Model Sinkronisasi Database Satu Arah dengan Metode Audit Log Menggunakan Apache Kafka

I Wayan Gus Arisna [a1], I Made Sukarsa [a2], Putu Wira Buana [a3]
[a]Program Studi Teknologi Informasi, Fakultas Teknik, Universitas Udayana, Bali, Indonesia
e-mail: [1]agus.aris83@gmail.com, [2]sukarsa@unud.ac.id, [3]wbhuana@unud.ac.id

Abstrak

Transaksi-transaksi bisnis semakin banyak dilakukan secara online, namun seiring perusahaan semakin maju, muncul masalah-masalah seperti ketidakmampuan dalam mengelola konsistensi data, khususnya ketika data tersebut berhubungan dengan lebih dari satu database. Sinkronisasi data adalah salah satu cara yang sering digunakan untuk penyelarasan data, dan replikasi merupakan metode yang paling banyak digunakan. Namun, agar data sepenuhnya konsisten, tidak cukup hanya dengan memanfaatkan proses replikasi. Permasalahan yang sering ditemui pada proses konfigurasi yang rumit. Pendekatan Audit Log merupakan salah satu objek alternatif yang dapat digunakan dalam mengembangkan sinkronisasi satu arah pada database. Audit Log adalah sebuah metode pencantatan perubahan yang ada pada database dengan menggunakan trigger. Data perubahan database tersebut yang nantinya menjadi acuan dalam sinkronisasi ke database lainnya. Pengiriman perubahan data pada database ke database yang lain memanfaatkan Apache Kafka. Apache merupakan open-source message broker yang memiliki perfoma tinggi sehingga dapat mengirim data secara realtime.

Kata kunci: *Audit Log, Database, Trigger, Sinkronisasi, Message Broker, Kafka*

Abstract

Business transactions are increasingly conducted online. However, as companies advance, problems arise, such as the inability to manage data consistency, particularly when the data is related to more than one database. Data synchronization is a commonly used method for data alignment, and replication is the most widely used method. However, to ensure data is fully consistent, relying solely on replication processes is not enough. Configuration complexities are often encountered. The Audit Log approach is an alternative method that can be used to develop one-way synchronization in databases. Audit Log is a method of recording changes in a database using triggers. The changed data in the database serves as a reference for synchronization with other databases. The transmission of data changes from one database to another utilizes Apache Kafka. Apache Kafka is an open-source message broker that delivers high-performance, enabling real-time data transmission.

Keywords : *Audit Log, Database, Trigger, Synchronization, Message Broker, Kafka*

1. Pendahuluan

Perkembangan teknologi informasi telah menciptakan peluang bisnis baru, diantaranya transaksi-transaksi bisnis yang dilakukan secara online. Masalah baru muncul sejalan dengan kemajuan teknologi seperti ketidakmampuan dalam mengelola integrasi data, khususnya ketika data tersebut berhubungan dengan sistem database terdistribusi.

Integrasi data merupakan bagian terpenting dalam penerapan database terdistribusi, dimana data dari berbagai sumber bisa disatukan dengan menerapkan integrasi. Sistem database terdistribusi memiliki beberapa kelebihan, misalnya kemampuan untuk menangani ekspansi (enhancement atau pelebaran) data, ketersediaan data dan kemampuan untuk mengelola pendistribusian data [2]. Replikasi dalam database terdistribusi merupakan salah satu cara yang berguna untuk mendistribusikan data tersebut.

Mewujudkan data yang sepenuhnya konsisten, tidak cukup hanya dengan memanfaatkan proses replikasi. Permasalahan yang sering ditemui antara lain adalah kegagalan koneksi atau client yang sedang offline. Prosedur atau model sinkronisasi data dapat digunakan

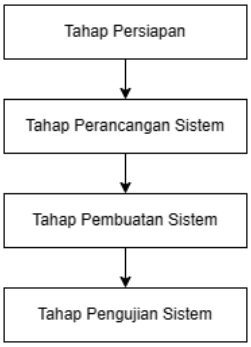
untuk mengatasi masalah tersebut. Sinkronisasi data adalah bagian dari replikasi, ini adalah proses untuk memastikan setiap salinan database berisi objek dan data yang serupa. Proses sinkronisasi memungkinkan data pada database tertentu diperbarui secara real time atau periodik pada database lainnya.

Gudakesa melakukan penelitian untuk menyelesaikan permasalah tersebut menggunakan metode Audit Log untuk mengetahui perubahan database melalui trigger dan mengirim perubahan database menggunakan socket [1]. Audit Log merupakan mekanisme pencatatan aktivitas pengguna database seperti perintah SQL yang dijalankan, waktu perintah dijalankan dan parameter yang digunakan. Audit Log dapat dikonfigurasi oleh pengguna untuk mencatat perubahan data yang spesifik sehingga dapat memberikan performa yang bagus. Penelitian tersebut telah berhasil menangani pencatatan perubahan database, namun penelitian ini memiliki kekurangan yanng mana menggunakan socket untuk pengiriman data yang dapat menimbulkan beberapa permasalah yaitu kegagalan sinkronisasi saat host dalam keadaan offline dan keamaan data dalam pesan.

