BAB III

METODOLOGI PENELITIAN

3.1 Gambaran Umum Objek Penelitian

PT ABC adalah perusahaan yang bergerak di bidang penyediaan layanan teknologi informasi (*IT Solutions Provider*) yang berfokus pada pengembangan dan implementasi sistem, aplikasi, serta infrastruktur untuk mendukung kebutuhan bisnis berbagai organisasi. Sebagai perusahaan IT, PT ABC menyediakan layanan pengembangan aplikasi berbasis web dan mobile, pengembangan sistem manajemen proyek, integrasi sistem, manajemen server dan cloud, hingga layanan strategi IT untuk membantu klien meningkatkan efisiensi operasional dan transformasi digital. Dalam aktivitas bisnisnya, perusahaan menangani berbagai data operasional yang mencakup data proyek, data pengguna, transaksi layanan, data pelanggan, serta aktivitas pemantauan sistem. Peningkatan jumlah klien dan bertambahnya penggunaan aplikasi internal maupun aplikasi yang dikembangkan untuk pihak eksternal menyebabkan volume data yang dikelola perusahaan terus bertambah setiap hari.

PT ABC menggunakan PostgreSQL sebagai sistem *database* utama dalam pengelolaan data operasionalnya karena sifatnya yang *open-source*, stabil, dan memiliki fitur lengkap untuk menangani skala data yang besar. *Database* tersebut menjadi komponen penting dalam berbagai aplikasi yang dikembangkan Perusahaan mulai dari aplikasi manajemen proyek, *dashboard monitoring*, sistem pelaporan, hingga layanan *backend* lainnya. Namun, seiring bertambahnya jumlah proyek dan aktivitas pengguna, perusahaan mulai menghadapi berbagai permasalahan performa pada *database*nya, terutama saat menangani tabel-tabel besar seperti data transaksi proyek, order, aktivitas pengguna, dan data historis yang terus bertambah dalam jangka panjang. Beberapa *query* tertentu membutuhkan waktu eksekusi yang lebih lama, terutama saat melakukan pencarian data berdasarkan rentang waktu, pemanggilan data historis, serta proses agregasi laporan yang kompleks.

Permasalahan performa ini disebabkan oleh meningkatnya skala data yang membuat PostgreSQL lebih sering melakukan *sequential scan* dan *full table scan* pada tabel-tabel besar. Kondisi tersebut berdampak pada lambatnya waktu respon aplikasi internal, keterlambatan penyajian laporan proyek, dan meningkatnya beban kerja server. Masalah ini juga memengaruhi produktivitas tim operasional dan manajerial, karena sebagian besar keputusan proyek dan evaluasi kinerja bergantung pada data yang harus diakses secara cepat. Oleh sebab itu, PT ABC menjadi objek penelitian yang sangat relevan untuk penerapan teknik optimasi *database* seperti *indexing* dan *partitioning*. Kedua teknik tersebut diperlukan untuk mempercepat proses pencarian data, memperkecil ukuran data pada proses query, serta meningkatkan efisiensi pemanggilan data historis dalam sistem operasional perusahaan.

Dengan latar belakang permasalahan tersebut, penelitian ini dilakukan untuk mengimplementasikan strategi optimasi *database* menggunakan teknik *indexing* dan *partitioning* pada tabel-tabel besar di sistem PostgreSQL yang digunakan PT ABC. Pendekatan ini diharapkan dapat meningkatkan performa query secara signifikan, mengurangi waktu eksekusi operasi *SELECT*, mempercepat proses akses data historis, dan meningkatkan stabilitas aplikasi yang bergantung pada *database* tersebut. Selain memberikan manfaat langsung bagi perusahaan, hasil penelitian ini juga dapat menjadi studi kasus praktis bagi organisasi lain yang memiliki karakteristik serupa dalam penggunaan PostgreSQL untuk mendukung aktivitas operasional ber*database*.

Gambar 3. 1 Relasi Data

Pada gambar 3.1 ditunjukkan rancangan *Entity Relationship Diagram* (ERD) yang menggambarkan struktur data PT ABC serta hubungan antar tabel yang digunakan dalam penelitian ini. Rancangan ini berfungsi sebagai dasar dalam proses implementasi *database tuning* pada sistem PostgreSQL. ERD tersebut terdiri dari beberapa entitas utama yang merepresentasikan data operasional PT ABC, seperti c_order, c_orderline, c_project, m_product, ad_user, dan entitas lain yang saling berelasi untuk membentuk satu kesatuan sistem informasi.

