DISCLOSURE ANALYSIS

FOR THE

1992 ECONOMIC CENSUS

DISCLAIMER

The Energy Information Administration (EIA) and the Census Bureau provide this software "as is", without warranty or support of any kind, either expressed or implied, including, but not limited to, the implied fitness for a particular purpose. EIA or the Census Bureau will provide, at their discretion, only limited consultation on problems encountered with this software. Improvements or changes to this software may be made at any time without an obligation to inform users.

DISCLOSURE ANALYSIS FOR THE 1992 ECONOMIC CENSUS

Written by Robert Jewett
Economic Programming Division
Bureau of the Census

Disclosure Analysis is a fascinating topic. It begins with the simple principle that we must not directly publish data received from individuals who respond to our surveys or censuses. On the surface, this rule seems easy to follow, but it also means we cannot publish a summary table that makes it possible for someone to derive detailed information about a respondent. To make sure our tables can be published, they must first be subjected to disclosure analysis, an analytical procedure performed by a collection of computer programs of mind-boggling complexity.

This document describes the disclosure analysis work being done at the Census Bureau for the 1992 Economic Censuses. Other papers written by Census Bureau staff members have explained the linear programming and network methodology that can be used for disclosure analysis. This document discusses the techniques that will actually be used in the production work, which begins in the Fall of 1993.

The first chapter contains the basic disclosure analysis theory, and the following two chapters explain how this theory is applied to our publication tables. Many of the Economic Census tables have data for a number of geographic areas, and the complicated additive relations between the geographic areas make the disclosure analysis much more difficult to implement. In Chapter IV I describe the various ways geographic areas can be combined to equal other geographic areas, and I give several examples. If you are not familiar with the definitions of the geographic areas like MSAs and CMSAs, it should help to read Chapter IV.

The last chapter contains a short summary of our experience with disclosure analysis on two projects during 1992.

TABLE OF CONTENTS

CHAPTER I: GENERAL PRINCIPLES OF DISCLOSURE ANALYSIS

SECTION I-A: Disclosing Information

In the Economic Censuses we collect data from a large number of respondents. The data may be tabulated and published in summary tables, but we are not allowed to reveal the data for an individual respondent. To be specific, Title 13 of the United States Code states that we must not "make any publication whereby the data furnished by any particular establishment or individual under this title can be identified."

In some cases the published numbers in a summary table can be used to derive the data for an individual respondent. For example, if we published the total motor home sales for Tucker County, West Virginia, and if there were only one business establishment selling motor homes in that county, we would in effect be publishing the sales data for that establishment. It would also be a problem if only two establishments sold motor homes in the county, because each establishment could subtract his own sales data from the published county total to derive the exact sales data for the other establishment. Both of these cases are considered to be disclosures.

The problem is trickier if more than two respondents contribute to a published number. For example, assume a summary table had the following cell:

1000	=	TOT	=	Cell total
500	=	R_1	=	Value for the largest respondent
420	=	R_2	=	Value for the second largest respondent
80	=	REM	=	Total value of the other respondents in the cell. This is called the remainder of the cell.

Note that REM = TOT - R_1 - R_2

In this document, we assume that no group of respondents will combine their data in an attempt to uncover the data for another respondent.

If this cell were published, respondent R_2 could subtract his data from the cell total and determine that the data for respondent R_1 is less than or equal to 580. Since the true value of R_1 is 500, we can say that R_1 has a protection of 80, which is 16% of his value. In other words, R_1 has 16% protection from R_2.

Is this a disclosure? Does R_2 know too much about R_1? If so, this cell total should not be published.

In my opinion, the best way to answer these questions is to first clearly define what it means to disclose the data for an individual respondent. Then we could apply that definition to this particular cell and determine if the data for R_1 is being disclosed. If we had such a definition, I would call it a disclosure rule.

To my knowledge, the Census Bureau does not have a well-defined disclosure rule, but we do have a set of suppression rules to help us decide if a cell should be suppressed and to determine if the respondents are protected. In the next section I will describe some possible suppression rules and show how they can be applied.

SECTION I-B: The Primary Suppression Rules

In this section, I will describe two rules that can be used to decide if a cell in a table can be published without revealing too much about its respondents. If a cell cannot be published, it is called a primary suppression. In the tables produced by the disclosure analysis programs, the primary suppressions have a "P" beside their cell value. In the publication tables, the cell value will be replaced with a "D".

The N-K Primary Suppression Rule:

Suppress the cell if N respondents make up K% of the cell total.

The P% Primary Suppression Rule

To decide if a cell is a primary suppression, we need to define the following terms:

TOT = the cell total
R_1 = the value for the largest respondent
R_2 = the value for the second largest respondent
REM = the remainder of the cell.

REM = TOT - R_1 - R_2

Suppress the cell if REM # (R_1)(P)/100.

For example, if P = 15, we suppress the cell if REM # (R_1)(.15).

These two primary suppression rules appear similar, but they are really quite different. As shown in the example in Section I-A, the amount of protection given to the respondents in the cell depends on the size of the remainder. Since the P% rule specifies that the remainder must be greater than a fixed percentage of the largest respondent's value, it guarantees the respondent has a certain percentage of protection. If the largest respondent is protected, so are the other respondents in the cell.

On the other hand, the N-K rule compares the combined value of the first N respondents to the cell total. Therefore, the protection given to the largest respondent does not directly depend on the value of that individual respondent. We will use the P% primary suppression rule with P = 15 in the examples given in this document.

SECTION I-C: Protecting a Primary Suppression in a One-Dimensional Table

Most of the Economic Census publications include summary totals. For example, consider this example of a one-dimensional table:

Total	=	1105
Row 1	=	1000 P
Row 2	=	12
Row 3	=	17
Row 4	=	35
Row 5	=	41

Assume the respondents in Row 1 had the following data:

1000	=	TOT = Cell Total
600	=	R_1
335	=	R_2
65	=	REM

Since REM < (600)(.15) = 90, Row 1 is a primary suppression. If we only suppress Row 1 and publish the rest of the rows, any data user could calculate the data for Row 1 by subtracting the data for the other rows from the total. It is obvious that another row has to be suppressed. The other suppressed row will be a complementary suppression.

We would like to choose a row with a small value to be the complementary suppression, but we have to make sure the respondents in the primary suppression are fully protected. In this example, Row 2 is smallest, but it is not large enough to protect the respondents in Row 1 because, if Row 1 and Row 2 were combined into one cell, it would still be a primary suppression.

1012	=	TOTC = Total of Row 1 and Row 2 combined
600	=	R_1
335	=	R_2
77	=	Total Remainder = Row 2 + REM

This combined cell is a primary suppression because the total remainder is less than 15% of the value of the largest respondent.

In fact, for the combined cell to not be a primary suppression, the complementary suppressions must have a value greater than $(R_1)(.15)$ - REM. This can be seen from the following inequalities. Let CV be the value of the complementary suppressions. The combined cell will not be a primary suppression if:

Remainder of the combined cell > (Largest respondent in the combined cell)(.15)

$$\text{REM} + \text{CV} > (R_1)(.15)$$

$$\text{CV} > (R_1)(.15) - \text{REM}$$

$$\text{CV} \geq (R_1)(.15) - \text{REM} + 1$$

In the example above, $\text{CV} \geq (R_1)(.15) - \text{REM} + 1 = 90 - 65 + 1 = 26$

Row 4 has a large enough value to protect the primary suppression in Row 1. The primary suppression would also be protected if both Row 2 and Row 3 were suppressed, because their combined total is 29.

In general, if a primary suppression has values R_1 and R_2 for its largest two respondents and if the other respondents in the cell have a total value of REM (the remainder of the cell), then we need to have complementary suppressions with a total value of $(R_1)(.15)$ - REM + 1. This is the required protection for the primary suppression.

Actually we have to do more than just look at the values of the cells we are considering to be complementary suppressions. We have to calculate the capacity of each cell to protect the primary suppression, and the capacity can be less than the value of the cell. Later in the documentation I will explain our method for computing the capacity of a cell, but for now we can assume the capacity is equal to the cell value.

In summary, to protect the respondents in a primary suppression in a one-dimensional table, we have to choose one or more complementary suppressions. The total capacities of these complementary suppressions must be greater than or equal to the required protection of the primary suppression. Other primary suppressions can serve as complementary suppressions. In fact, a one-dimensional table can have two or more primary suppressions and, if they have enough capacity, they can protect each other. There may be no need to select additional complementary suppressions.

SECTION I-D: Two-Dimensional Tables

Most of the tables published for the Business and Industry Censuses appear to be only one-dimensional. For example, the Census of Retail Trade publishes retail sales data for about 190 SIC codes for different geographic areas. The data for each geographic area is shown in a separate one-dimensional table. However, since the geographic areas can be added to equal other geographic areas, the tables are really two-dimensional. The rows of the tables correspond to different retail sales SIC codes, and the columns correspond to geographic areas.

Assume we have the following table in which the first column refers to an MSA and the other columns refer to the counties in that MSA. The rows refer to a group of SIC codes that can be added to equal another higher-level SIC code.

	MSA	C_1	C_2	C_3	C_4
SIC Total	1677	1086	141	133	317
SIC_1	1056	1000 **P**	13	18	25
SIC_2	112	12	10	40	50
SIC_3	90	17	35	15	23
SIC_4	298	30	28	40	200
SIC_5	121	27	55	20	19

Assume the cell for SIC_1 and County C_1 is a primary suppression because it has the following data for its respondents

$$\begin{aligned} 1000 &= \text{TOT} \\ 600 &= R_1 \\ 332 &= R_2 \\ 68 &= \text{REM} \end{aligned}$$

Required Protection = $(R_1)(.15)$ - REM + 1 = 90 - 68 + 1 = 23

Using the same logic as in the one-dimensional table, we know we have to choose complementary suppressions within row SIC_1 and within column C_1. To protect the primary suppression within row SIC_1, we could choose the cell in column C_4 with a value of 25. Of course, another cell has to also the suppressed in column C_4, or else the value of the complementary suppression could be easily calculated.

Since the primary suppression has a required protection of 23, we know the complementary suppression in row SIC_1 must have a value at least that large. That is why we chose the cell in column C_4 with a value of 25 to be the complementary suppression. But when we have to select another cell in column C_4 to protect the complementary suppression, does that cell also need to have a value of at least 23? We have decided that it does. This decision will be discussed in greater detail in the section on upper and lower protection.

The next table shows three complementary suppressions that can be chosen to protect the primary suppression.

	MSA	C_1	C_2	C_3	C_4
SIC Total	1677	1086	141	133	317
SIC_1	1056	1000 **P**	13	18	25 **C**
SIC_2	112	12	10	40	50
SIC_3	90	17	35	15	23
SIC_4	298	30 **C**	28	40	200 **C**
SIC_5	121	27	55	20	19

The total value of the complementary suppressions is 255. These three complementary suppressions protect the primary suppression because they each have enough value and they form a closed path. To have a closed path, you must be able to draw a line from the primary suppression horizontally to reach a complementary suppression. Then, without lifting your pencil, you must be able to draw a vertical line and reach another complementary suppression. From that point, you draw a horizontal line to another complementary suppression. This process continues until you can draw a vertical line and return to the primary suppression, which completes the closed path.

This "connect the dots" procedure may seem pretty silly at first, but at the end of the section I will show an example of a table that has a disclosure because one of the suppressed cells is not protected by a closed path.

Remember how it was fun to look at a "connect the dots" puzzle and try to guess what figure would eventually be created? It is almost as much fun to look at the group of complementary suppressions chosen by the disclosure analysis program and try to guess what the closed paths are. It may be a cheap thrill, but at my age I appreciate any thrill that comes along.

I will now give two alternative solutions to protect the primary suppression in the previous table. In each of these solutions, there are more cells but less total value suppressed.

Alternative Solution 1: a more complex closed path

	MSA	C_1	C_2	C_3	C_4
SIC Total	1677	1086	141	133	317
SIC_1	1056	1000 **P**	13	18	25 **C**
SIC_2	112	12	10	40	50
SIC_3	90	17	35 **C**	15	23 **C**
SIC_4	298	30	28	40	200
SIC_5	121	27 **C**	55 **C**	20	19

In this solution, there are more complementary suppressions, but their total value is only 165, which is less than the total value suppressed in the previous solution.

Alternative Solution 2: multiple closed paths

	MSA	C_1	C_2	C_3	C_4
SIC Total	1677	1086	141	133	317
SIC_1	1056	1000 **P**	13 **C**	18 **C**	25
SIC_2	112	12 **C**	10 **C**	40	50
SIC_3	90	17 **C**	35	15 **C**	23
SIC_4	298	30	28	40	200
SIC_5	121	27	55	20	19

The cells with values of 13, 10, and 12 form a closed path, but they cannot fully protect the primary suppression because their values are too small. Since the smallest cell in the closed path has a value of 10, we say that 10 units can flow through this closed path. These three suppressed cells can provide 10 of the 23 units of protection required by the primary suppression.

The second closed path contains the cells with values 18, 15, and 17. This closed path can carry a flow of 15. In other words, the closed path has a capacity of 15. The combined capacities from these two closed paths is greater than the required protection, so the primary suppression is protected.

In this solution there are six complementary suppressions but their total value is only 85. Our goal is to suppress the least total value, so this is the best solution.

The collection of cells that protect a primary suppression is called a suppression pattern. It consists of one or more closed paths. The next example will demonstrate why a primary suppression has to be protected by a closed path of other suppressed cells. In this table, cells a thru k are suppressed.

	Total	Column 1	Column 2	Column 3	Column 4
Total	510	100	100	160	150
Row 1	155	25	a	40	b
Row 2	125	e	20	f	30
Row 3	150	30	c	k	d
Row 4	80	g	10	h	20

Since every row and column has at least two suppressed cells, there are no obvious disclosures. Cell k appears to be especially well protected because there are three suppressions in the row and column which contain that cell. However, the following algebraic equations allow us to derive the exact value for cell k.

$$\begin{array}{rrcl rcr l}
\text{Column 2} \Rightarrow & 100 & = & a + 20 + c + 10 & \Rightarrow a + c & = & 70 & (1) \\
\text{Column 4} \Rightarrow & 150 & = & b + 30 + d + 20 & \Rightarrow b + d & = & 100 & (2) \\
\text{Row 1} \Rightarrow & 155 & = & 25 + a + 40 + b & \Rightarrow a + b & = & 90 & (3)
\end{array}$$

$$\begin{array}{l r c r}
\text{Adding (1) and (2) yields} & a + b + \;\; c + d & = & 170 \\
\text{and subtracting (3)} & \underline{[\qquad\quad a + b} & \underline{=} & \underline{90} \\
\text{yields} & c + d & = & 80
\end{array}$$

Now observe:

$$\begin{array}{rrcl rcl}
\text{Row 3} \Rightarrow & 150 & = & 30 + c + k + d & \Rightarrow \; 120 & = & k + (c + d) \\
 & & & & 120 & = & k + 80 \\
 & & & & 40 & = & k
\end{array}$$

You can see in the table that no closed path starts or stops at cell k.

We did not develop this example through our great insights into disclosure analysis. In the Spring of 1991, Errol Rowe created some test data for a disclosure analysis program he was writing, and I used the data for input to my program as well. The suppressed cells were arranged in a pattern like the example above, and I was surprised when the program kept insisting that cell k was not protected. After awhile, Jim Fagan and I did a little algebra and realized we could calculate the value of cell k. It sure taught me some respect for the closed path stuff.

SECTION I-E: Converting a Table into a Network

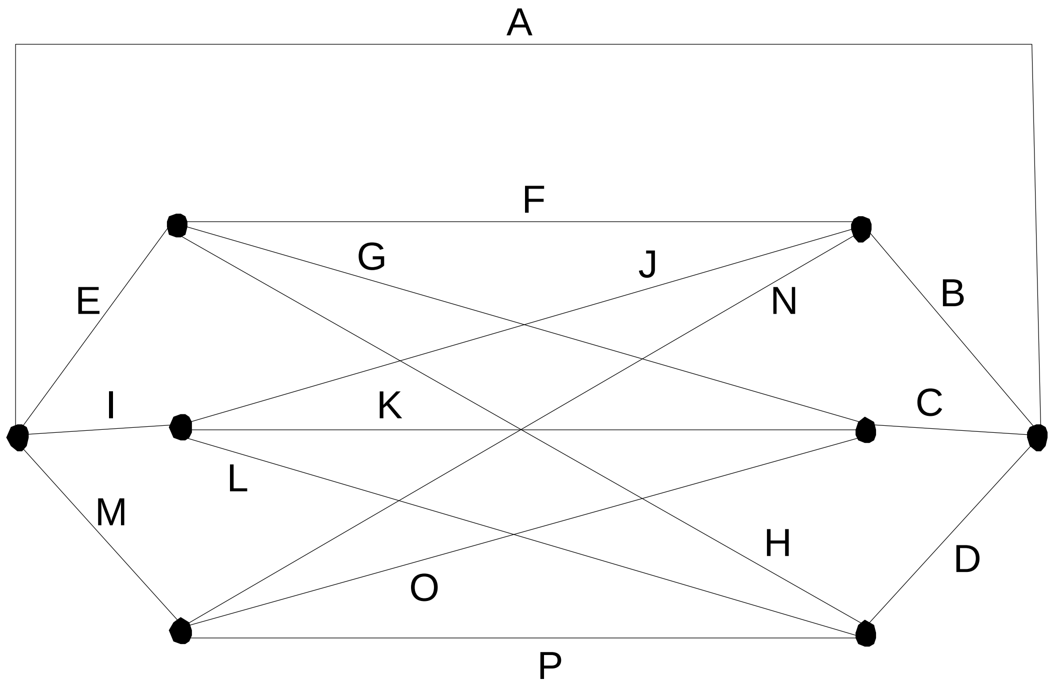
In this section I will show how a table is converted into a network, and I will explain how a set of connected arcs in the network corresponds to a closed path of suppressed cells in the table.

It is very difficult to write a computer program to find the best suppression pattern to protect a primary suppression. For the 1977 Economic Census, a program to choose the complementary suppressions was written from scratch, and the same basic procedure was used for the 1982 Economic Census. This group of disclosure analysis programs is called the INTRA system.

An entirely new method was used for the 1987 Economic Census. A two-dimensional table was converted into a network and the Minimal Cost Flow, a computer program purchased from the University of Texas, was used to select the complementary suppressions. This technique is also being used for the 1992 Economic Census.

