JIRK
Journal of Innovation Research and Knowledge
Vol.3, No.9, Februari 2024

SISTEM DATABASE TERDISTRIBUSI UNTUK PENINGKATAN KETERSEDIAAN DATA PADA APLIKASI REKAM MEDIS

Oleh
Pulut Suryati[1], Andi Susanto[2], Sudarmanto[3]
[1,2,3]Universitas Teknologi Digital Indonesia
E-mail: [1]lut_surya@utdi.ac.id, [3]darmanto@utdi.ac.id

Article History:
Received: 21-01-2024
Revised: 29-01-2024
Accepted: 24-02-2024

Keywords:
Database,
Terdistribusi,
Clustering,
Ketersediaan Data

Abstract: *Ketersediaan data merupakan salah satu layanan teknologi informasi yang mendukung kelancaran operasional organisasi. Selain memerlukan infrastruktur jaringan komputer yang handal, kebutuhan informasi terkait data organisasi juga tergantung pada infrastruktur pada server database. Risiko kegagalan suatu sistem pada server dapat terjadi sewaktu-waktu, dan apabila server database mengalami gangguan, dan berpotensi kehilangan data yang tersimpan di server tersebut. Salah satu solusi untuk mengatasi permasalahan tersebut adalah dengan menggunakan sistem database terdistribusi, dengan metode yang digunakan adalah clustering. Pendekatan clustering sangat efektif dalam memastikan ketersediaan layanan tinggi (High-Availability) aplikasi. Implementasi clustering database pada aplikasi rekam medis sehingga dapat diketahui ketersediaan layanan. Clustering database dilakukan menggunakan Patroni dan Ansible. Patroni adalah aplikasi yang memberikan kemampuan untuk mengkonfigurasi, menerapkan, dan menjalankan arsitektur PostgreSQL yang bersifat High-Availability. Hasil pengujian dari pengukuran response time PostgreSQL single instance pada simple mode mengalami kenaikan sebesar 23.13% dibanding database cluster. Pengukuran throughput pada clustering database mengalami peningkatan sebesar 27.09% dibanding database single instance.*

PENDAHULUAN

Aplikasi database semakin berkembang, baik dalam hal kegunaan, ukuran, maupun kompleksitas. Database di masa sekarang dituntut agar dapat berjalan dengan cepat, mempunyai reliability (keandalan) dan availability (ketersediaan) yang tinggi, agar keandalan tersebut dapat dicapai, ada beberapa solusi alternatif yang dapat digunakan dalam pengelolaan dan penanganan database tersebut salah satunya adalah dengan

pemanfaatan clustering database.

Clustering database adalah kumpulan dari beberapa server yang berdiri sendiri yang kemudian bekerja sama sebagai suatu sistem tunggal. Dengan clustering database, data yang disimpan dapat terbagi ke beberapa mesin dan pada saat aplikasi berjalan, semua mesin yang menyimpan data tersebut dianggap sebagai satu kesatuan. Metode clustering sangat baik untuk load balancing dalam penanganan kegagalan sistem karena kemampuan tiap mesin akan digunakan. Dan jika ada salah satu mesin yang mengalami kegagalan maka sistem tidak akan langsung terganggu

Aplikasi *database* semakin berkembang, baik dalam hal kegunaan, ukuran, maupun kompleksitas. *Database* di masa sekarang dituntut agar dapat berjalan dengan cepat, mempunyai *reliability (*keandalan) dan *availability* (ketersediaan) yang tinggi, agar keandalan tersebut dapat dicapai, ada beberapa solusi alternatif yang dapat digunakan dalam pengelolaan dan penanganan *database* tersebut salah satunya adalah dengan pemanfaatan *clustering database*. *Clustering database* adalah kumpulan dari beberapa server yang berdiri sendiri yang kemudian bekerja sama sebagai suatu sistem tunggal.

Dengan *clustering database,* data yang disimpan dapat terbagi ke beberapa mesin dan pada saat aplikasi berjalan, semua mesin yang menyimpan data tersebut dianggap sebagai satu kesatuan. Metode *clustering* sangat baik untuk *load balancing* dalam penanganan kegagalan sistem karena kemampuan tiap mesin akan digunakan. Dan jika ada salah satu mesin yang mengalami kegagalan maka sistem tidak akan langsung terganggu.

