P-ISSN : 2580-4316
E-ISSN : 2654-8054 https://doi.org/10.37817/ikraith-informatika.v10i2

Implementasi Arsitektur OLAP pada Database OLTP dengan Log-Based CDC: Studi Kasus SIMKOPDES

[1]Yonatan Yusak Lestari, [2]Dr. Oman Komarudin, S.Si., M.Kom., [3]Dr. Aries Suharso, S.Si., M.Kom.
[1,2,3]Program Studi Informatika, Universitas Singaperbangsa Karawang, Karawang
E-mail: [1]2210631170052@student.unsika.ac.id, [2]oman@unsika.ac.id, [3]aries.suharso@staff.unsika.ac.id

ABSTRAK

SIMKOPDES merupakan sistem informasi yang mendukung pengelolaan dan penyajian informasi Koperasi Desa/Kelurahan Merah Putih. Penggunaan basis data *Online Transaction Processing* (OLTP) secara langsung untuk pelaporan menimbulkan potensi persaingan sumber daya karena *query* analitik melibatkan banyak *join*, agregasi, dan pemrosesan data dalam jumlah besar. Penelitian ini bertujuan merancang dan mengimplementasikan arsitektur *Online Analytical Processing* (OLAP) pada basis data OLTP SIMKOPDES dengan mekanisme *log-based Change Data Capture* (CDC). Metodologi yang digunakan adalah *Database System Development Life Cycle* (DSDLC), sedangkan tahap perancangan menerapkan *Four-Step Dimensional Design* Kimball. Skema OLAP terdiri atas empat tabel dimensi dan tiga tabel fakta. Sinkronisasi perubahan memanfaatkan *write-ahead log* PostgreSQL, *replication slot*, plugin wal2json, dan *middleware* Python. Pengujian menggunakan EXPLAIN (ANALYZE, BUFFERS) pada lima skenario analitik menunjukkan rata-rata efisiensi sebesar 85,23% untuk *execution time*, 53,59% untuk penggunaan I/O *buffer*, dan 80,09% untuk *query cost* dibandingkan OLTP. Pada seluruh lima skenario pengujian, jumlah baris dan nilai agregasi OLAP konsisten dengan hasil OLTP, sedangkan seluruh tujuh skenario perubahan data berhasil ditangkap dan diterapkan pada tabel OLAP yang terdampak. Hasil tersebut menunjukkan bahwa arsitektur OLAP berbasis *log-based CDC* mampu meningkatkan efisiensi analitik sekaligus mempertahankan konsistensi data sumber.

Kata kunci: *OLAP, OLTP, log-based CDC, star schema, PostgreSQL, SIMKOPDES*

ABSTRACT

SIMKOPDES is an information system that supports the management and presentation of Koperasi Desa/Kelurahan Merah Putih information. Directly executing reporting workloads on the Online Transaction Processing (OLTP) database may create resource contention because analytical queries involve multiple joins, aggregations, and large-scale data processing. This study designs and implements an Online Analytical Processing (OLAP) architecture on the SIMKOPDES OLTP database using a log-based Change Data Capture (CDC) mechanism. The study adopts the Database System Development Life Cycle (DSDLC), while Kimball's Four-Step Dimensional Design is applied during database design. The OLAP schema consists of four dimension tables and three fact tables. Data changes are synchronized using PostgreSQL write-ahead logging, a replication slot, the wal2json plugin, and Python middleware. Performance evaluation using EXPLAIN (ANALYZE, BUFFERS) across five analytical scenarios yielded average efficiencies of 85.23% in execution time, 53.59% in I/O buffer usage, and 80.09% in query cost relative to OLTP. Across all five scenarios, the OLAP row counts and aggregate values matched the OLTP results. In addition, all seven tested data-change scenarios were successfully captured and applied to the affected OLAP tables. These results indicate that an OLAP architecture with log-based CDC can improve analytical query efficiency while maintaining consistency with the OLTP source data.

Keywords: *OLAP, OLTP, log-based CDC, star schema, PostgreSQL, SIMKOPDES*

1. PENDAHULUAN

Koperasi Desa/Kelurahan Merah Putih merupakan program yang membutuhkan dukungan informasi terintegrasi untuk kelembagaan, modal, transaksi, produk, Rapat Anggota Tahunan (RAT), dan pembangunan gerai. SIMKOPDES digunakan sebagai aplikasi pengelolaan dan penyajian informasi tersebut. Dalam arsitektur yang berjalan, basis data OLTP menjadi sumber utama data operasional sekaligus sumber bagi dashboard dan pelaporan. Kondisi ini membuat satu lingkungan basis data melayani dua pola beban kerja yang berbeda, yaitu transaksi harian dan analisis agregat.

