NARATIF : Jurnal Ilmiah Nasional Riset Aplikasi dan Teknik Informatika
Vol. 07 No. 02 Desember 2025
P-ISSN: 2656-7377 || E-ISSN: 2714-8467

INFORMATION RETRIEVAL DAN *DATABASE SYSTEMS: SYSTEMATIC LITERATURE REVIEW* TENTANG ANALISIS KOMPARATIF DAN STRATEGI INTEGRASI

Wemby Juniarrochman[1], Rachmat Selamat[2]
Sistem Informasi[1,2]
STIMIK LIKMI[1,2]
wemby.juniarrochman@gmail.com[1], rachmatselametskom@gmail.com[2]

Abstrak

Era digital telah menghadirkan tantangan kompleks dalam manajemen informasi seiring dengan meningkatnya volume data secara eksponensial, sehingga menuntut sistem yang mampu mengelola dan menyediakan akses informasi secara efektif. *Information Retrieval* (IR) Systems dan *Database Systems* merupakan dua paradigma fundamental yang berkembang untuk menjawab tantangan tersebut, namun keduanya memiliki perbedaan mendasar dalam konsep, pendekatan, dan mekanisme kerja. Permasalahan utama dalam penelitian ini adalah memahami perbedaan fundamental antara IR Systems dan *Database Systems* serta mengidentifikasi strategi integrasi yang optimal agar keduanya dapat saling melengkapi dalam mendukung kebutuhan aplikasi modern. Penelitian ini dilakukan melalui tinjauan literatur sistematis dengan menganalisis 50 publikasi akademik pada periode 2015–2025, disertai kajian dokumentasi teknis sistem dan analisis *case studies* yang relevan. Analisis difokuskan pada lima dimensi utama, yaitu model data, pemrosesan kueri, evaluasi performa, arsitektur sistem, dan aplikasi. Hasil kajian menunjukkan adanya perbedaan fundamental pada model data yang digunakan (schema-less dibandingkan schema-based), semantik kueri yang diterapkan (pencocokan aproksimasi dibandingkan pencocokan eksak), serta metrik evaluasi performa (precision–recall dibandingkan throughput–latency). Selain itu, penelitian ini mengidentifikasi tiga strategi integrasi yang umum digunakan, yaitu *tight integration* yang mengintegrasikan fungsi IR secara langsung ke dalam DBMS, *loose integration* yang memisahkan kedua sistem dengan koordinasi pada lapisan aplikasi, serta *hybrid architecture* yang mengombinasikan pendekatan integrasi secara selektif berdasarkan karakteristik *workload*. Kontribusi penelitian ini meliputi penyusunan kerangka komprehensif untuk analisis komparatif, pengembangan taksonomi karakteristik sistem dan strategi integrasi, penyediaan pedoman pemilihan sistem, serta identifikasi tren konvergensi antara IR Systems dan Database Systems.
Kata kunci : *Information Retrieval, Database Systems, Query Processing, System Integration, Hybrid Architecture*

Abstract

The digital era has introduced complex challenges in information management as data volumes grow exponentially, requiring systems that are capable of efficiently managing and providing access to information. Information Retrieval (IR) systems and Database Systems represent two fundamental paradigms that have evolved to address these challenges; however, they differ significantly in their underlying concepts, approaches, and operational mechanisms. The main problem addressed in this study is to understand the fundamental differences between IR systems and Database Systems and to identify optimal integration strategies that allow both paradigms to complement each other in supporting modern application requirements. This research is conducted through a systematic literature review analyzing 50 academic publications published between 2015 and 2025, complemented by technical system documentation and relevant case study analyses. The analysis focuses on five key dimensions, namely data models, query processing, performance evaluation, system architecture, and applications. The results reveal fundamental differences in data models (schema-less versus schema-based), query semantics (approximate matching versus exact matching), and performance evaluation metrics (precision–recall versus throughput–latency). Furthermore, the study identifies three commonly adopted integration strategies: tight integration, in which IR functionalities are embedded directly within a DBMS; loose integration, where the two systems operate separately with coordination at the application layer; and hybrid architecture, which selectively combines integration approaches based on workload characteristics. The contributions of this research include the development of a comprehensive framework for comparative analysis, the formulation of a taxonomy of system characteristics and integration strategies, the provision of guidelines for system selection, and the identification of convergence trends between IR systems and Database Systems.
Keywords : Information Retrieval, Database Systems, Query Processing, System Integration, Hybrid Architecture

I. PENDAHULUAN

Dunia digital saat ini menghadapi tantangan besar dalam mengelola informasi. Bayangkan sebuah perpustakaan yang setiap harinya menerima jutaan buku, majalah, dan dokumen baru. Itulah gambaran dari situasi data global kita volume informasi tumbuh sangat cepat hingga mencapai ratusan Zettabyte. Organisasi modern, baik perusahaan besar maupun startup kecil, harus menangani berbagai jenis informasi: dari data transaksi yang terstruktur rapi seperti tabel keuangan, hingga konten bebas seperti email, dokumen, dan postingan media sosial [1].

Untuk mengatasi tantangan ini, teknologi informasi telah mengembangkan dua pendekatan utama yang sangat berbeda. Pendekatan pertama adalah *Database Systems* atau sistem basis data. Sistem ini lahir dari kebutuhan dunia bisnis yang menginginkan cara terorganisir untuk menyimpan dan mengelola data. Seperti lemari arsip yang sangat teratur, setiap informasi memiliki tempat yang pasti dan mengikuti aturan yang ketat. Sistem ini berkembang sejak tahun 1960-an dan mencapai bentuk modernnya dengan model relasional yang diperkenalkan oleh E.F. Codd pada tahun 1970. Keunggulan utamanya adalah jaminan keandalan melalui properti ACID sebuah prinsip yang memastikan bahwa setiap transaksi terjadi dengan sempurna atau tidak terjadi sama sekali [2].

Pendekatan kedua adalah *Information Retrieval Systems* atau sistem pencarian informasi. Sistem ini berasal dari dunia perpustakaan dan dokumentasi, di mana masalahnya bukan menyimpan data secara terstruktur, melainkan menemukan informasi yang relevan dari tumpukan dokumen yang sangat besar. Berbeda dengan basis data yang mengharapkan data terstruktur rapi, sistem IR dirancang untuk menangani dokumen dengan format bebas artikel, laporan, email, atau apa pun. Yang lebih menarik, sistem ini tidak mencari kecocokan yang persis, melainkan mencoba memahami apa yang pengguna maksudkan dan menemukan dokumen yang paling relevan, meskipun kata-katanya berbeda [3].

Saat ini, banyak aplikasi modern membutuhkan kemampuan dari kedua pendekatan sekaligus. Ambil contoh toko *online* seperti Tokopedia atau Shopee. Di satu sisi, mereka memerlukan sistem pencarian produk yang canggih pengguna bisa mengetik 'sepatu olahraga pria murah' dan sistem harus mengerti maksudnya, mencari produk yang sesuai meskipun deskripsi produk menggunakan kata-kata berbeda. Di sisi lain, ketika pengguna membeli produk, transaksi harus diproses dengan sempurna uang didebet dari saldo, stok dikurangi, pesanan dicatat semua harus terjadi dengan benar atau tidak terjadi sama sekali. Tidak boleh ada situasi di mana uang hilang tapi pesanan tidak tercatat.

Contoh lain adalah sistem pencarian dokumen perusahaan. Karyawan perlu mencari berbagai dokumen memo, laporan, presentasi dengan kata kunci yang samar. Sistem harus bisa menemukan dokumen yang relevan meskipun kata kunci tidak cocok persis. Namun pada saat yang sama, sistem juga harus menjaga struktur metadata, hubungan antar dokumen, dan hak akses dengan ketat. Sistem analitik modern juga menggabungkan kedua kebutuhan ini mereka perlu melakukan pencarian teks untuk menganalisis dokumen, sekaligus menjalankan *query* SQL untuk mengolah data terstruktur [4].

Meskipun kedua sistem ini sudah digunakan secara luas, masih banyak yang belum dipahami dengan baik tentang kapan menggunakan sistem mana, atau bagaimana menggabungkan keduanya. Penelitian ini fokus pada tiga masalah utama yang sering membingungkan praktisi dan peneliti. Masalah pertama adalah kurangnya pemahaman sistematis tentang apa yang benar-benar membedakan kedua sistem ini. Banyak orang tahu bahwa sistem IR bagus untuk pencarian teks dan basis data bagus untuk data terstruktur, tapi pemahaman ini terlalu sederhana. Apa sebenarnya perbedaan mendasar dalam cara keduanya menyimpan data, memproses pencarian, dan mengukur keberhasilan? Mengapa sistem IR menggunakan *inverted index* sementara basis data menggunakan *B-tree*? Apa konsekuensi dari pilihan desain ini? Literatur yang ada sering membahas kedua sistem secara terpisah, jarang ada yang membandingkannya secara mendalam dan sistematis.