Tujuan penelitian ini adalah bagaimana mengimplementasi clustering database pada aplikasi rekam medis dan bagaimana pengujian high availability pada clustering database.

LANDASAN TEORI

Salah satu teknik clustering database adalah dengan replikasi. Replikasi merujuk pada langkah-langkah untuk mentransfer data dari suatu basis data utama ke basis data lain yang disimpan di komputer terpisah. Untuk menjalankan replikasi, diperlukan setidaknya dua komputer yang bertindak sebagai server, di mana satu berperan sebagai server utama (master) dan yang lainnya sebagai server pendukung (slave). Melalui replikasi, informasi dapat disebar ke lokasi yang berbeda, memungkinkan akses pengguna dari jarak jauh melalui berbagai jaringan seperti LAN, WAN, koneksi Dial-up, koneksi nirkabel, dan internet.

Setiap harinya, terjadi peningkatan yang sangat signifikan dalam jumlah pengguna internet yang mengakses aplikasi web, dengan permintaan beban (request) yang bervariasi Dampaknya adalah perlunya peningkatan kinerja server untuk mengatasi tuntutan ini. 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. Tindakan mendistribusikan kembali beban kerja di antara server dalam suatu sistem terdistribusi. Tujuannya adalah untuk meningkatkan penggunaan sumber daya, mengoptimalkan waktu respons, dan mencegah situasi di mana beberapa server sibuk sementara yang lain tidak terbebani atau melakukan pekerjaan yang minim, 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

ISSN 2798-3471 (Cetak)
ISSN 2798-3641 (Online)

koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan throughput, memperkecil waktu tanggap dan menghindari overload pada salah satu jalur koneksi. Selain itu, performa database sebagai media penyimpanan data juga perlu dijaga agar berjalan dengan cepat dan memiliki tingkat keandalan serta ketersediaan yang tinggi. Clustering database merupakan solusi yang tepat untuk mencapai tujuan ini, karena dengan metode ini, data yang tersimpan dapat dibagi menjadi beberapa bagian, memungkinkan distribusi beban yang seimbang.

Selain itu, kinerja database sebagai media penyimpanan data agar bisa berjalan cepat dan memiliki keandalan serta ketersediaan yang tinggi, clustering database dapat dijadikan solusi yang tepat dalam penyimpanan data. Karena dengan metode clustering database data yang tersimpan dapat dibagi menjadi beberapa server pada saat aplikasi berjalan, semua server yang menyimpan data dianggap sebagai satu kesatuan. [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 [3,5]. Basis data mendeskripsikan state organisasi atau perusahaan atau sistem dan merupakan komponen utama sistem informasi karena semua informasi untuk pengambilan keputusan berasal dari data pada database. Replikasi merujuk pada suatu proses di mana data dipindahkan dari satu basis data master ke basis data lain yang berperan sebagai salinan dan disimpan di komputer yang berbeda. Untuk menjalankan proses replikasi, diperlukan setidaknya dua komputer yang berfungsi sebagai server, di mana satu di antaranya bertindak sebagai master dan yang lainnya sebagai slave. Melalui replikasi ini, data dapat disebarkan ke lokasi yang berbeda, memungkinkan pengguna yang berada jauh dapat mengaksesnya melalui berbagai jenis koneksi seperti LAN, WAN, Dial-up Connection, koneksi nirkabel, dan internet.

METODE PENELITIAN

Adapun tahapan pada penelitian adalah persiapan, membuat rancangan *clustering database*, membuat rancangan implementasi dan konfigurasi, dan membuat rancangan pengujian. Pada tahap persiapan, dilakukan analisis kebutuhan dengan mengumpulkan data tentang perangkat keras dan perangkat lunak yang diperlukan. Selain itu, proses ini memerlukan satu server sebagai server aplikasi dan penyeimbang beban, serta tiga server untuk membangun pengelompokan database.

