BAB II

LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu digunakan sebagai landasan untuk memahami perkembangan penelitian yang relevan dengan topik yang diangkat. Pada bagian ini dibahas sejumlah penelitian mengenai perancangan dan pengelolaan basis data, pemanfaatan PostgreSQL sebagai basis data yang mendukung pengelolaan data vektor, penerapan *hybrid retrieval* yang mengintegrasikan *vector search* dengan data terstruktur, perancangan basis data pada domain layanan kesehatan, serta penerapan metode *Database Life Cycle* (DBLC) dalam menghasilkan rancangan basis data yang sistematis. Ringkasan penelitian terdahulu yang relevan disajikan pada **Tabel 2.1** sebagai dasar dalam mengidentifikasi *research gap* serta menentukan posisi dan kontribusi penelitian ini.

Tabel 2.1 Penelitian Terdahulu

Ref	Judul	Penulis	Tahun	Masalah	Metode	Hasil Penelitian
[17]	Are There Fundamental Limitations in Supporting Vector Data Management in Relational Databases? A Case Study of	Yunan Zhang, Shige Liu, Jianguo Wang	2024	Database relasional mulai dimanfaatkan untuk menyimpan data vektor, namun belum diketahui apakah arsitektur PostgreSQL mampu mendukung manajemen data vektor secara efisien untuk semantic retrieval.	Studi kasus PostgreSQL dengan evaluasi vector data management dan vector retrieval.	Penelitian menunjukkan bahwa PostgreSQL dapat digunakan sebagai basis data vektor, namun masih memiliki bottleneck pada proses indexing, query execution, dan pengelolaan data berdimensi tinggi. Penelitian juga mengidentifikasi beberapa keterbatasan arsitektur PostgreSQL serta memberikan rekomendasi untuk meningkatkan performa vector retrieval pada sistem basis data relasional.

Ref	Judul	Penulis	Tahun	Masalah	Metode	Hasil Penelitian
	PostgreSQL					
[18]	ACORN: Performant and Predicate-Agnostic Search Over Vector Embeddings and Structured Data	Liana Patel, Peter Kraft, Carlos Guestrin, Matei Zaharia	2024	Hybrid search yang menggabungkan vector similarity dengan structured filtering masih memiliki performa yang rendah atau hanya mendukung jenis filter tertentu.	Pendekatan predicate-agnostic hybrid search berbasis indeks HNSW.	Penelitian memperkenalkan ACORN sebagai metode hybrid search yang mampu menggabungkan vector retrieval dan structured filtering tanpa bergantung pada jenis predicate tertentu. Hasil evaluasi menunjukkan peningkatan throughput sebesar 2 hingga 1000 kali dibanding pendekatan sebelumnya dengan tingkat recall yang tetap tinggi.
[19]	Adaptive and Scalable Database Management with Machine Learning Integration: A PostgreSQL Case Study	Maryam Abbasi, Marco V. Bernardo, Paulo Váz, José Silva, dan Pedro Martins	2024	Pengelolaan basis data modern memerlukan mekanisme yang adaptif untuk menangani beban kerja yang dinamis dan mendukung aplikasi berbasis AI.	Integrasi Machine Learning pada PostgreSQL untuk optimasi manajemen basis data.	Penelitian menunjukkan bahwa integrasi machine learning ke dalam PostgreSQL mampu meningkatkan optimasi query, pengelolaan sumber daya, serta efisiensi sistem basis data pada workload yang berubah secara dinamis.
[20]	Building Ontology-Based Temporal Databases for Data Reuse: An Applied	Christina Khnaisser, Vincent Looten, Luc Lavoie,	2024	Data organisasi rumah sakit sering tersimpan dalam berbagai sistem sehingga sulit diintegrasikan dan dimanfaatkan kembali.	Perancangan basis data temporal berbasis ontology pada domain kesehatan.	Penelitian menghasilkan rancangan basis data kesehatan yang mampu mengintegrasikan data organisasi rumah sakit secara terstruktur sehingga meningkatkan interoperabilitas, konsistensi data, dan kemudahan pemanfaatan kembali pada berbagai aplikasi kesehatan.

Ref	Judul	Penulis	Tahun	Masalah	Metode	Hasil Penelitian
	Example on Hospital Organizational Structures	Anita Burgun, dan Jean-François Ethier				
[21]	MedT5SQL: A Transformers-Based Large Language Model for Text-to-SQL Conversion in the Healthcare Domain	Ahmed Khamis, Mohammed El-Hajjar, dan Mahmoud Elish	2024	Pengguna layanan kesehatan mengalami kesulitan dalam mengakses informasi dari basis data karena memerlukan pemahaman bahasa SQL.	Model Transformer (MedT5SQL) untuk konversi Natural Language ke SQL pada basis data kesehatan.	Penelitian menunjukkan bahwa model MedT5SQL mampu menghasilkan query SQL dengan akurasi tinggi pada domain kesehatan sehingga mempermudah akses terhadap data medis tanpa memerlukan pengetahuan SQL secara langsung.
[22]	Desain Model Database Mutasi Siswa Dengan Menerapkan Metode Database Life Cycle	Samidi1 Ramdhani Hidayat	2023	Pengelolaan data mutasi siswa membutuhkan rancangan basis data yang mampu menyimpan dan mengelola informasi secara efektif dan terstruktur.	Database Life Cycle (DBLC).	Penelitian ini bertujuan merancang desain model database mutasi siswa dengan menerapkan metode Database Life Cycle (DBLC). Hasil dari perancangan ini adalah sebuah model rancangan basis data relasional yang memiliki relasi antar tabel, terdiri dari 5 tabel dengan 6 relasi, yaitu tabel siswa, sekolah, mutasi, spmo (Surat Permohonan Mutasi dari Orang Tua), dan spas (Surat Pindah Dari Asal Sekolah). Database ini dirancang untuk memiliki fungsi utama menyimpan serta memelihara data dan mampu menjawab keperluan pengelolaan informasi mutasi siswa agar proses mutasi menjadi lebih mudah, cepat, dan tepat. Implementasi database ini menggunakan MySQL yang diuji dengan insert

