Beginning MongoDB Atlas with .NET

Flexible and Scalable Document Data Storage for .NET Developers

Luce Carter

Apress®

Beginning MongoDB Atlas with .NET

Flexible and Scalable Document Data Storage for .NET Developers

Luce Carter

Apress®

Beginning MongoDB Atlas with .NET: Flexible and Scalable Document Data Storage for .NET Developers

Luce Carter
Manchester, UK

ISBN-13 (pbk): 978-1-4842-9549-6 ISBN-13 (electronic): 978-1-4842-9550-2
https://doi.org/10.1007/978-1-4842-9550-2

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Jonathan Gennick
Development Editor: Laura Berendson
Editorial Project Manager: Shaul Elson

Cover designed by eStudioCalamar

Cover image by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media LLC, 1 New York Plaza, Suite 4600, New York, NY 10004. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at http://www.apress.com/bulk-sales.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub (https://github.com/Apress). For more detailed information, please visit https://www.apress.com/gp/services/source-code.

If disposing of this product, please recycle the paper

For Jay,

You encouraged me to become an author when the chance arose, never doubting I could do it. You never stopped believing in me or being proud of my achievements. You were one of my biggest cheerleaders and I will never forget that. Sadly, we lost you before I finished this book, but I miss you every day.

Table of Contents

About the Author

Luce Carter is a Developer Advocate for MongoDB with a passion for sharing knowledge and making technology and code seem less intimidating. She is a Microsoft MVP and an international public speaker, enjoying speaking at conferences and other local meetups to share things she is passionate about. When not at a computer, she can be found playing squash with her local club, swimming, or trying to find interesting new places to walk. Her work to educate developers includes helping them to battle imposter syndrome – one line of code and story at a time.

Acknowledgments

Much of this book would not have happened without various people in my life for different reasons.

First, my mum, who has had to listen to me talk about a range of things and be a chatterbox my entire life. You are wonderful! You always checked in on how this book went, even though you don't truly understand what it is about.

Megan, a great friend and my copy editor. The quality of this book is better because of you. Thanks for always reviewing chapters to a great standard and being excited about my writing.

Kayy, for supporting me when I needed a distraction or to free my brain of something on my mind so I could focus on the book.

Rita, for allowing me to take time to write the book when I needed to and providing me with an amazing job and colleagues that make me want to write a book.

Alex and Rick, for tech-reviewing the book at different stages.

Kailie, for believing in me and making me feel like a rockstar for writing a book, convincing me to do it when I doubted myself.

Josh, who invited me to speak at his user group, giving a talk on this topic for the first time, that led to this book existing.

Northern Rebound, the best squash club I could have possibly joined. You all took an interest, gave me a mental break each week, and whose weekly social gave me a place to go early every week to get the book over the line.

Introduction

Welcome to *Beginning MongoDB Atlas with .NET*. A lot of time and love has gone into this book, so I hope you enjoy reading it as much as I enjoyed writing it.

I never imagined I would become a published author, but when the chance came about, I couldn't resist. I have loved helping the community for many years, often public speaking or creating content. This led me to the honor of being recognized as a Microsoft MVP in 2018. Around this time, I discovered that there is a role that is like being an MVP but paid: Developer Advocate.

In June 2021, I was lucky enough to become a Developer Advocate for the first time, working at MongoDB. To help me get up to speed on MongoDB as a developer data platform, I committed to giving a talk on MongoDB Atlas with .NET at the Liverpool .NET user group.

A short while after I delivered this talk, an acquisition editor at Apress reached out after seeing I had delivered the talk and asked if I thought there was enough content in this area for a book. This got me thinking and I started to picture what the structure of the book might look like. Before I knew it, I was putting together a proposal to become the author of this very book.

Luckily for me, Apress accepted my proposal and the wheels were in motion. The proposal acted as a guide to each chapter, so some of the work had been done before I even began. So when I was accepted and could get started, I was able to start quickly.

Like with everything in technology, MongoDB products can move fast. So when you read this, screenshots may be out of date or versions different, so please bear that in mind. There may even be features available now that weren't in existence when I wrote the book!

This book is structured to be both informational and educational. It starts with a walk through of types of databases and the history of MongoDB, including why it was created.

After that, it becomes a hands-on tutorial, showing you how to get started with the C# driver using a Web API project. It starts with deploying your first cluster, creating the project, and hooking it all together. You can follow it chapter by chapter, or even just drop in to a chapter for a reminder of how to do something later on.