Pada tahap perancangan basisdata, Basis data atau database merupakan sekumpulan data yang saling berelasi atau berkaitan satu sama lain, dengan menggunakan basis data dapat digunakahan untuk penyimpanan data dan pengolahannya. Data di dalam suatu basis data, pada aplikasi rekam medis dibutuhkan database yang memiiki beberapa tabel-tabel. Selanjutnya tahap implementasi Perancangan database clustering server menggunakan PostgreSQL sebagai aplikasi database, patroni sebagai tools manajemen *clustering*, dan *ansible* sebagai alat untuk dalam mengkonfigurasi server. Untuk implementasi dan penggujian sistem langkah yang dilakukan adalah

1. Membuat Script Vagrant, Script vagrant bertujuan untuk membuat virtualisasi server yang dibutuhkan.
2. Membuat script otomatisasi ansible, untuk menginstal dan konfigurasi web server, load balancer, dan clustering database server

3. Menjalankan Virtualisasi Vagrant, Setelah proses membuat script vagrant dan membuat script otomatisasi ansible selanjutnya jalankan vagrant.
4. Menjalankan Script Ansible, Selanjutnya jalankan script ansible untuk menginstall dan konfigurasi web server, clustering database server, dan load balancer server
5. Cek Provisioning Ansible, Setelah proses ansible selesai, selanjutnya pengecekan konfigurasi pada server apakah sesuai dan service berjalan baik
6. Deploy Aplikasi Klinik Tumbuh Kembang, Tahap berikutnya deploy aplikasi tumbuh kembang untuk menguji integrasi server database dan web server.
7. Tahap rancangan pengujian, Rencana pengujian dilakukan menggunakan aplikasi SysBench yang terinstall pada komputer client. Proses pengujian dilakukan pada client dengan menjalankan aplikasi SysBench. Parameter yang diukur adalah Transaction per Second (TPS) dan response time yang diubah jumlah threads-nya. Perubahan threads tersebut dilakukan mulai dari 1, 2, 4, dan 8. Pengukuran tersebut dilakukan berdasarkan mode pengujian berikut:
   a. Simple-mode merupakan pengujian yang dilakukan hanya pada proses read-only data atau hanya pembacaan data saja.
   b. Complex-mode merupakan pengujian yang dilakukan dengan proses yang kompleks yang meliputi READ, INSERT, DELETE, dan UPDATE.

HASIL DAN PEMBAHASAN

Untuk mendukung tercapainya penelitian ini dibutuhkan data yang digunakan untuk mengukur tingkat keandalan clustering database dalam mengelola beban permintaan seperti CREATE, READ, UPDATE, dan DELETE. PostgreSQL adalah salah satu jenis database management system atau sistem manajemen berbasis data yang bersifat open-source. PostgreSQL adalah perangkat lunak yang dipakai untuk berbagai aktivitas terkait pengolahan data, seperti menyimpan data, mengedit data, menghapus data, dan replikasi data. Patroni adalah sebuah tools untuk membangun dan mengelola high availability pada PostgreSQL dengan konfigurasi yang dapat disesuaikan. Dibuat menggunakan bahasa pemrograman python. Perancangan database clustering server menggunakan PostgreSQL sebagai aplikasi database, patroni sebagai tools manajemen clustering, dan ansible sebagai tools otomatisasi dalam mengkonfigurasi server.

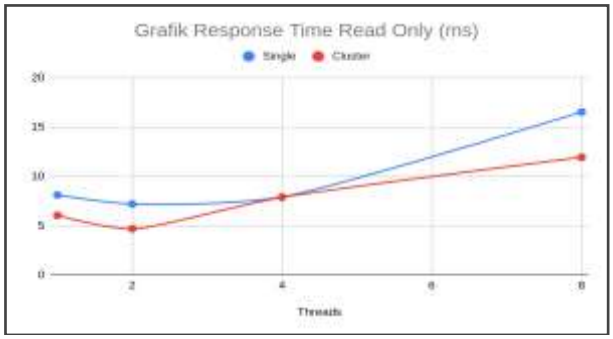
Pengujian dilakukan menggunakan skenario simple mode. Pada skenario ini ditampilkan data throughput dan response time dari hasil transaksi read only. Selengkapnya dapat dilihat pada Tabel 1 dapat dilihat bahwa database cluster mampu menangani lebih banyak request dibanding database single instance

Tabel 1 Hasil Penggujian database clustere simple mode

No	Threads	Throughput (TPS)		Response Time (ms)	
		Single	Cluster	Single	Cluster
1	1	122.88	164.82	8.13	6.06
2	2	277.16	424.84	7.21	4.7
3	4	503.97	506.83	7.93	7.89
4	8	483.54	667.02	16.54	11.95

