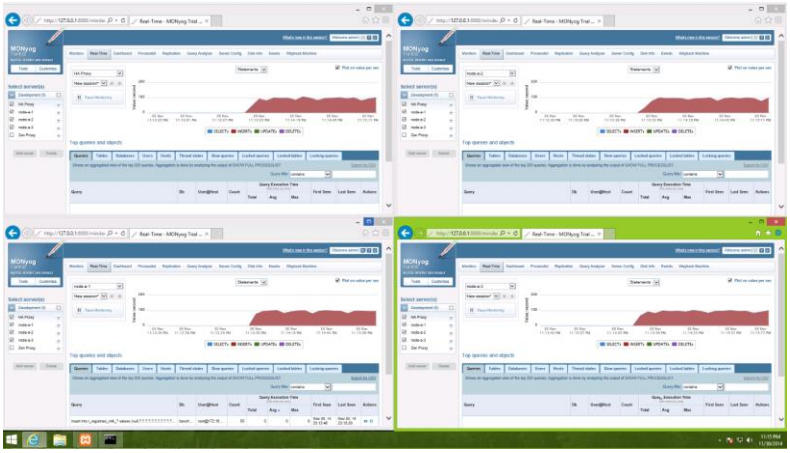
data sebanyak 150999 record pada sistem *cluster* database, Pengujian dilakukan pada sistem *cluster* yang terdiri dari 3 node, 2 node, dan single node database. Penggunaan waktu dalam proses input digunakan sebagai alat ukur performa dalam *scalability test*.

a. Cluster dengan 3 node database server

Pengujian pertama dilakukan pada *cluster* yang terdiri dari 3 *node server*. Dilakukan proses *insert* data sebanyak 150999 record pada *load balancer* server.

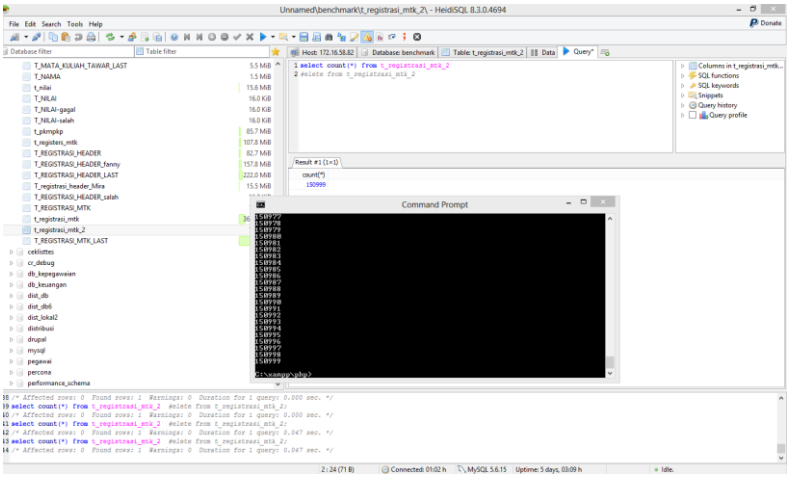
```
wsrep_local_state_comment    | Synced
wsrep_cert_index_size        | 0
wsrep_causal_reads           | 0
wsrep_cert_interval          | 0.000000
wsrep_incoming_addresses     | 172.16.58.237:3306,172.16.58.235:3306,172.16.58.236:3306
wsrep_cluster_conf_id        | 5
wsrep_cluster_size           | 3
wsrep_cluster_state_uuid     | 450f264a-fddf-11e3-b22e-6e0bdb00d62d
wsrep_cluster_status         | Primary
wsrep_connected              | ON
wsrep_local_bf_aborts        | 0
wsrep_local_index            | 0
wsrep_provider_name          | Galera
wsrep_provider_vendor        | Codership Oy <info@codership.com>
wsrep_provider_version       | 3.5(r178)
wsrep_ready                  | ON
```

