June 2000

Uplink

TAXES

A taxing issue

By Richard Dalton
Newsday

Nearly 15 years after a *Newsday* series revealed that minorities on Long Island pay disproportionately higher property taxes, the area's tax assessment system persists. So the newspaper decided to take another look at the issue earlier this year.

What we found was that despite hundreds of millions of dollars in payments to settle claims by overtaxed residents over the last decade and despite lawsuits by organizations including the New York Civil Liberties Union, Nassau County continued to defend a decades-old tax system that defied standard practices.

Shortly after our story appeared, the county legislature voted to reassess based on market value. While some legislators who voted for the change cited *Newsday's* findings, observers generally said the main reason for the about-face was the county's ever-worsening fiscal position, caused in large part by the multimillion-dollar cost of settling lawsuits stemming from incorrect assessments.

1938 prices

Across the country, most government entities base property taxes on the market value of a home. But Nassau County has been calculating property taxes based on what it would have cost to construct the home in 1938, when the system was implemented. That effectively means that minority and low-income areas pay a higher rate of property taxes.

Here's why: A home in a rich, white neighborhood may have had the same construction costs in 1938 as a home in a poor, minority neighborhood. So the two homeowners would pay the same taxes. Meanwhile, the home in the wealthy neighborhood may have soared in value over the decades, so the taxes would be a significantly smaller percentage of home value compared to the home in the poor neighborhood, where property values often have risen far less.

You might also be sitting on a great tax-assessment story in your area because government officials may not be reassessing homes often enough, resulting in an unfair system. Or, if you live in California, you have a powerful story thanks to Proposition 13. Passed in 1978, the law requires that homes be assessed at market value when they are built or sold, and limits in-

Continued on page six

TECH TIP

Starting with modules

By Tim Henderson
The Miami Herald

If you're new to programming, Access Modules are going to take you to the next level of database work.

If you're an old hand at coding in FoxPro, PERL or SAS, you probably know that modules are your ticket to getting under the hood of Access into the functionality you're used to getting from those programs.

Modules mean you're no longer limited to the query functions Access provides for queries, and you're no longer forced to depend on the Import Wizard to dive into text files.

In this article we'll see a module that takes a nightmare text file with no delimiters and data spread over multiple lines and suck it into a database for analysis. You can't do THIS with the Import Wizard!

But first we'll warm up with an intro-

Continued on page ten

Inside Uplink

Reassessments

Although *Newsday's* series on disproportionate property taxes (see story this page) may sound difficult to tackle for smaller news organizations, the topic is one that can be looked at anywhere. David Slade, of *The Morning Call*, explains how to look for property tax disparity in your area.

SEE PAGE SEVEN

Mapping

Those maps we all remember from elementary school weren't always right. The same goes for maps used in the real world. As Andy Lehren of *Dateline NBC* explains, the accuracy of a map depends on its type of projection.

SEE PAGE THREE

Perl

Most reporters aren't heavy users of Perl. But as Matthew Ericson of *The Philadelphia Inquirer* shows, Perl is a great – and free – tool for cleaning up data.

SEE PAGE EIGHT

Tech Tip

In the second part of his series on Access forms, David Herzog of the *Providence Journal* details how to create a program for sharing data with other reporters.

SEE PAGE FOURTEEN

Uplink

June 2000

Volume 12, Number 5
A newsletter of the National Institute for Computer-Assisted Reporting

EDITOR
Brant Houston

DIRECTOR OF PUBLICATIONS
Len Bruzzese

MANAGING EDITOR
Mary Jo Sylwester

ASSOCIATE EDITOR
Paul Monies

ART DIRECTOR
Kerrie Kirtland

SUBSCRIPTION ADMINISTRATOR
John Green

Uplink is published every month by the National Institute for Computer-Assisted Reporting, 138 Neff Hall Annex Columbia, MO 65211. (573) 882-0684. Subscriptions are $40 for IRE members, $60 for nonmembers.

Postmaster: Please send address changes to NICAR. Send e-mail to jgreen@nicar.org

NICAR is a joint effort of Investigative Reporters and Editors and the University of Missouri School of Journalism.

NICAR services include hands-on newsroom training in computer-assisted reporting, special academic and advanced training in data analysis.

MENTAL HEALTH

Death and dollars

By Debbie Cenziper
The Charlotte Observer

It seemed like a simple question at the time: How many people are dying under questionable circumstances in the North Carolina mental health facilities they turned to for care?

In July 1999, *The Charlotte Observer* decided to find out.

Unfortunately, no regulatory agency, advocacy group or mental health organization tracked deaths of the mentally ill and developmentally disabled in North Carolina, despite the more than $1.4 billion the state spends annually on its public health system and $500 million on private facilities.

With no one watching, dangerous trends went unnoticed, and negligent or abusive caregivers unpunished.

The state spends more than $1.4 billion annually on its public health system and $500 million on private facilities.

I turned to the state's center for health statistics for a database that offered information pulled from death certificates. The database had 370,000 deaths – every death in North Carolina since 1994.

With assistance from *Observer* database editor Ted Mellnik, I used Access to filter the database for deaths in state-run psychiatric hospitals. Then, we filtered that list by cause of death, and came up with a list of questionable deaths, deaths by such things as choking, drowning and suicide.

Unfortunately, there was no easy way to filter for deaths in the more than 3,300 psychiatric hospitals, group homes and other facilities not run by the state. So I went back to health statistics and learned of a more detailed database that listed each decedent's death certificate number. With that, I was able to look up death certificates in microfiche and learn more details about state hospital deaths, such as if the case was referred to the medical examiner. And – just as crucial – we were able to find out from the death certificates the address where the death took place and where the decedent was living.

I went back to the database and filtered by questionable deaths in hospital emergency rooms. Many mental health patients are injured in mental facilities, but die in general hospital emergency rooms. Several thousand deaths came up, and I searched death certificates by hand to cross-match decedents' addresses with a computer list of private mental health facilities.

I also tracked down 911 reports across the state, picking police departments with large psychiatric facilities in their jurisdiction. With only minimal information from the police report, such as the date of death, we were able to use the database and find names of the victims. Then, we went back to the death certificates to gather the rest of the information.

I was also tipped off to several deaths, but sources didn't have names. I used the database and death certificates again and again to track deaths with only minimal information.

The project took seven months to complete. Most of the time was spent simply researching the deaths, using autopsy reports, death certificates, police reports and other records, and working with medical experts to determine which deaths should be included in our final list. A five-part series, called "Broken Trust," ran Jan. 23-27, and results came quickly (IRE Resource Center Story #16409).