Penelitian ini dibuat untuk memperbaiki permasalahan sinkronasi database yang telah dilakukan pada penetian sebelumnya dengan menggunakan pendekatan yang berbeda untuk sinkronisasi database satu arah. Penelitian ini menggunakan metode Audit Log untuk mendapatkan perubahaan pada database dan message broker sebagai pengganti socket dalam pertukaran data. Apache Kafka adalah message broker open source yang memiliki perfoma tinggi. Apache Kafka sesuai digunakan dalam penelitian ini sebagai media pertukaran data karena memiliki reabilitas yang tinggi dimana data akan pasti terjaga saat salah satu host dalam keadaan offline dan keamanan pesan dijaga dengan TLS/SSL encryption.

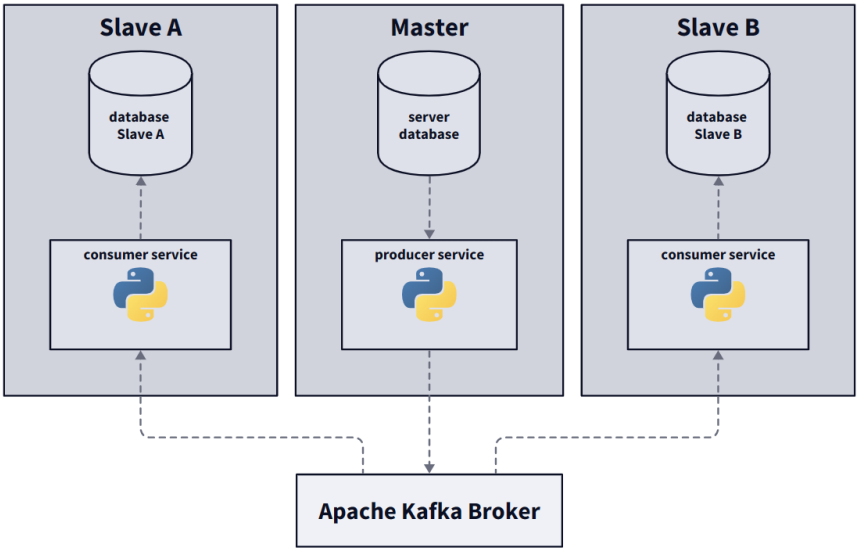
2. Metode Penelitian

Metode penelitian dari model sinkronisasi yang dikembangkan terdiri dari 4 langkah, yang dapat dilihat pada Gambar 1.



Gambar 1. Metode Penelitian

Gambar 1 menunjukan langkah-langkah yang dilakukan dalam pengembangan model sinkronisasi. Tahap Persiapan merupakan tahap melakukan pengumpulan informasi seperti jenis message broker dan bahasa pemrograman yang akan digunakan. Tahap perancangan sistem merupakan tahap melakukan perancangan bisnis proses sinkronasi, perancangan algoritma setiap service yang dibuat. Tahap pembuatan sistem merupakan melakukan pemrograman terhadap setiap service yang telah direncanakan. Tahap pengujian merupakan tahap pengujian terhadap service yang telah di-install pada dua buah host dengan jumlah data yang berbeda-beda. Gambaran umum dari sistem yang dikembangkan dapat dilihat pada Gambar 2.



Gambar 2. Gambaran Umum

Terdapat tiga komponen penting pada penelitian ini yaitu Master, Slave dan Broker. Master merupakan host yang terhubung dengan database utama yang dijadikan patokan untuk database lainnya. Slave merupakan host yang terhubung dengan database sekunder yang bersifat homogeneus dengan database utama. Broker merupakan host yang memegang Apache Kafka yang menghubungkan Master dan Slave agar dapat mengirim pesan satu sama lain. Pesan yang dikirim berisikan data perubahan database yang diambil dengan menggunakan metode Audit Log. Penelitian ini menggunakan publish-subscribe pattern yang ada pada Apache Kafka untuk mengirim pesan. Setiap host yang men-subscribe spesifik topic akan menerima pesan yang di-publish ke Apache Kafka. Master memerlukan Producer Service yang berfungsi untuk mengirim pesan yang berisi data perubahaan database ke Broker. Slave memerlukan Consumer Service yang berfungsi untuk mengambil pesan dari Broker dan melakukan perubahan data bedasarkan pesan tersebut. Producer Service dan Consumer dikembangkan dengan menggunakan bahasa pemrograman Python.