Ref	Judul	Penulis	Tahun	Masalah	Metode	Hasil Penelitian
						data menggunakan phpMyAdmin.
[23]	*Relational Database for Health Care*	Wanra Tarigan, Rafika Sari br Sembiring	2023	Pengelolaan layanan kesehatan berbasis file manual menyebabkan kesulitan pencarian data, redundansi, dan kurangnya integrasi data antar bagian.	Database Lifecycle (Conceptual Design, Logical Design, Physical Design)	Penelitian mengenai perancangan Relational Database for Health Care ini menunjukkan bahwa pengimplementasian database dan sistem informasi dapat menjadi solusi atas masalah operasional layanan kesehatan yang masih berbasis file manual, seperti kesulitan pencarian data, inefisiensi dokumentasi, hambatan integrasi data antar bagian, dan redundansi data. Perancangan database dilakukan melalui pendekatan siklus hidup pengembangan sistem database (Database Lifecycle), yang meliputi tahap desain konseptual, desain logikal, dan desain fisikal. Secara keseluruhan, hasil penelitian menyimpulkan bahwa pengimplementasian sistem aplikasi layanan kesehatan dengan database terintegrasi dapat membantu melacak data operasional, menjadikan operasional dan layanan masyarakat lebih efektif dan efisien, serta meminimalkan kesulitan dalam pembuatan laporan yang dibutuhkan manajemen untuk mendukung pengambilan keputusan. Selain itu, keamanan sistem yang diterapkan membuat data lebih terstruktur dan tidak mudah dimasuki pihak yang tidak berwenang.

Berdasarkan penelitian terdahulu yang telah disajikan, dapat disimpulkan bahwa perkembangan penelitian di bidang basis data menunjukkan adanya pergeseran dari pengelolaan data relasional konvensional menuju pengelolaan data yang mampu mendukung pencarian semantik melalui representasi vektor. Penelitian mengenai PostgreSQL sebagai basis data vektor menunjukkan bahwa sistem basis data relasional memiliki potensi untuk mendukung semantic retrieval, meskipun masih terdapat tantangan pada aspek pengelolaan data vektor, pengindeksan, dan efisiensi eksekusi kueri [17]. Selain itu, penelitian mengenai hybrid retrieval memperlihatkan bahwa integrasi antara vector search dan structured filtering mampu meningkatkan kualitas hasil pencarian tanpa menghilangkan kemampuan pemrosesan data relasional [18]. Penelitian lain juga menunjukkan bahwa integrasi teknologi kecerdasan buatan pada sistem manajemen basis data dapat meningkatkan efisiensi pengelolaan data dan optimasi kueri, sehingga mendukung kebutuhan aplikasi modern yang memanfaatkan AI [19].

Pada domain layanan kesehatan, berbagai penelitian telah menunjukkan bahwa perancangan basis data yang terstruktur berperan penting dalam meningkatkan integrasi, konsistensi, dan interoperabilitas data [20]. Selain itu, pemanfaatan model berbasis Artificial Intelligence untuk mengakses informasi pada basis data kesehatan juga mampu mempermudah proses pencarian informasi melalui bahasa alami [21]. Di sisi lain, penelitian mengenai metode *Database Life Cycle* (DBLC) menunjukkan bahwa pendekatan tersebut mampu menghasilkan rancangan basis data yang sistematis, mengurangi redundansi data, serta menjaga integritas data melalui tahapan analisis kebutuhan, desain konseptual, desain logikal, dan desain fisikal [15].

Meskipun demikian, berdasarkan penelitian terdahulu tersebut masih terdapat kesenjangan penelitian, yaitu belum adanya penelitian yang secara khusus membahas perancangan basis data hibrida yang mengintegrasikan data relasional dan data vektor menggunakan PostgreSQL dengan ekstensi pgvector pada konteks layanan customer service organisasi kemanusiaan. Sebagian besar penelitian berfokus pada pengembangan mekanisme vector retrieval, optimasi performa basis data, atau implementasi sistem pada domain tertentu, tanpa membahas secara komprehensif proses perancangan skema basis data hibrida beserta evaluasi

performanya [17], [18], [19]. Oleh karena itu, penelitian ini mengisi kesenjangan tersebut dengan merancang basis data hibrida menggunakan PostgreSQL dan ekstensi pgvector melalui pendekatan *Database Life Cycle* (DBLC), kemudian mengevaluasi performa sistem menggunakan metrik *Retrieval Quality under Filtering Conditions*, *Tail Latency*, serta *Workload Attributes* pada implementasi AI Chat Customer Service PMI Kabupaten Klaten. Dengan demikian, penelitian ini tidak hanya menghasilkan rancangan basis data yang mendukung integrasi data terstruktur dan data vektor, tetapi juga mengevaluasi kemampuan basis data tersebut dalam menyediakan pencarian informasi yang akurat dan efisien untuk mendukung layanan *customer service* berbasis AI.