Proses pengumpulan data untuk penelitian ini dilakukan pada tanggal 09 September 2025 di kantor PT ABC. Pada kesempatan tersebut dilakukan permohonan resmi kepada pihak perusahaan untuk memperoleh akses terhadap data operasional sebagai bahan penelitian dalam optimasi sistem *database* PostgreSQL. Data transaksi asli tidak dapat diberikan karena termasuk informasi yang bersifat rahasia dan dilindungi oleh kebijakan internal perusahaan. Sebagai bentuk dukungan, perusahaan menyediakan struktur *database* lengkap, termasuk schema tabel, tipe data, atribut utama, serta *Entity Relationship Diagram* (ERD) yang merepresentasikan sebagian hubungan antar entitas dalam sistem. Informasi ini diberikan secara langsung oleh staf IT perusahaan yang bertanggung jawab terhadap pengelolaan *database*.

Perusahaan juga memberikan akses untuk melihat schema *database* melalui lingkungan yang digunakan secara internal. Namun, jumlah relasi yang sangat besar menyebabkan proses pemuatan visualisasi schema tidak dapat ditampilkan secara menyeluruh, karena keterbatasan kapasitas perangkat yang digunakan untuk menampilkan diagram relasi penuh. Untuk mengatasi kendala tersebut, staf IT menyediakan sepuluh tabel utama dalam bentuk file CSV yang diekspor langsung dari sistem. Tabel-tabel tersebut kemudian digunakan sebagai dasar penyusunan ERD pada penelitian ini, sehingga struktur entitas, atribut, dan relasi yang ditampilkan tetap sesuai dengan *database* operasional perusahaan.

Penelitian ini menggunakan data dummy yang dibangun mengikuti struktur tabel pada file CSV yang diberikan perusahaan. Seluruh relasi, tipe data, dan karakteristik struktur skema direplikasi agar tetap menggambarkan kondisi nyata

dari sistem produksi yang digunakan oleh PT ABC. Periode data yang disimulasikan juga disesuaikan dengan pola transaksi yang umum terjadi dalam kegiatan operasional perusahaan, sehingga pengujian performa terhadap teknik indexing dan *partitioning* tetap relevan dan representatif. Dengan pendekatan ini, validitas penelitian tetap terjaga meskipun data yang digunakan merupakan hasil simulasi, karena skema, model hubungan, dan volume data yang diuji tetap berdasarkan struktur *database* aktual perusahaan.

Entitas bpartner merepresentasikan data mitra bisnis atau pelanggan yang terlibat dalam kegiatan transaksi perusahaan. Setiap mitra bisnis memiliki atribut seperti ad_client_id, *isactive*, *create d*, *update*d, *value*, *name*, dan *firstsale* yang berfungsi untuk menyimpan identitas serta status keaktifan mitra. Entitas ini menjadi referensi bagi entitas lain, seperti c_order dan c_project, untuk menunjukkan relasi antara proyek atau pesanan dengan pelanggan tertentu.

Entitas ad_user digunakan untuk menyimpan informasi pengguna sistem, seperti nama, email, dan kata sandi, yang terhubung dengan kolom c_bpartner_id sebagai relasi ke tabel bpartner. Hubungan ini menunjukkan bahwa setiap pengguna sistem dapat dikaitkan dengan satu mitra bisnis tertentu. Selain itu, entitas ini juga memiliki peran penting dalam sistem otorisasi dan pengelolaan aktivitas pengguna di dalam *database*.

Entitas c_project merepresentasikan data proyek yang dijalankan oleh perusahaan. Setiap proyek memiliki atribut seperti name, description, note, *create* d, *update*d, dan isactive. Kolom ad_user_id dan c_bpartner_id menunjukkan hubungan langsung proyek terhadap pengguna dan mitra bisnis. Tabel ini juga menjadi salah satu fokus utama dalam penelitian karena digunakan untuk penerapan *Range Partitioning* berdasarkan kolom *create d*, dengan tujuan mengoptimalkan proses pengambilan data proyek aktif dan nonaktif.

Entitas m_area berfungsi untuk menyimpan informasi area kerja atau wilayah proyek yang berkaitan dengan data fisik lapangan. Setiap area memiliki atribut seperti nama_pmabj, no_pmabj, dan link_logo yang menggambarkan

identitas wilayah. Entitas ini berelasi dengan tabel c_project untuk memetakan lokasi proyek tertentu.