The state found $1.6 million to hire 27 new mental health inspectors. Politicians called for reforms. Gov. Jim Hunt is forming an investigatory group to probe all questionable deaths in mental health facilities, and has ordered the state to create a public database with all deaths of the mentally ill and developmentally disabled.

Debbie Cenziper can be reached by e-mail at dcenziper@charlotteobserver.com

The right projection

By Andy Lehren
Dateline NBC

You may remember those big plastic maps that hung on your elementary school wall - the ones where Greenland was as big as South America.

Or you've been messing around with mapping software and notice it puts New York about 600 miles further than it really is to Los Angeles.

What's going on? Was your elementary school actually right? Is your computer telling you that L.A. is further out than you thought?

This shows how you can get tripped up with maps.

The short version of the problem: The world is round. Maps are flat.

When map makers try to convert our sphere onto a flat sheet of paper – or computer screen – something has to give. Maps are always going to get something wrong. The trick is getting them to give you the right answer to your question.

Don't hesitate to befriend a local geographer, perhaps at a university or planning agency, to make sure you are using the right projection.

The things that maps can get right or wrong is the shape of a region, its area, the distance from one place to another, and the direction.

The solution is picking the right projection. Every projection gets something right. And every projection gets something distorted. No projection can do all things perfectly all the time.

The mapping column in the March issue of *Uplink* showed how different maps use different calculations for the shape of the Earth. This is called datum. A latitude and longitude from one popular datum could be several hundred feet off from the datum used by your hand-held GPS unit. If you didn't know that, you could mistakenly report, say, that the scene of a flood was outside a town's official flood plain. That column outlined how to convert a map's datum.

Projections

Projections are different than datum, though they are often mentioned in the same breath. Datum are used for knowing where any measurement sits on the planet. Projections try to convert our planet into a flat map. Every electronic map has underlying datum. But not every map is projected. And projections – which are essentially a lot of math – are always built on top of some underlying datum.

There are hundreds of projections. And the things they try to get right, as they flatten our planet onto our computer screen, are: area, distance, direction and shape.

The map on your elementary school wall, with Greenland the size of South America, was not doing a good job of representing area. Put a coin on part of Greenland, and it might cover eight times the amount of land in real life, than if you put that same coin on part of South America. The computerized map that calculated New York much further than reality to Los Angeles was distorting distance.

Yet these same maps may do an excellent job of preserving shape and direction. Sailors used these kinds of maps to navigate. They could follow a straight line on their map and stay on course as they curved around the world.

If you've negotiated for a map, and a government agency tells you they keep all their maps in a State Plane coordinate system, that's a cue that the agency's maps are already projected.

For instance, a North Carolina map using a state plane coordinate system uses a projection called Lambert Conformal Conic. It's a projection that works well for elongated states. It does a decent job for measuring distances, and has minimal distortion of shape and area in small settings. But the direction from the northwest corner of the state down to the southeast corner is distorted; it's an

Continued on page four

Here are several books that are helpful guides for choosing the right projection:

Matching The Map Projection To The Need, by Arthur Howard Robinson and John P. Snyder. Published by the American Cartographic Association.

Map Projections – A Working Manual, by John P. Snyder. Published by the U.S. Geological Survey.

An Album of Map Projections, by John P. Snyder and Philip M. Voxland. Published by the U.S. Geological Survey.

Cartography: Thematic Map Design, by Borden D. Dent. Published by WCB/ McGraw-Hill.

Continued from page three:

Projection

arc, curving around the planet, rather than a straight line.

To make matters more confusing, not every State Plane uses the same projection, so you would have trouble edge matching a series of State Plane maps if you don't run conversions.

If you've heard that a map's projection is "Geographic" – like many U.S. Census Bureau geographic files – that means they are not projected. You'll need to project them depending on the analysis you want to do.

In any case, be sure to check out a map's metadata – the detailed file describing any electronic map - to learn about its projection before any conversion.

Converting maps

Don't convert your map if you don't need to. Think about the analysis you need to undertake. The reason is that every conversion involves estimation, and the more you convert a map from its original projection, the more locations could be imprecise.

Sometimes the name of the projection tells you what it does best – "cylindrical equidistant" or "Albers equal area." Direction tends to be best preserved with azimuthal maps. Shape is represented well with conformal maps.

Keep in mind that there is always some amount of estimation involved in using any projection, so it's important to be familiar with the projection you are using.

If you are only analyzing a small area, you may be able to get by using just your local Universal Transverse Mercator zone or State Plane coordinates. Keep in mind that there is always some amount of estimation involved in using any projection, so it's important to be familiar with the projection you are using. Also, to spare you a shock: a given projection will no longer refer to a point in its latitude and longitude, but in its own coordinate system.

Some handy reference books are listed on the margins of this column. Also, don't hesitate to befriend a local geographer, perhaps at a university or planning agency, to make sure you are using the right projection.

Analysis

When it comes to analyzing data, you'll probably be most concerned with preserving distance and area. Distance would be a concern when, say, you want to buffer to determine what's within some miles of a hazardous chemical spill. Area would be an issue when, for instance, you want to calculate the population density within that buffer.

As discussed in the March column, the new version of ArcView comes with something called the Projection Utility. You can run it separately from ArcView. Just click on the icon on your desktop.

Once you're in the utility, the first step is to point and click your way to the shape file you want to convert. Hit the next button. For step two, first check the box that says "Show Advanced Options."

Let's assume the data is not projected, so keep the coordinate system at "Geographic." Click the datum tab and highlight the correct datum, say North American 1927. Hit the next key. For step three, select the projection you want to convert to, like a particular UTM zone.

The selection should automatically fill in the correct datum, but it's still a good idea to check and be sure. Hit next. Step four is simply to navigate to where you want to store your converted files. Now you can hit the next key and then finish. The utility will create a reference file that keeps track of the projection and datum for your new file.

The conversion utility is not perfect. Those who wrote the code for this utility noted it may be off by a meter or more.

Andy Lehren can be reached by e-mail at alehren@nbc.com

Mapping Bootcamp
Oct. 20-22
Columbia, Mo.

You'll learn the basics of mapping, geocoding and spatial analysis using ArcView GIS.

More information and registration details are available at *www.ire.org/training* or by calling (573) 882-0684.