Masalah kedua adalah ketidakjelasan dalam strategi integrasi. Ketika sebuah aplikasi membutuhkan kemampuan dari kedua sistem, apa pilihan yang tersedia? Apakah lebih baik menambahkan fitur pencarian ke dalam basis data yang sudah ada? Atau lebih baik menggunakan dua sistem terpisah yang bekerja bersama? Atau ada pendekatan lain yang lebih baik? Untuk setiap pilihan, apa kelebihan dan kekurangannya? Dalam situasi apa satu pendekatan lebih baik dari yang lain? Pertanyaan-pertanyaan ini belum terjawab dengan jelas dalam literatur, menyebabkan banyak organisasi membuat keputusan arsitektur yang tidak optimal.

Masalah ketiga adalah kurangnya pemahaman tentang arah perkembangan teknologi. Sistem basis data modern seperti PostgreSQL kini memiliki kemampuan pencarian teks yang semakin baik. Sistem pencarian seperti Elasticsearch mulai mendukung operasi data terstruktur. Teknologi baru seperti *vector databases* muncul di tengah-tengah, menawarkan kemampuan dari kedua sisi. Ke mana arah teknologi ini? Apakah suatu hari nanti batas antara keduanya akan hilang sama sekali? Memahami tren ini penting untuk membuat keputusan teknologi yang tidak cepat usang. Analisis terhadap 50 publikasi mengidentifikasi tiga gap literatur utama: (1) Tidak ada framework sistematis yang membandingkan IR dan DBMS dari perspektif holistik - studi *existing* fokus pada aspek individual; (2) Kurangnya pedoman praktis untuk memilih strategi integrasi berdasarkan karakteristik *workload*; (3) Tidak ada analisis komprehensif tentang trade-offs antara konsistensi transaksional dan kemampuan pencarian. Untuk menjawab pertanyaan-pertanyaan ini, tinjauan literatur sistematis (*systematic literature review*) ini menggunakan pendekatan analisis komparatif terhadap literatur yang sudah ada untuk membangun pemahaman komprehensif tentang kedua paradigma sistem. Bayangkan seperti menyusun *puzzle* besar dari berbagai sumber informasi. Penelitian ini mengumpulkan dan menganalisis 50 publikasi ilmiah terpilih dari jurnal dan konferensi ternama di bidang pencarian informasi dan basis data, seperti ACM SIGIR, VLDB, IEEE TKDE, dan ACM TODS. Publikasi dipilih dari tahun 2015 hingga 2025 untuk memastikan informasi yang relevan dengan teknologi terkini. Perlu ditekankan bahwa penelitian ini tidak melakukan eksperimen atau implementasi sistem baru, melainkan menganalisis dan mensintesis temuan dari literatur yang sudah ada untuk membangun pemahaman komprehensif tentang kedua paradigma dan strategi integrasinya.

Analisis dilakukan dengan membandingkan kedua sistem pada lima aspek penting. Pertama, bagaimana cara keduanya menyimpan dan merepresentasikan data. Kedua, bagaimana mereka memproses permintaan pencarian atau *query*. Ketiga, bagaimana mereka mengukur keberhasilan dan kinerja. Keempat, bagaimana arsitektur dan komponen internal mereka dirancang. Kelima, untuk aplikasi apa masing-masing paling cocok. Dengan membandingkan secara sistematis pada kelima aspek ini, penelitian dapat mengidentifikasi perbedaan fundamental dan memahami mengapa perbedaan tersebut ada.

Selain publikasi ilmiah, penelitian juga mempelajari dokumentasi teknis dari sistem-sistem yang benar-benar digunakan di dunia nyata. Ini termasuk sistem *open-source* populer seperti PostgreSQL dan MySQL untuk basis data, serta Elasticsearch dan Apache Solr untuk pencarian. Penelitian juga menganalisis studi kasus dari berbagai perusahaan tentang bagaimana mereka menyelesaikan masalah integrasi mulai dari toko *online* besar, sistem pencarian perusahaan,

hingga platform media sosial. Pendekatan multi-sumber ini memberikan perspektif yang seimbang antara teori akademik dan praktik industri.

Penelitian ini diharapkan memberikan kontribusi praktis yang dapat langsung digunakan oleh praktisi dan peneliti. Kontribusi pertama adalah sebuah Framework komparatif lima dimensi (model data, query processing, evaluasi performa, arsitektur, dan aplikasi) yang membedakan penelitian ini dari studi sebelumnya yang hanya membandingkan aspek teknis terisolasi. Berbeda dengan Framework Stonebraker (2005) yang fokus pada 'one size fits all', atau analisis Zhang & Wang (2021) yang terbatas pada strategi integrasi, framework kami menyediakan analisis holistik yang menghubungkan karakteristik teknis dengan keputusan arsitektur Framework ini seperti sebuah 'kacamata' yang membantu melihat perbedaan dan persamaan antara kedua sistem dengan jelas. Dengan framework ini, praktisi dapat menganalisis kebutuhan aplikasi mereka dan memahami sistem mana yang paling cocok, atau apakah perlu menggabungkan keduanya.

Kontribusi kedua adalah Taksonomi tiga-lapis strategi integrasi (*tight, loose, hybrid*) dengan kriteria pemilihan berbasis *workload* yang tidak ada dalam literatur sebelumnya. Penelitian akan menjelaskan setiap strategi dengan jelas: apa itu, bagaimana cara kerjanya, apa kelebihan dan kekurangannya, dan dalam situasi apa strategi tersebut paling cocok. Ini seperti panduan yang menjelaskan berbagai cara untuk menggabungkan dua alat yang berbeda, lengkap dengan petunjuk kapan menggunakan cara yang mana. Taksonomi ini akan membantu arsitek sistem membuat keputusan yang tepat berdasarkan konteks spesifik mereka.

Kontribusi ketiga adalah pedoman praktis yang dapat langsung diterapkan. Pedoman ini akan mencakup kriteria pengambilan keputusan yang jelas: jika aplikasi Anda memiliki karakteristik X, maka strategi Y mungkin paling cocok. Jika volume pencarian Anda mencapai Z, pertimbangkan untuk menggunakan sistem terpisah. Pedoman ini akan menghemat waktu dan mengurangi risiko kesalahan dalam pemilihan teknologi, terutama untuk tim yang belum berpengalaman dengan integrasi sistem kompleks.

Kontribusi keempat adalah identifikasi tren teknologi masa depan. Dengan memahami bagaimana teknologi berkembang, praktisi dapat membuat keputusan yang lebih *future-proof* tidak cepat usang. Peneliti dapat menggunakan analisis tren ini untuk mengidentifikasi area yang menjanjikan untuk penelitian lebih lanjut. Misalnya, peran *vector databases* dan model bahasa besar dalam mengubah paradigma pencarian dan manajemen data.

Secara keseluruhan, penelitian ini bertujuan menjembatani kesenjangan pemahaman antara teori dan praktik, antara sistem IR dan basis data, serta memberikan landasan yang solid untuk membangun sistem manajemen informasi yang efektif. Dengan pemahaman yang lebih mendalam tentang kekuatan dan keterbatasan masing-masing sistem, organisasi dapat mendesain arsitektur yang menggabungkan yang terbaik dari kedua dunia kemampuan pencarian yang kuat dan keandalan data yang tinggi.

II. TINJAUAN PUSTAKA

1. Penelitian Tedahulu

Upaya mengintegrasikan sistem IR dan basis data telah menjadi fokus berbagai penelitian dalam dekade terakhir, dengan pendekatan yang bervariasi dari implementasi spesifik hingga analisis arsitektural. Chen et al. (2020) mengembangkan solusi hybrid untuk *e-commerce scale* yang menggabungkan Elasticsearch dengan PostgreSQL, mencapai peningkatan kecepatan pencarian 40% sambil mempertahankan jaminan transaksional [11]. Zhang dan Wang (2021) melakukan studi komparatif terhadap berbagai strategi integrasi, menemukan bahwa tidak ada solusi universal pilihan optimal bergantung pada karakteristik workload spesifik [12]. Kumar dan rekan (2022) mengeksplorasi pendekatan *machine learning* untuk *query routing* otomatis, berhasil mengurangi waktu respons 30% melalui pembelajaran pola penggunaan [13]. Sementara itu, Patel dan Smith (2023) mengeksplorasi vector databases sebagai jembatan antara IR dan DBMS, khususnya untuk aplikasi AI-powered [14], dan Liu et al. (2024) mengusulkan arsitektur cloud-native berbasis microservices yang memisahkan komponen pencarian dan basis data untuk fleksibilitas maksimal [15].