To demonstrate how a table is equivalent to a network, consider the following table and network. Each cell in the table is identified by a letter, and the same letter is used to label the arcs in the network.

	Column Total	Col 1	Col 2	Col 3
Row Total	A	B	C	D
Row 1	E	F	G	H
Row 2	I	J	K	L
Row 3	M	N	O	P



The lines in the network are called <u>arcs</u>, and there is one arc for each cell in the table. The large dots in the network where the arcs come together are called <u>nodes</u>, and they represent the additive relations among the cells in the table. For example, one node has arcs F, J, and N coming in one side and arc B leaving on the other side. This represents the fact that cell B is equal to the sum of cells F, J, and N.

A closed path of cells in the table corresponds to a set of connected arcs in the network. In order to select complementary suppressions that protect a primary suppression in the table, we have to first identify the arc that matches the primary suppression. Then we start at one end of the arc and find a path that takes us to the node at the other end of the arc. Every arc in this path corresponds to a cell that should be suppressed in the table.

For example, assume that cell F is a primary suppression, and assume that every cell in the table has enough capacity to protect cell F. If we start at the left end of arc F, we see that arcs H, P, and N take us to the other end of arc F. In the table, the closed path of cells H, P, and N do indeed protect the primary suppression. We could have chosen a more complicated set of arcs such as G, O, P, L, and J, and the equivalent cells would also protect the primary suppression.

Converting the table into a network does not really simplify the problem of identifying a good set of complementary suppressions, but it allows us to use a program such as the Minimal Cost Flow which was designed to solve network related problems. As I will explain later, this program does not meet our needs exactly, but it sure beats having to write a new program.

In the next few paragraphs I will describe how the Minimal Cost Flow program is used to select the complementary suppressions. This general logic will become the central part of the disclosure analysis program, so you should read it carefully.

Each arc in the network is assigned a cost per unit. The program will only try to flow units through closed, connected sets of arcs. The flow has to return to the same node from which it started. When units are flowed through the arc, the resulting cost is the number of units multiplied by the cost of the arc. If these costs are added over all arcs, we have the total cost which the program is trying to minimize. In other words,

$$\text{the total cost} = \sum_{\textit{all arcs}} \text{(cost of the arc) (number of units flowing through the arc).}$$

Unless at least one arc has a negative cost, nothing happens. The minimal cost of zero would be achieved when no units flow anywhere. If the cost for an arc is negative, the program will try to flow units through it. To construct a set of connected arcs, some arcs with a positive cost will probably have to be included. But as long as the negative costs outweigh the positive costs, units will flow through the set of arcs.

Once the program finds a connected set of arcs with an overall negative cost, it will flow all of the units it can. The maximum number of units that can flow through an arc is the <u>capacity</u> of the arc, and the capacity for a connected set of arcs is limited by the arc with the least capacity.

Before the Minimal Cost Flow program can be used, a cost and capacity must be determined for each arc. For the arc that corresponds to the primary suppression, the capacity equals the required protection and the cost is a large negative number like
-100,000,000. The other arcs have a cost equal to their cell value and their capacity depends on the amount of protection they give the primary suppression. After each arc is assigned a cost and capacity, the Minimal Cost Flow program will find the connected sets of arcs that give the least total cost and it will flow as many units as possible through those arcs. If an arc carries a non-zero flow, the corresponding cell in the table should be suppressed.

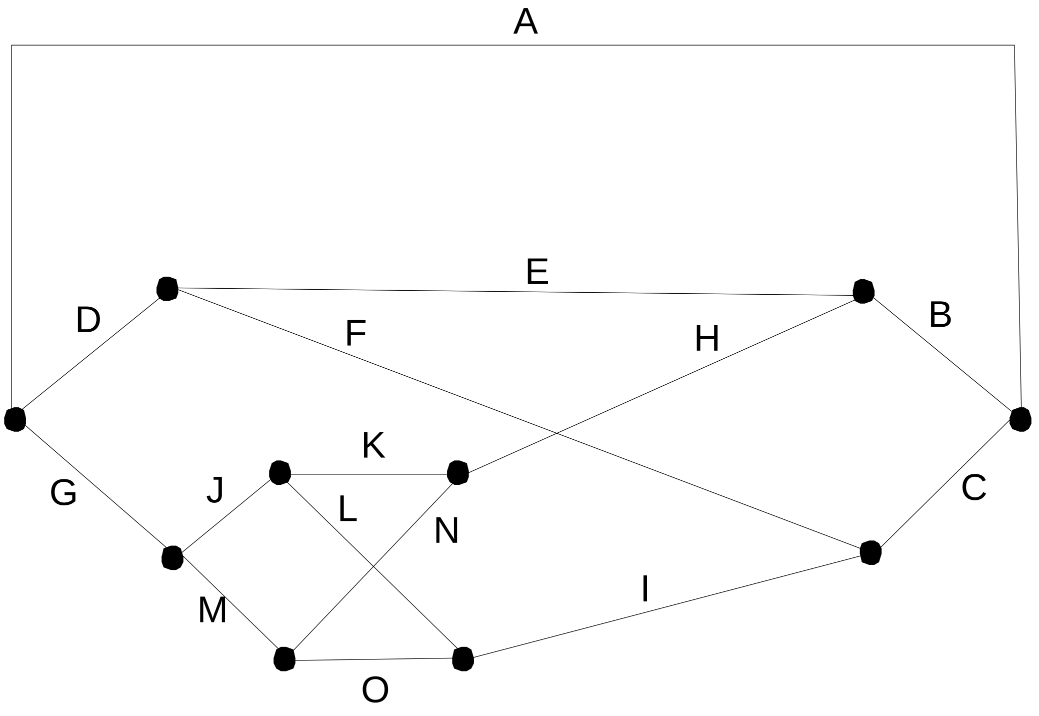
This discussion of network methodology has been oversimplified. According to network theory, an arc is able to carry a flow of units in only one direction. Each arc is like an arrow with a tail and a head, and the units can only flow from the tail to the head. In the previous examples we had the flow through the arcs going both ways. In some arcs, the units flowed from left to right, and in other arcs it went from right to left. This is because each arc in the network is really two arcs, a forward arc that allows flow from left to right and a backward arc that allows flow from right to left.

I stated earlier that each cell in a table corresponds to an arc in a network. Actually, each cell corresponds to two arcs - a forward and a backward arc. Since we want to allow a flow in both directions, the capacities of the forward and backward arc are the same. These facts are important to remember when reading the disclosure analysis program. Most of the arrays in the program use the arc number as an index and, to keep the arrays smaller, there is only one arc for each cell in the table. When the Minimal Cost Flow program is used, I have to create an extra set of arcs and define their costs and capacities.

The only table shown in this section of the documentation was a simple table where the rows and columns added to totals. When the rows of a table refer to SIC codes that have a multi-level hierarchical structure, the table is more complex and it is harder to convert into a network. Consider the following table and its accompanying network:

	Column Total	C_1	C_2
Row Total	A	B	C
R_1	D	E	F
R_2	G	H	I
R_{21}	J	K	L
R_{22}	M	N	O

Row $R_2 = R_{21} + R_{22}$



It is interesting to observe how the arcs in the network are structured. For example, if there were a primary suppression in cell E, we might want to choose cells F, I, and H to be complementary suppressions. This is not allowed because the arcs for cells I and H do not touch in the network. In other words, you can't go directly from cell I to cell H. A valid closed path might include cells E, F, I, O, N, and H because the arcs that correspond to those cells form a connected set.

At first it may seem strange that a closed path cannot go directly from cell I to cell H, but it makes sense if you think about it. If we allowed a closed path to connect these two cells, then cells E, F, I, and H would form a closed path to supposedly protect a primary suppression in cell E. However, a data user could easily add cells K and N to derive the value of cell H, and then subtract that figure from cell B to compute the value of cell E. Therefore, cell E would not be protected after all.

If both the rows and columns have a hierarchical structure, we can't convert the table into a network. Laura Zayatz and Colleen Sullivan are the champs at drawing networks for different types of tables, so direct your questions to them. If a table has a hierarchical structure in the columns, we have to divide it into subtables and process them separately.

SECTION I-F: Upper and Lower Protection

The major difference in disclosure analysis methodology between the 1987 and 1992 Economic Censuses is that we identify closed paths of cells that give full protection to a primary suppression, whereas in 1987 Bob Hemmig chose one set of closed paths to give upper protection to the primary suppression and another set of paths to give lower protection. In this section of the documentation I will discuss the idea of having separate paths for upper and lower protection. Since this technique is not used for the 1992 Economic Census, you can skip this section if you have something better to do with your life. I am describing the upper and lower protection technique because most people think it is valid, it may give better results, and we may want to use it in the future.

In section I-D, I discussed the concept of closed paths in two-dimensional tables, and I said that we "decided" each cell in the closed path should have enough capacity to protect the primary suppression. The amount of protection a closed path could offer the primary suppression was limited by the cell in the closed path with the least capacity.

There is another way to approach this problem. We can construct a closed path that gives complete upper protection to the primary suppression even though some cells in the closed path have a very low capacity. In a similar manner we can find a closed path that provides the lower protection, and then combine both closed paths to form a suppression pattern that fully protects the primary suppression.

To demonstrate how a closed path could give upper protection to the primary suppression, consider the following example:

	Total	Col 1	Col 2	Col 3	Col 4
Total	2830	110	1110	110	1500
Row 1	410	100	10	0	300
Row 2	1310	10	1000 **P**	100 **a**	200
Row 3	1110	0	100 **b**	10 **c**	1000

Assume that cell P is a primary suppression that requires a protection of 90, and assume we have chosen cells a, b, and c to be complementary suppressions. We know those three cells do not fully protect the primary suppression because cell c only has a value of 10, but how much protection do they provide? One way to determine the amount of protection is to imagine all four cells as being suppressed, and then calculate the highest and lowest values the primary suppression could possibly have.

The lowest possible value for the primary suppression is 990, as shown in the following table. For the value to go any lower, the value of c would have to be negative.

Lowest possible value for cell P:

	Total	Col 1	Col 2	Col 3	Col 4
Total	2830	110	1110	110	1500
Row 1	410	100	10	0	300
Row 2	1310	10	990 **P**	110 **a**	200
Row 3	1110	0	110 **b**	0 **c**	1000

Because the lowest value for cell P is only 10 less than the true value, we say that cell P has a lower protection of 10. The lower protection is limited by the value of cell c. In order to decrease the value for cell P, we have to add value to cells a and b, which leads to a decrease in cell c. Once cell c has reached zero, we have found the lowest possible value for cell P.

On the other hand, the highest possible value for cell P is a whopping 1100, as can be seen in this table:

Highest possible value for cell P:

	Total	Col 1	Col 2	Col 3	Col 4
Total	2830	110	1110	110	1500
Row 1	410	100	10	0	300
Row 2	1310	10	1100 **P**	0 **a**	200
Row 3	1110	0	0 **b**	110 **c**	1000

As cell P is increased, both cells a and b have to be decreased and cell c has to be increased. If we assume there is no limit how much we can increase cell c, we are only constrained by the values of cells a and b.

If cells a, b, and c were suppressed along with cell P, a data user could only determine that the primary suppression had a value less than or equal to 1100. Since this estimate of the value for cell P is 100 more than its true value, cell P has an upper protection of 100 which is more than its required protection of 90. Even though the complementary suppression in cell c only has a value of 10, the combination of cells gives full upper protection to the primary suppression.

However, cells a, b, and c do not provide enough lower protection for the primary suppression, so additional complementary suppressions would have to be chosen.

Instead of choosing those cells as complementary suppressions, what if we had chosen cells f, g, and h?

	Total	Col 1	Col 2	Col 3	Col 4
Total	2830	110	1110	110	1500
Row 1	410	100 **f**	10 **g**	0	300
Row 2	1310	10 **h**	1000 **P**	100	200
Row 3	1110	0	100	10	1000

Obviously, these cells cannot fully protect cell P because two of them only have a value of 10. To determine the amount of protection they give to cell P, we should assume all four cells are suppressed and then calculate the highest and lowest possible values for cell P, just like we did before. The highest possible value for cell P occurs when both cells g and h have values of zero.

Highest possible value for cell P:

	Total	Col 1	Col 2	Col 3	Col 4
Total	2830	110	1110	110	1500
Row 1	410	110 **f**	0 **g**	0	300
Row 2	1310	0 **h**	1010 **P**	100	200
Row 3	1110	0	100	10	1000

These three complementary suppressions do not give much upper protection to the primary suppression, but they give plenty of lower protection. The next table shows how the value of cell P could go as low as 900.

Lowest possible value for cell P:

	Total	Col 1	Col 2	Col 3	Col 4
Total	2830	110	1110	110	1500
Row 1	410	0 **f**	110 **g**	0	300
Row 2	1310	110 **h**	900 **P**	100	200
Row 3	1110	0	100	10	1000

In summary, we have identified one closed path that gives full upper protection and another closed path that gives lower protection. If we suppress all six cells from both paths, the primary suppression should be protected. The total value suppressed is 330.
To identify a single closed path where every cell has enough capacity to protect the primary suppression, we would have to select the cells shown in the following table, and there would be a good deal more value suppressed.

	Total	Col 1	Col 2	Col 3	Col 4
Total	2830	110	1110	110	1500
Row 1	410	100	10	0	300
Row 2	1310	10	1000 **P**	100	200 **C**
Row 3	1110	0	100 **C**	10	1000 **C**

This technique of finding different closed paths to give upper and lower protection sounds like a winner, but it didn't work that well in some test runs we did in the Fall of 1991. Since it checks for upper and lower protection separately, it required more computer time, but we expected that to happen. We were mainly surprised when the technique gave worse overall results than our other procedure that finds closed paths which give both upper and lower protection at the same time.

These test runs included multiple tables that had cells in common. A cell that was a complementary suppression in one table had to also be protected in the other tables. When a small cell has a lot of units added to it, the upper protection for that cell is set to a large value. In the previous example, to obtain upper protection for the primary suppression P we had to let the fictitious value of cell c go as high as 110. This gave an upper protection of 100 to cell c, and if that cell appeared in another table we would have to suppress enough additional cells to give cell C its full upper protection. When a small cell carries a lot of upper protection, strange things can happen.

This technique made the disclosure analysis more complicated, it took more CPU time, the output tables were harder to review, and it caused more value to be suppressed in several test runs. Needless to say, we decided not to use it during production.

SECTION I-G: The 1992 Disclosure Analysis Project

In early 1991, I began writing programs to do the disclosure analysis for the 1992 Business and Industry Censuses. These programs use the network procedure described in this chapter, and they were designed in a more general fashion so they could be adapted to other applications. The main purpose of this document is to describe these programs and to explain how they work.

Bob Hemmig used the network procedure when he wrote the disclosure analysis programs for the 1987 Economic Censuses, so we met with him and read his documentation to learn what he had done. I found it especially helpful to read a program Jim Fagan had written to test the network procedure on some data from the 1987 Agriculture Census. In his program it was very clear how a two-dimensional table is converted into a network, how costs and capacities are assigned to the arcs, and how the Minimal Cost Flow subroutine chooses the complementary suppressions.

With Jim's program as a guide, I wrote a new program to perform disclosure analysis on tables from the 1987 Census of Retail Trade. Duc-Mong Nguyen wrote programs to convert the 1987 Retail Trade data files into a new format. We also talked to Bill Wester about the different types of geographic areas, and Duc-Mong wrote programs to form the relations which specify how certain geographic areas can be combined to equal other geographic areas. Using the data files created by Duc-Mong, we began testing a preliminary version of the disclosure analysis program in June 1991, and it seemed to work pretty well. In the following months, I created a new version of the program to do disclosure analysis on three-dimensional tables. Both programs were refined and tested during 1992.

There are few things you should note about the 1992 disclosure analysis programs. The first thing is that all of the programs are new. Nothing has been carried over from earlier censuses. In the 1982 and 1987 disclosure analysis, some of the programs were adapted from the previous census, but in 1992 everything is new, even the record layouts for the data files.

The disclosure analysis programs were not designed in a top-down, structured manner, and I made no attempt to write them in a modular fashion. We use a collection of small programs to create the input files, but the disclosure analysis itself is done by one large program with few subroutines. I tried to make the logic clean and simple, and the programs have a great deal of internal documentation.

In my opinion, the most interesting thing about the 1992 disclosure analysis is that very little new methodology was used. For the most part, I employed the same techniques that Bob Hemmig used in 1987. In some parts of the program, like the procedure which calculates the cell capacities, I probably went into more detail than Bob did, but in other ways his program was more advanced than mine. I tried to keep the logic simple and produce clear outputs so the analysts could understand what the program was doing.

There are two main differences in the 1992 and 1987 disclosure analysis. As you know, the SIC codes have a multi-level hierarchical structure, and in 1987 Bob Hemmig had to process each additive SIC relation in a separate table. Thanks to a technique developed by SRD that allows us to convert a table with a hierarchical SIC structure into a single network, we are able to process larger tables and probably find better suppression patterns. The other difference is that we only accept closed paths that give both upper and lower protection to the primary suppressions, whereas Bob Hemmig allowed his program to identify one path for upper protection and a separate path for lower protection.

At this point you are probably wondering why we wrote an entirely new set of computer programs to implement basically the same methodology that was used in 1987. You would think we could have just modified the 1987 disclosure analysis programs and saved a lot of effort.

There are a couple reasons why we wrote new programs instead of modifying the old programs. In the first place, I didn't like the way the old programs were written, and I knew I would not enjoy modifying them. I couldn't understand them either. As I said a few paragraphs ago, I learned how to use the network procedure by reading Jim Fagan's program, not by reading the programs from the 1987 census.

In addition, I could not understand the documentation for the 1987 disclosure analysis. Even though I read it several times, I learned almost nothing from it. I mention this only because I want to stress how important it is for you to read this document thoroughly and tell me what parts you cannot understand. With your help, maybe we can produce a document that will benefit the people who write the disclosure analysis programs for the next census.

CHAPTER II: TWO-DIMENSIONAL (2-D) DISCLOSURE ANALYSIS

SECTION II-A: General Description

Most tables produced during the Economic Census have at least two dimensions. For example, the Census of Retail Trade publications have tables that give retail sales data for a number of SIC codes at various geographic levels. One table may have retail sales for 50 different SIC codes for a county, and another table may contain retail sales for 100 SIC codes for an MSA. Because there is just one column in each table, they appear to be only one dimensional. However, since there is a table for an MSA and there are tables for every county that make up that MSA, the collection of tables should be viewed as a single two dimensional table. The rows refer to SIC codes, and the columns refer to geographic areas.