2.2 Teori yang berkaitan

2.2.1 Pengertian Database

Database atau basis data merupakan struktur yang menyimpan data secara sistematis sehingga memungkinkan penyimpanan, pengambilan, pembaruan, dan pengelolaan informasi dilakukan secara konsisten dan terintegrasi. Basis data tidak hanya sekadar kumpulan data yang berdiri sendiri, tetapi merupakan representasi data yang disusun berdasarkan skema yang mendefinisikan entitas, atribut, dan relasi antar data guna mengurangi redundansi dan meningkatkan efisiensi akses informasi. Pengelolaan data yang terstruktur ini menjadi dasar bagi sistem informasi untuk menyediakan data yang akurat, andal, dan mudah diakses sesuai kebutuhan organisasi atau aplikasi yang menggunakannya [24], [25], [26].

Perangkat lunak yang digunakan untuk mengendalikan interaksi antara pengguna atau aplikasi dengan basis data dikenal sebagai *Database Management System* (DBMS). DBMS memfasilitasi operasi seperti pembuatan (*create*), pembacaan (*read*), pembaruan (*update*), dan penghapusan (*delete*) data secara terkontrol, sekaligus menjaga keamanan, integritas, dan konsistensi data melalui mekanisme seperti kontrol akses, transaksi, dan pemulihan data. Keberadaan DBMS memungkinkan pengembang dan pengguna berinteraksi dengan data tanpa perlu memahami detail fisik penyimpanan, sehingga mempercepat pengembangan dan pemeliharaan sistem informasi [27], [28], [29]. Dalam perancangan basis data, struktur skema memainkan peranan

penting karena menentukan bagaimana data direpresentasikan dan diorganisir dalam basis data. Skema yang baik mencakup pendefinisian entitas, atribut, serta relasi yang mencerminkan kebutuhan informasi sistem yang akan dibangun, sehingga basis data mampu menjamin kualitas data melalui atribut seperti kunci primer, kunci asing, normalisasi, dan aturan integritas. Perancangan skema yang matang sangat penting untuk memastikan bahwa basis data dapat berkembang tanpa terjadi inkonsistensi data ketika sistem diperluas atau dimodifikasi di masa mendatang [24], [28], [30].

Perkembangan teknologi informasi modern, khususnya kebutuhan aplikasi berbasis kecerdasan buatan dan pemrosesan data besar (*big data*), telah mendorong munculnya paradigma baru dalam pengelolaan data. Selain basis data relasional, penelitian terkini juga membahas model basis data yang mampu menyimpan dan mengelola representasi *embedding* vektor dari data seperti teks atau dokumen untuk mendukung pencarian informasi berdasarkan kedekatan semantik (*semantic search*). Pendekatan ini penting dalam aplikasi cerdas seperti sistem rekomendasi atau layanan AI yang memerlukan pencarian berdasarkan konteks dan arti, bukan sekadar pencocokan nilai (*exact match*). Integrasi basis data tradisional dengan kemampuan pencarian vektor merupakan langkah lanjutan dalam desain basis data yang adaptif terhadap kebutuhan pemrosesan data modern [12], [26], [31].

2.2.2 Pengertian Vector Database

Database vektor atau *vector database* merupakan jenis basis data yang dirancang untuk menyimpan data dalam bentuk vektor numerik berdimensi tinggi. Vektor tersebut dihasilkan dari proses *embedding*, yaitu proses transformasi data tidak terstruktur seperti teks atau dokumen menjadi representasi numerik yang merepresentasikan makna data tersebut. Dengan pendekatan ini, database vektor memungkinkan sistem melakukan pencarian informasi berdasarkan kedekatan makna atau *semantic similarity*, bukan sekadar kecocokan kata secara literal sebagaimana pada basis data relasional konvensional [32], [33].

Mekanisme utama yang digunakan dalam *database* vektor adalah pencarian berbasis kemiripan atau *similarity search*, di mana sistem

menghitung jarak antar vektor menggunakan metrik tertentu, seperti *cosine similarity* atau *euclidean distance*. Pendekatan ini memungkinkan sistem menemukan informasi yang relevan meskipun istilah yang digunakan dalam pertanyaan pengguna tidak sama persis dengan data yang tersimpan. Karakteristik tersebut menjadikan database vektor sesuai untuk mendukung sistem yang membutuhkan pemahaman konteks dan makna, seperti chatbot, sistem rekomendasi, dan layanan pencarian berbasis kecerdasan buatan [5], [34].

Dalam pengembangan sistem berbasis *artificial intelligence*, database vektor digunakan sebagai media penyimpanan pengetahuan yang dapat diakses melalui pencarian semantik. Database ini berperan penting dalam menyediakan konteks atau informasi pendukung yang relevan sebelum sistem AI menghasilkan respons. Dengan demikian, kualitas hasil pencarian dan akurasi informasi yang diperoleh pengguna sangat dipengaruhi oleh kemampuan database vektor dalam menyimpan representasi data secara tepat serta melakukan proses pencarian dengan akurasi yang baik [31], [35]. Penggunaan *database* vektor juga menuntut perhatian terhadap aspek perancangan dan pengelolaan data, terutama terkait efisiensi penyimpanan, pembentukan indeks vektor, serta kinerja pencarian pada data berdimensi tinggi. Untuk memenuhi kebutuhan tersebut, pendekatan basis data hibrida mulai digunakan dengan menggabungkan pengelolaan data terstruktur dan data vektor dalam satu sistem. Pendekatan ini memungkinkan sistem mempertahankan keunggulan basis data relasional sekaligus mendukung pencarian berbasis makna yang dibutuhkan oleh layanan AI, termasuk AI chat customer service [12], [36].