Entitas m_product menyimpan data produk atau barang yang digunakan dalam transaksi. Kolom-kolom seperti *value, name, description, volume*, dan *weight* menjelaskan karakteristik setiap produk. Produk dikelompokkan berdasarkan m_product_category_id untuk memudahkan pengelolaan inventori dan analisis data. Tabel ini juga digunakan untuk penerapan *indexing* pada kolom name dan m_product_category_id guna mempercepat pencarian produk.

Entitas c_order merupakan tabel utama yang menyimpan data transaksi pemesanan dari pelanggan. Atribut penting di dalamnya antara lain *documentno, docstatus, isdelivered, isinvoiced, totallines*, dan *grandtotal*. Tabel ini berelasi dengan c_orderline, c_project, ad_user, dan bpartner. Dalam penelitian ini, c_order digunakan untuk implementasi *Range Partitioning* berdasarkan kolom *create d*, dengan tujuan meningkatkan efisiensi *query* pada data transaksi yang memiliki rentang waktu yang panjang.

Entitas c_orderline menyimpan rincian dari setiap transaksi pada tabel c_order. Tabel ini memiliki atribut seperti *dateordered, datepromised, pricelist, discount*, dan m_product_id untuk menunjukkan hubungan antara pesanan dan produk. c_orderline merupakan salah satu tabel dengan jumlah data terbesar di dalam sistem (mencapai 30 juta baris), sehingga menjadi fokus utama dalam pengujian performa *indexing* dan *partitioning*.

Selain itu, terdapat entitas m_requisition dan m_requisitionline yang digunakan untuk mengelola permintaan barang dari pengguna. Hubungan antara m_requisition dan m_requisitionline bersifat *one-to-many*, di mana satu permintaan dapat memiliki beberapa item permintaan. Kolom ad_user_id digunakan untuk mencatat pengguna yang membuat permintaan, sedangkan kolom *processing* dan ispayed menggambarkan status eksekusi permintaan tersebut.

Entitas m_reorder berfungsi untuk mencatat proses pemesanan ulang barang berdasarkan data permintaan. Atribut seperti m_requisitionorder_id, qty,

dan c_orderline_id menunjukkan keterkaitan antara pesanan ulang dan transaksi pembelian sebelumnya.

Secara keseluruhan, hubungan antar entitas pada Gambar 3.1 membentuk sistem *database* yang saling terintegrasi antara pelanggan, pengguna, proyek, pesanan, produk, dan permintaan barang. Rancangan ERD ini menjadi dasar penting dalam proses performance *tuning database* PostgreSQL, karena struktur relasi antar tabel dan jumlah data yang besar berpengaruh langsung terhadap strategi optimasi yang diterapkan melalui teknik *indexing* dan *partitioning*.

Selama ini, sistem *database* yang digunakan perusahaan sering mengalami penurunan performa ketika melakukan pemrosesan data dalam jumlah besar, baik pada proses pencarian dan penampilan data historis atau data lama. Ketika sistem harus menampilkan data yang telah tersimpan dalam jangka waktu lama, waktu respon menjadi lambat karena seluruh tabel harus dipindai secara penuh (*full table scan*). Hal ini dapat menyebabkan keterlambatan dalam pengambilan keputusan oleh pihak manajemen, terutama pada divisi yang memerlukan akses cepat terhadap data transaksi masa lalu atau laporan periodik. Oleh karena itu, diperlukan langkah strategis untuk meningkatkan efisiensi dan kecepatan dalam proses akses data.

Salah satu pendekatan yang digunakan dalam penelitian ini adalah penerapan teknik *indexing* dan *partitioning* pada sistem *database* PostgreSQL.

a) *Indexing* dilakukan untuk mempercepat proses pencarian data dengan membangun struktur *index* pada kolom tertentu yang sering digunakan dalam *query*. Dengan adanya *index*, sistem tidak perlu memindai seluruh tabel, melainkan cukup mencari melalui struktur *index* yang lebih ringan, sehingga waktu respon dapat berkurang secara signifikan.
b) *Partitioning*, di sisi lain, digunakan untuk membagi tabel besar menjadi beberapa bagian yang lebih kecil (partisi) berdasarkan kriteria tertentu, seperti rentang waktu atau jenis data. Dengan cara ini, *query* yang ditujukan untuk data lama hanya akan menelusuri partisi yang relevan, bukan keseluruhan dataset. Hal ini terbukti dapat meningkatkan performa sistem

secara keseluruhan, terutama dalam proses pemanggilan data historis yang sering dilakukan oleh PT ABC.