To perform disclosure analysis on a 2-D table, we convert the table into a network and use the Minimal Cost Flow (MCF) program to select the complimentary suppressions. The table must have these two characteristics:

1) The first column must be a sum of the other columns. This is true when the columns refer to geographic areas. The first column may correspond to an MSA, and the other columns may refer to the counties in the MSA.

2) The rows must be related in a perfect hierarchical <u>tree structure</u>. This is hard to describe, so I will give examples of rows that satisfy this criterion and rows that fail.

Example 1: Hierarchical Rows **Columns**

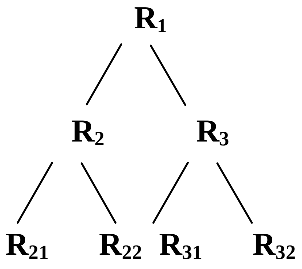
ROWS

	C_1	C_2	C_3
R_1			
R_2			
R_{21}			
R_{22}			
R_3			
R_{31}			
R_{32}			

$R_1 = R_2 + R_3$
$R_2 = R_{21} + R_{22}$
$R_3 = R_{31} + R_{32}$

These rows have a perfect hierarchical structure. There is only one way a group of rows can be summed to equal another row.

These rows are said to be related in a tree structure because they can be linked as shown in the following diagram.



Example 2: Non-Hierarchical Rows **Columns**

ROWS		C_1	C_2	C_3
	R_1			
	R_{11}			
	R_{12}			
	R_2			
	R_{21}			
	R_{22}			
	R_3			

$R_1 = R_{11} + R_{12}$
$R_1 = R_2 + R_3$
$R_2 = R_{21} + R_{22}$

This table does not have a hierarchical structure because there are two groups of rows that add to the total R_1.

In a later section of the documentation I will give more details about the way we use MCF to perform disclosure analysis on a single table. At this point, I want to describe how we process multiple tables in a single run.

In most of our Census applications, we need to perform disclosure analysis on a number of tables in a single run. For example, we may be asked to run the Retail Sales disclosure analysis for Ohio. All of these tables have the same rows based on SIC codes from the Census of Retail Trade, but they have different columns. In one table, the first column may refer to an MSA and the other columns may correspond to the counties that make up the MSA. In another table, the first column may contain county data, and the other columns may refer to the places that make up that county.

To carry out the disclosure analysis, we first create a random access (or "index") file that contains the retail sales data for Ohio. Each record on the file is identified by a unique SIC code and geographic code. For example, one record may have the data for SIC 5210 for Scioto County, the birthplace of myself and Leonard Slye (aka Roy Rogers).

In order to run the disclosure analysis for an MSA-to-county table (the first column has MSA data and the other columns have data for the counties in the MSA), we extract the records we need from the file and create the table. The MCF subroutine is used to determine the complementary suppressions. For each cell chosen to be a complementary suppression, we insert a "C" into the corresponding record on the data file.

If we later want to perform disclosure analysis on a county-to-place table (the first column has county data and the other columns contain data for the places in the county), we extract the necessary records from the input file, create the table, and again use MCF to choose the complementary suppressions. If one of the county cells was previously selected as a complementary suppression in the MSA-to-county table, that cell must also be suppressed in the county-to-place table. It is easy to identify these cells, because the matching record on the data file contains a "C", which was inserted after the cell was suppressed in the MSA-to-county table. Within the SIC row, we must make sure at least one place-level cell is suppressed to protect the county-level suppression. After all complementary suppressions have been chosen in the county-to-place table, a "C" is inserted into the corresponding records on the input file.

A more difficult problem arises if new county cells are suppressed when we process the county-to-place table. These cells must also be protected in the MSA-to-county table. To achieve this, the MSA-to-county table must be re-checked for disclosures. The records for the new county suppressions had a "C" inserted into them after the county-to-place table was processed, so the program will know these cells must be suppressed in the MSA-to-county table. If necessary, additional cells may be suppressed in the MSA-to-county table to protect the new suppressions.

This procedure to re-check tables for disclosures is called backtracking, and it cannot be avoided if we process more than one table in a single run and if the tables have cells in common.

Backtracking is very common, especially when the columns refer to geographic areas in a New England state. In that case, we have separate tables for State-to-MSA, State-to-county, MSA-to-place, and county-to-place. In the Retail Sales test runs for Maine, we had 29 tables to check for disclosures. During backtracking, there were 56 cases where a table had to be re-checked for disclosures. We have taken steps to reduce the backtracking by trying to avoid suppressing cells that are used in tables which have already been processed, but backtracking is still a major part of our disclosure analysis processing.

Protecting Cells That Appear in More Than One Table

If a suppressed cell is in two tables, it should have the same amount of protection in both tables. This is true for both primary and complementary suppressions.

When the first table is checked for disclosures, we may flow 100 units through a closed path constructed to protect a primary suppression. Each cell in the closed path is then assigned a protection of 100 units. If one of the cells in the closed path is a primary suppression that originally had a required protection of 60, its protection level is increased to 100. The existing protection for a cell is the maximum number of units that flowed through the cell during the disclosure analysis processing.

This can be a problem when a cell appears in two tables. When the second table is checked for disclosures, the required protection for the cell is equal to the existing protection carried over from the first table. If the existing protection is large, it can lead to an excessive number of suppressions in the second table.

To make things even more complicated, the cell may be used to protect other primary suppressions in the second table, and the number of units flowing through the cell may be greater than its existing protection. When this happens, the existing protection is increased. After the second table has been checked for disclosures, the first table should be re-checked to make sure we can construct a suppression pattern to give this cell its increased amount of protection.

Therefore, backtracking may be caused by increasing the protection on cells that appear in more than one table. In fact, most of the backtracking I have observed has been caused by the protection being increased for existing suppressions rather than entirely new cells being suppressed.

For example, in an MSA-to-county table we may suppress a county cell and assign it a protection of 100. When the county-to-place table is checked for disclosures, the county cell may be used to protect other primary suppressions, and may have 300 units flowed through it. This means the existing protection for this cell has been increased to 300, and we should re-check the MSA-to-county table to make sure the cell has enough protection in that table.

To be more specific, we have to find a suppression pattern in the MSA-to-county table that gives 300 units of protection to the county cell. In many cases, this cannot be accomplished without suppressing new cells in the MSA-to-county table.

Column Relations

In a single run of the disclosure analysis program, we may process a number of tables. All of the tables must have the same rows, but they may have different columns. A column relation defines the columns for a single table. For example, one column relation may specify the counties that add to an MSA, and another relation may specify the places that make up a county. One of the input files to the disclosure analysis program contains the column relations that will be used for the computer run.

Summary of the Disclosure Analysis Processing

The disclosure analysis program needs these four input files:

1) A file to give the valid row numbers.

2) A file to specify how some rows are sums of other rows. These are the row relations.

3) A file of column relations.

4) A file of data for the tables. This is the index file mentioned earlier in the document.

For each column relation, the program creates a two-dimensional table, converts it into a network, and uses the Minimal Cost Flow (MCF) subroutine to select complementary suppressions. We insert a "C" into the records on the index file that correspond to the suppressed cells. After all of the column relations have been processed, we do any backtracking that is necessary to re-check some tables for disclosures.

As the program is executing, it creates an output file that shows the tables before and after the complementary suppressions are chosen. One of the input parameters determines the amount of printing that will be produced. In addition, there are many write statements that can be "turned on" to provide more diagnostic information when the disclosure analysis program runs. Believe me, the biggest "turn on" is when the program seems to work correctly!

The program also creates a data file which contains some of the important intermediate calculations that occurred during the computer run. Laura Zayatz has written an interactive program that uses this file to create tables that can help the analysts better interpret the results of the disclosure analysis. If you understand disclosure analysis pretty well, you can simply look at this file with the computer editor to answer many of your questions about the results. By examining the file, you can tell exactly when each complementary suppression was chosen and what primary suppression it is protecting. For want of a better name, we call this file "File 50" because it is assigned a unit number of 50 in the program. This name will probably stick unless someone organizes a "Name That File" contest.

The Display Program

The disclosure analysis program can print out the tables as the complementary suppressions are selected, but a table may appear more than once. When a table is initially checked for disclosures it will be printed, but if it is re-checked during backtracking it will be printed again. From the analysts' point of view this is not desirable, because they only want to see the final tables.

With this in mind, we have written a separate program to display the final tables. It reads the same four input files as the disclosure analysis program, except that the fourth file (the index file) has all of the complementary suppressions identified. It prints the tables in pretty much the same format as the disclosure analysis program.

There are two other major advantages for having a separate display program:

a) The program is much simpler than the disclosure analysis program, so it would not be difficult to create a special version for an individual census application if the analysts wanted their tables displayed in a different format.

b) The tables are created from the final index file that EPD will use to create their publications, and the tables should give an accurate representation of the data contained on that file. These tables can be trusted more than the tables produced by the disclosure analysis program. Since the disclosure analysis program forms the output tables and updates the index file in separate parts of the program, it would be quite easy to show one set of data on an output table and not insert that data into the index file correctly.

The display program has two additional input files not used by the disclosure analysis program.

Unit 13: The Geographic Publication File. This file is described in Chapter IV of the documentation. The program uses this file to obtain the name of each geographic area. The file is an index file, with the Geographic Control Number as the key.

Unit 14: A file with the name of each state. This is used to give state names to the parts of metropolitan areas that cross state lines.

SECTION II-B: Flow Chart of the Disclosure Analysis Processing

The flow chart shown in this section gives a general plan for the disclosure analysis computer processing. Before I give the details of the plan, I want to discuss the main features.

1) A later section will give a detailed explanation of the four input files needed for the disclosure analysis program.

2) EPD will supply input file 4, which contains the data for the tables. Each record on the file will correspond to a cell in a table. The main purpose of the disclosure analysis program is to identify the cells that are complementary suppressions and insert a "C" into the matching record in the file.

3) The disclosure analysis programs will be used for several different applications. The programs are very difficult, so we will only be able to make limited modifications to suit the individual needs of each census. On the other hand, the programs to create the table displays are much simpler, and we are willing to have a separate program for each census. We would like to print the tables in the exact format requested by the subject matter analysts, even if it means having multiple display programs.

The rest of this document will provide more details about the computer programs and data files shown in the flow chart. The capital letters identify items on the flow chart.

A,B,C,D - These are the four input files that will be described in the next section. If the first row in a table is the sum of all other rows (like County Business Patterns), input file 2 is not needed.

E - The disclosure analysis computer program.

F - Unit 55 saves the number of column relations we intended to process, including the relations we need to backtrack.

G - Unit 56 contains the list of column relations we actually checked for disclosures. If the disclosure analysis run is stopped before completion, we can use files 55 and 56 to determine the column relations that remain to be processed.

To complete the disclosure analysis after a premature termination, copy file 55 to 53 and copy file 56 to 54. The two new files are used as input when the disclosure analysis program is re-run. The output file 9 from the initial computer run would be used as input file 9 to the re-run.

Input files 53 and 54 are not needed for the initial disclosure analysis runs.

H - The tallies and table listings produced by the disclosure analysis program. We can show 10 columns of a table on a single file. Larger tables have to be spread across multiple listings, so we allow for as many as 25 output files.

In production, the disclosure analysis program will probably only produce a few summary tallies, and the main output tables will be formed by the display programs.

I - This file contains intermediate calculations that identify the cells used to complement the primary suppressions. The original purpose of this file was to serve as input to the interactive analysis program. A person who has a good understanding of disclosure analysis can examine this file with the computer editor to answer most questions about the results of the computer run.

J - This is a shortened version of the file described above in section I. It only contains records for the new suppressions, and the computer editor can access it faster.

K - Laura Zayatz wrote this interactive computer program to help the analysts better understand the disclosure analysis results.

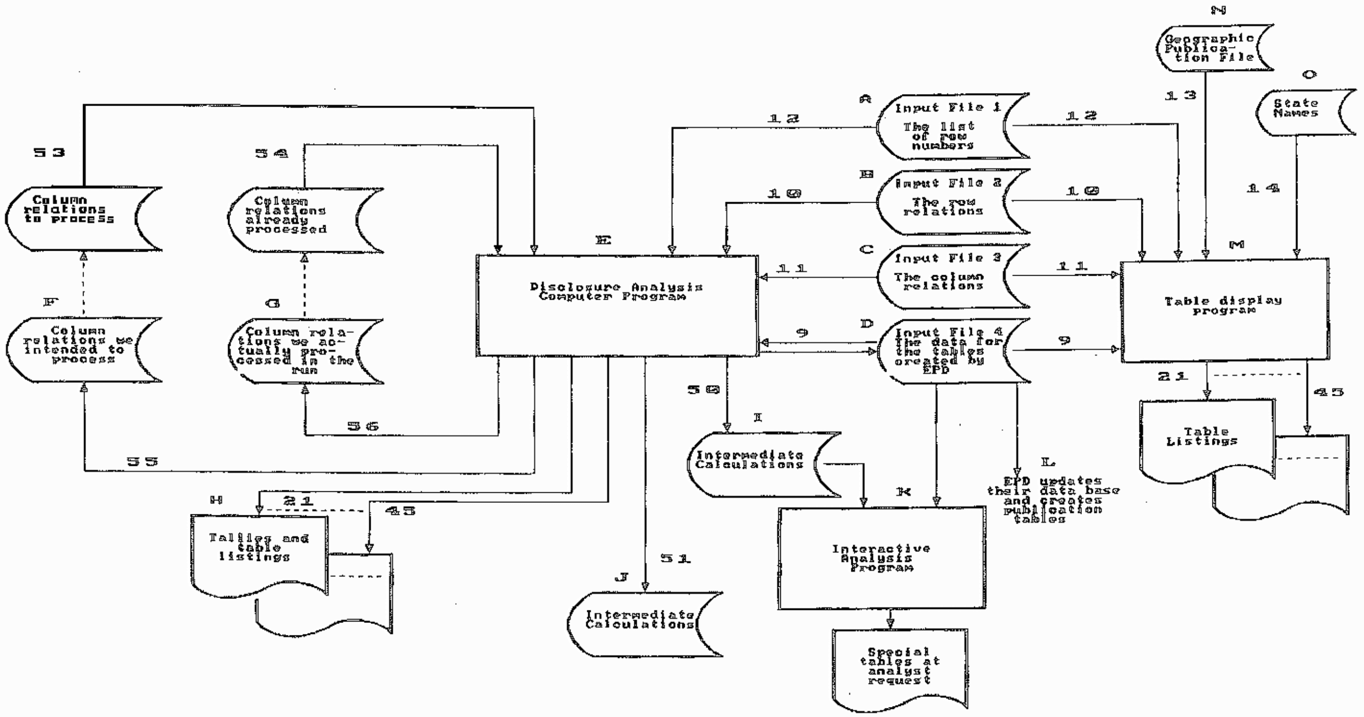
L - EPD will somehow use the disclosure analysis output file to update their data base, and they will produce final publication tables that include both primary and complementary suppressions.

M - The display programs will create tables showing the primary and complementary suppressions. As I said earlier, the display programs are relatively simple, so we can have different programs to meet the specific needs of each census.

N - An index version of the Geographic Publication File, which is described in Chapter IV. This file is needed to obtain the name of each geographic area.

O - A file of state names. It is used to give a state name to the parts of the metropolitan areas that cross state lines.

General Disclosure Analysis Flow Chart


A
Input File 1
The list of row numbers
B
Input File 2
The row relations
C
Input File 3
The column relations
D
Input File 4
The data for the tables created by EPD
E
Disclosure Analysis Computer Program
F
Column relations we intended to process
G
Column relations we actually processed in the run
H
Tallies and table listings
I
Intermediate Calculations
J
Intermediate Calculations
K
Interactive Analysis Program
Special tables at analyst request
L
EPD updates their data base and creates publication tables
M
Table display program
N
Geographic Publication File
O
State Names
Table Listings
Column relations to process
Column relations already processed
9
10
11
12
13
14
21
45
51
53
54
55
56
58

SECTION II-C: Running the Disclosure Analysis Program

The command procedure given in this section can be used to run the 2-D disclosure analysis program and the table display program. I will first describe the input and output files for the disclosure analysis program.

INPUT-TABLE is the file that contains the data for each cell in the tables. The FDL element DISCLOSE-RECORD converts the file into an index file INPUT-FILE4, which is updated by the disclosure analysis program. INPUT-FILE1, INPUT FILE2, and INPUT-FILE3 are the three other input files shown on the flow chart. They are described in detail in the next section.

The output files OUTPUT-FILE21 through OUTPUT-FILE45 may have displays of the final tables plus other information printed during the course of the run. We can only show 10 columns of a table on a single printout, so 25 files are created in case a table has 250 columns. They may be needed where we run the disclosure analysis for Texas and the columns refer to the counties in the state.

Output File 50 shows all of the cells used to complement a suppression as long as at least one cell is a new suppression or had its protection increased. File 51 is a shortened version which only shows the new suppressions or the cells that had increased protection assigned to them.

Input files 53 and 54 are not needed to run the program. If an earlier run of the program did not finish, these files can be used as input to a re-run so the program will only have to process the tables which were not checked in the earlier run.

Output file 55 contains a list of column relations we intended to process during the run. This includes the relations that need to be re-checked for disclosures during backtracking. Output file 56 has a list of the column relations actually checked in the run. These two files can become input files 53 and 54 if the job is re-run and if you want the re-run to start where the original run finished.

This is a description of the input parameters.

Parameter 1: The identifier for the run. This will appear on the output listings.

Parameter 2: An option to control the printing on files 21 through 45.

0 = final tallies only
1 = final tables and tallies
2 = additional diagnostic output

Parameter 3: First and last column relations to be processed.

Parameter 4: The primary suppression rule.

PP% = use the P% rule, where PP is the amount of protection given to each respondent. For example, 13% means to guarantee that each respondent has 13% protection.

Parameter 5: FIRST = This is the first run for this set of tables.
RERUN = This is a re-run to complete the disclosure analysis that was begun in an earlier run. Output files 55 and 56 from the original run will become input files 53 and 54 for the re-run, and the updated index file created by the original run should be an input to the re-run.

