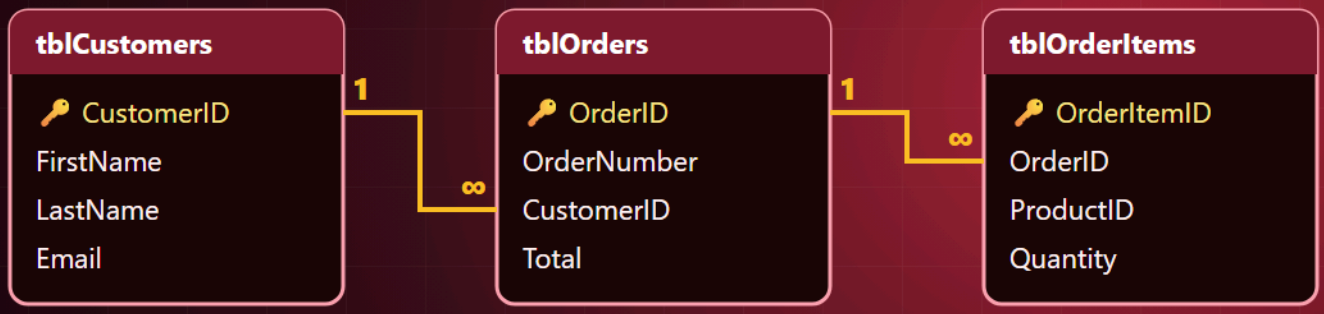
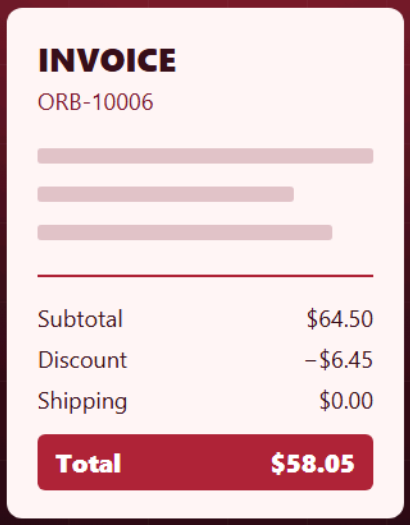



```
SELECT SellerName,
  Sum([Quantity]*[UnitPrice]) AS Sales
FROM qryOrderLines
WHERE Status <> "Cancelled"
GROUP BY SellerName;
```

```
Private Sub cmdRecalculate_Click()
  RecalcOrderTotals Me.OrderID
```



A COMPLETE, PRACTICAL GUIDE

Microsoft Access from Zero to Production

Design, build, and deploy real database applications for your business

Tables · Relationships · Queries · Access SQL · Forms
Reports · Macros · VBA · DAO · SQL Server · Deployment

DESIGN · QUERIES · FORMS · REPORTS · AUTOMATION

THENEXBOOKS

Microsoft Access from Zero to Production

Build Real Database Applications with Tables, Queries, Forms, Reports, and VBA
From your first table to a split, multi-user back-office system — built step by step

First Edition

THENEXBOOKS

Microsoft Access from Zero to Production
Build Real Database Applications with Tables, Queries, Forms, Reports, and VBA

First Edition, 2026

Copyright © 2026 thenexbooks (thenexbooks.gumroad.com). All rights reserved.

No part of this publication may be reproduced, distributed, or transmitted in any form or by any means without the prior written permission of the publisher, except for brief quotations in reviews and certain other non-commercial uses permitted by copyright law. Code examples, SQL, and VBA may be used freely in your own projects.

The information in this book is provided "as is" without warranty of any kind. While every precaution has been taken in its preparation, the publisher assumes no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein. Microsoft Access, Microsoft 365, and related products change over time; ribbon labels, dialogs, and feature availability can differ between versions and update channels. Always consult Microsoft's documentation for the version you use, and back up your data before running the examples that change it.

Microsoft, Microsoft 365, Access, Excel, Outlook, Word, SharePoint, SQL Server, Azure, Dataverse, Visual Basic, and Windows are trademarks of the Microsoft group of companies. Other product names are trademarks of their respective owners. Their use in this book is for identification only and does not imply endorsement. This book is independent and is not affiliated with, sponsored by, or endorsed by Microsoft Corporation.

Orbit Store, the online marketplace used as the running example in this book, is fictional, as are its sellers, products, customers, orders, and all sample data. Email addresses use reserved example domains.