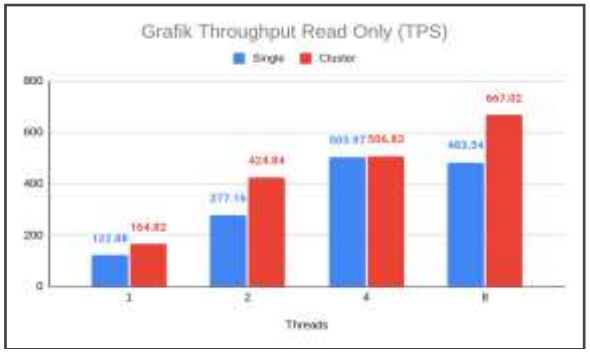
ISSN 2798-3471 (Cetak)
ISSN 2798-3641 (Online)

Dapat dilihat pada Gambar 1 merupakan grafik rata-rata *response time* dari pengujian dengan *simple mode*, *response time* adalah rentang waktu dari saat pengguna mengirimkan *request* ke *server* hingga *server* selesai merespon *request* tersebut. Gambar 1 memperlihatkan pengaruh penambahan *threads* terhadap *response time*. Parameter ini akan berdampak pada kecepatan akses yang dibutuhkan suatu *server cluster*. Semakin kecil nilai yang diperoleh menyatakan semakin baik pula kondisi *cluster* tersebut.



Gambar 1 Grafik *response time* transaksi read only

Selanjutnya *throughput* adalah jumlah *request* yang ditangani dalam satu detik pada Gambar 2 memperlihatkan pengaruh penambahan *threads* terhadap *throughput*. Parameter ini akan berdampak pada jumlah *request* yang ditangani suatu *server cluster* dalam satu detik. Semakin besar nilai yang diperoleh menyatakan semakin baik pula performa *cluster* tersebut.



Gambar 2 Grafik *throughput* transaksi read only

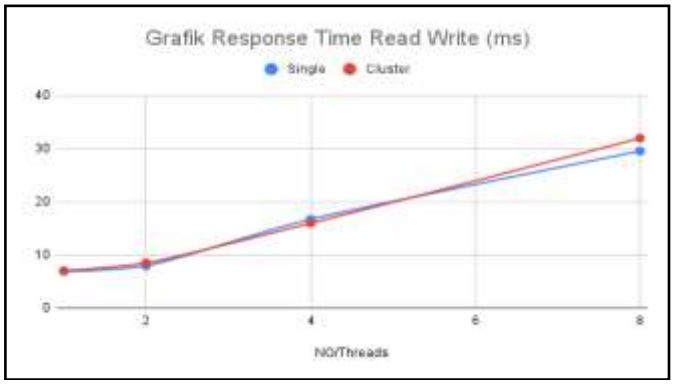
Dari uji simple mode terdapat kenaikan *response time* rata-rata sebesar 2.3025ms atau 23.13% pada *single instance* dibanding *database cluster*, hal ini menunjukkan terjadi perlambatan *response time* pada database *single instance.* Terdapat kenaikan pada *database cluster* sebesar 93.99 TPS atau sebesar 27.09% dibanding database *single instance*, hal ini menunjukkan *database cluster* mampu menangani beban *request* lebih banyak dibanding *single instance.*

Pengujian menggunakan skenario *complex mode*. *Complex-mode* merupakan pengujian yang dilakukan dengan proses yang kompleks yang meliputi READ, INSERT, DELETE, dan UPDATE. Pada mode ini data yang ditampilkan adalah *throughput* dan *response time*, untuk hasil selengkapnya dapat dilihat pada Tabel 2.

Tabel 2 Hasil pengujian database cluster complex mode

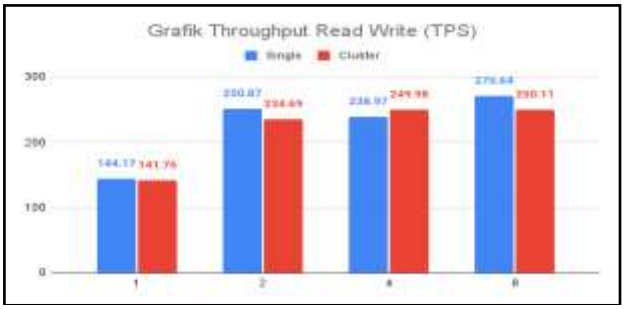
NO	Threads	Throughput (TPS)		Response Time (ms)	
		Single	Cluster	Single	Cluster
1	1	144.17	141.76	6.93	7.05
2	2	250.87	234.69	7.97	8.52
3	4	238.97	249.98	16.72	15.99
4	8	270.64	250.11	29.55	31.97

