PEMILIHAN ARSITEKTUR BASIS DATA BERDASARKAN ANALISIS KINERJA ORACLE *INSTANCE* TUNGGAL *DAN* RAC

[1]Sri Astuti, [2]Lulu Chaerani Munggaran
[1,2]Manajemen Sistem Informasi, Program Pasca Sarjana, Universitas Gunadarma
Jl. Margonda Raya No. 100, Depok 16424, Jawa Barat
[1]sriast.nirlam@gmail.com, [2]lulu@staff.gunadarma.ac.id

Abstrak

Beban yang semakin bertambah pada suatu server basis data dapat mengakibatkan menurunnya performa sistem untuk menyelesaikan perintah penggunanya. Berkurangnya performa dari server tersebut juga bisa berdampak terhadap response time yang semakin lambat. Salah satu cara meningkatkan kinerja server adalah dengan menambahkan sumber daya berupa server baru yang bisa dikonfigurasikan sebagai cluster. Penambahan server baru tersebut dapat disebut sebagai horizontal scaling. Horizontal scaling membantu dalam pembagian beban kerja dalam pemprosesan data. Tujuan dari penelitian ini adalah membandingkan kinerja Oracle basis data instance tunggal dengan basis data Real Application Cluster (RAC) sebagai dasar pemilihan arsitektur basis data, menggunakan pengujian throughput, response time dan error rate. Hasil dari pengujian untuk jumlah pengguna 100,200,300,400,500,600,700,800,900,1000, basis data instance tunggal lebih unggul dalam hal throughput dan response time dibandingkan dengan cluster RAC. Basis data RAC mendapatkan hasil yang lebih unggul dibandingkan basis data instance tunggal dalam hal performa error rate, hal ini menunjukkan bahwa basis data RAC dapat melayani transaksi yang lebih banyak tanpa menghasilkan transaksi error.

Kata kunci: *Basis Data, Cluster, Error Rate, Horizontal Scaling, Response Time, Throughput.*

Abstract

Increasing load on database server can result in performance decrease to complete requests from the users. It can also have impact to slower response time of the system. One way to increase the server performance is to install new server for additional resources configured as cluster. This is called horizontal scaling. Horizontal scaling helps in load balancing to process the data. This study aims to compare the performance Oracle Database with Single Instance and Real Application Cluster (RAC) as the basis to choose database architecture, by testing the throughput, response time and error rate. Test results for total of 100,200,300,400,500,600,700,800,900,1000 users, shows that single instance database are better in terms of throughput and response time compared to RAC cluster. RAC database have better results compared to single instance database in terms of error rate, which shows that RAC database can handle more transactions without resulting in transaction error.

Keywords: *Cluster, Database, Error Rate, Horizontal Scaling, Response Time, Throughput.*

PENDAHULUAN

Pada bisnis suatu perusahaan terutama yang berupa *startup*, sistem komputer yang dimiliki biasanya masih sederhana karena modal yang terbatas untuk investasi. Basis data yang digunakan masih berjalan pada *server* tunggal. Seiring dengan berkembangnya bisnis dari perusahaan tersebut, konfigurasi basis data yang sederhana ini akan tidak dapat lagi menangani beban yang bertambah besar. Kegagalan pada perangkat keras, sistem operasi, atau *server* yang tiba-tiba berhenti bekerja dapat mengakibatkan proses transaksi terhenti atau tidak tercatat dengan baik. Semakin bertambahnya beban pada suatu *server* basis data juga dapat mengakibatkan menurunnya performa sistem untuk menyelesaikan perintah dari penggunanya. Berkurangnya performa dari *server* tersebut juga bisa berdampak terhadap *response time* pada sistem menjadi semakin lambat. Hal ini dapat menjadi pertimbangan penting agar peningkatan kinerja sistem secara bertahap pada server dapat dengan mudah dilakukan sesuai dengan kebutuhan di masa yang akan datang.

Salah satu cara yang dapat dilakukan untuk meningkatkan performa server adalah dengan menambahkan CPU atau sumber daya lain yang diperlukan, disebut juga dengan *vertical scaling*. Cara ini bisa diibaratkan seperti memasang perban sementara saja, karena dibatasi oleh opsi yang tersedia pada *mainboard* yang digunakan. Cara lainnya untuk meningkatkan kinerja *server* adalah dengan menambahkan *server* baru sehingga dapat dikonfigurasi menjadi sebuah *cluster*, disebut juga dengan *horizontal scaling* [1]. Cara ini memiliki kelebihan dimana dapat dilakukan konfigurasi *load balancing*, workload *distribution* dan high *availability*.

Horizontal Scaling berkaitan dengan penambahan mesin *server* untuk membagikan beban kerja pemprosesan data. Setiap mesin *server* tersebut memiliki sumber daya yang terpisah. Proses ini bisa disebut juga sebagai *scaling out* [2]. Salah satu metode *horizontal scaling* yaitu dengan melakukan *clustering*.