Preface

Almost every organization has a spreadsheet that has outgrown itself. It started as a simple list, then gained more columns, more tabs, more copies emailed back and forth, and formulas that only one person understands. Prices disagree between versions, a customer appears three times with three spellings, and nobody can answer a simple question such as "which seller sold the most last month?" without an afternoon of copying and pasting.

Microsoft Access is built for exactly that moment. It combines a real relational database engine with the tools to build a complete application around it: tables that enforce rules, queries that answer questions in seconds, forms that make data entry fast and safe, reports that print professional documents, and automation with macros and VBA. A single person can build, in days, a system that would take a team weeks with other tools — and it runs on the Windows desktops many businesses already have.

This book teaches Access from the very beginning and takes you all the way to a production-quality application. We do not just tour the buttons. We build one real system step by step, introducing each feature at the moment it solves a problem we actually have, and explaining *why* it works the way it does, so you can design your own databases with confidence.

Who This Book Is For

- **Office workers and analysts** who live in Excel and need something more reliable for shared, growing data.
- **Small-business owners and managers** who want an in-house system for orders, inventory, customers, or operations without hiring a development team.
- **Students and beginners** who want to learn relational database design, SQL, and practical programming through a tool that shows results immediately.
- **Developers and IT staff** who inherit Access applications and need to understand, fix, secure, split, or scale them up to SQL Server.

What You Will Learn

- How to design a relational database properly: tables, keys, normalization, relationships, and referential integrity.
- How to build tables with the right data types, validation rules, defaults, and indexes — and how to import and link existing data.
- How to ask questions with queries: criteria, joins, parameters, totals, crosstabs, action queries, and Access SQL written by hand.
- How to build forms people enjoy using: main forms and subforms, combo boxes, search screens, conditional formatting, and navigation.

- How to produce reports and documents: grouped reports, invoices, packing slips, seller statements, charts, dashboards, PDF and Excel exports.
- How to automate with macros, data macros, and VBA: events, DAO recordsets, transactions, error handling, and Office automation.
- How to put it into production: splitting for multiple users, scaling up to SQL Server, security, maintenance, performance, ACCDE, the Access Runtime, and versioned deployment.

How This Book Is Organized

Part I — Foundations (Chapters 1–5) introduces Access, database design, tables, relationships, and importing data. **Part II — Queries** (Chapters 6–10) covers select, join, parameter, totals, crosstab, and action queries, and Access SQL. **Part III — Forms** (Chapters 11–15) builds the data-entry screens and teaches expressions and functions. **Part IV — Reports** (Chapters 16–19) covers reports, business documents, charts and dashboards, and distributing results. **Part V — Automation** (Chapters 20–24) covers macros, data macros, VBA, DAO, transactions, error handling, and Office automation. **Part VI — Production** (Chapters 25–28) covers multi-user deployment, SQL Server and other back ends, security and maintenance, and a complete deployment case study. The appendices provide a cheat sheet, templates and reference tables, a production checklist, and a glossary.

Each chapter ends with exercises and a **Key Takeaways** box for quick review.

The Running Example: Orbit Store

Throughout the book we build the back-office database for **Orbit Store**, a fictional online marketplace where three independent sellers — Northwind Ceramics, Pixel Forge Prints, and Cedar & Twine — list handmade products, and customers buy from them. Orbit Store needs to manage its catalog and stock, record customers and orders, apply coupons and shipping rules, print invoices and packing slips, pay each seller what they are owed after commission, and see at a glance how the business is doing. We start with a few well-designed tables and finish with a split, multi-user application with forms, reports, dashboards, automation, and a tested deployment process. The sample data is small enough to check by hand, so every query result in this book can be verified.

Versions and Conventions

Examples target **Access in Microsoft 365**, and almost everything also works in recent perpetual versions such as Access 2021 and Access 2024. Where a feature is newer or differs between versions, a Note says so. Access runs on **Windows** only. Screens, ribbon labels, and dialogs can differ slightly between versions and update channels; if a command is not exactly where the book says, look for it nearby or use the **Tell me** search box. The examples assume US English regional settings, so dates appear as `3/14/2026` and currency as `$24.00`.

> **TIP**
>
> A recommended practice or a technique that makes your work easier.

> **NOTE**
>
> Background information, version differences, or a useful aside.

> **WARNING**
>
> A common mistake or something that can cause data loss, errors, or security problems.