Analisis terhadap literatur existing mengidentifikasi beberapa pola konsisten namun juga keterbatasan signifikan. Pertama, sebagian besar penelitian fokus pada konteks implementasi spesifik (*e-commerce, enterprise search,* media sosial) tanpa mengabstraksi prinsip-prinsip umum yang dapat diterapkan lintas domain. Kedua, meskipun berbagai strategi integrasi telah diusulkan, tidak ada taksonomi sistematis yang mengklasifikasikan strategi berdasarkan karakteristik teknis dan *trade-offs* yang terlibat. Ketiga, diskusi tentang *trade-offs* antara konsistensi transaksional dan kemampuan pencarian masih tersebar dan tidak komprehensif beberapa studi menekankan pentingnya ACID tanpa mempertimbangkan implikasi terhadap performa pencarian, sementara yang lain fokus pada optimisasi IR tanpa *adequately addressing data integrity concerns*. Keempat, framework untuk pengambilan keputusan arsitektural yang membantu praktisi memilih strategi berdasarkan karakteristik *workload* mereka masih *absent* atau sangat terbatas.

Meskipun studi-studi di atas memberikan wawasan berharga tentang implementasi dan performa spesifik, masih terdapat gap signifikan yang perlu dijembatani: (1) Tidak ada framework sistematis yang mengintegrasikan temuan dari berbagai domain dan menyediakan analisis komparatif holistik terhadap karakteristik fundamental IR dan DBMS penelitian *existing* cenderung membandingkan aspek terisolasi (misalnya hanya indexing strategy atau hanya consistency model); (2) Kurangnya kriteria eksplisit dan terukur untuk pemilihan strategi integrasi berdasarkan parameter workload yang konkret (*volume data, query*

complexity, consistency requirements, latency tolerance) panduan yang ada terlalu abstrak atau hanya berlaku untuk kasus tertentu; (3) Tidak ada analisis mendalam tentang trade-offs antara *consistency* vs *search capability* yang menjelaskan implikasi teknis dan bisnis dari setiap pilihan arsitektural; (4) Kurangnya identifikasi tren konvergensi antara IR dan DBMS yang dapat menginformasikan arah teknologi masa depan. Penelitian ini berusaha menjembatani gap-gap tersebut dengan menyediakan framework komprehensif, taksonomi terstruktur, dan pedoman praktis yang divalidasi terhadap *case studies* dari berbagai domain industri.

2. *Information Retrieval Systems*

   Information Retrieval Systems didesain untuk menemukan informasi relevan dari koleksi dokumen tidak terstruktur. Berbeda dengan sistem basis data, IR menggunakan *approximate matching* dengan model perankingan seperti Vector Space dengan TF-IDF [5], probabilistik dengan BM25 [4], dan neural dengan BERT [6] untuk menangani variabilitas bahasa natural berdasarkan prinsip fundamental IR [1][3]. Karakteristik fundamental IR meliputi: (1) schema-less indexing menggunakan inverted index, (2) *relevance scoring* berbasis *statistical/semantic similarity*, (3) *evaluation metrics* (*precision, recall*, MAP) yang fokus pada *quality of ranking*. Detail implementasi dibahas dalam analisis komparatif.

3. *Database Systems*

   Database Systems dibangun untuk mengelola data terstruktur dengan jaminan integritas yang ketat. Properti ACID memastikan keandalan transaksi [8], sementara *schema rigid* memungkinkan validasi data dan *query optimization* [2][7]. Evolusi DBMS mencakup NoSQL untuk *scalability* [9] dan NewSQL untuk *distributed* ACID [8][10]. Trade-off CAP theorem menjadi pertimbangan arsitektur modern [19].

4. Kerangka Pemikiran

   Berdasarkan tinjauan literatur di atas, penelitian ini membangun kerangka pemikiran untuk membandingkan dan mengintegrasikan sistem IR dan basis data. Kerangka ini mengidentifikasi lima dimensi penting yang perlu dianalisis. Dimensi pertama adalah model data bagaimana masing-masing sistem menyimpan dan mengorganisasi informasi. Sistem IR menggunakan pendekatan tanpa skema yang fleksibel, cocok untuk dokumen dengan struktur bervariasi. Basis data menggunakan skema yang ketat, cocok untuk data terstruktur yang konsisten. Memahami perbedaan ini penting untuk memilih sistem yang tepat.

   Dimensi kedua adalah pemrosesan *query* bagaimana sistem mencari dan mengambil data. Sistem IR fokus pada relevansi dengan perankingan berdasarkan kemiripan semantik. Basis data fokus pada presisi dengan hasil yang eksak. Perbedaan ini mencerminkan tujuan yang berbeda: menemukan informasi yang mungkin berguna versus mengambil data yang pasti benar.

   Dimensi ketiga adalah evaluasi kinerja bagaimana mengukur keberhasilan sistem. Sistem IR diukur dengan metrik seperti *precision* (berapa persen hasil yang relevan) dan *recall* (berapa persen dokumen relevan yang ditemukan). Basis data diukur dengan *throughput* (berapa banyak transaksi per detik) dan latensi (seberapa cepat respons). Metrik yang berbeda ini menunjukkan prioritas yang berbeda.

   Dimensi keempat adalah arsitektur sistem bagaimana komponen internal dirancang dan bekerja sama. Sistem IR biasanya menggunakan arsitektur terdistribusi yang mudah diskalakan secara horizontal dengan menambah *server*. Basis data tradisional lebih sering menggunakan skalabilitas vertikal dengan meningkatkan kekuatan *server* tunggal, meskipun tren modern bergerak ke arah distribusi.

   Dimensi kelima adalah aplikasi dan *use cases* untuk masalah apa masing-masing sistem paling cocok. Sistem IR unggul untuk pencarian konten, analisis teks, dan aplikasi di mana fleksibilitas lebih penting daripada konsistensi ketat. Basis data unggul untuk transaksi bisnis, aplikasi keuangan, dan situasi di mana integritas data adalah prioritas utama.

   Kerangka pemikiran juga mengeksplorasi tiga strategi utama untuk integrasi. *Tight integration* menambahkan kemampuan pencarian ke dalam basis data sederhana tapi terbatas. *Loose integration* menggunakan sistem terpisah yang bekerja bersama kompleks tapi *powerful*. *Hybrid architecture* menggabungkan keduanya dengan kecerdasan di tingkat aplikasi paling fleksibel tapi paling menantang untuk diimplementasikan. Kerangka ini memberikan struktur sistematis untuk menganalisis, membandingkan, dan memilih solusi yang tepat. Dengan memahami kelima dimensi dan tiga strategi integrasi, praktisi dapat membuat keputusan yang lebih tepat berdasarkan kebutuhan spesifik aplikasi mereka.

III. METODE PENELITIAN

Penelitian ini menggunakan pendekatan kualitatif dengan metode analisis literatur yang sistematis dan mendalam. Bayangkan penelitian ini seperti seorang detektif yang mengumpulkan berbagai petunjuk dari berbagai sumber, kemudian menyusunnya menjadi gambaran yang lengkap dan jelas. Bedanya, 'petunjuk' yang dikumpulkan di sini adalah pengetahuan dari publikasi ilmiah, dokumentasi sistem, dan pengalaman nyata dari berbagai organisasi.

1. Desain Penelitian

   Penelitian ini adalah *systematic literature review* yang menggunakan pendekatan kualitatif. Desain penelitian ini dibangun dengan pendekatan analisis komparatif multidimensi. Analoginya seperti membandingkan dua mobil yang dirancang untuk tujuan berbeda satu untuk balap di sirkuit, satu untuk perjalanan keluarga. Kita tidak bisa hanya membandingkan kecepatan maksimum, tapi harus melihat berbagai aspek: efisiensi bahan bakar,

kenyamanan, kapasitas penumpang, biaya perawatan, dan seterusnya. Demikian juga dengan sistem IR dan basis data perbandingan harus dilakukan dari berbagai sudut pandang untuk mendapatkan pemahaman yang utuh.

Penelitian dibagi menjadi lima tahap yang saling terkait. Tahap pertama adalah identifikasi dimensi kunci untuk perbandingan. Ini seperti menentukan kriteria apa saja yang akan digunakan untuk membandingkan kedua mobil tadi. Setelah diskusi mendalam dengan literatur dan mempertimbangkan kebutuhan praktis, diidentifikasi lima dimensi: bagaimana data disimpan, bagaimana pencarian dilakukan, bagaimana kinerja diukur, bagaimana sistem dibangun, dan untuk aplikasi apa masing-masing cocok.