Gambar 6 Scalability test dengan 3 node cluster

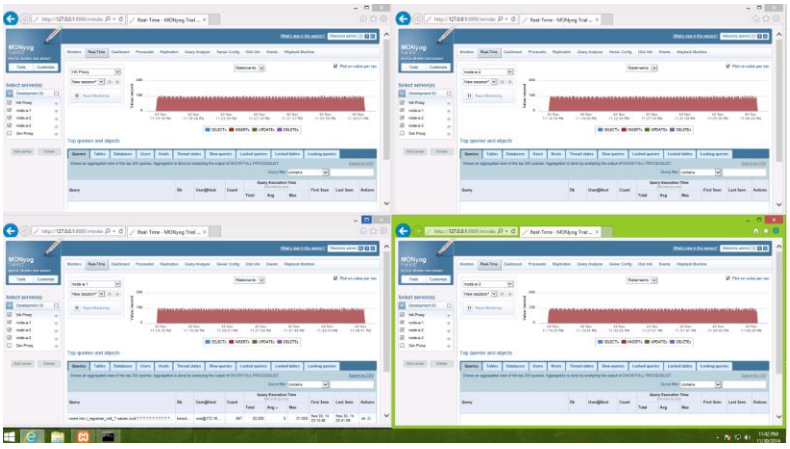


Gambar 7 Proses input 150999 pada sistem cluster 3 node

Setelah dilakukan proses insert data, data sebanyak 150999 record baru selesai diproses dalam waktu selama 27 menit 25 detik. Jumlah record pada node 1,2,3 memiliki jumlah hasil yang sama yaitu 150999 record.

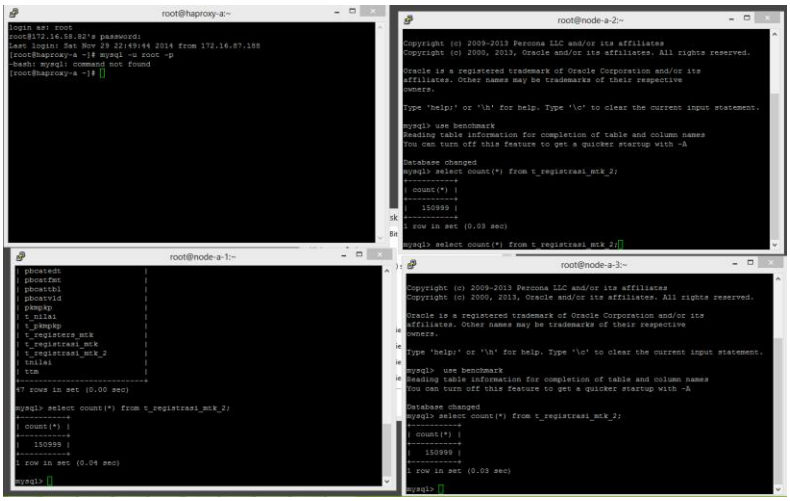


Gambar 8 Output 150999 pada Load Balancer



Gambar 9 Proses setelah dilakukan penginputan data

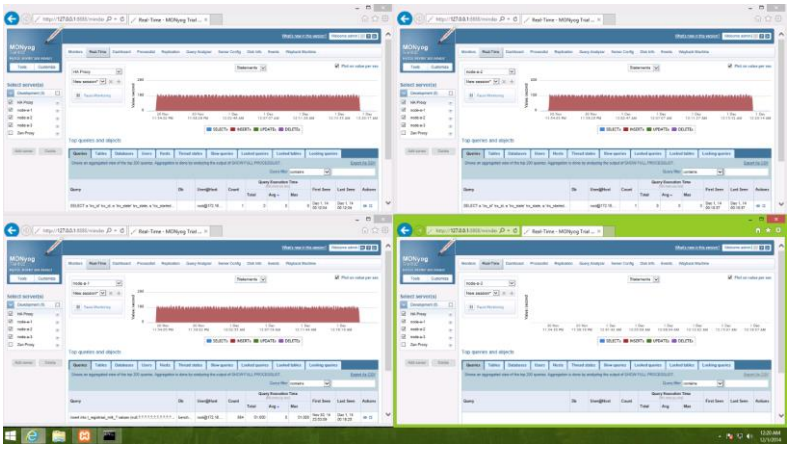
Gambar 10 Waktu input pada 3 node sistem cluster