Cluster merupakan kumpulan dari beberapa mesin yang beroperasi sebagai sistem tunggal. Pada aplikasi yang berjalan dalam waktu yang lama dan diakses secara bersamaan, *cluster* memberikan cara untuk membagi tugasnya atau bisa disebut dengan *load balancing*. *Load balancing* pada *cluster* digunakan untuk memperbaiki performa sistem dengan membagi beban kerjanya ke beberapa mesin yang sedang *idle* atau sumber dayanya tidak digunakan. Seiring meningkatnya waktu aktif aplikasi dan ukuran *cluster* yang semakin besar, menambahkan *failover* menjadi sangat penting. Perbaikan di sisi *availability* juga didapatkan karena aplikasi tetap berjalan walaupun hanya tersisa satu *node* yang berjalan pada *cluster* tersebut.

Komponen terpenting dari *cluster* adalah *node server*, *interconnect* dan *storage disk*. Oracle memiliki sistem *cluster* yang disebut Oracle *Real Application Cluster*

(RAC). Arsitektur Oracle RAC merupakan arsitektur basis data *cluster* yang berjalan pada *shared disk cluster*, sebuah *cluster* dimana setiap *node* dapat mengakses langsung ke semua *shared-disks*. *Shared disks* dapat menggunakan *Storage Area Network* (SAN) atau *array* RAID yang dapat diakses langsung oleh beberapa *node*. RAC memberikan perbaikan performa pada aplikasi ketika aplikasi tersebut diakses secara bersamaan oleh beberapa sistem. Oracle RAC mengatur akses data, ketika terdapat perubahan di antara *instance* masing-masing *node*, data yang dihasilkan akan tetap konsisten. *Cluster interconnect* mengizinkan *instance* untuk mendapatkan informasi dan data *images* antar *node*.

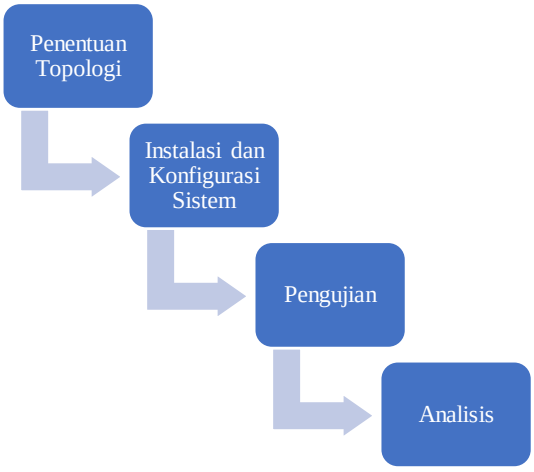
Perangkat keras, perangkat lunak dan sumber daya menjadi bagian penting dalam melakukan tes performa dari suatu sistem Perangkat keras, perangkat lunak dan sumber daya menjadi bagian dalam melakukan tes performa dari suatu sistem untuk menentukan apakah suatu sistem sesuai dengan SLA (*Service Level Aggrement*) dengan memperhatikan kepentingan kecepatan, kestabilan dan *scalability* sesuai dengan beban kerja yang sesuai harapan [3]. Sistem yang baik, cepat dan handal menjadikan kepuasan tersendiri bagi pengguna, terutama untuk organisasi bisnis yang besar yang mempedulikan kecepatan bisnis dan kepuasan pengguna. Pengujian performa membantu dalam menganalisis apakah *environment* sistem tersebut layak untuk digunakan mendapatkan kecepatan bisnis dan kepuasan pengguna. Analisis ini dapat dilakukan dengan melakukan *load test* pada *environment* sistem yang sebelum akan dijadikan sebagai *production*. Analisis dari *load test* ini dilakukan dengan memaksimalkan sumber daya baik dari perangkat keras dan perangkat lunak. Analisis dengan *load test* membantu dalam mencari *bottleneck* sistem yang akan diterapkan menjadi *production*. Tujuan dari tes performa tidak untuk menemukan *bug* tetapi mengurangi *bottleneck*. Konsep tes performa dan analisis perbandingan suatu sistem dalam hal *response time*, *throughput*, dan *deviation* [4]. Pada penelitian performa basis data *mysql cluster*, dilakukan perbandingan tes performa pada basis data dengan konfigurasi *default* mysql dan mysql *cluster*. Hasil *load test* performa tersebut membandingkan hasil *throughput* dan *response time* yang menghasilkan *throughput* dan *response time* pada mysql *cluster* cukup baik dibandingkan dengan mysql *server* (*default*) [5]. Persentase keberhasilan (*Error rate*) yang juga digunakan dalam analisis *load test* pada dua teknologi *microservices* yang berbeda yang menjelaskan *error rate* merupakan rata-rata *error* yang terjadi ketika proses *request* dilakukan. *Error rate* 0% berarti tidak ada terjadi error sama sekali untuk semua *request* yang dilakukan [6]. Penelitian dan melakukan evaluasi mengenai *arsitektur master-slave* sebagai solusi tradisional dan solusi *high availability* basis data *cluster-based* dengan melihat kualitas dan kuantitas dari performa. Basis data yang

digunakan yaitu menggunakan MariaDB. Perbandingan performa berdasarkan hasil *benchmark* dengan Open Stack/Dev Stack berbasis *cloud*.