Tahap kedua adalah pengumpulan data secara sistematis dari berbagai sumber. Tahap ketiga melibatkan analisis mendalam terhadap karakteristik masing-masing sistem pada setiap dimensi. Tahap keempat adalah identifikasi pola, perbedaan, dan persamaan antar sistem. Tahap kelima adalah sintesis temuan menjadi framework yang koheren dan berguna. Kelima tahap ini tidak berjalan linear tapi iteratif kadang perlu kembali ke tahap sebelumnya ketika menemukan informasi baru atau perspektif yang berbeda.

2. Sumber Data

Sumber data penelitian dipilih dengan sangat hati-hati untuk memastikan kualitas dan relevansi. Seperti memasak makanan berkualitas yang memerlukan bahan-bahan terbaik, penelitian yang baik memerlukan sumber informasi yang terpercaya dan relevan. Penelitian ini menggunakan tiga kategori sumber data yang saling melengkapi. Kategori pertama adalah publikasi akademik dari jurnal dan konferensi terkemuka. Ini seperti mendapatkan pendapat dari para ahli terbaik di bidangnya. Fokusnya adalah pada publikasi dari *venue* bergengsi seperti ACM SIGIR untuk *information retrieval*, VLDB dan IEEE TKDE untuk sistem basis data, serta ACM TODS untuk teori basis data. Mengapa jurnal-jurnal ini? Karena mereka memiliki proses *peer review* yang ketat, memastikan bahwa hanya penelitian berkualitas tinggi yang diterbitkan.

Dari ratusan publikasi yang tersedia, dipilih 50 yang paling relevan. Pemilihan ini tidak dilakukan secara acak, melainkan melalui kriteria yang jelas. Pertama, publikasi harus dari tahun 2015 hingga 2025 untuk memastikan relevansi dengan teknologi terkini. Kedua, publikasi harus memiliki kontribusi signifikan terhadap pemahaman karakteristik sistem atau strategi integrasi. Ketiga, publikasi harus memiliki metodologi yang solid dan hasil yang dapat dipercaya. Proses seleksi ini seperti menyaring emas dari pasir memerlukan waktu dan ketelitian, tapi hasilnya sangat berharga.

Kategori kedua adalah dokumentasi teknis dari sistem yang benar-benar digunakan di industri. Ini memberikan perspektif praktis yang sangat penting. Publikasi akademik memberikan teori dan prinsip, tapi dokumentasi teknis menunjukkan bagaimana teori tersebut diterapkan dalam sistem nyata. Sistem yang dipelajari mencakup PostgreSQL dan MySQL sebagai representasi basis data relasional, Elasticsearch dan Apache Solr sebagai representasi sistem IR, serta MongoDB dan Redis sebagai representasi sistem NoSQL. Setiap sistem dipilih karena popularitasnya yang tinggi dan dokumentasi yang lengkap.

Kategori ketiga adalah *case studies* atau studi kasus dari berbagai organisasi. Ini seperti belajar dari pengalaman orang lain apa yang berhasil, apa yang gagal, dan pelajaran apa yang dapat diambil. *Case studies* dikumpulkan dari berbagai domain: *e-commerce* besar seperti Amazon dan Alibaba, sistem pencarian perusahaan dari IBM dan Microsoft, platform media sosial seperti Twitter dan LinkedIn, serta layanan keuangan dari bank-bank besar. Setiap domain memiliki tantangan unik dan solusi yang berbeda, memberikan wawasan yang kaya tentang strategi integrasi dalam konteks yang beragam.

3. Kriteria Seleksi dan Proses *Review*

Proses seleksi publikasi mengikuti protokol PRISMA (*Preferred Reporting Items for Systematic Reviews*) untuk memastikan transparansi dan reprodusibilitas.

Kriteria Inklusi:

a. Publikasi akademik *peer-reviewed* dari jurnal atau konferensi bereputasi tinggi (ACM, IEEE, Springer).
b. Periode publikasi: 2015-2025 (10 tahun terakhir untuk memastikan relevansi teknologi).
c. Fokus topik: perbandingan IR dan DBMS, strategi integrasi, atau arsitektur hybrid.
d. Tersedia dalam bahasa Inggris dengan *full-text access*.
e. Mengandung data empiris, *case studies*, atau analisis teknis yang substantif.

Kriteria Ekslusi:

a. Publikasi non-peer-reviewed (*technical reports, white papers, blog posts*).
b. Artikel yang hanya membahas satu sistem tanpa analisis komparatif.
c. Publikasi yang fokus pada domain spesifik terbatas (misal: hanya bioinformatics).
d. Tutorial, survey pendek, atau *position papers* tanpa kontribusi teknis.
e. Publikasi dengan metodologi yang tidak jelas atau hasil yang tidak dapat diverifikasi.

Proses Seleksi Tiga Tahap:

a. Tahap 1 - *Initial Search*: Pencarian di ACM Digital Library, IEEE Xplore, Springer menghasilkan 347 publikasi potensial.
b. Tahap 2 - *Title/Abstract Screening*: Setelah screening judul dan abstrak berdasarkan kriteria inklusi/eksklusi, tersisa 128 publikasi.

c. Tahap 3 - *Full-Text Assessment*: Pembacaan lengkap dan evaluasi kualitas menghasilkan 50 publikasi final yang memenuhi semua kriteria.

Potensi Bias Seleksi:

a. *Publication bias*: Kemungkinan literatur lebih banyak melaporkan implementasi yang sukses daripada yang gagal.
b. *Language bias*: Hanya publikasi bahasa Inggris, berpotensi melewatkan penelitian penting dalam bahasa lain.
c. *Recency bias*: Penekanan pada publikasi terkini (70% dari 2020-2025) mungkin *underrepresent* pendekatan klasik yang masih relevan.
d. *Database bias*: Pencarian terbatas pada repositori akademik utama, berpotensi melewatkan publikasi dari *venue emerging*.

Mitigasi Bias:

a. Melakukan *backward/forward citation chaining* dari publikasi kunci untuk mengidentifikasi penelitian relevan yang mungkin terlewat Melakukan *backward/forward citation chaining* dari publikasi kunci untuk mengidentifikasi penelitian relevan yang mungkin terlewat.
b. Memasukkan dokumentasi teknis dan *case studies* industri untuk menyeimbangkan perspektif akademik.
c. Melakukan triangulasi temuan dari berbagai sumber (akademik, industri*, open-source*)

4. Teknik Pengumpulan Data

Pengumpulan data dilakukan secara sistematis menggunakan strategi pencarian yang terstruktur. Proses dimulai dengan mendefinisikan kata kunci pencarian yang mencakup berbagai aspek topik penelitian. Kata kunci utama mencakup *"information retrieval", "database systems", "query processing", "system integration", "full-text search",* dan *"hybrid architecture"*. Kombinasi kata kunci ini digunakan untuk mencari di berbagai basis data akademik seperti ACM Digital Library, IEEE Xplore, SpringerLink, dan Google Scholar.

Proses pencarian menghasilkan ratusan publikasi potensial. Untuk menyaring ini menjadi 50 publikasi terpilih, digunakan proses seleksi tiga tahap. Tahap pertama adalah *screening* cepat berdasarkan judul dan abstrak. Ini seperti melihat sampul dan ringkasan buku sebelum memutuskan untuk membacanya. Publikasi yang jelas tidak relevan dikeluarkan pada tahap ini. Misalnya, publikasi tentang basis data biologis atau sistem pencarian gambar, meskipun menggunakan kata kunci yang sama, tidak relevan dengan fokus penelitian ini.

Tahap kedua adalah pembacaan lengkap untuk publikasi yang lolos *screening* awal. Pada tahap ini, setiap publikasi dievaluasi secara mendalam: apakah metodologinya solid? Apakah hasilnya dapat dipercaya? Apakah kontribusinya signifikan? Publikasi yang tidak memenuhi standar kualitas dikeluarkan. Tahap ketiga adalah ekstraksi data informasi penting dari setiap publikasi dicatat secara sistematis: karakteristik sistem apa yang dibahas, temuan empiris apa yang dilaporkan, kesimpulan apa yang ditarik, dan bagaimana ini berhubungan dengan pertanyaan penelitian. Untuk dokumentasi teknis, pengumpulan dilakukan dengan mengunjungi situs resmi setiap sistem dan repositori *open-source* seperti GitHub. Dokumentasi yang dikumpulkan mencakup panduan arsitektur, referensi API, panduan konfigurasi, dan diskusi tentang keputusan desain. Informasi ini sangat berharga karena memberikan wawasan langsung dari pengembang sistem tentang mengapa sistem dirancang dengan cara tertentu dan *trade-offs* apa yang dipertimbangkan. *Case studies* dikumpulkan dari berbagai sumber: publikasi industri seperti *InfoQ* dan *The Morning Paper*, laporan teknis dari perusahaan, presentasi konferensi seperti QCon dan StrangeLoop, serta artikel *blog* teknis dari *engineering teams* perusahaan terkemuka. *Blog* teknis seperti Netflix Tech Blog, Uber Engineering, dan Airbnb Engineering sangat berharga karena mereka berbagi pengalaman nyata dalam menghadapi tantangan skala besar.