Optimalisasi *database* ini dilakukan dengan menganalisis struktur *index* yang ada, mengamati *query execution plan*, serta menyesuaikan konfigurasi server *database* agar lebih efisien dalam menangani beban kerja yang tinggi. Melalui penelitian ini, diharapkan diperoleh strategi optimasi yang tepat, baik dari sisi desain skema *database* maupun dari penggunaan fitur-fitur PostgreSQL yang mendukung performa tinggi seperti *indexing*, *partitioning*, dan pengelolaan cache.

Dengan adanya penerapan teknik tersebut, diharapkan proses pengolahan data pada PT ABC dapat berjalan dengan lebih cepat, stabil, dan efisien tanpa mengorbankan integritas serta konsistensi data yang ada. Hasil dari penelitian ini diharapkan tidak hanya memberikan peningkatan kinerja signifikan pada sistem *database* PT ABC, tetapi juga dapat menjadi referensi dan studi kasus bagi organisasi lain yang memiliki karakteristik sistem informasi dan volume data yang serupa, terutama dalam penerapan teknik optimasi *database* berbasis PostgreSQL.

3.2 Metode Penelitian

Metode penelitian yang digunakan mengacu pada pendekatan experimental setup dan performance benchmarking, sebagaimana diterapkan dalam penelitian Avula (2024) mengenai strategi partitioning pada PostgreSQL [13]. Pendekatan ini merupakan metode yang lazim digunakan dalam penelitian Sistem Informasi dan Ilmu Komputer untuk mengevaluasi perubahan performa sistem setelah diberikan perlakuan tertentu secara terkontrol dan terukur.

Penelitian dilakukan dengan membangun lingkungan uji (testing environment) yang mereplikasi struktur database operasional PT ABC berdasarkan skema yang diberikan oleh perusahaan. Teknik optimasi berupa indexing dan partitioning diterapkan sebagai perlakuan (treatment) terhadap struktur tabel, kemudian diuji menggunakan skenario query yang sama pada dua kondisi, yaitu sebelum dan sesudah optimasi. Metrik pengukuran yang digunakan meliputi execution time, query plan cost, penggunaan sequential scan, serta efisiensi akses data.

Pendekatan experimental setup ini memungkinkan evaluasi yang objektif dan sistematis terhadap efektivitas teknik optimasi yang diterapkan. Proses pengujian dilakukan dengan membandingkan hasil benchmark pada setiap perlakuan, sehingga hasil pengukuran dapat dianalisis secara empiris untuk menilai peningkatan performa sistem database PostgreSQL.

3.2.1. Desain Eksperimen

3.2.1.1. Penentuan Workload dan Query Pengujian

Workload dan query pengujian yang digunakan dalam penelitian ini tidak diambil secara langsung dari query log operasional PT ABC, melainkan disusun dalam bentuk simulasi berdasarkan pemahaman peneliti terhadap struktur database dan kebutuhan fungsional sistem. Hal ini dilakukan karena keterbatasan akses terhadap query log produksi yang bersifat rahasia dan tidak dapat dibagikan oleh perusahaan.

Penyusunan query pengujian dilakukan dengan mempelajari skema database PT ABC, relasi antar tabel, serta fungsi utama sistem informasi yang berjalan. Berdasarkan pemahaman tersebut, peneliti merancang beberapa query SELECT yang merepresentasikan skenario umum penggunaan sistem, seperti pencarian data berdasarkan rentang waktu, pengambilan data historis, serta penggabungan data dari beberapa tabel yang saling berelasi. Query yang disusun difokuskan pada operasi SELECT karena penelitian ini bertujuan untuk mengevaluasi performa database pada kondisi SELECT-heavy workload.

3.2.1.2. Prosedur Pengujian Eksperimen

Pengujian performa database dilakukan secara sistematis untuk mengevaluasi perbedaan kinerja sistem sebelum dan sesudah penerapan teknik optimasi. Pada tahap awal, query pengujian dijalankan pada kondisi baseline, yaitu kondisi database tanpa penerapan indexing dan partitioning. Waktu eksekusi setiap query

pada kondisi ini dicatat menggunakan perintah EXPLAIN ANALYZE sebagai acuan performa awal. Selanjutnya, teknik optimasi berupa indexing dan partitioning diterapkan sesuai dengan skenario penelitian yang telah ditetapkan, kemudian query yang sama dijalankan kembali pada kondisi setelah optimasi untuk memperoleh data performa pembanding.