PART I

Getting Started

CHAPTER 1

Choosing MongoDB

Technology is famous for moving at lightning speed. Things are forever changing and adapting, and new technologies and frameworks become popular. But there are some features that are often present in the majority of projects and applications. Examples include the *user interface (UI)*, network connectivity, and data storage.

Data is everywhere, whether it is the timetable for the public transport we take, our browsing history on our devices, our medical history, our finances, or something else entirely. This data needs to be stored somewhere so it can be accessed from one or many applications. This is where *databases* come in.

When it comes to choosing a database technology and vendor, there are many choices for your projects. This chapter aims to discuss and compare the different options as well as go further into what exactly *MongoDB* offers and where it fits in.

Relational vs. NoSQL Databases

Once you have decided that you need a database in your project, the first choice you face is what kind of database technology to go with.

Traditionally, the most common type of database has been a *relational*, or *tabular*, *database*. However, as technology has evolved, a competing set of options has surfaced: *NoSQL (not only SQL) databases*.

What Is a Relational Database?

A relational database, or *relational database management system (RDBMS)*, stores data in *tables*. A database can be made up of one or more tables. These tables store related data, and often, data is shared between tables to form relationships between them. This is where the name "relational database" comes from.

L. Carter, *Beginning MongoDB Atlas with .NET*, https://doi.org/10.1007/978-1-4842-9550-2_1

Inside these tables, *columns* are used to define the data and its shape, and *rows* are used to hold the records of this data. At least one column will be specified as holding unique values which identifies each row, and this column will become known as the *primary key*. This primary key is then used in other tables to form the relationships, whereby it becomes the *foreign key* in that table.

Let's look at an example of how data might be represented in a relational database to better understand this concept. Imagine we own our own business selling games. This business would have customers, products, orders, and suppliers. Customers would order products, and we would replenish products from our suppliers. We can see an example of how this might look across tables below in Table 1-1 for our customers, Table 1-2 for storing products, and Table 1-3 for orders.

Table 1-1. *Example of how data is represented in a relational database*

CustomerId	**Full_Name**	**Address**
1	Joe Bloggs	42 Data Lane, Information Land, SW1 1DB
2	Lisa Smith	18 Apple Way, Fruit Corner, N14 4RD
3	John Doe	27 Word Street, Binary, B12 1DN

Table 1-2. *Example of how the products table might look*

ProductId	**Name**	**Price**	**Category**
1	Monopoly	19.00	Board Game
2	Uno	9.99	Card Game
3	Carcassonne	6.99	Board Game

Table 1-3. *Example of how the orders table might look*

OrderId	**ProductId**	**Quantity**	**CustomerId**
1	1	1	2
2	3	2	1
3	2	1	1

These tables show how the ids for other tables are used to share information between them. For example, the orders table doesn't need to know product names or addresses, but instead, it can use the product and customer ids to look up that information when required, saving on storage space.

You may also hear relational databases referred to as "SQL" (sometimes pronounced "sequel"). This comes from *structured query language* (SQL) which is the language used to build, query, search, or filter one or more tables for the required data. SQL has both ANSI and ISO SQL standards and these have been evolving since 1986, the most recent being SQL:2016.

There are a few main providers when it comes to relational databases: *Oracle SQL*, *MySQL*, *Microsoft's SQL Server*, and *PostgreSQL*. Each of these has their own slight variation of SQL dialect, some with additional extensions. For example, SQL Server from Microsoft is the most different from the standard, using a dialect of SQL called *T-SQL* or Transact-SQL.

MySQL is the most popular open source database product. PostgreSQL is becoming more popular and is another open source database with advanced features. PostgreSQL is probably the best one for newcomers to relational databases today, as it has a free tier as well as a commercial distribution and has the most syntax in common with other variations should you need to use another type of RDBMS.

SQL Server is often used by Enterprises. There is a free community edition should you want to get started for free.

Oracle makes both Oracle SQL and MySQL. However, Oracle SQL is a paid-for product and considered more "commercial" as it is not open source and therefore not as easily changed.

As previously mentioned, there are slight syntax variations in the SQL implementations between the different databases. These differences are only small in basic queries, but once you move into more complex queries, the differences can add up and become quite significant.

The following code snippet shows a simple query in SQL which will fetch all the data for Lisa Smith from the customers table in Table 1-1. This syntax will be valid and generate the same results across all the main RDBMS.

```
Select *
FROM customers
WHERE
Full_name = 'Lisa Smith';
```