2.2.3 Entity Relationship Diagram

Entity Relationship Diagram atau ERD merupakan alat konseptual yang digunakan dalam proses perancangan basis data untuk memodelkan struktur data serta hubungan antar data secara grafis. ERD membantu perancang sistem dalam memvisualisasikan elemen-elemen data seperti entitas, atribut, dan relasi antar entitas sehingga dapat menggambarkan kebutuhan data secara logis sebelum diimplementasikan dalam basis data fisik. Dalam perancangan basis data, pemodelan menggunakan ERD merupakan langkah awal yang penting

karena dapat mengurangi kesalahan struktur data yang sering terjadi jika proses perancangan dilakukan tanpa representasi konseptual terlebih dahulu [37].

Diagram ini terdiri atas beberapa komponen dasar, yaitu: entitas yang merepresentasikan objek atau konsep utama yang datanya perlu disimpan; atribut yang menjelaskan karakteristik dari setiap entitas; dan relasi yang menunjukkan hubungan antarentitas. Notasi-notasi ini memungkinkan perancang untuk menunjukkan jenis hubungan seperti *one-to-one*, *one-to-many*, atau *many-to-many* antara entitas yang dipetakan dalam model. Pendekatan pemodelan seperti ini memberikan gambaran yang lebih jelas tentang keterkaitan data sehingga memudahkan proses normalisasi data dan desain skema basis data selanjutnya [38]. ERD juga berperan dalam memfasilitasi komunikasi antara pengembang sistem dan *stakeholder* karena bentuk visualnya memungkinkan pihak non-teknis memahami struktur data yang direncanakan. Selain itu, representasi grafis ERD menjadi dasar dalam transformasi model konseptual ke model relasional yang lebih rinci, termasuk pembuatan *primary key*, *foreign key*, serta aturan integritas data yang akan diterapkan pada basis data fisik. Hal ini penting untuk menjamin konsistensi dan akurasi data pada tahap implementasi [39].

Penggunaan ERD tidak hanya berperan pada tahap perancangan awal basis data, tetapi juga menjadi sarana untuk memahami kebutuhan sistem secara lebih mendalam, khususnya pada sistem dengan struktur data yang kompleks. Melalui pemodelan ERD, hubungan antar entitas dapat dianalisis secara sistematis sehingga mendukung pengembangan model data lanjutan yang lebih terstruktur dan konsisten. Selain itu, ERD memungkinkan perancang basis data untuk memastikan kesesuaian antara kebutuhan informasi dan representasi data yang akan diimplementasikan, sehingga proses pengembangan sistem dapat berjalan secara lebih terarah dan terkontrol. Misalnya, ERD dapat dipadukan dengan pendekatan analisis lain seperti pemodelan alur proses *Data Flow Diagram* atau DFD untuk memastikan bahwa desain data sejalan dengan alur logika proses bisnis yang lebih luas dalam sistem informasi [40]. Dengan demikian, ERD menjadi bagian penting dalam pengembangan basis data karena membantu menggambarkan hubungan antar data secara konseptual dan menjadi

dasar dalam penyusunan skema basis data yang konsisten serta sesuai dengan kebutuhan sistem informasi.

2.2.4 PostgreSQL

PostgreSQL merupakan sistem manajemen basis data relasional atau *Relational Database Management System* yang bersifat *open-source* dan banyak digunakan dalam penelitian dan industri karena kemampuannya dalam menangani data terstruktur secara konsisten dan berstandar tinggi. PostgreSQL memiliki dukungan penuh terhadap bahasa SQL serta fitur tingkat lanjut seperti transaksi ACID, yaitu *Atomicity*, *Consistency*, *Isolation*, dan *Durability* yang menjaga integritas data saat dilakukan operasi baca maupun tulis dalam jumlah besar [41]. Dalam konteks penelitian basis data, PostgreSQL sering dijadikan pilihan karena kemampuannya mengoptimalkan eksekusi *query* serta skalabilitas pada berbagai beban kerja yang beragam.

Sebagai sistem basis data relasional, PostgreSQL menyediakan berbagai macam struktur indeks seperti B-Tree, *Generalized Inverted Index* atau GIN, dan *Block Range Index* atau BRIN. Berbagai tipe indeks tersebut memungkinkan PostgreSQL untuk meningkatkan performa pencarian serta pengambilan data pada tabel yang besar atau kompleks. Analisis indeks dalam PostgreSQL menunjukkan bahwa pemilihan jenis indeks sangat mempengaruhi waktu eksekusi *query* dan penggunaan ruang penyimpanan, terutama pada *Online Transaction Processing* atau OLTP maupun *Online Analytical Processing* atau OLAP yang membutuhkan optimasi kecepatan akses data [42].