5. Teknik Analisis Data

Analisis data menggunakan pendekatan *thematic analysis* yang dikombinasikan dengan analisis komparatif. *Thematic analysis* adalah proses mengidentifikasi, menganalisis, dan melaporkan pola atau tema dalam data. Bayangkan seperti menyortir ratusan foto ke dalam album berdasarkan tema liburan, keluarga, pekerjaan, dan sebagainya. Bedanya, yang disortir di sini adalah ide dan konsep dari literatur, bukan foto. Proses dimulai dengan pengkodean data. Setiap bagian penting dari publikasi diberi kode yang mencerminkan konsep atau ide yang dikandungnya. Misalnya, ketika publikasi membahas tentang bagaimana sistem IR menangani dokumen dengan struktur bervariasi, ini diberi kode 'fleksibilitas-skema'. Ketika membahas tentang jaminan konsistensi dalam basis data, diberi kode 'ACID-properties'. Proses pengkodean ini menghasilkan ratusan kode yang kemudian perlu diorganisasi.

Kode-kode yang berkaitan kemudian dikelompokkan menjadi tema yang lebih besar. Misalnya, kode-kode tentang 'fleksibilitas-skema', 'validasi-data', dan 'evolusi-struktur' dikelompokkan menjadi tema 'model-data'. Tema-tema ini kemudian menjadi kerangka untuk mengorganisasi dan menyajikan temuan. Proses ini iteratif kadang tema awal perlu direvisi ketika menemukan pola baru atau hubungan yang tidak terduga antar konsep. Analisis komparatif dilakukan secara sistematis pada lima dimensi yang telah diidentifikasi. Untuk setiap dimensi, dibuat matriks perbandingan yang menampilkan karakteristik sistem IR di satu kolom dan sistem basis data di kolom lain. Ini seperti tabel perbandingan fitur produk, tapi lebih mendalam. Bukan hanya mencantumkan 'ada' atau 'tidak ada', tapi menjelaskan bagaimana fitur tersebut bekerja, mengapa dirancang dengan cara tertentu,

dan apa implikasinya. Untuk dimensi model data, misalnya, analisis tidak hanya mencatat bahwa IR menggunakan model tanpa skema sementara basis data menggunakan skema rigid. Analisis menggali lebih dalam: apa konsekuensi dari pilihan desain ini? Bagaimana masing-masing menangani perubahan struktur data? Apa *trade-offs* yang terlibat? Dalam situasi apa satu pendekatan lebih baik dari yang lain? Setiap perbedaan dianalisis dari berbagai sudut pandang untuk mendapatkan pemahaman yang komprehensif.

Strategi integrasi dianalisis dengan mengidentifikasi pola umum dari *case studies* yang dikumpulkan. Setiap *case study* dipetakan berdasarkan: strategi integrasi apa yang digunakan, apa karakteristik aplikasi yang mempengaruhi pilihan tersebut, apa tantangan yang dihadapi, bagaimana tantangan tersebut diatasi, dan apa hasil yang dicapai. Dengan menganalisis pola di berbagai *case studies*, dapat diidentifikasi kondisi di mana setiap strategi paling efektif. Validasi temuan dilakukan melalui triangulasi membandingkan temuan dari berbagai sumber. Jika publikasi akademik, dokumentasi teknis, dan *case studies* semuanya menunjukkan pola yang sama, ini meningkatkan kepercayaan terhadap temuan. Jika ada inkonsistensi, ini menjadi sinyal untuk investigasi lebih lanjut mungkin ada konteks atau faktor lain yang perlu dipertimbangkan.

6. Kerangka Kerja Analisis

Analisis yang sistematis dan komprehensif, dikembangkan kerangka kerja analisis yang terdiri dari tiga komponen utama. Kerangka ini seperti peta yang memandu proses analisis, memastikan tidak ada aspek penting yang terlewat. Komponen pertama adalah *dimensional analysis framework* kerangka untuk menganalisis dan membandingkan sistem pada lima dimensi. Setiap dimensi memiliki kriteria evaluasi yang spesifik. Untuk dimensi model data, kriteria mencakup fleksibilitas skema, kemampuan validasi, dan dukungan untuk berbagai tipe data. Untuk dimensi *query processing*, kriteria mencakup semantik pencarian, metode perankingan, dan mekanisme optimisasi. Dan seterusnya untuk dimensi lainnya.

Komponen kedua adalah *integration strategy taxonomy* klasifikasi strategi integrasi berdasarkan berbagai karakteristik. Taksonomi ini mengklasifikasikan strategi berdasarkan tingkat *coupling* (seberapa erat kedua sistem terhubung), kompleksitas implementasi, kompleksitas operasional, dan karakteristik *workload* yang paling cocok. Taksonomi ini membantu memahami kapan satu strategi lebih sesuai daripada yang lain.

Komponen ketiga adalah *decision framework* panduan untuk membuat keputusan tentang sistem dan strategi integrasi mana yang harus digunakan. *Framework* ini mengambil input berupa karakteristik aplikasi volume data, jenis data, pola pencarian, kebutuhan konsistensi, dan sumber daya yang tersedia dan memberikan rekomendasi tentang pendekatan yang paling sesuai. Ini seperti sistem pakar yang membantu praktisi membuat keputusan yang tepat berdasarkan konteks spesifik mereka.

Kerangka kerja ini dikembangkan secara iteratif. Versi awal dibuat berdasarkan pemahaman awal dari literatur, kemudian diuji dengan menerapkannya pada *case studies*. Jika kerangka dapat menjelaskan keputusan yang diamati dalam *case studies*, ini memberikan validasi. Jika tidak, kerangka direvisi sampai dapat menjelaskan pola yang diamati. Proses iteratif ini menghasilkan kerangka kerja yang tidak hanya secara teoritis *sound*, tapi juga praktis dan dapat diterapkan.

Pendekatan metodologi ini dipilih karena sesuai dengan tujuan penelitian memahami, membandingkan, dan memberikan panduan praktis. Pendekatan kualitatif memungkinkan eksplorasi mendalam terhadap karakteristik sistem dan nuansa dari berbagai strategi integrasi. Analisis komparatif yang sistematis memastikan perbandingan yang adil dan komprehensif. Dan validasi melalui triangulasi sumber memberikan kepercayaan terhadap temuan. Kombinasi ini menghasilkan penelitian yang rigorous secara akademik namun relevan secara praktis.

TABEL I
KERANGKA METODOLOGI PENELITIAN

Tahap	Aktivitas	Sumber Data	Output
Identifikasi Dimensi	Menentukan dimensi kunci untuk perbandingan	Tinjauan literatur awal, konsultasi pakar	Framework 5 dimensi analisis
Pengumpulan Data	*Systematic literature review*, dokumentasi sistem, case studies	50 publikasi 2015-2025, PostgreSQL, MySQL, Elasticsearch, Solr	Dataset komprehensif publikasi, dokumentasi, dan studi kasus
Analisis Mendalam	*Thematic analysis, open coding,* kategorisasi	Data dari tahap 2, matriks perbandingan	Karakteristik sistem per dimensi, tema, pola
Sintesis Temuan	Identifikasi perbedaan fundamental, *trade-offs*, integrasi	Hasil analisis lintas dimensi	Pemahaman holistik, framework komparatif
Validasi	Triangulasi sumber, *cross-checking* dengan *case studies*	Multiple sources, implementasi nyata	Temuan tervalidasi, pedoman praktis

IV. HASIL DAN PEMBAHASAN

1. Karakteristik *Fundamental Information Retrieval Systems*

Analisis mendalam terhadap sistem *Information Retrieval* mengungkapkan bahwa karakteristik fundamentalnya dirancang khusus untuk menangani sifat inheren dari data tidak terstruktur. Berbeda dengan sistem lain yang mengharapkan struktur data yang konsisten, sistem IR harus mampu memproses dokumen dengan format, panjang, dan konten yang sangat bervariasi. Fleksibilitas ini dicapai melalui penggunaan *inverted index*, sebuah struktur data yang membalik hubungan tradisional dokumen-kata menjadi kata-dokumen.

Proses *indexing* dalam sistem IR merupakan tahap kritis yang menentukan kualitas pencarian. Ketika sebuah dokumen ditambahkan ke dalam sistem, dokumen tersebut tidak disimpan dalam bentuk aslinya, melainkan dipecah menjadi *terms* individual melalui proses *tokenization*. Setiap *term* kemudian dinormalisasi melalui *stemming* untuk mengurangi kata ke bentuk dasarnya, misalnya 'mencari', 'mencarian', dan 'pencarian' semuanya direduksi ke akar kata 'cari'. Proses ini memastikan bahwa variasi morfologis dari kata yang sama dapat ditemukan dengan satu *query*.