Basis data OLTP dirancang untuk menjaga konsistensi transaksi melalui struktur relasional yang ternormalisasi. Struktur tersebut efektif untuk operasi pencatatan dan pembaruan data, tetapi *query* analitik sering kali harus menggabungkan banyak tabel, menghitung ulang agregasi, dan menelusuri data historis. Connolly dan Begg (2015) membedakan kebutuhan basis data operasional dari sistem analitik, sedangkan Kimball dan Ross (2013) menekankan bahwa lingkungan *data warehouse* perlu mudah dipahami dan memiliki performa *query* yang baik. Santosa dan Putra (2023) juga menunjukkan bahwa pemisahan basis data transaksional dan analitik dapat meningkatkan performa sistem dibandingkan penggunaan satu basis data untuk kedua kebutuhan.

Kebutuhan analitik SIMKOPDES mencakup jumlah koperasi, status akun dan legalitas,

simpanan, transaksi, RAT, status lahan, progres pembangunan gerai, serta analisis berdasarkan wilayah, waktu, koperasi, dan produk. Kebutuhan tersebut lebih sesuai dilayani melalui struktur dimensional karena tabel fakta menyimpan nilai pengukuran dan indikator kondisi, sedangkan tabel dimensi menyediakan konteks analisis. Garani et al. (2023) menjelaskan bahwa *data warehouse* dan OLAP mendukung analisis multidimensi terhadap data historis dari sistem operasional. Wijaya et al. (2024) juga menunjukkan bahwa *star schema* dapat menyederhanakan akses analitik dan meningkatkan waktu respons dibandingkan struktur relasional standar.

Pemisahan OLTP dan OLAP memunculkan kebutuhan sinkronisasi. Proses pemuatan ulang seluruh tabel secara berkala tetap membebani sumber karena data dipindai meskipun hanya sebagian kecil yang berubah. *Change Data Capture* (CDC) memproses perubahan INSERT, UPDATE, dan DELETE tanpa membaca ulang seluruh dataset (Gomstyn & Jonker, 2025). Pada pendekatan *log-based CDC*, perubahan dibaca dari log transaksi sehingga tidak memerlukan *trigger* tambahan atau pemindaian tabel sumber secara terus-menerus. Santosa et al. (2025) menunjukkan bahwa CDC dapat mendukung sinkronisasi data secara waktu nyata dengan memproses hanya data yang berubah.

Berdasarkan permasalahan tersebut, penelitian ini membangun arsitektur OLAP yang terpisah dari OLTP SIMKOPDES dan menghubungkannya melalui *log-based CDC*. Kontribusi penelitian meliputi: (1) rancangan *star schema* untuk indikator analitik SIMKOPDES; (2) implementasi sinkronisasi berbasis WAL PostgreSQL, wal2json, dan *middleware* Python; serta (3) evaluasi performa, integritas data, dan fungsionalitas sinkronisasi.

2. LANDASAN TEORI

2.1 OLTP, OLAP, dan Pemodelan Dimensional

OLTP merupakan lingkungan pemrosesan transaksi yang mengutamakan konsistensi, kecepatan perubahan data, dan pengendalian integritas. Sebaliknya, OLAP dirancang untuk pelaporan, agregasi, dan eksplorasi data dari berbagai sudut pandang. Pemisahan keduanya bertujuan mengisolasi beban kerja agar aktivitas analitik tidak bersaing dengan transaksi operasional.

Pemodelan dimensional menyusun data ke dalam tabel fakta dan tabel dimensi. Kimball dan Ross (2013) merumuskan empat langkah utama, yaitu memilih proses bisnis, menetapkan *grain*, mengidentifikasi dimensi, dan mengidentifikasi fakta. *Grain* menjelaskan arti satu baris pada tabel fakta dan menjadi dasar untuk menjaga konsistensi perhitungan. *Star schema* menghubungkan tabel fakta secara langsung ke dimensi sehingga relasi analitik lebih sederhana dibandingkan skema OLTP ternormalisasi.