Ribbon commands are written in bold with arrows, such as **Create > Query Design**. SQL, VBA, and expressions appear in `monospace`; long lines are wrapped to fit the page. Object names follow a consistent convention — `tblOrders`, `qryOrderLines`, `frmOrders`, `rptInvoice` — explained in Chapter 1 and summarized in Appendix B.

Contents

Preface

PART I — FOUNDATIONS

1 What Access Is: Databases, Versions, and Your First Look

2 Designing a Database: Tables, Keys, and Normalization

3 Building Tables: Data Types, Field Properties, and Validation

4 Relationships and Referential Integrity

5 Getting Data In and Out: Import, Export, and Linking

PART II — QUERIES

6 Select Queries: Criteria, Sorting, and Calculated Fields

7 Joins, Parameters, and Multi-Table Queries

8 Totals, Grouping, and Crosstab Queries

9 Action Queries: Update, Append, Delete, and Make-Table

10 Access SQL by Hand: SQL View, Subqueries, and UNION

PART III — FORMS

11 Form Fundamentals: Controls, Layouts, and Data Binding

12 Main Forms and Subforms: Building Order Entry

13 Combo Boxes, List Boxes, and Lookups Done Right

14 Usable Forms: Validation, Conditional Formatting, Search, and Navigation

15 Expressions, Functions, and Domain Aggregates

PART IV — REPORTS

16 Report Fundamentals: Grouping, Sorting, and Totals

17 Business Documents: Invoices, Packing Slips, and Seller Statements

18 Charts, Dashboards, and Visual Reports

19 Printing, PDF, Excel, and Email: Distributing Results

PART V — AUTOMATION

20 Macros and Data Macros

21 VBA Fundamentals: Modules, Variables, Procedures, and Debugging

22 VBA with Forms and Reports: Events and the Object Model

23 DAO Recordsets, Transactions, and Error Handling

24 Automating Office and Files: Excel, Outlook, Word, and the File System

PART VI — PRODUCTION

25 Multi-User Access: Splitting, Linked Tables, and Locking

26 Scaling Up: Access with SQL Server, SharePoint, and Dataverse

27 Security, Maintenance, and Performance

28 Deployment and Case Study: ACCDE, Runtime, Versioning, and the Orbit Store Back Office

APPENDICES

A Access Cheat Sheet

B Templates and Reference

C Production Checklist

D Glossary

PART I

Foundations

What Access is and when to use it, how to design a relational database that stays correct as it grows, building tables with the right data types and validation, connecting them with relationships, and bringing existing data in from spreadsheets and files.

1 · What Access Is

2 · Designing a Database

3 · Building Tables

4 · Relationships and Referential Integrity

5 · Import, Export, and Linking

CHAPTER 1 · PART I — FOUNDATIONS

What Access Is: Databases, Versions, and Your First Look

Before you build anything, you need a clear picture of what a database is, why it beats a spreadsheet for business records, and where Microsoft Access fits. In this chapter you open the Orbit Store sample, learn your way around the Access window, meet the six kinds of objects you will build in this book, and adopt the naming convention that keeps a growing application tidy.

Why a database and not another spreadsheet?

Almost every small business starts its records in a spreadsheet. A spreadsheet is a wonderful tool: you type a value in a cell, you write a formula, you see the answer instantly. For a list of a few dozen products or a budget for next quarter, nothing is faster. The trouble starts when the spreadsheet stops being a calculation and becomes a *record-keeping system* that several people update every day.

Imagine Orbit Store in its first month. Orbit Store is a small online marketplace: independent sellers list products, customers buy them, and Orbit Store takes a commission before paying each seller. The founders track orders in one workbook. Every row is an order line, and every row repeats the customer's name, city, and email, the product name, the seller name, and the price. That works for ten orders. After a few hundred orders, the same problems appear in every growing spreadsheet:

- **Repetition.** Maya Chen's email address is typed on every order she places. When she changes it, someone has to find and fix every copy, and one copy is always missed.
- **Inconsistency.** One person types "Cedar & Twine", another types "Cedar and Twine", a third types "Cedar & Twine " with a trailing space. To a formula, those are three different sellers.
- **No rules.** Nothing stops a quantity of minus three, a price typed as text, or an order date in the wrong year. Data validation in a spreadsheet exists, but it is easy to paste around it.
- **Fragile sharing.** Emailed copies and simultaneous edits lead to the question "Which version is the real one?"

