Printed by: rituadhikari20@yahoo.com. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

CHAPTER 3

Why Good Design Matters

Abstract

This chapter reinforces the idea that there are significant practical reasons to care about database design. It discusses the problems that occur frequently with poorly designed databases, as a prelude to the following chapters that teach the principles of good database design.

Keywords

database design
poor database design
insertion anomalies
deletion anomalies
entity identifiers

Many of today's businesses rely on their database systems for accurate, up-to-date information. Without those repositories of mission-critical data, most businesses are unable to perform their normal daily transactions, much less create summary reports that help management make corporate decisions. To be useful, the data in a database must be accurate, complete, and organized in such a way that data can be retrieved when needed and in the format desired.

Well-written database application programs—whether they execute locally, run over a local area network, or feed information to a Web site—are fundamental to timely and accurate data retrieval. However, without a good underlying database design, even the best program cannot avoid problems with inaccurate and inconsistent data.

Effects of Poor Database Design

To make it a bit clearer why the design of a database matters so much, let's take a look at a business that has a very bad design, and the problems that the poor design brings. The business is named *Antique Opticals and DVDs* (called *Antique Opticals* for short by its regular customers).

Note: Remember the definition of a database that was presented in Chapter 1. As you will see, the data storage used by Antique Opticals and DVDs isn't precisely a database.

Back in the early 1980s, when most people were just discovering videotapes, Mark Watkins and Emily Stone stumbled across a fledgling technology known as the laser disc. There were several competing formats, but by 1986 the industry had standardized on a 12-inch silver platter on which either 30 or 60 minutes of video and audio could be recorded on each side. Although the market was still small, it was at that time that Watkins and Stone opened the first incarnation of their business, originally named *Lasers Only*, to sell and rent laser discs. Some of its income came from sales and rentals directly from its single retail store. However, the largest part of its revenue came from mail-order sales.

The appearance of DVDs in 1995 put a quick end to the release of new laser discs. The last new title was released in 2000, but the number of new titles had decreased significantly even before that time. Watkins and Stone saw their business dwindling rapidly in the late 1990s.

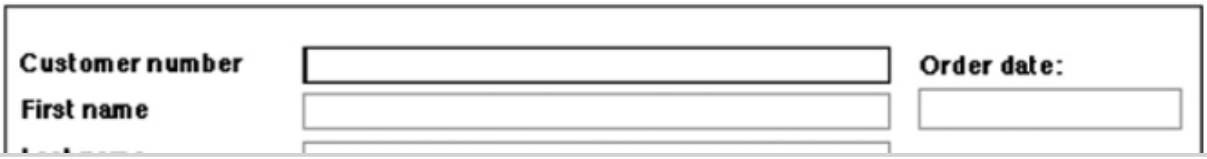
The owners of *Lasers Only* realized that they needed to change their merchandise focus, if they were to continue in business. Laser discs were now the "antiques" of the optical media world, with DVDs—and most recently Blu-ray high-definition discs—the currently technology.

Antique Opticals and DVDs now sells and rents DVDs and Blu-ray discs from the retail store. It also sells used laser discs it purchases, used, from those who wish to sell them.

The company has a primitive Web site for mail order sales. The current catalog is updated weekly and uploaded to the site, where users place orders. However, the catalog is not integrated with any live data storage. An employee must take the order from the Web site and then shop for the customer. As a result, customers occasionally do not receive their entire order, when ordered items have been sold at the retail store. This is particularly true in the case of used laser discs, where the quantity in stock is rarely more than one per title.

In 1990, when the store began its mail order business, Watkins create a "database" to handle the orders and sales. Customers were (and still are) enticed to order titles before the official release date by offering a 15–20 percent discount on preorders. (All titles are always discounted 10 percent from the suggested retail price.) The mail order database (which has evolved into today's Web order database) therefore needed to include a way to handle backorders so that preordered items could be shipped as soon as they came into the store.

At the time we visit *Antique Opticals*, it is still using the software Watkins created. The primary data entry interface is a form like that in Figure 3.1.