2.2 Log-Based Change Data Capture

CDC menangkap perubahan data pada sistem sumber dan meneruskannya ke sistem target. PostgreSQL mencatat perubahan transaksi pada *write-ahead log* (WAL). Melalui *logical decoding*, isi WAL dapat diterjemahkan menjadi perubahan logis yang dapat dibaca aplikasi. *Replication slot* mempertahankan posisi pembacaan agar perubahan tidak terlewat, sedangkan wal2json mengubah perubahan tersebut menjadi payload JSON (PostgreSQL Global Development Group, 2026; Oliveira, 2024).

Dalam penelitian ini, *middleware* Python berperan sebagai konsumen perubahan. Program membaca payload dari *replication slot*, mengenali tabel dan jenis operasi, mencari konteks koperasi atau produk, mencatat audit perubahan, dan memperbarui tabel OLAP. Peran ini sejalan dengan prinsip integrasi data yang mencakup ekstraksi, transformasi, dan pemuatan ke lingkungan analitik (Dinesh & Devi, 2024).

2.3 Pengujian Performa Query

PostgreSQL menyediakan EXPLAIN untuk menampilkan rencana eksekusi dan EXPLAIN ANALYZE untuk menghasilkan waktu aktual. Opsi BUFFERS menampilkan penggunaan blok data selama *query* dijalankan. Penelitian ini menggunakan tiga metrik, yaitu *execution time*, I/O *buffer usage*, dan *query cost*. *Execution time* menunjukkan waktu aktual; I/O *buffer* menggambarkan banyaknya blok yang diakses; sedangkan *query cost* merupakan estimasi relatif dari *query planner*. Khaerunnisa et al. (2025) menggunakan metrik serupa untuk mengevaluasi optimasi *data warehouse* PostgreSQL.

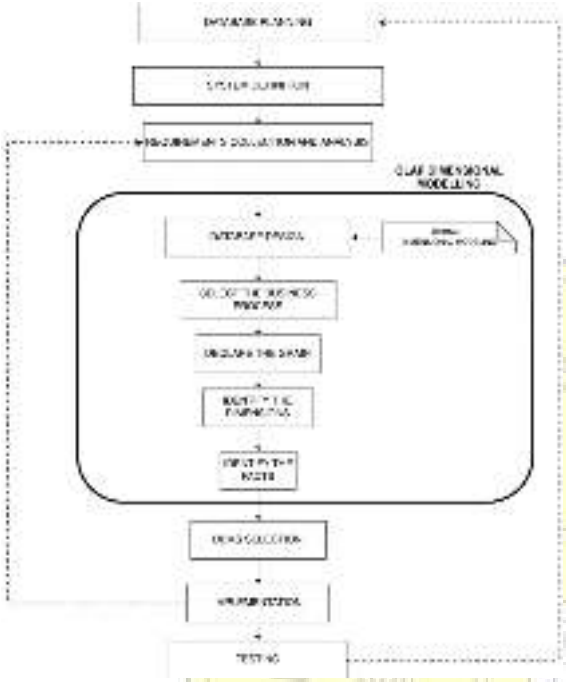
3. METODOLOGI

3.1 Tahapan Penelitian

Penelitian mengadaptasi tujuh tahap DSDLC, yaitu *Database Planning*, *System Definition*, *Requirements Collection and Analysis*, *Database Design*, *DBMS Selection*, *Implementation*, dan *Testing*. Pada tahap desain digunakan *Four-Step*

P-ISSN : 2580-4316
E-ISSN : 2654-8054 https://doi.org/10.37817/ikraith-informatika.v10i2

Dimensional Design. Apabila implementasi menemukan kebutuhan data yang belum terakomodasi, proses dikembalikan ke analisis kebutuhan. Apabila pengujian menunjukkan ketidaksesuaian mendasar terhadap tujuan atau ruang lingkup, proses dikembalikan ke perencanaan basis data. Alur tersebut diringkas pada Gambar 1.