A **relational database** solves these problems by storing each kind of thing exactly once, in its own table, and connecting the tables with keys. Customers live in a customers table. Products live in a products table. An order does not repeat Maya's email address; it stores a small number, her customer ID, that points to her single row in the customers table. Change her email once and every order, invoice, and report sees the new value immediately.

Tables, rows, and columns

A database **table** looks a lot like a worksheet: it has columns and rows. The difference is discipline. In a table, every **column** (Access calls it a **field**) has a name and a data type that applies to every row. The `UnitPrice` field in the products table is always a money amount; you cannot type "about twenty dollars" into it. Every **row** (Access calls it a **record**) describes exactly one thing: one product, one customer, one order. And every table has a **primary key**, a field whose value is unique for each record, so that every record can be identified without ambiguity.

Spreadsheet idea	Database idea	What changes
Worksheet	Table	Holds one kind of thing only (products, or customers, never both)
Column	Field	Has a fixed data type, size, and optional validation rule
Row	Record	Identified by a primary key; no blank rows used as spacers
Cell formula	Query or calculated control	Calculations live outside the stored data
VLOOKUP	Relationship and join	Defined once, enforced by the database engine
Filter and pivot table	Query and report	Saved, named, and repeatable

The word *relational* refers to the relationships between tables. Chapter 2 explains how to decide which tables you need, and Chapter 4 shows how Access enforces the relationships so that an order can never point to a customer who does not exist. For now, hold on to one idea: **each fact is stored once, in the right table, and everything else refers to it.**

What Microsoft Access actually is

Many database products store data. Access is unusual because it combines two things in a single file. First, it includes a **database engine**, the Access Database Engine (often called ACE, and descended from the older Jet engine), which stores tables, enforces keys and rules, and runs queries. Second, it includes a complete **application builder**: a designer for data-entry forms, a report writer for invoices and summaries, a macro designer, and the Visual Basic for Applications (VBA) programming language. With Access, one person can design the tables, write the queries, draw the screens, print the documents, and automate the routine work, all without installing a separate server or writing a web application.

What Access is good at

- **Small-team desktop applications.** Order entry, inventory, membership lists, job tracking, asset registers, and similar back-office systems used by a handful of people on the same network.

- **Prototypes.** You can model tables, try out screens, and show a working application to colleagues in days rather than months. If the idea later grows into a large system, the table design carries over.
- **Reporting on other data.** Access can link to Excel workbooks, text files, other Access databases, SQL Server, and other sources through ODBC. Many people use Access purely as a flexible query and reporting tool on top of data that lives somewhere else.
- **Printed business documents.** The Access report designer handles grouped reports, invoices, labels, and statements well, including export to PDF.

Where Access stops

Knowing the limits up front saves you from building the wrong thing. These are the boundaries we design around in this book:

- **Windows only.** Access runs on Windows. There is no version of Access for macOS, iPad, or Android. Mac users can run it only inside a Windows environment, such as a virtual machine.
- **File based.** An Access database is a file on a disk or a network share. There is no database server process managing connections; each user's copy of Access reads and writes the shared file directly. That is simple to deploy but sensitive to unreliable networks, which is why Chapter 25 explains how to split and share a database safely.
- **2 GB per file.** A single Access database file cannot grow beyond 2 GB, and that limit includes the system objects Access stores in the file. A back office like Orbit Store's stays far below that for years, but large logs or stored images can reach it sooner.
- **Not a web application.** Access builds desktop software for the staff behind the scenes; customers use a separate website. The old "Access web apps" were retired, and this book does not use them.
- **Many users need care.** How many people can work at once depends on the network, the design, and how much each person writes. When the load grows, the usual path is to move the tables to SQL Server (Chapter 26).

> **NOTE**
> A useful way to place Access: Excel is for calculating, a server database such as SQL Server stores data for many applications and many users, and Access sits in between as a complete small application with its own data. An Access application can later switch to SQL Server tables with little change on screen.

Versions, editions, and file formats

This book uses **Access in Microsoft 365**, the subscription version that receives regular feature updates, as its baseline. Access is included in many Microsoft 365 plans for Windows PCs, but not all of them, so check the plan details if you are not sure you have it. Microsoft also sells perpetual, "buy

once" releases, such as **Access 2021** and **Access 2024**, including Long-Term Servicing Channel (LTSC) editions used by organizations. These releases receive security fixes but not new features after release.

Almost everything in this book works the same way in all of them: tables, queries, forms, reports, macros, and VBA have been stable for many releases. Where a feature is newer, such as the Large Number and Date/Time Extended data types or the modern chart control, we say "recent versions of Access" and point out what to check. To see exactly which version you have, open **File > Account** and select **About Access**.