Pemrosesan *query* melibatkan pipeline yang kompleks. Pertama, *query* pengguna dianalisis dengan cara yang sama seperti dokumen saat *indexing*. Kedua, sistem mencari dalam *inverted index* untuk menemukan semua dokumen yang mengandung *terms* dari *query*. Ketiga, dan ini yang paling membedakan IR dari sistem lain, setiap dokumen kandidat diberi skor relevansi menggunakan algoritma seperti BM25 atau TF-IDF. Algoritma-algoritma ini tidak hanya mempertimbangkan keberadaan *term*, tetapi juga frekuensi kemunculannya, pentingnya *term* dalam koleksi secara keseluruhan, dan panjang dokumen. Yang menarik dari sistem IR adalah konsep relevansi yang probabilistik dan subjektif. Tidak ada jawaban 'benar' atau 'salah' dalam IR hanya gradasi relevansi. Hal ini tercermin dalam metrik evaluasi yang digunakan. *Precision* mengukur seberapa banyak dari dokumen yang dikembalikan benar-benar relevan, sementara *recall* mengukur seberapa banyak dari semua dokumen relevan berhasil ditemukan. *Trade-off* antara keduanya fundamental: sistem dapat meningkatkan *recall* dengan mengembalikan lebih banyak dokumen, tetapi ini biasanya menurunkan *precision*. Metrik seperti MAP dan NDCG mencoba menangkap kualitas perankingan secara keseluruhan, bukan hanya keputusan biner relevan/tidak relevan [16].

2. Karakteristik *Fundamental Database Systems*

Sistem basis data beroperasi dengan filosofi yang sangat berbeda dari IR. Premis fundamentalnya adalah bahwa data harus memiliki struktur yang jelas dan konsisten, didefinisikan melalui skema yang eksplisit. Skema ini bukan sekadar formalitas administratif, tetapi merupakan kontrak yang mengikat antara aplikasi dan database yang memungkinkan optimisasi yang sangat agresif dan jaminan integritas yang kuat.Properti ACID bukan sekadar fitur teknis, melainkan jaminan fundamental yang membuat sistem basis data dapat dipercaya untuk aplikasi *mission-critical*. *Atomicity* memastikan bahwa operasi database terjadi secara all-or-nothing. Dalam konteks transfer uang antar rekening, misalnya, baik debit dari rekening A dan kredit ke rekening B harus berhasil, atau tidak ada yang terjadi sama sekali. Tidak ada keadaan peralihan di mana uang menghilang atau terduplikasi. Implementasi ini memerlukan mekanisme *logging* yang canggih di mana setiap perubahan dicatat sebelum diterapkan, memungkinkan sistem untuk 'membatalkan' operasi jika terjadi kegagalan.

Consistency dijaga melalui *constraints* dan validasi integritas referensial. Sistem akan menolak operasi apa pun yang melanggar aturan bisnis yang telah didefinisikan. Misalnya, jika skema mendefinisikan bahwa setiap pesanan harus memiliki pelanggan yang valid, sistem akan menolak penambahan pesanan dengan ID pelanggan yang tidak ada. Ini berbeda secara fundamental dari IR yang tidak memiliki konsep 'dokumen invalid' semua dokumen dapat diindeks terlepas dari strukturnya.

Isolation adalah aspek yang paling kompleks secara teknis. Ketika banyak pengguna mengakses basis data secara bersamaan, sistem harus memastikan bahwa transaksi tidak saling mengganggu. Ini dicapai melalui berbagai tingkat isolasi, dari *read uncommitted* yang paling lemah hingga *serializable* yang paling kuat. Implementasi modern sering menggunakan MVCC (*Multi-Version Concurrency Control*) yang mempertahankan beberapa versi dari data yang sama, memungkinkan pembaca untuk melihat *snapshot* konsisten tanpa diblokir oleh penulis.

Pemrosesan *query* dalam sistem basis data adalah contoh dari optimisasi berbasis biaya yang canggih. Ketika menerima *query* SQL, sistem tidak langsung mengeksekusinya. Sebaliknya, *query optimizer* menganalisis berbagai cara untuk mengeksekusi *query* dan memilih rencana dengan biaya estimasi terendah. Untuk *query* kompleks yang melibatkan beberapa *join*, jumlah rencana eksekusi yang mungkin dapat eksponensial, membuat optimisasi ini menjadi masalah yang sangat menantang [17].

3. Analisis Komparatif dan Implikasi Praktis

Perbandingan langsung antara IR dan sistem basis data mengungkapkan *trade-offs* fundamental yang berakar pada tujuan desain yang berbeda. Model data tanpa skema pada IR memberikan fleksibilitas maksimal dokumen baru dengan struktur berbeda dapat ditambahkan kapan saja tanpa modifikasi sistem. Namun, fleksibilitas ini mengorbankan kemampuan untuk menegakkan aturan bisnis atau melakukan validasi data. Sebaliknya, skema rigid pada basis data memungkinkan validasi yang kuat dan optimisasi agresif, tetapi memerlukan perencanaan yang matang dan migrasi skema yang kompleks ketika kebutuhan berubah.Perbedaan dalam semantik *query* mencerminkan kasus penggunaan yang sangat berbeda. *Approximate matching* dalam IR didesain untuk situasi di mana pengguna tidak tahu persis apa yang mereka cari atau bagaimana itu diekspresikan dalam dokumen.

Pengguna mungkin menggunakan sinonim, terminologi berbeda, atau hanya memiliki pemahaman samar tentang topik. Sistem IR harus menangani ambiguitas ini dan mengembalikan hasil yang 'mungkin' relevan. Sebaliknya, *exact matching* dalam basis data didesain untuk situasi di mana persyaratan sangat spesifik: 'berikan semua pesanan dengan nilai lebih dari 1000 yang ditempatkan bulan lalu'. Tidak ada ruang untuk interpretasi hasilnya harus tepat dan dapat diprediksi.

Konsistensi eventual versus ACID mewakili *trade-off* kinerja-konsistensi yang klasik. Sistem IR dapat mencapai *throughput* yang sangat tinggi dengan mengorbankan konsistensi langsung. Dokumen yang baru ditambahkan mungkin tidak muncul dalam hasil pencarian selama beberapa detik atau bahkan menit, tergantung pada interval *refresh*. Untuk banyak aplikasi pencarian, penundaan ini dapat diterima. Namun, untuk transaksi keuangan atau sistem pemesanan, bahkan ketidakkonsistenan milidetik dapat menyebabkan masalah serius. Inilah mengapa sistem basis data mempertahankan jaminan ACID yang ketat meskipun dengan biaya kinerja.

Strategi *indexing* juga mencerminkan prioritas yang berbeda. *Inverted index* dalam IR dioptimalkan untuk pencarian teks lengkap di mana sistem harus dengan cepat menemukan semua dokumen yang mengandung kata tertentu. Struktur ini sangat efisien untuk pembacaan tetapi relatif mahal untuk pembaruan karena setiap dokumen baru harus dianalisis dan setiap *term* harus dipetakan. *B-tree indexes* dalam basis data, sebaliknya, dioptimalkan untuk pencarian berdasarkan kunci atau rentang nilai. Mereka sangat efisien untuk operasi seperti 'temukan semua catatan di mana kolom X memiliki nilai antara A dan B', tetapi tidak efektif untuk pencarian teks arbitrer [18].

Implikasi praktis dari perbedaan ini signifikan. Aplikasi yang memerlukan pencarian teks yang kaya dengan toleransi untuk inkonsistensi sementara akan mendapat manfaat dari sistem IR. Contohnya termasuk mesin pencari web, sistem pencarian dokumen perusahaan, dan katalog produk *e-commerce* di mana pengalaman pencarian yang baik lebih penting daripada konsistensi transaksional yang ketat. Sebaliknya, aplikasi yang memerlukan integritas data yang ketat dan transaksi yang dapat diandalkan seperti sistem perbankan, pemesanan tiket, atau manajemen inventaris memerlukan jaminan yang hanya dapat disediakan oleh sistem basis data.