Pengujian performa pada basis data dari solusi implementasi menggunakan MariaDB dengan Galera Cluster memperlihatkan bahwa *master-slave* memiliki hasil performa yang sama atau lebih baik dalam hal *throughput* dan *response time*. Solusi arsitektur *cluster-based* lebih baik dalam hal data konsistensi, *no loss data* dan *scalability* ketika mengalami kegagalan, serta menyediakan keunggulan dalam *scalability* untuk *read* dan *write* [7]. Penelitian perbandingan performa RAC di dilakukan pada fisikal *server* dan *server* virtual SAN, dengan membandingkan nilai TPM (*Transaction per minute*) dan IOPS [8]. Penelitian dalam pengujian pada aplikasi berbasis web. Pengujian dilakukan dengan metode *stress testing*, dimana aplikasi diberikan beban yang besar untuk mengetahui apakah sistem dapat bekerja dengan baik walaupun diakses oleh banyak pengguna dalam waktu yang bersamaan [9].

Tujuan dari penelitian ini adalah membandingkan kinerja Oracle basis data *instance* tunggal dengan basis data *Real Application Cluster* (RAC) sebagai dasar pemilihan arsitektur basis data. Tahap penelitian diawali dengan Penentuan topologi sistem yang merupakan rancangan berupa gambar dari sistem dari penelitian.

Tahap kedua adalah instalasi dan konfigurasi sistem merupakan langkah-langkah dari instalasi dan konfigurasi *virtual machine*, *network*, storage *disk* untuk *server* dan basis data *instance* tunggal dan *Real Application Cluster* (RAC). Tahap ketiga adalah pengujian yang dilakukan untuk mendapatkan *throughput*, *response time* dan *error rate* dengan menggunakan Apache JMeter. Tahap keempat adalah analisis yang merupakan nilai rata-rata yang didapatkan dari hasil pengujian *throughput*, *response time* dan *error rate* yang dilakukan.



Gambar 1. Tahap Penelitian

METODE PENELITIAN

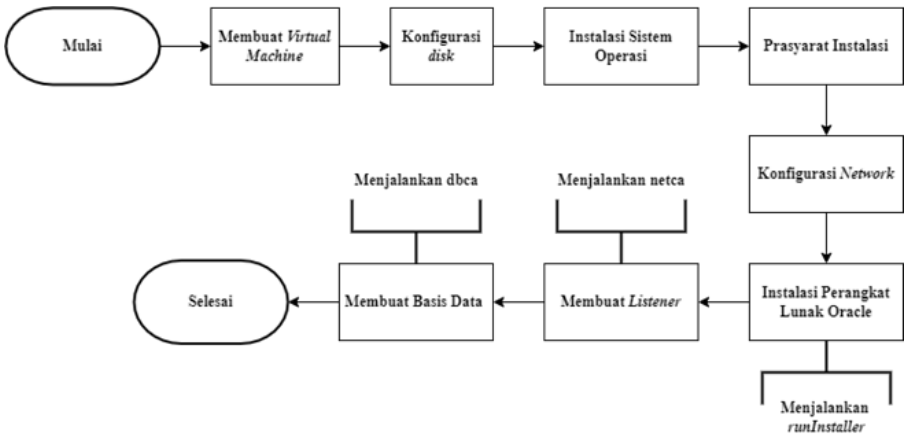
Penelitian ini terdiri dari empat tahap kegiatan yaitu: penentuan topologi sistem, instalasi dan konfigurasi sistem, pengujian dan analisis untuk dapat melakukan pengujian *throughput*, *response time* dan *error rate*. Pada Gambar 1 menunjukan tahap penelitian, penjelasan detail mengenai tahapan kegiatan penelitian yang terdiri dari penentuan topologi sistem, instalasi dan konfigurasi sistem, pengujian dan analisis, diuraikan sebagai berikut:

Penentuan Topologi Sistem

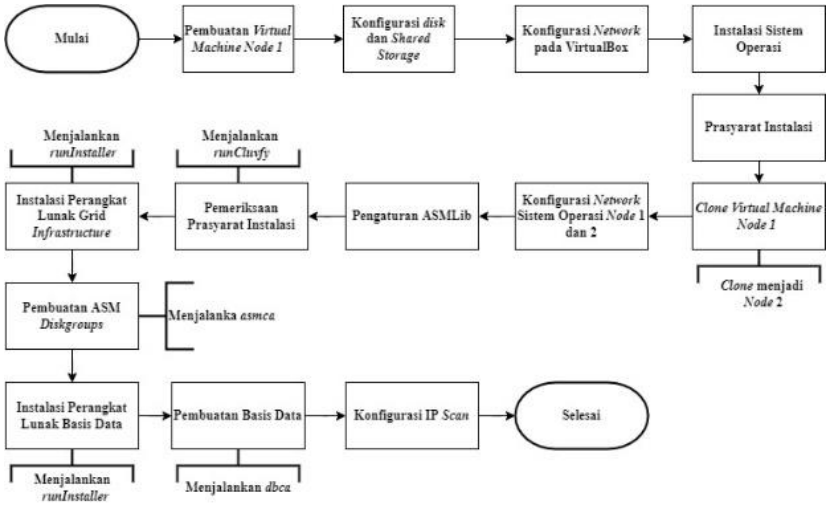
Topologi merupakan hubungan antara sistem dengan jaringan [10]. Topologi sistem terdiri dari uraian hubungan antar komputer, *server*, *storage* dan *network*. Topologi sistem pada penelitian ini terdiri dari dua bagian, yaitu topologi sistem pada basis data *instance* tunggal dan basis data *Real Application Cluster* (RAC).