Also learn more about mapping at the next IRE and NICAR National CAR Conference, Sept. 14-17 in Lexington, Ky.
See more information at *www.ire.org/training/lexington*

Track federal funds

By John Wilkerson
University of Missouri

Following the money is one of the mainstays in investigative journalism. When following money through the government, federal funding is often the spring that feeds the flow of money through all levels of government. Analyzing the Consolidated Federal Funds Report data (CFFR) can help put state, county and city government funding into perspective and give local reporting added depth.

The Bureau of the Census collects data from other local and state agencies and compiles the data into the CFFR. The data covers the Federal Government's expenditures and obligations at the state, county and subcounty (municipality, township) level.

Each geographic level is identified by name and a Federal Information Processing Standard (FIPS) code. County and Subcounty FIPS codes are only unique on the state level. For example, there are 50 states with the FIPS county code "001," so if you are running a query on a county with the code "001" you will be running a query for all 50 counties in the US with that code if you do not SELECT both fields. The same is true for the FIPS subcounty code field.

Analyzing the Consolidated Federal Funds Report data can help put state, county and city government funding into perspective and give local reporting added depth.

CFFR data also includes:

- Congressional district codes that identify where money is spent.
- Object codes, which are broad classifications of what money is spent on, such as grants, procurement contracts and insurance. There are nine object codes, and they generally correspond with the classifications used in the annual federal budget.
- Program codes that identify the various programs money is spent on.
- Agency codes that identify what agency received money.
- Population of the geographic area that received money.

The data does not tell you who or what company receives money. This kind of information can be found in the Federal Assistance Awards Data System (FAADS) and the Federal Procurement Data System. NICAR carries the procurement data, and FAADS can also be downloaded at *www.census.gov/govs/www/faads.html*.

NICAR has CFFR data available from 1986. The data has changed over the years, but 1986 was the first year that the federal funds report accounted for most grants on the local level.

Jerry Zremski of *The Buffalo News* analyzed all 13 years of the data to get an historical, geographic understanding of the data. T*he Buffalo News* series showed that showed Buffalo is part of a swath from the Northeast to the Midwest that has been receiving less and less money. At the same time, the federal government has been feeding more money into the Sunbelt.

A small number of the programs in the data are recorded on the state level even though the money for these programs is allocated on a county and local level. You can find these programs listed in a book put out by the General Service Administration.

In other instances, where the money ends up can be deceiving. Frank Kummer of the *New Jersey Courier-Post* was looking for towns that had received the most federal money, and how it was spent. He found that in some cases, federal money is recorded as going to a town or municipality but doesn't get used there. Kummer found that Paulsboro, N.J., a rundown town, accounted for a big part of federal spending. This was because a government contractor happened to be located in Paulsboro. The money passed through the company but was dispersed throughout the area.

Kummer also points out that CFFR data tells us only where the federal government spends money, not where it gets money

John Wilkerson can be reached by e-mail at gnilrits2@yahoo.com.

What you should know about CFFR data:

- **Program codes change. For program additions/deletions/changes, go to:** *www.cfda.gov/public/cat-changes.htm*

- **The population field is not accurate, especially on the county and subcounty level. This is because the population figures come from different years. If you want to calculate per capita information, use the population figures at:** *www.census.gov/population/www/estimates/co_99_1.html*

For any questions concerning the CFFR data, there is an in-depth record layout at: *www.census.gov/govs/www/cffr99.html*

Continued from page one:

A taxing issue

creases to 2 percent per year until the home is sold again, even if the market value soars.

Quirky results

In Nassau County, different laws brought about similarly quirky results. To determine which school districts were overassessed, we calculated each home's assessment ratio, which is the assessed value of a home divided by the market value, in this case, the sale price. A home with an assessment of $5,000 and a sale price of $150,000 would have an assessment ratio of 3 percent.

Using those figures, we calculated the county's median assessment ratio, which turned out to be 2.64 percent. Homes with assessment ratios above 2.64 percent are overassessed, while homes below that figure are underassessed. This is the same method the state uses.

You might also be sitting on a great tax-assessment story in your area because government officials may not be reassessing homes often enough, resulting in an unfair system.

New York state is quirky. It doesn't say that each county should assess homes at, say, 10 percent of market value. So we had to calculate what a fair percentage would be. It turns out to be the median of all of the assessment ratios in the county.

If your state says that homes must be assessed at a particular percentage of market value, you can skip the previous step of calculating the median assessment ratio. Just use the rate the state requires. If a home sold for $150,000, and the state says assessments must be 10 percent of market value, the home should be assessed at $15,000 ($150,000 x 0.10).

Then we created tables identifying which school districts are overassessed, and which ones are underassessed. Using statistical software (SPSS), we were able to show that minorities were more likely to be overassessed than whites. In areas where blacks and Hispanics made up four-fifths of school enrollment, 77 percent of schools were overassessed, far higher than the 46-percent figure for areas where those minorities made up less than a tenth of the school district.

Variance turned out to be a useful function to find out which school districts — or even which streets — had the goofiest assessments.

Variance indicates how much a given set of numbers varies from the average. A list of four numbers — 5, 4, 5, 5 — has little variance (0.25, to be exact) because most of the numbers are close to the average of all of them (4.75). Another list — 5, 10, 25, 46 — has a much bigger variance (339, to be exact.) because most numbers are far from the average, 21.5.

Don't worry about how to calculate variance. Access does it for you. Excel also provides a Var function.

Why is variance useful? Because homes with similar selling prices should have similar assessments, giving an area a low variance. An area with a high variance is one you might want to send some photographers to.

Remember, we're only interested in uncovering areas with a hodgepodge of assessments among homes that sold for about the same amount.

To do so, run an Access query grouping the homes by selling price, street and ZIP code. Add the assessment ratio to another column, and instead of "group by," choose "Var," and sort this column in descending order.

This will produce a table showing the streets with the wackiest assessments on top. You can also run the query using the selling price rounded to the nearest $10,000 to find homes with different taxes and similar — but not exactly the same — selling price.

As you call homeowners, you might want to let them know how much they're overassessed or underassessed? That's easy. Just calculate how much their assessment ratio of, say, 3 percent exceeds 2.64 percent, the

Continued on page twelve

Covering tax assessments:

- *Newsday's* property-tax assessment stories are at *www.newsday.com/special/assess/taxmain.htm.* For this story, click on "Uneven Tax Burden" on that page.

- Find sources for assessment around the country at *www.orps.state.ny.us/reflusamap/index.htm.* The site also has links to professional organizations, such as the International Association of Assessing Officers.

Exposing flaws

By David Slade
The Morning Call

Newspapers routinely devote ink to stories about property tax increases and property tax reform, but sometimes overlook the question of whether properties are correctly and fairly valued for taxation.