Dapat dilihat pada Gambar 3 merupakan grafik rata-rata *response time* dari pengujian dengan *complex mode*, memperlihatkan pengaruh penambahan *threads* terhadap *response time*. Parameter ini akan berdampak pada kecepatan akses yang dibutuhkan suatu *server cluster*. Semakin kecil nilai yang diperoleh menyatakan semakin baik pula kondisi *cluster* tersebut.



Gambar 3 Grafik *response time* transaksi *read write*

Gambar 4 memperlihatkan pengaruh penambahan *threads* terhadap *throughput* pada *complex mode*, parameter ini akan berdampak pada jumlah *request* yang ditangani suatu *server cluster* dalam satu detik. Semakin besar nilai yang diperoleh menyatakan semakin baik pula performa *cluster* tersebut.



Gambar 4 Grafik *throughput* transaksi *read write*

Uji *complex mode* didapat hasil yang tidak jauh berbeda, nilai *throughput* terendah yaitu 141.76 TPS pada beban 1 *threads,* dan nilai *throughput* tertinggi yaitu 270.64 TPS pada beban 8 *threads.* Perbedaan hasil juga dipengaruhi oleh proses yang berjalan bersamaan pada komputer *host*.

KESIMPULAN

Berdasarkan hasil dari pelaksanaan penelitian implementasi clustering database untuk *high availability* pada aplikasi rekam medis klinik tumbuh kembang anak, maka dapat diambil kesimpulan performa sebuah database meningkat jika dibandingkan dengan database PostgreSQL yang tidak menggunakan sistem clustering. Hasil dari pengukuran *response time* PostgreSQL *single instance* pada *simple mode* mengalami kenaikan sebesar 23.13% dibanding *database cluster.* Pengukuran *throughput* pada *clustering database* mengalami peningkatan sebesar 27.09% dibanding database single instance. Hal ini menunjukkan database cluster memiliki *response time* dan *throughput* lebih baik dibanding *database single instance.* Penggunaan ansible dalam membangun clustering database PostgreSQL berhasil dan dapat mempersingkat waktu konfigurasi.

Pengakuan/Acknowledgements

Terimakasih kepada semua tim yang telah perpastisipasi, terimakasih juga kepada Universitas Teknologi Digital Indonesia atas dukung yang telah diberikan.

DAFTAR PUSTAKA

[1] Hodges, R. Database High Availability and Scalability. CTO Continuent, Inc., (2007)
[2] Dolly Simon Kristian, Adian Fatchur Rochim, and Eko Didik Widianto. Pengembangan Sistem Replikasi Dan Redundansi Untuk Meningkatkan Kehandalan Basisdata Mysql. Jurnal Teknologi dan Sistem Komputer, Vol.3, No.4, Oktober 2015 (e-ISSN: 2338-0403).
[3] Halim Setya Muliyantoro. Penerapan Metode Load-Balancing Clusters Pada Database Server Guna Peningkatan Kinerja Pengaksesan Data. Techno Nusa Mandiri. Vol. IX No.1, Maret 2013.
[4] Ali M. Alakeel. A Guide to Dynamic Load Balancing in Distributed Computer Systems., IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.6, June 2010.
[5] Hariyanto, Bambang. Sistem Manajemen Basis Data. Informatika: Bandung.2004.
[6] Charles Bell, M. K. (2010). MySQL High Availability: Tools for Building Robust Data Centers. (p. Juni). O'Reilly Media
[7] Dolly Simon Kristian, Adian Fatchur Rochim, Eko Didik Widianto (2015). Pengembangan Sistem Replikasi Dan Redundansi Untuk Meningkatkan Kehandalan Basisdata Mysql. Jurnal Teknologi dan Sistem Komputer, Vol.3, No.4, Oktober 2015 (e-ISSN: 2338-0403).

HALAMAN INI SENGAJA DIKOSONGKAN

ISSN 2798-3471 (Cetak)
ISSN 2798-3641 (Online)