Gambar 1. Alur penelitian berbasis DSDLC dan pemodelan dimensional

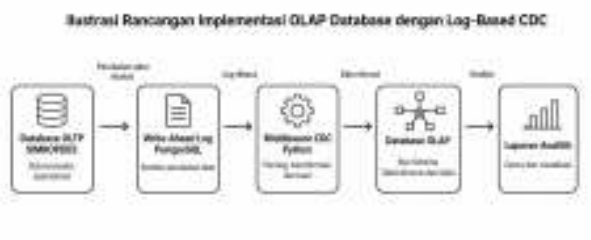
3.2 Pengumpulan dan Analisis Kebutuhan

Kebutuhan analitik dikumpulkan melalui observasi dashboard statistik SIMKOPDES dan analisis struktur tabel OLTP. Hasil observasi dikelompokkan menjadi delapan area: kelembagaan, modal, transaksi, produk, RAT, pembangunan gerai, wilayah, dan waktu. Hanya tabel sumber yang berkaitan langsung dengan indikator tersebut yang dipetakan ke OLAP.

Kebutuhan dari dashboard, tim analis, dan pimpinan memiliki irisan yang tinggi sehingga digunakan *centralized approach*. Indikator seperti total simpanan, volume dan nilai transaksi, status RAT, status lahan, dan progres pembangunan ditempatkan pada tabel fakta. Identitas koperasi, hierarki wilayah, periode, dan produk ditempatkan pada tabel dimensi.

3.3 Perancangan dan Implementasi Arsitektur

Arsitektur terdiri atas basis data OLTP sebagai sumber, WAL PostgreSQL sebagai sumber perubahan, *replication slot* dengan wal2json, *middleware* Python, dan basis data OLAP sebagai target. Struktur OLTP tidak diubah dan tidak ditambahkan *trigger*. *Middleware* menerapkan pola *query-on-event*: event CDC menjadi sinyal perubahan, kemudian nilai agregat terbaru dihitung kembali dari OLTP untuk entitas yang terdampak. Arsitektur implementasi ditunjukkan pada Gambar 2.



Gambar 2. Arsitektur sinkronisasi OLTP-CDC-OLAP

3.4 Desain Algoritma CDC

Proses awal dimulai dengan *initial load* untuk membentuk kondisi pertama tabel dimensi dan fakta. Setelah itu, perubahan dibaca menggunakan pg_logical_slot_get_changes. Fungsi pembacaan membatasi tabel yang dipantau, meminta format JSON dari wal2json, serta menyertakan informasi *timestamp*, skema, tipe operasi, dan nilai lama apabila tersedia. Potongan implementasi fungsi pembacaan ditunjukkan pada Gambar 3.

Gambar 3. Implementasi fungsi pembacaan perubahan dari replication slot

Payload yang diterima tidak selalu memuat id_koperasi atau id_produk. Perubahan pada detail penjualan, misalnya, hanya memiliki referensi transaksi. Oleh karena itu, fungsi *resolver* menelusuri relasi detail penjualan, penerimaan produk, stok, produk koperasi, dan transaksi induk untuk memperoleh konteks analitik. Prinsip yang sama diterapkan pada laporan pembangunan gerai melalui relasi ke aset gerai. Potongan fungsi *resolver* ditunjukkan pada Gambar 4.

Gambar 4. Mekanisme resolver untuk memperoleh konteks koperasi

https://doi.org/10.37817/ikraith-informatika.v10i2
P-ISSN : 2580-4316
E-ISSN : 2654-8054

Setelah konteks ditemukan, event dicatat pada fact_cdc_change_event. Pencatatan dilakukan sebelum pembaruan tabel analitik agar tersedia bukti perubahan apabila proses transformasi mengalami kegagalan. *Transform handler* kemudian menentukan fungsi penyegaran berdasarkan tabel sumber. Perubahan identitas koperasi memperbarui dimensi dan fakta kinerja; perubahan simpanan, RAT, aset, dan laporan pembangunan memperbarui fakta kinerja; sedangkan perubahan detail penjualan memperbarui fakta kinerja dan fakta transaksi produk. Alur pemilihan fungsi transformasi ditunjukkan pada Gambar 5.

Gambar 5. Implementasi transform handler berdasarkan tabel sumber

Pembaruan tabel fakta dilakukan dalam satu transaksi basis data dengan mengubah baris terkini menjadi is_current = false, kemudian memasukkan kondisi terbaru dengan effective_at berdasarkan waktu perubahan. Urutan ini mengurangi risiko terbentuknya lebih dari satu baris aktif untuk kombinasi grain yang sama. *Partial unique index* digunakan untuk membatasi hanya satu baris is_current = true pada setiap grain.

