JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 6 No. 2, Edisi Januari-April 2025, Page 1455-1460
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Rekontruksi Arsitektur DataBase untuk Peningkatan Proses Load Data

Suriansyah B[1*], Luqman Fanani Mz[2], Andi Ikmal Rachman[3], Gita Pratiwi[4]

[1,2,3,4]Sistem Informasi, Unversitas Almarisah Madanai, Makassar Sulawesi Selatan, Indonesia

Email: [1],*suriansyahb@univeral.ac.id, [2]luqmanfmz@univeral.ac.id, [3]andiikmal@univeral.ac.id, [4]Gitapratiwi@univeral.ac.id
Email Penulis Korespondensi: [1]suriansyahbachir@gmail.com

Abstrak− Proses load data menjadi aspek yang sangat penting dalam pengelolaan sistem informasi yang melibatkan basis data besar. Namun, dengan meningkatnya volume data dan kompleksitas sistem, waktu yang dibutuhkan untuk memuat data menjadi semakin lama dan menghambat efisiensi operasional . maka dari itu kami akan memberi solusi dengan membahas pendekatan rekonstruksi arsitektur database untuk meningkatkan performa proses load data, dengan fokus pada optimasi pemrosesan dan pengelolaan data. Penelitian ini mengidentifikasi tantangan utama dalam arsitektur database tradisional, dan mengeksplorasi solusi berbasis teknologi terkini seperti penggunaan partisi, indeks, dan pipeline ETL yang lebih efisien. Dengan merubah arsitekture database dari sebelumnya penerapan ini meningkatkan kecapatan load data 51% dari arsitekture sebelumnya .

Kata Kunci: Rekonstruksi arsitektur, load data, basis data, optimasi, ETL, partisi data, indeks

Abstract−.The process of loading data is a crucial aspect of managing information systems that involve large databases. However, with the increasing volume of data and system complexity, the time required to load data has become longer, hindering operational efficiency. Therefore, we will provide a solution by discussing a database architecture reconstruction approach to enhance the performance of the data loading process, focusing on optimizing data processing and management. This research identifies the main challenges in traditional database architectures and explores solutions based on the latest technologies, such as the use of partitioning, indexing, and more efficient ETL pipelines. By altering the database architecture from its previous implementation, this approach has increased data loading speed by 51% compared to the previous architecture.

Keywords: Reconstruction of architecture, data loading, database, optimization, ETL, data partitioning, indexing

1. PENDAHULUAN

Perkembangan ilmu komputer merupakan bagian penting dalam era teknologi saat ini. Berbagai teknik dan metode telah dibuat dengan penerapan ilmu komputer. Khususnya dalam teknik pengolahan basis data, banyak perusahaan menerapkan teknologi ini untuk mendukung sistem informasi agar menguntungkan analisis bisnis mereka. dan menggunakan data warehouse sebagai pusat data. Hal ini dilakukan untuk membantu ketersediaan data dan saat loading ke dalam suatu sistem informasi untuk memantau kinerja bisnis. Namun, seiring berjalannya waktu, transaksi akan semakin banyak tercatat. Sehingga data akan terkumpul dalam tabel-tabel. [1], [2], [3] Hal ini memberatkan proses query data yang akan diload ke dalam sistem informasi karena akan membutuhkan biaya waktu yang cukup besar.