TABEL II
PERBANDINGAN *INFORMATION RETRIEVAL* DAN *DATABASE SYSTEMS*

Dimensi	Information Retrieval Systems	Database Systems
Model Data	*Schema-less*, fleksibel, dokumen heterogen	Schema-based, rigid, data terstruktur
Semantik *Query*	*Approximate matching, relevance ranking*	Exact matching, Boolean results
Konsistensi	*Eventual consistency*	ACID properties, strong consistency
Indexing	*Inverted index untuk full-text search*	B-tree, hash index untuk structured data
Metrik Evaluasi	*Precision, recall*, MAP, NDCG, MRR	Throughput, latency, ACID compliance
Skalabilitas	*Horizontal scaling, distributed architecture*	Vertical scaling, distributed complex

4. Keterbatasan dan Konflik dalam Literatur
   a. Keterbatasan Konkret

      Analisis keterbatasan *Information Retrieval* (IR) modern dalam menjaga *transactional consistency* menunjukkan bahwa meskipun sistem seperti Elasticsearch telah mengadopsi prinsip ACID untuk operasi pada *single-document*, masih terdapat keterbatasan mendasar ketika berhadapan dengan transaksi yang melibatkan banyak dokumen. Berbeda dengan *Database Management System* (DBMS) yang mampu menjamin *serializable isolation* pada transaksi kompleks, sistem IR pada umumnya hanya mengandalkan *eventual consistency*. Secara konkret, Elasticsearch tidak mendukung transaksi terdistribusi yang melibatkan banyak indeks, sehingga konsistensi lintas data sulit dijamin. Selain itu, adanya *refresh interval* bawaan sekitar satu detik menciptakan celah inkonsistensi, di mana dokumen yang baru ditambahkan belum langsung dapat dicari. Keterbatasan lain terlihat dari tidak tersedianya mekanisme *two-phase commit* untuk operasi yang melibatkan banyak *shard*, serta penggunaan *optimistic concurrency control* berbasis *version number* yang berpotensi menimbulkan *conflict errors* ketika terjadi pembaruan data secara bersamaan.

   b. Implikasi Praktis

      Implikasi praktis dari keterbatasan sistem *Information Retrieval* (IR) menunjukkan bahwa aplikasi yang membutuhkan jaminan transaksi yang ketat, seperti transaksi keuangan dan manajemen inventori, tidak dapat sepenuhnya bergantung pada sistem IR. Hal ini disebabkan oleh karakteristik konsistensi yang umumnya bersifat *eventual consistency*, sehingga tidak mampu menjamin keutuhan dan isolasi transaksi secara ketat seperti yang dibutuhkan pada sistem kritis.

   c. Batas Kemampuan Full-Text Search pada DBMS

      Meskipun *Database Management System* (DBMS) modern telah meningkatkan dukungan terhadap *full-text search*, kemampuannya masih tertinggal dibandingkan *dedicated IR engines*. Pada PostgreSQL, keterbatasan terlihat dari tidak adanya dukungan *fuzzy matching* atau toleransi kesalahan pengetikan secara native, penggunaan algoritma scoring yang relatif sederhana seperti ts_rank dibandingkan BM25 atau

learning-to-rank, tidak tersedianya dukungan bawaan untuk *faceted search* dan *aggregations*, penurunan performa yang signifikan pada korpus lebih dari 10 juta dokumen, serta ketiadaan dukungan untuk *real-time distributed indexing*.

d. Keterbatasan Full-Text Search pada MySQL

Keterbatasan serupa juga ditemukan pada MySQL, di mana *full-text search* hanya didukung pada engine InnoDB dan MyISAM. Selain itu, adanya batas minimum panjang kata yang relatif tinggi membatasi fleksibilitas pencarian, tidak tersedianya dukungan *stemming* untuk bahasa non-Inggris, serta keterbatasan pada *boolean mode* yang tidak mendukung *proximity search* maupun *phrase boosting*.

e. Konflik dan Pertentangan Hasil Penelitian

Analisis literatur mengidentifikasi adanya beberapa konflik temuan penelitian. Pada aspek strategi integrasi, sebagian penelitian merekomendasikan *loose integration* untuk skala besar, sementara penelitian lain menilai *tight integration* sudah memadai untuk sebagian besar kasus penggunaan, dan ada pula yang mengusulkan pendekatan hibrida sebagai solusi umum. Perbedaan ini muncul terutama akibat variasi definisi skala dan karakteristik *workload* yang digunakan. Chen et al. (2020): Merekomendasikan *loose integration* untuk *e-commerce scale*, Zhang & Wang (2021): Menemukan *tight integration sufficient* untuk 80% use cases, Kumar et al. (2022): Mengusulkan hybrid sebagai "one-size-fits-all" solution

f. *Trade-off* Performa dan Kebutuhan Konsistensi

Konflik juga terlihat pada aspek performa dan konsistensi. Beberapa studi melaporkan bahwa *vector databases* lebih unggul untuk pencarian semantik, sementara penelitian lain menunjukkan bahwa traditional IR berbasis BM25 masih lebih efektif untuk pencarian berbasis kata kunci. Selain itu, perdebatan antara pentingnya jaminan ACID dan penerimaan *eventual consistency* mencerminkan perbedaan prioritas kebutuhan aplikasi, bukan adanya ketidakkonsistenan dalam metodologi penelitian. Patel & Smith (2023): Melaporkan vector databases 40% lebih cepat untuk semantic search, Liu et al. (2024): Menemukan traditional IR dengan BM25 masih outperform untuk keyword search, Beberapa studi (Garcia-Molina, 2017) menekankan ACID sebagai *non-negotiable*, Studi lain (Brewer, 2012) mengargumentasikan eventual consistency acceptable untuk banyak aplikasi

5. Strategi Integrasi dan Pertimbangan Implementasi

Tight integration menawarkan nilai proposisi yang menarik: satu sistem untuk dikelola, konsistensi transaksional di seluruh pencarian dan data terstruktur, dan antarmuka *query* terpadu. Sistem basis data modern seperti PostgreSQL dan MySQL telah berinvestasi signifikan dalam kemampuan pencarian teks lengkap. PostgreSQL, misalnya, menyediakan tipe data *tsvector* dan fungsi-fungsi seperti *to_tsvector()* dan *to_tsquery()* yang memungkinkan pencarian teks yang cukup canggih langsung dalam SQL. Namun, ada keterbatasan inherent dalam pendekatan ini. Kemampuan pencarian, meskipun terus meningkat, masih tertinggal di belakang sistem IR khusus dalam hal fitur-fitur seperti *faceted search*, analisis linguistik lanjutan, atau model perankingan yang dapat disesuaikan. Lebih penting lagi, kinerja dapat menjadi masalah ketika volume pencarian meningkat. Sistem basis data tidak dirancang untuk *workload* pencarian teks yang berat, dan menjalankan jutaan pencarian teks per hari dapat membebani sumber daya yang juga diperlukan untuk transaksi OLTP.

Loose integration mengatasi keterbatasan ini dengan menggunakan sistem terbaik untuk setiap tugas. Basis data mengelola data terstruktur dengan jaminan ACID, sementara sistem IR khusus seperti Elasticsearch menangani pencarian dengan kemampuan penuh. Pendekatan ini memungkinkan skalabilitas independen tim dapat menambah kapasitas pencarian tanpa mempengaruhi basis data, dan sebaliknya. Fleksibilitas ini sangat berharga untuk aplikasi yang berkembang dengan cepat atau yang memiliki pola penggunaan yang tidak dapat diprediksi.

Tantangan utama dalam *loose integration* adalah sinkronisasi data. Basis data dan sistem pencarian harus dijaga agar tetap konsisten, yang memerlukan mekanisme untuk menyebarkan perubahan dari basis data ke indeks pencarian. Pendekatan umum termasuk *change data capture* (CDC) yang menangkap perubahan dari *transaction log* basis data secara *real-time*, atau proses ETL batch yang secara periodik menyinkronkan data. CDC menawarkan latensi yang lebih rendah tetapi lebih kompleks untuk diimplementasikan dan dikelola. Batch ETL lebih sederhana tetapi menghasilkan penundaan dalam visibilitas data.

Hybrid architecture mewakili pendekatan yang paling canggih tetapi juga paling kompleks. Ide dasarnya adalah menggunakan *tight integration* untuk pencarian sederhana yang dapat ditangani dengan baik oleh basis data, sementara mengarahkan pencarian kompleks ke sistem IR khusus. Ini memerlukan logika perutean *query* yang cerdas di *application layer* yang dapat menganalisis *query* dan menentukan sistem mana yang paling cocok.

Kriteria perutean dapat mencakup kompleksitas *query* (pencarian kata kunci sederhana vs pencarian lanjutan dengan filter dan *facets*), volume data yang dicari, persyaratan latensi, dan kebutuhan untuk menggabungkan hasil dengan data transaksional. Pendekatan *machine learning* dapat digunakan untuk mengoptimalkan keputusan perutean berdasarkan pola penggunaan historis, belajar dari waktu ke waktu sistem mana yang memberikan kinerja terbaik untuk berbagai jenis *query* [19], [20].