3.5 Skenario Pengujian

Pengujian performa menggunakan lima skenario analitik yang setara pada OLTP dan OLAP: S1 jumlah koperasi berdasarkan wilayah; S2 total simpanan berdasarkan wilayah; S3 nilai transaksi berdasarkan kategori produk dan periode; S4 status RAT berdasarkan wilayah; dan S5 progres pembangunan gerai berdasarkan wilayah. Setiap *query* diuji menggunakan EXPLAIN (ANALYZE, BUFFERS) terhadap tiga metrik utama, yaitu *execution time*, I/O *buffer usage*, dan *query cost*. Pengujian OLTP dan OLAP dilakukan pada konfigurasi DBMS dan sumber daya komputasi yang sama. Nilai *query cost* digunakan sebagai estimasi relatif dari *query planner*, bukan sebagai ukuran waktu aktual.

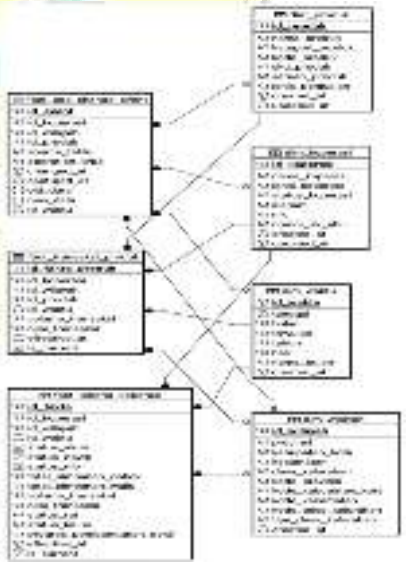
Validasi integritas membandingkan jumlah baris dan nilai agregasi hasil OLTP dengan OLAP. Uji fungsionalitas CDC mencakup tujuh skenario: pembaruan NPWP, simpanan wajib, status RAT, progres pembangunan, harga transaksi, penambahan simpanan, dan penambahan transaksi penjualan.

4. HASIL DAN PEMBAHASAN

4.1 Hasil Perancangan OLAP

Hasil desain menghasilkan empat tabel dimensi, yaitu dim_koperasi, dim_wilayah, dim_waktu, dan dim_produk, serta tiga tabel fakta, yaitu fact_cdc_change_event, fact_kinerja_koperasi, dan fact_transaksi_produk. fact_kinerja_koperasi menyimpan indikator kelembagaan, simpanan, transaksi, RAT, lahan, dan progres pembangunan. fact_transaksi_produk menyimpan volume serta nilai transaksi pada level koperasi dan produk, sedangkan fact_cdc_change_event menjadi audit teknis perubahan data.

Dimensi diperlakukan sebagai *current-state dimension* karena menyimpan atribut deskriptif terkini. Dua tabel fakta analitik menyimpan riwayat kondisi menggunakan kolom effective_at dan is_current. Ketika kondisi berubah, baris lama ditandai is_current = false dan baris baru dimasukkan sebagai kondisi terkini. Dengan demikian, data historis tidak ditimpa. Struktur hubungan antartabel diringkas pada Gambar 6.



Gambar 6. Struktur star schema OLAP SIMKOPDES

Perbedaan *grain* diterapkan secara eksplisit. Pada fact_kinerja_koperasi, volume_transaksi menunjukkan jumlah transaksi penjualan unik pada level koperasi. Pada fact_transaksi_produk, volume_transaksi menunjukkan total kuantitas produk yang terjual. Penegasan ini mencegah

P-ISSN : 2580-4316
E-ISSN : 2654-8054 https://doi.org/10.37817/ikraith-informatika.v10i2

penggunaan metrik yang sama dengan makna berbeda pada proses analisis.

4.2 Implementasi CDC dan Optimasi Fisik

Konfigurasi sumber menggunakan wal_level = logical, *replication slot*, dan plugin wal2json. *CDC worker* membaca payload JSON, mengekstraksi jenis operasi serta data lama dan baru, lalu menggunakan *resolver* untuk menentukan entitas koperasi, wilayah, atau produk. Event yang valid dicatat sebelum transformasi dilakukan sehingga tersedia jejak audit ketika terjadi kegagalan pembaruan target.

Pendekatan *query-on-event* digunakan karena sebagian indikator berasal dari agregasi beberapa tabel. Payload perubahan tidak selalu memuat seluruh konteks untuk menghitung ulang nilai fakta. Oleh sebab itu, worker melakukan *query* terbatas pada entitas terdampak, bukan membaca ulang seluruh dataset. Pendekatan ini mempertahankan konsistensi perhitungan sekaligus mengurangi pemrosesan yang tidak diperlukan.