Printed by: rituadhikari20@yahoo.com. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

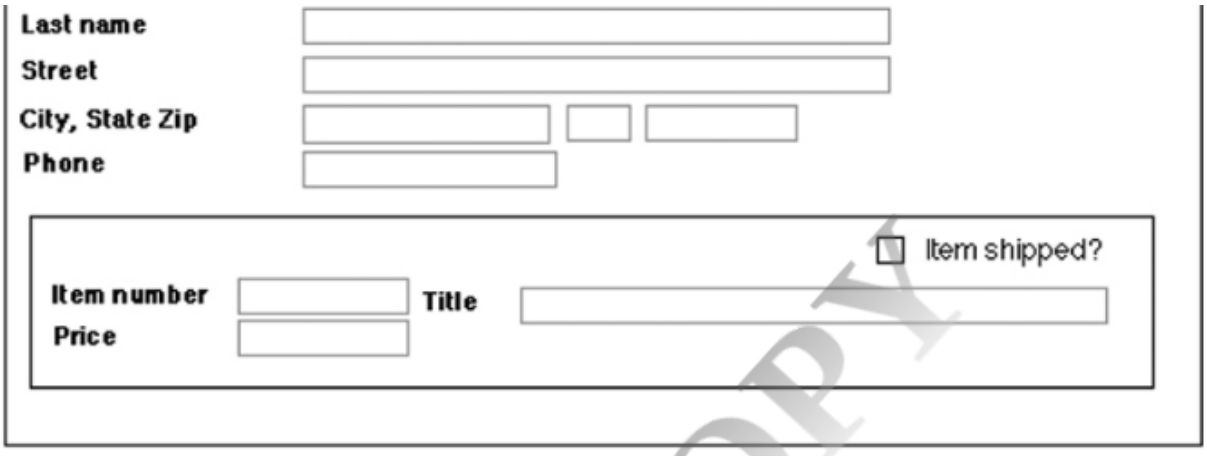
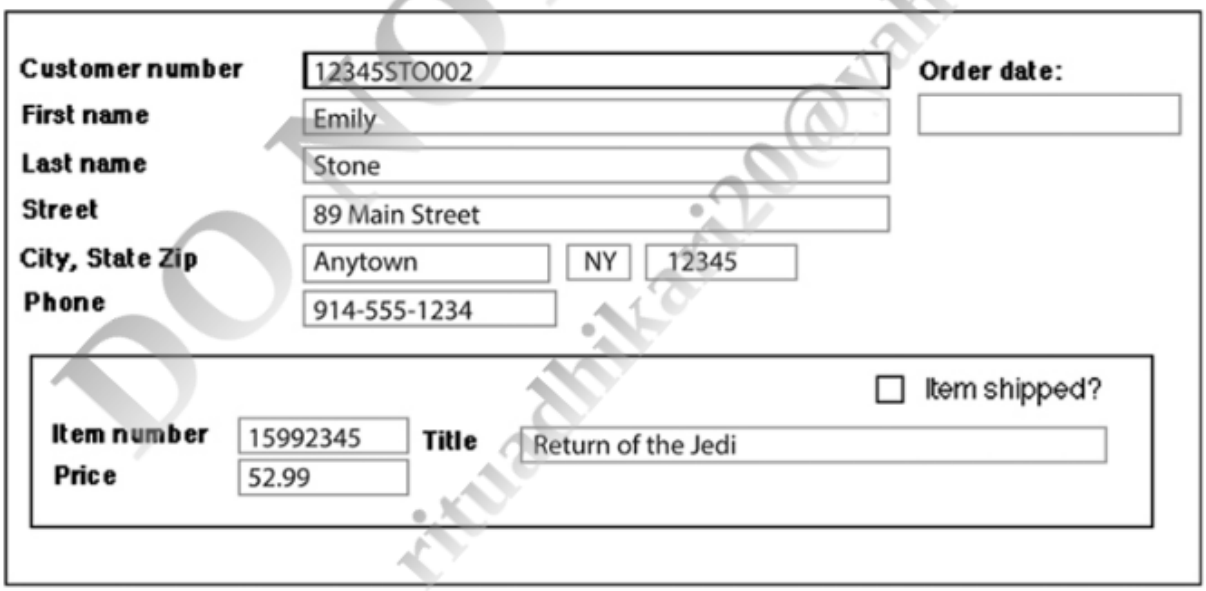
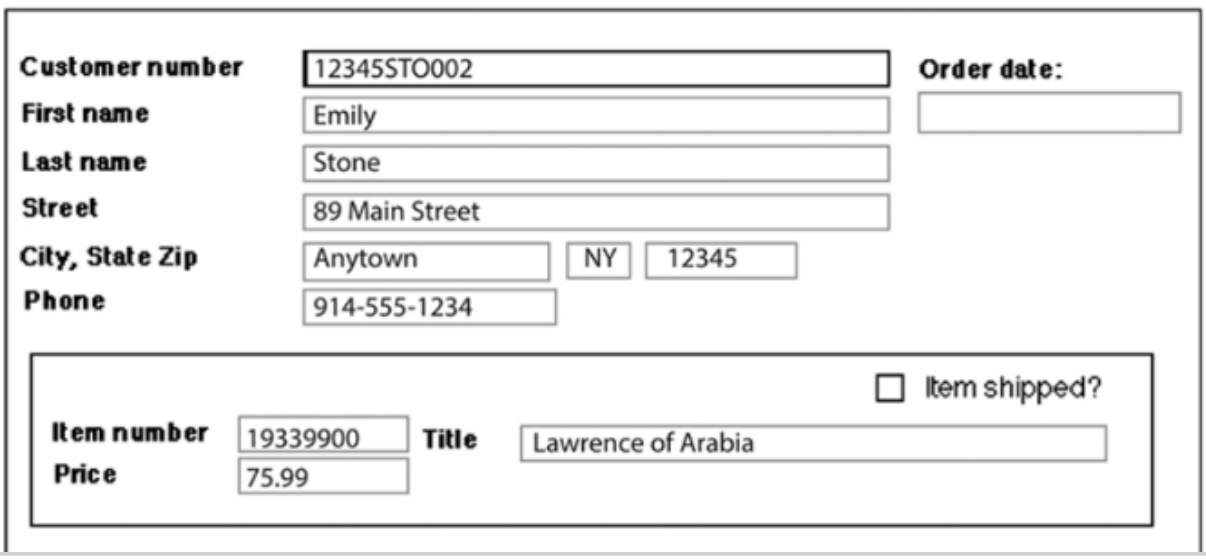