Instalasi dan Konfigurasi

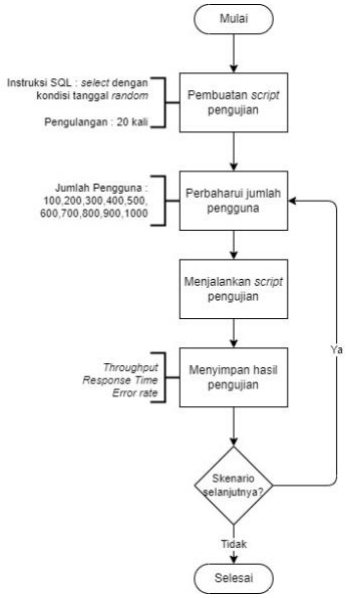
Instalasi dan konfigurasi merupakan pemasangan dan pengaturan komponen pada suatu sistem [11]. Tahap instalasi dan konfigurasi dilakukan untuk basis data *instance* tunggal dan basis data *Real Application Cluster* (RAC). Tahap Instalasi dan konfigurasi untuk basis data *instance* tunggal ditunjukkan pada Gambar 2. Tahap ini terdiri dari membuat *virtual machine*, melakukan konfigurasi *storage disk*, melakukan instalasi sistem operasi, melakukan prasyarat instalasi, melakukan konfigurasi *network*, melakukan instalasi perangkat lunak basis data dengan menggunakan perintah *runInstallter*, membuat *listener* dengan menggunakan perintah *netca* dan membuat basis data dengan perintah *dbca*.



Gambar 2. Tahap Instalasi dan Konfigurasi Basis Data *Instance* Tunggal



Gambar 3. Tahap Instalasi dan Konfigurasi Basis Data RAC



Gambar 4. Proses Pengujian

Pengujian

Pengujian merupakan suatu cara untuk mendapatkan sebuah informasi mengenai kualitas suatu sistem untuk mendeteksi adanya kesalahan yang menyebabkan kegagalan suatu sistem [12].

Proses pengujian dilakukan pada klien dengan menjalankan *load test* pada aplikasi JMeter [13] seperti ditunjukkan pada Gambar 4. Pengujian load test terdiri dari *throughput*, *response time* dan *error rate*. *Throughput* merupakan kemampuan basis data dalam melayani transaksi yang dibutuhkan klien dalam hitungan detik. *Response time* merupakan durasi waktu yang dibutuhkan oleh sistem ketika klien atau pengguna layanan mengirim permintaaan menuju *server*. *Error*

rate merupakan persentase kegagalan transaksi yang diminta oleh pengguna. Semakin sedikit *error rate* semakin baik kinerja *server* [14].

Pengujian dimulai dengan membuat *script* pengujian dengan menggunakan instruksi *simple mode* berupa *select* dan hanya melakukan *read only* dengan kondisi tanggal *random* dan pengulangan sebanyak 20 kali. Tahap kedua yaitu melakukan pembaruan pengujian dengan mengubah jumlah pengguna. Tahap ketiga yaitu menjalankan *script* pengujian. Tahap keempat yaitu menyimpan hasil pengujian yang terdiri dari *throughput*, *response time* dan *error rate*. Tahap kelima yaitu jika masih terdapat pengujan berdasarkan jumlah pengguna yang masih ada yang belum dilakukan, maka akan melakukan skenario selanjutnya dengan melakukan pengujian *script* kembali dengan memperbaharui jumlah pengguna dan jika pengujian berdasarkan jumlah pengguna sudah mencapai 1000, maka pengujian selesai dilakukan.

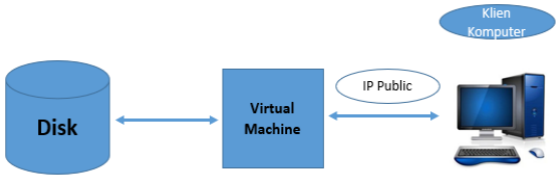
Analisis

Analisa merupakan proses mempelajari prosedur suatu sistem untuk mengidentifikasi tujuan dan sasarannya serta menciptakan prosedur yang digunakan untuk tujuan tersebut secara efisien [15]. Data yang diperoleh dan dikumpulkan akan disusun menjadi beberapa kriteria performa. Kriteria performa yang diperlukan tersebut yaitu nilai rata-rata *throughput*, *response time* dan *error rate*. Data-data tersebut dianalisis dengan mempresentasikan hasil perbandingan performa dari basis data Oracle *instance* tunggal dan basis data Oracle RAC berdasarkan *load test* untuk beberapa jumlah pengguna yaitu dimulai dari 100, 200, 300, 400, 500, 600, 700, 800, 900 dan 1000.

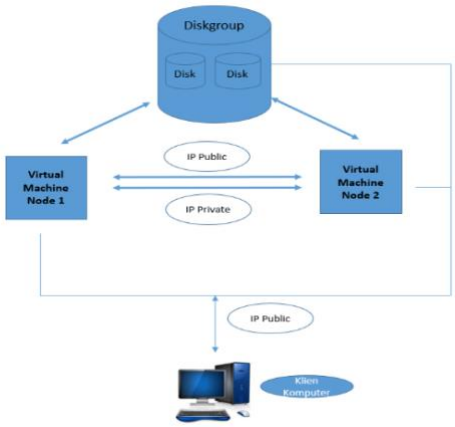
HASIL DAN PEMBAHASAN

Penentuan Topologi Sistem

Hasil penentuan topologi sistem terdiri dari topologi basis data Oracle *instance* tunggal dan basis data Oracle RAC. Berdasarkan pada Gambar 5 topologi basis data Oracle *instance* tunggal terdiri dari satu *node virtual machine*, satu *storage disk* dan klien komputer. *Virtual machine* mengakses basis data sebagai *storage disk*. Klien komputer akan melakukan akses melalui IP *Public* ke *virtual machine*.