There is also the free **Access Runtime**, which lets people use a finished Access application without owning the full product. They can enter data, run reports, and click your buttons, but they cannot open objects in Design view. Chapter 28 uses the Runtime when we deploy Orbit Store.

File extensions you will meet

Extension	What it is
`.accdb`	The standard Access database format, the default since Access 2007. Everything in this book uses it.
`.mdb`	The older format used before Access 2007. It still opens, but it lacks newer features; convert old files to `.accdb` when you can.
`.accde`	A compiled copy of an `.accdb` with the VBA source code removed and form and report design locked. Used for distributing a finished application (Chapter 28).
`.accdr`	An ordinary database renamed so that it opens in runtime mode, without design tools.
`.laccdb`	A temporary lock file Access creates next to an open database to coordinate users. It disappears when the last user closes the file.

> **TIP**
>
> Turn on file name extensions in Windows File Explorer (**View > Show > File name extensions** in Windows 11). You will work with `.accdb`, `.accde`, and `.laccdb` files side by side, and seeing the extensions prevents a lot of confusion.

Opening the Orbit Store sample database

The book comes with a sample database, `OrbitStore.accdb`, that already contains Orbit Store's tables and sample data. Copy it to a working folder of your own, for example `C:\OrbitStore\`, and keep the original untouched so you can always start again from a clean copy. The copy in your working folder is your **working copy**: it is the file you will build Orbit Store's application in as the book goes on. Then follow these steps:

1. Start Access. The start screen appears, with a list of templates (Blank database and several ready-made templates) and your recently opened files.

2. Select **Open** (or **Open Other Files**) in the left column, then **Browse**.
3. Find `C:\OrbitStore\OrbitStore.accdb` and select **Open**. You can also double-click the file in File Explorer.

The start screen, and the **File** menu you can reach at any time, is called the **Backstage view**. It is where you work with whole files: create, open, save as, compact and repair, and set options.

The Message Bar and "Enable Content"

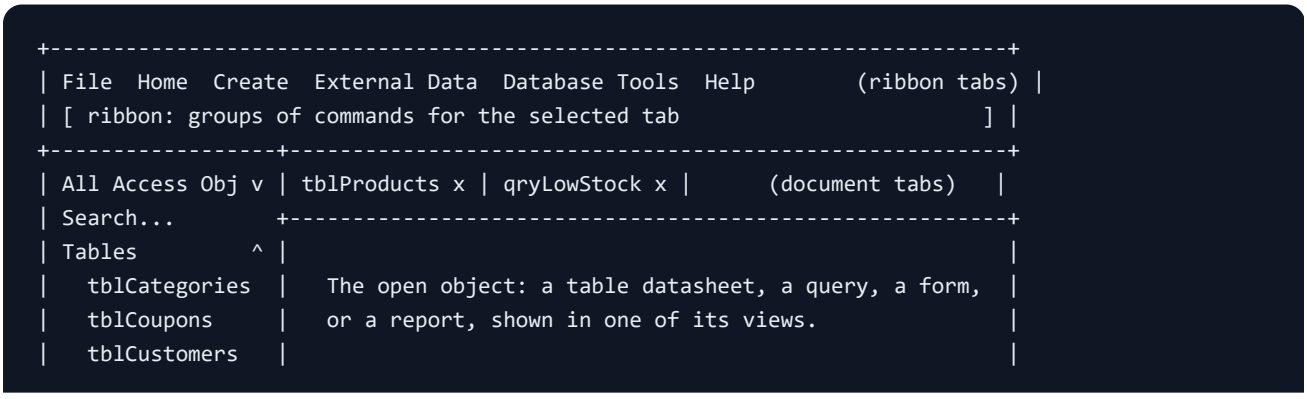
When the database opens, you may see a yellow **Message Bar** under the ribbon with a security warning that some active content has been disabled, and an **Enable Content** button. Access shows this when a database comes from a location it does not trust. In that state, tables and queries work normally, but VBA code and some macro actions are blocked. The Orbit Store sample has no code yet, so you can explore it either way, but from Part V onward the application depends on code, and you will want the content enabled.

Selecting **Enable Content** trusts that one file. A cleaner approach for your own projects is a **trusted location**: a folder whose databases Access always trusts. Open **File > Options > Trust Center > Trust Center Settings > Trusted Locations**, select **Add new location**, and add your working folder, such as `C:\OrbitStore\`. Databases in that folder open without the Message Bar.

