PENINGKATAN AKSESIBILITAS DATA DENGAN MENERAPKAN GEO-REPLICATION DATABASE

Indra Warman
Teknik Informatika Fakultas Teknik, Institut Teknologi Padang
Padang, Sumatera Barat, Indonesia
e-mail : *indra@itp.ac.id

Abstract

Database replication is a critical technology that needs to be implemented to improve database performance for applications that require fast data access and high reliability with many users. Very large data query process requests by applications impact decreasing database server performance because many applications use only one database resource. One solution to this is replication, placing the database server and several replications in different locations, which allows database performance to be maximized. This requires a study of research related to replication strategies, especially the implementation of multi-location database geo-replication mechanisms, which aim to ensure the availability of services and replication processes run well and that consistency between database servers can be achieved. The increasing number of replicated databases with different geographic locations will make data access load requests from applications can be directed to several replication servers and reduce the delay time to user responses (low latency). This research implemented a geo-replication database for three database servers on the Ubuntu 22.04.4 LTS Virtual Private Server located in Singapore, West Jakarta, and South Jakarta, using the MariaDB database version 10.6.18. To measure the replication results related to time, consistency, and accessibility, a database replication query processing time test was carried out using queries with data from 250 to 150,000 records. The test results from the consistency side can be seen from the same number of data records on the master and slave replication databases, and from the accessibility side, data query requests can be served from three database server locations.

Keywords *: Database Replication, Geo-Replication Database, Sql Query*

PENDAHULUAN

Peningkatan serta pertumbuhan jumlah transaksi data yang sangat banyak menjadikan pengolahan data seperti data *e-commerce*, perbankan, *cloud computing* dengan menggunakan *database server* merupakan salah satu alasan perlunya menjaga skalabilitas serta kehandalan (Pratama et al., 2024) layanan, untuk menangani beragam permintaan *query* data yang banyak pada waktu bersamaan diproses hanya satu layanan *server database* (Zmaranda et al., 2021), hal ini menimbulkan masalah terkait permintaan proses *query* data yang sangat banyak oleh aplikasi, sehingga berdampak pada turunnya kinerja banyak aplikasi yang menggunakan satu *resource database (Ramu, 2023)*, karena semakin lamanya waktu respon dari *database server.*

Replikasi *database* merupakan proses menduplikasi dan menyebarkan data dari satu *server database* ke *server* lain (Pohanka & Pechanec, 2020), replikasi *database* merupakan teknologi yang sangat penting penerapannya untuk meningkatkan kinerja *database* bagi aplikasi yang membutuhkan akses data yang cepat serta reliabilitas tinggi (Khan et al., 2023), menjadikan *load* pemintaan akses *query* data tidak selalu terpusat sehingga kinerja *database* lebih optimal. Tantangan utama dengan *database* replikasi adalah bagaimana menjaga konsistensi data serta waktu respon antara satu *database* dengan replikasi *database* lainnya (Isiaka et al., 2020), beragam tipe replikasi seperti *master-slave*, serta metode yang digunakan *synchronous* atau *asynchronous*. Pada tipe replikasi *master-slave* jika terjadi perubahan *database* pada *master*, maka *database* pada *slave* juga ikut berubah(Azizah et al., 2022), namun jika terjadi perubahan pada *slave*, tidak akan mempengaruhi pada *database master*, dalam hal ini *master slave* dapat menerapkan metode *synchronous* replikasi.

Pada penelitian sebelumnya (Pohanka & Pechanec, 2020) melakukan evaluasi berbagai mekanisme replikasi yang diterapkan pada sistem *database* serta perbandingan antara sistem *database MySQL* dan *PostgreSQL* untuk meningkatkan ketersediaan data, sedangkan penelitian lainnya (Azizah et al., 2022) menjelaskan tentang perbandingan strategi replikasi yang digunakan dalam sistem *database* terdistribusi. Sistem terdistribusi sering kali menghadapi tantangan dalam hal konsistensi dan ketersediaan data (Nookala et al., 2022). Penelitian lain membahas tentang metode replikasi untuk memastikan ketersediaan layanan dan proses replikasi berjalan dengan baik serta ketercapaian konsistensi data (Sugiharto et al., 2020).

Meskipun pada penelitian tersebut lebih fokus pada mekanisme replikasi, perbandingan *database*, dan strategi skenario replikasi *multi-master* pada *database* terdistribusi, namun belum membahas tentang penerapan *geo-replication* untuk menghadapi tantangan konsistensi dan ketersediaan data pada multi lokasi, sehingga untuk memastikan konsistensi dan mengukur kinerja proses replikasi *database* perlu dilakukan penelitian dengan penerapan *geo-replication database*.

Salah satu fungsi utama replikasi data adalah memastikan redundansi data yaitu memiliki lebih dari satu salinan data pada lokasi *database server* yang berbeda, dengan tujuan memastikan bahwa data tetap tersedia walaupun terjadi kegagalan sistem (Wulandari & Adji, 2021) pada satu lokasi, hal ini sangat penting untuk memastikan integritas data dapat terlindungi dari kehilangan data yang tidak diinginkan.

Penggunaan model replikasi untuk kehandalan layanan ada 2 model, seperti replikasi yang dilakukan pada satu lokasi *database server* dan model replikasi dengan penempatan data pada lokasi geografis berbeda (Putra et al., 2023), pemilihan terhadap model replikasi tentunya disesuaikan dengan kebutuhan dengan mempertimbangkan database replikasi latensi (Campêlo et al., 2020).

Proses sinkronisasi data dalam replikasi *database* mempunyai peran penting untuk layanan banyak pengguna (Nama et al., 2023), dengan tujuan meningkatkan respon terhadap data oleh beragam aplikasi dengan lokasi yang berbeda, hal ini tentunya dapat mengurangi latensi dan meningkatkan respon aplikasi serta memastikan ketersediaan data / *data availability* (Lestariningsih et al., 2024)

Mariadb sebagai *database management system (DBMS)* yang mendukung skema replikasi, mempunyai kelebihan dan menyediakan mekanisme dalam melakukan replikasi *database* terutama *geo-replication database* (Shrestha & Tandel, 2023) dan menyediakan fitur *log monitoring* serta *binary log* yang akan melakukan pencatatan aktifitas proses replikasi *database* serta perubahan data pada *binary log*, untuk aplikasi dengan permintaan *query* yang banyak dapat diatur atau dialihkan ke *server slave* dengan *resource server database* yang lebih baik (Vignesh et al., 2020).

Teknologi *VPS (Virtual Private Server)* merupakan layanan *server* yang dapat dikelola sendiri untuk penyimpanan data yang ditempatkan berbagai lokasi atau berbeda negara, teknologi *VPS* menerapkan opsi *scale up* yang memungkinkan pengelola *server database* dapat memperluas layanan sesuai dengan pertumbuhan bisnis, dengan cara antara *master database* dan beberapa replikasi *database* ditempatkan di lokasi berbeda (*geo-replication*), sehingga beban *query* tidak tertumpu pada satu lokasi *server* dan kinerja *server* dapat dimaksimalkan (Šušter & Ranisavljević, 2023). Menggunakan penyimpanan salinan data di beberapa lokasi *database server* menjadikan penanganan permintaan data dengan lebih cepat dan meminimalisasi resiko kehilangan data jika terjadi bencana dengan konsep replikasi beda wilayah geografis/negara (Patra & Goswami, 2022).

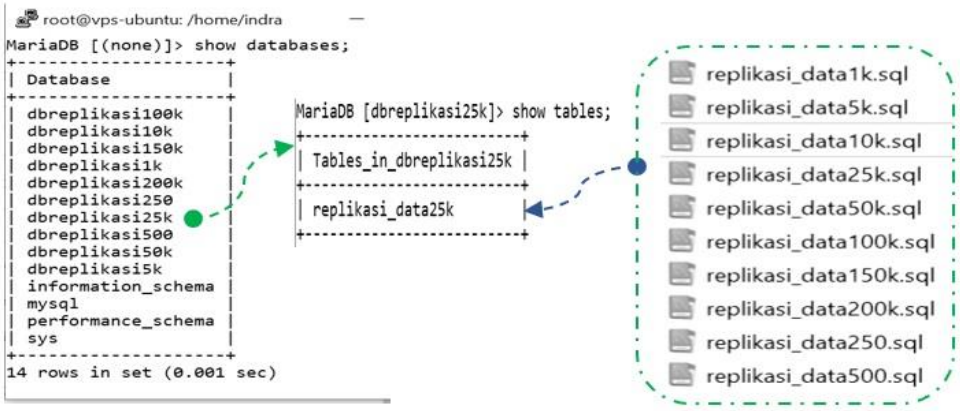
Penelitian ini menerapkan teknik replikasi *database* dengan pendekatan *geo-replication* pada 3 lokasi *database server* yaitu *Singapore*, Jakarta Barat & Jakarta Selatan, yang bertujuan untuk mengurangi waktu tunda terhadap respon dari pengguna (latensi rendah) serta memastikan konsistensi data.

METODE PENELITIAN

Pada penelitian ini dilakukan dengan metode eksperimen untuk mengamati proses penerapan *geo-replication database* lintas negara / multi lokasi, ini dilakukan untuk memastikan ketercapaian konsistensi data antara *database master* dan *replikasi* dapat dilakukan dengan baik (Ambara et al., 2020) serta mengukur proses replikasi untuk 3 *database* dengan melakukan 7 tahapan pengujian beban / *load SQL query,* jumlah *record* data dari 100 s,d 150.000 *record* data (1 m*aster database* dan 2 *replikasi/slave*). Pengujian menggunakan 3 *database server Virtual Private Server (VPS)* yang terdapat di *Singapore*, Jakarta Barat & Jakarta Selatan, hasil pengukuran proses replikasi yang telah disimpan pada *file log server* dari 3 *server geo-replication dapat diakses* menggunakan aplikasi *web log monitoring*.

Penelitian ini dilakukan berdasarkan tahapan berikut :

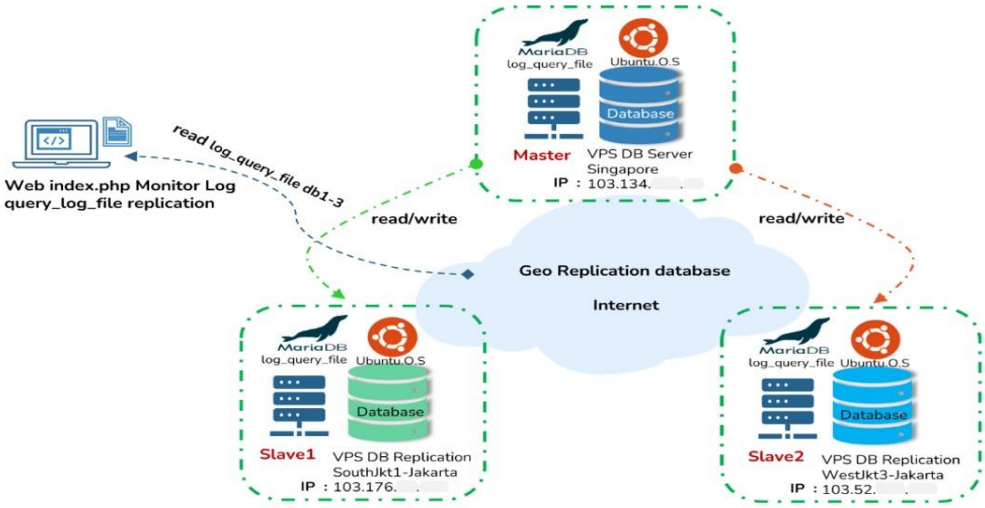
1. Menyiapkan serta konfigurasi *database geo-replication* untuk 3 lokasi *server* (*Singapore*, Jakarta Barat & Jakarta Selatan), dengan spesifikasi menggunakan *Ubuntu 22.04.4 LTS Kernel: Linux 5.15.0-125* dengan *Processor 2VCPU, Memory 2048MB dan Space 20GB* untuk masing-masing *server* multi lokasi, *database server* menggunakan *MariaDB* versi 10.6.18.
2. Menyiapkan kebutuhan *database* untuk pengujian *query* replikasi, pada pengujian disiapkan 7 *database*, pada *database* terdapat *table* yang nantinya akan dilakukan *inse*rt data *file .sql* dengan *record* data yang berbeda. Untuk memudahkan pengujian *query* dibuatkan nama *database* sesuai dengan ukuran jumlah *record* data pengujian seperti pada gambar 1 berikut :



Gambar 1. Rancangan *database table* dan pengujian *load query SQL* replikasi

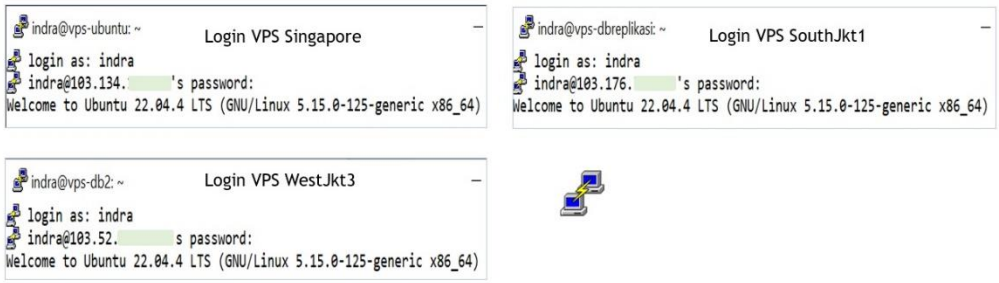
3. Menentukan konfigurasi masing-masing *database server* dengan opsi *server* sebagai *database master* atau *slave* pada *database mariadb server* di server *VPS Ubuntu 2.04.4 LTS*.
4. Skenario rencana penelitian *geo-replication database* lintas negara / multi lokasi dengan menempatkan 3 *server database* yang terdapat di *Singapore* sebagai *database master*, Jakarta Barat sebagai *slave replication-1* dan Jakarta Selatan *sebagai slave replication-2*, pengujian replikasi dilakukan dengan sejumlah *record* data *query* dari *master replication* ke masing-masing *database* replikasi, yang disimpan pada *server log query file*, selanjutnya dijadikan sebagai informasi hasil proses replikasi untuk dilakukan *monitoring* menggunakan aplikasi *web*, seperti pada gambar 2 berikut .