Gambar 5. Topologi Sistem Basis Data Oracle *Instance* Tunggal



Gambar 6. Topologi Sistem Basis Data Oracle *RAC*

Tabel 1. Instalasi dan Konfigurasi Sistem Basis Data *Instance* Tunggal

No	Komponen	Parameter	Nilai
1	*Virtual Machine*	Nama	fsdbvm
		Lokasi file	D:/fsdbvm/fsdbvm
		Memory	4G
2	Sistem Operasi	Nama	Oracle Linux
3	*Storage*	*Controller* SATA	fsdbvm.vdi, fsdbvmdisk1.vdi, fsdb-data.vdi, fsdb-archive.vdi, fsdb-redo.vdi
4	*Network*	*Interface eth0*	192.168.10.30
5	Netca	Nama	Listener, PROD

Tabel 2. Instalasi dan Konfigurasi Sistem Basis Data RAC

No	Komponen	Parameter	Nilai
1	*Virtual Machine*	Nama	racvm01, racvm02
		Lokasi File	D:/racvm01/racvm01, D:/racvm02/racvm02
		Memory	4GB
2	Sistem Operasi	Nama	Oracle Linux
3	Storage	Controller SATA	racvm01.vdi, racu01.vdi, asmdisk01.vdi, asmdisk02.vdi, asmdisk03.vdi, asmdisk04.vdi, asmdisk05.vdi, asmdisk06.vdi, asmdisk07.vdi, asmdisk08.vdi
4	Network	*Interface eth0* racvm01	192.168.10.21
		Interface eth3 racvm02	192.168.10.22
		Interface eth1 racvm01	10.10.10.21
		Interface eth4	10.10.10.22
		IP VIP	192.168.10.24 192.168.10.25
		IP Scan	192.168.10.61 192.168.10.62
5	ASMCA	Nama *Diskgroup*	DG_DATA, DG_RECO, DG_CRS

Berdasarkan Gambar 6 topologi basis data Oracle RAC terdiri dari dua *node virtual machine*, setiap *node* terkoneksi oleh IP *private* digunakan *interconnect* antar *node* dan IP Public digunakan sebagai manajemen IP. Setiap *node* melakukan koneksi ke *storage disk* yang dikonfigurasi sebagai *shareable dan* ASM *diskgroup*. Klien komputer mendapatkan akses dari IP *public* yang dikonfigurasikan sebagai IP *scan*.

Instalasi dan Konfigurasi

Hasil instalasi dan konfigurasi sistem pada basis data *instance* tunggal terdiri dari hasil konfigurasi satu *virtual machine*, konfigurasi *storage disk*, instalasi sistem operasi, konfigurasi *network*, instalasi perangkat lunak dan konfigurasi basis data *instance* tunggal Oracle serta konfigurasi *netca* pada basis data.

Pada Gambar 7 menunjukan daftar list dari pembuatan *virtual machine* fsdb sebagai *virtual machine* basis data Oracle *instance* tunggal, racvm01 dan racvm01 sebagai *virtual machine* basis data Oracle RAC. Hasil konfigurasi *grid cluster* (RAC) dan basis data dapat dilihat pada Gambar 7.

```
[oracle@racvm02 ~]$ crsctl stat res -t
--------------------------------------------------------------------------------
NAME           TARGET  STATE        SERVER                   STATE_DETAILS
--------------------------------------------------------------------------------
Local Resources
--------------------------------------------------------------------------------
ora.DG_CRS.dg
               ONLINE  ONLINE       racvm01
               ONLINE  ONLINE       racvm02
ora.DG_DATA.dg
               ONLINE  ONLINE       racvm01
               ONLINE  ONLINE       racvm02
ora.DG_RECO.dg
               ONLINE  ONLINE       racvm01
               ONLINE  ONLINE       racvm02
ora.DG_VOTE.dg
               OFFLINE OFFLINE      racvm01
               OFFLINE OFFLINE      racvm02
ora.LISTENER.lsnr
               ONLINE  ONLINE       racvm01
               ONLINE  ONLINE       racvm02
ora.asm
               ONLINE  ONLINE       racvm01                  Started
               ONLINE  ONLINE       racvm02                  Started
ora.gsd
               OFFLINE OFFLINE      racvm01
               OFFLINE OFFLINE      racvm02
ora.net1.network
               ONLINE  ONLINE       racvm01
               ONLINE  ONLINE       racvm02
ora.ons
               ONLINE  ONLINE       racvm01
               ONLINE  ONLINE       racvm02
ora.registry.acfs
               ONLINE  ONLINE       racvm01
               ONLINE  ONLINE       racvm02
--------------------------------------------------------------------------------
Cluster Resources
--------------------------------------------------------------------------------
ora.LISTENER_SCAN1.lsnr
      1        ONLINE  ONLINE       racvm02
ora.LISTENER_SCAN2.lsnr
      1        ONLINE  ONLINE       racvm01
ora.LISTENER_SCAN3.lsnr
      1        ONLINE  ONLINE       racvm01
ora.cvu
      1        ONLINE  ONLINE       racvm01
ora.oc4j
      1        ONLINE  ONLINE       racvm01
ora.prod.db
      1        ONLINE  ONLINE       racvm01                  Open
      2        ONLINE  ONLINE       racvm02                  Open
ora.racvm01.vip
      1        ONLINE  ONLINE       racvm01
ora.racvm02.vip
      1        ONLINE  ONLINE       racvm02
ora.scan1.vip
      1        ONLINE  ONLINE       racvm02
ora.scan2.vip
      1        ONLINE  ONLINE       racvm01
ora.scan3.vip
      1        ONLINE  ONLINE       racvm01
[oracle@racvm02 ~]$
```