Dan terdapat dilema yang di alami para praktisi data yaitu mempertahankan arsitekture database yang lama namun kinerjanya lambat, maka dalam paper ini penulis akan menjelaskan bagaimana rekontruksi aritekture database di lakukan untuk meningkatkan proses load data. Dari beberapa penelitan sebelumnya tentang data base Seperti pada paper yang berjudul "optimization of data Warehouse Architekture to improve information sistem performance" [4]dalam paper ini menjelaskan tantang proses ETL yang di optimasi dengan fokus pada penjadwalan load data penelitian lain mengenai database yaitu "Implementasi Database Massively Parallel Processing System untuk Membangun Skalabilitas pada Process Data Warehouse"[5]. Pembahasan dari jurnal ini adalah mengenai data warehouse dan pertumbuhan data yang banyak sehingga membutuhkan waktu biaya yang lebih banyak, maka sebagai solusinya diterapkan sistem MPP untuk penyajian data yang cepat. Penelitian lain mengenai optimasi data warehouse yang berjudul "Optimizing the performance of Data Warehouse by Query Cache Machanism" [6] dalam paper ini membahas bagaimana mengurangi beban pada proses extract transfer dan load data ke dalam data warehouse dengan menggunakan metode heuristic kemudian menganalisis seberapa baik kinerja heuristic dengan membandingkan sebelum dan sesudah optimasi proses ETL. Penelitian lain yang membahas mengenai optimasi load data berjudul "evaluating engagement and bucketing strategies for Hive-based big data warehousing systems"[7]. Tujuan dari penelitian ini adalah menganalisis optimasi query pada tabel-tabel yang dipartisi dan di-bucket. Dari penelitian ini, partitioning dan bucketing memberikan tambahan kecepatan ketika meng-query data. Namun, jika data yang digunakan masih manageable, bucketing memperlambat kinerja query . Penelitian lain yang membahas mengenai proses ETL ke data warehouse dengan menggunakan scheduling berjudul "ETL Scheduling in Real-Time data warehousing"[8]. Penelitian lain yang membahas mengenai penjadwalan beban data berjudul "Improved heuristic job scheduling method to enhance throughput for big data analytics" [9] dalam penelitian ini membandingkan metode Shortest Job First (SJF) dengan metode pengembangan

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024 Submitted: **15/01/2024**; Accepted **25/01/2024;** Published: **18/02/2025**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 6 No. 2, Edisi Januari-April 2025, Page 1455-1460
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Denset Job Set First (DJSF), dimana penjadwalan ini dengan memaksimalkan jumlah pekerjaan yang diselesaikan per unit bertujuan untuk mengurangi waktu penyelesaian dan meningkatkan beban data ke sistem.

Pada bagian ini, kami akan membahas berbagai penelitian dan teknik yang relevan dengan pengelolaan data dan optimasi database, termasuk teknik untuk mempercepat proses load data. Arsitektur database tradisional cenderung berfokus pada model relasional yang mengandalkan tabel besar untuk menyimpan data. Proses pemuatan data dalam arsitektur ini seringkali mengandalkan teknik ETL (*Extract*, *Transform*, *Load*),[10], [11], [12] yang meskipun efisien pada tingkat kecil, tidak mampu mengatasi volume data yang sangat besar, Penelitian sebelumnya menunjukkan bahwa teknik seperti pembagian data (*partitioning*), penggunaan indeks, dan optimasi *query* dapat membantu mengurangi waktu yang dibutuhkan untuk memuat data. Selain itu, arsitektur database yang lebih modular dan penggunaan sistem penyimpanan yang lebih efisien[13] .

Berbeda dengan beberapa penelitian yang disebutkan di atas, penelitian ini akan merekontuksi arsitekture database untuk meningkatkan proses load data yang di mana data base yang telah berjalan namun proses load datanya menjadi melambat dengan di terapkannya rekontruksi aristektur database . Permasalahan yang menjadi pokok bahasan dalam penelitian ini adalah membahas bagaimana proses ETL menuju data warehouse agar ketersediaan data pada data warehouse selalu ter-update serta mengurangi beban loading data atau query data dari data warehouse ke sistem informasi dengan cara melakukan optimasi pada data warehouse kemudian dilakukan analisa seberapa cepat loading data ke sistem informasi, setelah itu akan dilakukan perbandingan kecepatan pada sistem informasi dengan sebelumnya.

Penelitian ini bertujuan untuk merancang arsitektur aliran data. Dengan melakukan penjadwalan dari aliran data pusat ke gudang data, kemudian data dipartisi sesuai dengan kebutuhan menu sistem informasi. Kemudian membandingkan waktu eksekusi saat memuat data ke dalam sistem informasi dan menganalisis throughput sebelum dan sesudah optimasi.