Setiap query pengujian dijalankan sebanyak lima kali untuk mengurangi pengaruh fluktuasi sistem dan memperoleh hasil pengukuran yang lebih stabil. Nilai execution time yang digunakan dalam analisis merupakan nilai rata-rata dari lima kali eksekusi tersebut. Pendekatan ini bertujuan untuk meminimalkan bias pengukuran yang dapat disebabkan oleh aktivitas sistem latar belakang atau variasi penggunaan sumber daya selama proses pengujian.

Dalam pelaksanaan pengujian, kondisi cache juga diperhatikan untuk meningkatkan relevansi hasil eksperimen. Eksekusi pertama digunakan sebagai tahap warm-up, sedangkan eksekusi berikutnya merepresentasikan kondisi warm cache yang lebih mendekati perilaku sistem database dalam kondisi operasional normal. Pendekatan ini dipilih karena sistem database PT ABC pada lingkungan produksi umumnya berjalan dalam kondisi cache aktif, sehingga hasil pengujian yang diperoleh diharapkan lebih merefleksikan performa sistem pada penggunaan nyata.

3.2.1.3. Metrik Evaluasi Performa

Evaluasi performa dalam penelitian ini difokuskan pada satu metrik utama, *yaitu execution* time atau waktu eksekusi *query*. *Execution time* didefinisikan sebagai durasi waktu yang dibutuhkan oleh sistem database PostgreSQL untuk mengeksekusi sebuah query SELECT secara keseluruhan, mulai dari tahap perencanaan (*planning*) hingga proses pengambilan hasil (*execution*). Metrik ini

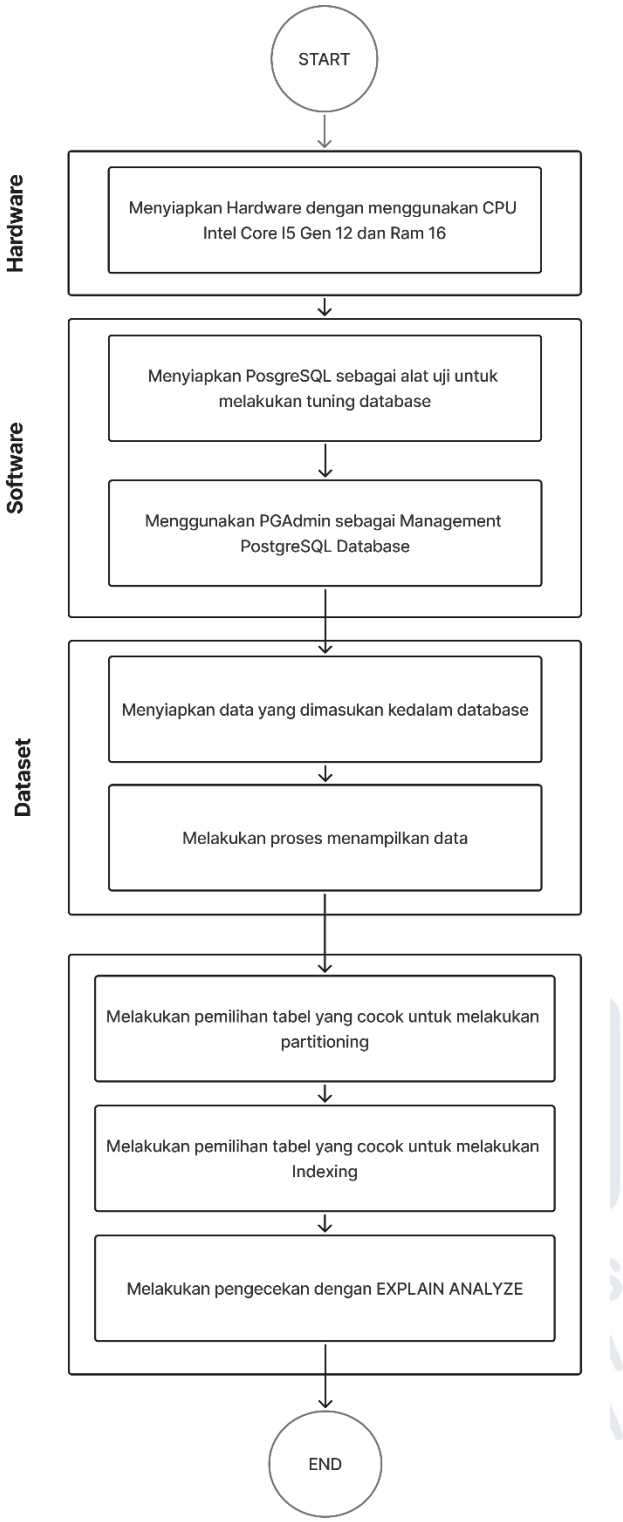
dipilih karena secara langsung merepresentasikan kinerja sistem database dari sudut pandang pengguna maupun aplikasi yang bergantung pada kecepatan respons *query*.