> **WARNING**
>
> Only trust folders that contain files you created or received from people you trust. Never make your whole Downloads folder or an entire drive a trusted location. Also be aware that recent versions of Office block VBA macros in files that carry the "downloaded from the internet" mark, and the Message Bar then offers no Enable Content button. If that happens, close Access, right-click the file in File Explorer, select **Properties**, and look for an **Unblock** check box on the General tab; Chapter 27 explains this protection in more detail.

A tour of the Access window

With the sample open, take a minute to identify the main parts of the screen. The exact labels on the ribbon may differ slightly between versions of Access, but the layout has been the same for many releases.



```
|   tblOrderItems  |                                                        |
|   tblOrders      |                                                        |
|   ...            |                                                        |
| (Navigation Pane)| Record: 1 of 6   >  >|   Search       (navigation bar)    |
+------------------+--------------------------------------------------------+
| Status bar: Datasheet View                          [view buttons]          |
+-----------------------------------------------------------------------------+
```

The ribbon

The ribbon runs across the top of the window. Each tab groups related commands:

Tab	What you use it for
Home	Views, clipboard, sorting and filtering, finding records, record commands such as New, Save, and Delete, and text formatting
Create	New tables, queries, forms, reports, macros, and modules
External Data	Import, link, and export (Excel, text files, other databases, ODBC)
Database Tools	Compact and Repair, Relationships, the VBA editor, analysis tools, and the Database Splitter
Help	Help, training, and feedback

Extra **contextual tabs** appear only when they make sense. Open a table in Datasheet view and you see tabs for table fields and table properties; open a form in Design view and you see form design, arrange, and format tabs. When this book gives a command, it writes the path in bold, for example **Create > Query Design** or **Database Tools > Relationships**.

The Navigation Pane

The panel on the left is the **Navigation Pane**. It lists every object in the database. Press **F11** to show or hide it. At the top is a drop-down that controls how objects are grouped. The default, **All Access Objects** grouped by **Object Type**, puts tables, queries, forms, reports, macros, and modules under separate headings, the arrangement we use throughout the book. Below it is a search box: type `order` and the pane shows only objects whose names contain that text.

Double-clicking an object opens it in its default view: tables and queries open as datasheets, forms open in Form view, and reports open in Report view. Right-clicking an object gives you the other choices, including **Design View**.

Tabbed documents or overlapping windows

By default, databases created in recent versions of Access show each open object as a **tab** across the top of the work area, like tabs in a web browser. Older databases sometimes use **overlapping windows** instead, where each object floats in its own window. You can choose per database: open

File > Options > Current Database, find **Document Window Options**, select **Tabbed Documents** or **Overlapping Windows**, and then close and reopen the database. We use tabbed documents for development because they are easier to manage, and in Chapter 14 we look at pop-up and modal forms that float above the tabs when an application needs them.

Views and the status bar

Every object has more than one **view**: a table has **Datasheet view** for records and **Design view** for fields; forms and reports add Layout view, and reports add Print Preview. Switch with **Home > View** or the view buttons at the right end of the status bar.

The six kinds of Access objects

An Access application is made of named **objects**, all stored inside the one `.accdb` file. You will build every one of these kinds for Orbit Store.

Object	Purpose	Orbit Store example
Table	Stores data. The only object that actually holds records.	`tblProducts`, `tblOrders`
Query	Asks a question of the tables or changes data in bulk. Stored as SQL.	`qryLowStock`, `qrySalesBySeller`
Form	A screen for viewing, entering, and editing data.	`frmOrders`, `frmProducts`
Report	A formatted document for printing or PDF.	`rptInvoice`, `rptSellerStatement`
Macro	A list of actions built with a designer, without writing code.	`AutoExec`, `mcrNavigation`
Module	VBA code: functions and procedures.	`modPricing`, `modOrders`

The key idea is that **only tables hold data**. A query does not store a copy of the records it shows; it runs against the tables each time you open it. A form displays records from a table or query, and when you type into a form, the change goes straight into the underlying table. A report reads from a table or query when you open it. Because of that, there is only ever one copy of each fact, and every form and report always shows current data.