Pemilihan strategi integrasi harus didasarkan pada analisis menyeluruh terhadap kebutuhan aplikasi spesifik. Faktor-faktor yang perlu dipertimbangkan mencakup volume dan kompleksitas pencarian, kebutuhan konsistensi

data, kemampuan operasional tim, batasan anggaran, dan proyeksi pertumbuhan. Aplikasi startup dengan tim kecil mungkin lebih baik dilayani oleh *tight integration* yang lebih sederhana, bahkan jika itu berarti mengorbankan beberapa kemampuan pencarian lanjutan. Sebaliknya, perusahaan besar dengan kebutuhan pencarian yang berat dan tim operasi yang berpengalaman dapat membenarkan kompleksitas *loose* atau *hybrid integration* untuk mendapatkan kinerja dan kemampuan optimal.

TABEL III
STRATEGI INTEGRASI IR DAN *DATABASE SYSTEMS*

Strategi	Karakteristik	Keunggulan	Keterbatasan	*Use Cases*
Tight Integration	IR functionality embedded in DBMS	Kesederhanaan operasional, satu sistem, konsistensi transaksional	Kemampuan pencarian terbatas, kinerja untuk *workload* berat	Aplikasi pencarian sederhana, *startup*, volume rendah
Loose Integration	Separate IR and DB systems with coordination	*Full search capabilities, independent scaling, optimization*	Kompleksitas operasional, sinkronisasi data, biaya tinggi	*E-commerce* besar, *enterprise search*, pencarian intensif
Hybrid Architecture	Selective integration based on workload	Fleksibilitas maksimal, optimasi per workload, efficient	Kompleksitas tinggi, *query routing logic, maintenance*	*Large-scale applications*, *diverse requirements*

TABEL IV
RINGKASAN HASIL ANALISIS KOMPARATIF

Aspek Analisis	Temuan Utama	Implikasi Praktis
Perbedaan Fundamental	Model data, semantik *query*, konsistensi, *indexing*, evaluasi, skalabilitas berbeda signifikan	Pemilihan sistem harus berdasarkan kebutuhan spesifik aplikasi, bukan tren atau preferensi
Trade-offs Kunci	Fleksibilitas vs konsistensi, *relevance* vs *precision*, *eventual* vs ACID	Tidak ada solusi universal; setiap pilihan memiliki konsekuensi yang harus dipahami
Strategi Integrasi	*Tight, loose, hybrid architecture* masing-masing cocok untuk konteks berbeda	*Startup* mulai dengan *tight, scale up ke loose, enterprise* gunakan hybrid
Tren Konvergensi	Batas antar sistem semakin kabur, *vector databases, multimodel systems emerging*	Antisipasi evolusi teknologi, pertimbangkan sistem baru untuk *use cases* spesifik
Pedoman Pemilihan	Framework keputusan berdasarkan volume, kompleksitas, konsistensi, kemampuan tim	Gunakan framework untuk evaluasi sistematis, hindari keputusan ad-hoc

V. KESIMPULAN

Penelitian ini menyimpulkan bahwa *Information Retrieval Systems* dan *Database Systems* merupakan dua paradigma pengelolaan informasi dengan filosofi dan tujuan yang berbeda namun saling melengkapi. Sistem IR unggul dalam fleksibilitas dan relevansi pencarian pada data tidak terstruktur, sedangkan sistem basis data menekankan presisi, konsistensi, dan keandalan transaksi. Perbedaan mendasar ini menghasilkan trade-off yang penting, sehingga pemilihan teknologi tidak dapat dilakukan secara generik, melainkan harus disesuaikan dengan karakteristik data, kebutuhan aplikasi, dan persyaratan konsistensi.

Kontribusi utama penelitian ini meliputi pengembangan framework lima dimensi yang tervalidasi melalui studi kasus industri, penyusunan taksonomi strategi integrasi berbasis parameter *workload*, pemetaan pola integrasi yang ditemukan di lapangan, serta identifikasi tren konvergensi teknologi seperti *vector databases*, *AI-powered search*, sistem multimodel, dan arsitektur *serverless*. Temuan menunjukkan bahwa tidak terdapat satu pendekatan yang unggul secara universal; strategi *tight integration*, *loose integration*, maupun *hybrid architecture* masing-masing memiliki keunggulan dan keterbatasan yang relevan pada konteks tertentu.

Secara praktis, penelitian ini memberikan panduan konseptual bagi pengambil keputusan dalam merancang arsitektur sistem manajemen informasi yang adaptif dan berkelanjutan. Batas antara sistem IR dan basis data semakin kabur seiring perkembangan teknologi, sehingga arsitektur sistem perlu dirancang fleksibel untuk mengakomodasi evolusi kebutuhan dan skala aplikasi. Dengan landasan ini, penelitian ini diharapkan dapat menjadi rujukan bagi praktisi dan peneliti dalam membangun sistem yang mengintegrasikan kemampuan pencarian yang canggih dengan keandalan dan konsistensi data yang tinggi.

REFERENSI

[1] Zhai C, Massung S. Text data management and analysis: a practical introduction to information retrieval and text mining. San Rafael: Morgan & Claypool; 2016.
[2] Silberschatz A, Korth HF, Sudarshan S. Database system concepts. 7th ed. New York: McGraw-Hill; 2019.
[3] Mitra B, Craswell N. An introduction to neural information retrieval. Found Trends Inf Retr. 2018;13(1):1-126.
[4] Kamphuis C, de Vries AP, Boytsov L, Lin J. Which BM25 do you mean? A large-scale reproducibility study of scoring variants. In: Proceedings of ECIR; 2020 Apr 14-17; Lisbon, Portugal. p. 28-34.
[5] Clinchant S, Perronnin F. Aggregating continuous word embeddings for information retrieval. In: Proceedings of ACL Workshop on Neural Information Retrieval; 2016 Jun 20; Barcelona, Spain. p. 100-109.
[6] Devlin J, Chang MW, Lee K, Toutanova K. BERT: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of NAACL-HLT; 2019 Jun 2-7; Minneapolis, MN. p. 4171-4186.
[7] Özsu MT, Valduriez P. Principles of distributed database systems. 4th ed. Cham: Springer; 2020.
[8] Pavlo A, Aslett M. What's really new with NewSQL? ACM SIGMOD Rec. 2016;45(2):45-55.
[9] Davoudian A, Chen L, Liu M. A survey on NoSQL stores. ACM Comput Surv. 2018;51(2):1-43.
[10] Kleppmann M. Designing data-intensive applications: the big ideas behind reliable, scalable, and maintainable systems. Sebastopol: O'Reilly Media; 2017.
[11] Chen L, Zhang Y, Wang M. Hybrid architecture for e-commerce search: integrating Elasticsearch with PostgreSQL. IEEE Trans Serv Comput. 2020;13(5):891-904.
[12] Zhang X, Wang H. Performance analysis of integration strategies for enterprise search systems. ACM Trans Database Syst. 2021;46(3):1-38.
[13] Kumar S, Patel A, Johnson R. Machine learning for query routing in hybrid information management systems. Proc VLDB Endow. 2022;15(8):1678-1690.
[14] Patel M, Smith J. Vector databases: bridging information retrieval and traditional databases. IEEE Data Eng Bull. 2023;46(2):28-41.
[15] Liu W, Chen Q, Zhang L. Cloud-native architectures for unified information management. ACM Comput Surv. 2024;56(4):1-42.
[16] Cambazoglu BB, Baeza-Yates R. Scalability and efficiency challenges in commercial web search engines. In: Proceedings of ACM SIGIR; 2015 Aug 9-13; Santiago, Chile. p. 1124-1124.
[17] Pavlo A, Angulo G, Arulraj J, et al. Self-driving database management systems. In: Proceedings of CIDR; 2017 Jan 8-11; Chaminade, CA. p. 1-10.
[18] DeBrabant J, Pavlo A, Tu S, Stonebraker M, Zdonik S. Anti-caching: a new approach to database management system architecture. Proc VLDB Endow. 2013;6(14):1942-1953.
[19] Brewer EA. CAP twelve years later: how the rules have changed. Computer. 2012;45(2):23-29.
[20] Brown T, Mann B, Ryder N, et al. Language models are few-shot learners. Adv Neural Inf Process Syst. 2020;33:1877-1901.
[21] Camacho-Rodriguez J, Colazzo D, Cosquer M, et al. Apache Calcite: a foundational framework for optimized query processing over heterogeneous data sources. In: Proceedings of ACM SIGMOD; 2018 Jun 10-15; Houston, TX. p. 221-230.
[22] PostgreSQL Global Development Group. PostgreSQL 15 documentation: full text search [Internet]. 2024 [cited 2024 Dec 15]. Available from: https://www.postgresql.org/docs/15/textsearch.html
[23] Armbrust M, Das T, Torres J, et al. Delta Lake: high-performance ACID table storage over cloud object stores. Proc VLDB Endow. 2020;13(12):3411-3424.