Optimasi fisik dilakukan dengan menambahkan indeks pada kolom yang sering digunakan untuk filter dan relasi, seperti id_koperasi, id_produk, id_wilayah, id_waktu, dan is_current. Pada lima skenario *drill-down* internal OLAP, penerapan indeks menurunkan *execution time* dengan efisiensi 99,145% sampai 99,501%. Hasil ini menunjukkan bahwa keuntungan struktur dimensional tetap perlu didukung oleh rancangan fisik yang sesuai.

4.3 Perbandingan Performa dan Integritas Data

Hasil perbandingan *execution time*, I/O *buffer*, dan *query cost* untuk lima skenario masing-masing ditunjukkan pada Tabel 1, Tabel 2, dan Tabel 3. Seluruh skenario menunjukkan nilai yang lebih rendah pada OLAP. Peningkatan terbesar pada *execution time* terjadi pada S2, yaitu total simpanan berdasarkan wilayah, karena nilai simpanan telah tersedia pada tabel fakta dan tidak perlu dihitung melalui relasi OLTP yang lebih kompleks.

Pada S1, S2, S4, dan S5, *query* OLAP membaca fact_kinerja_koperasi dan dim_wilayah dengan filter is_current = true. Sebaliknya, pasangan *query* OLTP harus menelusuri tabel koperasi, desa/kelurahan, kecamatan, kabupaten/kota, provinsi, serta tabel indikator yang sesuai. Pada S3, OLTP menelusuri rantai penjualan, detail penjualan, penerimaan, stok, produk koperasi, dan master produk, sedangkan OLAP menggunakan fact_transaksi_produk, dim_produk, dan dim_waktu. Perbedaan jumlah relasi tersebut menjelaskan penurunan waktu eksekusi dan pembacaan buffer.

Setiap pasangan *query* menggunakan definisi pengelompokan dan urutan hasil yang sama. Hasil pengujian performa baru diinterpretasikan setelah jumlah baris dan nilai agregasi dinyatakan setara. Prosedur ini diperlukan agar perbedaan waktu dan biaya eksekusi merepresentasikan perbedaan struktur penyimpanan, bukan perbedaan keluaran analitik.

Tabel 1. Perbandingan execution time query OLTP dan OLAP

Skenario	Execution Time OLTP (ms)	Execution Time OLAP (ms)	Efisiensi Execution Time (%)
S1	403,645	74,508	81,541206
S2	2.257,595	88,968	96,059169
S3	2.221,003	352,484	84,129513
S4	532,057	89,743	83,132822
S5	574,560	107,425	81,303084

Tabel 2. Perbandingan I/O buffer query OLTP dan OLAP

Skenario	I/O Buffer OLTP	I/O Buffer OLAP	Efisiensi I/O Buffer (%)
S1	3.809	2.713	28,773956
S2	8.317	2.713	67,380065
S3	9.200	1.633	82,250000
S4	5.217	2.713	47,996933
S5	4.641	2.713	41,542771

Tabel 3. Perbandingan query cost query OLTP dan OLAP

Skenario	Query Cost OLTP	Query Cost OLAP	Efisiensi Query Cost (%)
S1	18.283,32	6.931,94	62,085989
S2	642.179,20	7.182,98	98,881468
S3	353.121,92	15.315,53	95,662821
S4	31.198,75	9.427,59	69,782155
S5	34.325,33	8.913,19	74,03319

Berdasarkan Tabel 1 sampai Tabel 3, seluruh skenario menunjukkan nilai execution time, I/O buffer, dan query cost yang lebih rendah pada OLAP. Rata-rata efisiensi execution time mencapai 85,23%, efisiensi I/O buffer mencapai 53,59%, dan efisiensi query cost mencapai 80,09%. Perbedaan ini terjadi karena OLAP telah menyimpan hasil transformasi dalam tabel fakta dan dimensi, sedangkan OLTP masih memerlukan banyak join, pengelompokan, dan perhitungan ulang.

Validasi integritas menunjukkan jumlah baris yang sama pada seluruh skenario: 514 baris untuk S1 dan S2, 16.337 baris untuk S3, 1.561 baris untuk S4, serta 1.473 baris untuk S5. Perbandingan nilai agregasi juga menghasilkan result_equal = true pada seluruh skenario. Dengan demikian, peningkatan performa tidak diperoleh dengan mengurangi kelengkapan hasil analitik.