Dalam setiap skenario pengujian, baik sebelum maupun sesudah penerapan teknik indexing dan partitioning, setiap query dijalankan sebanyak lima (5) kali eksekusi berturut-turut. Nilai *execution time* yang digunakan dalam analisis merupakan nilai rata-rata (*average*) dari lima kali eksekusi tersebut. Pendekatan ini dilakukan untuk mengurangi pengaruh fluktuasi performa akibat kondisi *cold cache* pada eksekusi awal serta kondisi *warm cache* pada eksekusi selanjutnya, sehingga hasil pengukuran performa yang diperoleh bersifat lebih stabil, objektif, dan dapat dibandingkan secara adil antar skenario pengujian.

3.2.2. Alur Penelitian

Alur penelitian ini disusun berdasarkan hasil pengembangan dari penelitian terdahulu yang memiliki fokus serupa, yakni pada proses optimasi dan pengujian performa sistem *database*. Penelitian ini mengadaptasi tiga komponen utama yang saling berkaitan dalam pelaksanaannya, yaitu *hardware, software*, dan dataset. Ketiga komponen tersebut menjadi dasar yang menentukan validitas, reliabilitas, serta keberhasilan dari seluruh proses eksperimen yang dilakukan.

Secara umum, alur penelitian dimulai dari tahap persiapan perangkat keras dan perangkat lunak hingga tahap pengujian hasil optimasi dengan menggunakan perintah *EXPLAIN ANALYZE*. Setiap tahapan dilakukan secara sistematis agar hasil pengujian dapat menggambarkan secara akurat pengaruh penerapan teknik *indexing* dan *partitioning* terhadap waktu eksekusi *query* di sistem *database* PostgreSQL.



Gambar 3. 2 Alur Penelitian [13]

Pada Gambar 3.2 ditunjukkan alur penelitian yang menggambarkan tahapan proses implementasi *database tuning* menggunakan teknik *indexing* dan *partitioning*. Proses penelitian diawali dengan tahap persiapan perangkat keras (hardware), yaitu menggunakan laptop dengan spesifikasi CPU Intel Core i5 Generasi ke-12 dan RAM sebesar 16 GB. Spesifikasi tersebut dipilih untuk memastikan sistem mampu menjalankan proses pemrosesan data berskala besar secara stabil dan efisien.

Tahap berikutnya adalah persiapan perangkat lunak (*software*) dengan melakukan instalasi dan konfigurasi PostgreSQL sebagai sistem *database* utama yang digunakan dalam pengujian performa, serta pgAdmin 4 sebagai alat bantu (*management tool*) dalam mengelola *database*, menjalankan perintah SQL, dan memantau hasil *tuning*. Setelah lingkungan sistem siap, dilakukan proses pembuatan dan pengisian data uji (dataset) sebanyak 80.001.000 baris ke dalam beberapa tabel di *database*. Dataset berukuran besar ini digunakan untuk menguji kemampuan PostgreSQL dalam menangani beban kerja tinggi secara rea*list*is.

Setelah data berhasil dimasukkan, dilakukan tahap penampilan dan pengujian awal data untuk mengetahui performa sistem sebelum dilakukan proses optimasi. Pengujian ini bertujuan untuk memperoleh nilai dasar (*baseline*) terhadap waktu eksekusi *query* dan respon sistem pada kondisi awal. Selanjutnya, dilakukan pemilihan tabel yang akan diterapkan teknik *partitioning*. Tidak semua tabel dipartisi karena setiap tabel memiliki karakteristik dan pola akses data yang berbeda. Jenis partisi yang digunakan disesuaikan dengan kebutuhan dan karakteristik data, yaitu *Range Partitioning*, *List Partitioning*, dan *Hash Partitioning*. Pemilihan metode partisi yang kurang tepat dapat menyebabkan penurunan performa atau kompleksitas berlebih dalam pengelolaan data.