Gambar 7. Hasil Konfigurasi Basis Data RA

Hasil Pengujian

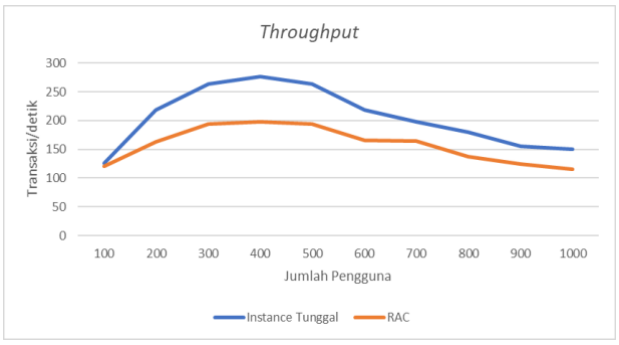
Hasil pengujian terdiri dari hasil pengujian *throughput*, *response time* dan *error rate* yang dikelompokan berdasarkan jumlah pengguna yaitu 100, 200, 300, 400, 500, 600, 700, 800, 900, dan 1000.

Hasil pengujian *throughput* ditunjukkan oleh Gambar 8.

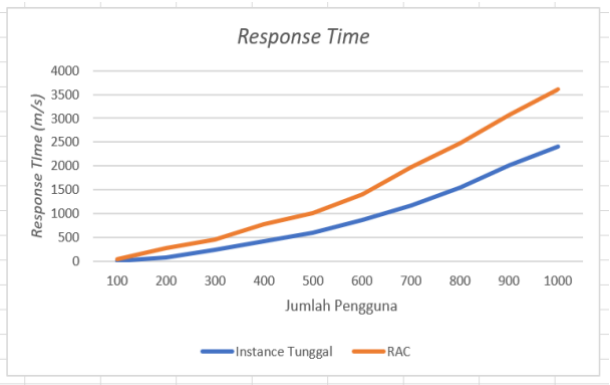
Hasil pengujian *response time* ditunjukkan oleh Gambar 9. Hasil pengujian *error rate* ditunjukkan oleh Gambar 10.

Analisis *Throughput*

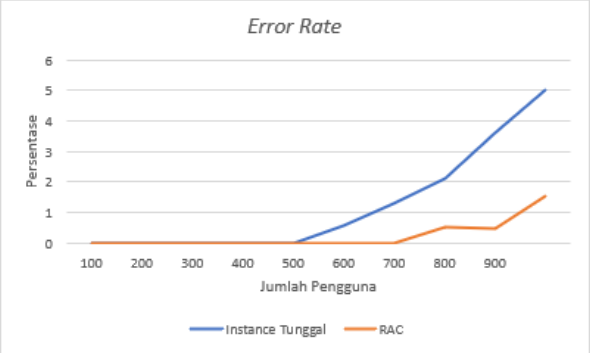
Data yang diperoleh dan dikumpulkan akan disusun menjadi beberapa kriteria performa. Kriteria performa yang diperlukan tersebut yaitu nilai rata-rata *throughput*, *response time* dan *error rate*. Data-data tersebut dianalisis dengan mempresentasikan hasil perbandingan performa dari basis data Oracle *instance* tunggal dan basis data Oracle RAC berdasarkan *load test* untuk beberapa jumlah pengguna yaitu dimulai dari 100, 200, 300, 400, 500, 600, 700, 800, 900 dan 1000.



Gambar 8. Grafik *Throughput*



Gambar 9. Grafik *Response Time*



Gambar 10. Grafik *Error Rate*

Tabel 3. Rata-rata Nilai *Throughput*

Jumlah Pengguna	*Loop*	*Throughput (t/s)*		Perbedaan
		Instance Tunggal	RAC	
100	20	126.3	120.6	4.51%
200	20	218.6	162.6	25.62%
300	20	263.4	193.8	26.42%
400	20	277.2	197.7	28.68%
500	20	264.4	194.4	26.48%
600	20	218.4	165.6	24.18%
700	20	197.8	164.9	16.63%
800	20	180	137.3	23.72%
900	20	155.8	124.3	20.22%
1000	20	150.1	115.5	23.05%

Hasil analisa merupakan Pada Tabel 3 terlihat bahwa basis data *instance* tunggal memiliki nilai *throughput* sebesar 277.2 transaksi per detik pada jumlah 400 pengguna aktif secara bersamaan. Pada basis data RAC memiliki nilai *throughput* tertinggi sebesar 197.7 transaksi per detik dengan jumlah pengguna aktif yang sama yaitu 400 pengguna. Persentase perbandingan dari kedua arsitektur basis data tersebut sebesar 28.68% dimana basis data *instance* tunggal memiliki performa *throughput* yang lebih tinggi dibandingkan dengan basis data RAC. Hal ini menunjukkan bahwa basis data *single instance* memiliki *throughput* yang lebih tinggi dibandingkan dengan basis data RAC.