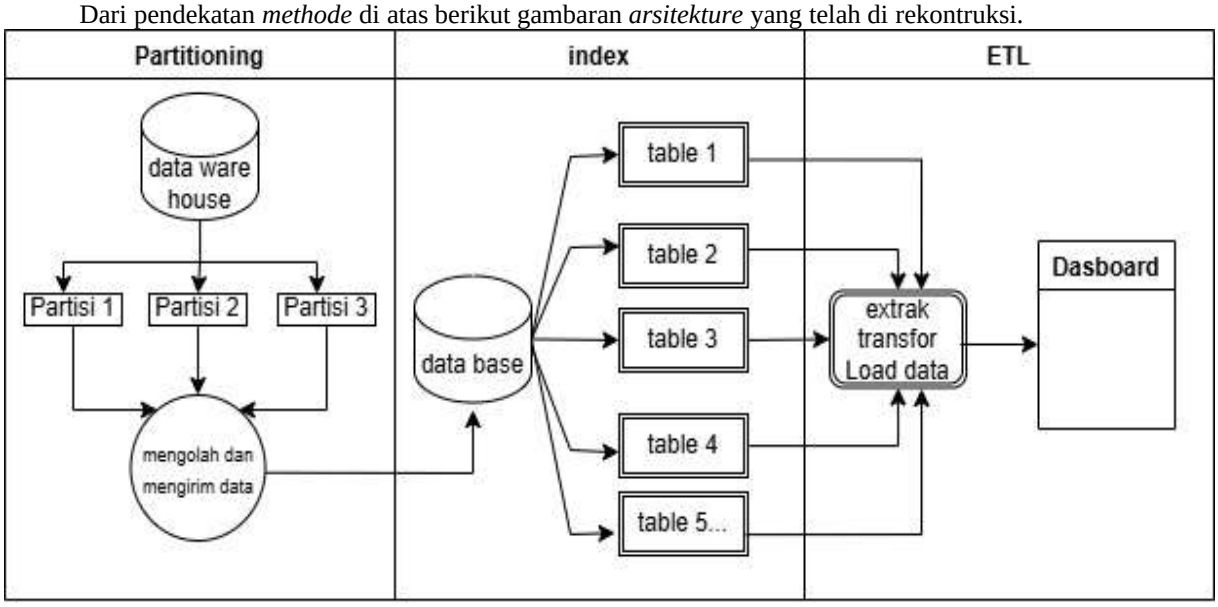
2. METODOLOGI PENELITIAN

Dalam penelitian ini, pendekatan yang digunakan melibatkan pengujian beberapa teknik rekonstruksi arsitektur database untuk meningkatkan proses load data. Kami akan melakukan analisis terhadap beberapa metode yang ada dan mengimplementasikan solusi yang dirancang untuk mengoptimalkan pemuatan data.

2.1. Pendekatan Rekonstruksi Arsitektur

a. Partitioning (Pembagian Data): Pembagian data ke dalam beberapa bagian (partition) untuk memungkinkan pemrosesan data secara paralel. Teknik ini dapat mengurangi waktu pemuatan data, terutama ketika data yang dimuat sangat besar.
b. Penggunaan Indeks: Dengan menggunakan indeks yang tepat pada kolom-kolom yang sering diquery, proses pencarian data dalam database dapat dipercepat, yang pada gilirannya mengurangi waktu load data.
c. ETL Pipeline yang Diperbarui: Merancang ulang proses ETL untuk memanfaatkan batch processing dan streaming untuk menangani data dalam volume besar secara lebih efisien.

Dari pendekatan *methode* di atas berikut gambaran *arsitekture* yang telah di rekontruksi.