The display program uses the same INPUT-FILE1, INPUT-FILE2, and INPUT-FILE3 as the disclosure analysis program. The display program also uses INPUT-FILE4, which was updated by the disclosure analysis program. The input parameters give the identifier that will appear on the displays and specify which column relations will be used to create the final tables.

The command procedure to run the disclosure analysis and display program is shown on the next three pages.

```
$!
$!
$!          This command procedure can be used to run the 2-D disclosure analysis program.
$!
$!            The file that contains the data for the tables is converted into an 'index' file
$!
$convert/fdl=disclose-record.fdl          input-table.dat        input-file4.dat
$!
$!          Delete the output files created by the disclosure analysis program.
$!
$delete    output-file2l.dat;*
$delete    output-file22.dat;*
$delete    output-file23.dat;*
$delete    output-file24.dat;*
$delete    output-file25.dat;*
$delete    output-file26.dat;*
$delete    output-file27.dat;*
$delete    output-file28.dat;*
$delete    output-file29.dat;*
$delete    output-file30.dat;*
$delete    output-file3l.dat;*
$delete    output-file32.dat;*
$delete    output-file33.dat;*
$delete    output-file34.dat;*
$delete    output-file35.dat;*
$delete    output-file36.dat;*
$delete    output-file37.dat;*
$delete    output-file38.dat;*
$delete    output-file39.dat;*
$delete    output-file40.dat;*
$delete    output-file4l.dat;*
$delete    output-file42.dat;*
$delete    output-file43.dat;*
$delete    output-file44.dat;*
$delete    output-file45.dat;*
$!
$delete    output-file50.dat;*
$delete    output-file5l.dat;*
$delete    output-file55.dat;*
$delete    output-file56.dat;*
$!
$!
$!          Assign the input and output files and run the disclosure analysis program.
$!
$assign  input-file4.dat                          for009              ! the index file
$assign  input-file2.dat                          for0l0              ! the row relations
$assign  input-file3.dat                          for0ll              ! the column relations
$assign  input-file1.dat                          for0l2              ! list of valid row numbers
$!
$assign  output-file2l.dat                        for02l              ! 25 files that may be used
$assign  output-file22.dat                        for022              ! to print the tables formed
$assign  output-file23.dat                        for023              ! during the run
$assign  output-file24.dat                        for024                        .
$assign  output-file25.dat                        for025                        .
$assign  output-file26.dat                        for026                        .
$assign  output-file27.dat                        for027
$assign  output-file28.dat                        for028
$assign  output-file29.dat                        for029
$assign  output-file30.dat                        for030
$assign  output-file31.dat                        for031
$assign  output-file32.dat                        for032
```

```
$assign  output-file33.dat                              for033
$assign  output-file34.dat                              for034
$assign  output-file35.dat                              for035
$assign  output-file36.dat                              for036
$assign  output-file37.dat                              for037
$assign  output-file38.dat                              for038
$assign  output-file39.dat                              for039
$assign  output-file40.dat                              for040
$assign  output-file41.dat                              for041
$assign  output-file42.dat                              for042
$assign  output-file43.dat                              for043
$assign  output-file44.dat                              for044
$ assign  output-file45.dat                             for045
$!
$assign  output-file50.dat                              for050              ! intermediate calculations.
$assign  output-file51.dat                              for051              ! short version of file 50.
$!
$assign  input-file53.dat                               for053              ! these files are needed if
$assign  input-file54.dat                               for054              ! this is a re-run.
$!
$assign  output-file55.dat                              for055              ! these files may be used if
$assign  output-file56.dat                              for056              ! this job is a re-run later.
$!
$fort/lis       disclose-2d
$fort/lis       mcfsub
$link           disclose-2d, mcfsub
$run            disclose-2d
RETAIL TRADE              identifier for the run (20 characters)
1               print option:   0=final tallies, 1=basic output, 2=more details
000 000         first and last column relations to process (000 = process all)
15%             primary suppression rule:  PP% (PP=percent protection)
FIRST           'FIRST'  = first run,  'RERUN'  = re-run if first did not finish
$!
$!
$!          Delete the output print files created by the display program.
$!
$delete    print-file21.dat;*
$delete    print-file22.dat;*
$delete    print-file23.dat;*
$delete    print-file24.dat;*
$delete    print-file25.dat;*
$delete    print-file26.dat;*
$delete    print-file27.dat;*
$delete    print-file28.dat;*
$delete    print-file29.dat;*
$delete    print-file30.dat;*
$delete    print-file31.dat;*
$delete    print-file32.dat;*
$delete    print-file33.dat;*
$delete    print-file34.dat;*
$delete    print-file35.dat;*
$delete    print-file36.dat;*
$delete    print-file37.dat;*
$delete    print-file38.dat;*
$delete    print-file39.dat;*
$delete    print-file40.dat;*
$delete    print-file4l.dat;*
$delete    print-file42.dat;*
$delete    print-file43.dat;*
$delete    print-file44.dat;*
$delete    print-file45.dat;*
$!
```

```
$!
$!        Assign the input and output print files and run the display program.
$!
$assign  input-file4.dat                          for009          ! the 'index' file
$assign  input-file2.dat                          for0l0          ! the row relations
$assign  input-file3.dat                          for0ll          ! the column relations
$assign  input-file1.dat                          for0l2          ! list of valid row numbers
$!
$assign  geog-pub-file.dat                        for013          ! the Geographic Publication
File
$!                                                                ! (index file with the Geographic
                                                                  !   Control Number as the key)
$assign  state-names.dat                          for014          ! the state names.
$!
$assign  print-file2l.dat                         for02l          ! these are the output
$assign  print-file22.dat                         for022          ! print files.
$assign  print-file23.dat                         for023                   .
$assign  print-file24.dat                         for024                   .
$assign  print-file25.dat                         for025                   .
$assign  print-file26.dat                         for026
$assign  print-file27.dat                         for027
$assign  print-file28.dat                         for028
$assign  print-file29.dat                         for029
$assign  print-file30.dat                         for030
$assign  print-file31.dat                         for031
$assign  print-file32.dat                         for032
$assign  print-file33.dat                         for033
$assign  print-file34.dat                         for034
$assign  print-file35.dat                         for035
$assign  print-file36.dat                         for036
$assign  print-file37.dat                         for037
$assign  print-file38.dat                         for038
$assign  print-file39.dat                         for039
$assign  print-file40.dat                         for040
$assign  print-file41.dat                         for041
$assign  print-file42.dat                         for042
$assign  print-file43.dat                         for043
$assign  print-file44.dat                         for044
$assign  print-file45.dat                         for045
$!
$fort/lis     print-2d
$link         print-2d
$run          print-2d
RETAIL TRADE          identifier for the run (20 characters)
000 000       first and last column relations to process (000 = print all)
```

SECTION II-D: The Input Files

This section describes the four main input files to the disclosure analysis program.

General Information

If a table contains primary suppressions, the disclosure analysis program chooses other cells to be complementary suppressions. One of the input files contains the data for each cell in the table, and the other input files specify how certain rows are sums of other rows and how certain columns are sums of other columns. The files should contain only ASCII characters. Most of the fields do not need to be zero-filled, but I will indicate the fields that do require it. The following paragraphs define the input files in detail and give examples of files that could be used.

INPUT FILE 1: List of Valid Row Numbers

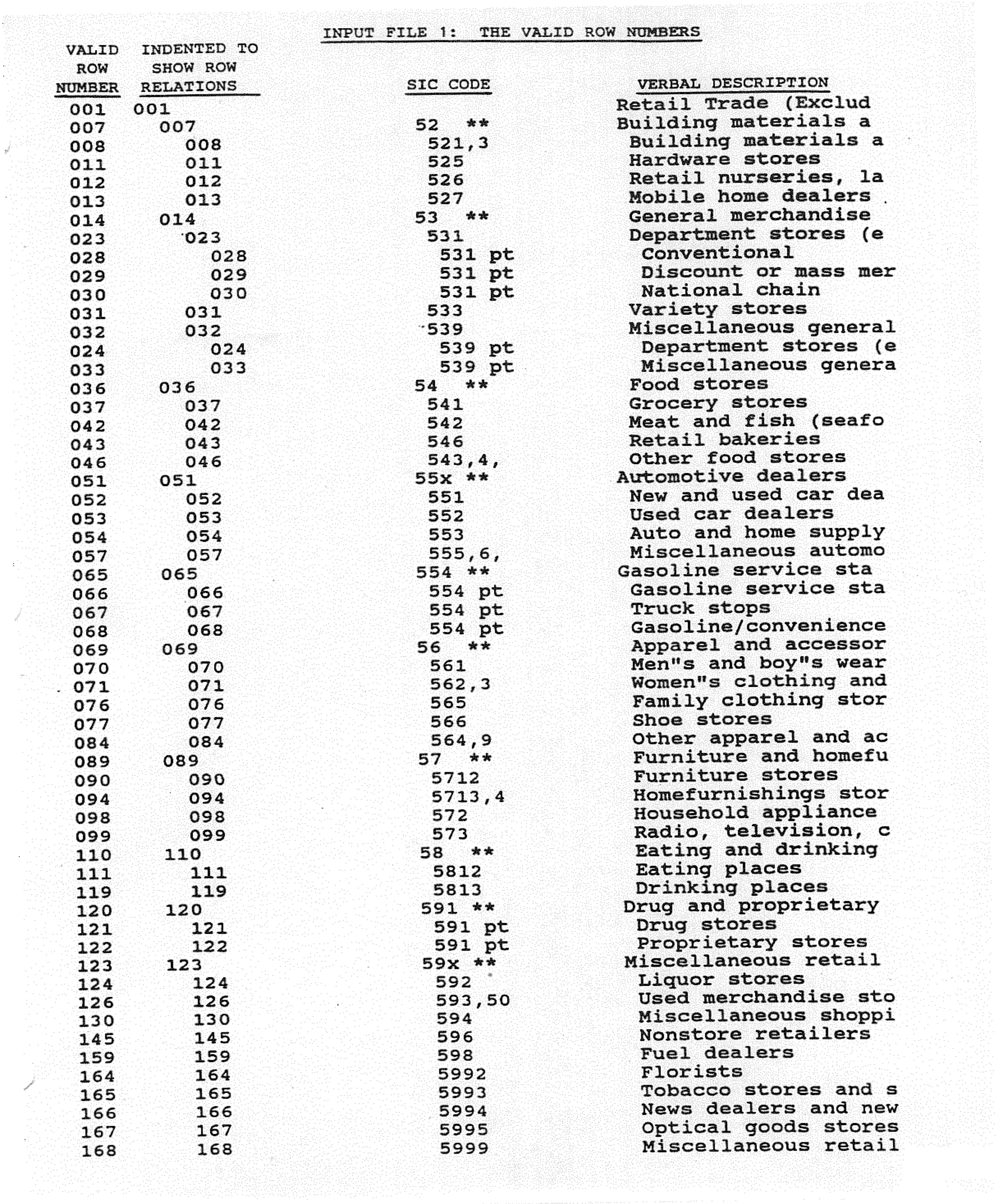
This file contains a list of the valid row numbers.

In many applications, the rows of the table refer to SIC codes, but the rows can have other meanings as well. When we ran the disclosure analysis for County Business Patterns, the first row referred to a state and the other rows referred to counties in the state. The next page contains a listing of the file that defines the row numbers we used to test the Retail Trade disclosure analysis.

The only essential fields on the file are the "valid row number" and the field showing how the row numbers should be indented when the table is printed. We ignore all other fields, such as the SIC code and the verbal description.

This is the record layout:

Character	2:7	The valid row number
Character	10:27	The row numbers indented to show the row relations
Character	30:80	The SIC codes that correspond to the row, and a verbal description of the row



INPUT FILE 1: THE VALID ROW NUMBERS

VALID ROW NUMBER	INDENTED TO SHOW ROW RELATIONS	SIC CODE	VERBAL DESCRIPTION
001	001		Retail Trade (Exclud
007	007	52 **	Building materials a
008	008	521,3	Building materials a
011	011	525	Hardware stores
012	012	526	Retail nurseries, la
013	013	527	Mobile home dealers
014	014	53 **	General merchandise
023	023	531	Department stores (e
028	028	531 pt	Conventional
029	029	531 pt	Discount or mass mer
030	030	531 pt	National chain
031	031	533	Variety stores
032	032	539	Miscellaneous general
024	024	539 pt	Department stores (e
033	033	539 pt	Miscellaneous genera
036	036	54 **	Food stores
037	037	541	Grocery stores
042	042	542	Meat and fish (seafo
043	043	546	Retail bakeries
046	046	543,4,	Other food stores
051	051	55x **	Automotive dealers
052	052	551	New and used car dea
053	053	552	Used car dealers
054	054	553	Auto and home supply
057	057	555,6,	Miscellaneous automo
065	065	554 **	Gasoline service sta
066	066	554 pt	Gasoline service sta
067	067	554 pt	Truck stops
068	068	554 pt	Gasoline/convenience
069	069	56 **	Apparel and accessor
070	070	561	Men"s and boy"s wear
071	071	562,3	Women"s clothing and
076	076	565	Family clothing stor
077	077	566	Shoe stores
084	084	564,9	Other apparel and ac
089	089	57 **	Furniture and homefu
090	090	5712	Furniture stores
094	094	5713,4	Homefurnishings stor
098	098	572	Household appliance
099	099	573	Radio, television, c
110	110	58 **	Eating and drinking
111	111	5812	Eating places
119	119	5813	Drinking places
120	120	591 **	Drug and proprietary
121	121	591 pt	Drug stores
122	122	591 pt	Proprietary stores
123	123	59x **	Miscellaneous retail
124	124	592	Liquor stores
126	126	593,50	Used merchandise sto
130	130	594	Miscellaneous shoppi
145	145	596	Nonstore retailers
159	159	598	Fuel dealers
164	164	5992	Florists
165	165	5993	Tobacco stores and s
166	166	5994	News dealers and new
167	167	5995	Optical goods stores
168	168	5999	Miscellaneous retail

INPUT FILE 2: The Row Relations

It is important that the rows be related in a hierarchical tree structure. That is, each row (except for the first row) must be a summand exactly once, and a row can be a sum in only one relation. We could not handle a table where Row 2 = Row 3 + Row 4 and Row 2 = Row 5 + Row 6, because Row 2 would be a sum in two different relations. We also could not perform disclosure analysis on a table if Row 2 = Row 5 + Row 7 and Row 3 = Row 6 + Row 7, because Row 7 would be a summand in two relations.

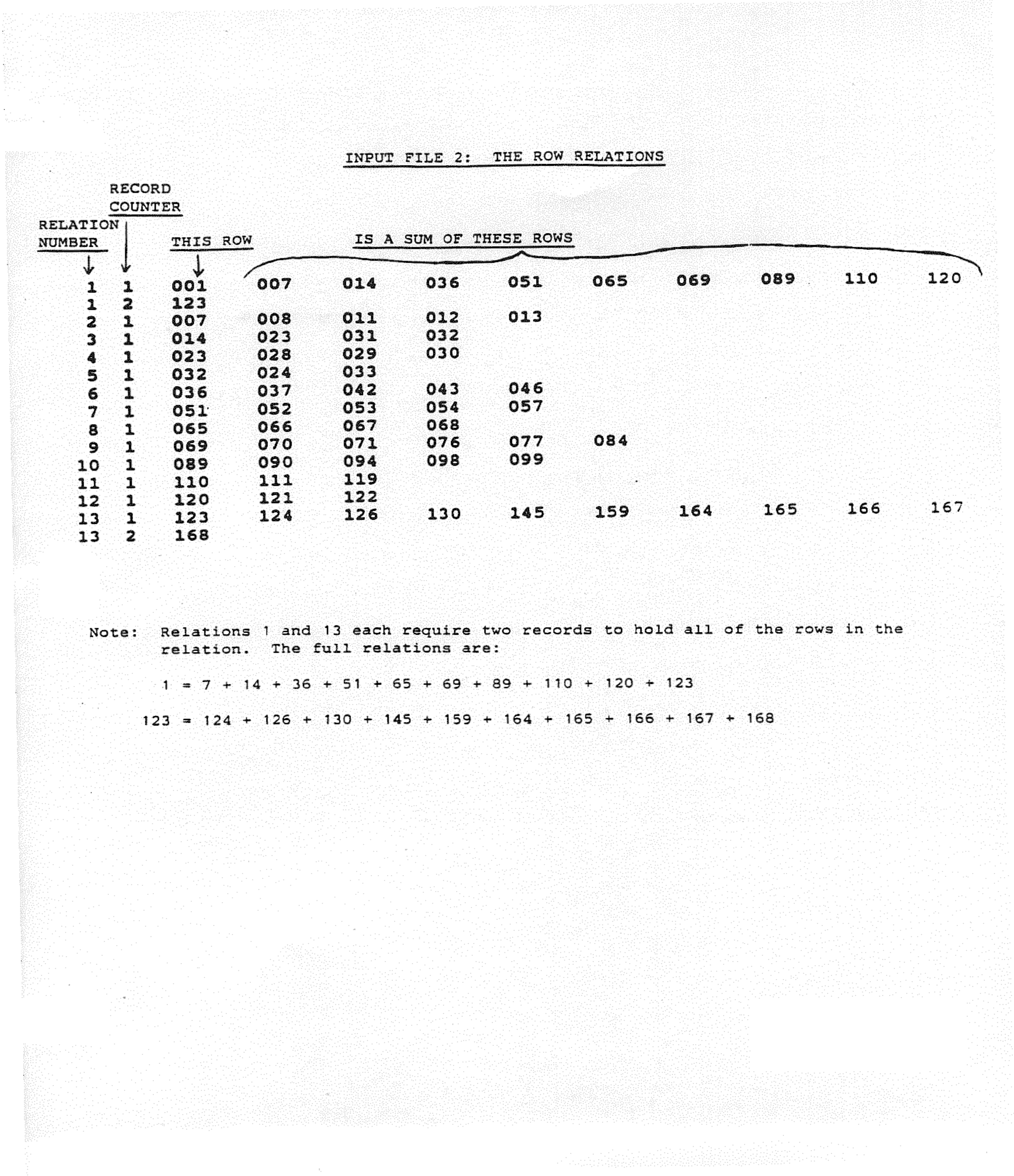
If the rows are not related in a tree structure, we will either convert the table into three dimensions or divide it into sub-tables.

This is the record layout for the row relations:

Character	2:7	The relation number
Character	9:10	The record count for the relation. Some relations are so long they require more than one record.
Character	12:80	The list of rows in the relation. Each row number is stored in 6 digits, with one space in between. There are at most 10 row numbers per record.

