Preserving the Whole

A Two-Track Approach to Rescuing Social Science Data and Metadata

---

June 1999

by Ann Green,
JoAnn Dionne, and
Martin Dennis

ISBN 1-887334-68-8

Published by:

The Digital Library Federation
Council on Library and Information Resources
1755 Massachusetts Avenue, NW, Suite 500
Washington, DC 20036

Additional copies are available for $15.00 from the above address. Orders must be prepaid, with checks made payable to the Council on Library and Information Resources.

The paper in this publication meets the minimum requirements of the American National Standard for Information Sciences—Permanence of Paper for Printed Library Materials ANSI Z39.48-1984.

The Digital Library Federation

On May 1, 1995, 16 institutions created the Digital Library Federation (additional partners have since joined the original 16). The DLF partners have committed themselves to "bring together—from across the nation and beyond—digitized materials that will be made accessible to students, scholars, and citizens everywhere." If they are to succeed in reaching their goals, all DLF participants realize that they must act quickly to build the infrastructure and the institutional capacity to sustain digital libraries. In support of DLF participants' efforts to these ends, DLF launched this publication series in 1999 to highlight and disseminate critical work.

Donald J. Waters
Director
Digital Library Federation

About the Authors

Ann Green is director of the Social Science Statistical Laboratory at Yale University, where she oversees social science research and instructional technologies, facilities, and support services. She has participated in the development of standards for social science metadata through the Data Documentation Initiative. She is vice president of the International Association for Social Science Information Service and Technology (IASSIST). From 1989 to 1996, she was consultant and technical manager of the Social Science Data Archive at Yale. She was data archivist from 1985 to 1989 at the Survey Research Center, University of California at Berkeley.

JoAnn Dionne is the social science data librarian at the University of Michigan Library where she is developing and providing data services for the campus. From 1977 to 1998, she was the social science data librarian at Yale University where the Yale Roper Collection was an integral part of her responsibilities.

Martin Dennis is a Ph.D. candidate in psychology at Yale University. His main area of research is in human reasoning in general and causal induction in particular. In addition to his studies, he works as a part-time statistics and computer consultant at the Yale Social Science Statistical Laboratory, where his responsibilities include helping users to access Yale's collection of public use data sets.

Acknowledgments

We wish to thank Scott Redinius of the Yale Economics Department for his work on the TextBridge Pro portion of the project and his advice on OCR software, our scanning workstation, and developing evaluation procedures. Soo Yeon Kim of the Yale Political Science Department provided welcome editorial comments. Thanks also goes to David Sheaves at the University of North Carolina's Institute for Social Science Research, for helping us evaluate column binary data options and spread ASCII formats and for providing very useful SAS programs. We also wish to acknowledge the help of Marilyn Potter and Marc Maynard, from the Roper Center for Public Opinion Research, for answering questions, rushing us replacement copies of data sets, and sharing their xray program. We also acknowledge Kathleen Eisenbeis, former director of the Yale Social Science Libraries and Information Services, for her contributions to the early stages of the project. And lastly, to Donald Waters for his encouragement, advice, and support—thank you.

Contents

Preface

Quantitative data, including social survey results, test measurements, economic and financial series, and government statistics, are vital resources for research and education in a variety of disciplines concerned with advancing the study of individuals and society. For decades, these data have been encoded, stored, and used primarily in digital form. Custodians who have collected, maintained, and provided access to numeric data resources thus have been building and managing digital libraries—and scholars and students have been effectively using them in the pursuit of historical, social, and scientific studies —long before the term *digital library* came into wide currency.

Those who are grappling with an explosion of digital information in a dizzying range of formats have much to learn from social science data librarians and users who have relatively long experience in managing and working with digital resources. Data producers, librarians, and scholarly users have come to invest in very sophisticated mechanisms for storing and distributing social science data. They have achieved valuable economies of scale in data storage and delivery through consortial developments, such as the data archives held by the Inter-university Consortium for Political and Social Research (ICPSR). Through years of experience with repeated changes in storage technologies and in the software for encoding and using the data, they have become particularly adept at the long-term maintenance of information in digital form.

In 1996, the Task Force on Archiving of Digital Information highlighted the difficulties of preserving digital information over long periods of time. As a way of addressing these difficulties, the task force recommended in part that its sponsors, the Commission on Preservation and Access and the Research Libraries Group, seek to document the experiences of communities already well practiced in the preservation of digital information. Responding to this recommendation, the Commission, which has since merged with the Council on Library Resources to become the Council on Library and Information Resources (CLIR), sought out the expertise of those managing university-based data archives. It contracted for the development of this paper with the authors, who at the time worked together in managing the Social Science Data Archives at Yale University, one of the oldest data archives in American universities.

Preserving the Whole appears as the second publication of the Digital Library Federation and reflects the Federation's interests both in advancing the state of the art of social science data archives and in building the infrastructure necessary for the long-term maintenance of digital information. The paper is especially valuable as a meticulously

detailed case study of migration as a preservation strategy. It explores the options available for migrating both data stored in a technically obsolete format and their associated documentation stored on paper, which may itself be rapidly deteriorating. The obsolete data format known as column binary was born in the same era of creatively parsimonious coding techniques that have given rise to the widely publicized Year 2000 (Y2K) computer problems.

Beyond its contributions to our understanding of migration as a particular strategy for the long-term maintenance of digital information, *Preserving the Whole* also provides more general lessons. It is a remarkable finding of this study that the column binary format, although technically obsolete, is so well documented that numerous options exist not just for migrating column binary files to other formats, but also for reading them in their native format. Moreover, the authors make the important observation that data sets will be indecipherable and cannot survive at all, regardless of the file format in which they are stored, if there is no effort made also to preserve their codebooks. A codebook is essential documentation that relates the numeric data to meaningful fields and values of information.

From more theoretical perspectives, Jeff Rothenberg (1999) and David Bearman (1999) both emphasize the critical importance of documentation, or metadata, for preserving digital information. The value of *Preserving the Whole* is that it makes a similar argument, but concretely and from the long experience of the data community in effectively managing digital information.

Donald J. Waters

Background and Project Description

In December 1994, the Commission on Preservation and Access and the Research Libraries Group created the Task Force on Archiving of Digital Information. The purpose of the task force was "to investigate the means of ensuring continued access indefinitely into the future of records stored in digital electronic form." Digital media are more fragile than paper and become unreadable more quickly because of changes in operating systems and applications software and the deterioration of physical media, and because no organization has accepted responsibility for preservation. In its definitive 1996 report, *Preserving Digital Information*, the task force warned that "owners or custodians who can no longer bear the expense and difficulty of migration will deliberately or inadvertently, through a simple failure to act, destroy the objects without regard for future use."

The task force's warning echoed the growing realization by researchers who were using social science statistical data in digital form and specialists who were archiving these data that major rescue efforts to identify, locate, and preserve computer files produced with rapidly outmoded technology could not be postponed. Because access to social science numeric data requires metadata—accompanying paper or machine-readable records—the loss of the metadata can also mean the loss of the data file.

The two approaches to preservation of digital files under evaluation in the early 1990s were *refreshing* and *migration*. Refreshing refers to the copying of information from one medium to another without changing the format or internal structure of the records in the files. Refreshing digital information will suffice as long as software exists to manipulate the format of the files. Since digital information is produced in varying degrees of dependence upon particular hardware and software, refreshing cannot serve as a general solution for preserving digital information. The task force emphasized migration of digital information, "designed to achieve the periodic transfer of digital materials from one hardware/software configuration to another, or from one generation of computer technology to a subsequent generation." Migration includes refreshing the media but also addresses the internal structure of the files so that the information within can be read on subsequent computer platforms, operating systems, and software.

The Yale Social Science Data Preservation Project

In 1996, the Commission on Preservation and Access commissioned the Social Science Library and the Social Science Statistical Laboratory at Yale (Statlab) to identify and evaluate the formats that would most likely provide the ability to migrate social science statistical

data and accompanying **documentation**[1] into future technical environments. The Yale University Library, one of the first academic libraries to form a collection of machine-readable data, began acquiring social science numeric data in 1972. Over the years, Yale has copied its data from one form of digital storage to another as mainframe computer technology has dictated. The copying of data, while labor-intensive, was straightforward in creating exact logical copies from out-of-date media in newer data storage formats. In the mid-1990s, as data use was moving from the mainframe to distributed computing systems and from one hardware/software configuration to another, digital formats began to require not just simple duplication, but restructuring. Files produced by standard statistical software on mainframes had to be converted into platform-independent formats before moving to personal computers. In addition, data stored on magnetic tapes had to be moved to new media as access to and support in using the Yale mainframe was discontinued.

Our social science data preservation project team was headed by Ann Green (director, Statlab) and JoAnn Dionne (data librarian, Social Science Library) with the assistance of Martin Dennis (consultant, Statlab, and graduate student, psychology). We began our work in June 1996, during a time when many academic institutions were in the process of transferring numeric social science data sets from mainframe environments to PC- and UNIX-based networks. Large collections of numeric data had been successfully moved across these platforms. Considerably less attention had been directed toward the greater problem of developing system-independent archival formats, while also preserving and digitizing the accompanying paper records (metadata) that must be available to analyze the data sets.

We were thus faced with a two-track preservation approach: converting deteriorating paper (the documentation) to digital form, and migrating digitized numeric data to an archival format that can be read by future operating systems and applications software. The Yale University Library had taken a lead in digitizing for preservation (Conway 1996), and we built on that base in digitizing the paper records accompanying the data file. The Statlab had taken a lead in migrating data collections from mainframe-dependent tape storage to networked online storage, and we built on that base in restructuring and migrating the numeric files.

On the documentation track, we scanned printed textual material for 10 surveys selected from the Yale Roper Collection and evaluated the outcomes of applying optical character recognition (**OCR**), creating image files, and producing Adobe Portable Document Format (**PDF**) files. On the data track, we investigated diligently and in detail the implications of preserving data in their original format vs. migrating to restructured formats. We evaluated the alternative formats for migrating the original data files from tape and focused upon the benefits and drawbacks of each alternative. Details are covered in the Findings and Recommendations section of this report.

[1] At its first occurance, a word defined in the Glossary is shown in **bold**.

While evaluations of computer storage media should not be ignored in an overall strategy for planning the future costs and viability of data collections, we did not include media evaluations in this project. Nor did we research the intellectual property issues involved in conversion, leaving that to a later discussion. However, there is a long-established ethic in the social science data community that data documentation should be shared freely. For example, the Inter-university Consortium for Political and Social Research (**ICPSR**) recently began making all its machine-readable documentation freely accessible on the Internet.

At the end of the project in the fall of 1997, we developed a collection of information, including sample programs and documents, relevant to the project and made it available at the Statlab Web page of the Yale Web site. The collection of information has since moved to the Council on Library and Information Resources' Web site at http://www.clir.org/pubs/reports/pub83/statlab. Included in the materials accessible at this site are:

- a link to the Interim Report to the Commission on Preservation and Access
- programs to create **spread ASCII** data files
- spread ASCII data file example
- sample data map for spread ASCII data file
- **SAS** programs for recoding data and producing ASCII data from SAS data files
- link for downloading Adobe Acrobat Reader
- multiple examples of Adobe PDF files

The Roper Collection at Yale

The Yale Roper Collection contains materials from the Roper Center for Public Opinion Research (the Roper Center), whose data sets comprise a rich resource for research in political psychology and sociology. They provide a record of public opinion research in the United States from 1935 to the present, along with surveys conducted abroad since the 1940s. In addition to the data files, the Yale Roper Collection includes paper records such as **questionnaires**, information on sample sizes, and other notes necessary for use of the data files. Many of the paper records are brittle, have handwritten notes, and were produced through unstable copying technologies such as mimeography.