Often they are not, and you can prove it using the databases of tax assessments maintained by county governments in any state.

In Pennsylvania, counties aren't required to update property values, and they routinely go decades without a reassessment. As a result, properties carry tax assessments bearing little relation to their relative value. (See Richard Dalton's story on page one)

Even in states like Ohio that do require regular reassessments of property values, there can be wide disparities, as *The Columbus Dispatch* reported in two-day series in March.

In the two counties I have examined in Pennsylvania, people who owned similar homes purchased for similar amounts often had substantially different assessments, meaning they were effectively paying different tax rates.

The database gave us the facts, but also gave us the names of property owners, whose comments helped tell the story.

I was part of a reporting team at *The Scranton Times* that analyzed tax assessments in Lackawanna County in 1997. We found so many examples of assessments that were clearly uneven and unfair that it was hard to decide which ones to use.

Waiting to be told

In county after county throughout the United States, the same stories are waiting to be told. The databases can expose serious flaws that go to the heart of local systems of taxation, and even when they don't, they are tremendous resources.

The databases typically show who owns every property in a county, what they pay in taxes, when they bought the property, and what they paid for it, the mailing address of the property owner, information on tax-exempt properties, and more.

At *The Scranton Times* we used Microsoft Access and Excel to search and sort Lackawanna County's database. Most of our useful information came from comparing purchase prices of recently sold properties to tax assessments that were in many cases set in the 1960s, and from researching properties with unusually high or low assessments.

The database gave us the facts, but also gave us the names of property owners, whose comments helped tell the story. Most of the people interviewed had no idea their properties were over – or under assessed, due to their confusion about assessments, mileage rates and taxes.

We also used the database to check out tips about people whose homes weren't taxed at all, like the brother of a local tax collector. And we found countless properties that were over – or under assessed due to mistakes or deliberate policies of the county assessor's office.

I'm now with *The Morning Call*, where I'm part of a team covering Carbon County's first tax reassessment in 31 years.

Among other things, we'll be using assessment databases from before, during and after the reassessment to track the winners and losers as the property tax burden shifts.

We'll also look at the flaws in the previous system, which are similar to the problems in Lackawanna County, but we won't be focusing on them because more than 600 property owners already demonstrated that the system was beyond repair and convinced a judge to order the reassessment.

What to expect

Here are some things you might encounter if you decide to do a tax assessment database project:

- Counties to overcharge for the data.

Continued on page thirteen

The assessment series that ran in *The Scranton Times* can be found at *www.nepanews.com/stories/Tax%20assesment/taxseries.htm*

Coverage of Carbon County's reassessment can be found on *The Morning Call's* Web site, *www.mcall.com.*

PROGRAMMING

The Perl-y gates

By Matthew Ericson
The Philadelphia Inquirer

Ok, it wasn't as thrilling as the chariot race in Ben Hur, but we found it exciting anyway.

Tom Torok, then *The Inquirer's* Visual Basic guru, and I, a Perl fanatic, squared off to see whose programming language could most quickly parse a report into a tab-delimited format. The results, as reported by Torok to the NICAR mailing list:

"Matt Ericson and I just had a Perl vs. VBScript race on parsing a 130 meg report-form file into a tab-delimited file suitable for import into SQL Server.

The outcome: Matt (and Perl) beat the pants off Torok (and VBScript). His Perl script parsed nearly 750,000 records in less than half the time (with less than half the code.)

Programmers discovered Perl was a great language to use when creating dynamic web pages — it's also a great tool for cleaning up data in computer-assisted reporting.

I'm not only sold, but I also have to dress like a chicken for a week."

Perl – "Practical Extraction and Report Language" or "Pathologically Eclectic Rubbish Lister" – is a programming language foreign to most users of Microsoft Windows. It started as a utility to read text files and produce reports on Unix-based computer systems. With the growth of the World Wide Web, programmers discovered it was a great language to use when creating dynamic Web pages.

It's also a great tool for cleaning up data in the computer-assisted reporting world. Here's a few reasons it works so well:

Quick, free and competent

Perl has a wide range of built-in functions for dealing with data in many formats – fixed-width text files, delimited text files, queries from databases – and then dealing efficiently with the data.

You don't have to worry about whether you've defined your variables as the right types – all variables are the same type in Perl – so you can use the same variable once as a number and later as a string. Perl doesn't care – it converts it transparently. A string that grows longer than 255 characters? Perl will expand it on the fly. The same goes for arrays – they can grow as big as needed.

And while it's not as fast a language as C, it more than holds its own with Visual Basic. And, unlike C, you can write useful, short programs quickly and easily.

Plus, it's free. How can you beat that? (Download it from *www.perl.com.*)

Data in many ways

Perl has myriad ways to read data in text files.

By default, it reads line-by-line. But if a record spans multiple lines, you can change the end-of-record delimiter. Or, read the whole file in one fell swoop.

Is the data tab-delimited? Then use Perl's split function to break a record into its respective fields.

Delimited by something other than tabs? The split will handle that, too. Just tell it what character (or characters) is the delimiter.

Is the data fixed-width? Use the unpack function to pull it apart.

Perl has drivers to access most types of databases. There are drivers for Microsoft SQL Server, Oracle, Sybase and others.

Plus, if you're using Perl on Microsoft Windows, you can use ODBC and ADO just like you can in Visual Basic (with a few slight syntax changes). And, with SQL Server, Perl is an option for writing Data Transformation Services scripts.

Powerful search and replace

Perl's strongest feature is its use of "regular expressions," a fancy Unix term for an extremely powerful search-and-replace mechanism.

Continue on page nine

Where to get started:

• The Perl Web site is at *www.perl.com.* You can download the latest version (it's free) and access a wide range of reference materials.

• For Perl for Microsoft Windows, you can find it and documentation at *www.activestate.com.*

National CAR Conference
Sept. 14-17, Lexington, Ky.
Are you registered yet?
Get a registration form and more information on the conference at *www.ire.org/training/lexington*

Continued from page eight:

Perl-y gates

Unlike most programs, where search--and-replace means searching for a word and replacing with another word, regular expressions let you search for patterns of characters.

It can, for example, find a pattern of 2 digits followed by a letter and then use pieces of that search in the replacement — taking that letter and putting it before the 2 digits, for example.

Powerful associative arrays

Another unusual Perl feature is its associative arrays – arrays that are indexed off of a string, rather than a number.