To obtain all of the rows in a relation, you must combine all of the records with the same relation number. The first row number in this combined list is a sum of the other rows in the list.

The next page defines the relations for the rows given on the previous page.



INPUT FILE 2: THE ROW RELATIONS

RELATION NUMBER	RECORD COUNTER	THIS ROW	IS A SUM OF THESE ROWS								
1	1	001	007	014	036	051	065	069	089	110	120
1	2	123									
2	1	007	008	011	012	013					
3	1	014	023	031	032						
4	1	023	028	029	030						
5	1	032	024	033							
6	1	036	037	042	043	046					
7	1	051	052	053	054	057					
8	1	065	066	067	068						
9	1	069	070	071	076	077	084				
10	1	089	090	094	098	099					
11	1	110	111	119							
12	1	120	121	122							
13	1	123	124	126	130	145	159	164	165	166	167
13	2	168									

Note: Relations 1 and 13 each require two records to hold all of the rows in the relation. The full relations are:

1 = 7 + 14 + 36 + 51 + 65 + 69 + 89 + 110 + 120 + 123

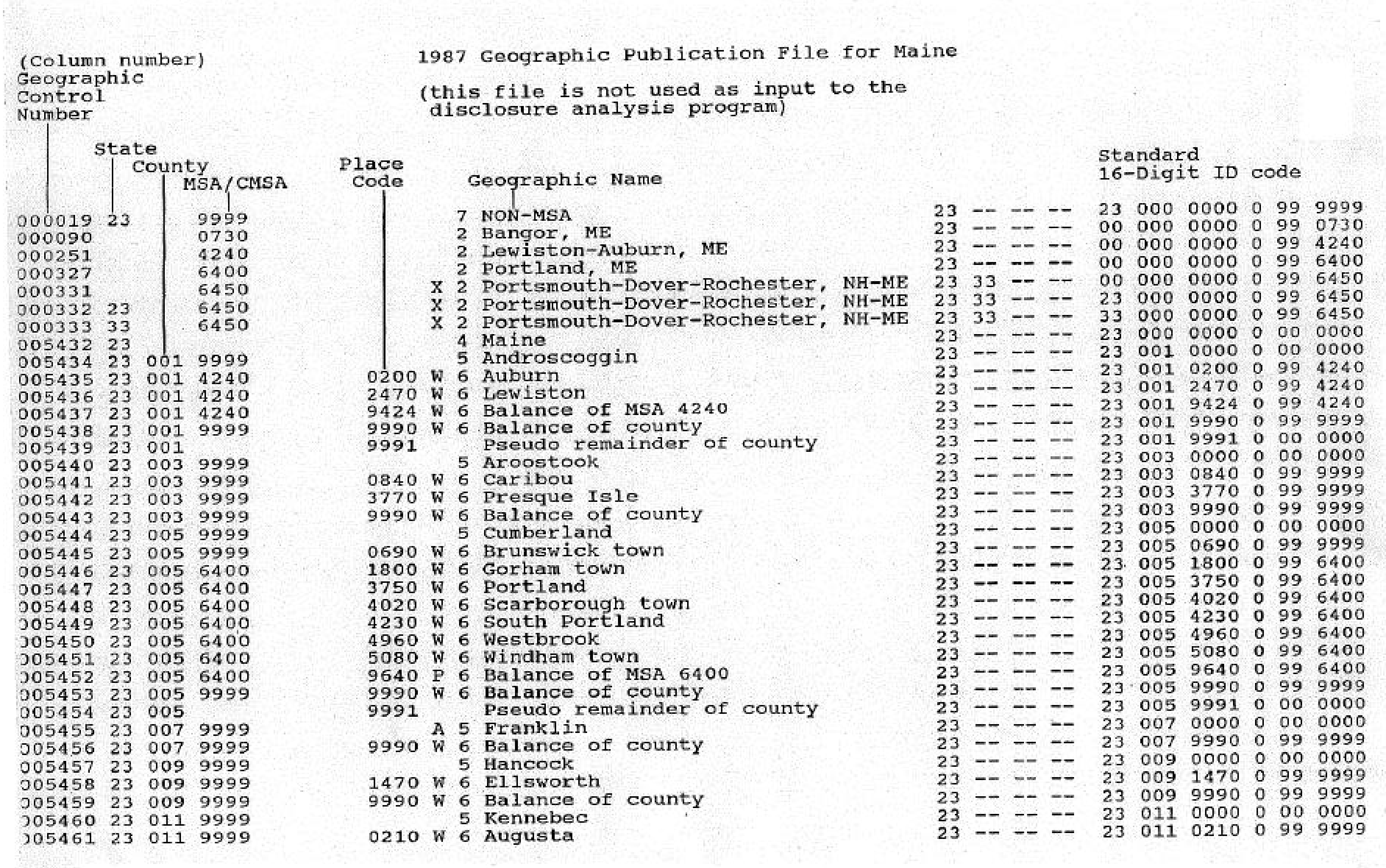
123 = 124 + 126 + 130 + 145 + 159 + 164 + 165 + 166 + 167 + 168

INPUT FILE 3: The Column Relations

This file specifies the columns that are equal to a sum of other columns. In many census applications, the columns refer to geographical areas, and the column relations describe how one area is the sum of other areas. In New England states, one relation may give the places that comprise an MSA, and another relation may list the places that make up a county. Since we process each column relation separately, the columns do not have to be related in a tree structure.

For the 1992 Economic Census, we have created a Geographic Publication File that has records for each geographic area. A unique Geographic Control Number is assigned to each area. This file is described in Chapter IV.

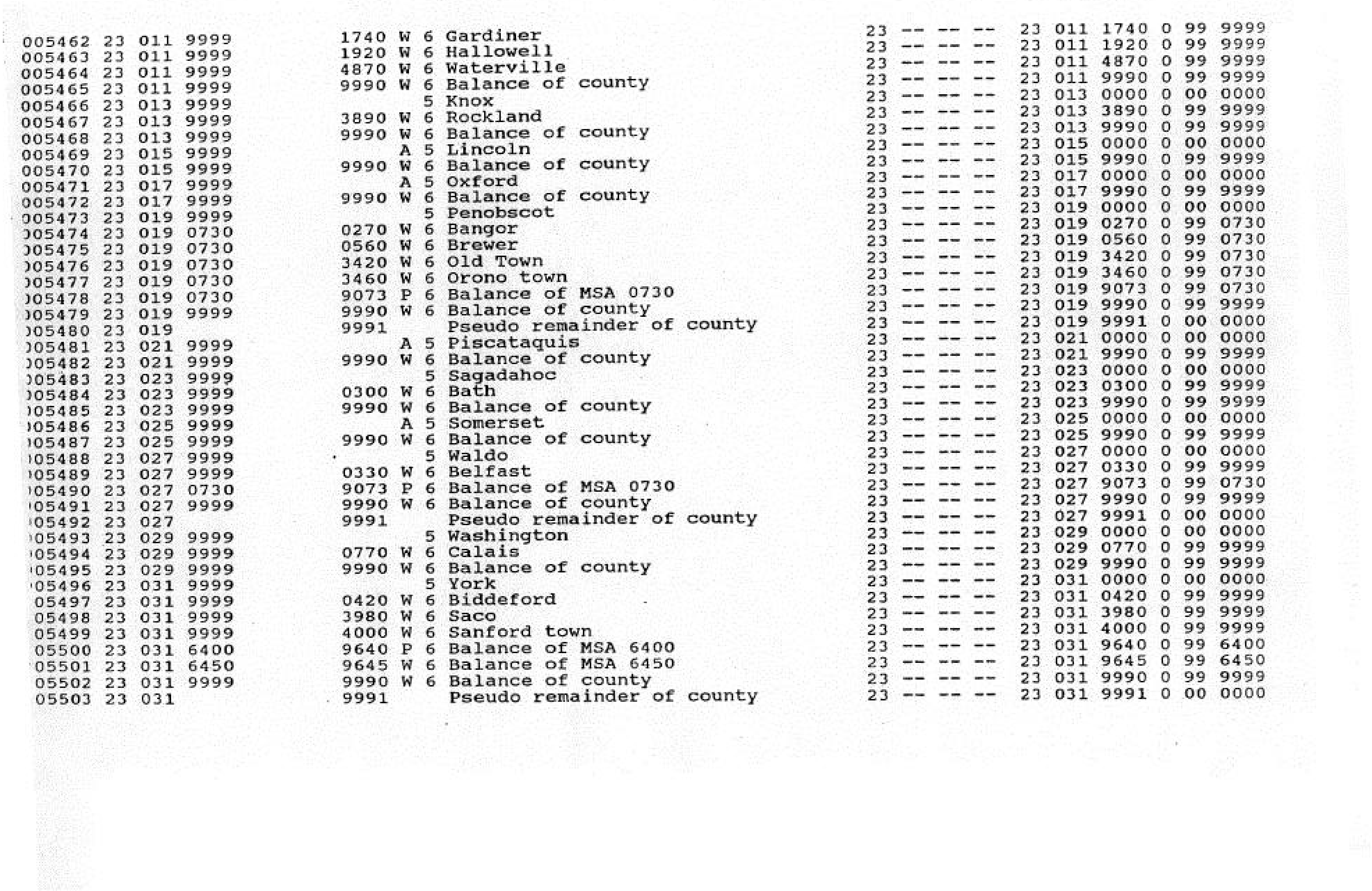
The following two pages contain a listing of the Geographic Publication File for Maine. This file is not used as input to the disclosure analysis program, but it is included so you can see how the Geographic Control Numbers are assigned. These Geographic Control Numbers are used in the column relations for many of the Economic Census tables.



```
(Column number)                             1987 Geographic Publication File for Maine
Geographic
Control                                     (this file is not used as input to the
Number                                       disclosure analysis program)

      State
         County                     Place                                                                Standard
             MSA/CMSA               Code      Geographic Name                                            16-Digit ID code

000019 23     9999                          7 NON-MSA                                 23 -- -- --        23 000 0000 0 99 9999
000090        0730                          2 Bangor, ME                              23 -- -- --        00 000 0000 0 99 0730
000251        4240                          2 Lewiston-Auburn, ME                     23 -- -- --        00 000 0000 0 99 4240
000327        6400                          2 Portland, ME                            23 -- -- --        00 000 0000 0 99 6400
000331        6450                        X 2 Portsmouth-Dover-Rochester, NH-ME       23 33 -- --        00 000 0000 0 99 6450
000332 23     6450                        X 2 Portsmouth-Dover-Rochester, NH-ME       23 33 -- --        23 000 0000 0 99 6450
000333 33     6450                        X 2 Portsmouth-Dover-Rochester, NH-ME       23 33 -- --        33 000 0000 0 99 6450
005432 23                                   4 Maine                                   23 -- -- --        23 000 0000 0 00 0000
005434 23 001 9999                          5 Androscoggin                            23 -- -- --        23 001 0000 0 00 0000
005435 23 001 4240                   0200 W 6 Auburn                                  23 -- -- --        23 001 0200 0 99 4240
005436 23 001 4240                   2470 W 6 Lewiston                                23 -- -- --        23 001 2470 0 99 4240
005437 23 001 4240                   9424 W 6 Balance of MSA 4240                     23 -- -- --        23 001 9424 0 99 4240
005438 23 001 9999                   9990 W 6 Balance of county                       23 -- -- --        23 001 9990 0 99 9999
005439 23 001                        9991     Pseudo remainder of county              23 -- -- --        23 001 9991 0 00 0000
005440 23 003 9999                          5 Aroostook                               23 -- -- --        23 003 0000 0 00 0000
005441 23 003 9999                   0840 W 6 Caribou                                 23 -- -- --        23 003 0840 0 99 9999
005442 23 003 9999                   3770 W 6 Presque Isle                            23 -- -- --        23 003 3770 0 99 9999
005443 23 003 9999                   9990 W 6 Balance of county                       23 -- -- --        23 003 9990 0 99 9999
005444 23 005 9999                          5 Cumberland                              23 -- -- --        23 005 0000 0 00 0000
005445 23 005 9999                   0690 W 6 Brunswick town                          23 -- -- --        23 005 0690 0 99 9999
005446 23 005 6400                   1800 W 6 Gorham town                             23 -- -- --        23 005 1800 0 99 6400
005447 23 005 6400                   3750 W 6 Portland                                23 -- -- --        23 005 3750 0 99 6400
005448 23 005 6400                   4020 W 6 Scarborough town                        23 -- -- --        23 005 4020 0 99 6400
005449 23 005 6400                   4230 W 6 South Portland                          23 -- -- --        23 005 4230 0 99 6400
005450 23 005 6400                   4960 W 6 Westbrook                               23 -- -- --        23 005 4960 0 99 6400
005451 23 005 6400                   5080 W 6 Windham town                            23 -- -- --        23 005 5080 0 99 6400
005452 23 005 6400                   9640 P 6 Balance of MSA 6400                     23 -- -- --        23 005 9640 0 99 6400
005453 23 005 9999                   9990 W 6 Balance of county                       23 -- -- --        23 005 9990 0 99 9999
005454 23 005                        9991     Pseudo remainder of county              23 -- -- --        23 005 9991 0 00 0000
005455 23 007 9999                        A 5 Franklin                                23 -- -- --        23 007 0000 0 00 0000
005456 23 007 9999                   9990 W 6 Balance of county                       23 -- -- --        23 007 9990 0 99 9999
005457 23 009 9999                          5 Hancock                                 23 -- -- --        23 009 0000 0 00 0000
005458 23 009 9999                   1470 W 6 Ellsworth                               23 -- -- --        23 009 1470 0 99 9999
005459 23 009 9999                   9990 W 6 Balance of county                       23 -- -- --        23 009 9990 0 99 9999
005460 23 011 9999                          5 Kennebec                                23 -- -- --        23 011 0000 0 00 0000
005461 23 011 9999                   0210 W 6 Augusta                                 23 -- -- --        23 011 0210 0 99 9999
```



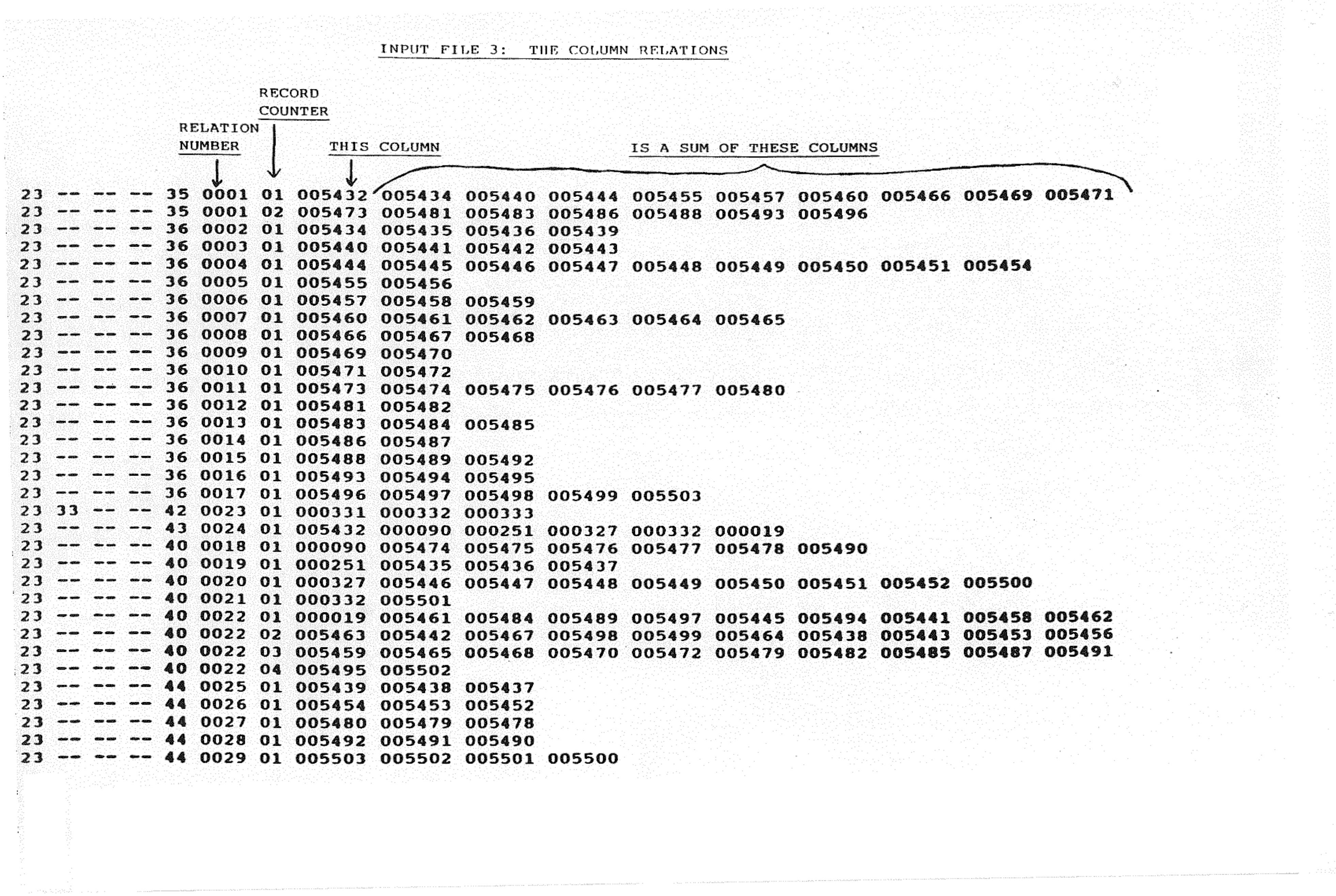
```
005462 23 011 9999   1740 W 6 Gardiner                      23 -- -- --   23 011 1740 0 99 9999
005463 23 011 9999   1920 W 6 Hallowell                     23 -- -- --   23 011 1920 0 99 9999
005464 23 011 9999   4870 W 6 Waterville                    23 -- -- --   23 011 4870 0 99 9999
005465 23 011 9999   9990 W 6 Balance of county             23 -- -- --   23 011 9990 0 99 9999
005466 23 013 9999          5 Knox                          23 -- -- --   23 013 0000 0 00 0000
005467 23 013 9999   3890 W 6 Rockland                      23 -- -- --   23 013 3890 0 99 9999
005468 23 013 9999   9990 W 6 Balance of county             23 -- -- --   23 013 9990 0 99 9999
005469 23 015 9999        A 5 Lincoln                       23 -- -- --   23 015 0000 0 00 0000
005470 23 015 9999   9990 W 6 Balance of county             23 -- -- --   23 015 9990 0 99 9999
005471 23 017 9999        A 5 Oxford                        23 -- -- --   23 017 0000 0 00 0000
005472 23 017 9999   9990 W 6 Balance of county             23 -- -- --   23 017 9990 0 99 9999
005473 23 019 9999          5 Penobscot                     23 -- -- --   23 019 0000 0 00 0000
005474 23 019 0730   0270 W 6 Bangor                        23 -- -- --   23 019 0270 0 99 0730
005475 23 019 0730   0560 W 6 Brewer                        23 -- -- --   23 019 0560 0 99 0730
005476 23 019 0730   3420 W 6 Old Town                      23 -- -- --   23 019 3420 0 99 0730
005477 23 019 0730   3460 W 6 Orono town                    23 -- -- --   23 019 3460 0 99 0730
005478 23 019 0730   9073 P 6 Balance of MSA 0730           23 -- -- --   23 019 9073 0 99 0730
005479 23 019 9999   9990 W 6 Balance of county             23 -- -- --   23 019 9990 0 99 9999
005480 23 019        9991     Pseudo remainder of county    23 -- -- --   23 019 9991 0 00 0000
005481 23 021 9999        A 5 Piscataquis                   23 -- -- --   23 021 0000 0 00 0000
005482 23 021 9999   9990 W 6 Balance of county             23 -- -- --   23 021 9990 0 99 9999
005483 23 023 9999          5 Sagadahoc                     23 -- -- --   23 023 0000 0 00 0000
005484 23 023 9999   0300 W 6 Bath                          23 -- -- --   23 023 0300 0 99 9999
005485 23 023 9999   9990 W 6 Balance of county             23 -- -- --   23 023 9990 0 99 9999
005486 23 025 9999        A 5 Somerset                      23 -- -- --   23 025 0000 0 00 0000
005487 23 025 9999   9990 W 6 Balance of county             23 -- -- --   23 025 9990 0 99 9999
005488 23 027 9999          5 Waldo                         23 -- -- --   23 027 0000 0 00 0000
005489 23 027 9999   0330 W 6 Belfast                       23 -- -- --   23 027 0330 0 99 9999
005490 23 027 0730   9073 P 6 Balance of MSA 0730           23 -- -- --   23 027 9073 0 99 0730
005491 23 027 9999   9990 W 6 Balance of county             23 -- -- --   23 027 9990 0 99 9999
005492 23 027        9991     Pseudo remainder of county    23 -- -- --   23 027 9991 0 00 0000
005493 23 029 9999          5 Washington                    23 -- -- --   23 029 0000 0 00 0000
005494 23 029 9999   0770 W 6 Calais                        23 -- -- --   23 029 0770 0 99 9999
005495 23 029 9999   9990 W 6 Balance of county             23 -- -- --   23 029 9990 0 99 9999
005496 23 031 9999          5 York                          23 -- -- --   23 031 0000 0 00 0000
 05497 23 031 9999   0420 W 6 Biddeford                     23 -- -- --   23 031 0420 0 99 9999
 05498 23 031 9999   3980 W 6 Saco                          23 -- -- --   23 031 3980 0 99 9999
 05499 23 031 9999   4000 W 6 Sanford town                  23 -- -- --   23 031 4000 0 99 9999
 05500 23 031 6400   9640 P 6 Balance of MSA 6400           23 -- -- --   23 031 9640 0 99 6400
 05501 23 031 6450   9645 W 6 Balance of MSA 6450           23 -- -- --   23 031 9645 0 99 6450
 05502 23 031 9999   9990 W 6 Balance of county             23 -- -- --   23 031 9990 0 99 9999
 05503 23 031        9991     Pseudo remainder of county    23 -- -- --   23 031 9991 0 00 0000
```