Gambar 1. Rekontruksi database

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024 Submitted: **15/01/2024**; Accepted **25/01/2024;** Published: **18/02/2025**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 6 No. 2, Edisi Januari-April 2025, Page 1455-1460
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

2.2. Pengumpulan Data

a. Pengumpulan data dalam penelitian ini merupakan langkah penting dalam manajemen database, terutama ketika berfokus pada peningkatan load data. Load data mengacu pada proses memasukkan data ke dalam sistem database dengan cara yang efisien dan terstruktur. Untuk mencapai kinerja optimal, berikut adalah beberapa aspek penting yang harus diperhatikan:
b. Identifikasi Sumber Data
   Sebelum melakukan pengumpulan data, identifikasi semua sumber data yang relevan, seperti file CSV, API, atau data streaming. Pastikan sumber-sumber ini dapat menyediakan data dalam format yang sesuai dengan struktur database Anda.
c. Validasi dan Pembersihan Data
   Data yang dikumpulkan sering kali mengandung duplikasi, data kosong, atau kesalahan format. Proses validasi dan pembersihan data penting untuk memastikan data yang di-load ke dalam database berkualitas tinggi. Gunakan skrip atau alat otomatis untuk menyaring dan memperbaiki data.
d. Optimasi Format Data
   Gunakan format data yang efisien seperti JSON, Parquet, atau Avro untuk mempercepat proses transfer dan meminimalkan konsumsi sumber daya. Pemilihan format ini dapat membantu mempercepat load data ke dalam database.
e. Penerapan Batch Loading
   Untuk meningkatkan efisiensi, lakukan pengumpulan dan pemuatan data secara batch dibandingkan dengan mode satu per satu (single insert). Proses batch loading dapat mengurangi beban pada sistem dan mempercepat waktu pemuatan.
f. Penggunaan Indeks dan Partisi
   Pastikan database telah diatur dengan indeks yang relevan dan partisi yang sesuai untuk mempermudah pengambilan data dan mengurangi waktu pemrosesan saat data dimuat.
g. Monitoring dan Evaluasi
   Gunakan alat monitoring untuk memantau kinerja sistem saat melakukan load data. Identifikasi bottleneck yang mungkin terjadi dan sesuaikan strategi berdasarkan hasil evaluasi.
h. Dengan memperhatikan langkah-langkah di atas, pengumpulan dan load data ke dalam database dapat dilakukan secara efisien, mendukung performa tinggi, dan memastikan integritas data yang dimasukkan ke dalam sistem.

2.3. Analisis Performa

Analisis performa adalah proses yang dilakukan untuk mengukur, memantau, dan mengevaluasi efisiensi sistem dalam menangani proses load data ke database. Peningkatan performa load data sangat penting untuk memastikan database dapat mengelola volume data yang besar dengan cepat dan andal. Berikut adalah beberapa langkah utama dalam analisis performa untuk load dataClick or tap here to enter text.:

a. Pengukuran Waktu Load Data untuk mengukur waktu yang dibutuhkan untuk memuat sejumlah data ke dalam database. Gunakan metrik seperti waktu pemrosesan per batch atau waktu rata-rata untuk setiap transaksi data. Alat monitoring seperti SQL Profiler, pgAdmin, atau tools pihak ketiga dapat membantu mengukur performa secara akurat.
b. Penggunaan Sumber Daya Sistem Analisis performa juga melibatkan pemantauan penggunaan sumber daya seperti CPU, memori, dan disk I/O selama proses load data. Kinerja yang optimal dicapai ketika sumber daya digunakan secara efisien tanpa *bottleneck*. *Tools* seperti *Prometheus*, *Grafana*, atau built-in database monitoring tools dapat membantu.
c. Identifikasi Bottleneck *Bottleneck* terjadi ketika ada bagian sistem yang memperlambat seluruh proses load data, seperti kecepatan disk yang lambat, *query* yang tidak optimal, atau penguncian tabel. Dengan menganalisis *query execution plan* atau menggunakan *profiler*, *bottleneck* dapat diidentifikasi dan diperbaiki.
d. Evaluasi *Query* dan *Indeks Query* yang digunakan untuk load data perlu dievaluasi. Optimasi seperti menambahkan indeks, menggunakan partisi tabel, atau menghindari query yang terlalu kompleks dapat mempercepat proses. Periksa apakah struktur tabel sudah mendukung skenario load data yang efisien.
e. Uji Skala (*Scalability Testing*) Lakukan uji skala dengan berbagai ukuran data untuk melihat sejauh mana database dapat menangani peningkatan volume data. Simulasi ini membantu memahami batas performa sistem dan kebutuhan peningkatan kapasitas.
f. Perbandingan Sebelum dan Sesudah Optimasi Untuk mengetahui peningkatan performa, bandingkan hasil pengukuran sebelum dan sesudah implementasi optimasi. Fokus pada parameter seperti waktu pemrosesan, *throughput* data, dan latensi sistem.