The first step in the project was to select a representative group of documents and accompanying data files from the collection. Our initial discussions led us to select the Roper Reports, a significant, heavily used part of the Yale Roper Collection. The Roper Reports have been produced since 1973 by the Roper Organization, a commercial polling company now known as Roper Starch Worldwide, Inc. The Roper Reports have 1,500-2,000 **respondents**, 200-300 **variables**, and polling for the reports is conducted 10 times per year in the United States. Data files contain demographic information such as age, sex, race, economic level, education, marital status, union

membership, religious and political affiliation, and responses to questions on a broad array of issues facing society such as energy, politics, media, health and medical care, consumer behavior, education, and foreign policy.

The Roper Reports in the Yale Roper Collection do not have machine-readable documentation supplied with the data files. The documentation consists of paper photocopies of questionnaires and computer output. Some parts of the documentation are poorly duplicated copies with blurred text on a gray background and some questionnaires have handwritten notes in the margins. Most of the questionnaires are printed in multiple columns on a page with no standard format or layout. The Roper Reports documentation collection thus represents the problems inherent in the rest of the Yale Roper Collection. Of the 200 Roper Reports in the Yale Collection at the time of the project, 10 studies were selected across the full span of years to include any differences in format or documentation.

Our selection of the Roper Report data files was particularly important in the context of migrating data files. The files were stored in **column binary** format with portions of the files coded in an archaic format based upon the IBM **punch card**. The responses of a single **case** or individual interview were represented on one or more punch cards. Each punch card had 80 columns and 12 rows. The non-column binary format allowed a maximum of one character per column and a maximum of 80 variables per card. The column binary format, however, made it possible to store more than one variable in the same column. With punches allowed in each of the 12 rows, the maximum number of items was increased by up to a factor of 12. This column binary format was especially popular in the 1960s and 1970s when information was stored almost exclusively on computer cards, making it desirable to compress the data into as small a space as possible, because it provided space for multiple answers to a single question.

Special instructions must be given in software programs to define this unique column and row structure. Since the format is based upon old technology, knowledge about its use and software input formats to read it are increasingly rare. Our challenge was to find a new format that preserved the full intellectual content of the binary coding while allowing current and future technology to read the data and convert the computer card punches into meaningful **values**.

Literature Search

We reviewed the library literature for this project and conducted searches of the Internet. Discussion of the issues involved in archiving digital information had been well detailed in *Preserving Digital Information*, so we limited our search to topics specific to the preservation of social science numeric data and documentation. The literature search revealed much information on imaging as a preservation technique for books but little on preserving documentation for data files (see Reference List). We uncovered no previously pub-

lished material on methods of preservation of electronic materials, other than duplicate copies moved from one storage medium to another. We found little information on the subject of copying data files and changing the way they are coded. We searched for reports on the conversion of multiple-punched data to other formats but found nothing. Nor did we find any discussion of standards for such conversions nor of the validity of various numeric data storage formats as archival media.

In addition, we inquired of the Center for Electronic Records of the National Archives and Records Administration (NARA), Archival Research and Evaluation Staff, to identify any standards they follow internally. NARA retains numeric data in the format they are received but will transform them on request (Adams 1996). There had been discussion among members of the data archive community about whether column binary was an acceptable archival format, but we found no published discussion of this issue. We also searched for reports on the use of proprietary formats in archiving electronic records. Again, we found almost no mention of numeric data in the published literature.

On the Web site of the ICPSR, we found one discussion of the conversion of questionnaire-type information from paper to electronic formats using OCR as opposed to imaging. This type of information may also be found in the business records management literature, which we did not review. JSTOR, the Journal Storage Project, was making images of journal pages available to subscribers via the Web and using OCR to index the pages (JSTOR 1996). This approach seemed to overcome the limitations of using either imaging or OCR technology alone.

We concluded that we needed to extrapolate from the more general literature on archiving textual data, which emphasized the desirability of storing information in formats independent of hardware and software (NARA 1990). The perils of using formats that depend on hardware and software in the case of textual data had been described by Jeff Rothenberg (1995). We had no reason to expect that numeric data would be any different.

During 1995 and 1996, we followed discussions on the informal list for ICPSR Official Representatives and the listserv for members of the International Association for Social Science Information Service and Technology, especially on the use of PDF for storage and distribution of **codebooks.** The discussions focused on the concern that PDF was not an acceptable archival format and would require reformatting during the lifetime of the documents. ICPSR also published a discussion of this issue (1996).

The Data Track

The preservation project's goals for the data track were to develop and evaluate a *process of migrating* digital numeric information from computer tape to hardware- and software-independent formats and to evaluate the utility of the resultant *formats*. The process of migrating was broken down into a series of nine steps.

1. Identify equipment
2. Copy files from mainframe-based media to local hard disks
3. Examine the documentation
4. Define the column binary format
5. Develop standard variable-naming classifications
6. Read in the data files with SAS and **SPSS**
7. Identify migration formats
8. **Recode** data files with SAS
9. Create spread ASCII data files without recoding

1. Identify Equipment

The computing environment used for the project consisted of IBM mainframes (all commands were submitted as JCL batch jobs issued from a CMS-based IBM mainframe to an MVS mainframe with tape access), and PC/Intel (Pentium 90) computers running Microsoft Windows for Workgroups on a Novell network.

2. Copy Files

We copied the column binary data files from old round reel tapes to new 3480 IBM cartridges on the Yale mainframe. This was the first step in refreshing the data from the old tape medium to a more stable magnetic medium. Next, the data were copied from the cartridges to mainframe disk and moved via standard file transfer protocols (in binary mode) to the Statlab Novell server, the home for the SAS program writing, record keeping, and output storage.

Once transferred to disks connected to the server, each data set was checked with a simple program that read in one variable—the **deck** or **card** number—for each observation in the original data file. The number of observations for each deck number was then compared with information in the documentation. Any discrepancies were noted and data files reordered from the Roper Center if errors were confirmed.

3. Examine Documentation

The primary document for a particular Roper Report was simply a copy of the original survey questionnaire containing the questions asked in the survey. The card number and column number (or numbers) for locating the question in the record for each respondent were given for each question. Furthermore, the punch location was listed for each response option in the question. All of this information—card number, column location, and punch location—was necessary for reading the original column binary data.

The questionnaires varied in length and, in some cases, multiple versions of a questionnaire were provided. For example, some of the surveys used a **split sample**, either to ask different questions or to ask the same questions in different sequence. Often, the questions asked of the two groups in the split sample were not identical. In some cases, the column location and format of the different variables were identical. The samples could differ in the order of items in a multiple-response question, or in using slightly different sets of items. In other cases, the format of the questions (along with their column and row locations) changed radically between samples.

The **xray**, a special form of printed output supplied by the Roper Center, provided a response frequency used to check the data during the migration processes. Organized by card, column, and row, the xray gave the total number of punched bits across observations for each variable in the data set. The xray also provided useful information about the types of questions being answered and the complexity of reading the column binary format. Because each question was usually encoded in its own column, the sum of all of the responses (plus any blank observations), for most types of questions, should add up to the total number of people in the survey. If it did not, then the question allowed multiple responses and would need to be read in a different manner from single response variables.

4. Define Format

The structure of the column binary format is illustrated in table 1. The sample question shown in the table uses two columns (47 and 48) to cover all the possible answers. The full text of this question may also be seen in Appendix 1. Each card contains 12 rows numbered from top to bottom, beginning with 1 at the top. Punch numbers are assigned in a different way: a 12 punch goes in the top row, an 11 punch in the second row, and the third through twelfth rows are reserved for 0 through 9 punches. The 11 and 12 punches are often used for "don't know," "no answer," or "not applicable" responses. In this example, a respondent chose the values 1, 2, and 4 ("living in poverty," "being abused as a child," and "drug abuse"), so punches were made in rows 4, 5, and 7, as shown in the last column of the table.

In this question, both columns 47 and 48 may have multiple punches representing the choices people were allowed to select from the list of causes of violent crime. Alternatively, in the non-column binary scheme allowing only one punch per column, each possible selection would have to be coded as a separate variable and therefore as a separate column, so coding of this question would take up 18 columns rather than two.

5. Develop Standard Classifications

A standard classification of types of variables, based upon the types of questions asked in the surveys, was used in the construction of

Table 1. *Question coding example*

Roper Report 9309, Question 8W: "…which three or four things do you think are the main causes of people committing violent crimes?"

Responses	Column location	Row	Punch number	Punched
a. Living in poverty	47	4	1	X
b. Parents not teaching right from wrong		5	2	X
c. Being abused as a child		6	3	
d. Drug abuse		7	4	X
e. What people see in TV programs		8	5	
f. A lack of morals		9	6	
g. A lack of education		10	7	
h. A person not seeing any harm in it		11	8	
i. A person being irresponsible		12	9	
j. Influence of friends		3	0	
k. Alcohol abuse		2	11 (or X)	
l. What people see in movies		1	12 (or Y)	
m. The advertising and marketing of toy guns, etc.	48	4	1	
n. Guns being too easy to get		5	2	
o. Low chance of being punished		6	3	
p. Seeing pornography		7	4	
None of these		8	5	
Don't know		1	12 (or Y)	

data set translation. The classification provided a scheme for the creation of standard templates and the logic behind the variable types used in recoding the data. In addition, the classification made it easier to construct appropriate variable names and to ensure consistent naming across data sets. Each of the variable types we defined required different variable-naming, formatting, and recoding procedures.

The four basic types of variables are regular numeric variables, numeric variables with special **missing values**, multiple-response questions, and single-response questions. Each variable type was associated with a different template for code creation, a specific form of variable names, and certain projected difficulties in recoding. For instance, a multiple-response variable was read in with a simple list of **single-punch** variables, was labeled simply with letter suffixes, and was not recoded. In contrast, an aggregated single-response variable was read in with a list of single-punch **intermediate variables**, was labeled with intermediate number suffixes, and was recoded into a final variable for the entire question. Retrieving the variable type directly from the documentation provided a guide to the particular piece of program code that was necessary for inputting and recoding the variable.

Regular numeric variables. Regular numeric variables range from 0 to 9. The values are computed by filling in each digit with the numbers in the appropriate columns. For example, Roper Report 9309, Question 68Y, asked "*Do you think government lotteries produce an unwholesome gambling spirit in this country?* Yes=1, No=2"

Numeric variables with special missing values. These variables are identical to regular numeric variables, except that there can be one or two special missing values (such as "don't know"), recorded at punch 11 and/or punch 12. For example, Roper Report 9309, Question 2X, asked, "*Do you feel things in this country are generally going in the right direction today, or do you feel that things have pretty seriously gotten off on the wrong track?* Right track=1, Wrong track=2, Don't know=Y"

Multiple-response questions. Multiple-response questions generally ask respondents to choose more than one option from a list of items, as in Roper Report 9309, Question 8W illustrated in table 1. If an item was checked, it was coded as 1 in the recoded data set; if it was not checked, it was coded as 0. Therefore, for any such question, there would be multiple binary variables corresponding to all of the possible responses (including special missing values), so that each possible response became a variable in a final recoded data set.

Single-response questions. Single-response questions allow one, and only one, response per item. They frequently include a special missing value at punch number 12, with several answer options between punch numbers 1 and 9. These questions require appropriate recoding for the final variable. For example, Roper 9309, Question 7Y, asked, *"And thinking about crime in the United States, what one type of crime do you feel presents the biggest threat for you and your family today?"* An example of multiple-column storage of a single-response question appears in table 2.

Table 2. *Example of multiple-column storage of single-response question*

Response	Punch number	Column location
a. Burglary, robbery, auto theft	1	45
b. Vandalism/hooliganism	2	
c. Official corruption, government bribe-taking	3	
d. Murder	4	
e. Rape	5	
f. Assault	6	
g. Racketeering, extortion	7	
h. Drugs	8	
i. Prostitution	9	
j. Speculation and swindling	0	
k. Other	1	46
None of the above	2	
Don't know	Y	

6. Read in Data

The first step in the SAS programming process was reading in the data using one of the three SAS column binary **informats**: **PUNCH.d**, **CBw.d**, and **ROWw.d**. The informat PUNCH.d reads a specific bit (with a value of either 0 or 1) in the original data set. The **d** value indicates which row in a column of data is to be read. The informat CBw.d, on the other hand, looks to see which one of the 10 numeric bits (0 through 9) has been punched, and returns that number as a value for the column. ROWw.d begins reading at a user-specified bit, looks to see which one of a user-specified number of bits (after the first) has been punched, and returns a number equal to the *relative* location of the punched bit. For instance, a ROW5.5 informat would start reading at PUNCH.5 and continue for four more bits through PUNCH.9; if bit 8 was punched, then the ROW5.5 informat would return a 4. The PUNCH.d informat was the most appropriate for this project. For clarification of its use, refer to the sample SAS program in Appendix 2. Depending on the final form of the question, the pattern of punches in a column usually had to be logically recoded later in the program from an intermediate variable to a final variable that matched the response options in the original documentation.

The Statistical Package for the Social Sciences (SPSS) is also able to read data, including multiple-response question responses, from a column binary data file by using the keywords MODE=MULTIPUNCH on the FILE HANDLE command. An example of an SPSS program that reads a single variable from Roper Report 9309 data with SPSS is:

```
file handle test / name='h:/roper/rprr9309.bin' /recform=fixed /
        mode=multipunch/lrecl=160.
data list file=test records=9 /1 feddep 24 (A) /9.
execute.
```

In this example, a variable called FEDDEP is read in column 24. It has a possible Y coded as "don't know", requiring that SPSS read this variable as a character string (see also SPSS 1988, 84-86).

7. Identify Migration Formats

We selected the following formats to test the migration process:

- SAS **system files** of recoded column binary data, with and without intermediate variables
- SAS system files with shortened integer byte lengths
- SAS **export files** of recoded column binary data
- ASCII files produced from recoded column binary data
- ASCII files of the binary data patterns in the original file, called spread ASCII

These formats were selected on the basis of the software's ability to read the column binary format, the availability of project staff pro-

gramming expertise, the transportability of output formats, storage requirements (size of output data sets), and long-term archival implications. While both SAS and SPSS software are able to read column binary data, the staff members working on the project had more experience with SAS, so we chose to work with that statistical package. With the exception of spread ASCII data files, each format we selected for testing contained completely recoded and renamed variables. Storage requirements were compared on the different platforms for many of the different types of SAS data files, as shown in Appendix 3.

8. Recode Data Files

Create intermediate variables and code missing values. After reading in the data with the SAS informat statements, the second step in the SAS programming process was to produce a set of if-then statements for recoding the individual punch data (the intermediate variables) into final variables. Each intermediate variable needed to have a column location, row location, variable name, and SAS informat instructions specified in the input statement, as seen in the example shown in table 3. The only things that changed while inputting this variable Q14 were the suffix and punch location for each intermediate variable. This redundancy increased when there were more intermediate variables, and especially with aggregated single-response variables. The logical statements used for recoding the intermediate variables also contained many repetitions of the same if-then commands.

Since the creation of these translation programs in SAS involved a large amount of repetitive typing, we created templates for both the input and recoding statements. Templates were created for sin-

Table 3. *Example of variable location, name, and SAS informat statement*

Column location	Variable name	Row location and informat
@30	Q14A_1	PUNCH.1
@30	Q14A_2	PUNCH.2
@30	Q14A_3	PUNCH.3
@30	Q14A_Y	PUNCH.12
@31	Q14B_1	PUNCH.1
@31	Q14B_2	PUNCH.2
@31	Q14B_3	PUNCH.3
@31	Q14B_Y	PUNCH.12

gle-response variables (simple and aggregated), multiple-response variables, and numeric variables with special missing codes. Each template included generic characters for column locations, variable names, and SAS informats, as well as pre-formed logical recoding statements. Furthermore, the templates contained repeated lines of code for different numbers of punches or aggregate variables so that it would not be necessary to enter the redundant information for the different variables.

These templates considerably sped up translation code creation. Once the type of the current variable had been determined, the appropriate input and recode statements were copied from the template file, pasted into the program code, and then the key characters (such as column location and variable name) were replaced using a find/replace function in the text editor. For example, to create the code for variable Q14, a piece of the template would have been copied from the template file and pasted into the program, as illustrated in table 4. The aggregated single-response variable template includ-

Table 4. *Examples of SAS input statements and recoding statements*

```
/* Three options + NR */
(for INPUT command)
 @COL1 Q000A_1 PUNCH.1
 @COL1 Q000A_2 PUNCH.2
 @COL1 Q000A_3 PUNCH.3
 @COL1 Q000A_Y PUNCH.12
 @COL2 Q000B_1 PUNCH.1
 @COL2 Q000B_2 PUNCH.2
 @COL2 Q000B_3 PUNCH.3
 @COL2 Q000B_Y PUNCH.12

(for recoding statements)
 IF Q000A_1=1 AND Q000A_2=0 AND Q000A_3=0 AND Q000A_Y=0
THEN Q000A=1;
 IF Q000A_1=0 AND Q000A_2=1 AND Q000A_3=0 AND Q000A_Y=0
THEN Q000A=2;
 IF Q000A_1=0 AND Q000A_2=0 AND Q000A_3=1 AND Q000A_Y=0
THEN Q000A=3;
 IF Q000A_1=0 AND Q000A_2=0 AND Q000A_3=0 AND Q000A_Y=1
THEN Q000A=9;
 IF Q000B_1=1 AND Q000B_2=0 AND Q000B_3=0 AND Q000B_Y=0
THEN Q000B=1;
 IF Q000B_1=0 AND Q000B_2=1 AND Q000B_3=0 AND Q000B_Y=0
THEN Q000B=2;
 IF Q000B_1=0 AND Q000B_2=0 AND Q000B_3=1 AND Q000B_Y=0
THEN Q000B=3;
 IF Q000B_1=0 AND Q000B_2=0 AND Q000B_3=0 AND Q000B_Y=1
THEN Q000B=9;
```

ed input and recode statements for sub-questions from A through Z; only A and B are shown in table 4.

Then the string Q000 would be replaced with Q14, and COL1 and COL2 would be replaced with 30 and 31, respectively, resulting in the final code for inputting and recoding the variables Q14A and Q14B.

Create macro programs to recode data files. Given that we could define fairly well the different types of variables in a data set, and that we could create input and recode template statements for these different variable types, it was tempting to think that we would be able to write programs to automate the whole procedure. That is, we might imagine a program that could perform all of the operations described above with minimal effort from the human operators. Although such automation may be possible in the future, the irregularity of the data files presented a major obstacle. Not only would an automatic translation program have to deal with some of the complexities of, for example, split samples with different variable types in the same column, but it would also have to handle the different types of errors that occur in the original data files.

For example, one fairly straightforward solution (though quite a lot of work) would have been to create a program that received some sort of variable list from a human operator, set up logical conditions to create code around split samples, and then write out a SAS program to translate and recode the data file. More sophisticated error-checking would have been necessary, however, to handle simple events such as a variable being incorrectly marked as a single-response type in the documentation, but actually having multiple responses coded in the data. Because the recoded data files needed to be of archival quality, this error-checking would have had to be quite rigorous. Furthermore, judgment calls would have sometimes arisen when dealing with irregularities (that is, should the variable be recoded differently or should the irregularity be ignored), so that leaving the decision solely to a computer program was not advisable. In short, creating an automatic translation program to recode the data would have involved several compromises that we would not recommend.

Debug programs and check data. Several types of errors occurred in the SAS programming: typographic errors, invalid informats, and unexpected changes in variable types (where the questionnaire did not match the data). The key indicators for these problems were INVALID DATA warnings that appeared in the SAS log.

Once the programs ran without INVALID DATA messages, the accuracy of the translation needed to be checked. Complete frequencies for *all* of the variables in the final data set had to be compared to the frequencies in the xray.

For example, if variable Q9 (deck 1, column 30) had the following frequencies (in this case, PUNCH.12 was recoded to the missing value of 9):

Value	Frequency
1	290
2	1300
9	403

then the xray for deck 1, column 30 should look like this :

Punch location	12	11	0	1	2	3	4	5	6	7	8	9
Sum	403	0	0	290	1300	0	0	0	0	0	0	0

It was necessary to check all of the variables in the final data set, because one badly coded variable in the conversion job could compromise the archival integrity of the entire data set. Unfortunately, a random subset of variables may miss the errors. If there was a mismatch between the xray and the frequencies, the difference needed to be tracked down and the SAS job edited. This frequency check was also the last chance to resolve ambiguities between the type of variable listed in the documentation and that shown in the xray. One final point to note during this check was the maximum number of possible digits in a recoded variable. If the data set was to be output in ASCII format, then the number of digits in the special missing value should be equal to the number of digits in the regular values. That is, if a variable had values 1 through 3, then the special missing value should be 9; if it had values 1 through 12, then the special missing should be 99; and so on.

Save recoded data. The final step in the reading and recoding process was to write to disk the resultant SAS data sets. For recoded SAS to SAS export files, we saved the files as SAS system files and export files with all intermediate and recoded variables, and as SAS system files and export files with just the recoded variables (see Appendix 3).

For recoded SAS to ASCII, once the SAS data set had been completely debugged and double-checked, PUT statements could be written to create an ASCII version of the recoded data (see Appendix 2). The simplest way to write these PUT statements was to copy the INPUT statements from the original job (or from the recoding statements), strip off the column binary informat information, and manually type in the new column locations for each variable. These PUT statements had two advantages: unlike automatic translation programs, they created ASCII data files with minimum wasted space and the PUT statement itself could be distributed as a **data dictionary**.

For the first data set translated, an ASCII version of the recoded

SAS data set was created using an automatic translation program called DBMSCOPY, version 5.10. However, this ASCII data set was not satisfactory. Although the translation program automatically created a data dictionary (based on the variable names in the SAS data set), the ASCII data set was not compact. The translation program seemed to be incapable of concatenating one variable's width of columns against the previous variable's columns, so that there would be no wasted space within the flat data set. Instead, the program inserted multiple spaces between each variable, which allowed a differing number of characters within each variable, but also inserted approximately 10 ASCII space characters between each variable. Because of the enormously increased storage requirements this insertion causes, this approach to creating ASCII data files was abandoned and program code for creating compact ASCII data files was written in SAS.

9. Create Spread ASCII Data Files

The original column binary data files did not necessarily have to be recoded during the translation process. Another possible method of translation was simply to convert, or spread, the column binary into ASCII data. Spread ASCII data files keep the binary structure of the column binary data, but encode each 0 or 1 as an ASCII character. For example, the following column in the original data (each 0 or 1 is one bit)

0
0
0
0
1
0
0
0
0
0
0
0

would become 000010000000 in the spread version. While this example uses a single-response variable, multiple-response questions may have more than one bit with a value of 1 in the column. To create a spread version of the data, each bit (or punch) in the column binary file could be read (via SAS in our case) as an individual variable; each variable was then written to a new ASCII data set. Because the punches in the column binary data were rectangular, and because the variables being written out did not have to be meaningfully recoded, the SAS code itself was largely a simple iteration, over columns and rows, of INPUT and PUT statements.

The iterative nature of the SAS program suggested that a macro program could be written to automate the creation of SAS code. A simple **C** program was written based on this iteration over columns

and rows. After prompting the user for unique information—the name of the SAS program file to be created, the name of the data set to be created, the name and path of the column binary file, and the number of decks and record length (LRECL) of the original data—the C program simply looped over columns and rows. The program passed through the loop twice. In the first pass, the program created an INPUT statement that would read in each punch (looping over columns and rows) as a new variable. The second pass created a PUT statement in which each new variable would be written to an ASCII file. The C program was not itself reading or writing the data files; instead, it created many repetitive lines of SAS code to read and write the data files (see Appendix 4).

Each card of column binary data was translated to a line of ASCII data. Since each card contained 960 data points (80 columns by 12 rows), each line of ASCII data contained 960 characters. The spread ASCII data set expanded about 600 percent from the original size. It took about 15 minutes to create the spread ASCII data on the IBM PC network, including all steps from running the C program to writing out the ASCII data.

We investigated the formats currently distributed by the Roper Center and the Institute for Research in the Social Sciences at the University of North Carolina at Chapel Hill (IRSS) to determine their utility over time, and to evaluate them in light of our findings. Both distributors were producing what we call the hybrid spread ASCII data files from the original column binary files. These files are produced by *horizontally concatenating* the data into a new format. The first 80 columns of the data set contained the 80 columns of the original file; however, any binary encoding of variables with multiple-punch coding was converted to ASCII. The entire file was then converted to spread ASCII and the resultant data were appended to the end of the record. Users could access the non-binary data in the original column locations; if they needed to access data that were originally punched in the binary mode, they could read those data from the spread ASCII portion of the record. Data dictionaries were produced that mapped the location of variables from both the original 80 columns and the spread ASCII portion of the record.

We obtained sample programs from IRSS to help us evaluate these hybrid spread file formats. We also acquired sample data files from IRSS and the Roper Center, with their enhanced data dictionaries, to evaluate their utility. We then adapted a SAS program from IRSS to produce the data map showing the new column locations of the data items. This data map allowed for quick translation from the column-by-row information necessary to read column binary data, to the column-only information necessary to read the spread ASCII data. A note at the top of the data map showed the order of punches in the spread ASCII data. For example, for a response located at column 56, row 1 in the original data set, the data map showed the same response's location as column 661 in the ASCII data set; a response at column 56, row 5 would be located at column 665; and a response at column 56, row Y(12) would be located at column 672 (see Appendix 5).

The Documentation Track

Several options were available for digitizing the documentation, including image scanning, OCR, combinations of image and text (PDF format), and text encoding. The initial part of the project included identifying formats to test and set up the scanning workstation. Complete documentation for each file was scanned, including the questionnaires, xrays, frequency counts, data maps, and other metadata whenever available.

Software and Equipment

The scanner used was a Hewlett-Packard ScanJet 4c with a document feeder and SCSI connection to an IBM 750-P90 PC. All files were stored on an external SCSI Iomega Jaz drive. For a major scanning project, a fast machine with 32 MB of memory was required. Also, a PCI bus SCSI card speeded up transfer rates from the scanner to the computer. An automatic document feeder reduced the labor by automating the page turning. Such labor-saving devices are cost-effective because scanning operations tend to absorb a lot of resources and to constrain work on other major tasks while the scanning is in progress.

Scanners differ in speed, and, for a given scanner, speed varies with the desired scanning resolution. Speed could be an important factor in making a purchasing decision for a major project, as it can have a considerable impact on labor costs. For the HP 4c, the time it takes to scan a page varied with the desired resolution as indicated below.

Desired Resolution	Scanner Speed
600 dpi	30 seconds
300 dpi	7.5 seconds
200 dpi	3.3 seconds

TextBridge Pro Optical Character Recognition

We first scanned the documentation for one Roper Report using the OCR software TextBridge Pro. We reviewed alternative OCR software products and, finding no significant benefits to using one over another, chose a package with which the staff had experience. Initial evaluation of the OCR output showed that there were significant numbers of errors in the resultant ASCII text. We tested various resolutions and documented time taken for setup and for scanning, optimum settings, range of file sizes, quality of proofing summaries, and procedures to follow.

The questionnaires we scanned with the TextBridge Pro software had an unacceptable rate of character recognition, including incorrect location information necessary for manipulating the accompanying data files. Handwritten notes were completely lost and the editing costs of reviewing the output and changing all errors

would have been prohibitive. This format did not present us with an adequate archival solution to preserving the textual material, so no further documentation was scanned using this process. TextBridge Pro does not work well with poor originals, determining optimum scanning settings was very time-consuming (and sometimes impossible), compression formats did not give good results, and raw ASCII format required time-consuming reformatting (see Appendix 6 for a photocopy of a page from Roper Report 9309, Question 10, and for the TextBridge Pro sample output of the same question).

Document condition. When used with printed clean originals, OCR is very accurate even when the font size is small, and can replicate the formatting of the original document. For example, TextBridge Pro is capable of producing low-resolution images and exporting both images and text to a word processing document that retains the format of the original printed page. However, there are far more problems with this process when the quality of the original is anything less than perfect, as in our case. TextBridge has a particular problem with italics and underlining, even with good quality originals.

TextBridge Pro also does not work well with columns and has some difficulty recognizing tables and columns in originals, let alone poor photocopies. This problem gets much worse when some of the entries in some of the table cells are blank because the columns get shifted; cleaning up the resulting output files becomes a major undertaking.

Scanner settings. TextBridge Pro allows considerable control over the way in which the document is scanned in as an image. For some settings, this task can be delegated to TextBridge Pro by setting options to "automatic"; in other words, TextBridge Pro tries to figure out what works as it scans the page. But TextBridge Pro does not always make these determinations successfully. Nor are photocopies ideal for scanning, particularly if not all of the characters are completely clear and if not all of the pages are of the same lightness/darkness. Successful recognition requires changing the settings periodically to account for the varying quality of the photocopies. Two outcomes are possible if the settings are not optimal: in some cases, the program is unsuccessful in recognizing text that is legible in the original; in others, it gives the frustratingly cryptic error, "page too complex for selected mode."

We finally became convinced that there was no simple system for setting these options. In the worst case, it was a frustrating process of trial and error. Too dark a setting meant that the program tried to decipher each small dot on the page as though it were part of a character. As a result, it was not possible to use 600x600 resolution in cases where originals were speckled. Likewise, selecting the "text only" option for the original page format forced the program to try to convert everything on the page into text, including imperfections in the image or dark binding patches. On the other hand, sometimes the auto brightness setting scans were so light that no text was rec-

ognized on the page. In some cases, we spent hours trying to correct the settings manually.

Proofing text. TextBridge Pro provides an optional feature that facilitates the correction of recognition errors. When text is scanned, it is possible to save "proofing text." This information is used by special modules that are installed into WordPerfect or Word and are implemented as macros. If the relevant module is installed and the document opened in the word processor, all words are highlighted that TextBridge Pro was unsure it recognized. The color of the highlighted word indicates the confidence TextBridge Pro has in its accuracy.

TextBridge Pro can be taught to recognize particular fonts with a higher degree of accuracy through a period of training during which it asks the user to help it recognize ambiguous characters. We did not use this feature extensively. Our limited experience showed it does increase the likelihood of successful recognition if originals were poor but not truly awful. The effort is only worthwhile if the same types of fonts will be encountered often, which was not the case with the Roper Reports documentation.

Final format. The available formats into which the output text can be saved depends on the options selected. There are a very large number of possible word processor, text, and spreadsheet formats. However, if "save proofing text" is selected, then the file can be saved in only Word or WordPerfect format. Similarly, if "reconstitute text" is selected, only file formats that support fairly complex formatting are available. When formatting text with the "reconstitute text" option, TextBridge Pro will use some of the new "style" features of WordPerfect or Word in the new document. This can make subsequent editing cumbersome. Though the styles themselves can be edited, an alternative is to save in a file format that supports the text features that are being "reconstituted" but does not support "styles." In this way, the formatting will appear as regular tabs, font changes, and so forth, that can be directly edited. The editing of ASCII text to recreate the format of the original is a major undertaking and the time required to reformat each document is extensive. We found that getting pagination to match the original is particularly difficult.

The scanned images produced by TextBridge Pro can be stored in CCITT-3, a compression standard, for later processing, but the results from the subsequent processing of these images were not as good as those obtained from processing the images directly from the scanner. We decided that using these compression formats would not give usable results.

PDF Files from Adobe Capture

The next step in the documentation portion of the project was to produce documents in the portable document format (PDF) used by Adobe Acrobat, a widely accepted de facto standard for encoding electronic documents. The viewing software provided by Adobe allows for reading and searching the text, viewing the structure of a document, and printing high-quality hard copy. PDF documents pro-

vided clear, accurate reproductions of the questionnaires. The Adobe Capture software produced an interim ASCII text file that could be edited to improve text searching. An example of a viewing screen may be found at the end of Appendix 6.

Basic structure of Acrobat files. Adobe Acrobat files (distinguished by the PDF suffix on the file names) can contain both text and image information from the original document. There are different types of PDF files containing different kinds of information: normal PDF, image-only PDF, and image+text PDF.

Normal PDF files, by default, display the text information derived from the OCR process. Where the text information is unknown (when there is a nontext picture on the page or there were difficulties in the OCR process), a normal PDF file will insert the original image into that space on the display page. Image-only PDF files are, in effect, paginated pictures of the original pages. Like **tagged image file format** (**TIFF**) files, the text in these images is not searchable. Like image-only files, image+text PDF files show the image of the original pages, but also contain searchable text.

The image+text files were chosen as the most appropriate for this project. The user would see a faithful reproduction of the original documentation (complete with handwritten notes) with the PDF browser, but could also search for specific text within the document. If text in the search function looked suspicious, a user could view the original image. In comparison, files produced by OCR programs contain only the text information, with no way to double-check the text against the image of the original.

Adobe Acrobat Capture procedures. When scanning the document with the Capture software, a set of pages was scanned in sequence. Each page was stored as a separate TIFF file, with the filenames numbered sequentially. For example, a 40-page document produced 40 TIFF files, named page01, page02, through page40. These original image files in TIFF format were stored separately from the final reformatted PDF documents, providing a set of image files for digital storage.

When translating the images into editable text, the TIFF files were concatenated into a single document during the OCR scanning process. Acrobat Capture analyzed the page layout and grouped text into regions. It then identified characters and grouped them into words. The words were looked up in the Acrobat dictionary (which can be customized) and spelling suspects noted. Fonts were analyzed and font suspects identified. The interim text layer of the final PDF file contains no image data and can be edited with the Capture Reviewer so that the text matches the original document as closely as possible. During the OCR process, each word was assigned a confidence rating, representing the software's estimate of its OCR accuracy.

For the searchable text of the final PDF file to be as accurate as possible, many OCR errors were corrected by editing the interim files. Fortunately, the program used to edit interim files, Acrobat Capture Reviewer, would highlight words whose confidence levels fell below a certain threshold, or that were not included in a diction-

ary file. The majority of unrecognized words could be easily spotted and corrected to match the original document. Although the Reviewer software allows one to change fonts and other formatting options, the only editing necessary for the project was in the content of the words used for searching purposes (words used to locate terms in question text and variable coding). Since the user would see only the image reproduction of the original, the underlying ASCII text need to be reformatted as a visual reproduction of the original. Once the document was edited, it was then saved in the image+text PDF format.

Time and storage requirements of PDF files. The total time required to process a 39-page document was approximately four hours, from scanning to saving the final PDF. The scanning itself took 30 minutes; the OCR process took 20 minutes; and editing the resulting file took 3 hours 15 minutes. The storage space required for this document is shown in table 5.

Table 5. *Example of storage space required for Acrobat PDF files*

File Type	Storage Space
39 TIFF files	1.19 MB
collated image file	1.42 MB
final PDF files: normal PDF (image) image+text PDF	 924 KB 1.49 MB
ASCII output file	100 KB

Documentation for the other nine Roper Report data files was also scanned and edited. Some data files with split samples had dual documentation. The time taken for the scanning process using a document feeder ranged from 5 to 30 minutes. The OCR software took between 15 and 35 minutes to process each document. The time taken to edit each document varied widely, from one to eight-and-a-half hours. The time it took to complete a single document depended largely on the quality of the original. Features such as background shadows, crooked lines of text, compressed fonts, jagged edges on letters, and handwriting increased both the time it took the OCR software to process the page, and, more importantly, the time it took to edit or insert accurate, searchable text. In some cases, blocks of text were so unrecognizable that whole questions needed to be typed in as hidden search text. Additional time was required for error-checking.

Problems encountered in text recognition and editing of PDF files. Although Adobe Acrobat Preview highlights most words with low confidence levels, some forms of errors are not so easily detected during the editing phase.

- A word was not recognized as a block of text by Capture software during the OCR process. This means that instead of a text form of the word, the document included simply a bitmapped image of the word from the original page. Such an image, of course, would not be searchable. Fortunately, these image blocks had some telltale signs. First, they appeared to be in a font that was usually quite different from the fonts assigned to the recognized words. Also, they were usually surrounded by a fine gray or blue box. After some experience, an editor could quickly spot the image boxes as the page was being read. Although these image boxes could not be changed to text, a Reviewer command can be used to insert hidden search text underneath them. A later search would highlight the image as if it were a normal word.

- A word was recognized as another, *valid* English word. In this case, the confidence level for the word might be quite high and the word would not normally be highlighted. For instance, if a falsely recognized word was assigned a confidence level of 97 percent, and Preview was set to highlight words at 95 percent *or below*, then the wrong word would not be highlighted. These words can be highlighted by raising the threshold setting, although at high levels (98 percent or 99 percent), virtually every word in the document would be highlighted. The only practical way to discover these errors was for an editor to read through the document carefully. Once errors were spotted, the correct words would be inserted.

- In rare cases, a line of text could be skipped in the OCR process. Again, nothing would be highlighted, but fortunately there would be obvious gaps in the text block where the line was skipped. A careful reading of the document would reveal these gaps. As a corrective, a new block of text could be created to overlay the gap. The correct text could then be typed in for the purpose of future searches.

In each of these cases, an attentive editor must catch the OCR error and make appropriate changes to the text to verify accurate search and retrieval. We did not edit enough documents to estimate the average time needed for cleaning a complete document. Future projects will need to budget extensive editing costs.

HTML and SGML/XML Marked-up Files

Conversion of scanned text to **hypertext markup language** (**HTML**) format would provide a more readily accessible browsing format. However, the text of each document would need to be fully edited and formatted. As indicated above, the ASCII output from the OCR

technology we used could not provide us with text clean enough to use in HTML. Moving text into documents adhering to the **standard general markup language/extensible markup language** (**SGML/XML**) is the most labor-intensive but also the most dynamic alternative for text applications. SGML/XML tagging allows customized and robust access to specific pieces of the documentation (such as question text and variable location information). SGML/XML Document Type Definitions maintain the integrity of document content and structure and also define the relationships among the elements. The emerging social science documentation standard for both formatting and content, the **Data Documentation Initiative** (**DDI**) Document Type Definition (DTD), provides standard elements developed specifically for social science numeric resources. The standard adheres to our requirement that text be stored in a system-independent, nonproprietary format. Furthermore, this standard, developed by representatives from the international social science research community, is intended to fill the need for a structured codebook standard that will serve as an interchange format and permit the development of new Web applications. However, this format requires that the text be fully edited and the components of the documentation tagged, and funding for this work was not included in the budget for this project.

Findings and Recommendations

Upon completing the steps defined in both the data and documentation tracks, we carefully examined the processes of migrating hardware- and software-dependent formats to independent formats. We also evaluated the formats in relation to their utility and ease of use over time. In both the data migration and the documentation migration processes, it was imperative that the original content be preserved. Migration that included *recoding* the content of the data files (changing the character or numeric equivalents in a data file) proved to be labor-intensive and error-prone, and produced unacceptable changes to the original content of the data. Editing the text output from the scanning process proved to be the same: error-prone, time-consuming, and incomplete. Therefore, recoding as a part of migration is not recommended. However, simply copying a file in its original format from medium to medium (refreshing) is not enough.

Software-dependent data file formats, such as the original column binary files examined in the project, cannot be read without specific software routines. If standard software packages do not offer those specific routines in the future, translation programs that emulate the software's reading of the column binary format could provide a solution. However, these emulation programs will themselves require migration strategies over time. We offer another alternative for the column binary format: *convert* the data out of the column binary format into ASCII without changing the coded values of the files. The spread ASCII format meets the criterion of software independence while simultaneously preserving the original content of the data set. It does, however, require a file-by-file migration strategy that would be time-consuming for a large collection of files.

Finding a parallel solution for the documentation files is not possible at this time. We can not accurately generate character-by-character equivalents of the paper records. We can, however, scan the paper into digital representations that could be used in future character recognition technologies. The Adobe PDF image+text format does provide an interim solution by producing digital versions of the image and limited ASCII representation of the text. However, the types of documentation files produced by the Adobe process are software dependent. If software packages move away from the format used to store the image+text files, translators will be necessary to search, print, and display the files. We therefore recommend archiving the images of the printed pages in both nonproprietary TIFF format and PDF image+text format.

User Evaluation

We asked faculty and graduate students to make an informal review of the findings and sample output from the project. All our evaluators had previous experience using data files from the Yale Roper Collection. Regarding the data conversions, they expressed relief at not having to use column binary input statements to read the data files. They had no difficulty in using the chart that mapped column binary to ASCII in order to locate variables in the spread ASCII

version of the data files, once it was explained that each variable mapped directly to a 12-column equivalent and instructions were given for finding single- and multiple-punch locations.

As for the documentation track, faculty and student reviewers found viewing and browsing PDF format files acceptable. Since most users had accessed PDF files on the Web, they seemed comfortable moving from the Internet browser to the Adobe Reader to locate question text and variable location information. This may not be the case with inexperienced users. These users were eager to have more questionnaire texts available for browsing and searching. The lack of a large sample collection of questionnaires did not allow evaluation of question text searching on a large scale.

Findings about Data Conversion

Column binary into SAS and SPSS. As long as software packages can read the SAS and SPSS export formats, recoding the column binary format into SAS and SPSS export files is an attractive option. These file formats can be used easily, are transportable to multiple operating systems and equipment configurations, and can be transformed into other software-specific formats. They do, however, have a number of drawbacks.

First, the original data file must be recoded, a process that is lengthy and potentially error-prone and one that places great reliance on the person doing the translation. If that person does not adequately check for errors, annotate the documentation for irregular variables, or properly recode the original patterns of punches, the translated data set becomes inconsistent with the original. Also, some irregularities in the original data set, which may be meaningful in the analysis of the data, can become lost when the data set is cleaned up. For instance, the original documentation might indicate that a question has four possible responses: PUNCH.1 through PUNCH.3 for a rating scale, and PUNCH.12 for a "don't know" response. On examination of the xray, though, it is discovered that about 200 people had their responses coded as PUNCH.11, not PUNCH.12. A decision must be made: will those PUNCH.11 observations be given the same special missing value code as that for PUNCH.12? This solution will put the responses in the data set into accordance with the documentation by assuming that the strange punches were simply due to errors in data entry. On the other hand, the strange punches could have been intentionally entered to mark out those observations for special reasons that are not listed in the documentation. In this case, it would be better to give the PUNCH.11 observations a special missing value different from that for PUNCH.12. Once the recoding is done, future researchers will be unable to re-create the original data set with its irregularities.

Second, this process of reading, recoding, and cleaning data files to produce SAS (or SPSS) system files and export files is very time-consuming. For example, it took 20 hours of work to write, debug, and double-check a SAS program to recode the data set for Roper

Report 9209, which includes a split sample and a variety of variable types. Assuming a wage of $15 an hour for an experienced SAS programmer, we would expect a cost of approximately $600 *per data set* for a complete job of data recoding. This estimate, of course, does not include the cost of consultations with data archivists about the recoding of particular variables or the cost of rewriting documentation to reflect the new format of the data set.

Third, data files stored in SAS and SPSS (and other statistical software) formats require proprietary software to read the information. Although there has been an increase in programs that can read and transfer data files from one program, and one version of a program, to another, there is no guarantee that programs for specific versions of software will be available in the future. U.S. Census data from the 1970s were produced in compressed format (called Dualabs) that relied on custom programs and can no longer be read on most of today's computing platforms. Such system- and software-dependent formats require expensive migration strategies to move them to future computing technologies.

Spread ASCII format. If an archival standard is defined as a non-column binary, nonproprietary format that faithfully reproduces the content of the original files, only the spread ASCII format meets these conditions. This spread format, however, is at least 600 percent larger than the original file and requires converting the original column binary structure. It also requires producing additional documentation, since each punch listed in the original documentation must be assigned a new column location in the spread ASCII data. We recommend that each file be converted to a standard spread ASCII format so that a single conversion map may be used for all the data files (see Appendix 5 for an example from this project). Producing such an ASCII data set has several advantages. Information is not lost from the original data set, because the pattern of 0s and 1s remains conceptually the same across data files. It is unnecessary for a data translator to interpose herself between the original data and the final user.

However, the spread ASCII format is not perfect. The storage requirements for spread ASCII data are on a par with the size requirements for a SAS data set containing both recoded *and* intermediate variables. For the overhead in storage cost, the SAS data set at least provides internally referenced variable names and meaningful variable values. The size of the spread ASCII data becomes even more apparent when it is contrasted with the size of a recoded ASCII data set. In a recoded data set, each unused bit can be left out of the final data; particular bits in a column need not even be input if they contain no values for the variable, and columns without variables may also be skipped. In contrast, in the spread ASCII data, each bit is input and translated to a character, whether it is used or not.

The spread ASCII format requires that users must know how the variables, particularly the multiple-response variables, relate to the 12-column equivalent. A spread ASCII data set is not as easily used as a fully recoded one; after all, recoding the 0s and 1s into usable

variable values will fall to the end user with ASCII data, just as it currently does with column binary data. Finally, each punch listed in the original documentation must be assigned a new column location in the spread ASCII data. Users must refer to an additional piece of documentation, the ASCII data map, to locate data of interest, and this extra step inevitably creates some initial confusion.

Hybrid spread ASCII. The hybrid spread ASCII format, distributed by the Roper Center and the Institute for Research in Social Science at University of North Carolina, offers another alternative. The original data are stored in column binary format in the first horizontal layer of the file. This format preserves the original structure of the data file in the first part of a new record. A second horizontal layer of converted data, in spread ASCII format, is added to the new record. The primary advantage of this hybrid spread ASCII data file format is that users can access the nonbinary portions of the original data file in their original column locations as indicated on the questionnaires. However, users have to know whether the question and variables of interest were coded in the binary format in order to determine whether to read the first part of the record or the second. The ASCII codes in the first horizontal layer are readable, but any binary coding in that first horizontal layer are not, since the binary coding is converted to ASCII to avoid problems while reading the data with statistical software. If users want to read data that were coded in binary in the original file, they can read the spread ASCII version of multiple-punched equivalents in the second horizontal layer without having to learn the column binary input statements to read the data.

We chose to produce converted ASCII files that did not have this two-part structure. To use the spread ASCII data files we constructed, users first determine the original column location of a particular variable from the questionnaire, and then use a simple data map to locate the column location in the new spread ASCII file (see Appendix 5).

Original column binary format. The column binary format itself turned out to be more attractive as a long-term archival standard than we had anticipated. It conserves space, it preserves the original coding of data and matches the column location information in the documentation, it can be transferred among computers in standard binary, and it can be read by standard statistical packages on all of the platforms we used in testing. Unfortunately, it is difficult to locate and decipher information about how to read column binary data with SAS and SPSS, as the latest manuals (for PC versions of the software) no longer contain supporting information about this format. This lack of documentation support indicates the possibility that the input formats will not be offered in subsequent software versions.

On the other hand, as long as the format exists, there seems to be some level of commitment to support it. As stated in the *SAS Language: Reference* text, "because **multipunched** decks and card-image data sets remain in existence… the SAS System provides informats for reading column-binary data." (SAS 1990, 38-9). If the column binary format is refreshed onto new media and preserved only in its

original form, we recommend that sample programs for reading the data with standard statistical packages, or a stand-alone translation program, be included in the collection of supporting files. But even with these supplemental programs, accessing and converting the files will continue to present significant challenges to researchers. Given these considerations, we do not recommend this format over the spread ASCII alternative.

Findings about Documentation Conversion

The OCR output files from TextBridge Pro did not provide us with an adequate means for archiving the textual material. The questionnaires we scanned had an unacceptable rate of character recognition, including incorrect location information necessary for manipulating the accompanying data files. Handwritten notes were completely lost. Although the format allowed for searching of a particular text that was successfully recognized, the amount of editing required to produce a legible version of the original, review the output, and correct all errors was found to be prohibitive. Subsequent viewing was poor without the formatting capabilities of proprietary word processing software.

The PDF format provided solutions to some of the documentation distribution and preservation problems we faced, but it did not meet all of our needs. For one thing, the format does not go far enough in providing internal structure for the manipulation, output, and analysis of the metadata. Like a tagged MARC record in an online public access catalog, full-text documentation for numeric data requires specific content tagging to allow search, retrieval, manipulation, and reformatting of individual sections of the information. Another drawback is that PDF files are produced and stored in a format that may be difficult to read and search in the future. The PDF format, although a published standard, depends on proprietary software that may not be available in future computing environments. (A similar problem can be seen with dBase data files that are rapidly becoming outmoded, causing major problems with large collections of CD-ROM products distributed by the U.S. government.) We see increasing numbers of PDF documents distributed on the Internet and the format will be used by ICPSR for the distribution of machine-readable documentation to its member institutions. So, given the large number of PDF files in distribution, software for conversion will most likely be developed over time. However, the current popularity of the PDF format does not guarantee that software to read it will continue to be available throughout the future of technological evolution.

The TIFF graphic image file format is useful for viewing and for distribution on the Web and as an intermediate archival format, allowing storage of files until they can be processed in the future using more advanced OCR technology. Even though this format does not allow text searching, tagging/mark up, or editing, it moves the endangered material into digital format. ICPSR has decided to retain

such digital images of its printed documentation collection for reprocessing as OCR technology evolves.

It is our recommendation that both PDF/Adobe Capture edited output and scanned image files be produced and archived. The PDF/Adobe image+text files allow searching and viewing of text in original formatting. The scanned image files can be archived for future character recognition and enhancement. We also want to emphasize the importance of the long-term development of tagged documentation using the DDI format. This is by far the most desirable format, albeit one that is difficult to produce from printed documentation, given the inadequacy of OCR technology and the costs of subsequent editing. We urge future producers and distributors of numeric data to help develop and adhere to standard system- independent documentation.

Recommendations to Data Producers

Design affects maintenance costs and long-term preservation. Producers of statistical data files need to be cognizant of preservation strategies and the importance of system-independent formats that can be migrated through generations of media and technological applications. In this project, we had significant problems with the column binary format of the data. Had long-term maintenance plans been considered, and the costs of migration been taken into consideration, the creators of the data format might not have chosen the column binary format. At the time, however, it was the most compact format to use and standard software could then be adapted to the format.

One thing is very clear: data producers would be advised and should be persuaded to take long-term maintenance and preservation considerations into account as they create data files and as they design value-added systems. Our experience shows that the most simple format is the best long-term format: the flat ASCII data file. We would urge producers to provide column-delimited ASCII files, accompanied by complete machine-readable documentation in nonproprietary and nonplatform-specific formats. Programming statements (also known as control cards) for SAS and SPSS are also highly recommended so that users can convert the raw data into system files. The control card files can be modified for later versions of statistical software and used for other programming applications and indexing.

In accordance with emerging standards for resource discovery, data files should contain a standard electronic header or be accompanied by machine-readable metadata identification information. This information should include complete citations to all the parts of a particular study (data files, documentation, control card files, and so forth) and serve as a study-level record of contents and structure.

Metadata standards. Not only must standards be considered for the structure of the numeric data files, but the metadata—information describing the data—must also conform to content and format standards for current use and for long-term preservation applica-

tions. Content standards require common elements, or a common set of "containers" that hold specific types of information. Standards for coding variables should be followed so that linkages and cross-study analysis are enhanced, and common thesauri for searching and mining data collections also need to be produced. As for format standards, metadata should be produced in system-independent formats that provide standard structures for the common elements and coding schemes. These format standards should provide consistent tagging of elements that can be mapped to resource discovery and viewing software and to statistical analysis and database systems.

For most social science data files, machine-readable documentation should be supplied in ASCII format that conforms to standard guidelines for both content and format. With some very large surveys using complex survey instruments, files in this ASCII format may be so big that complex structures in nonproprietary format need to be developed to reduce storage requirements. If paper documentation is distributed, it should be produced using high-quality duplication techniques with simple fonts, no underlining, no handwritten notations, and plenty of white space.

Of particular interest is the Data Documentation Initiative (DDI) DTD (Document Type Definition in XML), which is a developing standard for documentation describing numeric databases. It provides both content and format standards for creating digitized documentation in XML. Data producers in the United States, Canada, and Europe will be testing the DTD as part of their documentation production process. Data archives will be converting their digitized documentation into this format, and some will be scanning paper documentation and tagging the content with the DDI.

Complex statistical systems. We must also be concerned about the long-term preservation plans for complex systems. As we see more efforts to integrate data and documentation within linked systems, we see a growing tension between access and preservation. Complex database management systems such as ORACLE or SQL present us with more complex questions: What parts of the system need to be preserved to save the content of the information as well as its integrity? Will snapshots of the system provide future users with enough information to simulate access as we see it today? Is the content usable outside the context of the system? These are the database preservation challenges of the future.

Glossary

ASCII—American Standard Code for Information Interchange. A character encoding scheme used by many computers. The ASCII standard uses 7 of the 8 bits that make up a byte to define the codes for 128 characters. Example: in ASCII, the number seven is a treated as a character and is encoded as: 00010111. Because a byte can have a total of 256 possible values, there are an additional 128 possible characters that can be encoded into a byte, but there is no formal ASCII standard for those additional 128 characters. Most IBM-compatible personal computers do use an IBM extended character set that includes international characters, line and box drawing characters, Greek letters, and mathematical symbols.

C—A programming language.

CB*w.d*—Instructions that the SAS System uses to read standard numeric values from column-binary files, translating the data into standard binary format. The *w* value specifies the width of the variable, usually 8, but has a range between 1 and 32. The *d* value specifies the number of digits to the right of the decimal point in the numeric value.

card—Also known as deck, a physical record of data. A survey may have multiple cards for each respondent, all cards together comprising a **logical record**. Based on the IBM punch cards of 80-column length.

case—The unit of analysis in a particular data file. Can be an individual respondent to a questionnaire, a customer, or an industry. In the Roper Reports, each case is an interview respondent.

codebook—Description of the organization and content of a data file. Contains the code ranges and the code meanings needed to interpret the data file.

column binary—A code originally used with punched cards in which successive bits are represented by the presence or absence of punches in contiguous positions in columns. Using this method, responses to more than one question can be stored in a single column.

data dictionary—A file, part of a file, or part of a printed codebook containing information about a data file, including the name of the element, its format, location, and size.

Data Documentation Initiative (DDI)—An international committee sponsored by ICPSR that is developing a new metadata standard for social science documentation. This standard, developed by representatives from the international social science research community, is intended to fill the need for a structured codebook standard that will serve as an interchange format and permit the development of new Web applications. The Document Type Definition (DTD) for the DDI

standard is written in XML (Extensible Markup Language) and is available at http://www.icpsr.umich.edu/DDI/.

deck—Also known as card, a logical record of data. A survey may have multiple cards for each respondent, all cards together comprising a logical record. Based on the IBM punch cards of 80-column length.

documentation—Information that accompanies a data file, describing the condition of the data, the creation of the file, the location and size of variables in the file, and the values (or codes) of the variables.

export file—A file produced by a software package that is designed to be read on another computer, often with a different operating system, running a version of the same software package.

HTML—HyperText Markup Language

ICPSR—Inter-university Consortium for Political and Social Research

informat—The instructions that specify how SAS reads the numbers and characters in a data file.

intermediate variable—A variable used when recoding data to input information from individual punches in multipunch data. Sets of intermediate variables are then recoded to produce final variables.

logical record—A complete unit of data for a particular unit of analysis, in this project a single respondent. Multiple physical records, called cards or decks, may make up a logical record.

missing value—A value code that indicates no data are present for a variable for a particular case. To be distinguished from non-response values (respondent refused to answer or was not asked the question) and from invalid responses (the response did not have a valid value code equivalent). Non-responses and invalid responses may or may not have value categories provided in the questionnaire and may be treated differently from true missing data during analysis.

multipunched—A way of recording data, originally used with punched cards, in which successive bits are represented by the presence or absence of punches in contiguous positions in columns. Using this method, responses to more than one question can be stored in a single column.

OCR—Optical character recognition

PDF—Portable Document Format, a published standard format developed by Adobe Systems, accessed with proprietary software.

punch card—A paper medium used for recording computer-readable data. The card is punched by a special machine called a keypunch that works like a typewriter, except that it punches holes in cards instead of typing characters on paper. The punch cards are then processed with a card reader that transfers the punched information to a computer-readable digital format.

PUNCH.*d*—Instructions that the SAS system uses to read standard numeric values from column binary files. The *d* value specifies which row in a card column to read. Valid values for the *d* value are 1 through 12.

questionnaire—The set of questions asked in a survey. In the Yale Roper Collection, the questionnaire, with columns and codes written next to the question, may substitute for a codebook.

recode—Changing the value code of a variable from one value to another. For example, changing 0 and 1 values in column binary data files to value ranges of 0 through 12. Also known as data transformation.

respondent—In survey research, the person responding to the survey questions.

ROW*w.d*—Instructions that the SAS system uses to read a column-binary field down a card column. The *w* value specifies the row where the field begins, with a range between 1 and 12. The *d* value specifies the length in rows of the field. Valid values for *d* are 1 through 25, with the default value of 1. The informat assigns the relative position of the punch in the field range to a numeric variable.

SAS—Set of proprietary computer programs used for analysis of social science statistical data. (No longer an acronym; originally stood for Statistical Analysis System.)

SGML—Standard General Mark-Up Language

SPSS—Statistical Package for the Social Sciences. Set of proprietary computer programs used for analysis of social science statistical data.

single-punch—A single response coded in a column.

split sample—A method of data collection in which one group of respondents is queried with one form of a questionnaire and the second group is queried with a different form of the questionnaire.

spread—Recoding multiple responses that have been coded in a single column of a record to a separate column for each response.

system file—A data file or collection of data files specifically formatted for a particular software package; may not be readable by other software packages.

TIFF—Tagged Image File Format

values—The numeric or character equivalents for a particular variable in a data file.

variable—An item in a data file to which a value has been assigned. A data file contains the values of certain variables measured for a set of cases. In the Roper Report data files, variables are responses to questions or parts of questions from each person interviewed.

XML—Extensible Markup Language (XML) is a data format for structured document interchange on the Web.

xray—A form of output that is organized by card, column, and row; each bit has its own unique location within this framework. The total number of punched bits across all observations is recorded for each location in the data set. This sum often provides a response frequency for individual response options.

Sources of Glossary Terms

Armor, David J., and Arthur S. Couch. 1972. *Data-text Primer: An Introduction to Computerized Social Data Analysis*. New York: The Free Press.

Dodd, Sue A. 1982. *Cataloging Machine-readable Data Files: An interpretive Manual*. Chicago: American Library Association.

Dodd, Sue A., and Ann M. Sandberg-Fox. 1985. *Cataloging Microcomputer Files: a Manual of Interpretation for AACR2*. Chicago: American Library Association.

Geda, Carolyn L. [n.d.] *Data Preparation Manual*. Sponsored by John D. Peine, Project Coordinator, Heritage Conservation and Recreation Service, U.S. Department of the Interior.

Jacobs, Jim. *Glossary of Selected Social Science Computing Terms and Social Science Data Terms.* University of California, San Diego. Available at http://odwin.ucsd.edu/glossary/index.html.

SAS Institute. 1990. *SAS Language: Reference.* Version 6, 1st ed. Cary, NC: SAS Institute.

Sippl, Charles J. 1966. *Computer Dictionary*. Indianapolis: Howard W. Sams & Co., Inc.

Sippl, Charles J., and Roger J. Sippl. 1980. *Computer Dictionary*. 3rd ed. Indianapolis: Howard W. Sams & Co, Inc.

Spencer, Donald D. 1968. *Computer Programmer's Dictionary and Handbook*. Waltham, MA: Blaisell Publishing Company.

SPSS, Inc. 1988. *SPSS-X User's Guide*. 3rd ed. Chicago: SPSS, Inc.

Weik, Martin H. 1969. *Standard Dictionary of Computers and Information Processing*. New York: Hayden Book Company.

Reference List

Adams, Margaret. 1996. Private e-mail message to JoAnn Dionne, November 4.

Bearman, David. 1999. Reality and Chimeras in the Preservation of Electronic Records. *D-Lib Magazine* 5(4). Available at http://www.dlib.org/dlib/april99/bearman/04bearman.html.

Conway, Paul. 1996. *Conversion of Microfilm to Digital Imagery: A Demonstration Project.* Performance Report on the Production Conversion Phase of Project Open Book. New Haven, CT: Yale University Library.

Elkington, Nancy E., ed. 1994. *Digital Imaging Technology for Preservation.* Proceedings from an RLG Symposium held March 17 and 18, 1994, Cornell University, Ithaca, NY. Mountain View, CA: Research Libraries Group, Inc.

Kenney, Anne R., and Stephen Chapman. 1996. *Digital Imaging for Libraries and Archives.* Ithaca, NY: Department of Preservation and Conservation, Cornell University Library.

Inter-university Consortium for Political and Social Research. 1996. *Producing Electronic Forms of Documentation: The Experience at ICPSR.* Available at http://www.icpsr.umich.edu/ICPSR/Developments/nsf.pdf.

JSTOR. 1996. *Why Images?* Available at http://index.umdl.umich.edu/about/images.html.

Kenney, Anne R., and Stephen Chapman. 1995. *Tutorial: Digital Resolution Requirements for Replacing Text-Based Material: Methods for Benchmarking Image Quality.* Washington, DC: Commission on Preservation and Access.

Michelson, Avra, and Jeff Rothenberg. 1992. Scholarly Communication and Information Technology: Exploring the Impact of Changes in the Research Process on Archives. *American Archivist* 55:236-315.

National Archives and Records Administration, Archival Research and Evaluation Staff. 1990. *A National Archives Strategy for the Development and Implementation of Standards for the Creation, Transfer, Access, and Long-term Storage of Electronic Records of the Federal Government.* National Archives Technical Information Paper No. 8, June. Available at gopher://gopher.nara.gov:70/00/managers/archival/papers/strategy.txt.

Rockwell, Richard C. 1997. Message to the ICPSR OR-l (orl@majordomo.srv.ualbera.ca) of February 27. Subject: ORL discussion of PDF—ICPSR.

Rothenberg, Jeff. 1995. Ensuring the Longevity of Digital Documents. *Scientific American* 272(1):42-47.

Rothenberg, Jeff. 1999. *Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation.* Washington, DC: Council on Library and Information Resources (January). Available at http://www.clir.org/pubs/reports/reports.html.

SAS Institute. 1990. *SAS Language: Reference.* Version 6, 1st ed. Cary, NC: SAS Institute.

SPSS, Inc. 1988. *SPSS-X User's Guide.* 3rd ed. Chicago: SPSS, Inc.

Saffady, William. 1993. *Electronic Document Imaging Systems: Design, Evaluation, and Implementation*. Westport, CT: Meckler Publishing.

Stielow, Frederick J. 1992. Archival Theory and the Preservation of Electronic Media: Opportunities and Standards Below the Cutting Edge. *American Archivist* 55:332–43.

Task Force on Archiving of Digital Information. 1996. *Preserving Digital Information*. Report to the Commission on Preservation and Access and the Research Libraries Group. Washington, DC: Commission on Preservation and Access. Multiple versions are available at: http://www.rlg.org/ArchTF/.

Appendix 1: Roper Report documentation page 3W: Questions 7-9

photocopy

PAGE 3 W - CD-1

7. And thinking about crime in the United States, what <u>one</u> type of crime do you feel presents the biggest threat for you and your family today? (HAND RESPONDENT CARD)

a.	Burglary, robbery, auto theft 1	45/
b.	Vandalism/hooliganism 2	
c.	Official corruption, government bribe-taking 3	
d.	Murder 4	
e.	Rape 5	
f.	Assault 6	
g.	Racketeering, extortion 7	
h.	Drugs 8	
i.	Prostitution 9	
j.	Speculation and swindling 0	
k.	Other 1	46/
	None of the above (vol.) 2	
	Don't know ..... Y	

8. Of course, the problems we face in our society often have many different causes. I am going to ask you about some different problems in society and for each one I want you to tell me what <u>you</u> think are the major causes of each problem. First, violent crime. (HAND RESPONDENT CARD) Using this card, which <u>three or four</u> things do you think are the <u>main</u> causes of people committing violent crimes?

VIOLENT CRIME

a.	Living in poverty 1	47/
b.	Parents not teaching right from wrong 2	
c.	Being abused as a child 3	
d.	Drug abuse 4	
e.	What people see in TV programs 5	
f.	A lack of morals 6	
g.	A lack of education 7	
h.	A person not seeing any harm in it .. 8	
i.	A person being irresponsible 9	
j.	Influence of friends 0	
k.	Alcohol abuse X	
l.	What people see in movies ..... Y	
m.	The advertising and marketing of toy guns, etc. 1	48/
n.	Guns being too easy to get 2	
o.	Low chance of being punished 3	
p.	Seeing pornography 4	
	None of these 5	
	Don't know ..... Y	

9. What about drunk driving? (HAND RESPONDENT CARD) Using this card, which <u>three or four</u> things do you think are the <u>main</u> causes of people becoming drunk drivers?

DRUNK DRIVING

a.	Living in poverty 1	49/
b.	Parents not teaching right from wrong 2	
c.	Being abused as a child 3	
d.	Drug abuse 4	
e.	What people see in TV programs 5	
f.	A lack of morals 6	
g.	A lack of education 7	
h.	A person not seeing any harm in it .. 8	
i.	A person being irresponsible 9	
j.	Influence of friends 0	
k.	Being unable to control drinking because alcoholism is a disease X	
l.	What people see in movies ..... Y	
m.	What people see in advertising 1	50/
n.	Bartenders not being responsible 2	
o.	Low chance of being punished 3	
p.	Friends and family not looking out for each other 4	
	None of these 5	
	Don't know ..... Y	

Appendix 2: Sample SAS input and recode statements

```
/*
This snippet of a SAS program consists of two parts: the first part is an example of reading in column binary data, the second is an  example of inputting the same data in spread ASCII format
*/
...
DATA ONE;
INFILE IN LRECL=160 RECFM=F;
INPUT
#1 @26 Q10_1  PUNCH.1
  @26 Q10_2  PUNCH.2
  @26 Q10_Y  PUNCH.12
#9 @5  Q101_1 PUNCH.1
  @5  Q101_2 PUNCH.2
  @5  Q101_3 PUNCH.3
  @5  Q101_X PUNCH.11
  @5  Q101_Y PUNCH.11;

IF   Q10_1=1 AND Q10_2=0 AND Q10_Y=0 THEN Q10=1;
 IF Q10_1=0 AND Q10_2=1 AND Q10_Y=0 THEN Q10=2;
 IF Q10_1=0 AND Q10_2=0 AND Q10_Y=1 THEN Q10=99;
IF   Q101_1=1 AND Q101_2=0 AND Q101_3=0 AND Q101_X=0 AND Q101_Y=0
     THEN Q101=1;
 IF Q101_1=0 AND Q101_2=1 AND Q101_3=0 AND Q101_X=0 AND Q101_Y=0
     THEN Q101=2;
 IF Q101_1=0 AND Q101_2=0 AND Q101_3=1 AND Q101_X=0 AND Q101_Y=0
     THEN Q101=3;
 IF Q101_1=0 AND Q101_2=0 AND Q101_3=0 AND Q101_X=1 AND Q101_Y=0
     THEN Q101=4;
 IF Q101_1=0 AND Q101_2=0 AND Q101_3=0 AND Q101_X=0 AND Q101_Y=1
     THEN Q101=99;
...
DATA ONE;
INFILE IN LRECL=960 RECFM=F;
INPUT
#1 @301 Q10_1
  @302 Q10_2
  @312 Q10_Y
#9 @49  Q101_1
  @49  Q101_2
  @49  Q101_3
  @49  Q101_X
  @49  Q101_Y;

IF   Q10_1=1 AND Q10_2=0 AND Q10_Y=0 THEN Q10=1;
 IF Q10_1=0 AND Q10_2=1 AND Q10_Y=0 THEN Q10=2;
 IF Q10_1=0 AND Q10_2=0 AND Q10_Y=1 THEN Q10=99;
IF   Q101_1=1 AND Q101_2=0 AND Q101_3=0 AND Q101_X=0 AND Q101_Y=0
     THEN Q101=1;
 IF Q101_1=0 AND Q101_2=1 AND Q101_3=0 AND Q101_X=0 AND Q101_Y=0
     THEN Q101=2;
 IF Q101_1=0 AND Q101_2=0 AND Q101_3=1 AND Q101_X=0 AND Q101_Y=0
     THEN Q101=3;
 IF Q101_1=0 AND Q101_2=0 AND Q101_3=0 AND Q101_X=1 AND Q101_Y=0
     THEN Q101=4;
 IF Q101_1=0 AND Q101_2=0 AND Q101_3=0 AND Q101_X=0 AND Q101_Y=1
     THEN Q101=99;
```

Appendix 3: Data conversion formats and storage requirements

Platform	Data set Description (Sept. 1993 Roper Report)	Storage Requirements (in bytes)	Percentage of Original
All platforms	Original (column-binary)	2,858,400	
IBM PC data sets	SAS data set (full set of variables)	19,464,984	681
	SAS XPORT data set (full set of variables)	19,144,720	670
	SAS data set with 4 byte integers (full set of variables)	9,978,112	349
	SAS data set with 3 byte integers (full set of variables)	7,528,192	263
	SAS data set (partial set of variables)	8,879,360	311
	SAS XPORT data set (partial set of variables)	8,843,760	309
	SAS data set with 4 byte integers (partial set of variables)	4,571,392	160
	SAS data set with 3 byte integers (partial set of variables)	3,471,616	121
Mainframe data sets	SAS data set (full set of variables)	23,347,200	817
	SAS XPORT data set (full set of variables)	19,144,720	670
	SAS data set (partial set of variables)	11,059,200	387
	SAS XPORT data set (partial set of variables)	8,843,760	309

Appendix 4: Programs to create spread ASCII datasets

Part one: The SAS Macro

```
%INCLUDE 'j:\statlab\roper\data\spread.mac';
* ALWAYS INDICATE FULL PATH & DATASET NAME OF  *;
* COLUMN-BINARY DATA (DSN=) AND THE FULL PATH  *;
* & DATASET NAME OF THE SPREAD DATA (OUT=)     *;
* IN THE MACRO CALL                           *;
%SPREAD(DSN=h:\ssda\roper\reports\rprr9309\rprr9309.bin,
     OUT=h:\ssda\roper\reports\rprr9309\rprr9309.spr);
RUN;
```

Part two: Read in column binary data, convert to ASCII; create data map showing new column locations

```
%MACRO SPREAD(DSN=,OUT=);
OPTION ERRORS = 0; /* CHANGES INVALID DATA MESSAGES TO WARNINGS */
OPTIONS NONUMBER NODATE;
options ps=58;
DATA _NULL_;
INFILE "&DSN" lrecl=160 recfm=f;
FILE "&OUT" lrecl=960 ;
length a $1;
DO i=1 TO 80;
  INPUT @i A $CB1. @i row1  punch.1
              @i row2  punch.2
              @i row3  punch.3
              @i row4  punch.4
              @i row5  punch.5
              @i row6  punch.6
              @i row7  punch.7
              @i row8  punch.8
              @i row9  punch.9
              @i row10 punch.0
              @i row11 punch.11
              @i row12 punch.12 @;
  PUT @(1+(12*(i-1))) (row1-row12) (1.) @;
END;
put;
options ls=80;
data _null_;
file print;
title DATA MAP FOR COLUMN-BINARY SPREAD DATA;
title3 NOTE: Rows 1-9,0,X,Y correspond to columns 1-12.;
put; put; put;
do i = 1 to 40;
  col1 = 1 + (12*(i-1));
  col2 = 12 + (12*(i-1));
  col3 = 1 + (12*(40+i-1));
  col4 = 12 + (12*(40+i-1));
  cola = i;
  colb = 40 + i;
  put @1 "Column " cola 2. " maps to " col1 4. " through " col2 4. @;
  put @41 "Column " colb 2. " maps to " col3 4. " through " col4 4. @;
  put;
end;

run;

%MEND;
```

Appendix 5: Data map for column binary spread data

NOTE: Rows 1-9,0,X,Y correspond to columns 1-12.

Column 1 maps to 1 through 12
Column 2 maps to 13 through 24
Column 3 maps to 25 through 36
Column 4 maps to 37 through 48
Column 5 maps to 49 through 60
Column 6 maps to 61 through 72
Column 7 maps to 73 through 84
Column 8 maps to 85 through 96
Column 9 maps to 97 through 108
Column 10 maps to 109 through 120
Column 11 maps to 121 through 132
Column 12 maps to 133 through 144
Column 13 maps to 145 through 156
Column 14 maps to 157 through 168
Column 15 maps to 169 through 180
Column 16 maps to 181 through 192
Column 17 maps to 193 through 204
Column 18 maps to 205 through 216
Column 19 maps to 217 through 228
Column 20 maps to 229 through 240
Column 21 maps to 241 through 252
Column 22 maps to 253 through 264
Column 23 maps to 265 through 276
Column 24 maps to 277 through 288
Column 25 maps to 289 through 300
Column 26 maps to 301 through 312
Column 27 maps to 313 through 324
Column 28 maps to 325 through 336
Column 29 maps to 337 through 348
Column 30 maps to 349 through 360
Column 31 maps to 361 through 372
Column 32 maps to 373 through 384
Column 33 maps to 385 through 396
Column 34 maps to 397 through 408
Column 35 maps to 409 through 420
Column 36 maps to 421 through 432
Column 37 maps to 433 through 444
Column 38 maps to 445 through 456
Column 39 maps to 457 through 468
Column 40 maps to 469 through 480
Column 41 maps to 481 through 492
Column 42 maps to 493 through 504
Column 43 maps to 505 through 516
Column 44 maps to 517 through 528
Column 45 maps to 529 through 540
Column 46 maps to 541 through 552
Column 47 maps to 553 through 564
Column 48 maps to 565 through 576
Column 49 maps to 577 through 588
Column 50 maps to 589 through 600
Column 51 maps to 601 through 612
Column 52 maps to 613 through 624
Column 53 maps to 625 through 636
Column 54 maps to 637 through 648
Column 55 maps to 649 through 660
Column 56 maps to 661 through 672
Column 57 maps to 673 through 684
Column 58 maps to 685 through 696
Column 59 maps to 697 through 708
Column 60 maps to 709 through 720
Column 61 maps to 721 through 732
Column 62 maps to 733 through 744
Column 63 maps to 745 through 756
Column 64 maps to 757 through 768
Column 65 maps to 769 through 780
Column 66 maps to 781 through 792
Column 67 maps to 793 through 804
Column 68 maps to 805 through 816
Column 69 maps to 817 through 828
Column 70 maps to 829 through 840
Column 71 maps to 841 through 852
Column 72 maps to 853 through 864
Column 73 maps to 865 through 876
Column 74 maps to 877 through 888
Column 75 maps to 889 through 900
Column 76 maps to 901 through 912
Column 77 maps to 913 through 924
Column 78 maps to 925 through 936
Column 79 maps to 937 through 948
Column 80 maps to 949 through 960

Appendix 6: Roper Report documentation page 4 W/Y: Question 10

photocopy

PAGE 4 W/Y - CD-1

10. People have strong feelings about some issues, and not so strong about others. On this card (HAND RESPONDENT CARD) there is a scale of feelings--absolutely delighted, pleased, somewhat satisfied, no real feelings one way or the other, somewhat dissatisfied, angry, and boiling mad. Using this scale, how would you describe your feelings when you think about: (ASK ABOUT EACH)

	1. ABSOLUTELY DELIGHTED	2. PLEASED	3. SOMEWHAT SATISFIED	4. NO REAL FEELINGS	5. SOMEWHAT DISSATISFIED	6. ANGRY	7. BOILING MAD	DON'T KNOW	
a. The way things are going for you personally these days	1	2	3	4	5	6	7	Y	51/
b. The way things are going for the country generally	1	2	3	4	5	6	7	Y	52/
c. How President Clinton is dealing with the economy	1	2	3	4	5	6	7	Y	53/
d. How Congress is dealing with the economy	1	2	3	4	5	6	7	Y	54/
e. How your state government is dealing with the state's economy	1	2	3	4	5	6	7	Y	55/
f. How secure your job is	1	2	3	4	5	6	7	Y	56/
g. Your health care costs	1	2	3	4	5	6	7	Y	57/
h. The amount of your household income	1	2	3	4	5	6	7	Y	58/
i. The rate of inflation	1	2	3	4	5	6	7	Y	59/
j. The amount of taxes you pay	1	2	3	4	5	6	7	Y	60/
k. The level of interest rates	1	2	3	4	5	6	7	Y	61/
l. The cost and value of your housing	1	2	3	4	5	6	7	Y	62/
m. The level of your personal debts	1	2	3	4	5	6	7	Y	63/

79-0
80-1

Appendix 6: Roper Report documentation page 4 W/Y: Question 10

TextBridge Pro

10. People have strong feelings about some issues, and not so strong about others. on this card (HAND RESPONDENT CARD) there is a scale of feelings--absolutely delighted, pleased, somewhat satisfied, no real feelings one way or the other, somewhat dissatisfied, angry, and boiling mad. Using this scale, how would you describe your feelings when you think about: (ASK ABOUT EACH)

	Aa~ un-fY	'- PL~ASED	SATIS- FIED	REAL	DIS SATIS- FIED	-	BOILTh- Mir	~OT AL~
a. The way things are going for you personally these days	1	2	3	4	5	6	7	Y
51/								
b. The way things are going for the country generally	1	2	3	4	5	6	7	Y
52/								
C. How President Clinton is dealing with the -	1	2	3	4	5	6	7	y
53/								
d. How Congress is dealing withtheecono---	1	2	3	4	5	6	7	y
54/								
e. How your state government is dealing with the state's econo---r	1	2	3	4	5	6	7	Y
SSI								
f. Howsecureyourjobis 1		2	3	4	5	6	7	Y
56/								
g. Yourhealthcarecosts . .		.1	2	3	4	5	6	7
Y 57/								
h. Theamountofyour household inc~ne	1	2	3	4	5	6	7	Y
S8/								
i. Therateofinflation	1	2	3	4	5	6	7	Y
59/								
j. The amount of taxes youpay	1	2	3	4	S	6	7	Y
601								
k. The level of interest rates	1	2	3	4	5	6	7	Y
61/								
1. The cost and value of yourhousing	1	2	3	4	5	6	7	Y
62/								
m. Thelevelofyour personal debts	1	2	3	4	S	6	7	Y
63/								

Appendix 6: Roper Report documentation page 4 W/Y: Question 10

PDF in Acrobat Exchange