

# War Stories: Autonumber, Archiving, and Maternity Leaves

Tobi K. Hoffman



Come with Tobi Hoffman to track down an elusive bug. How soon can you spot the problem? And how would you fix it? Tobi shows the kinds of problems that can occur when you use Autonumbers.

**F**OUR years ago, I had a very small company as a client. My assignment was to convert an Access database for managing sales orders from Access 2.0 to Access 95. After an intensive month or so, I'd made the conversion, smoothed out the rough edges, and added some features.

Archiving data was one of the early additions I'd made to the database. The records had accumulated in the system so that there was a noticeable delay in locating orders. In this company, the records of primary interest were those that were in process, not the historical data in completed orders. Yet the details of those old orders were important, because they could be analyzed to provide some history of sales of particular items. And occasionally, it was necessary to re-open an old order so that corrections could be made to it. In my archiving routine, therefore, the order header records were moved to an archive table, but the detail records were kept in the main database. I also wrote a restore routine that would allow users to recover records from the archive.

I then moved on, and an employee, Heidi, took over entering orders, preparing reports, and archiving completed orders.

Over the next few years, Heidi called me occasionally to add some more functionality, or to fix or add a report, never totaling more than a few hours a year. I'd learned about the tendency of Access databases to bloat, especially during data archiving, so I created a new Access database with a set of routines to be run periodically to compact the databases that made up the main application.

## The problem

This past autumn, the company decided to make the jump not only to new computers but to Windows 2000 and Office 2000 on all of the machines. So I spent an evening installing a new version of the application, and another Saturday working with Heidi, who had a baby due, so that she could do some work at home.

The company was generous with Heidi's maternity leave, and she began doing part-time data entry at home. Suddenly, something odd began to happen. I got an emergency e-mail from Heidi: "The program's going whacko! I enter a new order, and it pops up with old shipping dates, and the quantities don't match!" Unmatched quantities activate a red flag in this program, and it has to be fixed before anything else. So Heidi would delete that order and try to go on. Eventually, the database would stop misbehaving, and she would be able to get back to entering more orders. Sooner or later, the problem would repeat, though.

I spent one evening with Heidi, watching her enter data—and no problems occurred at all. There was one minor change that I could make, just a little cosmetic improvement to increase the text box size on a few fields that needed it, but nothing that changed any real functionality.

Things seemed to work for another couple of weeks (mostly), but then came another session of frustration for Heidi. Her boss blamed the new computers; I suspected the Windows/Office 2000 upgrade (had it really been necessary?), but in the absence of hard data, we scheduled another evening of me watching Heidi.

This time, it only took one new entry for the program to go whacko. Heidi entered a brand-new order, and suddenly up popped a couple of completed order details for this supposedly new order. Since these order details stated that 200 units had already been shipped, a mismatch occurred. The screen turned bright red, and everything came to a screeching halt.

## Debugging

I switched from the compiled MDE file to the MDB version of the program so that I could trace the code. Unfortunately, that got me nowhere—the code looked fine. My next stop was to look at the data. I was able to retrieve the key value for the new order that Heidi had created, and I used that to explore the order detail table. Surprisingly, I found some records with that value in their foreign key fields, linking back to this new order. Since Heidi hadn't entered any detail records, where had these order detail records come from?

I followed a number of leads without luck, and, at

last, I looked at the archive table. The archive table is where completed sales are moved from the main table. In that table, I found records with the same value as the record that Heidi had just added. Suddenly, it all became clear.

In the normal course of events, sales data is entered, paperwork is generated, but the sale isn't complete until the parts have been shipped. Once the parts had been shipped, Heidi recorded the shipping date and billed for the order. Finally, the order is paid, everything is complete, and the archive routine moves the data from the orders table to the archive table. However, lately Heidi had been playing catch-up. By the time she entered the data, the order had gone through; so Heidi was entering the data, generating the reports, and marking the order as paid in one session. Then she'd archive the data.

With her work done, Heidi would exit the program and run my program to compact the database. The compacting program reset any Autonumber fields in the database to one more than the highest key in the table. If there were 2,000 records in a table, the last record in the table would have a key value of 2000. If the last five records in the table were deleted, the highest number in the table would be 1995. So, after compacting, the next Autonumber that would be generated would be one more than 1995, or 1996.

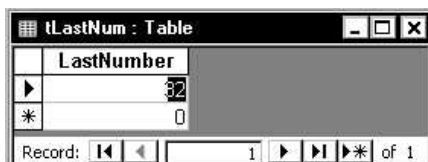
In our case, the archiving routine was deleting records from the main table and adding them to the archive table but leaving the detail records behind. Those five deleted records would be order header records, but when the Autonumber field got reset to 1996, the detail records for the original order 1996 would still be in the databases. Once Heidi entered her next record, it would be assigned a key value of 1996 and automatically link to the detail records for the original 1996 order's details. The problem would go away because Heidi would delete the bogus records from 1996 to 2000, and then things would begin to work again because she'd now gone beyond the highest numbered order detail record.

There were a number of things that I could have done to prevent this problem. For instance, if we hadn't wanted to keep the detail records, when I'd added relationships between the tables I could have enabled cascade delete.