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024 Submitted: **15/01/2024**; Accepted **25/01/2024;** Published: **18/02/2025**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 6 No. 2, Edisi Januari-April 2025, Page 1455-1460
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

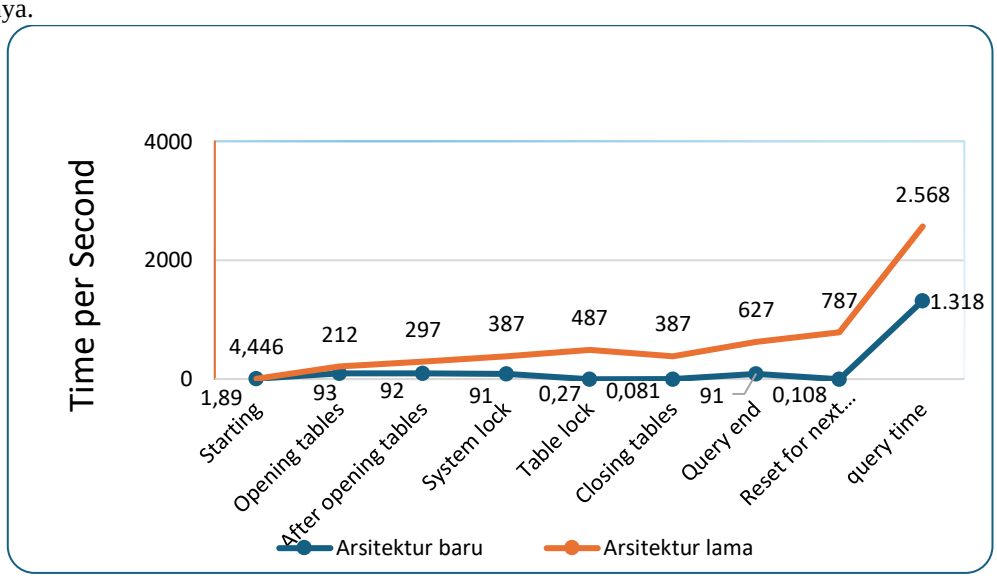
g. Penerapan *Cache* dan *Parallel Processing* Implementasi *caching* atau *parallel processing* dapat mempercepat load data. Cache mengurangi kebutuhan akses disk langsung, sementara parallel processing memungkinkan pemrosesan data secara simultan.
h. Pelaporan dan Rekomendasi Setelah analisis selesai, buat laporan yang merangkum hasil dan memberikan rekomendasi perbaikan. Laporan ini dapat mencakup hasil metrik, langkah optimasi yang diambil, dan dampak peningkatan pada kinerja sistem. Dengan analisis performa yang sistematis, organisasi dapat memastikan proses load data berjalan dengan efisien, mendukung pengambilan keputusan berbasis data, dan menjaga kestabilan sistem database di tengah peningkatan volume data.

3. HASIL DAN PEMBAHASAN

Rekonstruksi arsitektur database yang dilakukan berfokus pada peningkatan efisiensi waktu dan sumber daya dalam proses load data. Dengan penerapan teknologi seperti Partitoning, indexsing, serta extraksi transfor load , beberapa hasil nyata yang dapat diamati adalah:

3.1. Waktu Load Data Lebih Cepat

Implementasi Partitioning memungkinkan data diproses secara simultan dalam berbagai pipeline. Sebagai contoh, data yang sebelumnya membutuhkan waktu 2.568 detik untuk di-load, kini dapat diselesaikan hanya dalam waktu 1.318 detik.Incremental load berperan penting dalam memastikan hanya data baru atau data yang telah diubah yang diproses ulang, menghemat waktu hingga 51% dibandingkan proses ETL tradisional berikut lampiran hasil pengujianya.