Pada penelitian ini, penggunaan *list partitioning* dipilih karena kolom yang dijadikan dasar partisi memiliki nilai kategori yang relatif stabil dan jarang mengalami penambahan. Berdasarkan analisis terhadap struktur data PT ABC, perubahan nilai pada kolom tersebut tidak terjadi secara dinamis dalam periode pengujian. Dengan demikian, potensi penambahan partisi baru dapat diminimalkan

dan tidak berdampak signifikan terhadap hasil pengujian performa. Pemilihan list partitioning pada penelitian ini difokuskan untuk mengevaluasi dampak pembagian data terhadap performa query SELECT, bukan untuk menguji aspek skalabilitas jangka panjang.

Tahap berikutnya adalah penerapan *indexing* pada tabel yang memerlukan peningkatan kecepatan pencarian data. Teknik *index* tidak diterapkan pada seluruh tabel, melainkan hanya pada kolom-kolom yang sering digunakan dalam proses pencarian, penyaringan (*filtering*), atau pengurutan data (*sorting*). Jenis *index* yang digunakan dalam penelitian ini adalah B-Tree *Index*, karena tipe ini paling optimal untuk operasi berbasis nilai seperti perbandingan dan pencarian data.

Tahap terakhir adalah pengujian performa pasca-optimasi menggunakan perintah *EXPLAIN ANALYZE*. Perintah ini berfungsi untuk menganalisis rencana eksekusi (*execution plan*) yang dihasilkan PostgreSQL, sekaligus mengukur waktu eksekusi *query* sebelum dan sesudah penerapan teknik *indexing* serta *partitioning*. Melalui hasil pengujian ini, diperoleh perbandingan performa sistem secara kuantitatif yang menjadi dasar dalam evaluasi efektivitas strategi *tuning* yang diterapkan pada *database* PostgreSQL.

3.2.2.1. Hardware

Komponen *hardware* berperan sebagai lingkungan fisik tempat seluruh proses eksperimen dijalankan. Pemilihan perangkat keras dilakukan dengan mempertimbangkan kebutuhan sumber daya agar dapat menjalankan pengujian dengan beban data yang besar. Pada penelitian ini digunakan perangkat komputer dengan spesifikasi prosesor Intel Core i5 generasi ke-12 dan memori (RAM) sebesar 16 GB. Spesifikasi tersebut dipilih untuk memastikan sistem mampu menangani proses eksekusi *query*, pembuatan *index*, dan partisi data tanpa mengalami gangguan performa. Lingkungan pengujian yang stabil juga memastikan hasil pengukuran waktu eksekusi *query* lebih akurat dan dapat direplikasi.

Spesifikasi perangkat keras (*hardware*) yang digunakan dalam proses *partitioning* dan *indexing* pada sistem *database*.

Spesifikasi ini mencakup komponen utama seperti prosesor dan ram yang berperan penting dalam mendukung performa proses uji coba. Pemilihan perangkat keras dilakukan untuk memastikan sistem memiliki kemampuan komputasi yang memadai dalam menangani jumlah data besar serta beban kerja tinggi selama proses *database tuning* berlangsung.

3.2.2.2. Software

Komponen *software* mencakup seluruh perangkat lunak yang digunakan dalam proses penelitian, mulai dari sistem operasi, sistem manajemen *database* (DBMS), hingga alat bantu pengujian performa.

Dalam penelitian ini digunakan PostgreSQL sebagai DBMS utama yang berfungsi untuk mengelola data serta melakukan proses *tuning* terhadap performa *database*. Selain itu, digunakan PGAdmin sebagai alat bantu manajemen PostgreSQL untuk memudahkan pengaturan skema *database*, pembuatan *index*, serta pemantauan hasil pengujian.

Alat bantu tambahan seperti perintah *EXPLAIN ANALYZE* digunakan untuk mengukur waktu eksekusi *query* dan menampilkan *query* execution plan, yang menjadi dasar dalam analisis performa sebelum dan sesudah dilakukan optimasi. Penerapan teknik *indexing* dan *partitioning* dilakukan di dalam lingkungan PostgreSQL ini untuk mengamati sejauh mana peningkatan efisiensi yang dapat dicapai setelah dilakukan perubahan struktur data.