4.4 Uji Fungsionalitas Sinkronisasi

Tujuh skenario perubahan berhasil ditangkap pada fact_cdc_change_event dan diterapkan ke tabel target. Ringkasan hasil ditunjukkan pada Tabel 4.

Tabel 4. Ringkasan pengujian fungsionalitas CDC

Perubahan OLTP	Hasil pada OLAP
Update NPWP	Status NPWP berubah
Update simpanan wajib	Total simpanan berubah
Update status RAT	Status RAT berubah
Update progres gerai	Progres berubah
Update harga transaksi	Nilai transaksi berubah
Insert simpanan	Total simpanan berubah
Insert penjualan	Volume dan nilai berubah

Pengujian juga menunjukkan bahwa histori kondisi dapat dipertahankan. Setiap pembaruan menghasilkan baris baru dengan is_current = true, sedangkan kondisi sebelumnya menjadi is_current = false. Pola ini mendukung analisis kondisi terkini sekaligus penelusuran perubahan berdasarkan effective_at.

Hasil penelitian memperluas fokus penelitian terdahulu yang umumnya berhenti pada perancangan *star schema* atau optimasi *data warehouse*. Pada penelitian ini, struktur dimensional dipadukan dengan sinkronisasi berbasis log dan diuji sampai aspek performa, kesetaraan hasil, serta perubahan data. Walaupun demikian, transformasi operasi DELETE dan pemantauan latensi CDC pada lingkungan produksi masih memerlukan pengembangan lebih lanjut.

4.5 Pembahasan terhadap Tujuan Penelitian

Tujuan pertama, yaitu menghasilkan skema analitik yang efisien, dipenuhi melalui pemetaan indikator dashboard ke empat dimensi dan tiga fakta. Penyederhanaan relasi tidak dilakukan dengan menyalin seluruh tabel sumber, melainkan dengan memilih atribut yang memiliki nilai analitik. Pendekatan tersebut menjaga skema tetap terarah dan menghindari pemindahan data operasional yang tidak digunakan. Penetapan *grain* yang berbeda untuk fakta kinerja dan fakta transaksi produk juga mencegah agregasi ganda ketika data dianalisis berdasarkan produk.

Tujuan kedua, yaitu sinkronisasi perubahan data, dipenuhi melalui kombinasi WAL, *logical decoding*, wal2json, dan *middleware* Python. Keuntungan utama pendekatan berbasis log adalah tidak diperlukan *trigger* pada tabel OLTP. Namun, event log tidak selalu menyediakan konteks lengkap bagi indikator analitik. Penggunaan *query-on-event* menjadi kompromi: sistem tetap membaca ulang data, tetapi pembacaan dibatasi pada koperasi atau produk yang benar-benar terdampak. Hal ini berbeda dari pemuatan ulang penuh yang memindai seluruh tabel secara berkala.

Tujuan ketiga, yaitu mengukur pengaruh OLAP terhadap performa, dipenuhi melalui *comparative benchmark* pada lima skenario yang menghasilkan keluaran setara. Hasil memperlihatkan bahwa keuntungan terbesar muncul ketika OLTP membutuhkan agregasi dari tabel yang berelasi panjang. Pada skenario dengan struktur lebih sederhana, peningkatan tetap terjadi tetapi persentasenya lebih rendah. Temuan ini menunjukkan bahwa manfaat OLAP bergantung pada kompleksitas *query*, volume data, selektivitas filter, dan rancangan indeks.

Konsistensi hasil menjadi aspek penting karena performa yang lebih cepat tidak cukup apabila nilai analitik berbeda dari sumber. Validasi jumlah baris dan agregasi memastikan bahwa transformasi menghasilkan keluaran yang sama. Selain itu, tabel audit CDC dan pola historis memungkinkan proses perubahan ditelusuri. Kombinasi tersebut meningkatkan keterjelasan aliran data dari perubahan operasional sampai indikator analitik.

Penelitian masih memiliki batasan. Operasi DELETE telah dapat dicatat sebagai event, tetapi strategi propagasi penghapusan ke tabel fakta belum menjadi fokus utama. Pengujian juga dilakukan pada lingkungan penelitian, sehingga latensi end-to-end, pertumbuhan WAL, pemantauan *replication slot*, pengelolaan kegagalan berulang, serta kebutuhan *backup* dan pemulihan pada lingkungan produksi belum dievaluasi secara menyeluruh.