Gambar 11 Hasil query pada seluruh node dengan sistem 3 node cluster

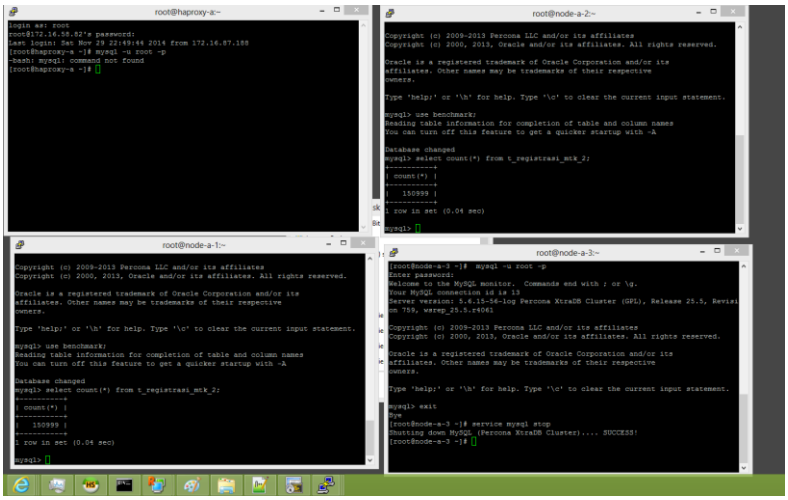
b. Cluster dengan 2 node database server

Untuk pengujian cluster dengan 2 node,digunakan cara yang sama pada percobaan sebelumnya. *Load balancer* diinputkan data sebanyak 150999 record, dan hasil pengujian yang dilakukan didapatkan untuk proses input data sebanyak 150999 record dibutuhkan waktu kurang lebih selama 25 menit 22 detik.



Gambar 12 Proses setelah dilakukan penginputan data pada 2 node

Gambar 13 Waktu input pada 2 node sistem cluster

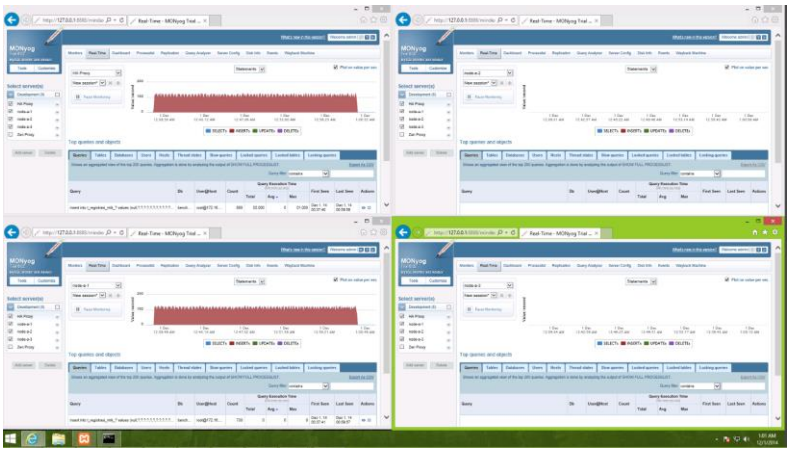


Gambar 14 Hasil query pada seluruh node dengan sistem 2 node cluster

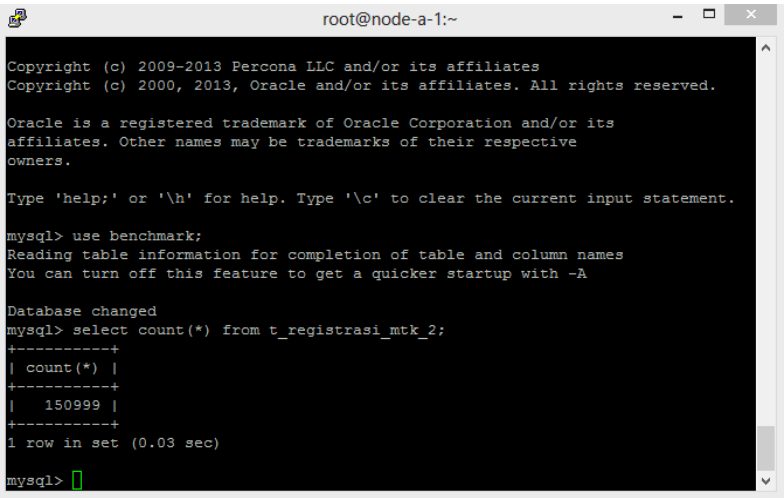
c. **Cluster dengan 1/ single node database server**