PostgreSQL dikenal sebagai sistem basis data relasional yang memiliki kinerja yang stabil dan dapat diandalkan dalam menangani operasi data terstruktur. Dalam kajian perbandingan antara PostgreSQL dan MySQL, PostgreSQL menunjukkan kemampuan menjaga latensi yang relatif rendah serta konsistensi kinerja pada eksekusi *query select* dan *insert* dengan volume data yang besar. Karakteristik ini menjadikan PostgreSQL sesuai untuk sistem yang menuntut pemrosesan transaksi yang berkelanjutan dan pengelolaan data yang konsisten [41]. Dalam konteks perancangan basis data, kemampuan tersebut mendukung kebutuhan sistem yang harus beroperasi pada variasi beban dan kompleksitas *query* yang terus berubah. Selain kinerja dan dukungan

mekanisme indeks yang matang, PostgreSQL juga dirancang untuk menangani integrasi data dan struktur skema yang kompleks. PostgreSQL menyediakan dukungan yang luas terhadap relasi antar tabel, constraint, serta pengelolaan tipe data yang beragam, sehingga memungkinkan perancangan skema basis data yang terstruktur dan konsisten. Di sisi pengelolaan sumber daya, PostgreSQL mampu memanfaatkan CPU, memori, dan media penyimpanan secara terkendali melalui mekanisme manajemen transaksi, *query planner*, dan *resource handling* yang terintegrasi. Karakteristik ini menjadikan PostgreSQL relevan tidak hanya untuk aplikasi transaksional konvensional, tetapi juga untuk sistem dengan kebutuhan pengolahan data berskala besar dan struktur data yang kompleks, tanpa mengorbankan keteraturan skema dan konsistensi data. [36]. Hal ini menegaskan bahwa PostgreSQL tidak hanya relevan untuk aplikasi tradisional, tetapi juga dalam konteks data besar yang kompleks. Dengan fitur-fitur tersebut, PostgreSQL cocok digunakan sebagai basis data dalam berbagai domain aplikasi, termasuk sistem informasi layanan yang memerlukan pengolahan *query* yang kompleks, konsistensi data yang tinggi, serta dukungan terhadap model relasional. Pada penelitian ini, PostgreSQL dipilih sebagai sistem manajemen basis data untuk mendukung perancangan basis data hibrida yang menggabungkan kemampuan relasional dengan fungsi pencarian berbasis vektor melalui ekstensi, sehingga struktur data, indeks, dan performa sistem dapat dianalisis secara komprehensif.

2.2.5 PGVector

PGVector merupakan ekstensi open-source pada PostgreSQL yang menambahkan kemampuan pencarian berbasis kesamaan vektor atau *vector similarity search* secara native di dalam DBMS relasional. Dengan ekstensi ini, PostgreSQL dapat menyimpan, mengindeks, dan mengeksekusi pencarian terhadap data vektor yang dihasilkan dari proses *embedding*, seperti teks, gambar, maupun representasi fitur lain dari model pembelajaran mesin. Keunggulan utama pgvector terletak pada kemampuannya mengintegrasikan operasi relasional berbasis SQL dengan pencarian semantik berbasis vektor dalam satu sistem, sehingga tidak diperlukan penggunaan basis data terpisah. Hal ini berdampak pada penyederhanaan arsitektur sistem serta menjaga

konsistensi dan integritas data. Dalam implementasinya, pgvector menyediakan tipe data khusus berupa vektor untuk menyimpan representasi numerik berdimensi tinggi. Data vektor ini umumnya dihasilkan dari proses embedding dan disimpan bersama atribut terstruktur lainnya dalam satu tabel. Dengan pendekatan ini, sistem memungkinkan eksekusi *query hybrid*, yaitu kombinasi antara pencarian berbasis kesamaan semantik dan filtering berbasis atribut SQL. Sebagai contoh, sistem dapat mencari dokumen yang paling mirip secara semantik terhadap *query* pengguna, sekaligus memfilter berdasarkan kategori atau metadata tertentu [43].

Untuk mendukung efisiensi pencarian pada data berdimensi tinggi, pgvector menyediakan mekanisme *indexing* khusus, yaitu *Hierarchical Navigable Small World* (HNSW) dan *Inverted File with Flat Quantization* (IVFFlat). Kedua metode tersebut termasuk dalam pendekatan *Approximate Nearest Neighbor* (ANN), yaitu metode pencarian yang bertujuan menemukan vektor paling mirip tanpa melakukan perhitungan secara eksak terhadap seluruh data. Pendekatan ini digunakan untuk meningkatkan efisiensi komputasi pada proses pencarian data berdimensi tinggi dengan kompromi tertentu pada tingkat akurasi. HNSW merupakan metode berbasis graf yang membangun struktur *multilayer* di mana setiap node terhubung dengan sejumlah tetangga terdekat. Proses pencarian dilakukan dengan menavigasi graf secara bertahap dari layer atas ke layer bawah untuk menemukan kandidat tetangga terdekat. Keunggulan HNSW terletak pada tingkat akurasi yang tinggi atau mendekati *exact search* serta performa query yang cepat, meskipun membutuhkan penggunaan memori yang lebih besar dan waktu pembangunan indeks yang relatif lebih lama. Oleh karena itu, HNSW lebih sesuai digunakan pada sistem yang membutuhkan tingkat relevansi hasil pencarian yang tinggi dan proses query yang intensif. Sebaliknya, IVFFlat bekerja menggunakan pendekatan *clustering*, di mana data vektor dikelompokkan ke dalam sejumlah centroid menggunakan algoritma seperti *k-means*. Pada saat proses query, sistem hanya melakukan pencarian pada cluster terdekat sehingga ruang pencarian menjadi lebih kecil dan proses pencarian menjadi lebih cepat. IVFFlat memiliki keunggulan dalam efisiensi memori dan kecepatan pembangunan indeks, namun memiliki *trade-off* berupa