P-ISSN: 2622-6901
E-ISSN: 2622-6375



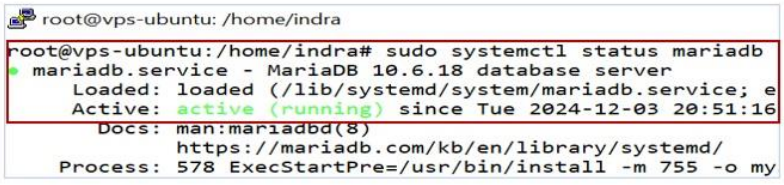
Gambar 2. Implementasi *geo-replication database* pada *VPS server database*

5. Konfigurasi *Database Server Master VPS Server*
   Untuk melakukan *setup database server* seperti konfigurasi *server id*, *log bin, bind address*, serta membuat *database* dan *table*, dilakukan dengan akses ke *IP Public server VPS Ubuntu 22.04.4* dengan menggunakan *putty SSH* ke *server VPS Singapore* , *SouthJkt1*, dan *Westjkt3* seperti gambar 3 berikut.



Gambar 3. *Setup login* akses 3 *Server Database* pada *O.S Ubuntu 22.04.4*

Setelah *login* menggunakan akses *putty SSH* ke masing-masing *database server*, pada tahapan ini semua *service mariadb server running* harus sudah *active (running)* untuk nantinya dilakukan proses *query database* dan replikasi, seperti gambar 4 berikut.

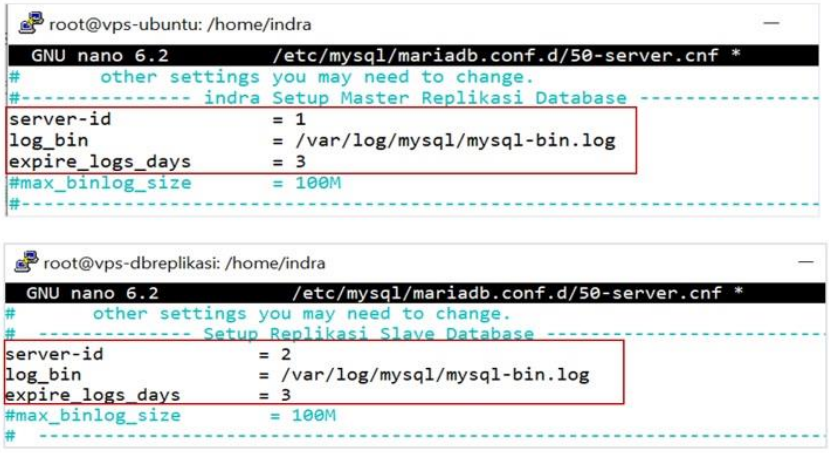


Gambar 4. *Setup service Active (running) database mariadb* pada *O.S Ubuntu 22.04.4*

6. Pengaturan *VPS server* replikasi *master* dan *slave geo-replication*
   Untuk melakukan konfigurasi *database Mariadb master & slave* pada *OS Ubuntu 2.04.4 LTS*, dimana dilakukan penentuan lokasi / negara *VPS server database* yang akan dijadikan sebagai m*aster* dan *slave*, berikut merupakan konfigurasi yang dilakukan pada *setup database* pada *mariadb.conf.d* dengan menentukan opsi *server* sebagai *server database master dengan*

P-ISSN: 2622-6901
E-ISSN: 2622-6375

server-id=1 dan slave, dengan melakukan konfigurasi pada *file /etc/mysql/mariadb.conf.d/ 50-server.cnf*, seperti gambar 5 berikut.