The next page gives the additive column relations for the geographic areas in Maine. The column numbers in each relation are Geographic Control Numbers taken from the previous two pages. In cases like this, the column relations are often called geographic relations.

This is a record layout for the column relations:

Character	1:16	These fields are not used by the disclosure analysis program. In the geographic relations given on the next page, they contain state codes and "relation type" fields which were needed to create the relations.
Character	17:20	The relation number
Character	22:23	The record count for the relation. Some relations are so long they require more than one record.
Character	25:93	The list of column numbers in the relation. Each column number is stored in 6 digits with a space in between. There are at most 10 numbers per record.

To obtain all of the columns in a relation, you must combine all of the records with the same relation number. The first column number in this combined list is a sum of the other columns in the list.



INPUT FILE 3: THE COLUMN RELATIONS

```
                RECORD
                COUNTER
           RELATION
           NUMBER          THIS COLUMN                 IS A SUM OF THESE COLUMNS
23 -- -- -- 35 0001 01 005432 005434 005440 005444 005455 005457 005460 005466 005469 005471
23 -- -- -- 35 0001 02 005473 005481 005483 005486 005488 005493 005496
23 -- -- -- 36 0002 01 005434 005435 005436 005439
23 -- -- -- 36 0003 01 005440 005441 005442 005443
23 -- -- -- 36 0004 01 005444 005445 005446 005447 005448 005449 005450 005451 005454
23 -- -- -- 36 0005 01 005455 005456
23 -- -- -- 36 0006 01 005457 005458 005459
23 -- -- -- 36 0007 01 005460 005461 005462 005463 005464 005465
23 -- -- -- 36 0008 01 005466 005467 005468
23 -- -- -- 36 0009 01 005469 005470
23 -- -- -- 36 0010 01 005471 005472
23 -- -- -- 36 0011 01 005473 005474 005475 005476 005477 005480
23 -- -- -- 36 0012 01 005481 005482
23 -- -- -- 36 0013 01 005483 005484 005485
23 -- -- -- 36 0014 01 005486 005487
23 -- -- -- 36 0015 01 005488 005489 005492
23 -- -- -- 36 0016 01 005493 005494 005495
23 -- -- -- 36 0017 01 005496 005497 005498 005499 005503
23 33 -- -- 42 0023 01 000331 000332 000333
23 -- -- -- 43 0024 01 005432 000090 000251 000327 000332 000019
23 -- -- -- 40 0018 01 000090 005474 005475 005476 005477 005478 005490
23 -- -- -- 40 0019 01 000251 005435 005436 005437
23 -- -- -- 40 0020 01 000327 005446 005447 005448 005449 005450 005451 005452 005500
23 -- -- -- 40 0021 01 000332 005501
23 -- -- -- 40 0022 01 000019 005461 005484 005489 005497 005445 005494 005441 005458 005462
23 -- -- -- 40 0022 02 005463 005442 005467 005498 005499 005464 005438 005443 005453 005456
23 -- -- -- 40 0022 03 005459 005465 005468 005470 005472 005479 005482 005485 005487 005491
23 -- -- -- 40 0022 04 005495 005502
23 -- -- -- 44 0025 01 005439 005438 005437
23 -- -- -- 44 0026 01 005454 005453 005452
23 -- -- -- 44 0027 01 005480 005479 005478
23 -- -- -- 44 0028 01 005492 005491 005490
23 -- -- -- 44 0029 01 005503 005502 005501 005500
```

INPUT FILE 4: The Data for the Table

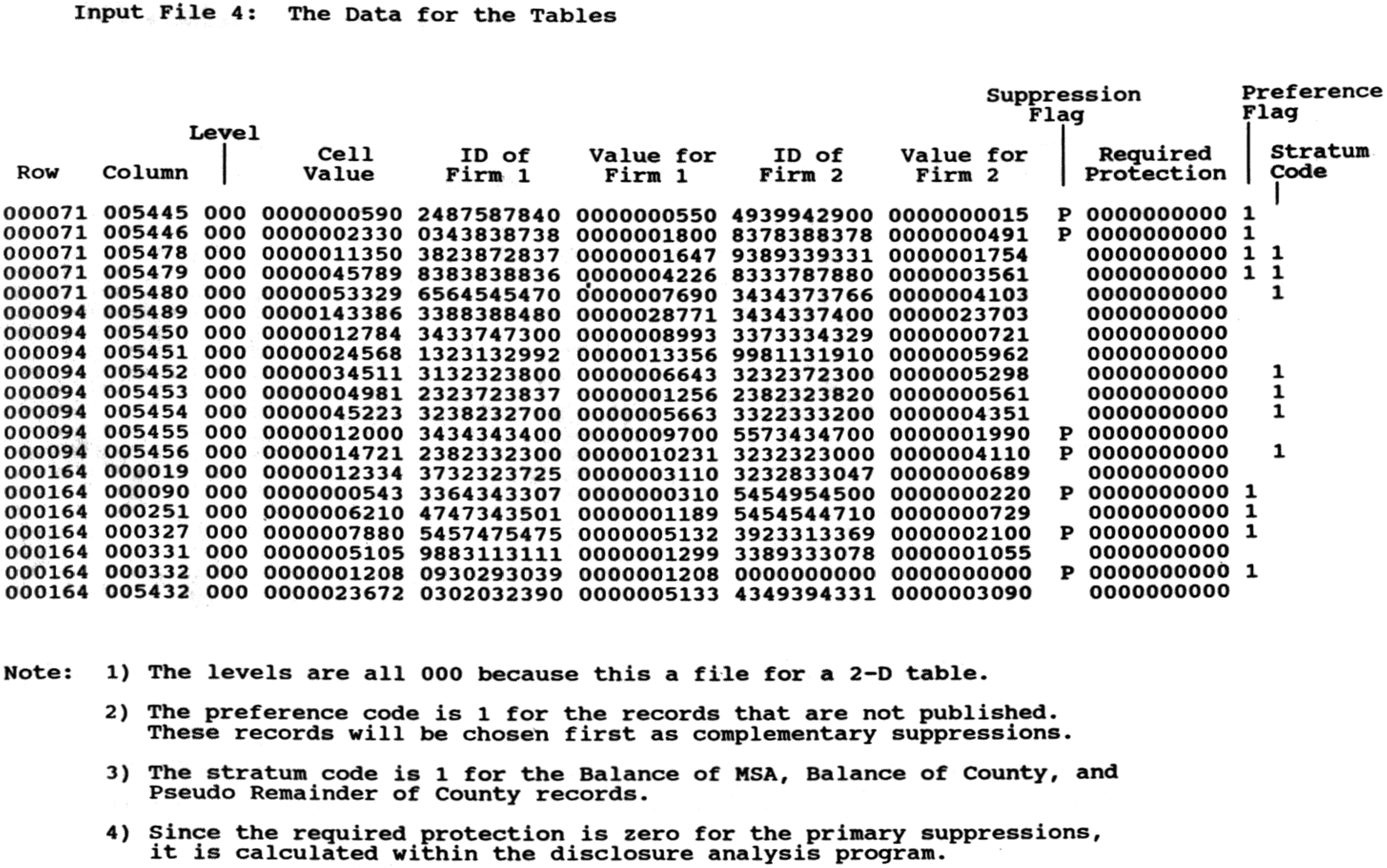
This file contains the data for the tables. Each record in the file corresponds to a cell in a table, and has row and column numbers to identify the cell. The record also contains the cell value, initial suppression flag, and other information necessary for the disclosure analysis. If the table has three dimensions, the record has a "level number" to identify the third dimension.

The record layout is given below. This record layout would be appropriate if we are using the P% primary suppression rule, because we only need to know the values for the first two respondents. The record layout may be different if a different primary suppression is used.

Characters	Field	Description
2:7	Row number (zero filled)	Often refers to an SIC code or a Kind-of-Business code.
9:14	Column number (zero filled)	Often refers to a geographic area as defined in the Geographic Publication File. The column number would be Geographic Control Number.
16:18	Level number (zero filled)	The third dimension in a 3-D table. For 2-D tables, this field is zero filled.
20:29	Cell value	This may be a figure like total sales or total value of shipments.
31:40	ID of largest respondent (zero filled)	The identification code for the respondent with the largest value in the cell.
42:51	Value of largest respondent	
53:62	ID of second largest respondent (zero filled)	The identification code for the respondent with the second largest value in the cell.
64:73	Value of second largest respondent	
76:76	Suppression flag	'P' = primary suppression 'C' = complementary suppression
78:87	Protection required	The amount of protection required if the cell is suppressed.

Characters	Field	Description
89:89	Preference code	blank = no preference '1' = When selecting complementary suppressions, choose these cells first, regardless of their value. The program is just as likely to suppress a '1' cell with a large value as it is to suppress a cell with a small value. Unpublished cells normally have this code. '9' = These cells will never be suppressed.
90:90	Publication code	blank = published cell. '1' = unpublished cell.
91:91	Stratum code	blank or '1' The program will suppress a cell in stratum '1' before suppressing a cell with a blank stratum code, but the size of the cell still matters. It will suppress a small cell in a stratum before suppressing a large cell in the same stratum.
93:94 95:97 98:101 102:102 103:104 105:108	State code County code Place code Consolidated city code CSA code MSA code	These geographic codes in characters 93 through 108 are optional. They are not required by the disclosure analysis program, but they may be useful when the output file is matched to other files.

The following page contains a sample of a data file in this format.



Input File 4: The Data for the Tables

```
                                                                                          Suppression         Preference
                                                                                             Flag             Flag
                   Level                                                                       |                |
                     |     Cell        ID of      Value for      ID of      Value for         |   Required     |  Stratum
 Row     Column      |     Value       Firm 1      Firm 1       Firm 2       Firm 2           |   Protection   |  Code
                                                                                                                   |
000071 005445 000 0000000590 2487587840 0000000550 4939942900 0000000015 P 0000000000 1
000071 005446 000 0000002330 0343838738 0000001800 8378388378 0000000491 P 0000000000 1
000071 005478 000 0000011350 3823872837 0000001647 9389339331 0000001754   0000000000 1 1
000071 005479 000 0000045789 8383838836 0000004226 8333787880 0000003561   0000000000 1 1
000071 005480 000 0000053329 6564545470 0000007690 3434373766 0000004103   0000000000   1
000094 005489 000 0000143386 3388388480 0000028771 3434337400 0000023703   0000000000
000094 005450 000 0000012784 3433747300 0000008993 3373334329 0000000721   0000000000
000094 005451 000 0000024568 1323132992 0000013356 9981131910 0000005962   0000000000
000094 005452 000 0000034511 3132323800 0000006643 3232372300 0000005298   0000000000   1
000094 005453 000 0000004981 2323723837 0000001256 2382323820 0000000561   0000000000   1
000094 005454 000 0000045223 3238232700 0000005663 3322333200 0000004351   0000000000   1
000094 005455 000 0000012000 3434343400 0000009700 5573434700 0000001990 P 0000000000
000094 005456 000 0000014721 2382332300 0000010231 3232323000 0000004110 P 0000000000   1
000164 000019 000 0000012334 3732323725 0000003110 3232833047 0000000689   0000000000
000164 000090 000 0000000543 3364343307 0000000310 5454954500 0000000220 P 0000000000 1
000164 000251 000 0000006210 4747343501 0000001189 5454544710 0000000729   0000000000 1
000164 000327 000 0000007880 5457475475 0000005132 3923313369 0000002100 P 0000000000 1
000164 000331 000 0000005105 9883113111 0000001299 3389333078 0000001055   0000000000
000164 000332 000 0000001208 0930293039 0000001208 0000000000 0000000000 P 0000000000 1
000164 005432 000 0000023672 0302032390 0000005133 4349394331 0000003090   0000000000
```

Note: 1) The levels are all 000 because this a file for a 2-D table.

2) The preference code is 1 for the records that are not published. These records will be chosen first as complementary suppressions.

3) The stratum code is 1 for the Balance of MSA, Balance of County, and Pseudo Remainder of County records.

4) Since the required protection is zero for the primary suppressions, it is calculated within the disclosure analysis program.

More information about the file.

a) When the column number is a Geographic Control Number

In the documentation about the Geographic Publication File, I describe how a Geographic Control Number is assigned to each geographical area. We find the Geographic Control Number to be extremely useful in the disclosure analysis program, but I realize that you may have no need for it in your tabulation systems. If you do not want to carry the Geographic Control Number on your data files, we can obtain it by matching your file to the Geographic Publication File as long as your file contains the 16-digit code identifying a geographical area (state, county, place, consolidated city, CSA, MSA).

b) Required Protection

The initial suppressions should be identified by a 'P' in character 76 of the record. If the record does not contain the required protection in characters 78 through 87, we will compute it.

If the required protection is less than or equal to zero, we set it equal to 1.

c) If a record has a zero value (characters 20:29), it should not be included on the file.

SECTION II-E: The Output Files

This section describes the output files created by the 2-D disclosure analysis program.

1) <u>OUTPUT FILES 21 through 45</u>

The amount of printing on these files depends on the value of the PRTOPT input parameter. If PRTOPT = 0, very little printing is produced. If PRTOPT = 1, the table for each column relation is printed after the disclosure analysis is done on that table. A table may appear several times - once when it is initially checked for disclosures, and again each time backtracking is done for that table.

The most interesting printing is done when PRTOPT = 2. Every time a new suppression is chosen, the program prints a table that identifies all cells in the suppression pattern so the user can see exactly how the initial suppression is protected. If you want to understand how the disclosure analysis program works, you should do a small run of the program, maybe with a couple column relations, and set PRTOPT equal to 2. Most of the suppression patterns are simple rectangles, but some are so unusual that you cannot help but be impressed by the Minimal Cost Flow subroutine that identifies the complementary suppressions.

It is easy to modify the program to display the suppression patterns for all initial suppressions, even if no new cells are suppressed.

When the tables are printed, these suppression flags are shown so you can identify the suppressed cells and the cells that protect the initial suppression.

'S' = The initial suppression we are trying to protect

'p' = A primary suppression used to protect the initial suppression.

'c' = An existing complementary suppression used to protect the initial suppression.

'n' = An unpublished cell used to protect the initial suppression. The input file had no data for this cell, so you might wonder how we ever assigned it a non-zero capacity. Well, it was very tricky.

'X' = A new complementary suppression chosen to protect the initial suppression.

The following cells are not used to protect the initial suppression.

'P' = A primary suppression.

'C' = A complementary suppression chosen earlier in the run.

'N' = An unpublished cell.