penurunan akurasi hasil pencarian dibandingkan HNSW. Oleh karena itu, IVFFlat lebih cocok digunakan pada dataset berskala besar yang memprioritaskan efisiensi penyimpanan dan performa pencarian dengan toleransi terhadap *approximate result* [44]. Pada penelitian ini, metode *indexing* yang digunakan adalah HNSW karena sistem AI Chat Customer Service PMI Kabupaten Klaten membutuhkan tingkat akurasi dan relevansi hasil pencarian yang tinggi dalam proses *semantic search*. Penggunaan HNSW dipilih agar sistem mampu menemukan informasi yang paling sesuai dengan pertanyaan pengguna secara lebih tepat dan konsisten sehingga dapat mendukung kualitas layanan informasi yang diberikan oleh chatbot.

Dalam evaluasi sistem pencarian berbasis vektor dan SQL hybrid retrieval, penelitian ini menggunakan metrik *Retrieval Quality under Filtering Conditions*, *Latency Metrics*, dan *Workload Attributes*. Metrik-metrik tersebut digunakan untuk mengukur kualitas hasil pencarian informasi serta performa sistem dalam menjalankan proses semantic search berbasis *filtered Top-K retrieval*. Penggunaan beberapa metrik evaluasi dilakukan agar hasil pengujian tidak hanya berfokus pada relevansi informasi yang ditemukan, tetapi juga pada stabilitas *latency* dan karakteristik *workload* saat sistem memproses query pengguna dengan filter terstruktur [16]. Dengan adanya evaluasi tersebut, performa sistem dapat dianalisis secara lebih menyeluruh dari sisi kualitas retrieval, efisiensi query, dan perilaku sistem terhadap variasi *filtering* pada *hybrid retrieval*. Selain itu, penggunaan beberapa metrik evaluasi juga membantu dalam menentukan apakah rancangan basis data hybrid yang dikembangkan telah mampu mendukung kebutuhan layanan AI Chat Customer Service secara optimal.

Retrieval Quality under Filtering Conditions digunakan untuk mengukur kemampuan sistem dalam menemukan dokumen relevan pada proses filtered *Top-K retrieval*. Evaluasi dilakukan menggunakan Recall@K dengan membandingkan hasil retrieval yang diberikan sistem terhadap ground truth pada candidate set yang telah difilter menggunakan predicate tertentu. Ground truth merupakan kumpulan data acuan yang telah ditentukan sebelumnya sebagai hasil *retrieval* yang benar terhadap suatu query pada *candidate set* yang

sama. Semakin tinggi nilai Recall@K, maka semakin baik kemampuan sistem dalam menemukan dokumen relevan berdasarkan *semantic similarity* dan *filtering conditions*. Rumus Recall@K dapat dituliskan sebagai berikut pada **Rumus 2.1** [16].

$$Recall@K = \frac{|GT@K \cap Pred@K|}{K}$$

Rumus 2.1 *Recall*

Keterangan:

GT@K = ground truth top-k

Pred@K = hasil retrieval sistem

K = jumlah top-k retrieval

Sementara itu, *Latency Metrics* digunakan untuk mengukur performa waktu respon sistem dalam menjalankan proses semantic search pada hybrid retrieval. Evaluasi latency dilakukan menggunakan nilai p50, p95, dan p99 untuk menggambarkan performa query pada kondisi normal maupun pada kondisi tail latency. Nilai p50 menunjukkan median waktu respon query, sedangkan p95 dan p99 digunakan untuk mengukur stabilitas sistem pada query dengan waktu respon tertinggi. Penggunaan tail latency penting karena performa hybrid retrieval dapat dipengaruhi oleh ukuran candidate set dan kompleksitas filtering yang berbeda pada setiap query [16]. Dengan demikian, evaluasi latency tidak hanya menggambarkan rata-rata kecepatan sistem, tetapi juga menunjukkan kestabilan performa query pada berbagai kondisi workload. Semakin rendah nilai *latency* yang diperoleh, maka semakin baik performa sistem dalam memproses retrieval informasi secara *real-time*. Rumus persamaan dapat dilihat pada **Rumus 2.2**.

$$P_p = X\left[\frac{p}{100} \times n\right]$$

Rumus 2.2 *Percentile Latency*

Keterangan:

P_p *= percentile latency*

p = persentase percentile

n = jumlah query

x = data latency yang telah diurutkan

Selain itu, *Workload Attributes* digunakan untuk menganalisis karakteristik workload yang mempengaruhi perilaku retrieval pada sistem hybrid database. Evaluasi ini mencakup analisis terhadap *filter selectivity*, ukuran *candidate set* (|C|), serta distribusi query berdasarkan kondisi filtering yang digunakan dalam proses retrieval. *Filter selectivity* menunjukkan tingkat penyaringan data berdasarkan predicate tertentu, sedangkan *candidate set size* menggambarkan jumlah dokumen yang memenuhi kondisi filtering sebelum dilakukan semantic ranking. Analisis workload attributes membantu dalam memahami hubungan antara filtering intensity, retrieval quality, dan latency pada sistem hybrid retrieval [16]. Dengan adanya evaluasi ini, performa sistem dapat dianalisis secara lebih komprehensif untuk mengetahui bagaimana perubahan struktur workload mempengaruhi efisiensi dan kualitas semantic search pada basis data hybrid yang dikembangkan. Persamaan dapat dilihat pada **Rumus 2.3.**

$$Recall@5 = \frac{|GT@5 \cap Pred@5|}{5}$$

Rumus 2.3 *Workload Attributes*

Keterangan:

GT@5 = ground truth top-5

Pred@5 = hasil retrieval sistem

Evaluasi terhadap metrik-metrik tersebut dilakukan untuk mengetahui kualitas hasil retrieval serta performa sistem basis data hibrida dalam mendukung layanan AI Chat Customer Service berbasis semantic search. Melalui proses evaluasi ini, sistem dianalisis dari sisi kualitas hasil retrieval pada kondisi filtering tertentu, performa latensi query, serta karakteristik workload selama proses pencarian informasi berlangsung. Retrieval Quality under Filtering Conditions digunakan untuk mengukur kemampuan sistem dalam menemukan dokumen yang relevan pada candidate set yang telah difilter berdasarkan structured predicate tertentu. Selanjutnya, Latency Metrics, Especially Tail Latency digunakan untuk mengevaluasi kestabilan dan efisiensi waktu eksekusi query melalui pengukuran latency pada beberapa tingkat

distribusi, seperti p50, p95, dan p99. Selain itu, Workload Attributes digunakan untuk menganalisis karakteristik workload yang mempengaruhi performa retrieval, seperti filter selectivity, ukuran candidate set, serta distribusi proses retrieval pada sistem hybrid. Hasil evaluasi tersebut kemudian digunakan untuk menilai efektivitas rancangan basis data hibrida yang dikembangkan dalam mendukung proses semantic retrieval secara cepat, relevan, stabil, dan efisien pada layanan AI Chat Customer Service PMI Kabupaten Klaten.

2.3 *Database Life Cycle (DBLC)*

Database Life Cycle atau DBLC merupakan suatu kerangka kerja sistematis yang digunakan untuk mengelola seluruh proses pengembangan basis data, mulai dari tahap perencanaan hingga tahap pemeliharaan. DBLC menekankan bahwa basis data tidak dibangun secara instan, melainkan melalui serangkaian tahapan terstruktur yang saling berkaitan. Setiap tahapan dalam DBLC memiliki tujuan yang jelas untuk memastikan bahwa basis data yang dirancang mampu memenuhi kebutuhan informasi, menjaga integritas data, serta mendukung kinerja sistem informasi secara berkelanjutan. Dengan menerapkan DBLC, proses perancangan basis data menjadi lebih terkontrol dan terarah, sehingga risiko terjadinya inkonsistensi struktur data maupun kesalahan implementasi dapat diminimalkan [22], [45].

Tahap awal dalam DBLC adalah *database planning* dan *requirements analysis*, yang berfokus pada penentuan tujuan sistem basis data serta identifikasi kebutuhan data dari pengguna dan organisasi. Pada tahap ini dilakukan analisis terhadap proses bisnis, jenis data yang dibutuhkan, sumber data, serta pola penggunaan data dalam sistem. Hasil dari tahap ini berupa spesifikasi kebutuhan data yang menjadi acuan dalam perancangan basis data selanjutnya. DBLC memandang tahap analisis kebutuhan sebagai fondasi utama, karena kualitas desain basis data sangat bergantung pada ketepatan dalam menangkap kebutuhan informasi. Basis data yang dirancang tanpa analisis kebutuhan yang matang berpotensi tidak mampu mendukung proses sistem secara optimal [Perancangan Database Menggunakan Metode Database Life Cycle][Implementasi Database Life Cycle pada Sistem Informasi Relasional]. Setelah kebutuhan data teridentifikasi, DBLC dilanjutkan ke tahap perancangan basis data konseptual. Pada tahap ini, struktur data

direpresentasikan secara abstrak menggunakan model data konseptual, seperti ERD. Perancangan konseptual bertujuan untuk menggambarkan entitas, atribut, serta hubungan antar entitas tanpa terikat pada teknologi atau DBMS tertentu. Pendekatan ini memungkinkan perancang basis data untuk memfokuskan perhatian pada struktur dan makna data, bukan pada aspek implementasi teknis. Model konseptual yang baik mampu merepresentasikan kebutuhan data secara menyeluruh dan menjadi dasar yang kuat untuk tahap perancangan berikutnya [15], [45].

Tahap berikutnya dalam DBLC adalah perancangan basis data logikal, yaitu proses transformasi model konseptual ke dalam skema logikal yang sesuai dengan model basis data yang digunakan, umumnya model relasional. Pada tahap ini dilakukan pemetaan entitas menjadi tabel, atribut menjadi kolom, serta penentuan kunci primer dan kunci asing untuk menjaga integritas relasi antar data. Normalisasi juga diterapkan pada tahap ini untuk mengurangi redundansi data dan meningkatkan konsistensi penyimpanan. Perancangan logikal memastikan bahwa struktur basis data telah siap untuk diimplementasikan secara teknis tanpa kehilangan makna data yang telah didefinisikan pada tahap konseptual [15], [46]. Setelah desain logikal terbentuk, DBLC memasuki tahap perancangan basis data fisik. Tahap ini berfokus pada penyesuaian desain basis data dengan karakteristik DBMS yang digunakan, termasuk penentuan tipe data, indeks, struktur penyimpanan, serta strategi akses data. Perancangan fisik bertujuan untuk mengoptimalkan kinerja basis data, baik dari sisi kecepatan akses maupun efisiensi penggunaan sumber daya. Keputusan pada tahap ini sangat mempengaruhi performa sistem secara keseluruhan, terutama pada sistem yang menangani data dalam jumlah besar dan memiliki intensitas akses yang tinggi [22], [45].