3. Kajian Pustaka

3.1. Sinkronisasi

Sinkronisasi data adalah suatu proses untuk menjaga konsistensi data pada suatu server dengan server lainnya. Terdapat proses penyalinan data yang disimpan ke dalam suatu tabel dan skema yang berada pada database yang lain. Proses sinkronisasi memungkinkan suatu data yang berada pada database tertentu dapat diperbarui secara langsung maupun berkala pada database yang lain. Teknik ini merupakan dasar dari konsep replikasi yang ada pada database. Mekanisme sinkronisasi ini diperlukan untuk mengubah data yang terdapat pada global schema secara langsung. Data yang diperbaharui pada global schema merupakan data yang berada pada local schema dari database yang terdistribusi. Antara local schema dengan global schema bisa jadi mempunyai skema yang secara struktur datanya dapat sama (homogeneous), ataupun berbeda (heterogeneous). Proses sinkronisasi pada database harus dapat menyesuaikan struktur data yang ada pada setiap database yang berbeda. Hal ini harus diterapkan untuk menjaga konsistensi data [2].

John Arissaputra melakukan penelitian serupa untuk menyelesaikan permasalah tersebut menggunakan metode Binary Log untuk mengetahui perubahan disetiap basis data lalu perubahan tersebut dikirim ke host lain yang mereplikasi basis data tersebut menggunakan socket. Penelitian tersebut berhasil menangani masalah koneksi jaringan dan host yang offline saat proses replikasi [4].

Hanafi melakukan nenelitian mengenai sinkronasi database menggunakan API Dropbox sebagai third party. Kebutuhan sistem yang harus multitasking dan memiliki reliabilitas yang tinggi mampu menjadikan database tidak hanya sebagai media penyimpanan data, namun juga

sebagai sarana pertukaran data. Mekanisme pertukaran data yang dilakukan yaitu melewati Dropbox sebagai perantara dan meneruskan ke tujuan pengiriman dengan memafaatkan identifier email dari pengguna Dropbox, sehingga pesan disingkronkan ke dalam database penerima [5].

3.2. MySQL Database

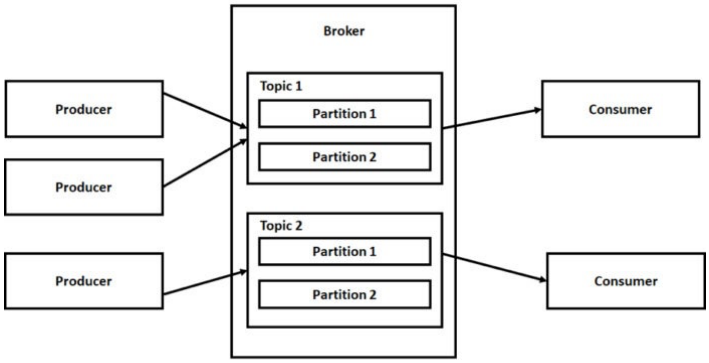
MySQL merupakan sebuah program database server yang mampu menerima dan mengirimkan data dengan sangat cepat, multi user serta menggunakan perintah dasar SQL (Structured Query Language). MySQL merupakan sebuah database server yang dapat digunakan secara bebas tanpa perlu membeli lisensi sehingga dapat digunakan untuk database ini untuk keperluan pribadi atau usaha. MySQL dikembangkan seorang programmer database bernama Michael Widenius. Selain database server, MySQL juga merupakan program yang dapat mengakses suatu database MySQL yang berposisi sebagai Server, yang berarti program kita berposisi sebagai Client. Jadi MySQL adalah sebuah database yang dapat digunakan sebagai Client maupun Server [6].

3.3. Audit Log

Konsep dari Audit Log adalah mendokumentasikan apabila terdapat perubahan pada sistem dan secara otomatis menciptakan log komputer. Bentuk dari log pada umumnya adalah berupa file, namun tabel database dapat digunakan. Audit Log pada database SQL adalah mekanisme pencatatan aktivitas pengguna seperti perintah SQL yang dijalankan. Penerapan Audit Log pada database MySQL dilakukan dengan memanfaatkan trigger dan tabel [1]. Trigger pada MySQL adalah perintah SQL yang dapat berjalan secara otomatis saat terjadi event insert, update dan delete. Penerapan Audit Log dapat dilakukan dengan cara memasang trigger ke setiap tabel pada database. Trigger akan mencatat setiap perubahan data selama proses insert, update dan delete berlangsung dan menyimpannya ke suatu tabel yang selanjutnya diproses untuk kebutuhan tertentu. Tabel dalam MySQL adalah struktur dasar yang digunakan untuk menyimpan dan mengorganisir data dalam basis data. Tabel terdiri dari kolom dan baris yang membentuk skema yang terstruktur. Setiap kolom dalam tabel memiliki nama dan tipe data yang menentukan jenis nilai yang dapat disimpan di dalamnya. Tabel pada penerapan Audit Log berfungsi untuk tempat penyimpanan data perubahaan database yang ditangkap oleh trigger.

3.4. Apache Kafka

Apache Kafka adalah sebuah platform open-source yang digunakan untuk mengelola aliran data secara real-time. Apache Kafka pada awalnya dikembangkan di LinkedIn dan sekarang menjadi bagian dari yayasan Apache Software Foundation. Apache Kafka dirancang untuk menangani volume data yang besar dan memungkinkan pertukaran data yang efisien antara producer dan consumer [7]. Arsitektur dari Apache Kafka dapat dilihat pada Gambar 3.



Gambar 3. Arsitektur Apache Kafka

Apache Kafka menggunakan konsep publish-subscribe yang mana produser mengirimkan data ke topic, dan consumer dapat men-subscribe ke topik tersebut untuk menerima data secara real-time. Data dalam Apache Kafka disimpan dalam urutan yang terorganisir dalam struktur log yang disebut commit log. Setiap pesan yang masuk diberi nomor urut dan disimpan dalam commit log, yang memungkinkan akses cepat dan efisien ke data. Kafka memiliki kehandalan yang tinggi dalam pengiriman pesan. Setiap pesan yang diterima oleh Kafka disimpan dalam log yang persisten dan tahan bencana. Pesan hanya dihapus setelah periode retensi tertentu atau dengan kebijakan penghapusan yang ditentukan. Ini memastikan bahwa data dapat diakses dalam jangka waktu yang lama dan terhindar dari kehilangan.

Guo Fu melakukan penelitian perbandingan diantara lima sistem message queue. Kelima sistem message queue tersebut diantara lain adalah RabbitMQ, Apache Kafka, RocketMQ, ActiveMQ and Pulsar. Penelitian tersebut membandingkan setiap sistem dalam beberapa aspek yaitu activity community, latecy, throughput, multifunction, transaction message, delivery guarantee, scalability, realibity dan order guarantee [8]. Hasil penelitian tersebut menunjukan Apache Kafka memiliki throughput tertinggi dan menjadi rekomendasi untuk sistem yang menangani pesan dalam jumlah besar seperti log processing, big data analysis dan stream processing.

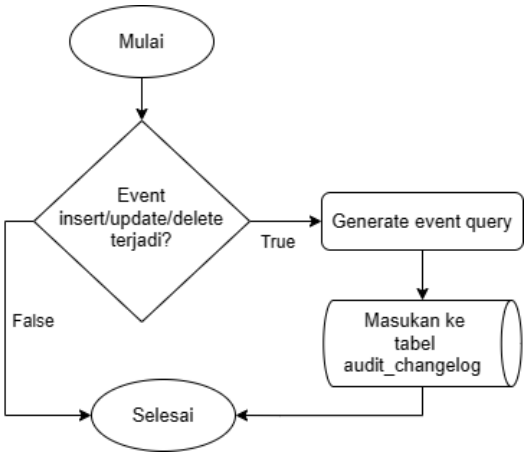
3.5. Python

Python adalah bahasa pemrograman tingkat tinggi yang bersifat interpretatif dan serba guna. Bahasa pemogragaman Python dikembangkan pada awal tahun 1990-an oleh Guido van Rossum, Python menekankan sintaks yang bersih, mudah dibaca, dan memungkinkan pengembangan perangkat lunak yang lebih produktif. Python memiliki beberapa keunggulan yaitu mudah dibaca, community support yang kuat, portabilitas, multi-paradigma dan developer experience yang bagus [9].

4. Hasil dan Pembahasan

4.1. Audit Log

Audit Log pada database SQL adalah mekanisme pencatatan aktivitas pengguna seperti perintah SQL yang dijalankan. Penerapan Audit Log pada database MySQL dilakukan dengan memanfaatkan trigger dan tabel. Setiap tabel pada database memiliki tiga trigger untuk menangkap event insert, update dan delete. Algoritma dari trigger-trigger yang digunakan untuk mencatat perubahan database dapat dilihat pada Gambar 4.



Gambar 4. Flowchart Trigger

Trigger akan berjalan saat terjadi event pada tabel database. Proses penentuan nilai kolom uuid terjadi hanya untuk event insert. Proses generate event query merupakan proses pembuatan sintaks query SQL bedasarkan event yang terjadi. Trigger akan menyimpan sintaks query SQL ke dalam tabel audit_changelog. Struktur dari tabel audit_changelog dapat dilihat pada Tabel 1.

Tabel 1. Struktur Tabel Audit Changelog

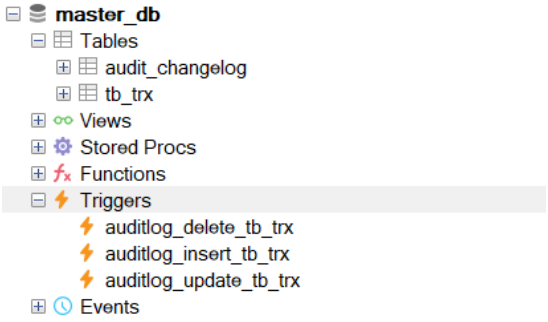
Nama Kolom	Tipe Data	Keterangan
changelog_id	bigint	Primary key dari tabel
query	text	Query hasil generasi dari trigger
table	varchar	Nama tabel dari event yang terjadi
pk	bigint	Primary key dari event yang terjadi
type	tinyint	Tipe dari event yang terjadi untuk insert (1), update (2) dan delete (3)
created_at	datetime	Timestamp dari event yang terjadi
published	tinyint	Status event dipublish ke Broker

Simulasi dilakukan untuk mengetahui mengetahui bagaimana Event Trigger dan Changelog Trigger berkerja. Contoh tabel transaksi untuk simulasi dapat dilihat pada Gambar 5.

id	nim	waktu	nominal_ukt	flag
1	16055501	2023-07-15 01:13	1000000	1
2	16055502	2023-07-16 01:13	2000000	1
3	16055503	2023-07-17 01:13	3000000	1
(Auto)	(NULL)	(NULL)	(NULL)	(NULL)

Gambar 5. Tabel Transaksi Master

Tabel transaksi mempunyai lima kolom dengan tipe data yang berbeda. Contoh trigger-trigger yang menangani event yang terjadi pada tabel transaksi dapat dilihat pada Gambar 6.



Gambar 6. Trigger Auditlog

Trigger-trigger pada Gambar 6 mempunyai tugas untuk menangani event insert, update dan delete pada tabel transaksi. Tabel tb_alphabet sudah mengalami event insert yang mana terdapat 3 baris data pada Gambar 5. Menerapkan event update dan delete pada tabel transaksi dilakukan memodifikasi baris data yang sudah ada. Kondisi tabel transaksi setelah modifikasi dapat dilihat pada Gambar 7.

id	nim	waktu	nominal_ukt	flag
1	16055501	2023-07-15 01:13	1000000	1
2	16055502	2023-07-16 01:13	5000000	0
(Auto)	(NULL)	(NULL)	(NULL)	(NULL)

Gambar 7. Tabel Transaksi Master Setelah Modifikasi

Menerapkan event update dilakukan dengan mengupdate kolom nominal_ukt dan kolom flag pada baris data dengan primary key 2. Menerapkan event delete dilakukan dengan menghapus baris data dengan primary key 3. Trigger melakukan pencatatan perubahan data dan menyimpan perubahan data ke dalam tabel audit_changelog. Data dari tabel audit_changelog dapat dilihat pada Gambar 8.

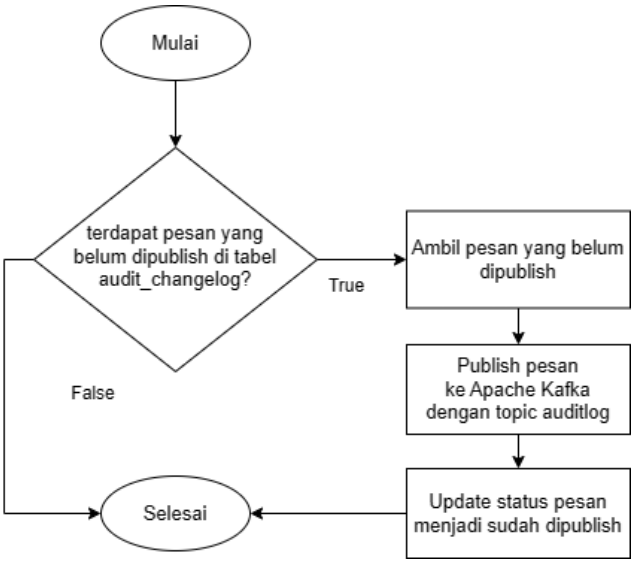
changelog_id	query		table	pk	type	published	created_at
1	INSERT INTO tb_trx (flag, nim, nominal_ukt, waktu) VALUE ...	109B	tb_trx	1	1	0	2023-07-14 17:13:21
2	INSERT INTO tb_trx (flag, nim, nominal_ukt, waktu) VALUE ...	109B	tb_trx	2	1	0	2023-07-14 17:13:45
3	INSERT INTO tb_trx (flag, nim, nominal_ukt, waktu) VALUE ...	109B	tb_trx	3	1	0	2023-07-14 17:14:05
4	UPDATE tb_trx SET flag = '0', nim = '16055502', nominal_ ...	117B	tb_trx	2	2	0	2023-07-14 17:23:42
5	DELETE FROM tb_trx WHERE id=3	29B	tb_trx	3	3	0	2023-07-14 17:23:45
(Auto)	(NULL)	0K	(NULL)	(NULL)	(NULL)	0	CURRENT_TIMESTAMP

Gambar 8. Tabel Audit Changelog

Hasil generasi query dari setiap event yang terjadi pada tabel transaksi dapat dilihat pada kolom query. Terdapat tiga query insert yang merupakan hasil dari event insert pada Gambar 5, satu query update dan satu query delete yang merupakan hasil dari event update dan delete pada Gambar 7. Simulasi ini berhasil menunjukan bahwa metode Audit Log berhasil dilakukan.

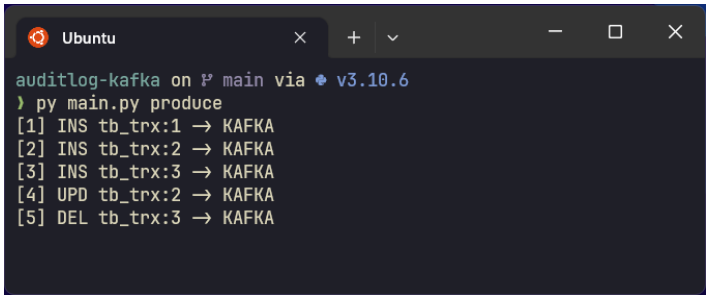
4.2. Producer Service

Producer Service merupakan service Master yang bertugas untuk mengirim pesan yang berisi perubahan database dari Master ke Broker. Flowchart dari Producer Service dapat dilihat pada Gambar 9.



Gambar 9. Flowchart Producer Service

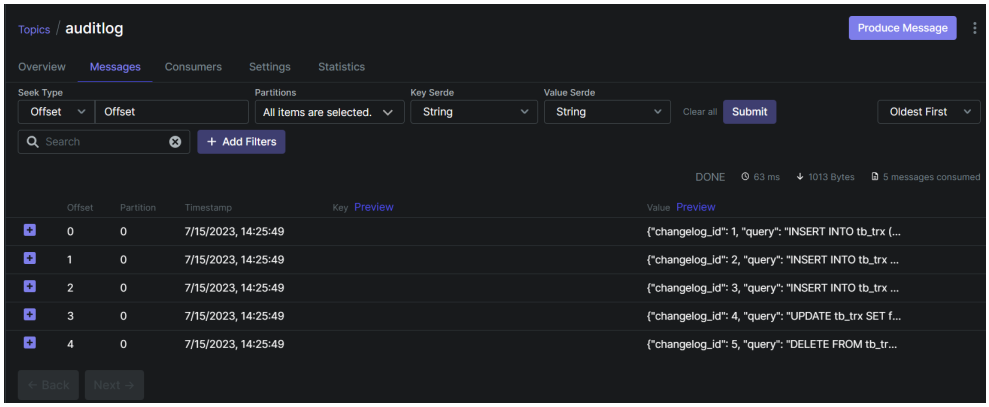
Producer Service terus mengecek pesan yang memiliki status belum dipublish pada tabel audit_changelog. Jika terdapat pesan yang belum dipublish pada tabel audit_changelog, Producer Service akan mengambil dan mengirim pesan tersebut ke Broker. Producer Service meng-update status pesan menjadi sudah dipublish setelah selesai mengirim pesan ke Broker. Simulasi dilakukan untuk mengetahui Producer Service berkerja. Simulasi dapat dilihat pada Gambar 10.



Gambar 10. Simulasi Producer Service

Producer Service mengambil semua pesan yang dipublish dari tabel audit_changelog dan mengirim semua pesan ke Broker. Producer Service meng-update status pesan-pesan tersebut menjadi sudah dipublish setelah selesai mengirim ke Broker untuk mencegah pesan

yang sama terkirim kembali. Pesan-pesan yang sudah dikirim dan berada dalam Apache Kafka Broker dapat pada Gambar 11.

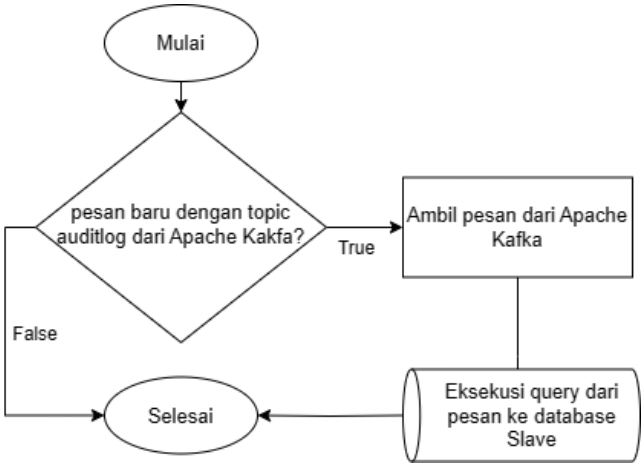


Gambar 11. Pesan-Pesan Pada Broker

Semua pesan yang dikirm oleh Producer Service berada dalam parition dengan topic auditlog pada Apache Kafka Broker. Jumlah pesan yang berada dalam Broker sesuai dengan jumlah pesan yang dikirim oleh Producer Serivce. Simulasi ini berhasil menunjukan bahwa Producer Service berhasil mengirim pesan dari Master ke Broker.

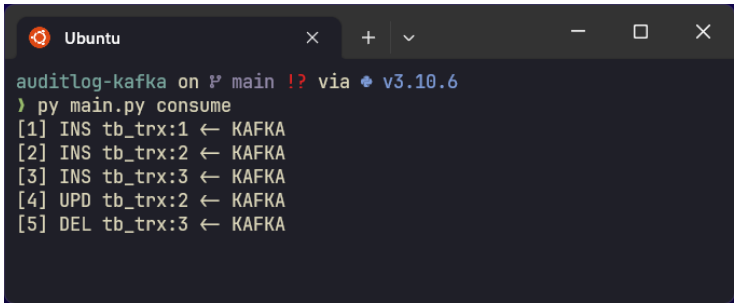
4.3. Consumer Service

Consumer Service merupakan service Slave yang bertugas untuk mengambil pesan dari Broker dan mengeksekusi query hasil generasi yang terdapat pada pesan ke database Slave. Flowchart dari Consumer Service dapat dilihat pada Gambar 12.



Gambar 12. Flowchart Consumer Service

Consumer Service men-subscribe topic auditlog yang berada pada Broker untuk menerima setiap pesan yang masuk dengan topic tersebut. Consumer Service akan mengeksekusi query dari setiap pesan yang diterima ke database Slave untuk melakukan sinkroniasi. Simulasi dilakukan untuk mengetahui Consumer Service berkerja. Simulasi dapat dilihat pada Gambar 13.



Gambar 13. Simulasi Consumer Service

Consumer Service mengambil semua pesan yang berada pada Broker dengan topic auditlog. Setiap pesan memiliki query yang dieksekusi oleh Consumer Service agar data pada tabel transaksi tersinkronisasi. Kondisi tabel transaksi pada Slave setelah proses sinkronisasi dapat dilihat pada Gambar 14.

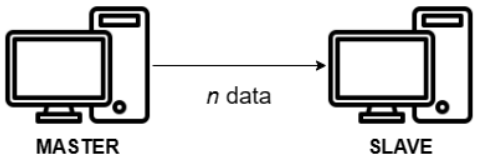
id	nim	waktu	nominal_ukt	flag
1	16055501	2023-07-15 01:13:07	1000000	1
2	16055502	2023-07-16 01:13:07	5000000	0
(Auto)	(NULL)	(NULL)	(NULL)	(NULL)

Gambar 14. Tabel Transaksi Slave

Gambar 14 menunjukan bahwa data tabel transaksi pada Slave sama dengan data tabel transaksi pada Master. Simulasi ini berhasil menunjukan bahwa Consumer Service berhasil mengambil pesan dari Broker dan mengeksekusi setiap query dari pesan untuk melakukan sinkronisasi data.

4.4. Pengujian Sistem

Pengujian dilakukan dengan menghitung waktu yang diperlukan untuk melakukan sinrkonisasi. Skenario pengujian dapat dilihat pada Gambar 15.



Gambar 15. Skenario Pengujian

Master mengirm data sejumlah n ke Slave. Sebagai contoh, jika n bernilai 100 maka Master mengirim 100 data ke Slave. Waktu sinkronisasi diambil dari rentang waktu diantara data pertama dan data terakhir. Spesifikasi dari setiap host dapat dilihat pada Table 2.

Tabel 2. Spesifikasi Master dan Slave

Komponen	Master	Slave
CPU	Intel Core i7 4.90 GHz (8 core)	Intel Core i7 4.90 GHz (4 core)
RAM	8 GB	4 GB
Storage	SSD 475 GB	SSD 32 GB

Pengujian dilakukan menggunakan satu tabel bernama tb_trx. Struktur dari tabel tb_trx dapat dilihat pada Gambar 16.

Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?
id	bigint			☑	☑	☐	☑
nim	varchar	20		☐	☐	☐	☐
waktu	datetime			☐	☐	☐	☐
nominal_ukt	double			☐	☐	☐	☐
flag	tinyint			☐	☐	☐	☐

Gambar 16. Tabel Pengujian

Database Master dan Slave mempunyai sebuah tabel bernama tb_trx untuk disinkronisasi. Semua data yang disinkronisasi merupakan data dari tabel tb_trx.

4.4. Hasil Pengujian Sistem

Terdapat 3 tahapan yang dilakukan dalam pengujian yaitu tahap pengujian insert, tahap pengujian update dan tahap pengujian delete. Hasil dari pengujian sistem dapat dilihat pada Table 3.

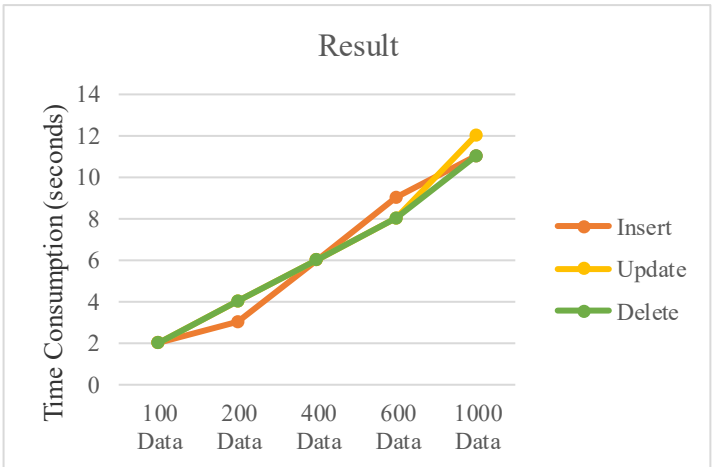
Tabel 3. Hasil Pengujian

DML	Time Consumption (seconds)				
	100	200	400	600	1000
Insert	2	3	6	9	11
Update	2	4	6	8	12
Delete	2	4	6	8	11

Pengujian insert dilakukan dengan mencatat waktu yang diperlukan untuk mereplikasikan data sejumlah n secara bersamaan. Hasil pengujian menunjukan sistem sinkronisasi memerlukan 2 detik untuk 100 data, 3 detik untuk 200 data, 6 detik untuk 400 data, 9 detik untuk 600 data dan 11 detik untuk 1000 data.

Pengujian update dilakukan dengan menginisiasi sejumlah n data di awal dan mencatat waktu diperlukan untuk meng-update setiap baris data sampai semua n data tersinkronisasi. Hasil pengujian menunjukan sistem sinkronisasi memerlukan 2 detik untuk 100 data, 4 detik untuk 200 data, 6 detik untuk 400 data, 8 detik untuk 600 data dan 12 detik untuk 1000 data.

Pengujian delete dilakukan dengan menginisiasi sejumlah n data di awal dan mencatat waktu diperlukan untuk menghapus setiap baris data sampai semua n data tersinkronisasi. Hasil pengujian menunjukan sistem sinkronisasi memerlukan 2 detik untuk 100 data, 4 detik untuk 200 data, 6 detik untuk 400 data, 8 detik untuk 600 data dan 11 detik untuk 1000 data. Hasil pengujian dalam bentuk grafik dapat dilihat pada Gambar 17.



Gambar 17. Grafik Hasil Pengujian

Gambar 17 menunjukan grafik dari hasil pengujian sistem sinkronisasi. Terlihat bahwa grafiknya relatif stabil meskipun data yang diproses semakin besar. Hal ini menunjukkan bahwa

sistem sinkronisasi yang dikembangkan cukup stabil untuk menangani pertumbuhan data yang terus menerus.

5. Kesimpulan

Metode Audit Log dapat digunakan dalam mengembangkan sistem sinkronisasi database dengan memanfaatkan Apache Kafka sebagai media pengiriman pesan. Sistem sinkronisasi yang dikembangkan memiliki perfoma yang stabil dalam menangani jumlah data yang terus bertambah. Bagian yang paling penting adalah bagaimana mencatat perubahan data yang terjadi pada database menggunakan trigger dan tabel. Penggunaan trigger dan tabel dengan algoritma tertentu dapat menghasilkan informasi untuk berbagai hal yang berguna.

Daftar Pustaka

[1] G.H. Surya, I.M. Sukarsa, I.G.M.A. Sasmita. Two-Ways Database Synchronization in Homogeneous DBMS Using Audit Log Approach. *Journal of Theoretical and Applied Information Technology*. 2014; 65(3): 854-859.

[2] G.H. Surya, I.M. Sukarsa, and I.G.M.A. Sasmita. Two Ways Database Synchronization In Homogenous Binary Log. *Journal of Theoretical and Applied Information Technology*. 2014; 65(1): 76-82.

[3] Malhotra, Naveen, Chaudhary A. Implementation of Database Synchronization Technique between Client and Server. *International Journal of Engineering Science and Innovative Technology*. 2014; 3(4): 460–465.

[4] I.G.J Arissaputra, I.M. Sukarsa, P.W. Buana, N.W. Wisswani. Binary Log Design for One-Way Data Replication with ZeroMQ. *International Journal of Modern Education and Computer Science.* 2018*;* 10(10): 22–30.

[5] H. Ahmad, I.M. Sukarsa, and A.A. Ketut Agung Cahyawan Wiranatha. Pertukaran Data Antar Database Dengan Menggunakan Teknologi API. *Lontar Komputer : Jurnal Ilmiah Teknologi Informasi*. 2017; 8(1): 22-30.

[6] S. Haris. Modul Pembelajaran Praktek Basis Data (MySQL). Semarang: Universitas Dian Nuswantoro, 2012.

[7] K. Jay. Kafka : a Distributed Messaging System for Log Processing. 2011.

[8] G. Fu, Y. Zhang, G. Yu. A Fair Comparison of Message Queuing Systems. IEEE Access. 2021; 9:421-432.

[9] K.R Srinath. Python – The Fastest Growing Programming Language. *International Research Journal of Engineering and Technology.* 2017; 4(2): 354-357.