'T' = A cell that is very likely to be suppressed later in the run. We gave it a very low cost when we used MCF.

Just before the suppression flag, a character is printed to indicate the preference, publication, or stratum code. Since the preference and stratum code affect the cost, knowing these codes can help the user understand why a particular cell was chosen to be a complementary suppression.

These are the characters used to identify the codes.

'-'	=	The preference code is 1
'*'	=	The preference code is 9
'.'	=	The stratum code is 1
'^'	=	The publication code is 1

2) <u>OUTPUT FILE 50</u>

If you use PRTOPT = 2 to create a table showing the suppression pattern that protects an initial suppression, just below the table you will notice a list of all cells in the pattern. This list of cells is saved in output File 50. For a large run, we cannot afford to use PRTOPT = 2 to display the tables, but we can usually create File 50, which lets the user identify the cells used to protect the initial suppression. Except in the more difficult cases, it is fairly easy to reconstruct the actual suppression pattern. File 50 helps the user justify the selection of each complementary suppression.

Laura Zayatz has written an interactive program that uses File 50 for input. Among other things, this program will allow the user to print the suppression pattern for most initial suppressions.

The following record layout describes the data the 2-D disclosure analysis program writes to File 50. The 3-D program writes a few more numbers to the file, which explains the gaps in the record layout.

<u>Character</u>

2:6 -The File 50 suppression counter. This counts the number of suppression patterns written to File 50. For each initial suppression, you will note a number of cells with the same File 50 suppression counter. These are all of the cells used to protect that initial suppression.

7:10 -The column relation counter. This is the value stored in the variable CHINDX in the disclosure analysis program. It is a counter for the number of column relations processed during the run. For example, if there are 30 column relations to process, the first relation we backtrack has CHINDX=31.

11:14 -The column relation number.

The next two fields have data only if the rows are divided into groups. If no groups are used, the fields are blank. The row groups are explained in the section that describes the shortcuts available in the disclosure analysis program.

15:17 - The row group counter. For each column relation, the program keeps a counter for the number of row groups processed. This is stored in the variable GPINDX in the program.

18:20 - The row group number.

27:30 - A number that indicates the order in which the suppressions were checked within a table.

31:36 - The row number of the initial suppression being protected.

37:42 - The column number of the initial suppression.

46:55 - The value of the initial suppression.

56:65 - The required protection for the initial suppression.

66:66 - A character that identifies the preference, publication, or stratum code of the complementary suppression. This is the same identifier used in the tables printed on files 21 through 45.

67:67 - The suppression flags for the complementary suppressions that appear in the tables printed on files 21 through 45.

68:73 - The row number of the complementary suppression.

74:79 - The column number of the complementary suppression.

83:92 - The value of the complementary suppression.

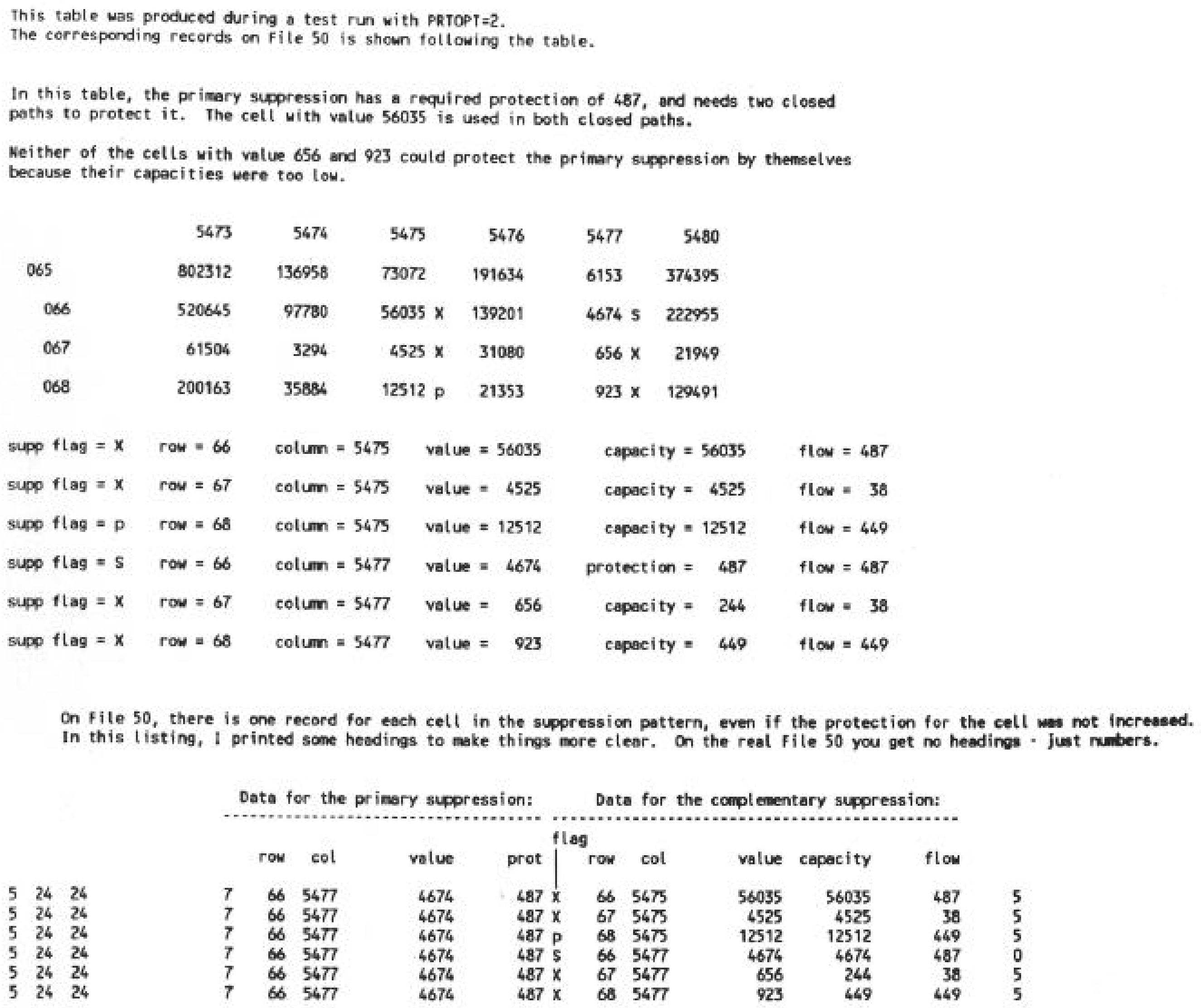
93:102 - The capacity of the complementary suppression to protect the initial suppression.

103:112 - The number of units flowing through the complementary suppression. This can be used to determine the importance of this cell in protecting the initial suppression. For example, if the initial suppression requires 1000 units of protection and this cell carries a flow of 50, the user can tell this cell is not very important. However, the cell may be very important in protecting other primary suppressions.

113:119 - The File 50 counter which indicates when this cell was assigned its maximum protection or, in other words, when the cell carried its greatest flow. This field can be very useful. For example, near the end of the run we may be protecting a primary suppression with a value of 1000, and it may require a protection of 900. The user may want to know why the protection is so high. This field on File 50 may indicate the maximum flow for the primary suppression occurred when the File 50 counter was 147. If the suppression pattern for that counter is examined, it should indicate that the primary suppression carried a flow of 900 to protect another initial suppression.

3) OUTPUT FILE 51

This is a shortened version of File 50. It only shows the new cells being suppressed and the cells whose protection was increased, whereas File 50 showed every cell in the suppression pattern. This file contains enough information to let the user identify the primary suppression that caused each complementary suppression, but in most cases it would be almost impossible for the user to reconstruct the entire suppression pattern.



This table was produced during a test run with PRTOPT=2.
The corresponding records on File 50 is shown following the table.

In this table, the primary suppression has a required protection of 487, and needs two closed paths to protect it. The cell with value 56035 is used in both closed paths.

Neither of the cells with value 656 and 923 could protect the primary suppression by themselves because their capacities were too low.

```
                  5473      5474      5475       5476       5477       5480

 065            802312    136958     73072     191634       6153     374395

   066          520645     97780     56035 X   139201       4674 S   222955

   067           61504      3294      4525 X    31080        656 X    21949

   068          200163     35884     12512 p    21353        923 X   129491

supp flag = X   row = 66   column = 5475   value = 56035   capacity = 56035   flow = 487

supp flag = X   row = 67   column = 5475   value =  4525   capacity =  4525   flow =  38

supp flag = p   row = 68   column = 5475   value = 12512   capacity = 12512   flow = 449

supp flag = S   row = 66   column = 5477   value =  4674  protection =   487   flow = 487

supp flag = X   row = 67   column = 5477   value =   656   capacity =   244   flow =  38

supp flag = X   row = 68   column = 5477   value =   923   capacity =   449   flow = 449
```

On File 50, there is one record for each cell in the suppression pattern, even if the protection for the cell was not increased.
In this listing, I printed some headings to make things more clear. On the real File 50 you get no headings - just numbers.

```
                   Data for the primary suppression:    Data for the complementary suppression:
                  -----------------------------------  -------------------------------------------
                                                   flag
                      row   col        value     prot |  row  col       value  capacity      flow
5  24  24        7    66  5477         4674      487 X   66  5475     56035     56035       487     5
5  24  24        7    66  5477         4674      487 X   67  5475      4525      4525        38     5
5  24  24        7    66  5477         4674      487 p   68  5475     12512     12512       449     5
5  24  24        7    66  5477         4674      487 S   66  5477      4674      4674       487     0
5  24  24        7    66  5477         4674      487 X   67  5477       656       244        38     5
5  24  24        7    66  5477         4674      487 X   68  5477       923       449       449     5
```

SECTION II-F: More Details about the Program

The purpose of this section of the documentation is to give more details about the computer program that selects the complementary suppressions for two-dimensional tables. I will first summarize the overall logic of the program.

a) Read the input files that define the valid row numbers, the additive row relations, and the additive column relations.

b) For each column relation, read in data from the index file to create a 2-D table.

c) Convert the table into a network and call the Minimal Cost Flow (MCF) subroutine to determine the complementary suppressions.

d) Update the index file by inserting a 'C' into the records that correspond to the new complementary suppressions.

e) Repeat this process until all column relations have been processed once.

f) When a table is being checked for disclosures, if we suppress a cell which also appears in an earlier table, backtracking must be done to re-check the earlier table for disclosures.

In the remainder of this section I will give more information about the way we use MCF to choose the complementary suppressions. The program does these six steps:

1) Determine the amount of protection required for each initial suppression.

An initial suppression may be a primary suppression defined by the programmers in EPD who prepare our input files, or it may be a complementary suppression chosen when we ran the disclosure analysis on a previous column relation. For example, a previous geographic relation may have included counties adding to an MSA, and we may have selected a particular county cell to be a complementary suppression. When we later process the column relation that defines how that county is the sum of places, the suppressed county cell will be treated as an initial suppression. We will have to suppress some place level cells to complement the county level suppression.

The required protection is the value that needs to be suppressed to complement the initial suppression. If the initial suppression is a primary suppression, the protection depends on the rule used to determine the primary suppressions. The required protection can vary a great deal depending on the values of the individual respondents within the cell.

If the initial suppression is a complementary suppression chosen in a prior column relation, the required protection is the number of units that flowed through that cell when the disclosure analysis was done on the prior relation.

2) Choose an initial suppression that needs protection.

The program checks each initial suppression individually to make sure it is protected. The order in which we check the suppressions is based upon the amount of protection they require. The suppression that needs the most protection is checked first.

Incidentally, this is one of the weaknesses in our disclosure analysis procedure. It would be better if we were able to identify the fewest number of cells needed to protect a group of initial suppressions, but we don't know how to do that very well. We make a feeble attempt to select complementary suppressions that protect more than one initial suppression, and it seems to help a little.

3) Decide how much protection each cell can give the initial suppression.

The amount of protection a cell can give the initial suppression is called the capacity of the cell. In other words, the capacity of a cell is the maximum number of units that can flow through the cell. For example, if the initial suppression has a required protection of 100, we have to select enough complementary suppressions to carry a combined flow of 100. If two cells have capacities of 70 and 80, we might decide to flow 40 units through one cell and 60 units through the other. Of course, both cells would then be chosen as complementary suppressions.

If the initial suppression is a complementary suppression chosen earlier in the run, we set the capacity of all other cells equal to their cell value. Even when the initial suppression is a primary suppression, the capacity of the other cells are usually equal to their value, but there are many exceptions. For example, if a primary suppression has only one respondent, we cannot choose another one-respondent primary suppression as a complement. If only these two cells were suppressed within a row or column of a table, each respondent could easily derive the value of the other.

Another problem occurs when respondents with the same ID code are in both the primary suppression and in a cell we want to select as a complementary suppression. In cases like this, it can be quite difficult to calculate a capacity for each cell in a theoretically correct manner. Later in this document there is a section that gives more details about the procedure to determine the capacity of a cell to protect a primary suppression. In my opinion, the method I use to calculate cell capacities is the hardest part of the disclosure analysis program, and I doubt that anyone besides myself will ever understand the logic completely.
If you have the courage to read the program in an attempt to learn exactly how I calculate cell capacities or perform any other part of the disclosure analysis, please let me know which parts you found difficult so I can improve the internal documentation.

4) Calculate a cost for suppressing each cell.

After we assign a cost and capacity to each cell, the MCF subroutine can be used to determine the cells needed to protect the initial suppression. It will select a group of cells that give the least total cost, according to the strange way MCF computes the total cost. The program tries to use other primary and existing complementary suppressions to protect the initial suppression but, if they do not fully protect the initial suppression, the program will choose new cells to suppress. To achieve this, we give a low cost to the cells that are already suppressed, and assign a higher cost to the unsuppressed cells. The costs are assigned in the following manner:

a) Unsuppressed cells

In general, we want to avoid suppressing large cells, so we would like to set the cost equal to the cell value. This causes problems when some of the cell values are too large, so I developed this technique to keep the cost within a reasonable range.

Let LIM1 and LIM2 be two parameters set in the program. For example, we could set LIM1 = 500,000, and LIM2 = 5,000,000.

If value ≤ LIM1, cost = value

If LIM1 < value ≤ LIM2, cost = LIM1 + (value - LIM1)/10 + 1

If LIM2 < value, cost = LIM1 + (LIM2 - LIM1)/10
+ (value - LIM2)/100 + 1

This procedure can have some unfortunate results. For example, assume that an initial suppression requires a protection of 100,000; and we have these three candidate cells with their costs calculated as I have described:

Cell 1:	value = 400,000	capacity = 400,000	cost = 400,000
Cell 2:	value = 450,000	capacity = 450,000	cost = 450,000
Cell 3:	value = 2,500,000	capacity = 2,500,000	cost = 500,000 + (2,500,000 - 500,000)/10 = 700,000

Assume the computer program has a choice of suppressing Cell 1 and Cell 2 and flowing 100,000 units through them both, or only suppressing Cell 3 and having it carry the entire flow of 100,000. The program would select Cell 3 to be the complementary suppression because the cost is only 700,000. Cell 1 and Cell 2 would not be chosen because their combined cost is 850,000. However, Cell 3 is really a poor choice because it contains a great deal more value than Cell 1 and Cell 2 combined.

b) Suppressed cells

We prefer to complement the initial suppression by selecting cells that were already suppressed because it would not increase the total value suppressed. The only problem is that a good deal of backtracking can result if the number of units flowing through a suppressed cell are increased. For example, assume that in a previous column relation we suppressed a cell and flowed 100 units through it. If we select that cell as a complementary suppression in a later relation and flow 500 units through it, we have to re-check the cell in the earlier relation to make sure it has a protection of 500. In many cases, it results in new suppressions being needed in the earlier relation, a prime example of the evils of backtracking.

To avoid this problem, we try to select suppressed cells that already have a large existing protection. The existing protection for a suppressed cell is equal to the number of units that flowed through the cell when it was chosen to be a complementary suppression. A cost is assigned to the suppressed cells in the following manner:

Let FLOW = the amount of protection required for the initial
suppression

PROTECT = the existing protection for a suppressed cell

If PROTECT $\geq$ FLOW, the cost of the suppressed cell = 1

If PROTECT < FLOW, the cost = (FLOW - PROTECT)/100 + 1

With this method for assigning costs, suppressed cells that have a small existing protection are given a higher cost than the suppressed cells that have a large existing protection.

5) Run MCF to select the complementary suppressions.

The MCF subroutine was also used in the 1987 disclosure analysis. Thanks to the efforts of Brian Greenberg, Jim Fagan, and Colleen Sullivan, we have a procedure to put the entire table into a single network so it can be run through MCF at one time. As I understand, in 1987 they divided the table into smaller sub-tables and ran them through MCF separately. I believe that we can choose fewer complementary suppressions by running the whole table at one time, but it may require more computer time than it would to run the sub-tables separately. We are also able to avoid the residual disclosures which are described later in the documentation.

When selecting cells to complement the initial suppression, it would be ideal to choose cells that are already suppressed and utilize the existing protection for those cells. If the protections on the suppressed cells were not increased, it would not create a need for backtracking.

With this in mind, I set the capacity of each suppressed cell equal to the existing protection for the cell, and the cost equals 1. Of course, the capacity of each unsuppressed cell is zero. MCF is then run to find out the amount of protection that can be given to the initial suppression. If the initial suppression is fully protected, then we have found a solution that assigns no new suppressions and requires no backtracking. The program prints out "Rejoice"!

Usually the initial suppression is only partially protected. For example, the initial suppression may need 100 units of protection, and we may have only been able to flow 60 units through the cells that were already suppressed. In this case, I save the cells that gave the 60 units of protection, run MCF a second time to select additional cells to provide 40 more units of protection, and combine them all into one solution.

When MCF is run the second time, the cells have their full capacity and the costs are calculated as described earlier. MCF is very ingenious in the way it selects the complementary suppressions to minimize the total cost. The main problem is that we do not like the way it computes the total cost.

For example, assume we have a primary suppression that requires a protection of 100 and we have two unsuppressed cells with values, costs, and capacities of 70 and 150. MCF will select both cells to be complementary suppressions - it will flow 70 units through the first cell and 30 units through the second cell. This is how it calculates the total cost:

total cost = (flow thru Cell 1)(cost of Cell 1) +
(flow thru Cell 2)(cost of Cell 2)

= (70)(70) + (30)(150) = 490 + 450 = 940

We would prefer to only select Cell 2 and flow all 100 units through it but MCF gives that solution a higher cost, as you can see in the following calculation.

total cost = (flow thru Cell 2)(cost of Cell 2) = (100)(150) = 1500

Therefore, MCF will choose both cells to be complementary suppressions, when actually only cell 2 is needed.