Analisis *Reponse Time*

Berdasarkan Tabel 4 dapat terlihat bahwa pada pengguna aktif 100 memiliki pengguna secara bersamaan, nilai durasi waktu pada basis data *instance* sebesar 13,22ms dan durasi waktu pada basis data RAC sebesar 46,81ms.

Pada pengguna aktif 1000 pengguna secara bersamaan, hasil durasi waktu pada basis data RAC memiliki nilai durasi waktu paling tinggi sebesar 3613,39ms durasi waktu

pada basis data *instance* tunggal memiliki durasi waktu paling tinggi sebesar 2402,68ms. Pengguna aktif 100 pengguna memiliki persentase perbedaan perbandingan yang lebih besar yaitu sebesar 254.08%, sedangkan pada jumlah pengguna aktif 1000 pengguna memiliki persentase perbedaan perbandingan sebesar 50.39%. Pada persentase perbedaan setiap jumlah pengguna terlihat bahwa semakin banyak jumlah pengguna, nilai response time semakin kecil. Hal ini menunjukkan bahwa basis data *instance* tunggal memiliki durasi waktu lebih rendah dibandingkan dengan basis data RAC.

Tabel 4. Rata-rata Nilai *Response Time*

Jumlah Pengguna	*Loop*	*Response Time* (ms)		Perbedaan
		Instance Tunggal	RAC	
100	20	13.22	46.81	254.08%
200	20	86.38	277.39	221.13%
300	20	248.92	457.73	83.89%
400	20	413.03	782.26	89.40%
500	20	601.79	1005.92	67.15%
600	20	867.33	1398.45	61.24%
700	20	1166.3	1986.64	70.34%
800	20	1553.61	2473.83	59.23%
900	20	2021.9	3078.22	52.24%
1000	20	2402.68	3613.39	50.39%

Tabel 5. Rata-rata Nilai *Error Rate*

Jumlah Pengguna	*Loop*	*Error Rate* (%)		Perbedaan (%)
		Instance Tunggal	RAC	
100	20	0	0	0
200	20	0	0	0
300	20	0	0	0
400	20	0	0	0
500	20	0	0	0
600	20	0.57	0	0.57%
700	20	1.29	0	1.29%
800	20	2.12	0.51	1.61%
900	20	3.6	0.48	3.12%
1000	20	5.01	1.52	3.49%

Analisis *Error Rate*

Berdasarkan Tabel 5 perbedaan *error rate* basis data *instance* tunggal dan basis data RAC yaitu *error rate* pada basis data *instance* tunggal terjadi pada pengguna aktif 600 dengan nilai *error rate* sebesar 0,57%. *Error rate* pada basis data RAC *error rate* muncul pada jumlah pengguna aktif 800 dengan nilai *error rate* sebesar 0.51%. *Error rate* tertinggi terjadi pada 1000 pengguna aktif disaat bersamaan. *Error rate* pada basis data *instance* tunggal lebih tinggi dibandingkan basis data RAC yaitu dengan nilai 5.01% dibandingkan dengan nilai *error rate* basis data RAC sebesar 1.52%. Persentase perbedaan perbandingan pada *error rate* tertinggi antara arsitektur tersebut yaitu 3,49%. Hal ini menunjukkan bahwa *error rate* pada basis data *instance* tunggal lebih tinggi dibandingkan dengan *error rate* pada basis data RAC.

Pada penelitian yang dilakukan oleh Shrestha, pengujian performa dilakukan dengan melihat nilai dari *throughput* dan *response time*. *Throughput* dan *response time* lebih baik pada arsitektur *master-slave* dibandingkan dengan *cluster-based* pada Galera Mariadb. Pada penelitian yang dilakukan oleh Wei Ying, dengan melakukan performa RAC di dilakukan pada fisikal *server* dengan *fiber storage*, *internal* PCIE *storage*, dan VMWare dengan menggunkana VSAN, dengan melihat nilai TPM (*Transaction* per *minute*) dan IOPS. Hasil yang didapatkan yaitu TPM pada VSAN lebih baik dibandingkan dengan *fiber storage external* dan *internal* PCIE *storage*. Pada penelitian yang dilakukan oleh Mayang, pengujian pada aplikasi berbasis web. Pengujian dilakukan dengan metode *stress testing*, aplikasi diberikan beban yang besar untuk mengetahui apakah sistem dapat bekerja dengan baik jika diakses oleh banyak pengguna dalam waktu yang bersamaan. Pengujian performa dilakukan dengan hanya melihat *Error rate.* Hasil yang didapatkan *Error rate* pada website memiliki nilai yang lebih besar, banyak terjadi *error* (tidak *valid*) pada pengujian tersebut.