FIGURE 3.1 **The data entry form used by *Antique Opticals* and DVDs for their mail order business.**

Customer numbers are created by combining the customer's zip code, the first three letters of his or her last name, and a three-digit sequence number. For example, if Stone lives in zip code 12345 and she is the second customer in that zip code with a last name beginning with STO, then her customer number is 12345STO002. The sequence numbers ensure that no two customer numbers will be the same.

Let's assume that Ms. Stone places an order for three items. One copy of all the data collected on the data entry form for each item ordered will be stored in the database. Therefore, an *Antique Opticals* employee fills out the form three times with the data found in Figure 3.2. The most important thing to notice is that the customer number, customer name, customer address, and customer phone number are duplicated for each item on the order. Those data will be stored three times.





DO NOT COPY
rituadhikari20@yahoo.com

Printed by: rituadhikari20@yahoo.com. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

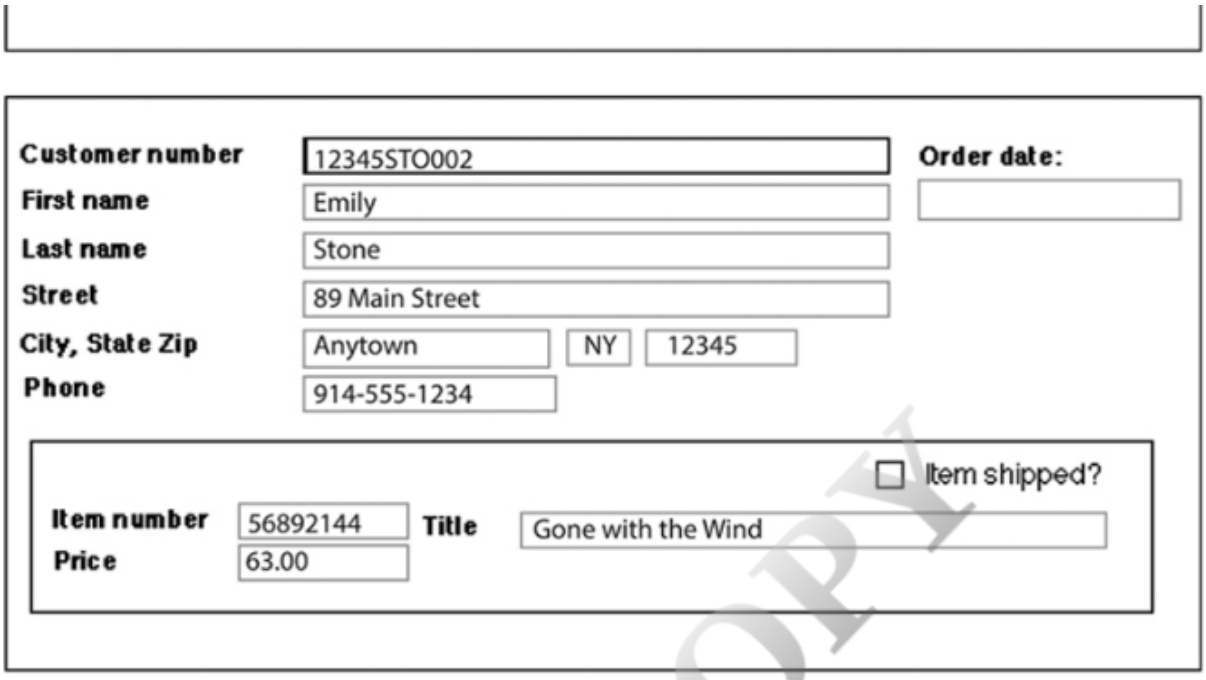


FIGURE 3.2 **The data entered by an *Antique Opticals* employee for a three-item order.**

When a new title comes into the store, an employee searches the database to find all people who have preordered that title. The employee prints a packing slip from the stored data and then places an X in the "item shipped" check box.

At first glance, *Antique Opticals*' software seems pretty simple and straightforward. Should work just fine, right? (This is assuming we ignore the issue of integration with the Web site.) Well, it worked well for a while, but as the business expanded, problems began to arise.

Unnecessary Duplicated Data and Data Consistency

The *Antique Opticals* database has a considerable amount of duplicated data that probably don't need to be duplicated:

- A customer's name, address, and phone number are duplicated for every item the customer orders.
- A merchandise item's title is duplicated every time the item is ordered.

What is the problem with this duplication? When you have duplicated data in this way, the data should be the same throughout the database. In other words, every order for a given customer should have the same name, address, and phone number, typed exactly the same way. Every order for a single title should have the exact same title, typed exactly the same way. We want the duplicated data to be consistent throughout the database.

As the database grows larger, this type of consistency is very hard to maintain. Most business-oriented database software is *case sensitive*, in that it considers upper- and lowercase letters to be different characters. In addition, no one is a perfect typist. A difference in capitalization, or even a single mistyped letter, will cause database software to consider two values to be distinct.

When an *Antique Opticals* employee performs a search to find all people who have ordered a specific title, the database software will retrieve only those orders that match the title entered by the employee exactly. For example, assume that a movie named *Summer Days* is scheduled to be released soon. In some orders, the title is stored correctly as "Summer Days." However, in others it is stored as "summer days" or even "sumer days." When an employee searches for all the people to whom the movie should be shipped, the orders for "summer days" and "sumer days" will not be retrieved. Those customers will not receive their orders, resulting in disgruntled customers and, probably, lost business.