Gambar 2. Hasil testing Query Load data

3.2. Downtime Sistem Berkurang

Sebelum rekonstruksi, proses ETL sering menyebabkan gangguan pada pengguna akhir karena resource yang besar digunakan untuk load data. Dengan arsitektur terdistribusi dan scheduling yang lebih baik, downtime sistem berkurang drastis.Penggunaan Sumber Daya yang Efisien.Pemanfaatan cloud database mengurangi kebutuhan hardware on-premise yang mahal dan sulit untuk dikelola. Misalnya, layanan seperti AWS Redshift menawarkan kemampuan autoscaling sehingga sistem tetap optimal meskipun terjadi lonjakan volume data.

3.3. Skalabilitas Sistem Kemampuan Menangani Pertumbuhan Data

Dengan desain arsitektur yang modular, penambahan kapasitas sistem menjadi lebih sederhana. Misalnya, sistem dapat dengan mudah menambahkan node baru untuk mengakomodasi pertumbuhan data tanpa perlu menghentikan operasi.Peningkatan Volume Data yang Didukung.Arsitektur yang terdistribusi memungkinkan sistem menangani data dalam skala petabyte dengan performa yang stabil. Data besar dari berbagai sumber seperti IoT, media sosial, atau log server dapat diproses secara bersamaan tanpa hambatan.

3.4. Pengurangan Error dalam DataAutomated Data Cleansing

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024 Submitted: **15/01/2024**; Accepted **25/01/2024;** Published: **18/02/2025**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 6 No. 2, Edisi Januari-April 2025, Page 1455-1460
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Implementasi algoritma cleansing otomatis membantu mendeteksi dan memperbaiki data yang rusak atau tidak konsisten sebelum data dimuat ke database utama. Hal ini mengurangi kesalahan manual hingga 80% dan meningkatkan kualitas data secara keseluruhan.Validasi Real-TimeProses validasi real-time diterapkan untuk mendeteksi error pada tahap awal. Contohnya, data yang tidak memenuhi standar tertentu langsung ditandai dan dikembalikan ke pipeline untuk diperbaiki tanpa mengganggu proses lainnya.

Rekonstruksi memungkinkan sistem terhubung dengan alat analitik canggih seperti Apache Spark dan Hadoop. Hal ini membuka peluang untuk analitik prediktif dan machine learning. NoSQL database seperti MongoDB atau Cassandra diintegrasikan untuk mendukung data tidak terstruktur seperti dokumen JSON atau data media. Hasilnya, proses analisis data menjadi lebih fleksibel dan cepat.Pemanfaatan Data StreamingDengan arsitektur baru, sistem mampu menangani data streaming secara real-time, seperti data dari sensor IoT, sehingga mendukung kebutuhan bisnis yang memerlukan respon cepat.

Transformasi Teknologi Database Rekonstruksi ini menekankan pada kombinasi berbagai teknologi database modern. Database relasional tetap relevan untuk kebutuhan transaksi, sedangkan NoSQL digunakan untuk data besar dan tidak terstruktur. Implementasi hybrid ini memberikan fleksibilitas tanpa mengorbankan performa. Relational database seperti MySQL atau PostgreSQL menawarkan integritas data dan dukungan terhadap query kompleks. Mereka ideal untuk data terstruktur seperti transaksi keuangan atau data pelanggan.NoSQL database mendukung skalabilitas horizontal, memungkinkan penanganan data yang bervariasi dan tidak terstruktur. MongoDB, misalnya, unggul dalam menyimpan data JSON yang terus bertambah dari aplikasi modern. Peran Cloud ComputingCloud computing memberikan pondasi untuk rekonstruksi ini. Berikut beberapa manfaat yang diperoleh antara lain

a. *Autoscaling*,
b. Sistem secara otomatis menyesuaikan resource berdasarkan kebutuhan,
c. Biaya Operasional yang Rendah:
d. *Model pay-as-you-go* menghindari pembelian hardware mahal.
e. *High Availability*

4. KESIMPULAN

Proses load data merupakan salah satu elemen krusial dalam pengelolaan data besar, terutama dalam konteks sistem informasi modern yang menangani volume data yang terus bertambah. Penelitian ini menyoroti berbagai tantangan yang dihadapi, termasuk waktu pemrosesan yang lama, penggunaan sumber daya yang tinggi, dan ketidakmampuan sistem tradisional dalam menangani data dalam skala besar secara efisien.