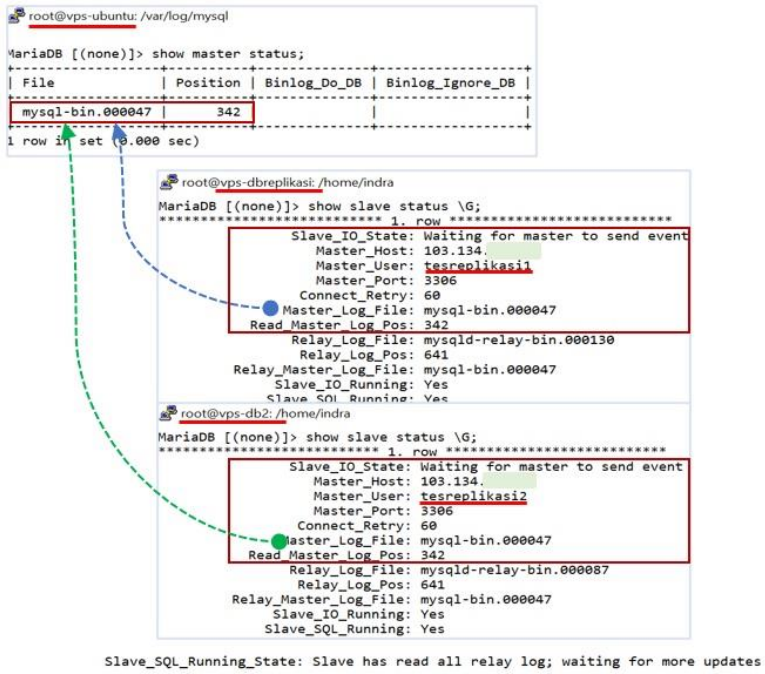


Gambar 5. Konfigurasi *master / slave server-id log bin database*

7. Pengaturan *Grant Privilege* pada *IP Address database server*
   Pengaturan *privilege* diperlukan untuk memastikan *server database* berhak untuk melakukan replikasi (Zambrano & Gonzalez, 2024), dengan cara melakukan *create user, host serta ip address public* yang telah ditetapkan untuk diberikan izin pada *server database master*, pada gambar 6 berikut ditampikan *user* dengan nama tesreplikasi1, tesreplikasi2 dan *host IP public server slave* replikasi 1 dan 2, yang telah ditambahkan dengan perintah *grant replication slave on.*

Gambar 6. Pengaturan *grant replication user database privileges IP & Host*

8. Pengaturan *master* replikasi informasi status *log file*
   Sebelum dilakukan konfigurasi pada *database server slave geo replication, untuk menentukan source dari master host, user, log_file, log_position*, terlebih dahulu didapatkan informasi mengenai status *log file biner* dan status *log pos* pada *master* replikasi *database server*, nantinya status *log* akan digunakan sebagai referensi pada *database slave*. Pada gambar 7 berikut menjelaskan *file mysql-bin.000047* dengan *position* 342 yang telah ditambahkan pada masing-masing konfigurasi *slave* yang sudah terkoneksi dengan informasi pada *slave* status adalah *waiting for master to send event.*



Gambar 7. Informasi status *log file bin & log pos slave SQL running*

9. Pengujian *database* replikasi

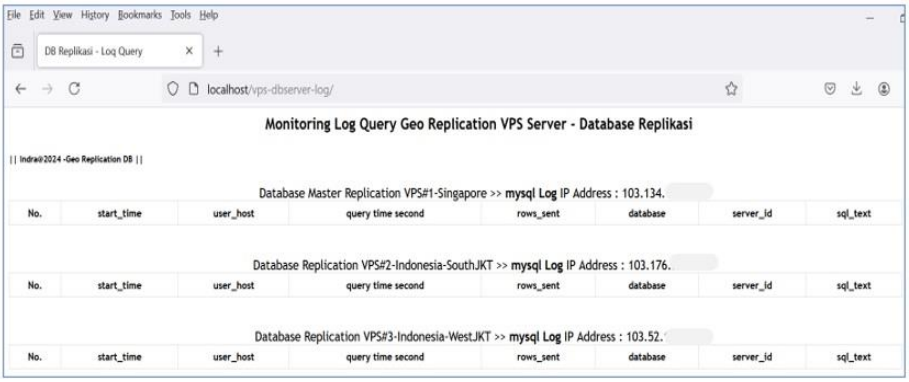
Setelah semua tahapan konfigurasi dan status untuk replikasi aktif dengan informasi status pada masing-masing *slave* adalah "*waiting for master to send event*" seperti pada gambar 7 diatas, selanjutnya dilakukan tahapan pengujian sesuai dengan rencana yang telah ditetapkan seperti pada rancangan *database table* dan pengujian q*uery database,* tahap awal dilakukan *query database* untuk *create database* serta *table* pada master *database* replikasi, setiap *database* dilakukan *query insert* data dari 250 sampai dengan 150.000 *record* untuk dilakukan pengujian proses replikasi, berikut gambar 8 menjelaskan *database* replikasi pada masing-masing *database server.*

Gambar 8. Pengujian *replikasi database master slave geo replication.*

10. Pembuatan *web log monitoring* aktivitas replikasi *database*

Hasil pengujian dari proses *query database* yang dilakukan perlu dianalisa untuk memastikan replikasi sesuai, dengan cara membuat *web monitoring log query geo-replication VPS server*