### And its solution

With the problem now located, I needed to resolve it. At this point, that could easily have turned into a week's full-time work. I would have had to implement a solution, make sure it worked, provide the ability to restore data,

Figure 1. Table for last number used.



and make sure that all of the reports would still perform as before. I opted instead to replace the Autonumber key field with a key that I generated. This required a new table to hold the last number used (see Figure 1), a query to increment that number (see Figure 2), and a bit of code to put the new key together. The table consists of the single field intLastNum (a long integer) because, since there's only one record in the table, no key field is needed.

I changed the key field in my order header table from Autonumber to Text and made it 10 characters in length, so that it would hold alphanumeric information. This would ensure that I wouldn't overwrite any existing key fields. Also, with a bit more complex code, I could, at some point, add the flexibility of having multiple data entry people, each with their own character prefix to the record's key number.

I was glad that I'd long ago put in an Add New Record button, rather than letting Heidi use the navigation "add record" button. In the buttons called in the cmdAdd\_Click event, I added a call to my GetNextNum function to generate the key:

```
Private Sub cmdAdd_Click()
On Error GoTo Err_cmdAdd_Click

DoCmd.GoToRecord , , acNewRec

Me!OrderID = GetNextNum
cboBuyer.Requery
subDeliverySchedule.Requery
Me.Refresh
cboCustomer.SetFocus

Exit_cmdAdd_Click:
Exit Sub

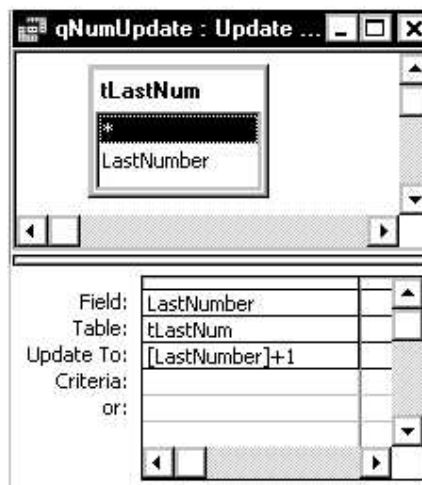
Err_cmdAdd_Click:
MsgBox Err.Description
Resume Exit_cmdAdd_Click
End Sub

Public Function GetNextNum() As String

DoCmd.OpenQuery "qNumUpdate"
GetNextNum = "A" & DLookup("[LastNumber]", _
"[tLastNum]")

End Function
```

Figure 2. Query to increment last number.



The key field is filled with an arbitrary letter and a number that comes from the last number table, and the function concatenates the letter and the number. I considered whether I should format the number to pad the field with zeros, but decided not to. The user never sees this number, and there was no need to ever order the entries using this field.

The old orders remain with the original key fields, but converted to text; there's no conflict because none of them begin with a letter. The existing archive and restore routines don't require any more than for the data being moved to have the same data type as the fields in the tables that they're moved to. It was all done in an evening's work!

So Autonumber is gone, at least from the main table. The archive works; the restore routine works; the compact routine works; and there will be no by-product of the compacting routine to create duplicate key field values. All in all, the entire program is working well for Heidi. These changes don't show up at all to anyone in the office, and hopefully this should keep them running smoothly for quite a while—even when her next baby comes! ▲

Tobi K. Hoffman has been an Access developer for six years, an avid programmer for far longer, and currently works for Worldwide Telecommunications Systems, a division of General Dynamics, in Needham, MA. [tobihoffman@mediaone.net](mailto:tobihoffman@mediaone.net).

---

## Access 2002 for Developers...

*Continued from page 8*

thanks to extended properties.

9. Code that uses the `PrtDevMode` and `PrtMip` properties can, in some cases, be easily replaced by using the new Printer objects. Once again, less code makes for easier maintenance.
10. If you've been working around any painful bugs in Access 2000, take a look at Access 2002 to see whether they're fixed. The "What's New" lists you'll find here and elsewhere can only cover the highlights. Microsoft routinely fixes thousands of minor bugs with every version, so it's possible that you'll be able to get rid of workarounds.

So, should you be moving to Access 2002? This depends somewhat on which version of Access you're currently using. If you never moved beyond Access 97, it's definitely time to upgrade. That's in part because of the accumulated new features across two versions of Access, but mainly because Microsoft will be dropping Access 97 support as part of its normal policy for phasing out old versions of software.

The bottom line for Access 2002 from the Access 2000 developer's point of view is somewhat more complex. There's no single compelling feature to force you to upgrade. Just about everything that you can do in Access 2002 you can also do in Access 2000. But, for the first time, there's no drawback to upgrading either, because you're not forced to go through a complete file format migration for every database that you're developing. That's enough to tip the balance for me to installing and using the new version. Being able to make a gradual transition means that you can introduce new features to applications at

your clients' pace while learning them at your own pace—a good deal, in my book. ▲

Mike Gunderloy is an independent consultant and author who lives in eastern Washington state. He's the co-author (with Joe Jorden) of *Mastering SQL Server 2000*, and the co-author (with Tim Sneath) of the forthcoming *SQL Server Developer's Guide to OLAP with Analysis Services*, both from Sybex. [MikeG1@larkfarm.com](mailto:MikeG1@larkfarm.com).