O'REILLY®
Business Guide

Building a Data and AI Platform with PostgreSQL

Transforming Your Business with Intelligent Data

Tom Taulli, Benjamin Anderson & Jozef de Vries

EDB POSTGRES AI

Postgres®

AI and data platforming

This book is the blueprint for your future as an AI and data platform company. Where data, AI, and intelligence operate as one, governed and grounded in open source strength.

It also brings together industry experts on how this world will evolve over the next 5-10 years. It is your blueprint, and your North Star.

With **EDB Postgres AI**, that vision becomes real — a unified foundation for building, running, and scaling intelligent applications with full control over your data, everywhere it lives.

Sovereign and secure

Scalable and agile

Based on open source foundations

Low-code/no-code AI factory

"This book answers the foundational question about how to build your AI and data platform on the most popular open source data layer in the world, Postgres."

— Nicholas D. Evans, Founder, Thinkers360 and Consulting Magazine Lifetime Achievement Award Winner

Building a Data and AI Platform with PostgreSQL

Transforming Your Business with Intelligent Data

Tom Taulli, Benjamin Anderson, and Jozef de Vries

O'REILLY®

Building a Data and AI Platform with PostgreSQL
by Tom Taulli, Benjamin Anderson, and Jozef de Vries

Published by O'Reilly Media, Inc., 141 Stony Circle, Suite 195, Santa Rosa, CA 95401.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (*https://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: Andy Kwan
Development Editor: Gary O'Brien
Production Editor: Elizabeth Faerm
Copyeditor: Vanessa Moore
Proofreader: Rachel Rossi

Cover Designer: Susan Brown
Cover Illustrator: Susan Brown
Interior Designer: David Futato
Interior Illustrator: Kate Dullea

December 2025: First Edition

Revision History for the First Edition
2025-12-12: First Release

979-8-341-62152-7

[LSI]

With gratitude to all the contributors of this book, to the decades of investment in Postgres from the global community, and to Tatum Pollard, whose editorial guidance and support were key for this collection.

Table of Contents

CHAPTER 1

Building Competitive Advantage with Sovereign AI Data Platforms

Look around at your own life experiences today, and you'll likely find many moments when generative AI (GenAI) agents are touching your life—ChatGPT, Claude, Gemini, Perplexity, Cursor, ElevenLabs, just to name a few. Then there are the times we interact with AI without realizing it: resolving customer service issues through a chatbot, receiving personalized ads curated just for us, or getting shopping recommendations that seem to read our minds. Imagine speaking only with these AI agents or these agents working only with each other. This new world reality is far closer than we realize, and it needs a new way of thinking and behaving.

But these examples barely scratch the surface. GenAI and agentic AI are not just another digital wave. We are now experiencing a fundamental shift in how organizations operate, create value, and compete. It's a natural extension to the digital-first world that dominates Gen Z and the way they live. For example, Gen Z "zoomers" already make the majority of their purchases online and live through their social media. Highly individualized AI will feed levels of personalization that will dominate a generation. The youngest adult generation now has over $44 billion in direct buying power and is expected to control trillions within the next decade. For them, AI is not optional. It is embedded in daily life and with those around them at work, socially, and in their zeitgeist.

The pace of innovation is staggering. AI models now require up to 100 times more data than traditional applications. Enterprises face rising expectations for instant, personalized, and predictive experiences. And yet, despite these pressures, many remain stuck in the experimental phase, unable to scale AI initiatives from pilots to production. Their data remains fragmented, trapped in silos, and their organizational strategies remain built for an era when digital transformation could take fifteen years. GenAI will likely reshape industries within one third of that time.

To navigate this change, enterprises need more than tools or experiments. They need a clear-cut strategy built on three conditions for success:

Invest in the right database architecture for growth and adaptability.
: AI thrives on large, high-quality datasets that are organized, secure, and accessible in real time. PostgreSQL has emerged as a critical enabler for this, with its extensibility, ability to integrate AI frameworks such as TensorFlow and PyTorch, and support for modern AI workloads through pgvector.

Choose a framework for decisions and investments that builds on a platform, not silos.
: The days of isolated AI projects are over. The winners will be those who treat AI and data as a platform. It's a unified, sovereign asset that powers intelligent systems across every line of business, adapting continuously through feedback loops and real-time learning.

Build a clean, compliant, secure, and sovereign environment for sustained and differentiated success.
: This requires governance, ethics, transparency, and a modern architecture that operates seamlessly across cloud and on-premise environments. Without this, AI remains risky and unscalable, increasing technical debt instead of solving it.

Before exploring each condition in depth, this chapter sets the context. You will see why traditional approaches to digital transformation are no longer sufficient and can even undermine GenAI initiatives.

Rethinking Digital Transformation for the GenAI Era

Digital transformation has unfolded in waves. Wave I focused on digitizing core operations and workflows, bringing processes online to improve efficiency. Wave II introduced mobile, cloud, and analytics capabilities, enabling greater scalability and flexibility. Wave III leveraged automation and advanced analytics to optimize business models and drive data-driven decision-making across the enterprise.

Each of these waves built upon the previous one, layering new capabilities over existing systems. But they also shared a common trait: they took years to reach maturity and impact. Strategies were linear, adoption was gradual, and organizational change management could span entire business cycles.

Today, however, we stand in a much different era. GenAI is not just another wave in this sequence. As Michael Gale, *Wall Street Journal* best-selling author of *The Digital Helix*, notes:

> The speed, depth and nature of the GenAI revolution puts the first, second and third waves of digital transformation to shame. Those revolutions…almost look quaint and polite compared to the voracious speed and power of agentic and GenAI. In hindsight, digital transformation looks like adding horses to a carriage, whereas GenAI and agentic in particular looks and feels like adding brake horsepower in the thousands.[1]

GenAI connects actions to outcomes with unprecedented speed. Agentic scales businesses in whole new ways. The combination of the two is fuel for transformative change. Unlike prior digital initiatives that incrementally improved workflows, GenAI redefines them entirely. It creates intelligent systems capable of autonomous decisions, adaptive learning, and real-time personalization. The shift is from static, disconnected systems to centralized, intelligent, dynamic platforms that can unify, analyze, and activate data across clouds and geographies.

1 Gale, Michael, and Chris Aarons. *The Digital Helix: Transforming Your Organization's DNA to Thrive in the Digital Age*. Austin, TX: Greenleaf Book Group, 2017.

This isn't just a technical evolution—it's a strategic imperative. Agentic and GenAI will touch every aspect of work and life, from customer experiences to product innovation and internal enterprise operations. It is becoming a platform for success in both front and back office operations for governments, commercial enterprises, and global organizations. As a result, GenAI and data are fast becoming sovereign assets, essential not only for competitiveness but for national and economic security.

What this means is that data has become even more important. How it is architected and managed will define the winners in the GenAI and agentic era.

The Data Challenge

Despite the promise of AI and data-driven transformation, enterprises continue to face fundamental obstacles that stall progress. The challenges are systemic, pervasive, and growing in complexity. Consider the realities confronting organizations today:

Enormous data volumes
: Amazingly, 90% of the world's data was generated in the past two years.

Disparate data systems
: Persistent silos and clunky data pipelines make it difficult to learn, observe, and act on data.

Legacy data infrastructure
: Older commercial databases limit innovation and lack integration with modern cloud-native stacks. Migrating to open source as an organizational standard can create hurdles for security, compliance, and performance.

Many different users
: A growing number of users—developers, data scientists, line-of-business (LOB) users—now need access to data.

General lack of understanding of data
: Gartner's 2024 CDAO survey showed that poor data literacy was among the top five challenges for data and analytics success. In addition, there was 2021 research from Forrester that revealed that only 27% of customer data was used for analytics.

Despite these challenges, enterprises continue to pour enormous resources into their data and AI initiatives. According to International Data Corporation (IDC), global spending on technology directly supporting AI is forecast to climb from $237 billion in 2023 to $337 billion by 2025, with investments spanning compute, storage, and networking. Meanwhile, cloud services spending is on track to exceed $1.3 trillion within the same period.

Yet data from DATAVERSITY reveals that 68% of enterprises now rank data silos as their leading concern in 2025, up 7% from the previous year. Gartner Research Analysts paint a similar picture, asserting that 80% of businesses admit their divisions continue to operate in isolation, creating fragmented data environments that undermine AI's transformative potential.

In an AI-driven world, fragmented and siloed data is no longer just an operational inefficiency. It threatens competitiveness, erodes innovation, and in some cases, poses existential risks. Strategic decisions are only as sound as the data underpinning them. When vital information remains locked within disconnected systems, enterprises lose the agility, intelligence, and responsiveness needed to navigate today's volatile markets.

This is precisely why the idea of sovereign AI data platforms is moving from being optional to a strategic necessity. These platforms unify and govern data—both structured and unstructured—ensuring it remains secure, compliant, and under local control no matter where it is accessed. They form the backbone that allows AI to operate effectively at scale, enabling enterprises to drive innovation without sacrificing performance, compliance, or sovereignty.

Moving Beyond Pilots: Embracing the GenAI and Agentic Platform Mentality

Despite growing urgency, many organizations remain stuck in pilot mode, testing GenAI in isolated use cases or departmental silos. In September 2025, EnterpriseDB (EDB) released its *Sovereignty Matters* research on AI and GenAI patterns across 13 countries. It showed that while the days of pure experimentation are over for most enterprises, the most advanced enterprises (13%) have barely 11 GenAI areas out of 15 tested and within mainstream production, as opposed to still being in the experimental phase. Of

these enterprises, 49% are barely at the one-third mainstream and two-thirds experimentation phase.

This research also shows how AI and data are still not fundamentally connected in the minds of senior executives. Regionally, between 21% (India) and 30% (the US, Saudi Arabia, and UAE) see the need to house both together now (i.e., data gravity in Figure 1-1). In three years, this goes to a high of 63% (Saudi Arabia and UAE) and a low of 51% (India). The desire to become their own AI and data platforms shows aspiration with a 50% compound annual growth rate (CAGR) over the next three years, from a low of 25% in Italy now to over 95% in all countries three years from now. However, for many this symbiotic development will remain unconnected and underutilized unless they get their data and their AI working together, side by side.

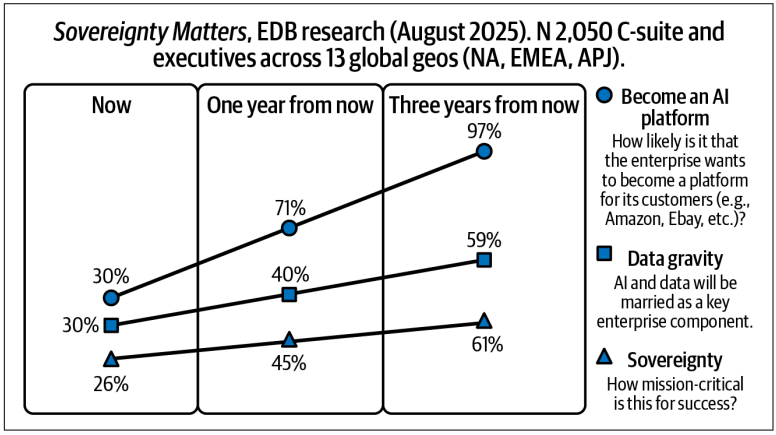

Figure 1-1. Enterprises expect sharp growth in AI platform adoption, data gravity, and sovereignty

"We're moving into an AI-first, sovereign-first world. The reason for that is we are seeing this explosion of AI applications. And because of that explosion, there's this need for a platform that can support these applications. So there is a responsibility around the security, compliance, and regulatory requirements of these applications," Kevin Dallas, CEO of EDB, told NYSE TV in May 2025. "The most successful enterprises have converged their aspiration to become their own AI and data platforms with the mission-critical imperatives to have their data and AI in their sovereign control."

This same EDB research was modeled on 15,000 simulations with 2,050 executives. The results showed that those organizations that have successfully married their AI and data—seeing sovereignty over both and building their own AI and data platforms—are 5.5 times more likely to be generating substantial ROI than the rest. In other words, the 13% that are becoming their own sovereign data and AI platforms are outperforming the other 87% by five to one. This is why platforming is essential for success. It's your data, it's your AI, and both should be working together for you.

A *platform mentality* is what differentiates fleeting AI experiments from enterprise-wide transformation. A platform is not merely a set of tools or a technology stack; it is an integrated architecture that combines AI, data, governance, and workflows into a unified system capable of scaling intelligence across every function.

The research behind *The Digital Helix* found that successful digital transformations consistently depended on three executive leadership traits: acting as explorers, seeing the world in themes and streams of data, and crafting strategies that were always one step ahead. These capabilities are essential for platform thinking, where leaders must envision AI not as an isolated app but as a foundational layer for reinventing products, services, and operations in real time.

Recent Accenture research in 2024 revealed that only 13% of executives feel they have the right digital capabilities to deliver on GenAI's potential.[2] Their concerns include technical debt, talent shortages, and the need for sustained innovation spending. However, these barriers are symptoms of an outdated project mindset. GenAI is not a static technology. It is a living, evolving system that thrives on continuous feedback loops and the culture of continuous integration and continuous deployment (CI/CD). These are the characteristics of a platform approach.

Organizations that continue to treat AI as a set of pilots will remain trapped in a cycle of limited wins and mounting technical debt. Those that embrace AI and data as an integrated platform will unlock compounding returns, adaptability, and defensibility that isolated projects cannot achieve.

2 Accenture. *Reinventing with a Digital Core*. July 16, 2024. *https://oreil.ly/ynUKr*.

With the platform mindset in place, the next step is execution. This begins by focusing on the three foundational conditions introduced at the beginning of this chapter.

Condition One: Invest in the Right Database Architecture for Growth and Adaptability

The marriage of GenAI and data is at the core of the intelligent systems factory idea, where everything is touched, created, or even has its *value revealed* using GenAI.

What you choose as your sovereign AI and data platform will be with you for years. So you need a platform that will grow with your enterprise. It's like the choice between hydrogen and electric power systems for vehicles in the late 1990s. There was no infrastructure for hydrogen success, little buy-in from ecosystem partners, and a limited universe focusing on delivering intellectual property. The same could be said for the Betamax and VHS wars of the 1970s and 1980s. Even though VHS was a less capable format, it was more affordable and had longer recording times. It also had much more open licensing terms. In the end, VHS would dominate the market and fuel the home video industry.

Getting this type of strategic decision right is not a guarantee of success, but getting it wrong could be debilitating. Something similar is happening with the GenAI opportunity—that is, it's about the database.

Of course, many vendors are vying for the opportunity. Yet the one that holds the most promise is PostgreSQL. It is the only database technology showing exponential growth in its usage. While 15% or more use it now for AI and fine-tuning, 42% or more will consider it for their next AI project, such as for GenAI development and deployment.[3] No other database shows a growth of over 300% in future consideration, as most run at best at 85% in growth, and there are few of these. Choosing the growing and most popular database platform will set you up for success. Those using PostgreSQL for AI-related development are significantly less concerned about technical

3 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. *https://www.enterprisedb.com/sovereignty-matters*.

debt—the key red flag in the Accenture research—than any other user base by 30%.

GenAI requires a sense of exploration. Existing processes might be a barrier to this level of innovation. However, PostgreSQL users for AI are one-third less concerned about the lack of processes to succeed than any other database management platform.

PostgreSQL has some key technological advantages that naturally mix with the needs for GenAI development and platform creation:

- PostgreSQL has *robust support for JSON and JSONB* so it can handle rich data that sits at the heart of GenAI development, deployment, and long-term success. This is especially the case as models get more and more advanced and can infuse semi-structured data including embeddings, model parameter changes, or new configuration files. Imagine building a copilot or agent that you want to be more intelligent and capable of detailed personalization. Over time, they will continue differentiating for users and/or customers.
- The idea of *extensibility* is vital. Ideas such as semantic search are essential for effective GenAI design. It's what drives personalization and the associated upsides in value for the data. This means users or developers can define their own functions, data types, and operators. It's like having a dictionary with you that can cover any possible conversation in nearly any style or language.
- GenAI requires *partitioning, indexing, and an efficient ability to query the large datasets* critical for GenAI models. Without it, the learning curve and depth become too painful to deliver, and the associated tech debt increases. This is especially true with time-series data.
- The *ability to seamlessly integrate with various programming languages* (e.g., Python, Java, and C++) as well as libraries and heavily used AI frameworks (e.g., TensorFlow), also means you will have wider access to skilled labor than other environments. PostgreSQL can also serve as a backend for feature stores, providing storage for preprocessed data used by AI models. This is part of the living ecosystem of capabilities that are important for GenAI success.

- GenAI applications inherently rely on *high data integrity and consistency*. PostgreSQL's ACID (atomicity, consistency, isolation, and durability) compliance ensures reliable transactions and data accuracy.
- PostgreSQL *supports advanced SQL features*, including common table expressions (CTEs), window functions, and full-text search, which are useful for data exploration and preprocessing in AI workflows.
- PostgreSQL has a *vast and active open source community*, contributing tools and extensions relevant to AI use cases. EDB is the leading company for investing in the complete health of PostgreSQL as a current and future-ready platform, with its contributions to the core codes and how it supports the ecosystem, nurtures PostgreSQL with components such as patching, and augments its capabilities for enterprise-grade performance. The community is constantly adding extensions such as PostGIS (for geospatial data) and HyperLogLog (for approximate distinct counting) to expand its utility for specific AI domains. No other open source database or other database does this.
- Cost control is going to be a perceived (and possibly actual) barrier to widespread experimentation and adoption. A 2022 Gartner study reported that only 54% of AI models move from design and simulation to production. This means everything that gets to production needs to cover the cost of previous failed models, too. PostgreSQL helps to address this because there are no licensing fees. Rather, there are only costs where needed for enterprise-grade support and associated tools.

MERMAID: A Case Study in Open Source Data Platforms Driving Real-World Impact

Marine Ecological Research Management Aid (MERMAID) shows how a sovereign AI data platform can transform outcomes. Developed by the Wildlife Conservation Society (WCS), MERMAID uses PostgreSQL to streamline coral reef monitoring, allowing researchers worldwide to collect, standardize, and analyze reef health data in real time. Previously, scientists including Dr. Emily Darling struggled with fragmented spreadsheets, delaying critical conservation decisions by years.

For example, when Dr. Darling discovered resilient coral reefs in northern Mozambique in 2006, it took six years to merge inconsistent global datasets to confirm similar climate refugia elsewhere—during which time two more mass bleaching events struck. MERMAID solved this by creating a standardized, PostgreSQL-based platform where data is uniformly formatted and instantly usable, boosting scientific productivity by three to five times:

> And so that's why we created MERMAID, as a way to have standardized and normalized data from the ways that coral reef scientists are studying coral reefs underwater. It's really a way to start and leave spreadsheets behind that are difficult to analyze in real time and bring coral reef analysis into the world of modern software and technology.
>
> — Dr. Emily Darling

Today, MERMAID supports over 20 million coral observations from 3,200 users across 51 countries. It exemplifies how PostgreSQL's extensibility, scalability, and open source foundation enable organizations to move beyond fragmented legacy systems. Like MERMAID's impact on coral reef conservation, enterprises adopting sovereign AI data platforms can unlock real-time insights, accelerate decisions, and gain a competitive advantage.

Condition Two: Choose the Right Framework for Decisions and Investments That Build on a Platform, Not Silos

If we go back to the original waves of digital transformation, we can see that the most successful enterprises viewed digital as the core of their processes. As noted in *The Digital Helix,* in 2013, the Economist Intelligence Unit (EIU) measured that a mere 16% had seen it and were delivering. By 2019, the number had grown to 27%.[4] Those that failed to deliver did so because they didn't gravitate to how to make digital a core for everyone, from the back office to the front office.

Three key variables help define a core strategy,[5] and they equally apply to GenAI as an intelligent data platform for this revolution:

4 Gale, Michael, and Chris Aarons. *The Digital Helix: Transforming Your Organization's DNA to Thrive in the Digital Age*. Austin, TX: Greenleaf Book Group, 2017.

5 Gale and Aarons. *The Digital Helix*.

- Your leaders need to be *GenAI explorers for the organization*. Their appetite for curiosity, cross learning, talking about, and enabling repeatable executions of GenAI to deliver the energy for change is vital. You can see this with the CEOs at companies such as NVIDIA, Salesforce, and Microsoft. Each asserts the power of GenAI as a platform to change the very nature of their industries. For this to work, executives need a simple framework and should be able to ask teams to plot where a GenAI idea can create or reveal value.
- Strategy in an ever-evolving world is rarely aimed at a destination but focused on the next few steps ahead. It's like a game of chess, where the permutations of moves are seemingly endless. The rapid speed of GenAI and the near instant measurable returns mean that there are often too many choices for the next project. A GenAI platform gives you a chessboard to play on where your data and GenAI agents, copilots, or other programs operate in parallel or in sequence. You cannot do this without a common framework or process, so building a platform becomes an organizing principle for how you design, react and learn, and then build further. José Raúl Capablanca, a Cuban chess master, was able to play 80 chess games in parallel because he took the idea of playing on a standard platform and worked only one or two steps ahead for each. GenAI and platform building will require the same approach. The idea of constant learning through experimentation and feedback loops—that is, the Capablanca approach to chess—drives success. Executives need to propagate this idea of learning as a strategic advantage. Most modern data development is achieved in a CI/CD manner; strategy should be, too.
- The world is a collection of themes, streams, and data. Imagine how much data your organization processes today. By 2028, the global volume of data created, captured, copied, and consumed is forecast to reach approximately 394 zettabytes, according to Statista's global data creation forecast. This underscores the idea that the world is increasingly structured as interwoven data, themes, and streams—a paradigm reflected in systems such as GDELT, which maps global events, themes, and actors in real time.

Condition Three: Build an Environment That Is Surgically Clean, Compliant, Secure, and Sovereign for Sustained and Differentiated Success

There are a number of well-understood adages about the core components for AI success. A state-of-the-art system needs powerful infrastructure, such as GPUs. Data wrangling is critical but usually does not get enough attention. And yes, models need to be monitored on an ongoing basis, as they can easily degrade.

Agentic and GenAI are no different when it comes to adages. This is why our previous two conditions have had larger coverage: they are more nuanced and owe their heritage to practitioners doing this work.

Now, let's take a look at some of the GenAI and agentic adages:

Adage 1: High-quality data is both an injection and a withdrawal moment.

> It starts with the essential need for high-quality data, or we get ghost models or false positives or negatives, which are amplified as GenAI models mature, diffuse, and increasingly dominate the way we work and play. Given the need to mix diverse datasets that reflect real-world experiences and applications, data quality becomes increasingly more important if parts of these models and interactions are taken out of their core purpose. For example, imagine using customer service AI agents' data in the wrong manner for new product development. You might assume an issue is with the product when it could simply be the documentation. There is a need to remove these types of inherent and nonintentional issues of bias.
>
> This requires new data skills and discipline for management at all levels. Moreover, data needs to be well organized and architected to be flexible. It should look less like a traditional database and more like an ACID canvas on which the data engineers, scientists, and application developers can draw value with the right frameworks for innovation, compliance, and transparency.
>
> PostgreSQL was born for this moment, especially for the increasingly diffuse range of experts (data scientists, application developers, new data infrastructure management) that will need to access, learn, and work with these systems. Skills will need to be constantly enhanced as new tools, new processes, and

ways to store and use the models become common. Having the right quality controls with agility is a challenge at the center of building a reliable and functional GenAI platform. Open source, enterprise-grade PostgreSQL checks each of these boxes.

Adage 2: You have to get past the monolithic debate of cloud versus on-premise.

Where you build the models may not be where you enact them. Your AI and data infrastructure needs to modernize past monolithic—for both cloud or on-premise. The paradox of agentic and GenAI is the enormous consumption of processing power, often best delivered in the cloud (or clouds) because of GPU needs, coupled with the tension of needing to protect and serve the models in real time in a sovereign and secure manner.