Rekonstruksi arsitektur database menjadi solusi yang efektif untuk mengatasi tantangan tersebut, dengan mengimplementasikan teknik-teknik seperti partitioning, penggunaan indeks, dan pipeline ETL yang lebih efisien. dan data streaming turut memberikan dampak signifikan terhadap efisiensi proses load data. Beberapa hasil utama dari rekonstruksi arsitektur ini meliputi:

a. Waktu load data yang lebih cepat berkat parallel processing dan incremental load.
b. Penurunan downtime sistem, yang meningkatkan pengalaman pengguna akhir.
c. Kemampuan skalabilitas sistem untuk menangani pertumbuhan data besar dengan penambahan kapasitas secara modular.
d. Pengurangan error dalam data melalui validasi real-time dan automated data cleansing, yang meningkatkan kualitas data.
e. Kemampuan analisis data yang lebih fleksibel dengan integrasi database modern dan dukungan terhadap data tidak terstruktur.

Rekonstruksi arsitektur yang menggabungkan relational database dengan NoSQL, serta memanfaatkan teknologi cloud, menghasilkan sistem yang lebih efisien, fleksibel, dan mampu menangani kebutuhan bisnis modern dengan responsivitas yang tinggi.

Masukan

a. Eksplorasi Teknologi Tambahan
   Teknologi seperti Apache Kafka untuk data streaming real-time, atau Delta Lake untuk pengelolaan data pada skala besar, dapat dieksplorasi lebih lanjut untuk mendukung kebutuhan arsitektur yang lebih kompleks.
b. Evaluasi Cost-Benefit
   Mengingat teknologi cloud menawarkan fleksibilitas tinggi tetapi dengan model pay-as-you-go, perlu dilakukan analisis cost-benefit untuk memastikan efisiensi biaya dalam jangka panjang.
c. Penerapan AI untuk Optimasi Load Data
   Algoritma berbasis kecerdasan buatan (AI) dapat digunakan untuk memprediksi bottleneck, mengoptimalkan scheduling ETL, atau memilih indeks yang optimal berdasarkan pola data dan query.
d. Pengukuran Efisiensi yang Terstandarisasi

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024 Submitted: **15/01/2024**; Accepted **25/01/2024;** Published: **18/02/2025**

JURNAL MEDIA INFORMATIKA [JUMIN]
Volume 6 No. 2, Edisi Januari-April 2025, Page 1455-1460
ISSN 2808-005X (media online)
Available Online at http://ejournal.sisfokomtek.org/index.php/jumin

Menyediakan metrik kinerja terstandarisasi seperti waktu rata-rata load data, efisiensi resource, dan error rate akan membantu dalam membandingkan efektivitas teknik yang diimplementasikan dengan metode tradisional.

Dengan fokus pada penerapan teknologi modern dan pengelolaan yang terencana, optimasi proses load data dapat memberikan kontribusi besar dalam mendukung pengambilan keputusan berbasis data yang lebih cepat, akurat, dan efisien.

UCAPAN TERIMAKASIH

Terima kasih disampaikan kepada teman-teman di priodi sistem informasi telah mendukung terlaksananya penelitian ini.