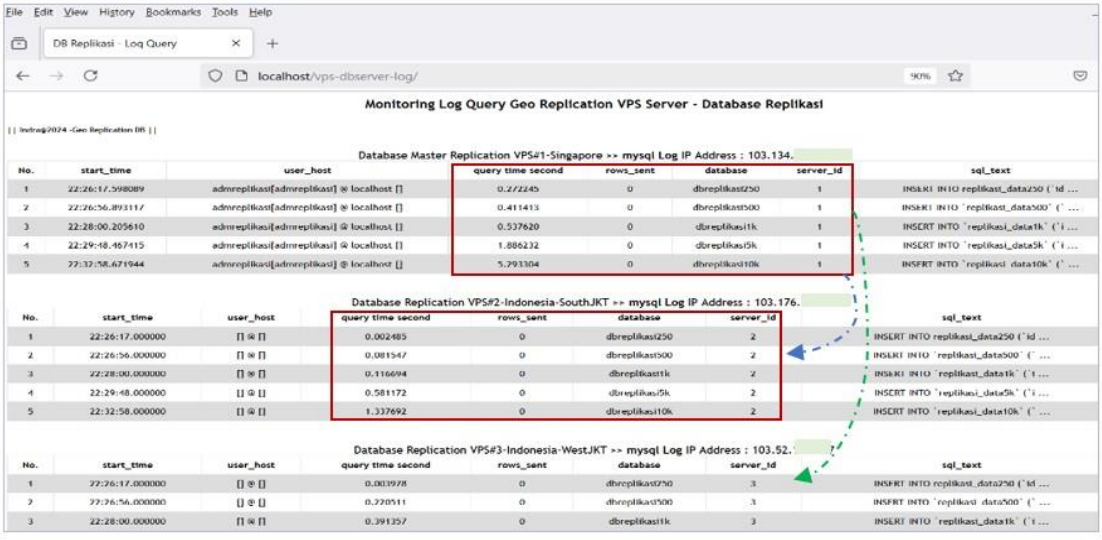
yang terkoneksi dengan *file log* untuk mencatat aktifitas replikasi yang terjadi, seperti *user host, query time, database dan server id* dari 3 *database* tersebut, berikut gambar 9 merupakan tampilan *log monitoring* sebelum adanya proses replikasi *database*.



Gambar 9. *Web Monitoring log query geo-replication database server.*

HASIL DAN PEMBAHASAN

Berikut merupakan pengujian pertama yang dilakukan untuk *query database insert* data mulai dari 250, 500, 1000, 5000 dan 10.000 *record* data pada *master database server*, selanjutnya *log* proses dan waktu replikasi yang terjadi pada 3 *database server Singapore*, Jakarta Selatan (*SouthJkt1*) & Jakarta Barat (*WestJkt3*) disimpan dan ditampilkan pada *web monitoring log query geo-replication* sebagai bahan analisa terkait konsistensi data, pada gambar 10 menunjukkan *log* proses hasil replikasi yang terjadi pada 3 *database* dengan hasil *record* data yang sama untuk *database master* dan *slave* replikasi.



Gambar 10. *Web log monitoring* hasil pengujian *insert* data 250-10.000 *record* data

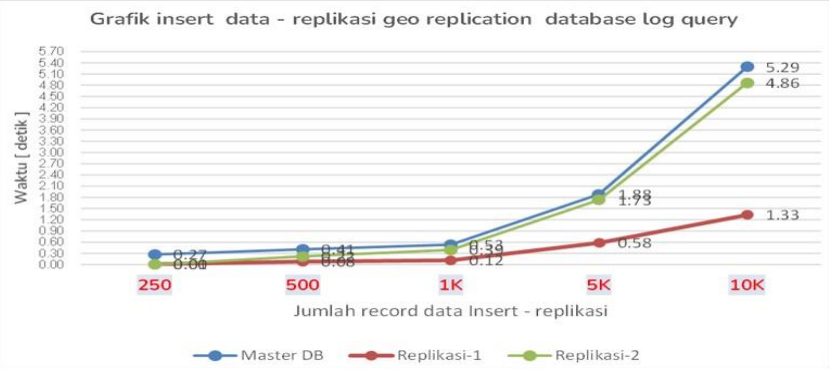
dari data hasil *log* proses replikasi dibuatkan tabel hasil dari waktu proses replikasi yang terjadi berdasarkan jumlah *insert record* data dan lama proses *insert data* yang dilakukan pada masing-masing *database server*, seperti pada tabel 1 berikut.

P-ISSN: 2622-6901
E-ISSN: 2622-6375

Tabel 1. Hasil pengujian *Load Query Insert* data serta waktu replikasi *master slave*

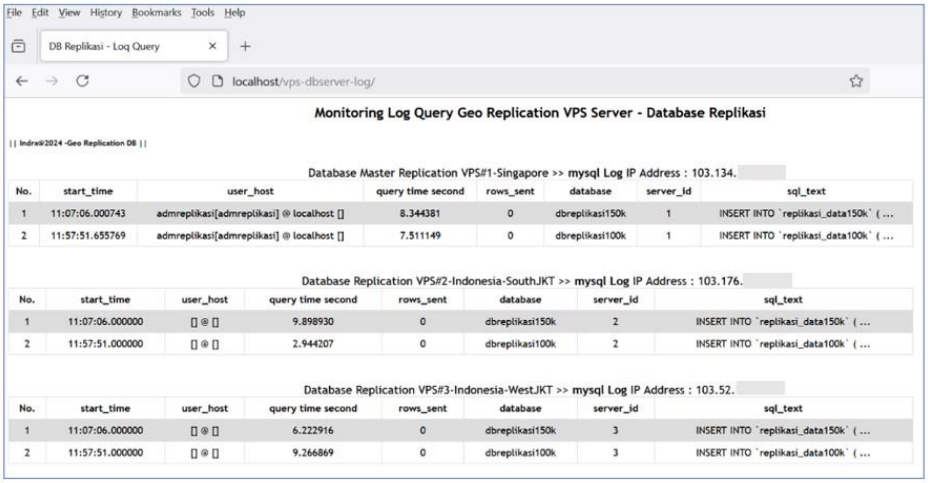
Jumlah *record* data	*Master DB* (detik)	Replikasi-1 (detik)	Replikasi-2 (detik)
250	0.27	0.01	0.00
500	0.41	0.08	0.22
1000 (1K)	0.53	0.12	0.39
5000 (5K)	1.88	0.58	1.73
10000(10K)	5.29	1.33	4.86

Dari gambar 11 dapat dijelaskan bahwa hasil *insert record* data replikasi pada *table database*, seperti pada jumlah 10.000 *record* pada *database master* dibutuhkan waktu 5.29 detik, sedangkan untuk waktu replikasi jumlah *record* data yang sama pada *slave* replikasi-1 1.33 detik dan replikasi-2 4.86 detik, dengan hasil replikasi jumlah *record* data yang sama pada *database master* dan *slave* replikasi.



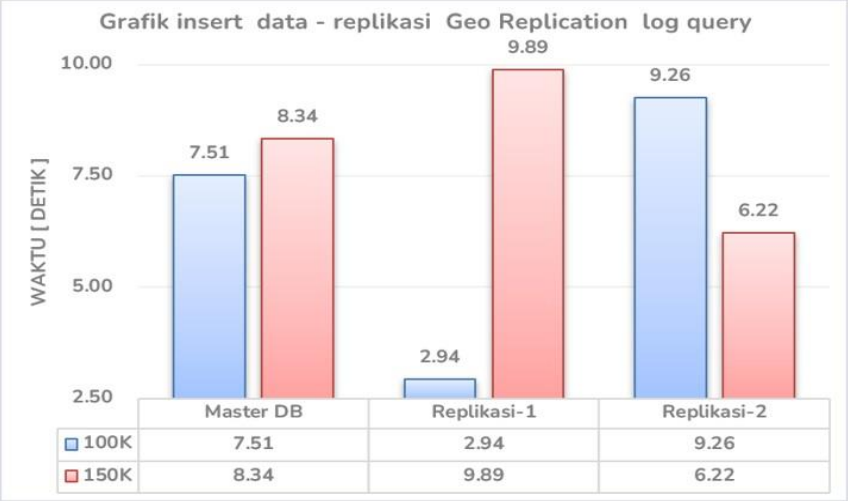
Gambar 11. Grafik waktu proses pengujian *insert* data 250 – 10.000 *record*

Pada gambar 12 menunjukkan *log* proses replikasi yang terjadi pada 3 *database* dengan hasil *record* data yang sama untuk replikasi *database master* dan *slave* pada pengujian kedua, dengan jumlah *insert record* data 100.000 dan 150.000, untuk *user host* pengguna, lama *query, database* yang digunakan, waktu replikasi, serta perintah *query* disimpan dan ditampilkan pada *web monitoring log query geo replication.*



Gambar 12. *Web log monitoring* hasil pengujian *insert* data 100.000 -150.000 *record*

Pada gambar 13 menampilkan jumlah *record* data dan waktu proses replikasi yang terjadi pada 3 *database*, dengan hasil *record* data yang sama untuk replikasi *database master* dan *slave* pada jumlah *query insert record* data 100.000 dan 150.000 dengan waktu proses *insert* data *master* 8.34 detik, sedangkan pada *database server* replikasi-1 2.94 detik dan replikasi-2 6.86 detik.



Gambar 13. Grafik jumlah *record* 100.000 dan 150.000 & waktu *insert data.*

KESIMPULAN

Penelitian ini telah berhasil menerapkan teknik replikasi *database* dengan pendekatan *geo-replication* pada 3 lokasi *database server*. Hasil pengujian dari sisi konsistensi terlihat dari jumlah *record* data yang sama pada *database master* dan *slave* replikasi, dan sisi aksesibilitas permintaan *query* data dapat dilayani dari 3 lokasi *database server.*

SARAN

Berdasarkan hasil penelitian serta kesimpulan, dapat disarankan untuk dilakukan penggunaan model replikasi *database* dengan skema fleksibel yang mendukung data tidak terstruktur (*non-relational / NoSQL)* pada *geo-replication*.