Tahapan selanjutnya dalam DBLC mencakup implementasi dan pengujian basis data. Implementasi dilakukan dengan membangun skema basis data sesuai desain fisik yang telah ditetapkan pada DBMS yang dipilih. Setelah itu, pengujian dilakukan untuk memastikan bahwa basis data mampu menyimpan, mengelola, dan menyediakan data sesuai dengan kebutuhan sistem. Pengujian meliputi validasi struktur tabel, pengujian integritas data, serta evaluasi kinerja query. Tahap ini memastikan bahwa basis data tidak hanya benar secara struktural, tetapi juga andal dalam mendukung operasi sistem [Perancangan Basis Data Sistem Informasi

Menggunakan Database Life Cycle][Implementasi Database Life Cycle pada Sistem Informasi Relasional]. Tahap terakhir dalam DBLC adalah operasi dan pemeliharaan basis data. Pada tahap ini, basis data digunakan secara aktif dalam sistem dan terus dipantau untuk menjaga kinerja, keamanan, serta konsistensi data. Pemeliharaan mencakup kegiatan seperti backup, optimasi query, serta penyesuaian skema basis data apabila terjadi perubahan kebutuhan sistem. DBLC menekankan bahwa basis data bersifat dinamis dan harus mampu beradaptasi terhadap perkembangan sistem, sehingga proses pemeliharaan menjadi bagian yang tidak terpisahkan dari siklus hidup basis data [15], [45].

2.4 Tools/software yang digunakan

2.4.1 PgAdmin4

PgAdmin merupakan perangkat lunak berbasis antarmuka grafis atau *graphical user interface* (GUI) yang digunakan untuk mengelola dan mengadministrasikan basis data PostgreSQL. PgAdmin dikembangkan sebagai *tools* resmi PostgreSQL yang menyediakan berbagai fitur untuk membantu pengguna dalam melakukan pengelolaan basis data secara visual sehingga memudahkan proses administrasi tanpa harus sepenuhnya menggunakan perintah berbasis *command line*. Sebagai *tools* administrasi PostgreSQL, PgAdmin mendukung berbagai aktivitas seperti pembuatan basis data, pengelolaan tabel, manipulasi data, pengaturan pengguna dan hak akses, hingga eksekusi query SQL secara terintegrasi. Dengan kemampuan tersebut, PgAdmin menjadi salah satu *tools* yang banyak digunakan dalam proses pengembangan maupun pengelolaan sistem basis data berbasis PostgreSQL. Dalam konteks pengelolaan basis data, PgAdmin menyediakan fasilitas untuk menampilkan struktur skema basis data secara terorganisasi, termasuk tabel, atribut, relasi, indeks, *constraint*, *function*, dan *trigger*. Antarmuka visual yang dimiliki PgAdmin membantu pengguna memahami hubungan antar objek basis data dengan lebih mudah sehingga mendukung proses perancangan, implementasi, dan evaluasi struktur basis data. Selain itu, PgAdmin juga menyediakan fitur *Query Tool* yang memungkinkan pengguna menjalankan, menguji, dan memvalidasi query SQL secara langsung terhadap basis data PostgreSQL. Fitur tersebut mendukung aktivitas pengembangan dan pengujian

basis data secara lebih efisien dan terstruktur. Perkembangan PgAdmin saat ini telah mencapai PgAdmin 4 (*pgAdmin4*), yang merupakan versi terbaru dengan dukungan teknologi berbasis web dan desktop. pgAdmin4 dirancang dengan antarmuka yang lebih modern, fleksibel, dan kompatibel pada berbagai sistem operasi seperti Windows, Linux, dan macOS. Dibandingkan versi sebelumnya, pgAdmin4 memiliki peningkatan pada aspek performa, manajemen koneksi, visualisasi objek basis data, serta kemudahan navigasi dalam pengelolaan basis data PostgreSQL. Selain itu, pgAdmin4 juga mendukung pengelolaan ekstensi PostgreSQL, termasuk pgvector, yang digunakan untuk menyimpan dan memproses data vektor pada implementasi *semantic search* dan *AI-based retrieval system*. Penggunaan pgAdmin4 dalam penelitian ini berfungsi sebagai *database management tool* untuk mendukung proses perancangan, pengelolaan, dan pengujian basis data PostgreSQL secara visual dan terkontrol. Melalui pgAdmin4, proses pembuatan skema basis data, pengelolaan tabel, pengujian query, serta validasi struktur basis data dapat dilakukan dengan lebih sistematis dan efisien. Oleh karena itu, pgAdmin4 relevan digunakan dalam penelitian ini sebagai alat bantu administrasi dan pengelolaan basis data hibrid berbasis PostgreSQL dengan ekstensi pgvector.

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA