O'REILLY®

Vector Databases

A Practical Introduction

Nitin Borwankar

"*Vector Databases* by Nitin Borwankar demystifies the technology powering modern AI—from embeddings to semantic search and RAG systems. If you're building AI applications today, this is a book you need to understand and apply these concepts in practice."

Tom Taulli
Author of *AI-Assisted Programming*

Vector Databases

The AI revolution is here, and at its core lies a game-changing technology that most developers haven't fully explored: vector databases. From powering semantic search to enabling large language models (LLMs) and generative AI, vector databases are reshaping how we build applications with unstructured data like text, images, and audio. But how do you go from curious to capable with this vital technology? That's where this book comes in.

In this hands-on guide, author Nitin Borwankar takes you through the why, what, and how of vector databases, starting with the basic theory behind vector embeddings and progressing to building applications with real-world tools. You'll learn about Word2vec, how to convert open source SQL databases like SQLite3 and PostgreSQL into vector databases, and how to integrate them into retrieval-augmented generation (RAG) applications. Whether you're a Python developer, data engineer, or ML practitioner, this book gives you the foundation to leverage vector databases confidently in your AI projects.

- **Understand the connection between vector databases, embeddings, and LLMs**
- **Learn practical approaches for transforming SQL databases into vector databases**
- **Build RAG applications for both personal and enterprise use**
- **Apply vector databases to solve real-world AI challenges**
- **Learn how to use vector databases with LLMs to build applications**

Nitin Borwankar is a seasoned data scientist and database professional, known for his contributions to data education and open source machine learning tools over a career spanning three decades. He is a frequent conference speaker, offering a pragmatic approach to AI and large language models.

DATA

US $69.99 CAN $87.99
ISBN: 978-1-098-17759-1

9 781098 177591 56999

O'REILLY®

Vector Databases

A Practical Introduction

Nitin Borwankar

O'REILLY®

Vector Databases
by Nitin Borwankar

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 (*http://oreilly.com*). For more information, contact our corporate/institutional sales department: 800-998-9938 or *corporate@oreilly.com*.

Acquisitions Editor: Andy Kwan
Development Editor: Sara Hunter
Production Editor: Beth Kelly
Copyeditor: Audrey Doyle
Proofreader: Krsta Technology Solutions
Indexer: WordCo Indexing Services, Inc.
Cover Designer: Susan Thompson
Cover Illustrator: José Marzan Jr.
Interior Designer: David Futato
Interior Illustrator: Kate Dullea

April 2026: First Edition

Revision History for the First Edition
2026-04-06: First Release

See *http://oreilly.com/catalog/errata.csp?isbn=9781098177591* for release details.

978-1-098-17759-1

[LSI]

Table of Contents

Preface

To repeat a well-worn phrase, this is the book I wish I had when I started exploring vector databases. I was looking for a single source of basic theory and practical applications of vector databases, but I found none.

Existing vector database resources are fragmented across vendor documentation, blog posts, YouTube videos, and GitHub code. I wanted practical working examples rather than theoretical coverage. I also have a personal preference for proceeding systematically from basics to simple working systems to help organize data locally—especially private data.

Thus began my journey to pull together information from multiple sources and multiple directions into a coherent whole. And this book is what emerged. It is meant to be useful to the intermediate developer, from the theoretically minded to the "just show me the code" type. There's something for everyone.

The book follows a clear pedagogical arc: concepts → tools → applications → future directions. Each chapter builds on previous ones while remaining practical with working code examples. The progression moves from understanding vector databases to building working applications that are meant for personal use, hands-on interaction, and extension by developers who want to build personal data management systems. It culminates in a forward-looking proposal for industry standardization of a Vector Query Language modeled after SQL.

The somewhat opinionated part of this book's approach is a focus on simple hybrid vector relational databases—created by adding vector extension plug-ins to the well-known open source databases SQLite3 and PostgreSQL. The reason for this is that these databases are well known among database developers. Simply adding a data type of "vector" opens up a whole new world of AI data applications. This makes onboarding as low-effort as possible.

It can't be emphasized enough that these applications are for hands-on learning and not for web-scale production deployment. Modify them as you see fit; make them your own.

And now let's dive into the contents.

What's in This Book

Chapters 1 and 2 establish foundational concepts—vector databases, embeddings, and semantic search fundamentals:

Chapter 1, "Introduction to Vector Databases"

- Introduces vector databases as foundational technology for AI applications, explaining why unstructured data (text, images, audio, video) requires semantic search rather than keyword matching
- Covers the vector data type, similarity search, and key operations (cosine similarity, nearest-neighbor search, vector arithmetic)
- Contrasts vector databases with SQL and NoSQL databases, highlighting hybrid architectures that combine structured metadata with vector similarity search
- Establishes the conceptual foundation for understanding how embeddings preserve semantic meaning in vector space

Chapter 2, "Embeddings"

- Takes a deep dive into the history, evolution, and current usage of vector embeddings with code examples for language model applications
- Explains how embeddings bridge the gap between unstructured data and machine-processable representations, mapping raw data into vectors that capture semantic relationships
- Covers the mathematical foundations and how proximity in vector space reflects semantic similarity
- Provides the conceptual framework needed for practical applications later in the book

Chapters 3 and 4 discuss practical implementations with Facebook AI Similarity Search (FAISS) and SQLite, showing how to build personal-scale search systems:

Chapter 3, "Similarity Search with FAISS"

- Explores FAISS as both a production similarity search engine and a flexible toolkit for building custom vector databases

- Is designed for developers interested in the internals of vector search and those wanting to extend capabilities of systems such as sqlite-vss
- Covers the mathematical foundations and implementation details of efficient similarity search at scale
- Focuses on CPU-based implementations for maximum accessibility

Chapter 4, "Semantic Search with SQLite3"

- Builds a personal knowledge management system using semantic search over Reddit content, demonstrating meaning-based search versus keyword matching
- Introduces sqlite-vss, a SQLite extension that wraps FAISS and integrates vector search into the relational database world
- Combines semantic retrieval with relational metadata filtering in a single SQL workflow
- Uses an "overfetch-then-filter" pattern to handle the fact that metadata filters aren't pushed down into FAISS search

Chapters 5 through 8 progress through increasingly sophisticated applications—ArXiv paper search, local retrieval-augmented generation (RAG) systems, scientific RAG, and personal conversation search—moving from public to private data:

Chapter 5, "Building an ArXiv Paper Search System with PostgreSQL pgvector"

- Constructs a scientific literature search system addressing the challenge of discovering relevant research across millions of papers
- Demonstrates how vector embeddings capture semantic relationships that keyword search misses (e.g., connecting "neural network optimization" with "gradient descent improvements")
- Leverages PostgreSQL with the pgvector extension for enterprise-scale applications
- Shows how to handle technical terminology, structured academic content, and metadata in a unified search system

Chapter 6, "Building a Retrieval-Augmented Generation System with SQLite VSS and Ollama"

- Assembles a complete RAG system combining vector search with local large language model (LLM) capabilities
- Builds an entirely local, private system running on a single desktop using SQLite Vector Similarity Search (SQLite VSS) and Ollama
- Addresses the fundamental LLM limitation: knowledge frozen at training time without access to private or recent information
- Creates a question-answering system that grounds LLM responses in actual retrieved data to reduce hallucinations

Chapter 7, "Building a Scientific RAG System with PostgreSQL and pgvector"

- Develops a RAG system specifically designed for scientific literature with unique challenges: technical terminology, structured content, citation networks, and evidence quality
- Enables semantic discovery, cross-paper synthesis, and contextual understanding across ArXiv's 15,000+ monthly publications
- Uses PostgreSQL with pgvector for scientific knowledge retrieval
- Demonstrates handling domain-specific language and structured academic conventions

Chapter 8, "Building a Complete Conversation Search and RAG System"

- Moves from searching public data to building a "second brain" for personal chat history with AI assistants
- Addresses unique characteristics of conversational data: contextual dependencies, conversational flow across exchanges, personal language patterns, and privacy requirements
- Handles the messiness of real conversational data where ideas develop across multiple message exchanges
- Creates a system for retrieving valuable insights, solutions, and learned knowledge from thousands of personal AI conversations

Chapter 9 proposes an experimental Vector Query Language (VQL) to standardize the fragmented vector database landscape:

Chapter 9, "Vector Query Language"

- Presents a prototype SQL-inspired query language for vector databases, designed to start community discussion and consensus building
- Addresses the needs of three communities: application developers seeking database abstractions, database maintainers needing cross-vendor tooling, and AI/machine learning researchers requiring efficient data manipulation
- Defines a data model (collections, tables, vectors, embeddings, metadata) and query syntax combining SQL familiarity with vector operations
- Covers similarity search, hybrid search, range search, batch operations, vector functions, and aggregations
- Proposes a path toward standardization across the fragmented vector database landscape

Who This Book Is For

This book's target audience comprises:

- Enterprise software developers building their first AI applications
- Data scientists implementing semantic search
- Machine learning engineers working on RAG systems
- Backend developers adding vector capabilities to existing systems

Additionally, it is geared toward people who learn by doing, so there are also practical applications focused on individual use.

Prerequisites include Python programming proficiency, a basic understanding of databases and SQL, and familiarity with running LLMs locally for the RAG chapters. Readers should also be comfortable with simple mathematical concepts.

The book is for those who want to learn how to build small- to medium-scale semantic search systems, implement RAG applications with local LLMs, optimize vector similarity search, choose the right indexes, and integrate vector search with relational databases.

The book focuses on Python implementations rather than other languages, covers CPU-based solutions rather than GPU optimization, and excludes cloud-managed vector database services in favor of self-hosted solutions. The scope of the applications and the theory is to cover enough to get started and do something useful with personal data as a first step. This is not a textbook on how to build web-scale production vector database applications. However, the lessons in the chapters can serve as a foundation and a springboard to take the next leap up in scope and complexity.

How to Use This Book

For the reader who is coming from a purely relational database background, reading the first two chapters is a must. Chapter 3 is more detailed on the innards of FAISS, but you can just use the code examples to dive right in and use it.

Chapters 4 through 8 use SQLite3 and Postgres to build semantic search and RAG applications using local data and local LLMs. Chapter 8 is especially topical, as it creates a semantic search over your LLM conversations—in this case, exported from Claude.

Finally, those interested in exploring similarities with SQL can dive into Chapter 9, which is a speculative design of a Vector Query Language to enable a unified, vendor-independent, client access metaphor for vector databases.

Software, Environment, and Resource Requirements

With the following resources, the example applications will run on standard developer laptops:

- Python version 3.11+
- Key dependencies: PostgreSQL, SQLite3, FAISS, Ollama
- OS: Linux, macOS, or WSL on Windows
- Minimum 16 GB RAM; preferably 24 GB

For a developer machine that can run PostgreSQL and an 8B LLM:

- Storage requirements: 512 GB to 1 TB

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
: Indicates new terms, URLs, email addresses, filenames, and file extensions.

`Constant width`
: Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords.

`Constant width bold`
: Shows commands or other text that should be typed literally by the user.

`Constant width italic`
: Shows text that should be replaced with user-supplied values or by values determined by context.

This element signifies a tip or suggestion.

This element signifies a general note.

This element indicates a warning or caution.

Using Code Examples

Supplemental material (code examples, exercises, etc.) is available for download at *https://github.com/nborwankar/VectorDatabaseBook*.

If you have a technical question or a problem using the code examples, please send email to *support@oreilly.com*.

This book is here to help you get your job done. In general, if example code is offered with this book, you may use it in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but generally do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*Vector Databases* by Nitin Borwankar (O'Reilly). Copyright 2026 Nitin Borwankar, 978-1-098-17759-1."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at *permissions@oreilly.com*.

O'Reilly Online Learning

For more than 40 years, *O'Reilly Media* has provided technology and business training, knowledge, and insight to help companies succeed.

Our unique network of experts and innovators share their knowledge and expertise through books, articles, and our online learning platform. O'Reilly's online learning platform gives you on-demand access to live training courses, in-depth learning paths, interactive coding environments, and a vast collection of text and video from O'Reilly and 200+ other publishers. For more information, visit *https://oreilly.com*.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
141 Stony Circle, Suite 195
Santa Rosa, CA 95401
800-889-8969 (in the United States or Canada)
707-827-7019 (international or local)
707-829-0104 (fax)
support@oreilly.com
https://oreilly.com/about/contact.html

We have a web page for this book, where we list errata and any additional information. You can access this page at *https://oreil.ly/vector-databases*.

For news and information about our books and courses, visit *https://oreilly.com*.

Find us on LinkedIn: *https://linkedin.com/company/oreilly-media*

Watch us on YouTube: *https://youtube.com/oreillymedia*

Acknowledgments

I want to thank all the technical reviewers, especially Sunil Sawant. Because of their efforts, the book is a much better product.

Sara Hunter, my development editor, deserves praise for her patience during the yo-yo delivery cadence of the chapters. I would also like to thank the rest of the O'Reilly editorial team—both the early release and final production teams—for putting up with the inexperience of a first-time author.

My wife, Garima, and daughter, Vanita, had to tolerate my minimal participation in family events during all of last year. I would like to thank them deeply for their support and understanding during the writing of "the book." It's finally done, guys!

Of course, none of this would be possible without the open source developer communities of PostgreSQL, SQLite, and the vector extensions, whose work enabled the example code.