DAFTAR PUSTAKA

Ambara, M. P., Widiartana, P. K., & Atmojo, Y. P. (2020). Implementasi Socket Programming Sebagai Media Sinkronisasi Database Terdistribusi dengan Teknik Multi Master Replication. *Jurnal Sistem Dan Informatika (JSI)*, *14*(2), 113–124.

Azizah, N., Hartajaya, V., & Riady, S. (2022). Comparison of replication strategies on distributed database systems. *International Journal of Cyber and IT Service Management*, *2*(1), 20–29.

Campêlo, R. A., Casanova, M. A., Guedes, D. O., & Laender, A. H. F. (2020). A brief survey on replica consistency in cloud environments. *Journal of Internet Services and Applications*, *11*, 1–13.

Isiaka, F. M., Audu, S. A., & Umar, M. A. (2020). Developing a fail-safe culture in a cyber environment using MySQL replication technique. *International Journal of Crowd Science*, *4*(2), 149–170.

Khan, W., Kumar, T., Zhang, C., Raj, K., Roy, A. M., & Luo, B. (2023). SQL and NoSQL database software architecture performance analysis and assessments—a systematic literature review. *Big Data and Cognitive Computing*, *7*(2), 97.

Lestariningsih, E., Redjeki, R. S., Supriyanto, E., Murti, H., Ardhianto, E., Wahyudi, E. N., & Handoko, W. T. (2024). Studi Literasi Tersistematis Implementasi Metode-Metode Pengembangan Sistem Informasi Berbasis Basis Data Terdistribusi. *Jurnal Informatika Dan Rekayasa Perangkat Lunak*, *6*(1), 56–61.

Nama, P., Bhoyar, M., Chinta, S., & Reddy, P. (2023). Optimizing Database Replication Strategies through Machine Learning for Enhanced Fault Tolerance in Cloud-Based Environments. *Machine Learning (ML)*, *63*(3).

Nookala, G., Gade, K. R., Dulam, N., & Thumburu, S. K. R. (2022). The Shift Towards Distributed Data Architectures in Cloud Environments. *Innovative Computer Sciences Journal*, *8*(1).

Patra, S. S., & Goswami, V. (2022). Performance Enhancement of Cloud Datacenters Through Replicated Database Server. *Journal of Information Technology Research (JITR)*, *15*(1), 1–23.

Pohanka, T., & Pechanec, V. (2020). Evaluation of replication mechanisms on selected database systems. *ISPRS International Journal of Geo-Information*, *9*(4), 249.

Pratama, B. P. A., Hardianto, B. G., & Dharmayanti, D. (2024). Implementation of High Availability Based on Load Balancing and Failover Method in E-commerce Using MySQL Database. *Asian Journal of Engineering, Social and Health*, *3*(10), 2226–2239.

Putra, T. O. D., Ansyah, A. S. S., Arifin, M., & Ijtihadie, R. M. (2023). Geo-Replication Dalam Tinjuaun Latensi Dan Efektifitas Biaya. *Jurnal Ilmiah Teknologi Informasi*, *21*(2), 98–108. https://doi.org/10.12962/j24068535.v21i2.a1165

Ramu, V. B. (2023). Optimizing Database Performance: Strategies for Efficient Query Execution and Resource Utilization. *International Journal of Computer Trends and Technology*, *71*(7), 15–21.

Shrestha, R., & Tandel, T. (2023). An Evaluation Method and Comparison of Modern Cluster-Based Highly Available Database Solutions. *CLOSER*, 131–138.

Sugiharto, W. H., Ghozali, M. I., Susanto, H., Budihardjo, M. A., & Suryono, S. (2020). Database replication method for real-time measurement pH parameter of fishery using a wireless sensor system. *Journal of Physics: Conference Series*, *1524*(1), 012021.

Šušter, I., & Ranisavljević, T. (2023). Optimization of MySQL database. *Journal of Process Management and New Technologies*, *11*(1–2), 141–151.

Vignesh, R., Deepa, D., Anitha, P., Divya, S., & Roobini, S. (2020). Dynamic enforcement of causal consistency for a geo-replicated cloud storage system. *International Journal of Electrical Engineering and Technology*, *11*(3).

Wulandari, S., & Adji, G. S. (2021). Distributed Database Implementation in Point of Sale System with Method Synchronous Replication. *International Journal of Engineering Technology and Natural Sciences*, *3*(2), 47–51.

Zambrano, M., & Gonzalez, V. (2024). Database's Disaster Recovery Meets a Ransomware Attack. *19th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'23), Cape Town, South Africa, 09-13 October 2023*, 280–284.

Zmaranda, D. R., Moisi, C. I., Győrödi, C. A., Győrödi, R. Ş., & Bandici, L. (2021). An analysis of the performance and configuration features of MySQL document store and elasticsearch as an alternative backend in a data replication solution. *Applied Sciences*, *11*(24), 11590.