REFERENCES

[1] D. P. Kristiadi, H. L. H. S. Warnars, R. Randriatoamanana, F. Megantara, L. Nulhakim, and M. Zarlis, "Big Data implementation for Inventory warehouse systems," *1st 2018 Indonesian Association for Pattern Recognition International Conference, INAPR 2018 - Proceedings*, pp. 207–212, 2019, doi: 10.1109/INAPR.2018.8627030.
[2] "Database Systems SIXTH EDITION."
[3] R. K. Meleppat, M. V. Matham, and L. K. Seah, "An efficient phase analysis-based wavenumber linearization scheme for swept source optical coherence tomography systems," *Laser Phys Lett*, vol. 12, no. 5, p. 55601, 2015, doi: 10.1088/1612-2011/12/5/055601.
[4] S. B, A. A. Ilham, and A. W. Paundu, "Optimization of Data Warehouse Architecture to Improve Information System Performance," in *2023 International Conference on Computer Science, Information Technology and Engineering (ICCoSITE)*, 2023, pp. 240–245. doi: 10.1109/ICCoSITE57641.2023.10127721.
[5] F. C. Daeng Bani, Suharjito, Diana, and A. S. Girsang, "Implementation of Database Massively Parallel Processing System to Build Scalability on Process Data Warehouse," *Procedia Comput Sci*, vol. 135, pp. 68–79, 2018, doi: 10.1016/j.procs.2018.08.151.
[6] C. A. U. Hassan *et al.*, "Optimizing the Performance of Data Warehouse by Query Cache Mechanism," *IEEE Access*, vol. 10, pp. 13472–13480, 2022, doi: 10.1109/ACCESS.2022.3148131.
[7] E. Costa, C. Costa, and M. Y. Santos, "Evaluating partitioning and bucketing strategies for Hive-based Big Data Warehousing systems," *J Big Data*, vol. 6, no. 1, pp. 1–38, 2019, doi: 10.1186/s40537-019-0196-1.
[8] R. Sreekumar and S. B. B, "ETL Scheduling in Real-Time Data Warehousing," vol. 5, no. 04, pp. 416–418, 2014.
[9] Z. Hu and D. Li, "Improved heuristic job scheduling method to enhance throughput for big data analytics," *Tsinghua Sci Technol*, vol. 27, no. 2, pp. 344–357, 2022, doi: 10.26599/TST.2020.9010047.
[10] A. Y. Berliantara, S. A. Wicaksono, and A. Pinandito, "Optimasi Scheduling untuk Proses Extract , Transform , Load (ETL) pada Data Warehouse Menggunakan Metode Round Robin Data Partitioning (Studi Kasus : Universitas XYZ)," *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer (J-PTIIK) Universitas Brawijaya*, vol. 1, no. 11, pp. 1358–1366, 2017.
[11] A. R. C. Aloysius Adhyatma Herfangsyah1, Willy Sudiarto Raharjo2, "Analisis Faktor Optimasi untuk Data Warehouse dengan Data Tabungan pada Bank XYZ," *Jurnal Terapan Teknologi Informasi*, vol. 4, no. 1, pp. 33–43, 2021, doi: 10.21460/jutei.2020.41.192.
[12] J. Song, Y. Bao, and J. Shi, "A Triggering and scheduling approach for ETL in a real-time data warehouse," *Proceedings - 10th IEEE International Conference on Computer and Information Technology, CIT-2010, 7th IEEE International Conference on Embedded Software and Systems, ICESS-2010, ScalCom-2010*, no. Cit, pp. 91–98, 2010, doi: 10.1109/CIT.2010.57.
[13] Y. Zhu, E. Haihong, and M. Song, "A Scheduling System for Big Data Hybrid Computing Workflow," *Proceedings of the IEEE International Conference on Software Engineering and Service Sciences, ICSESS*, vol. 2020-Octob, pp. 102–106, 2020, doi: 10.1109/ICSESS49938.2020.9237729.
[14] I. Sokolov and I. Turkin, "Resource efficient data warehouse optimization," *Proceedings of 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies, DESSERT 2018*, pp. 491–495, 2018, doi: 10.1109/DESSERT.2018.8409183.
[15] W. H.Inmon, *Building the Data Warehouse*, vol. 62, no. 6. Wiley Publishing, Inc, 2005.

This is an open access article under the CC–BY-SA license

Terakreditasi SINTA 5 SK :72/E/KPT/2024 Submitted: **15/01/2024**; Accepted **25/01/2024;** Published: **18/02/2025**