Untuk pengujian cluster dengan 1 node single. dilakukan untuk mengetahui performa optimal dari database server percona. Dengan single node database waktu temput yang dibutuhkan untuk menerima proses input sebanyak 150999 record adalah 22 menit 34 detik, lebih cepat 5 menit jika dibandingkan dengan system cluster 3 node.

Gambar 15 Waktu input pada single node sistem cluster



Gambar 16 Proses setelah dilakukan penginputan data pada 1 node

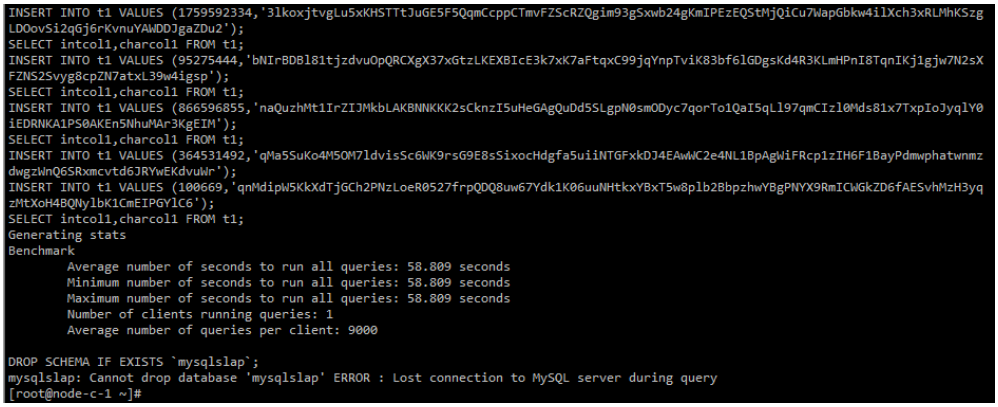


Gambar 17 Hasil query pada seluruh node dengan sistem single node cluster

4.1.4 Load test

Proses pengujian *load test* pada evaluasi HA penelitian ini tidak jauh berbeda dengan *scalability test* yang sudah dilakukan sebelumnya, jika pada *scalability test* digunakan proses input oleh satu user tanpa menjalankan multiple proses, pada pengujian *load test* user menjalankan multiple proses pada server dengan tujuan untuk membuat kondisi stress pada server. Pada pengujian ini dilakukan pembanding load test dimana database sistem cluster yang terdiri dari 3 node server akan dibandingkan dengan single node database.

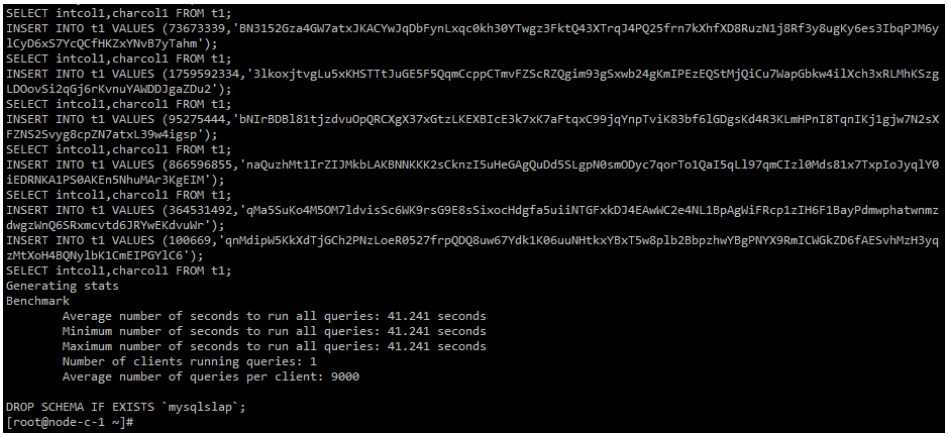
Load test pada sistem clusteryang terdiri dari 3 node dilakukan dengan cara melakukan simulasi query insert pada *load balancer*. Simulasi *Query insert* yang dilakukan di eksekusi oleh 1 user/pengguna yang menjalankan proses query insert sebayak 900 quey.