Looking at real data

In the Navigation Pane, double-click `tblProducts`. The table opens in Datasheet view, a grid that looks like a worksheet. Orbit Store has six products from three sellers:

`tblProducts (Datasheet view, selected columns)`

```
ProductID  SKU          ProductName             UnitPrice  UnitsInStock  ReorderLevel
---------  -----------  ----------------------  ---------  ------------  ------------
        1  ORB-CER-001  Speckled Stoneware Mug      $24.00            35            10
        2  ORB-CER-002  Glazed Serving Bowl         $48.00            12             5
```

```
3  ORB-PRT-001  Retro Rocket Poster    $19.99   80   20
4  ORB-PRT-002  Night Sky Art Print    $35.00    0   10
5  ORB-CDT-001  Hand-Poured Candle     $16.50   50   15
6  ORB-CDT-002  Linen Tote Bag         $22.00   40   10
```

Look at the **record navigation bar** at the bottom of the datasheet. It shows "Record: 1 of 6", buttons to move to the first, previous, next, and last records, a **New (blank) record** button, and a search box. The gray box at the left of each row is the **record selector**. When you start editing a record, a pencil icon appears in it, which means the record has unsaved changes.

One behavior surprises people coming from Word and Excel: **Access saves a record automatically as soon as you move to another record**, or when you close the table or form. There is no "Save" step for data and no "close without saving" escape hatch for records you have already left. While the pencil is showing, pressing **Esc** cancels your edits to the current record. If you experiment with a value, put it back; later chapters rely on the Night Sky Art Print being out of stock.

> **NOTE**
>
> Saving *data* and saving *design* are different. Records are saved automatically. Changes to the design of an object, such as adding a field or moving a control on a form, are saved only when you press **Ctrl+S** or confirm the "Do you want to save changes" prompt when closing the object.

Next, switch the table to Design view with **Home > View > Design View**. Instead of records, you now see the list of fields with their data types, and below them the property sheet for the selected field. This is where a table's rules live: the `UnitPrice` field is a Currency field with a validation rule of `>=0`, and the `Status` field accepts only the words Draft, Active, or Archived. Chapter 3 is entirely about this screen. Close the table without changing anything.

A first look at a query

You will spend all of Part II learning queries, but it helps to see one now. Select **Create > Query Design**. Close the dialog or pane that offers tables to add, then switch to SQL view with **Home > View > SQL View**. Replace any text in the window with the following statement:

A first query (SQL view)

```
SELECT SKU, ProductName, UnitPrice, UnitsInStock
FROM tblProducts
ORDER BY SKU;
```

Select **Query Design > Run** (the red exclamation mark; in some versions the tab is labeled simply **Design**). Access shows the result as a datasheet, sorted by SKU:

Result

```
SKU          ProductName              UnitPrice  UnitsInStock
-----------  -----------------------  ---------  ------------
ORB-CDT-001  Hand-Poured Candle          $16.50            50
ORB-CDT-002  Linen Tote Bag              $22.00            40
ORB-CER-001  Speckled Stoneware Mug      $24.00            35
ORB-CER-002  Glazed Serving Bowl         $48.00            12
ORB-PRT-001  Retro Rocket Poster         $19.99            80
ORB-PRT-002  Night Sky Art Print         $35.00             0
```

That statement is written in **SQL** (Structured Query Language), the language almost every relational database understands. Access lets you build queries visually in a design grid and writes the SQL for you, or you can type SQL directly as we did here. Close the query without saving it; Chapter 6 builds saved queries properly.

Meet Orbit Store

Orbit Store is a fictional online marketplace. Independent sellers list products, customers place orders that can include products from several sellers, and Orbit Store collects the money, keeps a commission, and pays each seller the rest once a month. The public website is out of scope for this book. What we build is Orbit Store's **back-office database application**: the system the staff use to manage the catalog and customers, enter and track orders, print invoices and packing slips, produce seller statements and payouts, watch dashboards, and automate the routine work, deployed as a split, multi-user application.