Most programming languages let you store data in arrays: you put one piece of data in the first element of the array, another piece of data in the second element, and so on. Associative arrays, also called "hashes," are arrays that are indexed not on integers, but on arbitrary string values.

Associative arrays:

For example, the array here is what you might store in an array named "color:"

```
$color{"apple"} = "Red";
$color{"banana"} = "Yellow";
$color{"grape"} = "Purple";
```

If you are accessing a database of fruit, you can find the color of the fruit by looking in the associative array. If the current record in the database is for an apple, and has a field named name_of_fruit, then $color{$name_of_fruit} will return "Red."

Associative arrays let you build lookup tables on the fly, and you can also use associative arrays to count things. If you are parsing a text file containing street addresses, you can keep track of how often a street appears in the file by adding 1 to the array for each street name:

```
($count{$street} = $count{$street} + 1)
```

When you're done with the file, to find out how often Broad Street appeared, print out $count{'BROAD STREET'}.

Sure, it's not going to replace Group By queries in SQL, but for quick-and-dirty counting, it can't be beat.

It flies on the Web

Many of the same functions that make Perl superb for data manipulation also make it a great language for generating dynamic Web pages. On Microsoft Windows computers, Perl can be embedded in Active Server Pages just like VBScript. Plus, you get access to all of the ASP objects – the Request objects to let you process data from an HTML form, the Server objects to open connections to ODBC and ADO databases, and so forth.

The bad news

What's the catch? The most intimidating thing to Perl newcomers is its cryptic syntax.

If you've grown up with C and Unix utilities, Perl will seem right at home. However, if you're coming from a DOS and Visual Basic background, the abundance of strange characters — all those tildes, curly braces, carets, slashes, backslashes and so forth – can be downright befuddling.

You end up with commands such as:

```
foreach $a ( @data ) { $a =~ s/^ +//; }
```

This makes perfect sense to Perl programmers, but resembles nothing you ever saw in Visual Basic. (FYI: that command removes any extra spaces from the start of each element in an array.)

It's worth it, though. Take the time to learn Perl and soon you'll find yourself cringing when someone suggests you write a script or Web page in Visual Basic. (Especially when you find that one-line Perl script can sometimes do more than 20 lines of VB.)

More resources:

- For beginners using Microsoft Windows systems, Learning Perl on Win32 Systems, by Randal L. Schwartz, offers a good place to start. It's published by O'Reilly and Associates.
- The bible of the language, Programming Perl, was written by Larry Wall, the language's creator, and is probably the most complete guide to the language. Also published by O'Reilly, it's commonly known as "the camel book" because of the picture of the camel on the cover. Compared to Learning Perl, it's less of a tutorial and more of a reference manual. While it assumes some Unix familiarity, it's still excellent for users who haven't used Unix.

Matthew Ericson can be reached by e-mail at mericson@phillynews.com.

Here's the command to split tab-delimited data into three fields:

```
($field1, $field2, $field3) =
split ( "\t" );
```

This unpacks a fixed-width record where the first two fields are 2 characters wide, and the third field is 4 characters wide:

```
( $field1, $field2, $field3 ) =
unpack ( "A2A2A4", $_);
```

Command to find a pattern of 2 digits followed by a letter and then use pieces of that search in the replacement — taking that letter and putting it before the 2 digits, for example.

```
s/(\d\d)([A-Z])/$2$1/
```

More about Perl will be included in the July/August issue of *Uplink*.

Continued from page one:

Modules

duction to the concept of modules for those who haven't looked into them before.

A quick 'Hello'

Launch Access and open the exercise file (Henderson.mdb) or open any Access database. The exercise file can be downloaded from the NICAR Web site, *www.nicar.org.*

Click on the Modules tab and hit the 'New' button.

You may see a couple of lines of text like this:

```
Option Compare Database
Option Explicit
```

Don't worry about what they mean for now. UNDER those lines, type this:

```
Sub SayHi()
```

And press 'Enter.' Notice that as you move off the line, Access colorcodes the keyword 'Sub' (for subroutine—a kind of program), adds a line setting it off, and automatically adds another line:

```
End Sub
```

These are the starting and ending points of our program. All our program code will go between them, so type these lines there:

```
Dim theName As String
theName = InputBox("What's your
name?")
MsgBox ("Hi there, " & theName & "!")
```

Modules are your ticket to getting under the hood of Access

If you have trouble, make sure you didn't add spaces to 'theName', 'InputBox' or 'MsgBox' - these must be single words. Notice the way Access follows your typing and nudges you in the right direction at every turn. If you mistyped or moved off a line before it was finished, Access complained. Menus appeared and guided you at various points.

Now, to run your program hit F5 (or use Run...Go/Continue, or the arrow on your toolbar).

It may not be the most useful program in the world, but you've already declared a variable, assigned it a value based on user input, and given feedback to the user. You'll use these basic building blocks time and time again in any programming environment.

If you're intrigued and want to get more familiar with Visual Basic for Applications (VBA) before continuing, I suggest the introductions and tutorials at *www.athree.com/modules/index.html*

Subroutines and functions

Let's talk quickly about subroutines. As in the example above, they run independently and leave no trace behind when they're finished. We're going to stick to subroutines here because they're easier to explain. But you should know that you can also create functions to supplement the Access built-in functions, like Trim() and Substring().

The coding procedure is very similar, and once created in a module, the functions become available in all your Access queries, just like the built-in functions.

For examples of Functions check the Neatcode link from Microsoft (see greybox on the side) which includes many helpful and instructive functions you can import into your Access files. Add your own code, and you'll have the foundation of a library of custom functions available whenever you query your data.

Newsroom database pros often build such a library to solve recurring problems like formatting text files for publication in a particular front-end system, a dependable way of calculating age from date of birth, or extending the Proper() function to do a better job of normalizing capitalization.

Importing tricky data

Now let's get into a meatier project, where we'll learn to use code to scan through a text document for fields to import to a database. This is the kind of thing you would have done in a FoxPro .prg file way back in 1995 when I joined NICAR.

Make sure pacer.txt (the text file available as part of the download from the NICAR Web site) is in the same directory as Henderson.mdb and open the Access file, if you haven't done that already. Open pacer.txt in a text editor like Notepad, make sure Word

Continued on page eleven

- List of articles available on Modules and Macros from Microsoft's Knowledge Base – lots of goodies here and advice on specific problems and approaches: *http://support.microsoft.com/support/kb/articles/Q162/0/68.asp*
- Nice module tutorials and history ofVBA: *www.athree.com/modules/index.html*
- Expert Approaches to VBA – only for serious geeks with programming experience and a good summary of the Access Object Model that underlies VBA: *www.ldfinfo.com/c11.htm*
- Sample modules for the whole family including the infamous "Age from DOB": *http://bpro.com/Modules/modules.htm*
- Link to neatcode.mdb *http://support.microsoft.com/support/kb/articles/Q148/4/02.ASP*

Continued from page ten:

Modules

Wrap is off and take a look.

What a mess, huh? We need information on cases in the table called Bankruptcy. The saving grace for the text file is that at least we can predict where the information, such as debtors' names, will fall within each line so we can pull them out with the Access MID() function.

Hit the Modules tab, select Bankruptcy and hit the 'Design' button to see the code. Examine the ImportBankruptcy() subroutine and run it with F5 to see that it does in fact fill the Bankruptcy table with values from the file.

A good place to start a job like this is not programming at all but documentation. Notice that there are lines in green that serve only as reading material for someone looking at the code – to do this, you precede the line with a single quote mark. It's called 'commenting' and it's what we all do best.

It's a good idea, to get started on a complicated coding job, to just go ahead and write what you want to do, step by step, in comments. For this job, it would look like this:

```
'Declare variables we need for all the
'jobs below
'Open the text file we want to import
'Read each line into a text variable and
'note the line number
'When we get to a line with fields for
'our table, pull them out into variables
'Put together an SQL statement to
'update the table
'Run the SQL statement.
```

Now you can start coding each task separately between the comments, and you've broken the job down so it doesn't seem quite so intimidating.

Inside and out

The cool thing about modules is that they let you automate tasks you could do anyway in the point-and-click part of Access, but also let you delve into areas beyond that.

For instance, part of what you're doing in this module is creating and running an SQL query, an Append query, that you could have produced in the query grid or in SQL View.

You do that with these lines of code:

```
Dim db As Database
Set db = CurrentDb
```

(code making an SQL statement and assigning it to the variable 'theSQL')

```
db.Execute theSQL
```

But at the same time modules give you the power to create and manipulate text files, and otherwise interact with Windows in ways you can't with interactive Access. We open our import file and read it in line by line with this code:

```
Dim myFile As Integer
Dim myFileName As String

myFileName = "pacerfile.txt"
myFile = FreeFile

Open myFileName For Input As
#myFile
```

To open a file, you need to give it a numeric 'File Handle' and the FreeFile function assigns it a free handle automatically so you don't need to worry about it. If you wish, you could accomplish the same thing this way:

```
Open "c:/data/uplink/pacerfile.txt"
For Input as #1
```

"For Input" just means that you're going to read stuff from the file into your program. See Help on the Open statement for more options.

Once we have the file open, VBA doesn't give us a straightforward way (that I know of) to move to a particular line in the file. But the Line Input function will read a new line into a variable every time we call it, so we deduce the line we're on and keep track of it in the 'linecount' variable.

For efficiency you'd probably want to stop the Line Input statement after it reaches Line 8 – since we don't need to read any more of the file at that point – but I left it as is to make the code more portable for your own purposes.

Once we've gathered the data we need into

Continued on page twelve

Dim lights, thick smoke

Here are a few coding conventions to help you understand what's going on in the exercise code:

Sub () and Function()
A sub, or subroutine just means you want to do a job but you don't need a result reported back from that job. If you need something that gives a result back when you run it, use a function.

Dim ...As (type)
For Dimension; means essentially "Set aside some space in memory for a widget I'm going to need in this program." You need to specify the type – character string, integer, etc. – to tell the module how much space it will need. This is called "declaring a variable."

More coding conventions are available with the download at *www.nicar.org*.

Recent CAR stories:

Investigation of Michigan grocery stores
Detroit Free Press
April 7, 2000
www.freep.com/news/metro/groc7_20000407.htm
According to a *Free Press* analysis of the Michigan Department of Agriculture records "more than 2 million pounds of spoiled, contaminated or otherwise unsafe foods have been seized from 590 Michigan grocery stores during the last two years." Resource Center story #16523.

"Home Values"
The Seattle Times
March 5-12, 2000
www.seattletimes.com/homes/
The project, analyzing the Seattle-area housing market, involved the analysis and mapping of county assessor's data on 99 geographic areas. Resource Center story #16511.

Continued from page six:

A taxing issue

countywide median. (3.0 - 2.64) / 2.64 = .136, or 13.6 percent.

Then call up the homeowner. And be prepared to listen to an irate taxpayer.

Databases required

Two databases were central to the story:

• A database of property tax assessments and sale prices from the New York State Office of Real Property, whose mission in part is to ensure fair tax assessments by local governments. Each state has a counterpart to this agency. (See tips box on page six or links to your state.)

• A database of property taxes from Nassau County.

These two databases were linked using the section, block and lot number of the property. When requesting databases, ask to have the section, block and lot as three separate fields.

We supplemented those databases with several others:

• A database of the age of homes obtained from a private company. This allowed us to determine that pre-war homes tend to have lower assessments.

• A database of the racial makeup of school districts from the New York State Department of Education. The data enabled us to gauge differences in taxes based on race.

Maps and worksheets

Based on data in Microsoft Excel spreadsheet, and using ArcView mapping software, we produced a map showing the percentage of overassessed homes in each school district.

By culling the Microsoft Access database, we found good examples of homes with the same selling price but vastly different taxes. The date of construction also was a factor because some discounts applied only to older homes.

Using Excel and Access, we produced a table that showed the median selling price, the median assessment ratio, median taxes and the degree to which homes are overassessed or underassessed in each school district.

We produced a worksheet so homeowners could calculate how much they are overtaxed or undertaxed. We assumed that if someone was overassessed by, say, 10 percent, that they were also overtaxed by 10 percent.

Richard Dalton can be reached by e-mail at rdalton@newsday.com.

Continued from page eleven:

Modules

variables, we parse those variables into an SQL statement and run it.

I've included just one file, with fictitious names, but you could easily loop this code through all the files in a directory to populate a database.

Bugging out

The MsgBox function from our first exercise is a great way to handle the inevitable bugs that pop up when you write any kind of code. Here we've commented out a section that shows us our SQL in a message box, rather than executing it, so you can be sure it's OK before you run the program.

That's how I figured out I need to surround the text values with single quotes, and the dates with pound signs, in order for the SQL to successfully transfer the values into the table.

If something's going wrong with your program, a message box here and there can report the value of various variables that might be causing you trouble and will save you hours of agonizing.

Moving along

Now that you've been titillated by the possibilities of modules and VBA, try following some of the links with this article and check out some of the sample modules supplied by Microsoft in Neatcode, free for the downloading. Take a look at some of the online tutorials and take advantage of your Help files to learn about specific functions available only in VBA. And the NICAR-L crowd includes masters of VBA ready to respond when you get stuck. So go ahead, code yourself silly.

Tim Henderson can be reached by e-mail at thenderson@herald.com

Continued from page seven:

Flaws

The Scranton Times bought Lackawanna County's database on nine-track tape for $200. *The Morning Call*, after initially being asked to pay about $1,500 for a 46,000-record database in Carbon County, paid $250 after threatening to sue over the cost.

• Dirty data.

In counties that haven't reassessed for 30 years, much of the information in the database will have been typed in from decades-old paper files. In the assessment databases I've worked with the name and street address fields were incomplete and full of misspellings, and other fields, such as purchase price and property size, were often blank.

Regardless of appeals, it's worth checking the assessments of local big businesses and politicians.

The good news is that properties are typically identified by a unique number, like a social security number for properties, that ties each record to a map location and property record.

• The need to check paper files.

Plan to check the county's paper files on each individual property you plan to write about. Between the time you request the data and the time you report on it, assessments and property owners can change.

Also, the paper records often contained more information than the electronic ones. Lackawanna County's paper records included handwritten notes. My favorite paper record in Lackawanna County was an assessor's card noting that a house had a low, reduced assessment because it had burned down. A more recent note on the same card showed that the house, which we found had been renovated, had just sold for more than $100,000.

• Learn the local rules.

Counties often have quirky assessment rules, often dating back decades. Knowing them before you start searching the data will save you time, and some of the rules might be worth a story.

For example, in Lackawanna County corner lots are assessed at 25 percent more than non-corner lots. And church rectories are assessed at 10 percent of true value.

Quick tips

• In Pennsylvania, talk to the State Tax Equalization Board to get the big picture and learn about the common-level ratio, a formula used to level out old assessments with new ones. Also check with the Pennsylvania Economy League, which keeps track of property tax reassessments. Similar organizations may exist in your state.

• Check the courthouse for lawsuits against the county assessor's office. Large companies often appeal tax assessments.

• Sort your database by municipality, or by school district, to see how property taxes fall in different parts of the same county. Because assessments don't change when a property is sold in Pennsylvania, areas that have prospered since the last reassessment tend to have relatively lower assessments than areas that have not.

• Check out the top businesses and politicians.

In Pennsylvania, tax assessments can be reduced by appeals boards filled with political appointees, so it's worth checking to see who has received a break.

Regardless of appeals, it's worth checking the assessments of local big businesses and politicians. In Lackawanna County, I discovered through the database that two huge landfills were each paying less in taxes than a typical large retailer would. One landfill, worth an estimated $600 million, was assessed at a lower value than a local hospital's parking garage because the county assessor's office considered landfills to be unimproved land – a position other that assessors elsewhere scoffed at.

David Slade can be reached by e-mail at david.slade@mcall.com

Tipsheets:

"Explore Home Ownership Using CAR," by Neill Borowski of The *Philadelphia Inquirer*, 1999 NICAR Conference, tipsheet #878.

Other stories on reassessments:

"Tax Trouble," *WTAE-TV* (Pittsburgh) uncovered Allegheny County's mismanagement of what some say is a corrupt tax assessment system that forced some people to pay more than they should, while others pay too little. Story #14517 on tape.

"Armstrong Assessments a Question of Fairness," *Valley News Dispatch* (Tarentum, Pa.), looks at the reassessment of property in Pennsylvania. Story #13573.

These tipsheets and stories are available from the IRE Resource Center by calling (573) 882-3364. Search the databases at *www.ire.org/resourcecenter*

TECH TIP

Fronts for forms

By David Herzog
Providence Journal

Once you've learned how to use Microsoft Access to build forms for entering and querying data (See *Uplink*, April 2000), you can link them to create a program for sharing data with other reporters.

In this exercise, we'll build upon the chronology we created as part of the April Tech Tip by linking it to a startup screen. If you need a copy of the database we used last month, you can download it from the NICAR web site (*www.nicar.org*).

Before getting into the how-to, it's important to understand the structure of an Access database. An Access database is a container for objects. Tables, forms, queries and reports are the most commonly used objects. Access also includes macros (sequences of defined actions) and modules (Visual Basic code) to help you control programs. We'll use macros, which are powerful enough for our program.

Create the startup screen

Begin by opening the program and creating a new form. Once you've opened the database, click on the Forms tab, then New. Make sure Design View is selected, then click OK and a blank form appears. This blank form is your canvas for creating a startup screen. By setting the form's properties, you can change the color. By clicking buttons on the toolbar you can add text and graphics.

An Access database is a container for objects. Tables, forms, queries and reports are the most commonly used objects.

First, let's make sure that this startup screen launches whenever we start the program. Close the form and save it; call it Welcome. Now we will use a macro to make it the startup screen. Click on the Macros tab, then pick New. The Macro design grid opens. In the Action column we well specify what we want the macro to do.

Click on the drop down arrow and scroll down the list until you find OpenForm. Select it. Then, in the Action Arguments section on the bottom part of the screen, click in the Form Name box and select Welcome from the list. Keep all other settings as they are. Close the macro design grid and name it AutoExec. When Access opens a database, it looks to see the whether there is an AutoExec macro. If there is, Access runs it. Test your macro by closing your database, then opening it again.

Open the Welcome form again by clicking on the Forms tab, highlighting Welcome and clicking the Design button. Let's modify the form's appearance.

Change the color by clicking once anywhere on the form with the right mouse button to get the pop-up menu. Select Properties. Click the Format tab to get a list of the formatting properties you can set. Click on Back Color, then the ellipsis button to get a color palette. Select a color.

Now we will set the properties of the form to remove some of the frills. Click once on the form select box, the small square located at the intersection of the rulers at the top left corner. Click the right mouse button to get the pop-up menu and select Properties. Click on the Format tab to set the following properties:

Views allowed: Form
Scroll bars: Neither
Record selectors: No
Navigation buttons: No
Control Box: No

Close the form properties box.

Give the form a title by clicking on the Label button on the toolbox, then clicking on the form where you would like the title to go. Type "Digging for data" without the quotation marks, then hit Enter. Look at the text on your form. Obviously, this could look better. We can change the text properties by right-clicking the text, then clicking Properties in the pop-up menu. Use the Fore Color, Font Name, Font Size, Font Weight, Font Italic, Font Underline and Text Align properties to modify your label's appearance.

Now add a picture to your form. I grabbed

Continue on page fifteen

For more information on forms in Microsoft Access, check out the following articles on the Web:

- *http://eu.microsoft.com/AccessDev/articles/balter13.htm*
- *www.zdjournals.com/ima/9710/ima97a1.htm*

Continued from page fourteen: # Forms

the logo from NICAR's web site. Here's how I added it to the form: First I clicked on the Image button on the toolbox. Then, using the mouse, I drew a frame for the logo by left clicking on the form, dragging diagonally and releasing the button. I added the logo by finding the graphic file on my computer.

Button up

So far, all we've done is work on the window dressing. It's time to put the form to work with some buttons.

The first button will shut Access down. Click on the Command Button item on the toolbox, then use the cross to draw the shape of your button, just as you did to add a frame for your graphic. When you release the mouse button, the Command Button Wizard starts. The wizard makes creating command buttons a snap. In the Categories box, pick Application. In the Actions box, pick Quit Application. Then click on the Next button. In the next step, select the type of button you want, then click Next. At the next screen, click Finish.

Test your command button by switching to form view, then clicking it to see what happens.

Start the database again and open the Welcome form in design view, where we'll add a button that minimizes the startup screen and opens the Chronology form. We will link the button to a new macro that will manipulate the forms.

Click on the Macros tab and select New. In the first Action cell, click Minimize. This minimizes the active window (the Welcome form, in our case). In the second Action cell, select Open Form. In the Action Arguments section at the bottom of the screen, select Chronology in the Form Name option. Close the macro design view and give it a logical name, such as open_chrono.

Go back to the Welcome form in design view. Add a command button using the Command Button item on the toolbox. Draw a button on the form using your mouse and release the button to launch the Command Button Wizard. In the Categories box, select Miscellaneous, and in the Actions box select Run Macro and click the Next button. Select "open_chrono" as the macro to run and click the Next button. Design the button by clicking on Text and typing "Chronology" without the quotation marks. Click Next, then Finish.

Test your work by switching to the Welcome form's form view. Click on the Chronology button and see what happens. The Welcome form should shrink and the Chronology form should open. Now click on the Close button at the bottom of the Chronology form.

The Chronology form closes and the Welcome form remains minimized. We will add a macro and a button to make sure the Chronology form also re-opens the Welcome form. Create a new macro by clicking on the Macros tab, then clicking New. In the first Actions cell, select Close. In Action Arguments section, select Form as the Object Type and Chronology as the Object Name. In the second Actions cell, select Open Form. In the Action Arguments section, pick Welcome as the form name. Close the macro design view and save this as close_chrono.

Now we need to link the macro to a new button on the Chronology form. Open the Chronology form in design view. Click on its close button and then hit the delete key to get rid of it. Create a new button by clicking on the Command Button item on the toolbox and drawing the outline of the button using the mouse. When you release the mouse button, the Command Button Wizard starts. Pick Miscellaneous from Categories and Run Macro from Actions and click Next. Then select "close_chrono" from the list of macros and click Next. Select the style of button you want and click Next, then Finish. Close your form and save it.

It's time to see how everything works together. Close Access and open the database again. Use the command buttons to navigate between the Welcome form and Chronology form.

There's more to know about creating programs to share data (such as how to secure the program on a network). But using the principles in this Tech Tip and April's Tech Tip, you can add all the tables you need and build more forms for adding and querying data.

David Herzog can be reached by e-mail at david_herzog@projo.com.

Recent CAR stories:

"Diversity in the workplace"
Newsday
April 9-13, 2000
www.newsday.com/special/diverse/diverse.htm
Newsday **obtained a federal database of workers and managers in 1,500 companies and business divisions to examine the degree of diversity in management on Long Island.**

Political donations to "Virginia state legislators On The Lege"
April 24, 2000
http://saturn.vcu.edu/~jcsouth/on-the-lege/
Using Access and Excel, students of a legislative reporting class at Virginia Commonwealth University "analyzed all donations received during the 1999 election cycle by members of the Virginia General Assembly. They found that more than half of the contributions came from 150 donors – a small pool of political organizations, special-interest groups, corporations and individuals with deep pockets."

Bits, Bytes and Barks

IRE and NICAR training calendar

Here's a glimpse at some of the training opportunities IRE and NICAR will be offering in the coming months. Check out the IRE Web site, *www.ire.org/training*, for more events, details and registration information.

Aug. 4-5: Census Seminar, Columbia, Mo.

Aug. 6-11: CAR Bootcamp, Columbia, Mo.

Sept. 14-17: National Computer Assisted-Reporting Conference, Lexington, Ky. Registration forms and more information are available on the IRE Web site training page.

Sept. 29-Oct. 1: CAR workshop, jointly sponsored with the University of Florida, Gainesville, Fl.

Oct. 20-22: Mapping Bootcamp, Columbia, Mo.

Nov. 12-17: CAR Bootcamp, jointly sponsored with The Poynter Institute, St. Petersburg, Fl.

Dec. 8-10: Census Seminar, College Park, Md.

More information is also available by calling (573) 882-0684.

Data update

The NICAR database library has updated the National Bridge Inventory System database to include all of the 1998 inspections as well some 1999 inspections through December 1999.

This dataset includes maintenance records collected by the Federal Highway Administration for assessing the nation's bridges. It includes information about the bridge's structure, repair and last inspection.

Data for the entire U.S. costs $100 (under 50,000 circulation or 50-200 market), $125 (50-100,000 circulation. or 25-50 market), or $150 (100,000-plus or top-25 market). You can also order per state for $50 (Under 50,000 circ. or 50-200 market), $75 (50-100,000 circulation. or 25-50 market) or $100 (100,000-plus or top-25 market).

Contact the data library to order this or any other data set by calling (573) 884-7332. More information is also available at *www.nicar.org/data/bridges/*

NICAR on the Web

The NICAR data library has recently finished updating the information on its Web site about the databases available to journalists. Check out the improved site at *www.nicar.org/datalibrary*

For each database, there is a detailed description of the data, sample slices and record layouts for each table and information about how much it will cost.

We've also added useful information about stories others have done on this topic or using this particular data. It includes references to IRE Resource Center stories and/or tipsheets that might be useful when getting started on a similar story.

Investigative Reporters and Editors, Inc.
138 Neff Annex
University of Missouri
School of Journalism
Columbia, MO 65211