Gambar 18 Load test query pada cluster 3 node

Pada pengujian yang telah dilakukan pada Gambar 18 terlihat bahwa waktu rata-rata untuk menjalankan 9000 query oleh 1 user membutuhkan waktu selama 58,809 detik. Sedangkan jika

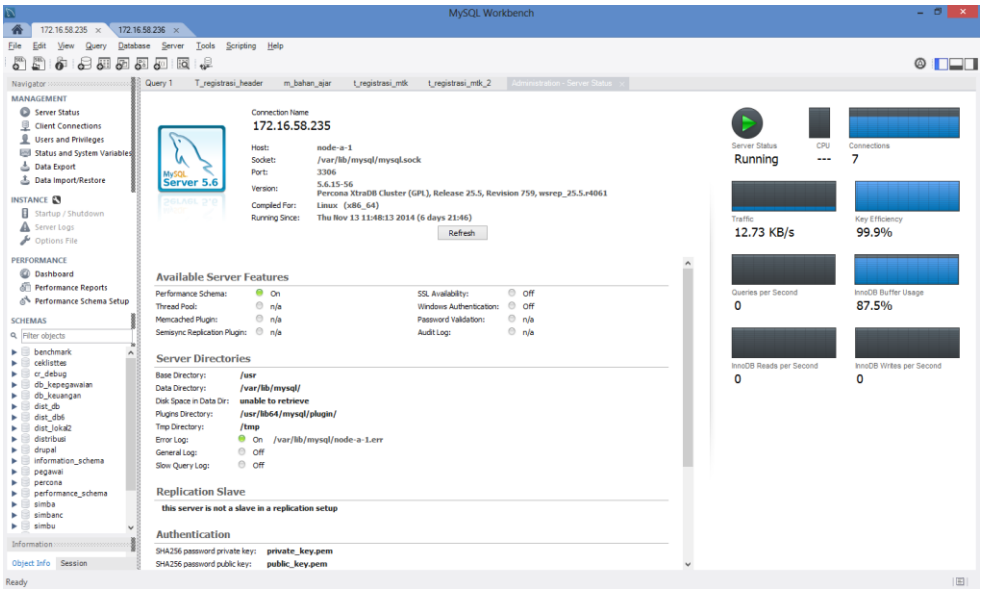
dijalankan pada database percona server yang *stand-alone* waktu yang dibutuhkan rata-rata sebesar 41,241 second17 detik lebih baik jika dibandingkan dengan sistem cluster.



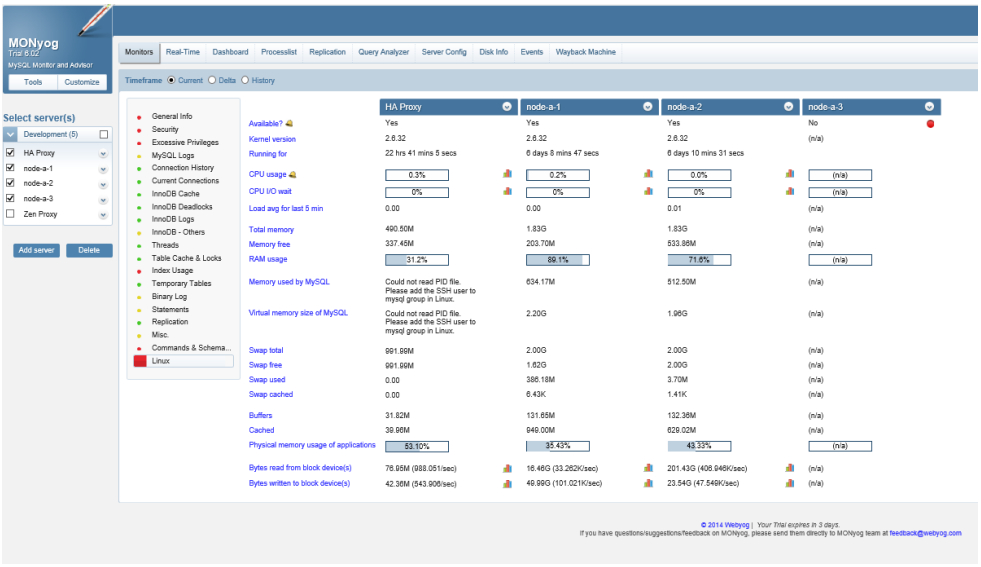
Gambar 19 Load test pada single node server