3.2.2.3. Dataset

Komponen dataset berfungsi sebagai sumber data yang digunakan untuk pengujian performa *database*. Dataset ini disusun sedemikian rupa agar menyerupai kondisi data aktual yang dimiliki oleh PT ABC, baik dari segi struktur tabel maupun volume datanya. Dataset berisi data transaksi dan data historis dengan ukuran yang

cukup besar untuk mensimulasikan beban kerja nyata sistem *database* perusahaan.

Data tersebut dimasukkan ke dalam PostgreSQL dan digunakan untuk menjalankan *query* pengujian, seperti proses pencarian, pemanggilan data lama, serta pemrosesan data agregat. Penggunaan dataset yang realistis ini memungkinkan untuk diamati secara langsung bagaimana penerapan teknik *indexing* dan *partitioning* dapat mempengaruhi waktu eksekusi *query*. Dengan demikian, hasil penelitian dapat mencerminkan situasi operasional yang sesungguhnya dan memberikan gambaran konkret mengenai efektivitas teknik optimasi yang diterapkan.

3.3 Teknik Pengumpulan Data

Teknik pengumpulan data dalam penelitian ini dilakukan dengan cara membangun dan menggunakan data dummy yang disimulasikan ke dalam sistem database PostgreSQL. Data dummy tersebut dibuat untuk merepresentasikan kondisi nyata dari data yang dimiliki oleh PT ABC, baik dari segi jumlah maupun struktur tabel yang digunakan. Pembuatan data dummy ini bertujuan agar proses pengujian dapat dilakukan secara terkendali tanpa bergantung pada data operasional asli perusahaan, sekaligus menjaga kerahasiaan informasi internal.

Dalam penelitian ini, jumlah data yang digunakan mencapai 80.001.000 baris (*records*). Jumlah tersebut dipilih untuk menciptakan skenario yang realistis terhadap kondisi sistem database berskala besar, sehingga hasil pengujian dapat menunjukkan secara akurat bagaimana kinerja sistem berubah setelah dilakukan proses optimasi. Data dummy dihasilkan menggunakan *script generator* yang dirancang untuk menghasilkan variasi data yang beragam dan menyerupai pola distribusi data aktual pada sistem perusahaan, seperti data transaksi, data pelanggan, serta data historis.

Data dummy yang digunakan dalam penelitian ini dirancang agar tidak mengandung data redundant, khususnya pada atribut-atribut yang bersifat kunci, seperti *primary key* dan *unique key*. Hal ini dilakukan untuk memastikan bahwa penurunan atau peningkatan performa yang terjadi benar-benar disebabkan oleh

penerapan teknik optimasi database, bukan akibat adanya duplikasi data yang dapat memengaruhi hasil pengujian. Dengan demikian, struktur dan integritas data tetap terjaga sesuai dengan kaidah normalisasi database.

Setelah data dummy berhasil dibuat, langkah selanjutnya adalah memasukkan data tersebut ke dalam tabel database PostgreSQL. Proses ini dilakukan secara bertahap untuk memastikan seluruh data tersimpan dengan benar dan sistem dapat menanganinya tanpa kendala. Setelah data terisi penuh, dilakukan pengujian performa awal dengan menjalankan beberapa query dasar, seperti query pencarian (*SELECT*), penyaringan (*WHERE*), serta pengambilan data berdasarkan kondisi tertentu.

Pengumpulan data dilakukan dengan mencatat waktu eksekusi setiap query menggunakan perintah EXPLAIN ANALYZE, baik sebelum maupun sesudah dilakukan penerapan teknik optimasi. Dengan cara ini, diperoleh data numerik yang valid dan terukur mengenai perbandingan performa sistem database. Data hasil pengukuran tersebut kemudian disimpan dan diolah untuk dianalisis secara kuantitatif, dengan tujuan untuk mengetahui sejauh mana penerapan teknik *indexing* dan *partitioning* berpengaruh terhadap peningkatan kecepatan pemrosesan data.

Dengan demikian, teknik pengumpulan data pada penelitian ini berperan penting dalam menyediakan data empiris yang akurat, terkontrol, dan dapat diukur secara objektif, sehingga hasil penelitian dapat dijadikan dasar untuk menarik kesimpulan yang valid mengenai efektivitas metode optimasi database pada PostgreSQL.