5. KESIMPULAN

Arsitektur OLAP SIMKOPDES berhasil dibangun sebagai lingkungan analitik terpisah dari basis data OLTP. Rancangan menggunakan empat tabel dimensi dan tiga tabel fakta, dengan dimensi yang menyimpan atribut deskriptif terkini serta tabel fakta historis yang menggunakan effective_at dan is_current. Struktur tersebut menyederhanakan kebutuhan analisis multidimensi berdasarkan koperasi, wilayah, waktu, dan produk.

Mekanisme *log-based CDC* berhasil meneruskan perubahan dari WAL PostgreSQL melalui *replication slot*, wal2json, dan *middleware* Python. Pola *query-on-event* serta fungsi *resolver* memungkinkan perubahan pada tabel turunan tetap dikaitkan dengan entitas analitik yang terdampak. Pada tujuh skenario uji, event berhasil dicatat dan nilai OLAP diperbarui dengan mempertahankan histori.

Dibandingkan OLTP, OLAP menghasilkan rata-rata efisiensi 85,23% pada *execution time*, 53,59% pada penggunaan I/O *buffer*, dan 80,09% pada *query cost*. Pada seluruh lima skenario pengujian, jumlah baris dan nilai agregasi OLAP konsisten dengan hasil OLTP. Dengan demikian, pemisahan beban analitik melalui OLAP yang disinkronkan menggunakan *log-based CDC* efektif untuk meningkatkan efisiensi *query* sekaligus menjaga konsistensi data SIMKOPDES.

6. UCAPAN TERIMA KASIH

Penulis menyampaikan terima kasih kepada Program Studi Informatika dan Fakultas Ilmu Komputer Universitas Singaperbangsa Karawang, dosen pembimbing, serta Kementerian Koperasi Republik Indonesia yang telah mendukung pelaksanaan penelitian ini.

DAFTAR PUSTAKA

Connolly, T., & Begg, C. (2015). *Database systems: A practical approach to design, implementation, and management* (6th ed.). Pearson.

Dinesh, L., & Devi, K. G. (2024). An efficient hybrid optimization of ETL process in data warehouse of cloud architecture. *Journal of Cloud Computing, 13*, Article 12. https://doi.org/10.1186/s13677-023-00571-y

Garani, G., Tolis, D., & Savvas, I. K. (2023). A trajectory data warehouse solution for workforce management decision-making. *Data Science and Management, 6*(2), 88-97. https://doi.org/10.1016/j.dsm.2023.03.002

Gomstyn, A., & Jonker, A. (2025). *What is change data capture?* IBM.

Khaerunnisa, L. S., Al Qodr, M. R. S., Putri, J. W. D., Firdaus, J. R., & Rozikin, C. (2025). Optimasi proses data warehouse menggunakan partisi dan indexing pada PostgreSQL untuk meningkatkan performa query. *JITET (Jurnal Informatika dan Teknik Elektro Terapan), 13*(3S1), 1109-1120. https://doi.org/10.23960/jitet.v13i3S1.8012

Kimball, R., & Ross, M. (2013). *The data warehouse toolkit: The definitive guide to dimensional modeling* (3rd ed.). Wiley.

Oliveira, E. T. (2024). *wal2json: JSON output plugin for changeset extraction*. GitHub.

PostgreSQL Global Development Group. (2026). *PostgreSQL 18 documentation*.

Santosa, I., Cahyadi, A., Suhartana, I., & Rahayuda, I. G. S. (2025). Data processing using change data capture for real-time data synchronization. *JELIKU (Jurnal Elektronik Ilmu Komputer Udayana), 13*(4), 743-754. https://doi.org/10.24843/JLK.2025.v13.i04.p02

Santosa, I. M. A. M., & Putra, I. G. N. A. C. (2023). Analisis performa sistem pada pemisahan database analitik dengan transaksional. *Jurnal Nasional Teknologi Informasi dan Aplikasinya, 1*(3), 847-856.

Wijaya, W., Wiratama, J., & Wijaya, S. F. (2024). The implementation of data warehouse and star schema for optimizing property business decision making. *G-Tech: Jurnal Teknologi Terapan, 8*(2), 1242-1250. https://doi.org/10.33379/gtech.v8i2.4091