The scale of the costs, though, will increasingly cause a shift in a lot of processing and usage to a more balanced view of a hybrid-first world. This means your infrastructure needs to be robust to handle data securely where, when, and how it is needed and can be used. It should not be about the first order of priority (maybe a chatbot), but also across other themes and streams of data and applications that will want to use the data and models. Models need to be trained and deployed to be valuable, so siloing that investment is not the solution.

The ability to integrate this learning into current systems, from workflows to platforms to tools, is vital. Otherwise, we will have a new generation of AI systems with no scale or speed. Common APIs become an increasingly important prerogative for everybody, so the role of the chief strategy officer (CSO) will only increase in importance. This also makes core imperatives of the ability to monitor—whether ongoing, real time, or in parallel systems—and optimize in real time.

Adage 3: Ethics, learning, and iteration require a new type of decision-making for organizations that have traditionally isolated these as central functions.

We are all going to need to take responsibility for how we think about, orchestrate, and collaborate with these GenAI models. This is going to demand new mechanics to address issues including bias, transparency, dynamic administration and access rights, and accountability. User trust and adoption will be driven by the combination of clear rules, guidelines, and

procedures, as well as the power to act in the moment to resolve or avoid potential traps. As we increasingly lean into experimentation and strategy, there will need to be proactive feedback and ongoing training to ensure an enterprise is doing the right thing for its customers, employees, ecosystem partners, and management.

Adage 4: Solving for the triptych of innovation, agility, and cost efficiency has to drive how we build infrastructure.

Mixing the priorities for agility, innovation, performance, and cost efficiency may sound like a Gordian Knot. However, we know from initial research that the C-suite expects GenAI should deliver all three. As models increasingly handle more and more complex and integrated datasets, organizations will have to measure both short-term ROI potential in simulations and the ROI for models in production. PostgreSQL, with its extensive community in AI development and overall momentum, should be an essential database platform for helping with these variables. Without this ability, organizations are likely to transfer legacy technical debt challenges from one generation to the next.

Recent research shows that those using PostgreSQL for AI fine-tuning are more optimistic about solving technical debt issues than those using any other DB environment.[6] PostgreSQL's ACID capabilities enable a level of scalability and flexibility not present in other data platforms.

The Agentic Future

The next phase of digital transformation will be about systems interacting with other systems. It's an agent-to-agent (A2A) world, where autonomous agents negotiate and transact at speeds far beyond human capabilities. This new model of computing will reshape how enterprises make decisions, deliver value, and orchestrate outcomes. Once organizations see the singular power of GenAI within functions and then across functions, they will naturally see the collective power of their GenAI and agentics as an interconnected ecosystem.

6 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. *https://www.enterprisedb.com/sovereignty-matters*.

The role of the database and data layers will also fundamentally change. It will become about coordinating autonomy, trust, and intelligence. This means executives will need to think of their database strategy as a strategic foundation for sovereignty, resilience, and growth.

This transition requires rethinking leadership as well. No longer is software simply a helpful tool. This technology needs to be viewed as a coarchitect of organizational design. Data turns into a dialogue, governance becomes continuous, and autonomy is the driver for productivity and efficiency.

While this promises to be transformative, there are major challenges. A system must balance speed and intelligence with business priorities and regulatory constraints.

True agentic AI is still emerging. Yet just within the year it has taken us to write this book, the technology has advanced so rapidly, we would be remiss not to include this discussion. Already, emerging research shows a major split between leaders and laggards. As we noted earlier in the chapter, only 13% of enterprises are capturing outsized returns from agentic and GenAI initiatives.[7] These organizations create five times the value of peers and deploy applications at twice the scale.

Why is this so? It's about sovereignty. By designing systems where data and AI are always available but always governed, they turn isolated use cases into interconnected "agentic flywheels." Supply chains can become self-healing, compliance processes can run autonomously, and marketing systems can adjust dynamically to local conditions.

But without sovereignty at its core, agentic AI can easily spin out of control. After all, it has the ability to make decisions and take actions. This is why enterprises must design platforms with machine-readable guardrails. These are rules that encode business goals, compliance standards, and ethical boundaries.

However, regulation is rarely black and white. That's why the future depends on hybrid intelligence. This is where agents deliver with

7 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. *https://www.enterprisedb.com/sovereignty-matters*.

speed and efficiency, paired with human oversight to manage ambiguity and critical edge cases.

Global organizations face added complexity as agents operate across political and geographical borders where laws differ. Jurisdiction-aware design, along with verifiable digital identities for agents, will be critical to maintain trust and accountability.

Let's look at two examples that highlight what sovereignty looks like in action:

- JPMorganChase has built a sovereign-by-design system called LLM Suite. It leverages various large language models (LLMs) for fraud detection, research, and document summarization for 200,000 employees. As for the data, it's kept in tightly defined sovereign zones under internal governance structures, such as an architecture review board.
- Workday's Illuminate platform allows AI agents to work across HR, finance, and planning. They do this with an orchestration system called the Agent System of Record (ASR), which works with Workday and third-party agents. By treating agents like digital employees—with defined roles, access controls, and audit trails—the ASR ensures their actions remain compliant, governed, and aligned with enterprise policies.

What these cases reveal is that leaders no longer treat AI as a utility. They design agents as coarchitects of sovereign, compliant, self-operating enterprises. The next era of digital transformation will belong to those who master sovereignty in an agent-to-agent world, where autonomy becomes not a liability, but a force multiplier for resilience, innovation, and progress.

Conclusion

An open data and AI platform is essential for success with GenAI and this agent-driven future. The three conditions for success need to be equally focused on your sovereign AI and data platforms to thrive and not become siloed or have ceilings placed on them.

Condition one gives you the freedom and security to thrive and break free from old legacy thinking. Condition two is critical to feed the right decision-making and design across the organization.

Condition three is necessary but not sufficient for success unless you include conditions one and two.

Again, GenAI and agentic AI are living systems that need a platform designed for success as they evolve. You can see it as Capablanca's handling of chess. The ability to respond and react only a few moves ahead of you is vital, as this world is a rapid learn-and-respond environment, and not one where plans can be set and left alone.

PostgreSQL at enterprise grade gives you an open source platform that has low cost, ecosystem support, and agility. These satisfy the first condition. The same is the case for the second condition. This is because of PostgreSQL's ability to be scalable as you learn, adjust, adapt, and protect your sovereign data and AI, regardless of the environment. If you want your business to be an intelligent systems company with your own AI and data platform at its core, then the choice is fairly obvious.

CHAPTER 2

Why Data Decides AI's Success

When it comes to deploying AI in the enterprise, much of the attention is focused on cutting-edge models and flashy applications. They dominate the headlines. But there is something critically important that gets short shrift: data. Even when it's mentioned, it seems more like an afterthought or something where you check boxes.

Yet, without high-quality data backed up with a solid strategy, any AI implementation is doomed to fail. Data is the foundation of successful AI. It allows for better decision-making, accurate content, and effective searches, just to name a few. A successful AI strategy must be rooted in a deep understanding of data, not just where it's coming from but how it's processed, stored, and continuously optimized.

Consider a survey from S&P Global Market Intelligence and WEKA. According to more than 1,500 AI practitioners and leaders, the biggest challenge for AI deployments is storage and data management. By comparison, 26% of the respondents pointed to computing resources such as access to GPUs (graphics processing units), and 23% to security. This survey is not an outlier, either. Various others back up this conclusion.

In this chapter, we will look at the landscape of AI, but with a focus on the importance of data and its new lexicon, which is essential to this conversation.

Understanding the Different Modes of AI

At a conference in 1955 in Dartmouth, Stanford professor John McCarthy coined the term "artificial intelligence". He described it as "how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves."

Since then, the definition of AI has expanded to include many types of algorithms, models, and systems, such as machine learning (ML), deep learning, reinforcement learning, natural language processing (NLP), computer vision, and GenAI.

There are many ways to classify AI technologies, depending on factors such as whether to emphasize technical features, business outcomes, or industry use cases. For this book, we focus on three main categories: *descriptive AI*, *predictive AI*, and *GenAI*. The reason for this structure is that it aligns most directly with how enterprises evaluate, deploy, and scale the technology. It's a practical way to cut through the hype and clarify where each type of AI delivers value (see Table 2-1).

Table 2-1. Categories of AI

Type of AI	Description and example
Descriptive AI	• Analyzes historical data to understand what happened and why • Goes beyond business intelligence by using ML/NLP to find patterns in structured and unstructured data • Automates insight generation • Example: finds sales decline linked to customer complaints
Predictive AI	• Uses ML to forecast future outcomes • Requires strong data infrastructure (e.g., data lake) • Works best in real-time scenarios • Example: predicts hospital readmissions to enable early care
Generative AI	• Creates new content (e.g., text, images, audio, code) • Powered by LLMs trained on massive datasets • Useful for personalization and creativity • Example: drafts personalized marketing emails to boost engagement

Descriptive AI

Descriptive AI, which is similar to business intelligence (BI), is about analyzing historical data to make better decisions. It focuses on what has happened and understanding why. A user can query a BI system for many applications, whether to identify trends in sales, analyze inventory levels, or find customer insights.

Descriptive AI and BI both aim to analyze data and provide insights, but they differ in methodology and capabilities. Traditional BI relies on predefined queries, dashboards, and reports to summarize structured data. These tools often require manual setup and intervention. Descriptive AI, on the other hand, leverages ML and NLP to analyze both structured and unstructured data dynamically. These techniques help uncover deeper patterns and anomalies.

For example, a retail company using BI might generate a static report showing monthly sales by region, while a descriptive AI system could analyze the same dataset alongside customer reviews to identify that sales in a specific region are declining due to complaints about delayed deliveries.

This ability to integrate diverse data types and automate insight generation distinguishes descriptive AI from traditional BI.

Predictive AI

Predictive AI leverages ML models to generate predictions. ML is a subset of AI that uses statistical methods based on training on large datasets. This allows for learning about the underlying patterns, which can help generate accurate predictions. For this, it's critical to have a modern data infrastructure, such as a data lake or data lakehouse.

Generally, the most impactful use cases are where the predictive AI is processed in real time. For example, predictive AI is widely used in healthcare to anticipate patient needs and improve outcomes. A hospital might use predictive AI to analyze patient records, lab results, and historical data to predict the likelihood of a patient being readmitted within 30 days of discharge. By identifying high-risk patients, the system allows healthcare providers to intervene early with personalized follow-up plans, such as scheduling additional check-ups or offering post-discharge support. This proactive

approach improves patient care while reducing readmission rates and costs.

Generative AI

Generative AI is based on an LLM and trained on enormous amounts of information. In the case of LLMs from companies such as OpenAI, Anthropic, and Google, the data includes much of the publicly available information from the internet as well as proprietary sources. By using sophisticated deep learning models, GenAI can create content such as text, audio, and video. In fact, it can often seem humanlike. GenAI has also proven effective in translating languages, summarizing documents, writing code, and identifying sentiment.

For example, a marketing team at a large company might use GenAI to draft personalized email campaigns for thousands of customers. The AI analyzes customer data, such as purchase history and engagement patterns, to craft tailored messages that resonate with individual preferences. Using GenAI in this way can not only save time but also enhance campaign effectiveness by delivering more relevant and engaging content. The greater relevance of these messages, in turn, can lead to improvements in customer satisfaction and conversion rates.

However, GenAI is far from a silver bullet. This is why it's important to understand the different types of AI. There also needs to be a focus on how data relates to them. We'll cover this in the following sections.

How Most Generative AI Actually Works

Before we dive into specific techniques, let's look further into the fundamentals of GenAI models. An LLM is essentially a highly advanced text completion engine. You will enter a prompt—some initial text, a question or instruction—and it processes this with the GenAI, which will generate a response or completion. That's the core interaction: input and output, prompt and completion. This straightforward mechanism, however, rests on sophisticated mechanics.

So, What Exactly Does an LLM Do?

When you give an LLM a prompt—for example, "What are the key factors to consider when expanding a business into another geography, like where the nature of language and responses can have a fundamentally different structure?"—it does not understand the content the way we do. The LLM does not have inherent knowledge. Instead, it generates a response based on patterns and relationships between words it has learned from vast amounts of data. Think of it as a huge, probabilistic system: it knows that certain words and phrases are likely to follow others, and it predicts these based on statistical relationships. But it does this at a massive scale. It's like what your smartphone's text completion engine does, as it guesses the next word.

An LLM, however, can process patterns and relationships between words across paragraphs for a document, including entire books. The key to how this works is based on the transformer model, which is a recent innovation. Members of Google Research published the framework in a paper in 2017, entitled "Attention Is All You Need". The breakthrough was the use of *attention*. For the most part, it allowed for understanding the context of data. This was done by using stacks of deep learning models to weigh the importance of words.

To see how this works, let's take a simple example. Suppose we want to process this sentence: "Mary withdrew money from the bank." The word "bank" can potentially have four meanings. But attention will evaluate the surrounding words—especially "withdrew"—to predict that "bank" means a financial institution.

When using attention with an LLM, the processing can be managed at an enormous scale. The result is that the model can seem to understand and generate content on anything.

Public Data in, Often Reasonable (but Sometimes Wrong) Data out

LLMs trained on public or semi-public data are suited to answering questions about things such as historical events, scientific concepts, or popular culture. Ask ChatGPT who won the 1966 World Cup, and it will probably come back with "England." Ask it to explain

quantum mechanics in simple terms, and it might generate a reasonable, if simplified, explanation.

But there's a hitch: these models are also susceptible to "hallucinations." That is, they sometimes make up facts or give answers that seem plausible but are false. If you ask an LLM for an explanation of complex, nuanced topics—especially ones that are not widely discussed on the internet—the response will not be grounded in fact. While it is effective with general language patterns, the LLM does not verify information or cross-reference multiple sources. It's not searching a database or running calculations. This is why an LLM's response should be taken with a grain of salt.

Why Public Knowledge Alone Isn't Enough

Many real-world applications—especially business use cases—require models to work with specialized or proprietary knowledge. A customer support chatbot, for instance, isn't very useful if it knows general facts about email or online shopping but has no idea how your company's specific product works or even what it is.

Take a customer support bot designed for a software company. This bot might need to answer questions about the latest features, troubleshoot issues unique to the company's software, or provide specific guidance that doesn't exist in public forums.

In addition to private data, real-time data is often needed to respond dynamically to current conditions. For instance, in a retail setting, GenAI can create personalized marketing content or product recommendations that reflect the latest customer interactions or seasonal demands. This ensures that outputs are both relevant and impactful.

Without contextualizing your generative models within the parameters that your business operates in, in addition to making use of real-time data, a generative model limited to public data is more of a parlor trick than a practical tool. It may sound good, but it won't meet the actual needs of the task, which helps explain why so many GenAI implementations fail.

Here's another quick example: imagine a virtual assistant at a hospital. If it only knows general medical knowledge but has no data on the hospital's protocols, personnel, or patient records, it's not going to be very useful to doctors or nurses. It may even be dangerous.

Bringing Private Data into the Model: Fine-Tuning and Context-Aware Generation

So, how do we get these models to work with private, real-time, or specialized data? There are two main approaches:

Fine-tuning
: This is where we take a pretrained model and train it further on domain-specific data. Fine-tuning customizes the model's internal parameters and weights, so it finds patterns and facts specific to the domain. However, while fine-tuning can produce highly accurate results on domain-specific tasks, it's not always practical. Fine-tuning requires significant computational resources, a lot of carefully prepared data, ongoing maintenance, and the assistance of data scientists. It's not the most flexible option either. Once a model is fine-tuned, retraining it on new data can be a time-consuming and costly process. We'll go into more depth on this in Chapter 3, but for now, just know that fine-tuning is powerful but can be clunky.

Context-aware generation
: A combination of techniques, including in-context learning and data enrichment, enables the model to use knowledge from private data sources without permanently embedding that knowledge in the model itself. Instead, this approach feeds relevant context into the model as part of the prompt, essentially giving it temporary access to the information it needs. Imagine it as giving the model a cheat sheet just before it answers a question. Unlike fine-tuning, this approach doesn't require changing the model's underlying parameters and weights. It uses the information supplied in the prompt to generate more accurate responses. Context-aware generation is fast, adaptable, and relatively inexpensive.

Retrieval-Augmented Generation

At the time of writing, retrieval-augmented generation (RAG), a type of context-aware generation, is the most common approach to bridge the gap between general-purpose language generation and timely, private, or domain-specific data. RAG has gained popularity because of its simplicity, cost-effectiveness, and adaptability to many different types of private data. With this approach, an LLM can

access and use real-time information without needing retraining or fine-tuning.

What Is Retrieval-Augmented Generation, or "RAG"?

Any AI model is only as good as the information it is trained on, and it certainly doesn't know everything. RAG solves this by providing a type of on-demand memory boost to an LLM.

The essential steps in RAG are simple in theory but require a fair bit of computational and algorithmic juggling to pull off smoothly. Here's the basic three-step process:

1. *Retrieve*: Given a user's query, the AI *application* searches through relevant databases or knowledge sources and retrieves information that seems related to the user's question.
2. *Augment*: This retrieved information is then paired with the original query in the prompt. This gives the model the additional data it needs to generate an accurate and context-aware response.
3. *Generate*: The augmented prompt is passed to the LLM, which uses this extra information to produce a response. Ideally, the model now has the context needed to answer the question with more precision and relevance.

It is important to clarify that the retrieval step involves the AI application, *not the LLM*, actively searching relevant knowledge sources. The LLM does not perform the retrieval itself. Rather, it processes the results retrieved by the AI application. Once the relevant information is fetched, the LLM generates responses by combining this external knowledge with its own pretrained understanding of language. This separation of roles ensures that the LLM has access to up-to-date and domain-specific information, allowing it to produce more accurate and contextually relevant outputs.

This retrieval-augmentation-generation pipeline enables LLMs to act as both generative language processors and knowledge retrievers. They combine their inherent fluency with access to real-time or domain-specific data.

An Example: An LLM Answering with and Without Retrieved Context

Let's take an example that illustrates the difference RAG makes. Suppose you're working for a software company, and you need GPT-4 to answer a question about your latest product release. Here's a question your customer might ask:

> *Prompt:* What's new in version 17 of EDB Postgres Advanced Server?

Without any special knowledge about the product, the LLM will likely hallucinate a plausible, but wrong, answer:

> *AI Response:* EPAS 17 was released in October 2022. Some of the notable changes and features introduced in EPAS 17 include the JSONB data type, dynamic query optimization, and the COPRINT framework.

This response is confidently written but incorrect on nearly every factual point. The LLM is simply guessing based on general patterns, and it has no way to know about the specific updates in version 17.

Now, let's apply RAG. Before asking the LLM for an answer, we'll retrieve the internal release notes, the company blog post detailing the new features, and other support documents about version 17. These retrieved snippets are then added to the prompt, with a much improved generated response:

> *AI Response:* EPAS 17 was released in November 2024. It introduced several new features, including support for the BFILE data type and the DBMS_LOB, DBMS_XMLDOM, and DBMS_ASSERT packages.

In this scenario, the LLM now has relevant, specific information about version 17, allowing it to respond in a way that's much more useful for the user.

Retrieval is where it gets complicated

The retrieval step is actually quite complicated with RAG. It requires a system capable of performing highly effective, domain-specific searches. You basically need a tailored, mini-Google that can scour your organization's data on demand. Even the definition of "your organization's data" is complicated, as we'll discuss in the following sections of this chapter. If the success of your application depends

on a private context, then low-quality retrieval is going to result in unhappy users.

Techniques for building effective retrieval systems are evolving rapidly both in industry and academia, but there are some fundamental concerns that every application needs to deal with, including search techniques, context windows, and data quality.

Search techniques

How do you find the most relevant data to send to the LLM? Search is a trillion-dollar problem, with myriad solutions, but fortunately, the transformer model can provide a fantastic approach.

It can use a technique called *vector similarity search*. The transformer model will encode the input's semantics in a list of numbers, called a *vector*. To search, the input query is encoded into a vector and compared to other encoded vectors, keeping track of the nearest ones.

To illustrate this, let's take the example of navigating along latitudinal and longitudinal lines. It's about finding proximity within a defined space. This means pinpointing a location on the Earth's surface by determining how far north/south and east/west it is. Vector similarity search is about identifying the closest data points in a multidimensional space by measuring the distance between vectors. Both methods rely on calculating relative positions to identify nearby points of interest. It's conceptually quite simple, but figuring out how to do it in real time is complex. An entire cottage industry of vector databases has sprung up to try to solve this problem efficiently, but it looks likely that general-purpose databases such as PostgreSQL will add high-powered vector search capabilities in the near future.

The emergence of vector search is a happy coincidence. The transformer model that is making search more important also gives us tools to make it easier. But it's not the full picture. More traditional methods of search, such as keyword-based search and filtering, are also essential in many applications. An example is when a user searches for "red running shoes" on an e-commerce website. The system scans product listings using the keywords "red" and "running shoes" to return relevant results, then applies filtering options such as size, brand, or price to narrow down the selection. This goes

a long way in helping the user find a suitable product. Figure 2-1 shows the workflow using similarity search.

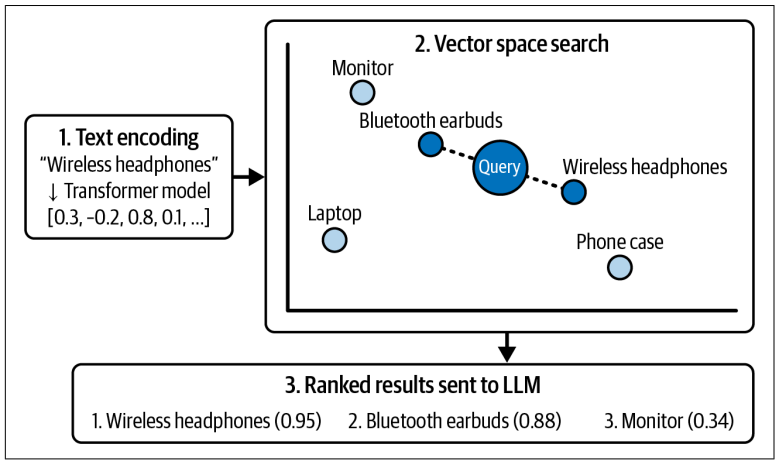

Figure 2-1. Workflow for vector similarity search

Context windows

In any LLM-based application, the full prompt must fit within the LLM's context window. It is a fixed amount of text the model can handle at one time, which includes any retrieved context, system-level instructions to the LLM, and the user's query. Measuring the context window is nontrivial. It's measured in tokens (where a token is roughly a word or part of a word or punctuation), but different LLMs use different tokenizers, so a given chunk of text may consume 100 tokens with an LLM from OpenAI but only 75 tokens with an LLM from Meta.

Many LLMs' context windows are quite small. OpenAI's GPT-3 has a context window of a mere 2,048 tokens. With such a small context window, it's critical to have a highly accurate retriever—every token counts.

Additionally, the typical context window size is highly variable and has changed as LLMs have evolved. For example, GPT-4 has a context window of up to 32,768 tokens, whereas Google's Gemini 1.5 has a context window of 2 million tokens. That variance means that you might need to implement a completely different retrieval system when a new model is released. Even worse, there is evidence that context windows are not uniformly effective. The LLM better remembers data placed either early or late in a large

prompt, whereas data in the middle tends to get lost and is less likely to be used to inform the generated response from the LLM. Even if context windows were infinite, retrieval would still be a hard problem.

In short, managing context windows is a critical part of building effective AI applications, and this means you must be thoughtful about designing an effective retriever. How much information should you retrieve? What sort of pre- or post-processing is necessary to ensure that you're in the "Goldilocks" range of context?

These are critical questions. But they also take data science experience and expertise.

Data quality

We'll discuss data quality in depth in Chapter 3, but it's worth mentioning this essential "third leg" of building high-quality RAG systems. There's no use in designing a high-quality search system and getting context windows "just right" if the underlying data in your knowledge base is of low quality—that is, messy, out-of-date, not trustworthy, or otherwise unsafe. AI applications are, ultimately, data applications. Without high-quality data, the results will inevitably be subpar.

Model Evaluation

Over 1.9 million models are available for download on Hugging Face, and thousands more are added every day. Granted, it's important to keep in mind that many of these are existing models that have been fine-tuned and adapted for specific tasks and datasets. Others may be experimental or incomplete.

Regardless, there are too many models for any enterprise to keep up with. True, you can focus on the leaderboards for LLMs on Hugging Face, but this is far from foolproof. The pace of innovation is rapid, so the lists have a high turnover rate.

This is why an enterprise will likely focus on just a few models. Otherwise, the support and maintenance will be too burdensome. After all, switching a model can be onerous because there will be a need to redo the fine-tuning, RAG systems, and embeddings. According to a survey from Lightspeed Capital, an enterprise uses an average of 2.3 LLMs.

The evaluation process for models is also challenging. It requires a balance of technical performance with practical and ethical considerations to ensure the model is robust, reliable, and suitable for its intended application. A major component, though, is the nature of human language. It's filled with ambiguity and reliant on context for meaning, which makes determining the accuracy of an LLM's output tricky. Whether a model's response is correct or not can vary depending on the personal or situational context. The inevitable hallucinations further worsen this.

Then there are the challenges with the transparency and explainability of LLMs. They often operate as "black boxes." This is where the internal processes are not easily understandable or interpretable by humans and involve training on vast amounts of data. This is even the case with open source models. The challenges range from a lack of access to the training sets to the complexity of the inner workings.

Finally, evaluating how well an LLM handles adversarial attacks or unexpected inputs is a critical area of focus. But the tools and methods for measuring these aspects are still in the early stages of development. As a result, human evaluation is often necessary to assess safety, but this approach is costly, time-consuming, and inherently subjective.

Then what about using benchmarks to evaluate LLMs? This approach can certainly be useful. But there are still some issues to consider. For example, benchmarks typically focus on specific, narrow aspects of a model's performance. Since LLMs are employed for a range of tasks, each might require a unique set of evaluation criteria, making it difficult to apply a universal standard. Moreover, benchmarks can be manipulated, as developers might artificially inflate performance metrics by training models on preexisting test data.

Reliability

Even with the rapid pace of innovation, modern LLMs still have reliability issues. According to research from Vectara, systems such as GPT-4 and GPT-4 Turbo have hallucination rates of 3% for certain use cases; and others may be as high as 27%.

Researchers and LLM developers have been focused on this problem, and there has been continued progress. For example, OpenAI's GPT-5 model uses sophisticated agentic capabilities to reflect upon and reason about its responses. *Agentic* refers to the ability of a system or AI to take actions autonomously to achieve goals. This involves capabilities including decision-making, learning from the environment, using tools, and adapting behaviors without direct human input.

But even a small hallucination rate is likely to be unacceptable for enterprises. This is why techniques such as fine-tuning models and RAG are so popular. They allow for grounding LLMs in proprietary datasets, which helps to boost accuracy.

Another technique is to use guardrails for the input. That is, certain types of input may be flagged, and the LLM will not provide an answer.

Using feedback mechanisms can be helpful as well. For example, ChatGPT has buttons for thumbs-up and thumbs-down to evaluate responses, which can be useful in gauging the effectiveness of an application.

But, for some enterprises, the decision may be to forgo high-risk use cases. This would certainly be in situations where there are customer-facing interactions. Instead, a company may want to focus on backend or internal applications.

Besides hallucinations, another major issue with AI is bias. This has been a long-standing problem because the datasets used for training often reflect society's prejudices and stereotypes. A report from UNESCO's International Research Centre on Artificial Intelligence highlights this. It showed "pervasive" bias with LLMs regarding gender and LGBTQ+ topics.

Monitoring AI systems and implementing iterative improvements, such as diverse data sampling and fairness audits, is important. These measures help mitigate bias and promote the development of more equitable AI systems.

Conclusion

In this chapter, we looked at the critical role that data plays in AI implementations and how it influences the success of AI strategies. The effectiveness of AI hinges on high-quality data, as flawed or incomplete data can undermine even the most advanced LLMs. We also explored the drawbacks of AI, including challenges such as data quality issues and the complexity of managing unstructured data. Furthermore, we examined various AI types—descriptive, predictive, and generative—and their reliance on both structured and unstructured data.

In the context of AI adoption, the importance of RAG strategies also stands out. RAG can greatly enhance AI models by combining real-time, relevant external knowledge with pretrained information, making it a powerful tool for enterprises seeking to improve the accuracy and adaptability of their AI applications. As enterprises navigate AI adoption, understanding the data's origins, management, and continuous optimization, along with leveraging RAG, is key to unlocking AI's full potential.

In this chapter, we covered several common traps in the AI landscape, including:

- Overhyping models while undervaluing the role of data
- Treating descriptive, predictive, and GenAI as interchangeable
- Assuming public LLMs are sufficient without grounding them in proprietary or real-time data
- Building RAG pipelines without strong retrieval, domain tuning, or context window management
- Choosing models based only on benchmarks or hype cycles
- Rushing unproven AI systems into customer-facing use cases and risking brand damage

When deploying AI in the enterprise, there are 10 key questions that leaders should ask their teams:

1. Do we have a shared understanding of the difference between descriptive, predictive, and GenAI—and where each adds value?
2. How confident are we in the quality, timeliness, and governance of our enterprise data?

3. Are we matching the right AI type to the right business problem?
4. What criteria beyond raw performance guide our model selection (e.g., explainability, adaptability, cost)?
5. How are we ensuring retrieval quality in RAG systems and managing context windows effectively?
6. What safeguards and feedback mechanisms do we have in place to minimize hallucinations and bias?
7. Which AI use cases are too high-risk for external deployment and should remain internal until systems mature?
8. How do we evaluate the trade-offs between public, proprietary, and hybrid data sources?
9. Do we have a process for monitoring bias, fairness, and compliance across AI initiatives?
10. Who is accountable for governance, evaluation, and long-term oversight of AI in our organization?

Successful AI adoption starts with clarity, understanding what type of AI you are deploying, ensuring the data foundation is strong, and matching technology to business value. Enterprises that ask these questions early avoid costly missteps and unlock reliable, scalable impact.

CHAPTER 3

The Role of Data with AI

Not all data is born in the same shape, form, and value. While AI sees data as its oxygen, the level of nutritional value that data brings to AI is not linear. Imagine a spreadsheet compared to a picture of your summer holiday with your family on a beach in northern France. These are both data but are fundamentally different in how their value can be mapped and extracted for an AI engine.

Of course, the world is awash with enormous amounts of data—and the growth rate has been staggering. The point: businesses that treat all data equally, without asking the right questions about what kind of data they are working with, risk wasting enormous amounts of time and money.

In this chapter, we'll focus on the critical importance of data. It's the fuel that powers effective AI applications to lead to positive business outcomes.

Structured and Unstructured Data

At a high level, AI systems use two types of data: structured and unstructured. Each allows for uncovering valuable insights. But it's important to understand that both types of data have distinct approaches for analysis and they can both have a major impact on AI projects.

We'll start with unstructured data, as this is generally what LLMs are trained on.

Unstructured Data

Think of unstructured data as standing on a street and writing down what all your senses perceive. Cars whizzing by. Pedestrians walking. Lights flashing. Horns honking. Categorizing all this information would be challenging, if not impossible. Think of the tens of billions of unstructured data points you could capture every day. Think of unstructured data as what all your senses could capture within just a few seconds, compared to structured data that feels and looks more like a traditional spreadsheet or database. Unstructured data now offers us a whole new world of possibilities.

This is what unstructured data is. It lacks a predefined schema or organized format. This makes it more difficult to process compared to structured data. Common examples of unstructured data include PDFs, text files, Word documents, images, videos, chat logs, emails, and social media posts.

The enterprise world is awash in unstructured data. It represents up to 90% of data compared to only 10% for structured data. This has been due to secular trends, including the following:

Mobile
: More than 7 billion people have smartphones. The average amount of data consumed is roughly 20 GB per month.

Social media
: More than 5 billion people use social media platforms to generate text, audio, and video content.

Video streaming
: This accounts for 82.5% of global internet traffic.

AI
: More and more content is AI-generated. Some studies predict that this could reach 90% of all new internet content created by 2026.

IoT (Internet-of-Things) devices
: Devices such as wearables and industrial sensors were expected to generate 80 zettabytes of data by the end of 2025.

But unstructured data is more than digital information. The vision of the paperless office is far from reality. There continues to be large amounts of data in analog or physical form, such as paper documents, blueprints, video footage, magnetic tapes, and handwritten notes.

However, in the corporate world, core software systems, such as CRMs (customer relationship management) and ERPs (enterprise resource planning), have traditionally been built to manage structured data. Being systems of record, they need high levels of accuracy. This is certainly critical. But it is still a small part of what's important.

Take the example of customer sentiment analysis. True, you will use structured data such as purchase history, complaint categories, or survey scores. United Airlines, for instance, has used social media sentiment analysis to identify recurring passenger frustrations and adjust its service strategy. Structured booking and loyalty data couldn't reveal the *why* behind declining satisfaction, but unstructured data from tweets and Facebook posts showed patterns including dissatisfaction with delays and baggage handling.

However, these root causes or deeper feelings behind customer sentiment often reside in unstructured data, such as customer reviews, social media posts, or call center transcripts. The insight gained from these forms of unstructured data is not something that can be extracted from traditional structured data. Without incorporating unstructured data, businesses miss out on valuable insights essential for unlocking the full potential of AI.

Yet unstructured data poses major challenges. The lack of consistency means that unstructured data is harder to clean, prepare, and organize for AI systems. Enterprises may need to spend considerable time and resources on tasks such as tagging, labeling, and cleansing this data.

In fact, this is becoming one of the biggest costs for AI projects. A CEO of a data labeling company has said that data preparation can account for up to 80% of an AI project's time and budget. A key reason for this is that these specialized datasets often need the help of experts for evaluation.

The sheer scale of unstructured data adds another layer of complexity. There must be modern data platforms that are designed to manage this effectively, such as data lakes or the Hadoop Distributed File System (HDFS) that allows for the storage and processing of large amounts of data across a distributed computing environment.

Despite these challenges, unstructured data is essential for the success of AI initiatives. Enterprises stand to gain significant competitive advantages by harnessing the untapped potential.

Structured Data

Structured data refers to information that is organized on a predefined schema or data model. A common format is for rows and columns, as in a relational database, spreadsheet, or CSV file. Querying languages, such as SQL (Structured Query Language), can create, read, update, and delete the data (also known as CRUD). Structured data is generally stored in relational databases, data warehouses, or transmitted through real-time feeds.

Examples of structured data include:

- Customer records with fields such as names, email addresses, and phone numbers, as well as financial transactions with details such as account numbers, transaction dates, and amounts
- Inventory management data containing product IDs, descriptions, prices, and stock levels
- In retail, loyalty program data showing frequency and product preferences; in airlines, flight schedules and ticket classes; in logistics, barcode scans for packages in transit

Structured data requires less preprocessing and preparation. Its predictable format also minimizes the chances of errors or misinterpretations, helping to improve the accuracy of the results that your AI systems produce.

It also helps that structured data is supported by many systems, such as relational databases. This makes it easier and more cost-effective to implement AI projects. There is also less need for expensive and time-consuming data migrations.

Despite the advantages, structured data has several notable drawbacks. Its rigid structure can make it difficult to adapt when changes are required. Structured data also often lacks the richness, depth, and nuances of unstructured data. This can limit the insights of an AI system.

Unlocking Insights from Unstructured Data

In practice, the type of data dictates the type of AI that is most effective. Leaders should not ask, "Can we apply AI?" but instead, "Which AI approach best fits this data and this business problem?"

A 2018 Gartner survey found that around 85% of AI projects don't succeed. Even by 2024, RAND Corporation research showed that 80% were still not succeeding. Why? A lack of understanding of what AI can do means that the wrong methods can be applied to problems.

When it comes to AI, it should be no different from any other technology project. It's about clearly identifying the problem and choosing the right solution. It's not about starting with a particular technology and forcing a fit.

Here are a few examples of how the different types of AI are best suited for a given type of data:

Natural language data
: GenAI can process text from customer reviews on ecommerce sites, emails from clients, and IT support tickets. For example, it can power chatbots that automatically respond to routine service questions, generate contract summaries for legal teams, or tailor marketing emails based on prior purchase history.

Social media data
: GenAI can analyze posts on platforms such as X, Instagram, or Reddit threads to detect trending topics or viral conversations. Retailers, for instance, might identify a spike in social media buzz around a particular sneaker style and quickly adjust inventory levels or launch targeted promotions.

Multimodal functions
: GenAI can combine text, images, audio, and video to produce compelling marketing content at scale. For example, a brand could feed product photos and campaigns into a multimodal

AI system to quickly generate promo videos, banner ads, or personalized social media creatives.

Sensor and IoT data
: Predictive AI can process sensor output from factory machines to anticipate part failures, monitor smart home energy usage to optimize consumption, or use real-time shopping container sensors to prevent supply chain delays.

Financial transaction data
: Descriptive AI can analyze streams of credit transactions to flag unusual spending patterns that may indicate fraud, review loan applicant data to improve credit risk scoring, or track trading volumes to highlight market sentiment shifts.

Geospatial and satellite data
: Descriptive AI can scan satellite images to map wildfire spread for emergency response or analyze urban sprawl to guide city infrastructure planning.

Data Quality

Ensuring data quality is crucial to the success of any AI initiative. When data is clean, organized, and relevant, AI models can perform at their best.

As AI plays an increasingly important role in decision-making and operations, the potential consequences of poor data quality—such as faulty strategies, inefficiencies, financial setbacks, and legal liability—can be quite serious. But the consequences might not be obvious at first. Instead, they can cause a gradual deterioration over time. For example, in a system that assesses creditworthiness for borrowers, flawed datasets could lead to unintended bias. This can mean that certain groups are unfairly discriminated against. Yet it may seem as if nothing is wrong.

Real-world trap: one financial services firm used outdated demographic data for credit scoring. By the time the model was applied, the data no longer reflected true customer profiles, resulting in massive loan default exposure. Data quality issues are rarely obvious at the start. They show up later as compounding business risk.

The best practices for data quality are beyond the scope of this book. But there are a few factors to keep in mind:

Relevant data and completeness
: AI systems require data that closely aligns with the problem they are addressing. For example, in healthcare, accurate patient records, such as medical history and current treatments, are critical for predicting recovery outcomes.

Coverage and inclusive representation
: The data must span a wide range of scenarios. For example, in retail, an AI model predicting customer preferences needs data from a diverse set of customers across different demographics, regions, and shopping behaviors.

Minimizing bias
: Reducing bias is essential when AI models have real-world consequences. In criminal justice, biased data could lead to unfair sentencing outcomes; in hiring, historical biases in recruitment data can perpetuate discrimination. If you can't adequately address the bias, AI might not be the right tool for the job.

Up-to-date information
: Using recent data is critical, especially when you need to make decisions quickly. Having access to the most current information can give businesses a competitive edge.

A common reason for data quality issues is the fragmentation of data across silos; that is, where different departments or systems store information separately, making it difficult to integrate and aggregate. For example, a retail company may have customer data split between sales, marketing, and inventory systems. This can lead to discrepancies in customer profiles and result in misguided targeting for marketing campaigns.

Furthermore, inconsistencies between datasets—such as varying formats, standards, or definitions—can create confusion and bias in AI outputs. Outdated data, if not regularly updated, can render AI models less effective, if not useless.

This is why you need a modern data infrastructure, such as a data warehouse, data lake, or data lakehouse.

Data warehouses, originally created as centralized repositories for structured business data during the 1980s, have undergone a significant transformation. Today, they are sophisticated, cloud-based platforms that enable the processing of diverse data types in near real time. This has expanded their usefulness from basic reporting to powering advanced analytics and AI-driven insights across industries. Modern data warehouses can manage complex queries on enormous volumes of structured data.

However, this power and flexibility comes with trade-offs. While data warehouses provide high-quality data services, such as data integration, cleansing, and transformation that ensure the data is accurate, consistent, and accessible, they are generally more expensive than alternatives. They also introduce complexity. This is especially the case with ETL (extract, transform, load) processes, which can cause delays in data availability. The rigid structure that makes warehouses efficient for certain tasks, such as consolidating large volumes of structured data from multiple sources for quick and accurate analysis, can also limit their flexibility when dealing with unstructured data. Organizations must weigh these factors and consider their needs for data consistency and accuracy, query performance, and analytical capabilities against the costs and potential limitations of implementing and maintaining a data warehouse solution.

Data lakes emerged as a response to the growing volume and variety of data that organizations needed to store and analyze. Data lakes are large-scale repositories that can ingest and store raw data in its native format, whether structured, unstructured, or semi-structured (data with some level of organization, such as JSON or XML). This flexibility allows organizations to capture and retain all types of data without the need for up-front processing or schema definitions.

The primary advantages of data lakes include their cost-effectiveness for storing large volumes of diverse data, ability to support a wide range of analytics use cases from ML to data science experiments, and agility in adapting to new data sources and types.

But data lakes also pose challenges. Without proper governance, they can become "data swamps" where data is difficult to find, understand, or trust. They also typically require more technical expertise to query effectively and lack built-in data quality controls.

Data lakehouses have become increasingly popular in recent years. They essentially combine the best features of data lakes and data warehouses. Data lakehouses offer a unified approach to handling large volumes of diverse data while allowing for advanced analytics capabilities. By merging the cost-effective, scalable storage of data lakes with the advanced analytics features and quality of service typically associated with data warehouses, data lakehouses provide a versatile solution for modern data needs. They achieve this through a metadata management layer that offers data warehouse-like capabilities on top of a data lake architecture. In addition, data lakehouse systems make use of open table formats such as Apache Iceberg and Delta Lake. They allow for seamless integration and management of both structured and unstructured data, which creates a unified layer for storage and analytics. These open formats ensure flexibility and interoperability across various tools, while the use of open source software enables organizations to customize and extend their systems without being locked into proprietary solutions and making data integration easier.

The key benefits of data lakehouses include:

- Cost-effective and scalable storage for massive data volumes
- Support for structured, semi-structured, and unstructured data
- Transactional consistency
- Fine-grained access controls
- Low-latency, interactive queries suitable for real-time dashboards

They also eliminate the need for separate ETL processes between data lakes and warehouses and allow direct ingestion of real-time data streams. This makes them particularly well suited for organizations dealing with diverse data types and requiring both storage scalability and advanced analytics. For example, a retail company could use a data lakehouse to store and analyze customer transaction data, social media interactions, and inventory information all in one place. This can enable real-time personalized marketing and efficient supply chain management.

Overall, data lakehouses represent a significant step forward in data architecture. They offer a powerful and flexible platform for managing and analyzing data.

Fine-Tuning Versus RAG

Structured and unstructured data are the "what." Fine-tuning and RAG are the "how." Once you know your data types, the next question is: how do you make them useful to models?

As discussed briefly in Chapter 2, RAG and other in-context approaches are not the only methods for making private data accessible to LLMs. Model fine-tuning, which involves taking a pretrained model, such as GPT-4, and adjusting its parameters to specialize in a particular domain, is the other. The fine-tuning process allows the model to learn from a smaller, domain-specific dataset, which can improve its accuracy and performance.

There are various fine-tuning methods, from so-called "full-parameter" fine-tuning, which adjusts the entire model, to methods such as low-rank adaptation, or LoRA, which only updates a subset of model parameters. But no matter what method is chosen, fine-tuning has significant drawbacks that need careful consideration.

First and foremost, going down the fine-tuning path will slow down the development cycle. In-context approaches such as RAG benefit from rapid feedback. From a developer's point of view, it's a very interactive process. Fine-tuning, on the other hand, is a data science–oriented approach. Training and test datasets must be built and maintained. Moreover, the retraining takes time and costs money. This can create bottlenecks in an AI application development model, which can change daily.

Additionally, and more importantly, model fine-tuning is static. There's no way for it alone to incorporate real-time data or user-specific inputs into its training. This makes it ill-suited for any application that requires continuous learning or personalization for users. From a data perspective, fine-tuning is a rather limited approach.

In truth, the best approach for mature AI applications is generally a combination of fine-tuning and in-context approaches. However, starting your AI journey with RAG or in-context learning offers a faster, more adaptable solution, while fine-tuning can be progressively introduced as the application matures and more specific, stable requirements emerge.

Costs

All of these decisions—the types of data you prioritize, how you govern them, and the methods you use to apply them—roll up into a final question every leader must ask: what does this cost, and what is the ROI?

The costs for implementing AI can add up quickly. A key factor of this is managing large volumes of data as well as the labor, infrastructure, and vendor costs in ensuring that data is available, secure, and governed. As organizations scale their AI capabilities, so do their costs.

Structured data is easier to manage due to its organized format. However, as data volumes grow, costs increase in terms of both storage and data processing. For instance, organizations may need to invest in more powerful cloud infrastructure or dedicated servers to accommodate the ever-expanding datasets. Furthermore, managing data quality, and ensuring it is accurate, consistent, and ready for analysis, can be resource intensive. This requires investment in automated tools and skilled engineers to handle the complexity of integration and maintenance.

In contrast, unstructured data introduces even more significant cost challenges. Storing and processing unstructured data typically requires scalable storage solutions such as data lakes, which can be more expensive than traditional databases due to their ability to handle diverse, large-scale datasets. Extracting value from unstructured data requires specialized AI algorithms, such as NLP or image recognition models, which require significant computational resources. Running these models often requires high-performance computing, such as GPUs, and incurs additional computing costs for both training and inference. This makes the overall cost of unstructured data management much higher than that of structured data.

Ultimately, the use of structured and unstructured data within AI systems will be a major cost driver. As data continues to grow and evolve, businesses must carefully balance the infrastructure, tools, and talent required to manage both structured and unstructured data. This will ensure long-term sustainability while keeping costs under control.

Conclusion

Data plays a central role in the success of AI applications, with both structured and unstructured data serving as the foundation for building robust, intelligent systems.

High-quality data is essential for AI systems to deliver accurate and reliable outcomes. Managing this data, whether structured or unstructured, requires data quality management, including data cleansing, integration, and continuous optimization.

While fine-tuning AI models on specific datasets can enhance their performance, the use of RAG offers an alternative and sometimes more interactive approach by augmenting models with relevant external knowledge. This helps them remain adaptable and context aware.

And finally, the management of large volumes of data, combined with the need for advanced processing tools and AI models, introduces significant cost implications. Organizations must navigate the complexities of infrastructure, talent, and technology to ensure they can handle the demands of both structured and unstructured data while balancing cost efficiency and long-term AI scalability.

In this chapter, we covered several common traps in the AI landscape, including:

- Treating structured and unstructured data as interchangeable
- Overinvesting in data lakes without governance, leading to "data swamps" or small outposts of data that are disconnected
- Assuming fine-tuning is always better than RAG (or vice versa)
- Ignoring the hidden costs of labeling and cleansing unstructured data

When considering the critical importance of data, there are 10 key questions that leaders should ask their teams:

1. Language or a shared lexicon and meanings are critical here, so it's easy to talk across business units and technical units in the design and delivery phases. If our data disappeared tomorrow, what critical business functions would stop first? What kinds of data are most critical to our business problems—structured, unstructured, or both?
2. Are we investing in cleaning and governing our data as much as we are in AI tools?
3. Where is our unstructured data today, and how are we capturing and making use of it?
4. What are the biggest risks if our data quality is poor (bias, compliance, customer trust)?
5. Are we prepared to manage the costs of storing and processing rapidly growing data volumes?
6. Do we understand the trade-offs between RAG and fine-tuning, and when to apply each?
7. Are there common bottlenecks (such as labeling or cleansing) that are slowing down AI projects?
8. Do we have a clear governance model to prevent "data swamps" or "data villages"?
9. How do we measure ROI on data initiatives, not just AI outcomes?
10. Do we have a single, trusted data foundation—spanning structured and unstructured sources—that our AI systems can continuously learn from and improve on?

There is no AI without data, and there is no business value without quality, well-governed, cost-conscious data strategies. Leaders who ask these questions early avoid the most common traps and set their AI programs up for measurable success.

CHAPTER 4

Transactional Data: The Unsung Hero of the AI Era

At the core of any enterprise software system—whether enterprise resource planning (ERP), customer relationship management (CRM), or social customer relationship management (SCRM)—is transactional data. It represents the everyday events that keep the global economy humming: sales orders, invoices, customer interactions, purchase orders, deliveries, and payments. On the surface, it looks mundane. In reality, transactional data is the single most valuable resource that most enterprises are underusing. If you are walking over your transactional data every day without properly activating it, you are walking over your competitive advantage.

Transactional data provides consistency, structure, and context, which allows for boosting efficiency, reducing manual processes, and providing for accountability and transparency. But more important, it grounds AI in reality. It's what turns impressive demos into systems that actually deliver measurable business value.

The reliance on transactional data helps to explain why the enterprise software market remains one of the largest. In 2023, global spending reached $356 billion, a 12% increase from the prior year, according to IDC. And it's not slowing down. IDC forecasts that by 2028, the market will surpass $600 billion.

The scale of transactional data within enterprises is also staggering. For a single organization, this can amount to petabytes of records—spanning decades of activity, decisions, and interactions.

To put this into perspective, here are a few snapshots from 2024 that highlight the sheer volume of transactional data that enterprises manage:

- Visa processed about 233.8 billion transactions or $639 million daily.
- Marriott hotels booked about 281.8 million room nights per year.
- Walmart store locations handle about 36.4 million transactions daily.

These are not just records. Each of these transactions is a live signal that powers personalization, supply chain optimization, and fraud detection. If fed into AI, they can drive automation and prediction at scale.

Despite the scale and complexity of enterprise software platforms, the core architecture remains remarkably unchanged. Microsoft CEO Satya Nadella boiled it down with clarity: at its heart, enterprise software is still a user interface—forms, buttons, sliders—layered over "CRUD databases with a bunch of business logic."

CRUD—short for create, read, update, and delete—represents the foundational operations of any persistent data system. It's a framework that has powered enterprise workflows for decades, and it still underpins most of the applications running global business today.

But as the world moves quickly toward AI, this model is poised for dramatic change, if not disruption. According to Nadella:

> The business logic is all going to these agents, and these agents are going to be multi repo CRUD, right? So they're not going to discriminate between what the back end is. They're going to update multiple databases, and all the logic will be in the AI tier, so to speak. And once the AI tier becomes the place where all the logic is, then people will start replacing the back ends, right?

This is certainly an intriguing—if not controversial—insight. But it highlights something crucial. That is, databases aren't going anywhere. As AI accelerates its growth and adoption, the role of transactional data becomes even more essential. The systems may

evolve, the interfaces may modernize, but the need to reliably create, read, update, and delete data at scale remains foundational.

Here's what Nadella had to say about this:

> [W]e are seeing new AI-driven data patterns emerge. If you look underneath ChatGPT or [MS] Copilot, or enterprise AI apps, you see the growth of raw storage, database services, and app platform services as these workloads scale. The number of Azure OpenAI apps running on Azure databases and Azure App Services more than doubled year over year, driving significant growth in adoption across SQL, hyperscale, and Cosmos DB.

This is the key point leaders often miss: AI thrives on unstructured data, but without transactional data it floats unanchored. Transactional signals ground AI outputs in what is actually happening across your customers, employees, and operations.

No doubt, agentic and generative AI is also producing and consuming enormous amounts of unstructured data, at a pace we've never seen before. This is especially the case with multimodal systems, which blend multiple data types into a single interface.

An eye-opening example came in March 2025, when OpenAI unveiled its latest image generation platform. CEO Sam Altman took to X with a simple but telling post (see Figure 4-1).

Figure 4-1. Sam Altman's March 2025 X post about challenges scaling for its image generation system

It's true that AI works particularly well with unstructured data. Just some of the use cases include summarization, sentiment analysis, anomaly detection, and code generation.

But amid the excitement and hype, one thing must remain clear: transactional data still matters. It is the structural backbone of AI applications, grounding models in the realities of business processes and user behavior.

A notable example of this is how ChatGPT integrates structured data into its image generation system. It binds specific attributes—such as color and shape—directly to objects, which dramatically cuts down on common mistakes. If you ask for "a red triangle next to a blue circle," that's exactly what you'll get.

So yes, the most powerful AI systems won't rely on unstructured data alone. They'll blend it with transactional signals to deliver applications that don't just impress—but drive meaningful, measurable value.

That is what we'll cover in this chapter.

The Power of Transactional Data

Today's default AI playbook is too often "throw documents into RAG." That gets you answers. But without transactional data, those answers don't translate into action. The value comes from fusing unstructured knowledge with the structured, contextual richness of transactional data.

To better understand the importance of transactional data in AI applications, let's consider a practical scenario. Suppose we want to build an internal company support bot to assist employees with matters such as IT, HR policies, and benefits.

Let's first see how this is done with a focus on unstructured data, which is a fairly standard playbook. Here's the workflow for creating the application:

Data gathering
: You start by identifying relevant internal sources including PDFs, policy documents, manuals, and FAQs. This is the raw material the bot will rely on.

Data cleaning
: Not everything is usable, of course. You'll need to vet the content for accuracy, remove outdated information, and resolve inconsistencies.

Indexing and vectorization
: The documents are then broken into smaller, manageable segments, a process known as *chunking*. Each chunk is converted into a vector embedding, which is a numerical representation of text that captures its semantic meaning and relationships between words.

RAG
: When an employee submits a question, the system transforms the query into an embedding, retrieves the most relevant content, and generates a response based on that context.

With this system, suppose an employee enters the following prompt:

> *Prompt:* What is the company's policy on remote work for employees based in California?

Here's the response:

> *AI Response:* According to the company's employee handbook, employees based in California are eligible for remote work arrangements. To formalize this, you must submit a remote work request form to your department manager for approval. Once approved, ensure compliance with all remote work guidelines outlined in the handbook, including maintaining regular communication with your team and adhering to data security protocols.

This is certainly useful. The response is grounded in the enterprise's proprietary data, which allows for higher accuracy and relevancy. Using a support bot also means that human support teams have more time to focus on more important matters.

Yet this type of generative AI application is still quite limited. What's needed is to layer in transactional data. This will allow for benefits such as personalization, task automation, and predictive analytics.

Personalization

By leveraging transactional data, a support bot goes well beyond a typical virtual assistant. Instead of just providing generic answers, it can tailor responses to the user based on their historical records,

role, or activity within the system. It is much more personalized to the user and is context aware. In a way, the support bot becomes more like a virtual colleague.

Here's what this looks like in action:

Onboarding
: The support bot doesn't just present generic HR policies; it cross-references transactional data (such as role-based access logs, past training completion rates, and location-based compliance requirements) to tailor the onboarding plan for a particular employee's role, department, and region.

Employee development
: Training will no longer be a one-size-fits-all process, which often gets disappointing results. Instead, the support bot will recommend training paths and courses that are based on an employee's preferences, background, and career goals. This will significantly improve the learning experience.

IT support
: When an employee uses the support bot for a technical issue, the response will not be boilerplate instructions. Responses will instead be based on an employee's history with the device in question, past support tickets, and current IT configurations.

Task Automation

By using transactional data, the support bot will be more than just about providing insightful responses. It will also be able to take actions. For example, an application can automate routine tasks on systems such as ERPs and CRMs.

These are some notable examples:

Leave requests
: Suppose an employee asks how much vacation time they have remaining. The support queries the ERP system, returns the balance, and follows up: "Would you like to submit a leave request?" If the answer is "yes," the support bot handles the submission, updates the system, and triggers the necessary approvals and notifications—end to end.

Expense reimbursements
: Rather than navigating a portal or filling out forms manually, the employee uploads receipts directly through the support bot. The system validates the data, sends it to the ERP, and initiates the reimbursement process.

Support ticket escalation
: When the support bot hits a wall with a complex issue, it doesn't just stop. It generates a support ticket automatically, routes it to the right team member, and provides the employee with a reference ID.

Predictive Analytics

The traditional approach for predictive analytics is to use business intelligence (BI) tools. While powerful, they have major limitations. One of the biggest is that these systems require specialized training and expertise.

But this changes when using transactional data with AI. For example, with our support bot example, any employee—not just data analysts—can tap the power of predictive analytics insights by using natural language prompts. There is no need for cryptic scripts. The support bot will then generate forecasts and recommendations on demand, unlocking data-driven decision-making across the enterprise.

Here are some sample prompts:

> *Prompt:* What are the projected sales figures for the next quarter?
>
> *Prompt:* Based on past trends, when should we expect a surge in customer inquiries?
>
> *Prompt:* What is the current churn rate? What are ways we can lower it?
>
> *Prompt:* How soon will we need to reorder product X, given current sales rates?
>
> *Prompt:* Are there any indicators of increased employee turnover in the upcoming months?
>
> *Prompt:* Considering our current pace, when can we expect to complete the development phase of the project?

The lesson: it's not about plugging AI into your systems; it's about asking what transactional data you already own that can make those AI systems smarter.

Adding Unstructured Data

In our support bot examples, we've already seen how unstructured data—specifically, the free-text content of user prompts—can be combined with structured transactional data to create smarter responses. But we can add richer context. For instance, consider sentiment analysis. Here, the bot analyzes the emotional tone hidden within a user's words—data that isn't neatly organized or labeled like transactions are. By tapping into this unstructured emotional layer, the bot can recognize signs of frustration, confusion, or urgency. If the signals turn sharply negative, it can proactively shift its approach or escalate the conversation to a human agent. Structured data tells the bot what happened; unstructured data reveals how the user feels about it. Fusing these two types of information gives the support bot a much deeper, more human-like understanding—and unlocks a powerful set of new advantages:

Real-time information retrieval
: An application can operate on demand, providing timely and contextually relevant responses.

Improved decision-making
: With more data sources, an application can provide broader insights. This can lead to more informed decisions.

Improved efficiency
: An application can intelligently automate complex tasks, reducing tedious and time-consuming manual workloads.

Democratization
: Everyone in the organization can use the application. There is no need to be trained, such as for complex skills with BI or data analytics.

Personalized user experiences
: The responses are based on interactions, feedback, and other important data. This allows for much better experiences, which help to improve user satisfaction and engagement.

Of course, recognizing the value of transactional data is only the first step. In reality, this data does not live in one neat place. It's scattered across ERPs, CRMs, HR systems, finance platforms, and dozens of other applications owned by various teams. Bringing these datasets, ensuring consistency, and making them usable for AI is no

small task. The very systems that create competitive advantage can also create silos and complexity. Which brings us to the next critical question: how do you simplify and unify this fragmented landscape?

The Data Management Landscape and the Importance of Simplification

Here's the trap many fall into: chasing shiny tools instead of activating the data they already have. Transactional data is already sitting inside the ERP, CRM, and HR systems. Layering on more niche databases without consolidation often adds cost without adding value. The smarter play is to unify, simplify, and unlock the value of what you already own.

Modern data management traces its roots to the early 1970s when Edgar F. Codd introduced the relational model—a breakthrough that ignited a wave of innovation and gave rise to franchise platforms such as IBM Db2, Oracle, MySQL, Microsoft SQL Server, and PostgreSQL.

For decades, relational databases powered the backbone of business systems. But as the digital landscape evolved—with the rise of the internet, cloud computing, and mobile—so did the nature of data. It became more varied, more voluminous, and more dynamic than traditional systems were designed to handle.

Legacy relational database management system (RDBMS) platforms began to show their limits. They were costly, rigid, and struggled to scale in a world that demanded flexibility and speed. This opened the door to a new generation of innovation—one that included NoSQL databases, purpose-built to handle the realities of modern data.

But layering on more systems—especially niche vector databases—only deepens the complexity. It drives up costs, fragments the data stack, and makes it harder to build generative AI applications and agents that are truly scalable and production ready.

This is the reality: building modern AI systems requires a database that can handle both transactional and unstructured data with equal strength. Introducing stand-alone vector databases may seem like progress—but in practice, it often introduces friction.

The smarter path? It's about consolidation. Streamline your architecture by reducing the number of disparate data systems. Platforms such as EDB Postgres® AI (EDB PG AI) offer a unified foundation that brings transactional, vector, and unstructured data together—eliminating silos and accelerating AI outcomes.

We're already seeing this trend play out across the industry. Many companies are moving quickly to simplify their data management stacks, replacing complex webs of specialized databases with unified platforms that can handle a broader range of workloads. It's not just about cost savings—it's about speed, agility, and setting up a foundation where AI projects can actually scale.

Here is the better playbook:

Unified data handling
: The system must efficiently manage both structured and unstructured data, as well as vector embeddings.

Scalability and performance
: Modern AI applications place heavy demands on data systems, often requiring the ability to handle massive workloads.

Strong data governance and security
: The datasets for AI applications often involve sensitive data. A data management system should have robust capabilities for compliance, regulations, and protection against unauthorized access.

Flexibility and extensibility
: A system should be adaptable to evolving AI technologies and capable of integrating with various tools and frameworks.

Community-driven innovation
: This is possible by using a well-supported, open source data management system. There will also be the benefits of flexibility and transparency.

Conclusion

Transactional data isn't just a record of the past—it's the foundation for the future of AI. Ignore it, and you're leaving competitive value on the table. Activate it, and you're building AI systems that actually drive enterprise value.

While generative models and vector embeddings dominate today's conversations, their true potential remains untapped without the rich, structured context of transactional data. These datasets represent years—often decades—of business operations, decisions, and customer behavior. Ignoring them in favor of flashy, one-dimensional applications means leaving massive enterprise value underutilized.

Modern AI applications demand more than just natural language prompts and static document retrieval. They require personalization, automation, and predictive analytics. Moreover, the path forward isn't about layering on more databases or adding complexity; it's about consolidation, extensibility, and activating the data you already have.

In this chapter, we covered several common traps in the AI landscape, including:

- Treating transactional data as "just logs" rather than the backbone of AI
- Building flashy AI pilots on documents alone, without grounding in transactions
- Adding niche databases that increase complexity instead of consolidating
- Failing to integrate transactional signals into personalization, automation, or prediction workflows
- Focusing only on unstructured data because it *feels* newer or trendier

When considering the importance of transactional data within an enterprise software system, there are 10 key questions that leaders should ask their teams:

1. Are we treating transactional data as a commodity or as a core strategic asset?
2. What are the most valuable transactional signals we walk over every day?
3. How are we currently using transactional data to personalize employee or customer experiences?
4. Can our AI systems take direct action based on transactions, or do they stop at answers?

5. Where are opportunities to automate transactional workflows?
6. Are we blending transactional and unstructured data effectively, or keeping them siloed?
7. How much value are we leaving untapped in predictive analytics by not using our own transaction logs?
8. Is our data architecture growing more complex, or are we simplifying around unified platforms?
9. How do we measure the ROI of our transactional data initiatives?
10. If our transactional systems went offline tomorrow, which business processes would grind to a halt first?

You don't compete in AI by throwing more documents at models. You compete by turning every transaction into intelligence. Transactional data is what can transform AI from experiments into ROI-driving enterprise systems.

CHAPTER 5

AI Application Design Patterns

In September 2023, David Cahn, a venture partner at Sequoia Capital, published an intriguing blog post: "AI's $200B Question". His thesis was clear—AI's infrastructure bill, from GPUs to data centers to energy, had already crossed $200 billion. And with revenues lagging far behind, the return on investment seemed perilously out of reach. Less than a year later, Cahn posted an update with a new title and a new number: "AI's $600B Question". The gap between infrastructure spend and realized value was widening.

This isn't new. In the 1990s, billions were poured into fiber-optic infrastructure in anticipation of the internet's rise. But when the revenue didn't arrive fast enough, the dot-com bubble burst. And yet—those investments weren't in vain. The infrastructure itself paved the way for Amazon, Google, and others to dominate the next era.

AI is facing a similar crucible. The question isn't whether the infrastructure matters; it's how quickly we can turn that spend into applications that create value. And for enterprises, that comes down to a fundamental choice: do you migrate your existing systems, or do you transform them outright?

The choice doesn't happen in a vacuum. It's shaped by three major forces pulling every CIO and CTO:

- The rise of open source: mature, community-driven platforms such as PostgreSQL now rival and pass proprietary systems.
- The explosion of new applications: from copilots to autonomous agents, the pace of app innovation is outstripping most IT teams' ability to keep up.
- The migration-versus-transformation dilemma: do you port what you have, or reimagine for the AI era?

These tensions define the architecture of the modern AI application. And they point to where value gets created: beyond chatbots, beyond RAG demos. As we explored in Chapter 4, real progress requires AI to address real problems—problems that matter, problems that unlock true productivity.

Today's AI ecosystem is facing a similar test. What will determine success isn't infrastructure alone—it's whether these systems can move beyond novelty.

Cahn's long-term view is instructive:

> A huge amount of economic value is going to be created by AI. Company builders focused on delivering value to end users will be rewarded handsomely. We are living through what has the potential to be a generation-defining technology wave.

We see the same horizon. We believe that companies willing to invest in AI today—who are willing to experiment, learn, and evolve—will gain a lasting competitive advantage. That's been central to our own work in integrating AI into our products.

In this chapter, we'll walk through how we brought this to life with our AI copilot for Oracle migrations. The road hasn't been linear, but each detour taught us something valuable.

Using AI to Transform Database Migrations from Oracle to PostgreSQL

Back in 2004, our company was founded on a simple but urgent need: enterprises wanted to break free from proprietary database systems without breaking everything else in the process. Oracle migrations were costly, risky, and time-consuming. We built tools, services, and compatibility features that made PostgreSQL a practical enterprise alternative.

That mission hasn't changed, but the methods have. AI has shifted the conversation from migration as a mechanical process to transformation as a value-creation process. The question is no longer just, "How do I move from Oracle to PostgreSQL?" It's also, "How do I redesign my workflows, apps, and data flows so they are agentic by default?"

What followed was a reinvention of our tools and our thinking.

The Problem

At a glance, Oracle and PostgreSQL can feel remarkably alike. Both are mature, battle-tested relational databases shaped by decades of industry evolution. They speak a shared language of data integrity, transactional reliability, and system performance.

So yes, there's a natural bridge between the two. And it's precisely this foundation that makes PostgreSQL such a compelling destination for Oracle migrations. But here's the reality: the similarities are only half the story.

Beneath the surface lie crucial differences. Some are obvious, others are subtle—but all are important. While both databases support SQL, ANSI standardization is not absolute. Each vendor has evolved its own dialects, extensions, and behaviors over time. Oracle, for instance, includes proprietary PL/SQL features, optimizer hints, and built-in packages that have no direct PostgreSQL equivalent. PostgreSQL, meanwhile, implements its own set of extensions and interpretations of the standard. Migrations can't succeed on one-to-one translations alone.

Consider the scenario of legacy Oracle implementations. These systems often span decades, evolving through countless iterations, patches, and personnel changes. The original architects may be long gone. Documentation—if it exists—is often fragmented or outdated. Entire segments of logic can feel more like artifacts than assets, sealed inside stored procedures and business rules that few truly understand.

In these cases, the database and applications were becoming closed and opaque. This opacity introduces risk. Unmapped dependencies, nested logic, and procedural sprawl can make organizations hesitant to act, even when they know change is needed.

Still, the momentum is clear. Enterprises see the value in open source—flexibility, cost savings, and a solid foundation for modern AI and analytics. The pull toward PostgreSQL is growing stronger by the year. The good news? AI is rising to meet this challenge. Generative models and intelligent assistants are beginning to deconstruct the complexity, automate the translation of procedural logic, and illuminate the hidden contours of legacy systems.

Bringing AI to Bear

In the early stages of a database migration, state-of-the-art LLMs such as ChatGPT, Claude, or Gemini can pull their weight on certain tasks. For example, when working with simple SQL constructs, generative AI can produce useful results with a little guidance.

Here's a simple prompt to illustrate:

> *Prompt:* Please convert the following Oracle SQL `CREATE TABLE` statement to its PostgreSQL equivalent:

```
CREATE TABLE "ACTOR_ROLE_INFO"
(
   "ACTOR_ROLE_ID" NUMBER,
   "ACTOR_ID" NUMBER,
   "ROLE_ID" NUMBER,
   "STATUS" NUMBER,
   "CREATED_BY" NUMBER,
   "CREATED_ON" TIMESTAMP (6) WITH TIME ZONE,
   "MODIFIED_BY" NUMBER,
   "MODIFIED_ON" TIMESTAMP (6) WITH TIME ZONE
);
```

The response from an LLM might look something like this:

```
CREATE TABLE actor_role_info (
   actor_role_id INTEGER,
   actor_id INTEGER,
   role_id INTEGER,
   status INTEGER,
   created_by INTEGER,
   created_on TIMESTAMP WITH TIME ZONE,
   modified_by INTEGER,
   modified_on TIMESTAMP WITH TIME ZONE
);
```

This is straightforward and effective. The model translates data types, adjusts formatting, and preserves key attributes such as time zone awareness. All of this from a general-purpose model, with no

fine-tuning, no RAG, and no domain-specific optimization. But this kind of success only goes so far.

Once you move beyond structural syntax into more nuanced or advanced behavior—think autonomous transactions, PL/SQL package logic, or complex triggers—things get murkier. Hallucinations can creep in, misinterpretations become more frequent, and critical functionality may be dropped or misrepresented altogether.

There's also a deeper issue: most public LLMs haven't been trained to understand the unique Oracle compatibility features embedded in EDB PG AI. This matters because it isn't just open source PostgreSQL; it's an enterprise-grade, sovereign platform that simplifies and accelerates Oracle migrations. When you combine that with AI-powered tools, you're not just speeding up the process—you're accelerating what was already built for speed. When you combine AI with a platform already designed to simplify Oracle migrations, you amplify both.

Integrating AI into an Existing Application

These early experiments showed us both the promise and limits of using AI for migration. Simple translations worked, but the real complexity of Oracle-to-PostgreSQL migrations required more than syntax files. We realized that if AI was going to add real enterprise value, it needed to be part of the workflow itself, not a side experiment. That realization led us to invest in building a copilot.

And here's an important lesson we learned along the way: when most teams begin exploring AI, the first instinct is often to build a chatbot.

It's understandable. After all, we're in the midst of an agentic AI, and chatbots have become its most visible expression. But it's time to pause that instinct. Because the truth is, users don't necessarily want another chatbot.

In fact, introducing multiple chat interfaces across an organization can lead to fragmentation and fatigue. Users are left wondering: Which chatbot do I use for which task? What is it supposed to know? Why does it live in its own silo? Meanwhile, IT teams are left juggling maintenance, governance, and security across a growing patchwork of AI tools.

There's a better path—one that's quieter, smarter, and far more effective. Rather than building stand-alone AI experiences, consider weaving AI directly into your existing workflows. Let it enhance the software users already rely on. In doing so, you preserve context, reduce friction, and meet people where they are.

That's the route we took with our own AI copilot for Oracle migrations. Instead of standing up a separate application, we embedded AI directly into the core functionality of our migration tooling. The result was a more intuitive and productive experience.

And there's another important shift: not everything needs to be natural language–driven. Sometimes the best AI is invisible.

Think AI agents and background automation: these aren't just passive tools waiting for instructions. They work continuously behind the scenes, handling real-time summarization, intelligent error detection, and adaptive data cleaning. No prompts required. The agent autonomously knows what to do, surfacing insights and recommended actions only when they're truly needed.

When a user does need to interact with AI, it doesn't have to feel like chatting with a bot, either. The experience can be as simple and elegant as tapping a button. Consider Apple's approach in Mail: tap "Summarize," and an AI-generated recap appears above the message thread—no typing, no back-and-forth, just context-aware assistance delivered seamlessly. This is the future of enterprise AI: embedded, intuitive, and designed to enhance workflows without getting in the way.

Understanding What Data You Have, and What Data Is Safe to Use

Before any AI system can deliver value, there's a far more foundational question to answer: do you actually know where your data is, and whether it's safe to use?

For many organizations, the honest answer is, "Not really, no." Data often lives in silos—fragmented across formats, platforms, and purposes. Some is buried in public documentation. Some in blog posts. Some in internal wikis, support portals, and external knowledge bases. Rarely is it structured in a way that's AI-ready. And even more rarely is it governed.

When we began our AI journey, one of the first things we had to do was inventory our content. Not just locate it but organize it. Align it. Reshape it into a format that a model could reason over effectively. That work wasn't glamorous—but it was essential.

Because data is more than an ingredient; it's the infrastructure. And if it's messy, disconnected, or unlabeled, no model in the world will give you meaningful output. But content structure and location are only part of the challenge. There's also safety and governance.

Especially in regulated industries, the risk of exposing private, customer-sensitive, or noncompliant information is significant. The General Data Protection Regulation (GDPR), the Health Insurance Portability and Accountability Act (HIPAA) of 1996, and evolving data sovereignty laws create new constraints. So, too, do customer contracts, which may not have anticipated the ways generative AI might interact with stored data.

That's why the most critical step isn't technical. It's organizational. This is why it's a good idea to assemble a governance team. It doesn't have to be large—less than a dozen people is often enough. But it should include legal, IT, product, and data experts. Their first task is to set clear, simple guidelines. Define what data can be used, and how. Then iterate from there.

There's also another tool worth exploring, particularly when data sensitivity is a concern: synthetic data. Synthetic data is artificially generated to mimic the patterns, structures, and distributions of real-world datasets, without exposing any actual, identifiable information. It can be created through:

- Advanced statistical modeling
- Simulation-based environments
- Machine-learning generation techniques

For example, we used synthetic data to represent customer migration rules that couldn't be shared with an LLM. The original data was anonymized and then abstracted into representative examples. This approach allowed us to train and refine models without compromising trust.

And while synthetic data is still emerging as a discipline, it holds enormous promise. But it comes with caveats:

Accuracy
: It may miss the subtle, real-world variability that authentic data offers.

Overfitting
: If training relies too heavily on synthetic examples, the model could mostly memorize patterns.

Model collapse
: If synthetic data is repeatedly used in closed-loop retraining, diversity and relevance can erode, leading to performance degradation over time.

Still, with care and strategy, synthetic data offers a safe, scalable path forward—particularly in early experimentation phases. In our case, it proved instrumental in getting our AI copilot for Oracle migrations off the ground.

Understanding What Data Is Useful—or How to Make It Useful

There's a common misconception that building a generative AI application is simply a matter of feeding an LLM as much data as possible. *Just embed all your documentation, toss it into a RAG pipeline, and let the model work its magic.*

But that's not how meaningful AI is built. Yes, LLMs are powerful. But they also can be indiscriminate. Without structure, context, and curation, these systems are just as likely to reproduce contradictions as they are to deliver insights. And in most enterprises, the data landscape is anything but clean. Different teams produce documentation. Different formats emerge. Subtle inconsistencies—sometimes even outright contradictions—creep in.

That's why data quality isn't optional. It's fundamental.

When we built our AI copilot for Oracle migrations, we didn't start by amassing everything we could find. We started with intent. We asked: what makes data useful to this specific system, and what does "good" look like for AI in the context of database migration?

We developed a targeted framework around four key dimensions:

Accuracy
: Is the data correct? Is it current?

Relevance
: Does it speak to our Oracle-to-PostgreSQL migration journey?

Completeness
: Are common migration scenarios represented, and edge cases included?

Consistency
: Is the guidance uniform across documentation, code samples, and support materials?

From there, we focused on the highest-value sources:

- Structured documentation detailing syntax and behavior differences between Oracle and PostgreSQL
- Before/after code samples from real-world migrations
- Patterns of common pitfalls and how to avoid them
- Deep institutional knowledge, distilled through expert interviews and peer-reviewed inputs

In short, we weren't building a general-purpose chatbot. We were designing a purpose-built copilot, optimized for our platform, EDB PG AI.

Once the content was curated, the next step was transformation. Data has to be shaped for retrieval—chunked into focused, context-aware segments that allow the AI to deliver precise and useful responses.

We also invested in metadata. Granted, it took manual work. But the payoff in retrieval quality was undeniable. Tagging content with source type, scenarios, and context improved precision and reduced ambiguity.

Then came canonical examples—what we called "golden patterns." These were high-frequency, high-impact scenarios we wanted the copilot to get right every time. They became reference points, not just for training but for grounding the system in reality.

And finally: testing. No AI application should be deployed without rigorous validation. We built testing frameworks to ensure our system understood its inputs, returned consistent results, and respected the complexity of real-world migrations.

The lesson? Building meaningful AI isn't about *more* data. It's about the *right* data—accurate, relevant, complete, consistent.

Key Considerations for Application Architecture

When Microsoft introduced Clippy in 1997, the goal was to make technology feel approachable—an animated assistant to guide users through the complexities of Word and Excel. But what they got instead was a symbol of digital overreach. Despite the ambition and investment behind it, Clippy became less of a companion and more of a punchline. Why? Timing, tone, and relevance.

Clippy had a habit of interrupting users mid-task with prompts that felt obvious at best and out of touch at worst. "It looks like you're writing a letter," it would say, its cartoon eyes darting around the screen. The problem wasn't that it offered help—it's that it assumed too much and understood too little. It lacked context. And it demanded attention.

The lesson still applies today. Even the most well-meaning AI can undermine an experience if it doesn't respect the user's workflow. We took those lessons to heart when building our own copilot. We focused on seamlessness. Not a new interface. Not another command prompt. Just a simple gesture: right-click to assist.

It may seem small. But it changed everything. Here's what that design taught us:

- Context is essential. AI should appear at the point of need—not before, not later.
- Insight beats intrusion. The most helpful systems do their thinking in the background, surfacing only what matters.
- Less is more. The most elegant AI features often go unnoticed—because they blend perfectly into the moment.

No animations. No interruptions. No blinking eyes. Just quiet intelligence, ready when you are.

A good example of this is Slack's "thread summaries," which automatically generate concise recaps at the top of busy conversations. No prompts, no new interface. Just instant context where teams already work. This is the same principle: embed AI without making it feel like another system to learn.

Designing Effective Data Access Patterns

When you're working with enterprise-grade data—especially data rooted in legacy systems such as Oracle—security isn't an afterthought. It's the foundation. That was our mindset when developing the AI copilot for Oracle migrations.

From day one, we knew that trust would be earned through discipline. Every architectural choice, every interaction, every system handshake needed to be deliberate and defensible. Here's how we approached it:

Secure data access channels
: We implemented encrypted connections and authenticated APIs, ensuring that every request passed through verified and protected gateways. No blind access. No shortcuts.

Optimized data retrieval
: Performance couldn't come at the cost of safety. We built a hybrid system that balanced real-time access with intelligent caching—preserving both speed and integrity.

Sensitive data management
: Through a combination of anonymization, encryption, and selective filtering, we ensured that private or regulated information remained protected throughout the AI lifecycle.

Granular permission models
: Access followed principle—not convenience. We aligned permissions with enterprise identity systems to ensure that users could access only what they were authorized to see.

Monitoring and auditing
: We established robust logging and audit trails, enabling visibility into who accessed what, when, and why—critical for compliance, transparency, and continuous improvement.

Data sovereignty compliance
: We accounted for regional regulations and privacy requirements.

Model version control
: We introduced strict versioning protocols so that every update could be tracked, validated, and rolled out consistently across environments.

Resilience planning
: We anticipated failure modes and built fallback systems to ensure continuity in the event of model unavailability or degradation.

Leveraging Existing Tools and Frameworks

With that foundation of security and resilience in place, the next step was choosing the right tools to build on.

When we set out to build the AI copilot for Oracle migrations, we knew that starting from scratch wasn't the answer. The generative AI ecosystem was already rich with frameworks designed to handle the heavy lifting of integration, orchestration, and interaction. So instead of reinventing the wheel, we focused on delivering value by building on an already proven foundation.

The benefits went beyond technical capabilities. The open source ethos gave us full visibility into the code, which allowed us to audit for security, customize for compliance, and align everything with our internal standards. And because these solutions are supported by active communities, we gained access to a deep pool of best practices, updates, and innovation.

That said, flexibility comes with responsibility.

Open source projects move fast. Features change. Stability isn't always guaranteed, especially at the edge of innovation. That's why we approached integration with discipline: testing thoroughly, locking dependencies, and relying on stable releases when deploying to production.

Conclusion

As this chapter has illustrated, building an AI application—especially one that grapples with the intricacies of Oracle-to-PostgreSQL migration—isn't about trend-chasing or layering on jargon. It's about designing with intent: patterns that elevate user experience, safeguard data fidelity, honor context, and earn trust. The focus needs to be on making AI quietly powerful, invisible when appropriate, and unmistakably useful when it counts.

In this chapter, we covered several common traps in the AI landscape, including:

- Treating AI as a stand-alone chatbot instead of embedding it into workflows
- Migrating code without transforming workflows for automation and agentic patterns
- Assuming all enterprise data is AI-ready without inventory, governance, or cleaning
- Relying on synthetic data without accounting for its caveats (variability, overfitting, collapse risk)
- Building on open source frameworks without locking dependencies and testing stability
- Ignoring context and user experience—AI that interrupts creates distrust

When considering AI application design, there are 10 key questions that leaders should ask their teams:

1. Are we clear on whether our priority is migration or transformation?
2. What workflows actually create value for users, and how could AI enhance them?
3. Which of our transactional systems represents "ground truth" for the business?
4. Have we inventoried where our data lives, and do we know which parts are safe to use?
5. What governance guardrails are in place to protect sensitive or regulated information?

6. How are we defining "useful data" for this application (accuracy, relevance, completeness, consistency)?
7. Where can AI be invisible and embedded, rather than a new interface or silo?
8. How will we ensure context—that AI appears at the point of need, not as an interruption?
9. What resilience measures (versioning, fallback, monitoring) are built into our architecture?
10. Are we prepared to test and validate continuously, not just at launch?

Every design choice in AI applications is a decision about where value will be created. Treat migration as transformation, and you'll build systems that last.

CHAPTER 6

Sixteen Critical Fault Lines That Will Make or Break Your AI Build

Before you build, there are sixteen critical fault lines that you need to address—six on business and process, ten on technical design. These aren't academic. Each represents a real-world derailment we've seen in enterprise deployments. From our 2025 global research, only 13% of organizations have AI in mainstream production (defined as more than 50% of deployments out of experimentation mode).[1] They succeeded by treating these as *design requirements*, not afterthoughts.

Business and Process Fault Lines

Let's first take a look at the six concerns for businesses and processes.

1. Misalignment Between Teams (Traps and Assumptions)

AI projects cut across data, IT, compliance, policy, and frontline ops. Without early alignment on goals and responsibilities, initiatives stall.

1 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. https://www.enterprisedb.com/sovereignty-matters.

In a traditional software project, most of the heavy lifting sits with product and engineering teams, with input from customers, sales, and support functions.

In an AI project, the scope is broader. Data management, governance, hosting, and compliance all have direct, tangible responsibilities that are just as critical as the code itself. It's like building a house. You start with the end-state vision, but then begin building truly from the ground up. Sewer, water, electric, foundation, framing, and plumbing all must be sequenced and coordinated. In AI, that's data pipelines, governance, hosting, and core development. If one misses, the whole system leaks—sometimes literally in the form of data exposure or broken dependencies.

Enterprises that get this right start by assigning explicit ownership across functions up front, reviewing contributions as work products, not opinions.

2. Overreliance on Generative User Experience (GenUX) Too Early

In the same vein as the home construction analogy, think of the difference between builder-grade and craftsman-built homes. A builder-grade house compromises on quality for speed and appearance. A craftsman home not only looks appealing but it also invests in the internals, the bones of the house.

The same dynamic plays out in AI projects. It's easy to create a flashy chat interface. It takes much more discipline to ensure that retrieval integrity, data governance, and compliance are solid. If you rush the UI before building the foundation, you'll end up with something that looks polished in demos but fails in production.

We've seen organizations roll out attractive copilots that collapse under real usage because the underlying retrieval was inaccurate or the compliance filters were missing. In regulated industries, this isn't just embarrassing; it's a liability. The lesson? Don't rush or skimp on structural integrity. Invest first in the layers no one sees, and your interface will stand up under pressure.

3. Unscalable Feedback Loops

User input and reaction are paramount in AI systems, to some extent, regardless of where and how they're delivered. Not only will it help you refine your models, but direct user engagement and up/down voting will help train your models directly.

When considering traditional software projects, teams typically focus mostly on the binary divide between whether it works or not. User experience often takes a secondary role in improving the product. In the case of AI projects, especially generative AI, it's much more subjective. Even if there is the case of whether the statements generated are true or false, it's not always readily apparent what true or false *is* in a given context. With traditional software, if you click a button and expect a window to open, and it does not, it is not working. With GenAI, if you ask a question about something you don't know the answer to, how do you know the answer is correct, even if it seems completely plausible? If the user didn't know the answer to begin with, how will they judge?

This is why early copilots need deliberate mechanisms: up/down voting, inline corrections, or structured feedback forms tied directly to model retraining. Without them, valuable insights die in chat threads.

4. Undefined ROI or Success Metrics (the Enthusiasm-to-Performance Gap)

AI projects are often launched with excitement, but without clear key performance indicators (KPIs), enthusiasm fades fast. To measure and demonstrate continued value, define three categories of success before launching:

Efficiency
: Measurable improvements in time-to-resolution, reduced steps in a workflow, or fewer escalations

Innovation
: New workflows unlocked or content produced that wasn't previously attainable

Risk reduction
: Fewer policy violations, reduced manual errors, or improved compliance tracking

Programs that skip this step often collapse under scrutiny. Leaders want to know if the investment pays off. Set the metrics early, measure consistently, and you'll bridge the gap between initial excitement and long-term adoption.

5. AI Ethics and Data Privacy Oversights

Ethics, accuracy, and privacy sit at the core of your data governance strategies, and the rules and conditions applied at this layer inform the overall experience your users will have with your solution. Not to mention that, in an increasingly polarized society, mishaps on this front can mean the demise of your product, your reputation, and potentially your company.

The answer is to embed governance into design from the start. Mastercard's AI Council is a leading example. Its governance framework oversees all AI efforts and embeds ethical guardrails into design and build phases, not post-launch.

This means enforcing data minimization rules, role-based retrieval filters, and audit logging from day one. It means red-teaming against prompt injection, running safety classifiers, and designing escalation paths for ambiguous queries.

6. Insufficient Data Product Thinking

Data is seldom treated like a product—with owners, versioning, and curation. This goes back to the home construction analogy. In AI projects, pipelines are often treated like plumbing—built once, left running, and only noticed when they break. That mindset is fatal. Pipelines are not plumbing; they are products with users, owners, and service-level expectations.

Those groups (sales, customer experience) outside of product and engineering are not just passive observers and commenters—they are owners, responsible for a specific and critical function, and should be treated and held accountable as such.

Technical Design Fault Lines

Next, let's take a look at the remaining concerns for technical design.

7. Schema Drift and Query Fragility

Schema changes quietly break retrieval and SQL generation. Automating schema tracking and regression testing for generated queries is crucial because schema changes can silently break applications. Without these guardrails, a seemingly minor schema change, such as renaming a column, altering a data type, or dropping a table, can cause queries to fail or return incorrect data. The result is production errors, compromised data integrity, and loss of trust in the data. The team needs to take steps, including:

Fixing silent failures and broken functionality
: Schema changes don't always cause an immediate, obvious error. A change might result in a query returning no results instead of an error, or it might produce incorrect data that appears valid. This makes it difficult but crucial for the team to detect the problem before it's too late, thus protecting both business decisions and customer-facing features.

Preventing regression
: Regression testing ensures that new changes don't break existing functionality. When queries are automatically rerun against the new schema, any breakage is immediately flagged. This is especially vital in modern development environments with frequent updates, as it allows teams to quickly and confidently make changes without fear of unintended side effects.

Maintaining data quality and integrity
: Generated queries, particularly in data analytics and business intelligence, rely on a stable schema to produce accurate results. When the underlying structure changes, the queries can become invalid or produce misleading information. Automated tracking and testing validate that the data pipelines and reporting tools continue to function correctly, ensuring data quality is maintained.

Increased efficiency and reduced manual effort

Manually tracking schema changes and testing every generated query is a time-consuming and error-prone process. Automating this work frees up engineers and analysts to focus on more complex, strategic tasks. It provides a fast, consistent, and repeatable way to validate the system's integrity after every change.

Enabling CI/CD pipelines

Automated schema tracking and regression testing are essential components of a robust continuous integration/continuous deployment (CI/CD) pipeline. By integrating these checks into the development workflow, you ensure that any proposed schema change is automatically validated against existing queries before it's deployed to a production environment, preventing bugs from ever reaching users.

8. Latent Infrastructure Debt

Legacy systems slow down deployments and block hybrid options. Auditing and modularizing legacy infrastructure early is vital because these outdated systems act as a significant roadblock to modern IT practices such as faster deployments and hybrid cloud adoption. They're often monolithic, tightly coupled, and use outdated technology, which makes them difficult and risky to change:

Slower deployments

Legacy systems typically have complex, manual deployment processes. Because their components are tightly intertwined (a "monolithic" architecture), a small change to one part of the code can have unforeseen side effects elsewhere. This forces long, cautious testing cycles and slow, infrequent releases. In a fast-paced market, this lack of agility is a significant competitive disadvantage.

Blockers to hybrid and cloud adoption

Legacy systems were designed for on-premise environments and are often incompatible with modern cloud technologies. They may not support modern APIs or data formats, making it extremely difficult to integrate them with cloud-based services. This prevents companies from leveraging the scalability, flexibility, and cost-efficiency of a hybrid cloud model, in which some workloads remain on-premise while others move to the cloud.

Increased risk and cost
: Maintaining legacy systems is expensive. They often require specialized knowledge from a small, aging group of experts, and finding replacement parts or technical support can be a challenge. Furthermore, these systems frequently have security vulnerabilities that are no longer patched, making them prime targets for cyberattacks.

Technical debt accrual
: Each time a work-around is built to make a legacy system function with a new technology, it adds to the company's technical debt. This debt accrues over time, making the system more complex and difficult to maintain, which eventually stalls innovation and requires a massive, costly overhaul.

Auditing for these risks early, modularizing where possible, and moving to containerized, portable architectures is the only way to keep infrastructure from becoming the anchor that drags AI adoption under.

9. Lack of Embedding and Retrieval Optimization

Poor chunking, indexing, and embedding tuning lead to bad answers.

Effective chunking, indexing, and embedding tuning are critical for the quality of AI-generated answers, especially in RAG systems. Poorly executed data preparation at this stage is a primary cause of bad or "hallucinated" answers, as it directly impacts the relevance of the information the model retrieves.

Experimenting with parameters such as chunk size and overlap, using hybrid scoring, and fine-tuning embeddings is important because it allows you to optimize the entire retrieval pipeline to ensure the AI has the best possible context to work with.

Think of the process as a librarian finding a book to answer a question:

Poor chunking
: If the librarian breaks a book into random sentences, they might hand you a meaningless fragment that's missing the key context. This is what happens with poor chunking. If the chunks are too small, they lose context, and if they're too large, they introduce irrelevant "noise" that confuses the model.

Poor indexing/embedding

If the librarian's card catalog is disorganized or uses a flawed categorization system (like an embedding model that doesn't understand nuances), they might pull the wrong book entirely. The indexing and embedding process is what makes the knowledge base searchable. A bad embedding will misrepresent the semantic meaning of the text, causing the retrieval system to pull irrelevant information.

Experimenting with these parameters is the process of building a better "librarian," or model:

Chunk size and overlap

The optimal chunk size is a balancing act. It must be large enough to contain a coherent thought or topic but small enough to be precise and not introduce unnecessary information. Slider overlap margins create a "sliding window" effect in which chunks overlap slightly, ensuring that important context isn't lost at the boundaries of a chunk. Experimenting with these settings allows you to find the sweet spot for your specific data, improving the relevance of what's retrieved.

Hybrid scoring

Semantic and keyword searches are both popular formats:

- *Semantic search (dense retrieval):* finds documents based on meaning and context
- *Keyword search (sparse retrieval):* finds documents based on exact keyword matches

But purely semantic (vector) search can fail with very specific queries, such as those with acronyms, names, or unique product codes. And keyword search, on the other hand, can struggle with the following:

- Synonyms or natural language phrasing (e.g., "feline" versus "cat")
- Longer or more conversational queries where relevance depends on context, not exact matches

By using a hybrid approach, the system gets the best of both worlds, leading to more accurate and reliable search results, especially for mixed-style queries.

Embedding tuning

The quality of the embedding model itself is fundamental. A model trained on general text might not understand the specific jargon in a technical manual or legal document. Embedding tuning involves either choosing a domain-specific model or fine-tuning an existing one on your own data. This ensures that the embeddings accurately capture the nuances and relationships within your specific knowledge base, directly leading to more precise and relevant retrievals.

10. Overdependence on a Single AI Provider

Vendor lock-in is a serious risk because it creates a state of dependency on a single provider, which can lead to higher costs, slower innovation, and poor service. When you're "locked in," switching to a different vendor becomes so difficult and expensive that you're essentially at the mercy of your current provider.

To help solve this, build abstraction layers. Use model gateways and standardized interfaces to make providers swappable. Benchmark multiple LLMs against your golden datasets. Establish fallbacks so traffic can fail over between providers. Even limited dual sourcing creates negotiation leverage and resilience.

The takeaway: don't let your entire AI program hinge on a single vendor's uptime or roadmap. In a critical workload, that's not a strategy; it's a liability.

11. Missing Real-World Use Case Playbooks

Teams waste time—and introduce risk—when they reinvent solutions for every new AI project.

Encouraging reuse by creating internal playbooks is crucial because it prevents teams from "reinventing the wheel," which is a significant drain on time, resources, and institutional knowledge. When every team or individual starts a project from scratch, they not only duplicate effort but also miss out on the valuable lessons learned by their colleagues. This leads to inconsistent results, higher costs, and a slower pace of innovation.

There are several key reasons why playbooks with real-world uses matter:

Standardization and consistency
: Playbooks, such as those for RAG pipelines or compliance checks, codify best practices into a single, accessible resource. This ensures that every team follows the same high-quality standards. Instead of each developer figuring out their own way to structure a query or handle sensitive data, they can refer to the playbook for a proven, reliable method. This consistency is crucial for maintaining quality, especially in a large organization.

Accelerated development and innovation
: When foundational components and processes are already documented and ready to use, teams can start a new project with a significant head start. They can pull from a library of prebuilt code, templates, and proven workflows. This allows them to focus their energy on solving new, unique problems rather than re-creating solutions to common ones. Faster development cycles mean faster time-to-market and a greater ability to respond to changing business needs.

Knowledge transfer and onboarding
: Playbooks act as a living repository of institutional knowledge. They capture the expertise of senior engineers and subject-matter experts, making it available to everyone. This is a game-changer for onboarding new hires, as it provides them with a structured guide to the team's processes and standards, significantly reducing the learning curve and making them productive much faster.

Risk mitigation and compliance
: For critical areas including compliance and security, playbooks are nonnegotiable. They ensure that all teams are consistently following the necessary protocols, such as data privacy regulations or security checks. This reduces the risk of human error, which could lead to a data breach or a costly compliance fine. A playbook provides a clear, defensible record of how these requirements are met across the organization.

Fostering a culture of collaboration
: When teams are encouraged to contribute to and reuse from a shared knowledge base, it breaks down information silos. This collaborative approach encourages a culture of "inner-sourcing," in which teams operate like internal open source communities, sharing code, ideas, and feedback. This not only improves the quality of the shared resources but also strengthens relationships and communication between teams.

The best organizations treat playbooks as living repositories—updated quarterly, with contributions from across teams. This isn't bureaucracy; it's institutional memory. Reusable playbooks let enterprises move faster without cutting corners.

12. SQL Generation Without LLM Guardrails

Sandboxing and validating generated SQL is important because flawed queries can expose sensitive data, corrupt databases, or return incorrect results. In today's data-driven world, where many systems use automated tools to generate SQL, this validation is a critical security and data integrity measure.

The risks of data generation without guardrails include:

Data exposure and security risks
: Uncontrolled SQL generation can lead to security vulnerabilities such as SQL injection. A poorly designed query could unintentionally bypass security checks, allowing it to access or expose sensitive information such as customer data, financial records, or internal passwords.

Data corruption
: If a generated query isn't explicitly read-only, it could inadvertently execute a `DELETE`, `UPDATE`, or `DROP TABLE` command. This can lead to irreversible data loss or corruption, which can bring a business to a halt and result in significant financial and reputational damage.

"Garbage in, garbage out"
: A query that's syntactically correct might still be logically flawed. For example, it could contain a subtle join error, an incorrect filter, or a faulty calculation. Running this query would return what looks like a valid result but is in fact "garbage"—incorrect or misleading data that could lead to poor business decisions.

Sandboxing and validating generated SQL addresses these issues proactively:

Sandboxing and read-only enforcement
: A sandbox environment is a safe, isolated space where generated queries can be run without affecting the production database. Enforcing `read-only` access at this stage ensures that even if a query is malicious or flawed, it can't alter or delete any data. This is a fundamental security practice that prevents unintended side effects.

Validation against live schema
: Automatically validating a generated query against the live database schema (i.e., its structure of tables, columns, and data types) ensures that the query is targeting the correct data. This check immediately catches common errors such as misspelled table names or references to columns that no longer exist, preventing runtime failures.

Validation against test datasets
: Running the validated query against a controlled test dataset is the final step in ensuring its accuracy. By comparing the query's output to a known good result set, you can confirm that the query is returning the correct information. This step catches logical errors that a simple schema check would miss and ensures the data being consumed is reliable and accurate.

13. Lack of Multitiered Architecture Alignment

Modern AI workloads span three domains: transactional (OLTP), analytical (OLAP), and vector search. Too often, enterprises spread these across disconnected systems. The result: data silos, duplicated governance, and latency that kills responsiveness.

Using a unified platform such as EDB PG AI to align different database layers—vector, analytical, and transactional—is essential because it eliminates data silos, reduces complexity and duplication, and ensures data consistency. Instead of managing multiple, specialized databases that don't communicate well with each other, a unified system puts all your data in one place, which is crucial for modern applications, especially those using AI.

Unified platforms solve these problems by consolidating everything into a single, cohesive system. Benefits include:

Single source of truth
: All data—whether it's raw transactional data, aggregated analytical data, or vector embeddings for AI—lives in one database. This eliminates the need for complex data pipelines and guarantees data consistency, ensuring that a change to a piece of data is immediately available to all parts of the application.

Simplified operations
: A unified platform is a single system to manage, secure, and scale. This drastically reduces the operational burden on IT teams and lowers the total cost of ownership (TCO) by reducing the number of tools and licenses needed.

Faster, more powerful applications
: With a unified system, you can perform hybrid queries that combine different data types in a single request. For example, you can query a transactional database for all products sold in a specific region and then use a vector search to find products with similar descriptions. This enables powerful, real-time analytics and AI features that would be nearly impossible with a fragmented architecture.

14. Inadequate Prompt Engineering Lifecycle

Treating prompts like code is important because it ensures your AI applications are reliable, consistent, and secure. Without versioning and testing, prompts can "drift," meaning they subtly change in behavior over time, leading to unpredictable or incorrect responses, security vulnerabilities, and system failures:

Versioning
: Just as you track changes to your codebase with Git, you should version your prompts. This allows you to revert to a stable version if a new one performs poorly. It also provides a clear history of changes and lets you audit how a prompt has evolved.

Testing
: Unit and integration testing for prompts is crucial. You should have a suite of test cases with known inputs and expected outputs. Running these tests regularly ensures that a prompt continues to deliver correct results. This process helps you catch regressions before they impact users.

Monitoring and refining
: Prompts need continuous monitoring to track their performance in a live environment. Key metrics to monitor include response time, accuracy, and the rate of failures. This data helps you identify drift and gives you the information needed to refine prompts to improve their performance over time.

By treating prompts as a core part of your application's code, you move from a reactive, unpredictable development process to a proactive, reliable one. This approach is essential for building robust and trustworthy AI systems.

15. Lack of Real-Time Schema Awareness

Agents working with an outdated schema context behave unpredictably. Keep schema metadata in sync, refreshing with every deployment or ingestion cycle.

Keeping schema metadata in sync is fundamentally important because it provides the "blueprint" that allows AI agents to accurately and reliably interact with your data. When this blueprint is outdated, the agents become like workers with an old, incorrect architectural plan: they are prone to unpredictable and often catastrophic failures.

By refreshing schema metadata with every deployment or ingestion cycle, you ensure that the AI agent's blueprint is always accurate. Benefits include:

Reliability and accuracy
: Agents will always have the correct information to generate accurate queries and perform tasks correctly. This is the foundation of a trustworthy, production-ready AI system.

Reduced development and debugging time
: Instead of spending hours or days trying to figure out why an agent's query failed, you can immediately identify the root cause: a schema mismatch. Automated synchronization makes this a nonissue.

Scalability
: A system that automatically keeps its schema metadata in sync is more resilient and scalable. As your data and systems evolve, the agents can adapt seamlessly, without requiring manual intervention.

Enhanced security
: A synchronized schema context is a critical security layer that helps prevent unauthorized data access and ensures agents are operating within the defined data structures and constraints.

16. Lack of Observability Across ML and SQL Layers

Without end-to-end observability, diagnosing failures in complex AI applications is incredibly difficult because you're essentially flying blind. A unified monitoring stack that logs key metrics across all layers of your system is the only way to get a holistic view and quickly pinpoint the root cause of an issue.

A unified monitoring stack brings all the telemetry data—logs, metrics, and traces—into a single place, giving you a comprehensive view of your system's health. Benefits include:

Pinpointing the root cause
: By logging and correlating metrics such as ML inference latency and SQL performance, you can see exactly where a bottleneck occurred. If a user complains about a slow response, you can immediately check if it was due to a slow LLM call or a database query that timed out.

Understanding and refining AI performance
: Fallback rates tell you how often your system is failing to produce a good response and must resort to a predefined or simpler answer.

  Guardrail hits inform you when your safety and compliance rules are being triggered. This helps you refine both your prompts (to avoid triggering unnecessary guardrails) and your guardrails themselves (to ensure they're not too restrictive). This data is crucial for improving your system's accuracy and reliability over time.

Reducing mean time to resolution (MTTR)

When a failure occurs, observability tools provide the context needed to diagnose and fix the problem quickly. Instead of sifting through disparate logs and dashboards, you can see the entire user journey in a single view, from the initial request to the final response, which drastically reduces the time it takes to resolve an issue.

Proactive problem detection

By monitoring these metrics in real time, you can set up alerts to warn you of a problem before it escalates. For example, you can get an alert if the average LLM latency suddenly spikes or if the rate of guardrail hits jumps unexpectedly. This allows you to address issues proactively rather than reactively.

Conclusion

It is important that you do not take the sixteen critical fault lines outlined in this chapter as optional exercises. They are a blueprint for whether your AI initiative thrives or stalls. Long-term AI success depends on this proactive mindset. It's about aligning business and technical foundations, embedding governance from day one, and building systems that are resilient, ethical, and scalable. Enterprises that focus on these essentials give themselves the best chance of sustaining meaningful impact.

CHAPTER 7

How to Build an AI Application: An Introductory Guide

If you're building AI for the enterprise, don't think in terms of copilots. Think in terms of platforms. In an increasingly AI-driven and agentic world, you need a scalable foundation that can span functions, integrate with partner and customer systems, and allow you to navigate regulatory guardrails across geographies. Platforms outlast copilots because they standardize data access, governance, and reuse, so each new use case is better and faster than the last.

What felt like a distant future—AI agents interacting with other agents with minimal human intervention—is now an immediate reality. To get there, enterprises need platforms that deliver both technical capabilities (retrieval, model orchestration, hybrid data access) and operational guardrails (governance, compliance, observability). Together, these form the flywheel for scaling agentic workloads across your data and AI platform. For example, JPMorganChase created a firm-wide AI governance council to ensure that retrieval checks, model selection, and compliance logging are all embedded before new copilots go live.[1]

1 JPMorgan Chase & Co. "AI and Model Risk Governance." May 29, 2023. *https://www.jpmorganchase.com/about/technology/news/ai-and-model-risk-governance*.

The path doesn't start with moonshots. It starts with internal copilots, AI assistants for your own teams. They are lower risk, faster to measure, and ideal for building the operational muscle memory your enterprise will need later. A good starting point could be an IT help desk copilot that handles virtual private network (VPN) resets or single sign-on (SSO) issues, a sales operations copilot that manages customer relationship management (CRM) hygiene and pricing lookups, or a finance copilot that guides employees through expense codes. These are bounded problems with clear policies and measurable outcomes.

Retrieval-augmented generation (RAG) is the foundational pattern here: pair a generative AI model with a PostgreSQL backend, and you can deliver immediate utility while building the skills and infrastructure required for more advanced agentic systems. RAG means "don't guess, retrieve first." Instead of asking a model to invent an answer, you inject it with current, governed context so outputs are accurate and auditable.

A clear example comes from Morgan Stanley, which built a generative AI assistant for its 16,000 financial advisors. Every response is grounded in the firm's proprietary research library, rather than generated from scratch, to ensure compliance and accuracy.[2]

We'll go through the process step-by-step, because this can easily be the foundation stone. By the end, you should have a practical sense of how these concepts translate into a working application and what it takes to bring an AI idea to life in the enterprise. We'll anchor this walkthrough on one concrete pattern: an internal support copilot.

The most successful enterprises in the world are thriving with extensive generative and agentic AI. They deliver two times the amount of the mainstream approaches to AI, and they deliver five times the comparable ROI. They do it by starting in the right places. That "right place" is usually inside the enterprise, where feedback is fast, data is known, and risk is manageable.

2 Davenport, Tom. "How Morgan Stanley Is Training GPT to Help Financial Advisors." *Forbes*, March 20, 2023. *https://www.forbes.com/sites/tomdavenport/2023/03/20/how-morgan-stanley-is-training-gpt-to-help-financial-advisors*.

Why Choose an Internal Support Copilot?

An internal support copilot provides a helpful use case that highlights important concepts for generative AI. From EDB's 2025 global research, only 13% of organizations have AI in mainstream production (defined as more than 50% of deployments out of experimentation mode). This sounds like an obvious place to start, with your own people, but historically it's often been left as one of the last places, as external needs tend to come first.[3] Those that are proving successful often start internally—because the lessons compound outward. Here are five reasons this approach works:

1. *Tight feedback loops*

   You have the largest and most vocal critics and advocates inside your organization, so as you learn, your feedback loops are closer to the points of development and action. *Leverage point for broader AI adoption:* when employees experience how AI can make their workday smoother—fewer bottlenecks, quicker answers, less friction—they're more likely to champion AI in other contexts. These kinds of early successes can build institutional confidence and lay the groundwork for future initiatives. *It's better for you to be customer zero.*

   Here's a real-world example. After piloting with ~10,000 employees, Goldman Sachs rolled its GenAI assistant to the entire firm to summarize documents, draft content, and analyze data—explicitly aimed at productivity lift.[4] By starting with a focused internal group, Goldman was able to validate the tool's usefulness, refine it based on real-time employee feedback, and build confidence in its value before broader organizational adoption. The firm's approach shows how internal deployments create accelerated feedback cycles: employees testing the assistant immediately exposed where it added value, where it fell short, and where it could replace repetitive tasks. In turn, those insights informed refinements that made the tool more effective

3 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. *https://www.enterprisedb.com/sovereignty-matters*.

4 Azhar, Saeed, and Pritam Biswas. "Goldman Sachs Launches AI Assistant Firmwide, Memo Shows." Reuters, June 23, 2025. *https://www.reuters.com/business/goldman-sachs-launches-ai-assistant-firmwide-memo-shows-2025-06-23*.

across functions. It's a classic case of being "customer zero"—learning inside the enterprise before scaling outward.

2. *The ability to deliver near-instant measurable returns*

   EDB's research showed that 45%+ of the measurable ROI from copilots came from a sense of improved efficiencies and opportunities to innovate, and not just typical economic measures such as operating expense shifts or new revenue opportunities.[5] These metrics create a strong baseline for scaling because they are visible, immediate, and easily understood by employees and leaders alike.

   A compelling illustration comes from JPMorganChase, which deployed an AI coding assistant to support its software engineers. According to Reuters, the bank saw 10%–20% productivity gains, freeing engineers to focus on higher-value AI and data initiatives. Building on this success, JPMorgan announced plans to scale from roughly 450 AI use cases toward 1,000. By starting a copilot that delivered measurable internal efficiencies, the firm created early momentum and a clear business case for expansion into more ambitious AI initiatives.[6]

3. *Compliance and regulatory issues*

   Compliance and regulatory issues require specific guardrails and an understanding of systems and procedures that are best initiated inside because the next inevitable step is integration with external facing copilots, so knowing likely outcomes at those design stages is likely to drive faster program development and measurements. *Internal usage is the foundation for external success.*

   DBS Bank is a perfect case study. The bank has built more than 1,500 AI models spanning roughly 370 use cases, all governed by a comprehensive, responsible AI framework. By putting compliance and governance at the core of its program, DBS has been able to safely scale AI across critical banking operations. In 2025, the bank reported that the economic value generated by these initiatives is expected to exceed S$1 billion (Singapore

5 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. *https://www.enterprisedb.com/sovereignty-matters*.

6 Suresh, Haripriya. "JPMorgan Engineers' Efficiency Jumps as Much as 20% from Using Coding Assistant." Reuters, March 14, 2025. *https://oreil.ly/GZpXD*.

dollars). Its approach underscores that when compliance is integrated early, it doesn't slow innovation—it accelerates it, enabling both scale and measurable outcomes.[7]

4. *It's critical to have all teams learn as they go*
   An internal copilot acts as a growth mindset moment for the whole organization. The need for highly adaptive learning is a critical component in this new world because everybody needs to be part of the platform approach to be successful.

5. *It's likely a lower risk*
   Unlike external-facing AI products, internal copilots operate in controlled environments. The data sources are known, the user base is internal, and the compliance risks are easier to manage. That makes this type of project a good option for organizations that want to build foundations and then momentum with agentic and GenAI without biting off more than they can chew.

The Starting Point: The Copilot's Goals and Architecture

Before discussing the technology stack, it's worth stepping back to look at what the copilot is designed to achieve. At its core, the goal is simple: give employees fast, accurate answers by connecting their queries to the right internal knowledge within the enterprise. A user starts with a request, such as with chat, voice, or web form. The query is then routed into a retrieval system, which filters and ranks content from a vector store, applies metadata to refine the relevance, and then packages the result into a prompt for the LLM. Next, the model generates a grounded response, which is passed back to the user with built-in guardrails and transparency.

To bring this workflow to life, the system needs a platform capable of securely managing both the data and the AI components behind it. This internal support copilot runs on the EDB PG AI platform, a sovereign-ready system that merges data and AI tooling into one secure, enterprise-grade environment.

7 DBS Bank. "Responsible AI in Banking: Gaining a Competitive Edge." June 11, 2025. *https://www.dbs.com/artificial-intelligence-machine-learning/artificial-intelligence/responsible-ai-in-banking-gaining-a-competitive-edge.html*.

At its heart sits AIDB, an extension that transforms PostgreSQL into an AI-native database. This adds vector column support and similarity search directly in PostgreSQL using extensions such as pgvector. This allows for semantic queries and embeddings storage via SQL. There's no need for an external vector database because everything operates within PostgreSQL. This streamlines the architecture and improves efficiency for RAG workflows.

Here are other key factors to consider with the EDB PG AI platform:

- The AI Factory introduces low-code/no-code capabilities through a graphical interface and SDKs. This allows developers and analysts to build AI pipelines with just a few SQL statements or clicks. In terms of making the workflows, it is about dragging and dropping items on a canvas.
- With support for on-premise or hybrid deployments, EDB PG AI ensures data never leaves your control. Features such as encryption, role-based access, audit logging, and local model hosting provide for strong compliance and data sovereignty.

With a unified data and AI platform, there is a more simplified architecture and deployment:

- AIDB enables PostgreSQL to handle transactional, analytical, and vector workloads natively. This eliminates separate vector stores or ETL (extract, transform, load) pipelines. This helps to reduce data duplication and simplify governance.
- Using AI Factory, RAG pipelines and embedding synchronization can be configured in as few as five SQL lines. This allows for proofs of concept to reach production in days instead of weeks or months.
- Because the entire AI stack—from data storage to model serving and compliance—lives within your environment, data leakage risk is minimized, compliance is simplified, and vendor lock-in is avoided.
- With real-time observability and optimization across all workloads, teams no longer need separate dashboards or manual coordination. Everything is visible and actionable through one interface.

Laying the Groundwork: Building the Knowledge Base

For an AI copilot to be useful, it needs a relevant knowledge base. This includes content such as internal documentation, Git repos (commits and changelogs included), SQL scripts, configuration files, and conversations in tools including Slack. But pulling content from these systems is only step one. The real challenge is preparing it for reliable, accurate retrieval.

Why Chunking and Metadata Matter

Effective RAG systems depend on how that data is prepared, a critical topic we have covered throughout this book. Simply feeding entire documents into a retriever rarely yields useful results. The most accurate, high-signal responses tend to come from content that has undergone chunking.

Over time, a few practical approaches have emerged as particularly effective:

Fixed-length segmentation
: This is the most straightforward method; that is, to break content into equal-sized blocks, such as 500 tokens or words. It's fast to implement and works reasonably well for documents with consistent formatting. That said, it can introduce odd breaks, such as splitting a sentence mid-thought or cutting across a paragraph boundary. This can lead to less relevant retrieval results.

Sentence or paragraph splits
: A more linguistically aligned approach involves dividing text along natural language lines. By preserving complete sentences or paragraphs, you maintain logical flow within each chunk, making them easier for the retriever and LLM to interpret correctly. This is especially useful for narrative-style documents or support threads where coherence matters.

Recursive or structure-guided splitting
: When working with structured documents, it helps to follow the hierarchy. Start with larger units such as sections or headings, then break them down to paragraphs or sentences only as needed to meet context window limits. This method ensures that each chunk remains contextually meaningful, even when the source material varies in complexity.

Semantic chunking
: This strategy focuses on meaning, not mechanics. Instead of slicing text based on size or formatting, semantic chunking groups related ideas together using embeddings or topic modeling. Each chunk reflects a single cohesive thought. It's more computationally intensive, but it often yields the most useful retrieval candidates. This is especially the case with knowledge bases with dense or nuanced content.

Sliding windows with overlap
: Regardless of which primary strategy you use, adding a bit of overlap between adjacent chunks can help preserve context that might otherwise be lost at the edges. This may be about 10%–20%. True, this comes with some redundancy, but it improves the chances that critical information near a boundary isn't missed during retrieval.

Chunking is only half the equation. To make retrieval more intelligent, tag each chunk with rich metadata, such as document source, author, timestamp, department, or content type. This added context allows your retriever to filter and rank results more precisely.

For instance, a user searching for the "latest IT password policy" isn't just looking for a relevant paragraph. They need the most current version, ideally from the IT department. Metadata tagging enables that kind of nuanced filtering, improving both speed and accuracy.

Avoiding Knowledge Drift Through Automation

One of the most common pitfalls in AI deployments is the slow decay of the knowledge base. Organizations change constantly, and unless the AI system keeps up, its answers start to fall behind.

EDB PG AI solves these challenges with a native process. Features such as AIDB and AI Factory allow you to define ingestion pipelines

directly in PostgreSQL, with automated embedding updates and vector index refreshes tied to change data capture (CDC) events. Rather than building custom Lambda functions or brittle polling jobs, new data flows automatically into your AI stack.

Typically, the pipeline parses the content, breaks it into chunks, generates embeddings, and indexes the output for semantic search. By embedding this workflow directly into PostgreSQL with observability hooks, EDB PG AI ensures that pipelines stay fast, reliable, and invisible to end users—without the glue code most enterprises struggle to maintain.

Unfortunately, it can be difficult to get this right. EDB PG AI knowledge bases help by wrapping this pipeline automation in database capabilities that make it easy to deploy, monitor, and manage knowledge bases in production.

A Living Knowledge Base

When all these elements come together, you get more than a searchable archive. You build a living knowledge system that reflects what's happening inside your organization in real time.

The benefits are straightforward. Changelogs are indexed as they're written. New policies are discoverable the moment they're published. Slack conversations from last week can inform how the copilot answers support questions today. This is what makes a copilot truly useful: it's not a generic chatbot but a tool that understands your environment, stays current, and reflects your institutional memory with precision.

Making It Work

Figure 7-1 outlines the architecture of EDB's internal support copilot. First, the user makes contact, whether through a voice call, messaging app, or web interface. Next, the query flows into a retrieval system. This system fetches relevant contextual data from a vector store and applies metadata filters to narrow the scope. The resulting context is packaged into a prompt and sent to an LLM. The model's response, shaped by predefined rules, is then routed back to the user.

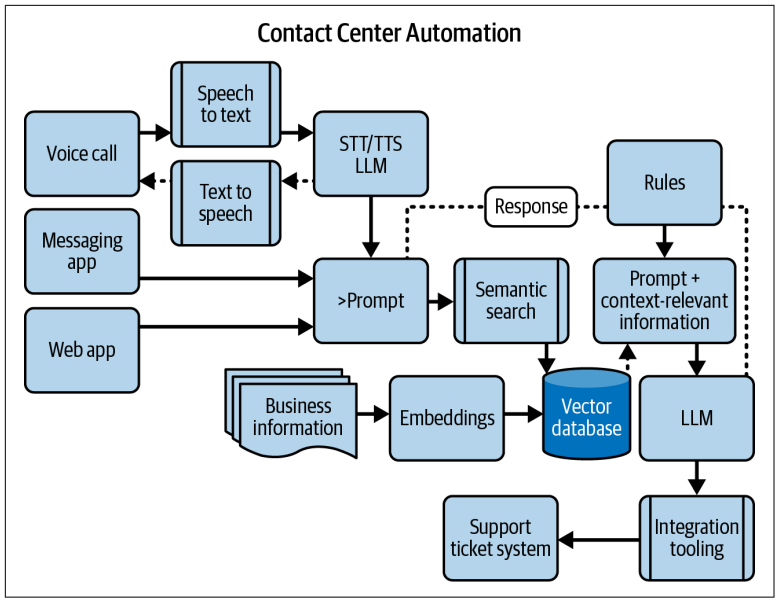

Figure 7-1. Workflow for an internal support copilot

In the sections that follow, we'll break down how to configure and optimize the retriever on the EDB PG AI platform. We'll look at how to strike a balance between relevance and coverage and examine some common implementation missteps to avoid.

Building the Retriever

The retriever is the operational core of the internal support copilot. Its role is to deliver just the right slices of information for the LLM to reason over. No more, no less.

As we mentioned in "Why Chunking and Metadata Matter" on page 97, you will need to use chunking on the data. It's important to think of your content as self-contained units. Each piece is paired with metadata, creating a multilayered structure that enables more nuanced retrieval.

The retriever in our system follows a hybrid approach. It begins by applying metadata features, such as document type, author, or creation date. This helps to narrow the pool of possible chunks and quickly trims down the search space. From there, the retriever shifts to vector similarity ranking, which involves comparing embeddings to find the chunks that are semantically closest to the user's query.

By layering these two techniques, we combine precision with speed. Metadata cuts away the noise, while vectors ensure we capture the meaning. But performance ultimately depends on how well you tune two factors: the density of the retrieval graph and the depth of each search. A denser graph gives the algorithm more potential connections to explore, while deeper probing increases recall at the cost of speed. Finding the right balance takes tuning—and retuning—based on how the system is used in practice.

As with most things in applied machine learning, optimization isn't a one-time task. Ongoing monitoring and refinement are critical to ensure that your retriever stays responsive, accurate, and aligned with the evolving shape of your data.

Balancing Relevance and Coverage

Another key for effective retrieval is striking the right balance between relevance and coverage. Let's take a look at the trade-off in practice.

Finding the right chunk size

Start with chunking. If your segments are too long, the core idea can get buried in noise. Too short, and you lose the necessary context to make sense of a query. Most systems work best with chunks in the range of a few hundred tokens. This is long enough to preserve meaning but short enough to stay precise. Take the example of a 20-page document. If you split it into single sentences, each chunk loses context. But if you keep the entire document as one chunk, the LLM may miss the specific answer, say, on page 12. The sweet spot is a few paragraphs at a time.

Tuning the index: graph density and search depth

You need to effectively configure your vector index. In graph-based structures such as HNSW (hierarchical navigable small world), two parameters matter most: graph density (how many connections each node maintains) and search depth (how thoroughly the system explores that graph during a query).

Denser graphs and deeper searches can surface more relevant results, especially for ambiguous or complex queries. But there's a cost: latency rises with each additional edge explored. The trade-off curve is usually concave. The initial improvements come cheap, but

diminishing returns set in fast once you pass certain thresholds. In other words, tuning isn't about maximizing every parameter. It's about calibrating to your workload. For real-time systems, small latency gains might be worth sacrificing some of the recall. For batch or asynchronous workflows, the trade-off leans the other way.

You can think of index tuning like optimizing a social network's search: balancing how deeply you traverse connections to find meaningful results without slowing everything down. If you tried connecting every user to hundreds of friends-of-friends, the search would quickly bog down. A leaner graph delivers "good enough" results far faster.

Beyond vector similarity: hybrid scoring

Hybrid scoring lets you layer in metadata-based weighting to boost results that are not only relevant but also timely or authoritative. For example, you might prioritize documents created in the past three months or authored by a specific team. This kind of hybrid approach can significantly improve both the perceived and actual usefulness of the results.

Suppose an employee searches for "incident response policy." Without hybrid scoring, they might get an outdated 2019 PDF. But with hybrid scoring, the 2025 version rises to the top.

Tailoring your scoring function to reflect business context (e.g., favoring security docs from the InfoSec team during a policy audit) makes your retrieval system smarter and more aligned with user expectations.

Maintaining coverage and diversity

If all your chunks are too similar, you risk narrowing the field too much and missing out on edge cases or alternative perspectives. A common reason for this is when training data is overly skewed for a certain category, such as marketing or engineering. Inevitably, the results will be off the mark.

To counter this, there should not only be diverse datasets but also filtering of near-duplicate chunks or sampling from long-tail documents. These adjustments help your system surface less obvious but potentially critical insights.

Scaling through partitioning

As your corpus grows, retrieval speed can suffer, such as when queries include complex filters. One solution is partitioning: segment your content based on metadata dimensions such as source system, department, or creation date.

At query time, this allows you to restrict the search to the relevant partitions. This helps to reduce overhead and improve speed. For example, if a user is looking for "deliveries between April 1 and June 30," there's no need to scan five years of HR policies. Partitioning helps you stay focused and fast.

In practice, relevance and coverage aren't opposing forces. They're levers. With the right chunk sizes, tuned index parameters, hybrid scoring, and smart data segmentation, you can build a retrieval system that consistently delivers high-quality answers without compromising on scope or speed.

Common Pitfalls When Using PostgreSQL as a Vector Store

Using PostgreSQL as the backend for embedding retrieval, as with tools such as pgvector, offers a streamlined architecture. But it's not without trade-offs. If you're building or scaling a system around PostgreSQL as a vector store, here are a few common pitfalls to be aware of.

Over-filtering leads to sparse results

It's tempting to apply strict metadata filters after running your vector search. An example is: "only policy documents from last month." But if you're not careful, these filters can shrink the result set so much that you end up with little or nothing. Say your index returns 40 nearest neighbors, but only 10% of chunks match the filter. You're effectively down to four usable results. That's not a retrieval strategy; that's a bottleneck.

Fortunately, newer versions of pgvector (from 0.8.0 onward) help mitigate this with iterative scanning. If initial results are too few, the retriever keeps probing until it either finds enough valid candidates or hits a configured threshold. Still, it's better to structure your filters and queries to avoid hitting that wall in the first place.

EDB PG AI adds another safeguard here: AI Factory lets you design hybrid retrieval pipelines that blend vector similarity with metadata filters. This ensures you don't end up with "zero results" when filters are too strict, while still prioritizing the most relevant and recent data.

Index builds can be resource hogs

HNSW indexes are great for fast, approximate vector search. But building these indexes, especially across millions of chunks, can require significant memory and CPU. In some cases, you may be looking at build times measured in hours, not minutes.

Tuning `maintenance_work_mem` can help, and so can upgrading. Starting with PostgreSQL v0.6.0, parallel index builds were introduced, shaving down build times by an order of magnitude.

Regardless, if your system is regularly ingesting large batches of new data, you'll want to plan around this overhead.

This is where EDB PG AI's observability layer comes in: real-time metrics on indexing performance and resource utilization give teams clear visibility. By monitoring these bottlenecks directly inside the control plane, you can tune or scale before index jobs impact production workloads.

High-dimensional embeddings can bog down performance

Using embeddings with 1,536 dimensions—common with some OpenAI or BERT-derived models—offers deep semantic context. But that richness comes at a cost. More dimensions mean more storage and longer query execution times. You don't always need the full embedding. Depending on your use case, reducing the dimensionality or using half-precision types such as `half_vec` can significantly reduce storage and improve speed, often without a noticeable drop in retrieval quality.

EDB PG AI makes this easier by supporting efficient storage types and providing observability into vector query latency and storage impact, so you can see when dimensionality is becoming a drag.

Stale embeddings = outdated answers

Of course, if your embeddings are stale, your copilot is effectively looking at an old snapshot of your knowledge base. This is a

common failure point in systems that don't support incremental updates. As mentioned earlier in this chapter, you can use automation techniques to address the problem.

With EDB PG AI, AIDB ties embedding refresh directly to CDC events, so new data automatically triggers reembedding and reindexing. This keeps your knowledge base synchronized without manual intervention or custom glue code.

From Knowledge to Conversation: Building the Copilot

In our setup, every user query triggers a combination of metadata filtering and semantic search using embeddings. We then craft prompts designed to guide the LLM toward accurate, efficient, and traceable responses. It's a straightforward playbook, and it works. RAG remains a go-to architecture in enterprise environments because it strikes a balance between factors such as strong performance and easier implementation.

But let's look more at this process.

Beyond Chat: A Modular Toolkit

We wanted our copilot to adapt to different types of user needs, so we built a set of modular tools that can be invoked depending on the intent behind the query. This flexibility also makes it easier to expand later without overhauling the whole system.

One example is our *summarizer*. When someone wants a quick overview, this module pulls the most relevant content snippets and prompts for generating clear bullet points or an executive-style summary. The key is tight alignment between what's retrieved and what's generated.

Then there's the *explainer*. This one kicks in when users ask things such as, "Why does this work?" or "How was this set up?" It pulls from internal documentation, inline comments, or any content with explanatory context and prompts the model to synthesize an answer grounded in what's there.

For technical requests, such as "Generate a SQL statement to list employees who were hired in the past 30 days," we hand off to the text-to-SQL generator. This tool pulls relevant schema details,

table definitions, and examples, then builds a prompt that guides the model to generate a read-only SQL query. The query is validated against the schema and reviewed for safety before being returned, along with a plain-English explanation.

Because everything is routed through intent detection, it's easy to plug in new modules—whether it's for document translation, code reviews, or policy checks. The pipeline remains the same: retrieve, prompt, post-process. This consistency makes the whole system easier to maintain and extend.

Guardrails, Fail-Safes, and Earning Trust

For our internal support copilot, we've built in multiple layers of protection.

On the input side, we sanitize queries to catch potentially dangerous operations, and we screen for prompt injection tactics, such as those that might come in through UI elements. Then on the output side, we enforce schema validation, run plausibility checks, and verify citations. If a response doesn't match up with what was retrieved, we block or flag it.

We've also baked in fallback behavior. If the system isn't confident in its answer, it'll say so, or hand off the query to a human agent. This helps avoid the kind of overconfident, misleading responses that erode trust. Everything the copilot does is logged: the query, what was retrieved, the final prompt, and any guardrail triggers. Users can even opt to "see source" so they know exactly where the information came from. Finally, we've automated recovery paths for common failure modes: API errors, schema mismatches, or policy violations. These are caught early and handled through retries or validations so they don't reach the end user. Think of it like a layered defense system. Every step is designed to catch issues before they become problems.

We're also constantly tracking performance metrics, such as how long retrieval takes, how often fallbacks are triggered, what guardrails are hit, and where human handoffs happen. These insights feed directly into our development cycles, helping us tighten up prompts, improve accuracy, and make the system more resilient over time.

Shipping It: Deploying the Copilot Internally

The best infrastructure choice really depends on what your organization values most. For teams with strict compliance or data residency requirements, hybrid or on-prem deployments are often the way to go. Tools such as WideBot offer flexibility on that front, supporting cloud, on-prem, or hybrid setups that plug into existing enterprise systems and route data securely. It's about finding a setup that checks your boxes, in terms of performance, security, and compatibility.

No matter how you deploy, there are a few best practices worth calling out:

Plan for scalability
: Whether you're containerizing services or using an auto-scaling setup, you'll want infrastructure that can be flexible during high-traffic periods without falling over.

Separate workloads cleanly
: Retrieval-heavy operations such as vector search can be isolated to their own databases or replicas. This keeps them from interfering with core transactional systems.

Use secure authentication
: Tie into your organization's identity provider (via SAML or OAuth) to make sure both users and backend services are properly verified.

Encrypt everything
: Use TLS for data in transit, encryption for data at rest, and strong role-based access controls.

Rolling Out the Copilot on WhatsApp and the Web

To make sure the copilot is easy to reach, we deployed it in two familiar places: WhatsApp and a web-based UI. That way, employees can engage however they prefer.

For WhatsApp, we used the Business API and routed incoming messages through a secure backend (Twilio, Gupshup, and BotPenguin are some of the options). Once messages hit our system, they're authenticated, processed through the RAG pipeline, and returned in

the same friendly conversational format users expect from chat. The bot offers greetings, smart follow-up prompts, and knows when to escalate to a human when it's out of its depth.

On the web side, we mirrored that experience but added more bells and whistles, such as single sign-on, citation previews, feedback tools, and latency tracking. The goal is to deliver a smooth, transparent, and reliable experience, no matter the channel.

Learning from the Field: What We Track and How We Adapt

We don't treat the copilot as a finished product. It's more like a living system that gets better over time. After all, we're constantly learning from how people actually use it.

Here's how that works in practice: we collect structured feedback (thumbs up/down, short surveys) and log every interaction end-to-end. This includes the user query, what was retrieved, what the model said, whether a fallback was triggered, and timestamps for everything. Then we use NLP tools to comb through that data and find patterns. For example, maybe there's a recurring misunderstanding, a section of stale knowledge, or a certain type of question that keeps tripping things up.

Once we've spotted issues, we make targeted updates such as tweaking prompts, refreshing the knowledge base, refining filters, or shoring up the guardrails. We usually roll out changes in stages, often A/B testing them first to make sure they actually help. And transparency stays front and center. Users can always check where an answer came from or see why something got escalated. We also track metrics including fallback frequency, citation accuracy, and user satisfaction to keep the system moving in the right direction.

It's not glamorous work, but this kind of tight feedback loop—observe, analyze, improve—has made a big difference in keeping the copilot useful and trustworthy as it evolves.

From Capabilities to Agents: What Comes After the Copilot

Rolling out an internal support copilot is a good starting point. For many enterprises, it's already delivering measurable impact across

workflows. But once that foundation is in place, it opens the door to something more ambitious: customer-facing AI and agent-based automation.

Yes, it's a big step up in complexity. But by this point, your team will have gained valuable hands-on experience deploying generative AI in production, such as what works, what breaks, and what it takes to keep a system reliable. That's knowledge you can build on.

The next phase involves exploring more advanced architectures—namely, AI agents. These systems go beyond passive Q&A. Instead, they're designed to take specific actions with minimal human oversight. Think less chatbot, more digital teammate.

For example, one agent might handle knowledge summarization, another might route support tickets, and a third might monitor system health and trigger alerts or remediation steps. The point is, these agents are specialized, purpose-driven, and designed to work as part of a larger system.

Real Automation: When Agents Start Delivering Value

To be truly useful, agents can't operate in a vacuum. They need deep integration with your existing stack, with tools such as Jira or ServiceNow for ticketing, GitHub Actions or Jenkins for CI/CD, and observability platforms including Dynatrace for monitoring.

Picture this: a user submits a support issue. An agent extracts key details, creates a ticket, updates the requester, and begins triaging—all without human involvement. Another agent monitors logs, filters out noise, and runs predefined remediation workflows if needed.

Behind the scenes, an orchestration layer manages the flow, assigning tasks, coordinating agents, resolving dependencies, and tracking results. It's like a conductor ensuring every agent plays its part in sync, even when things don't go exactly as planned.

A key enabler is model context protocol (MCP), introduced by Anthropic in late 2024. MCP provides a standardized way for LLMs to interact with external tools and applications using natural language. It's essentially a bridge between LLMs and enterprise systems. And it's quickly gaining traction.

Thousands of MCPs are already available, making it easier to wire agents into real infrastructure. That said, the technology is still young. Security and permissioning remain open challenges, and enterprises should tread carefully. Google's emerging Agent2Agent (A2A) protocol offers another approach, and the space is evolving quickly.

Orchestration: The Intelligence Behind the System

As agents grow more capable, orchestration becomes the key to scaling. This intelligent layer interprets goals, selects the right agents, manages data exchange, enforces guardrails, and tracks execution status in real time.

In practice, this means:

Automating end-to-end workflows
: From triggering a support ticket → retrieving relevant knowledge → assessing SLA priority → delivering a resolution or escalating if needed.

Learning from experience
: Agents log their actions and results, allowing the system to adapt based on what's working and what isn't.

Operating across your data estate
: Whether it's live telemetry, source code, or incident reports, agents can act on diverse data to generate insights or take action.

Conclusion

Agent-based systems hold enormous potential, but they're still in the early days. There are real risks, especially around safety, reliability, and governance. Enterprises exploring this space should move carefully, pilot with purpose, and always design with traceability in mind.

Still, the direction is clear: over time, AI won't just assist—it will act. And organizations that start building that muscle today will be better positioned to unlock the next wave of intelligent automation.

CHAPTER 8

Your Journey and Your Future: Predictions for the (Possible) Future of This Technology, from Eight Experts

PostgreSQL is over 30 years old, yet it is the most loved and fastest-growing data management platform on the planet. It's rare that something so established can capture the zeitgeist of the moment with AI, and so successfully worldwide.

Something good is happening for PostgreSQL with the collisions of AI and data and the desire by enterprises to be their own AI and data platforms within the next three years. As much as 95% or more of major enterprises globally believe they could be Amazon-like in that time frame.[1] This is incredible optimism, given that the first waves of digital transformation extended over a 10-year-plus time period. However, it's clear how deeply GenAI and agentic applications are taking root across enterprises globally.

PostgreSQL is playing a vital role in that new landscape. Daniel Kahneman's prospect theory argues that people—and by extension, organizations—do not make decisions based purely on maximizing

1 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. *https://www.enterprisedb.com/sovereignty-matters*.

utility. Instead, their choices are shaped by perceived gains and losses, often in ways that defy rational expectations.

We can apply this same principle to AI and data, and PostgreSQL's critical role in the evolution of these new intelligent applications. According to Michael Gale, *Wall Street Journal* best-selling author of *The Digital Helix*, a way to think about this is with the VUCA framework. It describes volatile, uncertain, complex, and ambiguous environments or moments. VUCA worlds require a level of responsiveness to what is happening around us. A world dominated by AI as a frontline choice, especially agentic and GenAI, needs highly agile and responsive systems. Learning systems will win as data and AI in motion become the norm—as will living and thriving in a VUCA world. The capacity to learn and adjust will define thrivers and nonthrivers.

PostgreSQL's capacity as a platform for iterative learning is better suited to this future world than its original world of the 1990s because of its ACID capabilities and the fact that it is thriving globally and with agentic and GenAI. Moreover, the open source architecture is designed to take the best ideas from around the world and bring them to the good of the many.

In some ways, PostgreSQL will replicate the world we all live in. Once we thought about choosing a career path and evolving through jobs. Now we imagine gaining skills that we can apply across a range of careers over a lifetime.

GenAI and agentic AI are force multipliers for change, and PostgreSQL is poised to benefit. This system is rapidly becoming a de facto data management platform for dealing with a multitude of needs because of the characteristics we've described throughout this book. This is the essence of VUCA adaptability that PostgreSQL naturally has in its architecture and DNA.

To this end, we asked experts to predict the shape of things to come for PostgreSQL as it increasingly becomes an agent of change for our world.

Each contributor has expertise and opinions that can fit together or, on some occasions, be diametrically opposed. We present these opinions as much for their thought-provoking nature as for the time each expert in data management, AI, or the organizational dynamics of PostgreSQL has invested in thinking about the future.

How Will AI Impact Relational Databases?

Robert Haas, PostgreSQL major contributor and committer, and Chief Architect, Database Servers, EDB

Robert Haas brings a pragmatist's perspective. Where others forecast wholesale reinvention, Haas reminds us that the near-term reality is more incremental. He points out that while LLMs excel at interpreting natural language, they are not yet replacing the mathematical or algorithmic optimization foundations of databases. His view underscores an important thread in this book: the adoption curve of AI in enterprise systems is evolutionary before it is revolutionary.

The impact of AI on relational databases may not be what you expect. Let's consider the possibilities.

First, while it's true that AI could, in some cases, be used to produce database recommendations, and probably will be used for that in some cases, the great promise of LLMs is their superior understanding of human language, not their ability to do math or statistics. Perhaps one day, AI will be better able than traditional algorithms to figure out when database operations should be performed, but that day has not yet arrived.

Second, let's look at storing AI-related data, such as vectors, in the database. This is a real use case, and an increasingly popular one that has led to the rise of extensions such as pgvector. Luckily, a great strong point of PostgreSQL is its extensible type system. For the most part, a new data type can be plugged into the system quite easily by an extension, with minimal impact on the rest of the system. Indexing is an exception: new data types may require new indexing strategies for good performance. However, PostgreSQL's indexing system is also relatively extensible. A new operator class can be added to an existing index access method, such as Generalized Search Tree (GiST) or Generalized Inverted Index (GIN), with a very small amount of new code, and entirely new index access methods can also be added with somewhat more effort. Therefore, while this use of PostgreSQL can be expected to drive the development of some new technology, it seems unlikely that it will reshape the core database in any deep way.

Finally, storing inputs to or outputs from LLMs in the database requires no special support within PostgreSQL at all. An end-user conversation with an AI chatbot agent can and should be recorded

in the database just like a conversation with a human agent. Conversely, text that will be read and processed by an AI chatbot can be stored just like text that will be read by a human being. It seems very likely indeed that this will turn out to be the single most common way that PostgreSQL is used with large language models, just as it is the single most common way that PostgreSQL is used with so many other technologies. In the end, from the point of view of the database, large language models are simply one more way of generating data, and, as with all other data, that data will need to be stored, indexed, and retrieved. PostgreSQL, as the world's most advanced open source database, is here to help!

Haas reminds us that the future isn't always disruption on a grand scale. Sometimes, it's continuity that matters most. PostgreSQL doesn't need to be reinvented to accommodate AI. Its extensibility means it can absorb new workloads such as vectors and embeddings without fracturing its core. That kind of evolutionary resilience is its own form of innovation, the steady foundation on which others will build.

PostgreSQL + AI Extensibility

- EDB's 2025 research found that 33% of enterprises are considering PostgreSQL as their AI data platform of choice across transactional, analytic, and AI workloads.[2]
- Extensions such as pgvector are already making PostgreSQL the natural home for vector data, demonstrating exactly the type of incremental extensibility Haas highlights.
- As Haas has said in conference talks, the ability to "add new operator classes with minimal code" is a feature few databases can match—and it's why PostgreSQL adapts quickly to new paradigms without breaking its core.

2 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. *https://www.enterprisedb.com/sovereignty-matters*.

How AI Will Transform Access to Relational Data

Bruce Momjian, PostgreSQL core member and Vice President, PostgreSQL Evangelist, EDB

If Haas emphasizes continuity, Bruce Momjian points to transformation in accessibility. For decades, relational systems have endured every wave of disruption—object databases, XML, NoSQL—because they provide precision where precision is nonnegotiable. However, Momjian believes AI will change who gets access to that precision.

Relational system design has been around for 55 years, and people keep wondering when it will end. Object databases, XML databases, and NoSQL have not been able to kill them, so maybe AI will. For all the excitement around AI, it is fundamentally imprecise. At many levels, you need precise data, and relational systems have proven ideal for this for over 50 years.

However, I think a human text interface to relational system data will be a game-changer for enterprises. While SQL looks like English, it requires training to use and an understanding of the database schema. RAG and GenAI promise to make relational data accessible to novice users, allowing for new insights.

It's true that today we still have large teams who are responsible for collecting and summarizing digital data to create actionable information. It is clear that semantic search, RAG, and generative AI will allow people who need to make decisions to more easily query relevant data, leading to more accurate decision-making and eliminating the role of intermediate information gatherers.

The implication is profound: relational databases won't be displaced by AI. They will be democratized by it. Momjian points to a future in which relational systems continue to provide the precision enterprises require, but with a radically lower barrier to entry. In this vision, PostgreSQL becomes not just a system of record only for experts but an accessible decision engine for every business user.

Relational Durability + Democratized Access

- In EDB's global research, 95% of major enterprises said they aim to build their own sovereign AI and data platforms within three years—a future in which broad access to precise relational data will be critical.[3]
- RAG patterns built on PostgreSQL AI illustrate exactly what Momjian describes: combining semantic retrieval with structured relational data so users can query in natural language, without needing to know SQL.
- Momjian has often emphasized in conference talks that "precision is why relational systems endure." In the AI era, that precision, paired with accessibility, is why relational systems will thrive.

GenAI Will Create a Data Management Flywheel Effect with PostgreSQL

Michael Gale,* Wall Street Journal *and Amazon number-one best-selling author on digital transformation

Michael Gale draws on decades of studying digital transformation to show how AI will reshape organizational leadership itself. Just as past waves created new C-suite roles and governance models, he sees AI driving a proliferation of new executive functions and a deeper interdependence between data and decision-making.

A world led by AI will have a lot of AI leadership roles as more corporations have a Chief Data Officer, a Chief AI Officer, and maybe a Chief Digital Officer. We know from the last waves of digital transformation that organizations that constructed two or more relevant job titles, such as Chief Data Office and Chief Transformational Officer, had a 65%+ higher chance of economic success than organizations with just one of these titles.[4] That same pattern,

3 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. https://www.enterprisedb.com/sovereignty-matters.

4 Gale, Michael, and Chris Aarons. *The Digital Helix: Transforming Your Organization's DNA to Thrive in the Digital Age*. Austin, TX: Greenleaf Book Group, 2017.

or pressure, will work here as the processes and power centers need multiple advocates and owners of success.

We will have more AI officers, more data officers, and eventually, agentic and GenAI officers. This means we will see an increased pressure for real-time development, feedback, and adjustments from the GenAI, agents, and models across functions.

The most successful agentic and GenAI models now (economic and other ROI models) have a level of cross-functional density in their agentic that is twice that of their competitors and 50% more mainstream. We call them the "deeply committed." If they are setting the pattern for the future, then they will need a data management architecture, a platform that is open source and architected to handle a vast array of unstructured and structured data for multiple business applications, and with ease. As AI becomes a central and first-stop mantra for solving business problems, PostgreSQL's ACID architecture makes it the perfect whiteboard and factory for this process.

In this new world, a vast array of tasks will need to be automated, including vector search across distributed computing environments at core, edge, customer, and supply chain levels. Data and AI will be in near-constant motion and require a data management backbone that is agile, automatable, and open source. PostgreSQL, supported by its global community of experts and extensions, is well suited to be that intelligent data platform.

Gale argues that leadership and architecture will coevolve, as enterprises appoint AI officers and push for cross-functional accountability. PostgreSQL provides the trusted, extensible platform to support this new flywheel for continuous innovation.

Leadership + Data Management

- EDB's 2025 research identified the "deeply committed," 13% of enterprises, as seeing two to five times ROI from AI adoption, echoing Gale's argument that density of commitment is what separates leaders from laggards.[5]
- The ability of EDB PG AI to unify transactional, analytical, and AI workloads illustrates the data management "backbone" Gale envisions.
- Gale's research in *The Digital Helix* shows organizations with multiple digital leadership roles had a 65% higher chance of economic success, a pattern repeating with AI governance today.

The Emergence of Hybrid AI-Human Database Management

Thomas Koulopoulos, Chairman, Delphi Group, author, and founder of the Babson College Center for Business Innovation

Thomas Koulopoulos, a futurist and longtime chronicler of technological shifts, challenges us to consider whether AI will ultimately outpace PostgreSQL or partner with it. His perspective surfaces existential questions of trust, governance, and control in an AI-driven data landscape.

The fundamental question that I grapple with is whether AI will leapfrog PostgreSQL or whether it will work in collaboration or in concert with it. And while it's difficult to project that beyond five years, I suspect that certain things are going to influence that. In the near term, it's clear to me that PostgreSQL will continue evolving rapidly. I think there is a tremendous gravity to open source, generally speaking, and we're seeing that across the board, so I don't think that will be any different in the realm of databases.

5 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. https://www.enterprisedb.com/sovereignty-matters.

But the real issue here isn't its capabilities. It's whether the community-driven approach, that ethos, and that model, can accelerate fast enough, or whether it will eventually become a bit of a bottleneck in comparison to what AI can do. AI-driven databases can self-optimize, self-repair, and self-modernize. Modernization is a huge part of database management, and that could make PostgreSQL, or any database for that matter, feel pretty much outdated and be relatively useless in comparison to the capabilities of an AI-optimized and auto-modernizing database.

My sense, however, is that what we'll end up with is a hybrid model, at least over the course of the next four or five years. The challenge isn't just a technological one—it's about whether open source governance can adapt fast enough in this sort of AI-driven world.

I think what we have is a crisis of a black box that's coming at us. AI's ability to generate code is evolving so rapidly, and so will its ability. This isn't just creating a new layer of complexity (which it will). It's also creating a black box that is so sophisticated that developers simply won't fully understand what's going on and how their own systems are optimized and managed—even architected, for that matter.

But it doesn't have to be black and white; it doesn't have to be absolute. Just as companies today still balance cloud and on-premise strategies, organizations will also decide whether to retain human control over database performance and logic or cede it to AI. Some domains—finance, defense, healthcare—simply won't surrender. They won't abdicate full control to AI, for regulatory issues or just for basic security issues; they'll want to maintain control.

So the future of databases and database management won't be about writing just queries. It'll be about deciding where and when to trust an AI-driven automation over one that is human-managed or over one that is hybrid. And again, we're dealing with a spectrum here, and we're going to live in the gray space in between those two extremes, but there will be some cases where a purely human-driven approach with minimal AI is going to be the preferred alternative.

If you ever saw the movie *WarGames* with WOPR (War Operation Plan Response), you'll know the characters could not work out what was a simulated attack versus what was a real-world attack. This conversion of synthesis, trust of the system, and the physical world we live in is going to become a problem.

Up until now, certainly for the entirety of my career, data has been something that we store. It's been relatively static. There have been dynamic elements of it, certainly, but as AI evolves, data will be something that we manufacture.

We're seeing that in synthetic data that's being created by AI today. It's a constantly generated deluge of data that is transformed and adapted in real time—a resilient database that reflects the world as it really is, our business as it really is, the market as it really is, and that moves much faster than any human being could possibly keep up with.

And that shift creates an existential challenge. If AI is constantly producing and modifying data, there's the notion of whether its truth even holds anymore. It gets a bit philosophical, but it's relevant. We will see this new sort of data inflation, where human discretion and discernment are no longer enough to really extract meaningful insights from the data. AI becomes the only entity capable of making sense of the data that it generates. So the philosophical and the ethical implications of this are the critical ones.

In this future of AI-driven organizations and AI-managed and AI-optimized databases, everything is built on shifting sands of data. We will not, as humans, be able to infer what is real and what is not, what is true and what is not. We will have to use AI to do that for us.

Ultimately, PostgreSQL and every other database face a choice. I think for PostgreSQL, that choice is an easier one to make, and because it provides a community-driven model, if we can figure out the governance around that model and manage that well, I think it will work seamlessly with AI and emerging technologies. If not, it will get left behind.

Koulopoulos pushes us to confront an unsettling future: databases no longer just storing truth but manufacturing it. The black box of AI will force enterprises to decide how much control to yield, when to trust automation, and where human judgment must remain nonnegotiable. For PostgreSQL, the open governance model may be its greatest defense—or its greatest test—in a world where the very definition of truth is shifting under our feet.

Governance + Hybrid Futures

- In EDB's global research, 95% of enterprises expect to build sovereign AI and data platforms within three years—reflecting the tension between autonomy and control that Koulopoulos highlights.[6]
- Gartner projects that by 2030, 80% of enterprise data will be machine generated, underscoring Koulopoulos's warning about synthetic data inflation.
- As Koulopoulos often says, "We don't just store data anymore, we manufacture it." This insight frames PostgreSQL not just as a database but as an arbiter of reality itself in the AI era.

The Economics of Data Will Fundamentally Change

Michael Fauscette, founder and CEO, Arion Research; author,* Building the Digital Workforce*; host,* Disambiguation *podcast; Agentic AI Top 10 Thought Leader and Ambassador, Thinkers360

Michael Fauscette looks at AI not only as a technology shift but as an economic one. His view is that the economics of data management—how we store, process, and extract value—are being rewritten by AI-native systems. What matters most is no longer storage but insight.

Traditional data architectures treat all data equally in terms of storage and processing resources. Yet this approach has become economically unsustainable as data volumes grow exponentially.

Rather, there will be a transformation from storage-centric, batch-oriented, schema-dependent systems to intelligent, insight-focused platforms. During the next three years, I predict:

Real-time semantic layers
: AI-native platforms will automatically create and maintain business-relevant views of data that evolve in real time, eliminating the need for predefined schemas and ETL processes.

6 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. *https://www.enterprisedb.com/sovereignty-matters*.

Embedded analytics processing
: Intelligence will be pushed directly to data storage layers, with AI models running where data resides rather than requiring movement to separate processing environments.

Self-organizing data structures
: Data will be automatically organized based on usage patterns and query requirements rather than predefined schemas, optimizing for both structured and unstructured access patterns.

Dynamic data virtualization
: Physical data location becomes irrelevant as AI platforms create unified access layers that intelligently cache and move data based on access patterns and latency requirements.

This is what I see in seven years:

Insight-first architectures
: Data platforms will be designed around delivering insights rather than storing data, with storage decisions automatically made based on the types of decisions the data supports.

Autonomous data economics
: AI systems will automatically determine the economic value of data assets and optimize storage, processing, and retention based on business value rather than technical parameters.

Causal intelligence layers
: Platforms will move beyond correlative analytics to automatically identify causal relationships in data, enabling true predictive and prescriptive capabilities without data scientist intervention.

Cognitive data fabrics
: The distinction between raw data, models, and insights will disappear as platforms maintain a continuously updated cognitive fabric that represents the organization's collective intelligence.

Fauscette challenges us to see data economics as the frontier where AI exerts its greatest force. PostgreSQL and its ecosystem are not just fighting for performance or scale; they are competing to be the economic engine of insight-first architectures.

The Economics of AI Data

- In EDB's global study, 45% of ROI from copilots came not from cost savings but from new opportunities to innovate, reflecting the shift Fauscette describes from storage cost to insight value.[7]
- As AI drives "data inflation," Fauscette's prediction of autonomous data economics highlights why enterprises are pushing for sovereignty: they want control over the value of their data, not just the cost.

AI's Role in the Future of Database Work

***Ayal Steinberg, General Manager, IBM's Growth Segment, and author of* Generative AI for Business**

Paul Zikopoulos, Vice President, IBM Technology Group Skills and Enablement; author of 20 books, including* AI Value Creators *and* Cloud Without Compromise*; and Analytics Insight's Top 100 Global AI and Big Data Influencer

Steinberg and Zikopoulos have long chronicled how data and AI intersect. Their perspective reframes the role of the database administrator (DBA) in the era of transformers and generative AI. Far from obsolete, the DBA becomes the bridge between business needs and AI-driven automation.

We can't stress this enough—that GenAI you're relying on to supercharge your business runs on data *and* on the people and systems that produce it. We've moved past the days when machine learning relied on painstakingly wrangled and labeled data just to produce useful insights. Sure, that era had its wins, but progress was slow because prepping that data took too much manual effort. And it's that very grind that led to the too-often-heard (but true) decree that 80% of a data scientist's time is spent on data preparation.

Everything changed with the arrival of the transformer—a real turning point. Suddenly, AI could start organizing and labeling raw data on its own, without requiring humans to hand-hold every step. That

7 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. https://www.enterprisedb.com/sovereignty-matters.

shift opened the door to the GenAI world we're living in today. Once machines didn't need us to label every single data point; they could train faster and at a scale we'd never seen before (and, truthfully, never imagined). Traditional AI evolved into something far more powerful—and while the idea might sound simple, we show just how big the leap was in Table 8-1.

Table 8-1. A comparison of the initial effort required to begin with traditional AI versus generative AI

Traditional AI	Generative AI
Supervised learning.	Self-supervised learning.
Humans needed to scale.	Compute power needed to scale.
Intense labeling efforts.	Little labeling, just for fine-tuning or dynamic shot-prompting.
Long, manual, tedious, expensive. Slow to get going in a meaningful way.	Quick, automated, and efficient (because of transformers). Get going quickly in a meaningful way.

But there's more to this (and the future) moment than just getting started with AI. Today, AI doesn't just enhance data, it enhances the systems that manage it—and from what we can tell, it's going to do much more. For example, AI is improving the performance of databases, query engines, and other foundational technologies. No more writing optimization hints to get the best performance out of your database: let AI figure it out. No more DBA architecture reviews with the business team where a decision is made to create some sort of materialized view to accelerate end-of-quarter reporting; AI will know how to create it because, over time, it understands the demands of the last week of the quarter. What's more, AI will destroy (logged or not logged), or perhaps back up and *then* destroy, such artifacts, depending on your policies and intent; in short, it'll do the management for you. Quite simply, in the same way we've seen AI-driven tools transform software development with code assistants—streamlining code generation and improving productivity—we're now seeing the same shift in the domain of database administration.

The future of database work is evolving quickly. AI depends on databases, but it's also increasingly responsible for managing them. Generative AI will surely transform how SQL is written and run. Instead of writing queries by hand, users will describe their intent with all the expressivity of natural language, and the AI will build, manage, and query the database accordingly.

Then does this mean the role of the DBA is dead? Nope, it's just leveling up. Think of it this way: why obsess over every microscopic tweak in a query when a machine can now fine-tune it faster than you can say "table scan"? It's like arguing over the best way to sharpen a quill pen while your tablet is already drafting your emails, booking your meetings, and reminding you to take out the garbage. Let AI sweat the syntax—you've got bigger things to optimize.

And what are those bigger things? One example is the emergence of reasoning models and autonomous agents. As they grow more capable, DBAs will be tasked with integrating the precious data their professional lives are dedicated to into workflows and chains of thought going on within the AI.

For example, one of the hottest topics in the AI world right now is model context protocol (MCP). MCP is the very method DBAs will use to expose data artifacts to LLMs seamlessly. However, someone has to work with the business to understand what should be surfaced and what shouldn't, what kind of access controls should be in place, how access should be audited, and so much more.

The takeaway? AI will no longer sit outside the database—it will be part of it, and that will free up a DBA's time to guide intelligent systems, set policies, and make fewer emergency coffee runs due to some SLA violation.

DBAs + The PostgreSQL Advantage

- In EDB's *Sovereignty Matters* research, 87% of enterprises are still early in AI implementation—meaning the greatest opportunity lies in how technical teams, like DBAs, evolve alongside automation.[8]
- "Deeply committed" organizations already see five times greater ROI from their data and AI investments, showing that skill development and ownership matter as much as technology choice.
- For DBAs, PostgreSQL is the most natural bridge: familiar, extensible, and already at the center of AI-ready data platforms around the world.

The Future of AI ROI: Measuring Return on Intelligence

Michael Schrage, Research Fellow, MIT Sloan School's Initiative on the Digital Economy

Michael Schrage's argument is simple: enterprises loudly declare that "data is an asset," yet almost none can show a KPI for return on data—or a risk-adjusted "return on AI." The gap between what leaders say and what they instrument is the biggest drag on value creation in the AI era. His remedy isn't more rhetoric; it's building virtuous cycles that tie insight to incentive to implementation.

I open almost every executive session with a blunt question: if you swear "data is an asset," what's your KPI for return on data? And while the room nods at the slogan, almost no one can point to a disciplined, risk-adjusted metric—let alone a credible "return on AI." That gap between what leaders say and what they actually instrument is where most of the value leaks out. Too much mimetic aspiration—"let's be Amazon/Google/TikTok"—too little introspection about what that really requires in your context. My remedy isn't rhetoric; it's operationalizing virtuous cycles where AI aligns

8 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. https://www.enterprisedb.com/sovereignty-matters.

self-awareness (the checklists you must attend to) with *situational awareness* (what's emerging that you'd better be ready for).

Here's an example almost everyone still ignores: record the meeting, interrogate the transcript, synthesize and prioritize the decisions, and auto-draft the next agenda. Five minutes of prompting, outsized productivity. I've watched a conservative bank turn this into a quiet revolution; yet I'd bet less than 1% or 2% of the Fortune 1,000 does this reliably. This is not just low-hanging fruit; it's money growing on trees. Pick it. Make the "meeting flywheel" a default behavior, not a pilot.

This failure points to something deeper: a *crisis of capability*. We've built organizations that satisfy by design. "Good enough" has become the cultural default. But good enough is no longer good enough. We need to *annihilate average*—to use AI to surface the best of what we can be, not to just automate the routines of who we already are.

Zoom out, and the human question looms larger than the technical one. By 2030, most people's devices will be "smarter than they are," which makes epistemic (active) humility the meta-skill for getting consistently reliable value from LLMs and agents. Ask first, "What is my ignorance?" and design incentives that reward better questions, not louder answers. I've argued for years that "self-ware" beats "software": don't chase a single "better self" but cultivate multiple purposeful selves—analyst, operator, supporter—coached by generative systems. Better boards and CEOs will use these lenses to create constructive tension with management—governance as an accelerator of learning, not a brake.

In Schrage's view, AI's true contribution isn't insight but introspection. The organizations that win will be those that measure what they learn, not just what they earn. AI will distinguish companies that instrument learning from those that simply talk about it. His view reframes PostgreSQL as a feedback engine, turning awareness into action and data into measurable behavioral change.

The Return on Data and AI

- In EDB's global research, only 13% of enterprises—those "deeply committed" to AI—measure ROI across both data and AI initiatives, seeing two to five times greater value creation.[9]
- Schrage's call for "instrumenting return on data" mirrors the movement toward quantifiable AI governance and hybrid performance metrics in sovereign data strategies.
- PostgreSQL's extensibility makes it the perfect substrate for this new feedback economy—where platforms don't just store data, they measure the return on intelligence itself.

Navigating the Quantum Future

Quantum computing represents one of the most disruptive shifts in technology since the invention of the microprocessor. It has the potential to unleash exponential gains in processing power.

Yet there needs to be caution. The same capabilities that could accelerate breakthroughs can also pose serious risks to today's digital security and data protection infrastructure.

So, for the last predictions in this chapter, Ayal Steinberg, Paul Zikopoulos, and Michael Gale highlight both the urgency and opportunity of this transformation. Steinberg and Zikopoulos show us that the risks are already present, while Gale illustrates the strategic role of PostgreSQL as a bridge between classical and quantum architecture.

The Quantum Advantage

Ayal Steinberg and Paul Zikopoulos

Ayal Steinberg and Paul Zikopoulos emphasize that the security threats from quantum computing are not years out. They are actually emerging today. This is why enterprises must act now with urgency to understand, identify, and address them. The focus must be on becoming quantum-safe.

9 EnterpriseDB. *Sovereignty Matters: A Global Blueprint for Sovereign, Agentic, and Generative AI*. September 18, 2025. https://www.enterprisedb.com/sovereignty-matters.

Quantum computing will explode in the next five years. This style of computing centers around the qubit, which fundamentally behaves differently from the bit. Specifically, quantum computing operates at the subatomic layer of the world we live in, and it uses all kinds of physics that are well outside the scope of this book (superposition, entanglement, and so on) to give what we call the quantum advantage to specific workloads.

I expect the urgency of getting your company to be quantum safe, which refers to the ability of a quantum computer to quickly crack traditionally encrypted data. After all, the very approach used to encrypt data traditionally wasn't based on some secret code; it was formulated around the amount of impossible work it would take to reverse engineer the math to read the data.

That's going to change with quantum. You can be sure data (perhaps yours) has already been stolen that can't be read today, but tomorrow it will be easy. This is referred to as "steal now, crack later." Because of their different versions and distributed nature, it's common to see all kinds of encryption algorithms used inside and outside of databases. All companies need to start taking inventory of what algorithms they are using today (that's a lot of work in and of itself) and make plans to migrate their business to use quantum-safe algorithms.

Becoming Quantum Safe

- Quantum threats are emerging now, not years away—bad actors are already stealing encrypted data to decrypt later ("steal now, crack later").
- Enterprises must inventory current encryption algorithms and begin migrating to post-quantum cryptography to stay secure.
- The move to quantum-safe mirrors past transformations such as cloud and AI readiness—but this time, delay risks decades of data exposure overnight.

The Bridge for the Quantum Future

Michael Gale

Michael Gale highlights how PostgreSQL, with its extensibility and open source foundation, becomes more than a database. It's the bridge between classical and quantum architectures. As organizations experiment in a VUCA world, PostgreSQL provides the flexibility to integrate post-quantum cryptography and quantum-enhanced capabilities.

As more and more data is needed to drive decisions and quantum computing becomes a real possibility, PostgreSQL offers several important data management capabilities for many of these analytical and AI workloads that will be essential for quantum computing success in the future. Its extensibility and its open source nature make it the most extensible database possible. Elements such as custom storage engines, complex query optimizers, and the use of indexing methods, all essential components for quantum success, are natural components of PostgreSQL for the inevitable design of a hybrid classical–quantum architecture. This is especially true as organizations need to experiment and learn (in the VUCA world) with both advanced query optimization and the inevitable need for parallelism.

The potential integration of quantum-enhanced extensions and post-quantum cryptography with a hybrid, enterprise-ready architecture makes it the perfect data management bridge between your own sovereign AI and data platform to the quantum computing world, especially if you will need to leverage quantum speed for high-volume transactional workloads.

PostgreSQL as the Quantum Bridge

- PostgreSQL's open, extensible architecture makes it a natural bridge between classical and quantum systems.
- Its flexibility supports post-quantum cryptography, hybrid architectures, and advanced query optimization for parallelism.
- In a VUCA world, PostgreSQL enables organizations to experiment, adapt, and evolve safely toward quantum-enhanced, sovereign AI platforms.

Conclusion

While AI will introduce dramatic shifts in how data is generated, accessed, and managed, relational systems such as PostgreSQL will not be displaced. Its extensible architecture makes it uniquely capable of absorbing new workloads such as vectors and embeddings, while its governance model and open source community provide resilience and innovation. In a world where synthetic data, quantum threats, and hybrid AI–human systems redefine trust and decision-making, PostgreSQL is positioned not only to survive but to act as a bridge. It's between humans and AI, between classical and quantum computing, and between the present and the next frontier of data-driven innovation.

In this chapter, we saw insightful glimpses into the future. There are also some key takeaways:

- PostgreSQL thrives because it doesn't need to be reinvented. With extensions such as pgvector, you can store and query AI-related data such as embeddings without disrupting your existing workflows. Its modular design means PostgreSQL adapts incrementally and remains useful as new AI use cases emerge.
- Generative AI and RAG are democratizing access to relational systems. Instead of relying solely on SQL experts, organizations can empower broader teams to query data using natural language.
- AI is rapidly gaining the ability to self-optimize and self-manage databases, but complete reliance on automation won't be acceptable in all industries. Highly regulated sectors such as finance and healthcare will require clear policies to determine when to trust AI-driven automation and when to preserve human oversight.
- As the database administrator's role evolves, teams need to understand how to manage AI-augmented data systems. This means shifting the focus from manual query tuning to shaping governance, setting policies, and ensuring that databases integrate effectively into agentic workflows.
- Quantum computing is something that is happening now, especially with the impact on encryption. Organizations must take inventory of their current algorithms, begin transitioning to quantum-safe approaches, and explore how PostgreSQL's extensibility can support analytical and AI workloads.

We've reached the end of this book, and we hope you've enjoyed the journey. What you've read is still very much a work in progress, as AI is moving incredibly fast. While writing this business guide, we had to make adjustments and course corrections along the way. It's what's normal in the VUCA world.

But the dynamism is what makes it so exciting. The opportunities are vast, and PostgreSQL sits at the center of it all. To better act on these opportunities, here are a few key points to guide your path forward:

- Adopt a platform mentality for AI and data. You need to move beyond pilots and experiments into an integrated strategy that touches every part of your organization. This is how you unlock transformative productivity, efficiency, and innovation.
- Prioritize high-quality data and strong governance. Without it, AI initiatives will stall or fail. PostgreSQL's ACID compliance, security, and governance features ensure the foundation is solid.
- Transform data access with RAG and GenAI. With RAG, you ground LLMs in your proprietary, real-time data for accurate, context-aware results. Extensions such as pgvector let you store vectors directly in PostgreSQL, cost-effectively and at scale.
- Ensure a secure, compliant, and sovereign AI environment. This means leveraging PostgreSQL's features for RBAC, encryption, and regulatory compliance to protect your data and meet evolving standards.
- Prepare for the agentic future. PostgreSQL's adaptability and ACID compliance make it a natural fit for managing continuous, evolving data streams that fuel intelligent, agent-driven applications.
- Get quantum-ready. Take inventory of your encryption algorithms and migrate toward quantum-safe approaches, while exploring PostgreSQL's extensibility for workloads that may one day demand quantum acceleration.

More than anything, we appreciate the time you've invested in reading this book. Now, it's your turn: take what you've learned, experiment, adapt, and chart your own course. The future of AI and data is wide open. Good luck on your journey.

About the Authors

Tom Taulli (@ttaulli) is a consultant to various companies, such as Aisera, a venture-backed generative AI startup. He has written several books, including *Artificial Intelligence Basics* and *Generative AI*, which covers ChatGPT, GPT-4, and other large language models. Tom has also taught IT courses for UCLA, PluralSight, and O'Reilly. For these, he has provided lessons in using Python to create deep learning and machine learning models. He has also taught on topics such as natural language processing.

Benjamin Anderson is a seasoned technologist specializing in open source data platforms and AI innovation. Anderson drives EDB's portfolio technical strategy, bridging the needs of the individual developer and the enterprise C-suite exec.

With a career spanning over a decade in engineering and leadership roles, Anderson has a proven track record of unifying complex technical systems and strategic business outcomes. His experience in open source technology and data sovereignty fuels his mission to empower enterprises to innovate at scale while maintaining control and integrity over their data.

Jozef de Vries spearheads innovation and product development across EDB's hybrid portfolio, helping enterprises unlock the power of PostgreSQL for modern data challenges. A visionary in database technologies, de Vries joined EDB in 2020, bringing nearly 15 years of leadership experience from IBM, where he built and scaled the IBM Cloud Database development organization through a strategic combination of organic growth, mergers, and acquisitions.

With a passion for driving transformational change through open source technology, de Vries is committed to helping businesses navigate the complex intersection of AI, data, and infrastructure to achieve long-term success.