4.1.5 Stability test

Dalam uji coba *stability test*, uji coba digunakan untuk mengukur stabilitas database setelah digunakan dalam satu waktu tertentu. Penggunaan *database* percona cluster dalam penelitian ini telah diujicobakan sebagai database server pengembangan aplikasi Gran Desaign, dalam penggunaanya database ini masih berjalan dengan normal pada penggunaan selama 6 hari 21 jam yang didalamnya terdapat database sebanyak 15 database.



Gambar 20 Stability test Server pada node-a-1



Gambar 21 Stability test Server pada seluruh node

4.2 Evaluasi Penelitian

Dari hasil pengumpulan data dan observasi yang telah dilakukan penggunaan database Percona XtraDB Cluster memberikan solusi untuk meningkatkan availabilitas database, penggunaan master-master database membuat penggunaan database dapat berbagi beban penggunaannya, namun diperlukan pertimbangan dalam melakukan penambahan/penggunaan *node* pada *cluster*, semakin banyak *node* pada *cluster* akan meningkatkan tingkat ketersediaan layanan database namun akan menurunkan performa database. Hal dikarenakan bertambahnya beban traffic data pada jaringan untuk melakukan proses replikasi. Tabel berikut adalah hasil evaluasi uji coba *load test* yang telah dilakukan.

JUMLAH NODE PADA 1 CLUSTER	TINGKAT AVAILABILITY	WAKTU INSERT
1 node server	0%	22.34
2 node server	75%	25.22
3 node server	100%	27.25

Tabel 2 Evaluasi load test

Untuk mengurangi permasalahan penurunan performa pada database peneliti merekomendasikan untuk melakukan pemecahan segment pada database server sehingga traffic insert dan replikasi tidak menjadi satu pada NIC dan switch. Penambahan prosesor dan memory pada node juga relatif dapat meningkatkan performa cluster pada HA Database.

BAB 5. KESIMPULAN DAN SARAN

Pada bab ini dijelaskan beberapa simpulan yang didapat berdasarkan hasil evaluasi dan saran untuk kemungkinan pengembangan lebih lanjut.

5.1 Kesimpulan

Kesimpulan yang dapat di tarik dari hasil Implementasi High Availability pada database khususnya dalam studi kasus di Universitas Terbuka antara lain adalah :

1. Aplikasi Percona XtraDB menjadi salah satu solusi untuk penerapan database yang membutuhkan tingkat HA yang tinggi dengan harga yang relatif murah, tingkat sanity yang sesuai dan installasi yang mudah.
2. Penerapan aplikasi Percona XtraDB Cluster dan load balancer sebagai solusi High Avalability database dapat meningkatkan ketersediaan database hingga 99% tanpa down time.
3. Kemampuan rata-rata dalam proses penulisan data pada solusi HA yang telah dilakuakan pada penelitian ini sebesar 102 write/second, Semakin banyak penggunaan node pada cluster akan mengurangi kemampuan penulisan pada cluster tetapi akan meninggkatkan redudansi pada *load balancer*. Penggunaan segmentasi switch pada rancangan database HA akan membantu meningkatkan performa penulisan pada database.