The current *Antique Opticals* software has no way to ensure that duplicated data are entered consistently. There are two solutions. The first is to eliminate as much of the duplicated data as possible. (As you will see, it is neither possible nor desirable to eliminate all of it.) The second is to provide some mechanism for verifying that, when data must be duplicated, they are entered correctly. A well-designed database will do both.

Note: Unnecessary duplicated data also take up extra disk space, but given that disk space is relatively inexpensive today, that isn't a major reason for getting rid of redundant data.

Data Insertion Problems

When operations first began, the *Lasers Only* staff generated the catalog of forthcoming titles by hand. By 1995, however, Stone realized that this was a very cumbersome process and thought it would be much better if the catalog could be generated from the database.

Why not get a list of forthcoming titles from the database and have an application program generate the entire catalog? As Stone discovered, it could not be done from the existing database. There were two major reasons.

First, the database did not contain all of the information needed for the catalog, in particular a synopsis of all the content of the disc. This problem could be remedied by adding that information to the database. However, doing so would only exacerbate the problem with unnecessary duplicated data if the company were to include the summary with every order. If the summary were to be included only once, how would the company know which order contained the summary?

Second, and by far more important, there is no way to enter data about a title unless someone has ordered it. This presents a large Catch-22. The company could not insert data about a title unless it had been ordered at least once, but customers would not know that it was available to be ordered without seeing it in the catalog. But the catalog could not contain data about the new title until someone was able to get the data into the database, and that could not happen until the title has been ordered.

Note: This problem is more formally known as an "insertion anomaly," and you will learn about it more formally throughout this book.

Printed by: rituadhikari20@yahoo.com. Printing is for personal, private use only. No part of this book may be reproduced or transmitted without publisher's prior permission. Violators will be prosecuted.

Note: This problem is more formally known as an "insertion anomaly," and you will learn about it more formally throughout this book.

Antique Opticals solved the problem by creating a second database for forthcoming titles from which the catalog could be generated. Unfortunately, the second database produced problems of its own, in particular because it introduced yet another source of duplicated data. The catalog database and the orders database did not communicate to verify that duplicated data are consistent, creating another potential source of errors in the orders database.

Data Deletion Problems

Antique Opticals also has problems when it comes to deleting data. Assume, for example, that a customer orders only one item. After the order has been processed, the item is discontinued by the manufacturer. *Antique Opticals* therefore wants to delete all references to the item from its database because the item is no longer available.

When the orders containing the item are deleted, information about any customer who has ordered only that item is also deleted. No other orders remain in the database for that customer. *Antique Opticals* will be unable to e-mail that customer any more catalogs and must hope that the customer visits the Web site without being contacted by the company. A very real potential exists that *Antique Opticals* has lost that individual as a customer.

Note: This problem is more formally known as a "deletion anomaly." It, too, will be discussed in greater depth throughout this book.

Meaningful Identifiers

The *Antique Opticals* orders database has another major problem: those customer numbers. It is very tempting to code meaning into identifiers, and it usually works well—until the values on which the identifiers are based change.

Consider what happens when an *Antique Opticals* customer moves into a different zip code. The person's customer number must change. At that point, there will be orders for the same customer with two different customer numbers in the same database.

If a customer who has moved since first ordering from the store calls and asks for a list of all items he or she has on order, the first thing the employee who answers the telephone does is ask the customer for his or her customer number. The customer, of course, provides the current value, which means that anything ordered under the old customer number will be missed during a search. The customer may assume that titles ordered under the old customer number are not on order. As a result, the customer may place another order, causing two copies of the same item to be shipped. *Antique Opticals* is then faced with another disgruntled customer who has to make the effort to send back the duplicate, and get the second charge removed from his or her credit card.

The Bottom Line

The bottom line is that what might initially seem like a simple, good design for data storage may present problems that will affect the consistency and accuracy of the data in the long run. Don't forget that the major goal with a database is to retrieve data accurately when it is needed. A bad design can make achieving that goal very difficult, especially because you may not always know that a problem exists. It is therefore worth the time and effort needed to design your database well from the start.

DO NOT COPY
rituadhikari2006@yahoo.com