Penelitian ini menggunakan pengujian *load test* untuk mendapatkan nilai *throughput, response time* dan *error rate* sebagai dasar untuk pemilihan arsitektur basis data. Pada nilai rata-rata *throughput* dan *response time* menunjukkan bahwa basis data *instance* tunggal lebih unggul dalam hal penanganan transaksi per detik dan durasi untuk menjalankan transaksi. Pada nilai rata-rata *Error rate* menunjukkan bahwa basis data RAC lebih unggul dalam hal pelayanan transaksi yang lebih banyak tanpa menghasilkan transaksi *error (tidak valid).*

KESIMPULAN DAN SARAN

Perbandingan performa pada kedua arsitektur tersebut berhasil dilakukan dengan menggunakan *load test*. *Load test* dilakukan dengan menggunakan jumlah pengguna 100,200,300,400,500,600,700,800,900 dan 1000. Hasil dari pengujian untuk jumlah pengguna yang sama, basis data *instance*

tunggal lebih unggul dalam hal *throughput* dan *response time* dibandingkan dengan *cluster* RAC. Basis data RAC mendapatkan hasil yang lebih unggul dibandingkan basis data *instance* tunggal dalam hal performa *error rate*, hal ini menunjukkan bahwa basis data RAC dapat melayani transaksi yang lebih banyak tanpa menghasilkan transaksi *error* (tidak *valid*).

Pengujian yang dilakukan dalam penelitian ini belum menggunakan *server baremetal* (fisik) dan mesin yang digunakan belum memiliki sumber daya yang lebih banyak dalam alokasi *memory*, serta belum dilakukan *tuning* lebih lanjut. Oleh karena itu, hasil penelitian ini dapat ditingkatkan dengan melakukan pengujian pada server *baremetal* (*fisik*) dengan sumber daya yang lebih banyak serta melakukan *tuning* lebih lanjut untuk untuk mendapatkan *throughput*, *response time* dan *error rate* yang lebih baik serta untuk menambah performa secara keseluruhan, terutama penambahan performa pada pembagian *workload* ke setiap *node* pada RAC.

DAFTAR PUSTAKA

[1] C. Dhall. *Scalability Patterns: Best Practices for Designing High Volume Websites*. Germany: Apress, 2018.

[2] A. H. Ali, M. Z. Abdullah, "A survey on vertical and horizontal scaling platforms for big data analytics." *International Journal of Integrated Engineering*, vol. 11, no. 6, pp. 138-150, 2019.

[3] M. A. R. Maniyar, M. A. Hakeem, and M. K. M. Zafar, "Bottom-up Approach for Performance Testing of Software Applications or Products." *International Journal of Innovative Research in Computer and Communication Engineering* ,Vol. 6, Issue 9, Sep., pp. 7812-7817, 2018.

[4] N. Jha and R. Popli, "Comparative Analysis of Web Applications using JMeter." *International Journal of Advanced Research in Computer Science*, Vol. 8, Issue 3, Apr., pp 774-777, 2017.

[5] Sugiyatno, "Perancangan Clustering Database Server Untuk Meningkatkan Unjuk Kerja Server Dan Menjamin Ketersediaan Layanan," *Jurnal Cendikia*, Vol. 18, no. 1, Okt., pp. 281-289, 2019.

[6] D. A. Fansha, M. Y. H. Setyawan, and M. N. Fauzan, "Load Test pada Microservice yang menerapkan CQRS dan Event Sourcing," *Jurnal Buana Informatika*, Vol. 12, no. 2, Okt., pp. 126-134, 2021

[7] R. Shrestha, "High Availability and Performance of Database in the Cloud-Traditional Master-slave Replication versus Modern Cluster-based Solutions," In Proc. 7th International Conference on Cloud Computing and Services Science, 2017, pp. 413-420.

[8] W. Ying, "Oracle RAC performance Comparison on Physical Servers and

VMware VSAN Servers," In Proc. IOP Conference Series: Materials Science and Engineering, vol. 569, no. 3, 2019.

[9] M. A. Putri, H. N. Hadi, and F. Ramdani, "Performance testing analysis on web application: Study case student admission web system," in Proc. International conference on sustainable information engineering and technology (SIET), 2107, pp. 1-5.

[10] Ihwaldi, "Analisis Topologi Jaringan Pada Smk Muhammadiyah Palopo," *PhD diss.,* Universitas Cokroaminoto Palopo, South Sulawesi, 2020.

[11] A. G. Gani, "Konfigurasi Sistem Keamanan Jaringan," *JSI (Jurnal Sistem Informasi) Universitas Suryadarma*, Vol. 6, no. 1, pp. 134-149, 2019.

[12] N. N. A. Trisnawati, I.M. S. Putra, and A. A. K. O. Sudana, "Uji Fungsionalitas Sistem Informasi Manajemen Pegawai dengan Metode Black Box," *Jurnal Ilmiah Teknologi dan Komputer,* Vol. 2, no. 3, Des., pp. 524-534, 2021.

[13] S. Matam and J. Jain, *Pro Apache JMeter: Web Application Performance Testing.* United States: Apress, 2017.

[14] M. R. Maulana, E. B. Santoso, and S. W. Binabar, "Analisis Kinerja Website Pemerintah Kota Pekalongan," *Jurnal Litbang Kota Pekalongan,* Vol. 19, no. 1, pp. 48-54, 2021.

[15] D. Vincensius and B. Wasito, "Analisis dan Perancangan Sistem Informasi Point Of Sales pada CV. Sanjaya Abadi," *Jurnal Informatika dan Bisnis Institut Bisnis dan Informatika Kwik Kian Gie*, 2019.