The sample database already contains the tables and a small, carefully chosen set of sample data. Every number in this book comes from that data, so you can check every result on your own machine. Here is what is in it:

Table	Records	What it holds
`tblSellers`	3	Northwind Ceramics, Pixel Forge Prints, and Cedar & Twine, with their commission percentages
`tblCategories`	3	Home & Kitchen, Art & Prints, Gifts & Accessories
`tblProducts`	6	The catalog, with SKU, price, stock, and reorder level
`tblCustomers`	8	Customers from Portland to Boston
`tblOrderStatuses`	6	Pending, Paid, Shipped, Delivered, Cancelled, Refunded
`tblCoupons`	2	WELCOME10 (10% off a first order) and FREESHIP
`tblOrders`	12	Orders ORB-10001 to ORB-10012, January to March 2026
`tblOrderItems`	20	The lines of each order: product, quantity, and price
`tblPayouts`	0	Empty for now; Chapter 23 fills it with monthly seller payouts

You can confirm the counts yourself. For example, this query counts the order lines:

Counting records

```
SELECT Count(*) AS N
FROM tblOrderItems;
```

Result

```
N
--
20
```

One order appears again and again in this book as our worked example: **ORB-10006**, placed on 2/14/2026 by Leo Okafor from Austin, Texas. It was Leo's first order, so he used the WELCOME10 coupon. He bought two Speckled Stoneware Mugs at $24.00 and one Hand-Poured Candle at $16.50, for a subtotal of $64.50. The coupon took off 10%, $6.45, which brought the amount to $58.05, above the $50.00 free-shipping threshold, so shipping was free and the total was $58.05. By the end of the book you will see that order in a form, on an invoice, in a seller statement, in a chart, and in VBA code that recalculates it.

The business rules

A database application is only as good as the rules it enforces. Orbit Store's rules are simple but realistic, and we will implement them several times in increasingly robust ways: as query expressions in Part II, as form logic in Part III, and in VBA in Part V.

- **Discount:** WELCOME10 gives 10% off the subtotal, only on a customer's first order, rounded to the nearest cent with halves rounded up.
- **Shipping:** a flat $5.99 when the subtotal after discount is below $50.00, free at $50.00 or more, and always free with the FREESHIP coupon.
- **Total:** subtotal minus discount plus shipping.
- **Commission:** each seller pays its commission percentage on its item sales; the payout is item sales minus commission. Cancelled orders are excluded from sales.
- **Low stock:** a product is low on stock when units in stock are at or below its reorder level.

The naming convention

An application with forty objects quickly becomes confusing if a table, a query, and a form are all called "Orders". This book uses a widely adopted prefix convention: every object name starts with a short lowercase tag that says what kind of object it is, followed by a descriptive name in PascalCase with no spaces.

Object or control	Prefix	Example

Table	`tbl`	`tblOrders`
Query (select)	`qry`	`qryOrderLines`
Crosstab query	`qxtb`	`qxtbSalesByCategoryMonth`
Action queries	`qryUpd` , `qryApp` , `qryDel` , `qryMak`	`qryUpdArchiveProduct`
Form / subform	`frm` / `sfrm`	`frmOrders` , `sfrmOrderItems`
Report / subreport	`rpt` / `srpt`	`rptInvoice` , `srptInvoiceLines`
Macro	`mcr`	`mcrNavigation`
Module / class module	`mod` / `cls`	`modPricing`

Controls on forms and reports get prefixes too: `txt` for text boxes, `cbo` for combo boxes, `lst` for list boxes, `cmd` for command buttons, `chk` for check boxes, `lbl` for labels, `opt` or `fra` for option groups, `sub` for subform controls, and `tab` for tab controls. So the order form, `frmOrders` , will have a combo box named `cboCustomerID` and a button named `cmdPrintInvoice` .

Why bother? When you read `[Forms]![frmOrders]![txtTotal]` , you know at once that it is a text box on a form. And names without spaces never need square brackets in SQL and VBA, and survive a move to SQL Server. Fields follow the same no-spaces rule, which is why the sample uses `UnitPrice` rather than "Unit Price"; the friendly label a user sees is set separately with the **Caption** property, as Chapter 3 shows.

> **TIP**
>
> The one exception to the prefix rule is `AutoExec` , a macro name that Access recognizes specially and runs when the database opens. Chapter 20 creates it.

How this book is organized

The book follows the order in which you would build a real application, and each chapter adds working pieces to Orbit Store:

- **Part I, Foundations (Chapters 1 to 5):** table design, data types, relationships, import and export.
- **Part II, Queries (Chapters 6 to 10):** select, join, totals, crosstab, and action queries, and SQL.
- **Part III, Forms (Chapters 11 to 15):** data entry, order entry with a subform, combo boxes, and expressions.
- **Part IV, Reports (Chapters 16 to 19):** grouped reports, invoices, statements, charts, and PDF.
- **Part V, Automation (Chapters 20 to 24):** macros, VBA, events, DAO, and Office automation.
- **Part VI, Production (Chapters 25 to 28):** splitting, SQL Server, security, and deployment.