5.2 Saran

Untuk pengembangan lebih lanjut berdasarkan proses penelitian yang dilakukan maka saran yang dapat diberikan adalah sebagai berikut :

1. Diperlukan penelitian lebih lanjut mengenai performa dari database engine percona yang lebih mendalam.
2. Tetap melakukan proses backup server untuk keperluan *recovery* database jika terjadi kerusakan atau disaster pada pusat database/data center dan juga sebagai keperluan arsip data tahunan jika diperlukan.
3. Melakukan pemisahan segment antara segment replikasi database dan akses database.

DAFTAR PUSTAKA

Cannaò, R. (2013, October 23). *ProxySQL Simple benchmark*. Retrieved April 25, 2014, from ProxySQL: http://www.proxysql.com/2013/10/simple-benchmark-on-mysql-proxy-vs.html

Jagannatha, S. (1999). System Testing Strategies For Highly Available Clustered System. *Software Testing, Analysis & Revies* , 1-24.

Lewis, M. (1999, Agustus 01). *A High-Availability Cluster for Linux*. Retrieved 04 24, 2014, from A High-Availability Cluster for Linux: http://www.linuxjournal.com/article/3247

Marcus, E. (2003). Blueprints For High Availability, Second Edition. In I. Wiley Publishing, *Blueprints For High Availability, Second Edition* (p. 575). Indianapolis: Wiley Publishing, Inc.

Chendramata, Aidil dan Wenang, Rikky. (2008). *Keamanan Aplikasi dan Database Server*. Departemen Komunikasi dan Informatika, Jakarta

Jagannatha, S. (1999). System Testing Strategies For Highly Available Cluster Systems. *Software Testing, Analysis & Review*, 3-24.

Nugroho, Bunafit. (2004). *Database Relasional MySQL*. Andi, Yogyakarta.

Wikipedia. (2013, 11 Maret). Centralized *database*. Diperoleh 28 Februari 2014, dari http://en.wikipedia.org/wiki/Centralized_*database*.

Wikipedia. (2014, 19 Februari). Distributed *database*. Diperoleh 28 Februari 2014, dari http://en.wikipedia.org/wiki/Distributed_*database*.

GudangLinux InfoCenter. (2013). MariaDB 5.5.29 Dengan Galera Cluster. Diperoleh 28 Februari 2014, dari http://gudanglinux.info/info/mariadb-5-5-29-dengan-galera-cluster/

Wikipedia. (2014). HAProxy. Diperoleh 28 Februari 2014, dari http://en.wikipedia.org/wiki/HAProxy.

Jagannatha, S. (1999). System Testing Strategies For Highly Available Cluster Systems. *Software Testing, Analysis & Review*, 3-24.

CURRICULUM VITAE

A. KETUA TIM

DATA PRIBADI		
Nama	:	Harris Rovandi
Tempat – Tanggal Lahir	:	Jakarta, 20 Januari 1978
Alamat	:	Jl. Agung Raya II No.50 RT.004/07 Lenteng Agung Jakarta Selatan
Handphone	:	08561316955
E-Mail	:	harris@ut.ac.id, harris@ecampus.ut.ac.id
PENDIDIKAN FORMAL		
2000	:	Sarjana Sistem Informasi Fakultas Ilmu Komputer Gunadarma
PENGALAMAN KERJA		
2003 – Sekarang 2000 – 2003	: :	Universitas Terbuka Teknikal Support PT MINCOM

B. ANGGOTA

DATA PRIBADI		
Nama	:	M. Novan Billiranto, S.Kom
Tempat – Tanggal Lahir	:	Lhokseumawe, 28 November 1985
Alamat	:	Jl. H. Piih No.28 RT.03/05 Sawangan Depok 16516
Handphone	:	08561121185
E-Mail	:	novan@ut.ac.id, novan@ecampus.ut.ac.id
PENDIDIKAN FORMAL		
2007	:	Sarjana Teknik Informatika Fakultas Ilmu Komputer Bina Nusantara
PENGALAMAN KERJA		
2010 – Sekarang 2007 - 2010	: :	Universitas Terbuka Network Engineer PT. RTI Infokom