From what I understand, there is no hope of us modifying the way MCF computes the total cost. We would like the total cost to be the sum of the costs of the cells chosen to protect the initial suppression. When I asked about modifying the cost equation in MCF, I was told that MCF was designed to only solve linear programming problems, and our cost equation would make it an integer programming problem. I should understand the difference, but I have to admit that I don't.

Since the MCF cost function does not meet our needs exactly, we have to be very creative in the way we use MCF to choose the complementary suppressions. The next few paragraphs describe how the capacities and costs are modified so MCF will give us better results.

6) Change the costs and run MCF again.

We have developed a procedure to revise the costs assigned to each cell and re-run MCF to obtain a more optimal set of complementary suppressions. I believe this procedure is very similar to one implemented by Bob Hemmig for the 1987 Economic Census disclosure analysis.

If a cell was not selected to be a complementary suppression when MCF was run the second time, the capacity of that cell is set equal to zero. If the cell was chosen to be a new complementary suppression, the cost is made inversely proportional to capacity of the cell. This is done in the following manner.

Let: REQPROT = The required protection of the initial suppression

CAPAC = The capacity of the cell to protect the initial suppression.

If CAPAC $\geq$ REQPROT, the revised cost of the cell = 1

If CAPAC < REQPROT, the revised cost of the cell =
(REQPROT - CAPAC) + 10.

In the previous example, the initial suppression required a protection of 100. These are the revised costs of the two cells chosen to be complementary suppressions during the second run of MCF.

Cell 1: value = 70, capacity = 70, revised cost = (100 - 70) + 10 = 40

Cell 2: value = 150, capacity = 150, revised cost = 1

When MCF is run a third time, it will select Cell 2 as the first choice for a complementary suppression because its cost is lower. MCF will try to flow all 100 units of required protection through that cell. Since the capacity of Cell 2 is greater than 100, it can fully protect the initial suppression, and there will be no need to suppress Cell 1. This is the solution we were trying to achieve.

SECTION II-G: Cell Capacities

I. Introduction

The purpose of this section is to describe how we calculate the capacity of a cell to protect a primary suppression. The first few paragraphs summarize the ideas discussed earlier in the documentation.

You will probably find the last half of this section hard to understand. The whole procedure to compute cell capacities is very difficult, and I wish there was an easier way to do it. I also wish there was a better way to explain it.

II. General Disclosure Analysis Logic

A primary suppression is a cell in a table that must be suppressed to protect the confidentiality of the respondents whose data falls within that cell. For example, if a cell contains only two respondents, it must be suppressed because, if the cell were published, each respondent could derive the exact data for the other respondent.

In most of our tables, the columns add to other columns and the rows add to other rows. If a primary suppression is the only unpublished cell in a particular row, a data user could easily derive the value of the primary suppression by subtracting the values of the other cells from the published row total. Therefore, we have to select other cells to be complementary suppressions to protect the primary suppression. The procedure to determine the complementary suppressions is called disclosure analysis.

These are the basic steps in our method of disclosure analysis:

1) Identify a primary suppression that needs to be protected.

2) Calculate the amount of protection required by the primary suppression.

3) Calculate the amount of protection each cell in the table would give the primary suppression if the cell were suppressed. A small cell cannot give full protection to a primary suppression that requires a great deal of protection. The capacity of the cell is the amount of protection that cell could give to the primary suppression.

4) Calculate the cost of suppressing each cell in the table. If a cell is already suppressed, there is a very low cost for using that cell to complement the primary suppression. In general, the larger cells are given a larger cost because we want to avoid suppressing them if possible.

5) Use the Minimal Cost Flow (MCF) computer program to select the set of complementary suppressions. This program makes sure the complementary suppressions it chooses have enough capacity to fully protect the primary suppression, and it tries to choose a pattern of complementary suppressions that gives the least total cost.

There are two main problems with this procedure:

Problem 1) Each primary suppression is complemented separately, and not enough effort is made to find an overall optimal solution.

Problem 2) The MCF program uses network theory to select the complementary suppressions, and the built-in total cost function is different than the cost function we would like to use. We would like the total cost of a suppression pattern to be the sum of the costs of the complementary suppressions. MCF says the total cost should be the sum of the costs of the complementary suppressions multiplied by the number of units flowing through each suppression. We have not been able to modify the cost function in MCF, but we have been able to use it in such a way that it gives good solutions in most cases.

III. Initial Capacity - A Simple Example

In the following paragraphs, I will describe how we determine the capacity of each cell in the table. All of these calculations are based on the P% primary suppression rule with P = 15. Assume we are trying to protect the respondents in the following primary suppression:

Cell 1: Total Value = 970
First Respondent = R_1 = 900
Second Respondent = R_2 = 40
Remainder = REM_1 = 30

The required protection for this cell is

$(R_1)(.15) - REM_1 + 1 = 135 - 30 + 1 = 106.$

Assume we have another cell in the same row:

Cell 2: Total Value = 150
First Respondent = R_3 = 100
Second Respondent = R_4 = 30
Remainder = REM_2 = 20

The question is - Does Cell 2 protect the respondents in Cell 1, or do we have to suppress other cells in the row to protect those respondents? At first glance, it would appear that Cell 2 fully protects Cell 1 because its value is greater than the amount of protection required by Cell 1. To really answer the question, we have to think of Cell 1 and Cell 2 as one combined cell, and then see if the data for the largest respondent in Cell 1 is being disclosed. The combined cell would consist of:

$$\begin{aligned} \textit{Cell Total} &= TOT = 1{,}120 \\ \textit{First Respondent} &= R_1 = 900 \\ \textit{Second Respondent} &= R_3 = 100 \end{aligned}$$

$$\begin{array}{r} \textit{Total} \\ \textit{Remainder} \end{array} \left\{ \begin{array}{r} R_2=40 \\ REM_1 = 30 \\ REM_1 = 30 \\ REM_1 = 30 \end{array} \right\} = REM = 120$$

The data for R_1 is still being disclosed because REM < $(R_1)(.15) = (900)(.15) = 135$.
The required protection = $(R_1)(.15)$ - REM + 1 = 135 - 120 + 1 = 16.

As a result, we have determined that Cell 2 definitely helps to protect Cell 1, but it does not protect Cell 1 completely. By itself Cell 1 requires a protection of 106, and suppressing Cell 2 reduces the required protection to 16. Therefore we can say that Cell 2 has a capacity of 90 to protect the respondents in Cell 1.

This method of computing the capacity of a cell is based on an idea that Larry Cox expressed in November 1991. We knew that we could not set the capacity of a cell equal to the value of the cell, but we did not know of a simple, effective way to determine the capacity. Larry suggested we compute the required protection for the primary suppression by itself, and then compute the required protection if the primary suppression was combined with another cell. The capacity of the other cell is equal to the amount the required protection is decreased. After we determine a capacity for each cell in the table, the Minimal Cost Flow procedure can be used to select the complementary suppressions.

We know this method has flaws and can lead to cells not being given enough capacity, which can cause oversuppression. However, we all agree it is the best procedure developed so far.

IV. Initial Capacity - A More Complex Example

The problem becomes more complex when the same business firm is represented in more than one cell. Assume the first cell has the following data:

Cell 1: Total Value = 680
First Respondent = R_1 = 600
Second Respondent = R_2 = 50
Remainder = REM_1 = 30

The required protection for this cell is

$$(R_1)(.15) - REM_1 + 1 = 90 - 30 + 1 = 61.$$

Assume we have another cell in the same row:

Cell 2: Total Value = 120
First Respondent = R_3 = 70
Second Respondent = R_4 = 40
Remainder = REM_2 = 10

In addition, assume the second respondent R_4 in Cell 2 represents the same business firm as the first respondent R_1 in Cell 1. We want to calculate the capacity of Cell 2 to protect the respondents in Cell 1. If we consider Cell 1 and Cell 2 to be one combined cell, we have:

Cell Total = TOT = 800
First Respondent =$R_1 + R_4$= 640
Second Respondent = R_3 = 70

$$\textit{Total Remainder} \left\{ \begin{array}{r} R_2 = 50 \\ REM_1 = 30 \\ REM_2 = 10 \end{array} \right\} = REM = 90$$

The data for the largest respondent is being disclosed because:

$$REM < (R_1 + R_4)(.15) = (640)(.15) = 96.$$

The required protection is

$$(R_1 + R_4)(.15) - REM + 1 = (640)(.15) - 90 + 1 = 96 - 90 + 1 = 7.$$

Cell 2 helps to protect the largest respondent in Cell 1, but does not protect it completely. It reduces the required protection from 61 to 7, so we give Cell 2 a capacity of 54.

As you can imagine, there are many more cases to consider. For example, the second respondent in the first cell could represent the same business firm as the first respondent in the second cell. If you want to know exactly how we calculate the capacity of the second cell in each case, I invite you to read my computer program. I think the FORTRAN code is just as clear as any verbal description I could write. Peggy Allen (ECSD) had the courage to read the program, and she even managed to find an error in it. Thanks to her efforts, her division now has a better understanding of the way to determine the capacity of a cell, and we have a more correct computer program to do it.

V. Calculating the Final Capacity

a) The method

In the previous sections, I described how to determine the capacity of a cell to protect the respondents in a primary suppression. However, in some cases a primary suppression may need more protection than is necessary to simply protect its respondents. This can occur when the primary suppression was used to complement another suppression that needed a larger amount of protection.

We must take special steps to account for this increased protection since we process each geographic (or column) relation in a separate table. A certain geographic area may be in several tables, and we must make sure that the required protection assigned to a cell in a table is carried over to the other tables that contain the same cell. This is the reason for the procedure we call "backtracking."

For example, assume we have a primary suppression P_1 that requires a protection of 500. In addition, assume we have another primary suppression P_2 that needs 200 units of protection, and has enough capacity to protect the first primary suppression.

P_2 will be chosen to complement P_1 and will have 500 units flowed through it. This means that in any other tables we process, P_2 will require a protection of 500. The total protection of 500 assigned to P_2 does more than only protect the respondents in P_2, it also protects the respondents in P_1. In summary,

The protection needed for the respondents in P_2 = 200

The new required protection of P_2 =

$$\max \left\{ \begin{matrix} \text{protection needed for} \\ \text{the respondents in } P_2 \end{matrix} \right\} \text{, flow through } P_2 \; =$$

max {200,500} = 500

The extra protection carried by P_2
to protect the respondents in P_1 =500-200=300

Later in the disclosure analysis processing, we must make sure that P_2 is fully protected. We may need to compute the capacity of a cell C that has a value of 610. Using the technique described in the previous section, we may determine that cell C has a capacity of 150 to protect the respondents in P_2.

We have calculated that cell C only gives 150 of the 200 units of protection needed for the respondents in P_2. But what about the 300 extra units of protection carried by P_2 for the purpose of protecting P_1? Since it is too much trouble to compare the respondents in cells P_1 and C, I decided to assume that cell C can provide all 300 extra units of protection. Therefore, cell C is given a capacity of 150 + 300 = 450. In summary,

Final capacity of cell C to protect cell P_2 =

capacity of cell C to protect the respondents in P_2 +

extra protection carried by P_2 to protect the
respondents in other cells, such as P_1 =
150 + 300 = 450

Assigning a capacity to cell C in this manner allows us to approximate what the program would have done if it could have processed all of the geographic relationships in a single table.

b) An example

If the previous explanation was hard for you to understand, maybe an example would help. Assume that we are doing disclosure analysis on manufacturing data for Maine, and we are checking the places that add to an MSA. The data for two of the places are:

Place P_1: Total = T_1 = 10901
First Respondent = P_{11} = 6000
Second Respondent = P_{12} = 4500
Remainder = R_1 = 401

Required Protection = $(P_{11})(.15) - R_1 + 1 = (6000)(.15) - 401 + 1 = 900 - 400 = 500$

Place P_2: Total = T_2 = 2141
First Respondent = P_{21} = 2000
Second Respondent = P_{22} = 40
Remainder = R_2 = 101

Required Protection = $(P_{21})(.15) - R_2 + 1 = (2000)(.15) - 101 + 1 = 300 - 100 = 200$

In order to calculate the capacity of P_2 to protect P_1, we see if the combined cell would be a primary suppression. Assume the respondents in the two cells are from different business firms.

P_1 and P_2 combined: Total = 13042
First Respondent = 6000
Second Respondent = 4500
Remainder = $R_1 + P_{21} + P_{22} + R_2 = 401 + 2000 + 40 + 101 = 2542$

Because the remainder is greater than 15% of the first respondent, the combined cell is not a primary suppression. Therefore, P_2 is able to protect P_1. The 500 units of required protection for P_1 are flowed through P_2, which means that P_2 now has a new required protection of 500. P_2 needs 200 units of protection for its own respondents, and it carries 300 more units of protection that comes from P_1.

Later in the program we may be checking how the places add to a county. Place P_2 may be in a different county than P_1, so we have to make sure P_2 is protected within its own county. Assume that another Place C is in the same county as P_2, and has the following data:

Place C: Total = 610
First Respondent = $C_1 = 500$
Second Respondent = $C_2 = 30$
Remainder = $CR = 80$

To calculate the capacity of place C to protect the respondents in place P_2, we combine the data for the two places.

P_2 and C combined: Total = 2751
First Respondent = $P_{21} = 2000$
Second Respondent = $C_1 = 500$
Remainder = $P_{22} + R_2 + C_2 + CR = 40 + 101 + 30 + 80 = 251$

The required protection = $(P_{21})(.15)$ - remainder + 1
$= (2000)(.15) - 251 + 1 = 300 - 250 = 50$

By itself, P_2 required 200 units of protection for its respondents. When combined with C, the required protection is reduced to 50. Therefore, we can say that C has a capacity of 150 to protect the respondents in P_2. As described earlier, we give C a total capacity of 450 so it will be able to provide the additional 300 units of protection that came from P_1.

In summary, P_2 has a total required protection of 500, and C has a capacity of 450 to protect P_2. Therefore, we need to find other complementary suppressions to give the extra 50 units of protection to P_2.

This is pretty much what would have happened if P_2 had never even been used to protect P_1. In that case, P_2 would have required a protection of 200 for its own respondents, and C would have had a capacity of 150 to protect P_2. We would still have needed to find extra suppressions to give 50 more units of protection to P_2.

VI. Capacities of Cells Related to the Primary Suppression

If a cell contains some of the same respondents as the primary suppression we are trying to protect, I set capacity of the cell equal to its value. Consider the following table:

	C_1	C_2	C_3	C_4	Column C_1=C_2+C_3+C_4 Row R_1=R_2+R_3 Row R_2=R_{21}+R_{22} Row R_3=R_{31}+R_{32} Row R_{31}=R_{311}+R_{312} Row R_{32}=R_{321}+R_{322}
R_1					
R_2					
R_{21}					
R_{22}					
R_3					
R_{31}			**P**		
R_{311}					
R_{312}					
R_{32}					
R_{321}					
R_{322}					

Assume the cell in row R_{31} and column C_3 is the primary suppression we wish to protect. We need to first calculate the capacity of every other cell in the table to protect that primary suppression.

I first determine which cells contain the respondents from the primary suppression. These cells are flagged with a V in the following table.

	C_1	C_2	C_3	C_4
R_1	V		V	
R_2				
R_{21}				
R_{22}				
R_3	V		V	
R_{31}	V		**P**	
R_{311}	V		V	
R_{312}	V		V	
R_{32}				
R_{321}				
R_{322}				

As you can see, the cells have to be in a row that is an 'ancestor' or 'descendant' of row R_{31} which contains the primary suppression. The cells can be in the same column as the primary suppression or they may be in the first column. I set the capacity equal to the cell value for these cells that are 'related' to the primary suppression.

The procedures described in the previous sections are used to calculate the capacities of the rest of the cells.

At the present time, I cannot explain exactly why I think it is valid to set the capacity equal to the cell value for each cell related to the primary suppression. If a cell like (R_{311}, C_3) is descended from the primary suppression, every respondent in the cell is also contained in the primary suppression. If both cells are suppressed, those respondents are being totally hidden from the data user, which certainly protects them fully. The same principle holds if a cell like (R_{31}, C_1) is an ancestor of the primary suppression, or if a cell like (R_{312}, C_1) is an ancestor of a descendant.

If you find my explanation less than convincing, please try to construct an example where a cell related to the primary suppression should have a capacity less than its cell value. Maybe you can demonstrate that my technique is incorrect, or maybe you will gain such a good understanding of the problem that you will be able to explain it better than I have done.

SECTION II-H: Unpublished Cells

Throughout this documentation, I have referred to unpublished cells being in the tables. Actually, there are two types of unpublished cells. The first type have data records on the input file. Each record has a preference code of 1, which tells the disclosure analysis program to assign these cells an extremely low cost when the complementary suppressions are selected. These cells will be the first ones chosen to protect other primary suppressions.

Since we have data records for these unpublished cells, we can calculate their capacities in the normal way. Except for the low cost, the program treats these cells just like the published cells.

The other type of unpublished cells cause more difficulty because the input file does not have records for them. In most cases, we know a group of those cells could not all be zero because, according to the table structure, they can be combined to equal another cell which has a non-zero value. Without data records for these cells, it is hard to guess what their values could possibly be. Unpublished cells of this type are flagged with an `N' on some printouts of the tables.

The rest of this section is probably of more interest to computer programmers who need to understand the detailed logic of the disclosure analysis programs. The statisticians may look at the unpublished cells flagged with an `N' and think to themselves, "I guess Bob either pretended these cells had a non-zero capacity or just bypassed them when he chose the complementary suppressions". That statement pretty much summarizes the techniques described in the remainder of this section, and if that is all you want to know about the unpublished cells, there is no reason to read further.

If you have come this far, you must have a burning desire to learn how we handle the second type of unpublished cells. Since we have no data for these cells, we cannot calculate their capacities in the normal way. We should not give the cells a zero capacity, because they are often needed to protect the primary suppressions in the table. We have two choices:

a) assign the unpublished cells a non-zero capacity, or

b) remove the cells from the table and re-structure the network so they are not needed as complementary suppressions.

As you will see in the following paragraphs, both techniques are used.

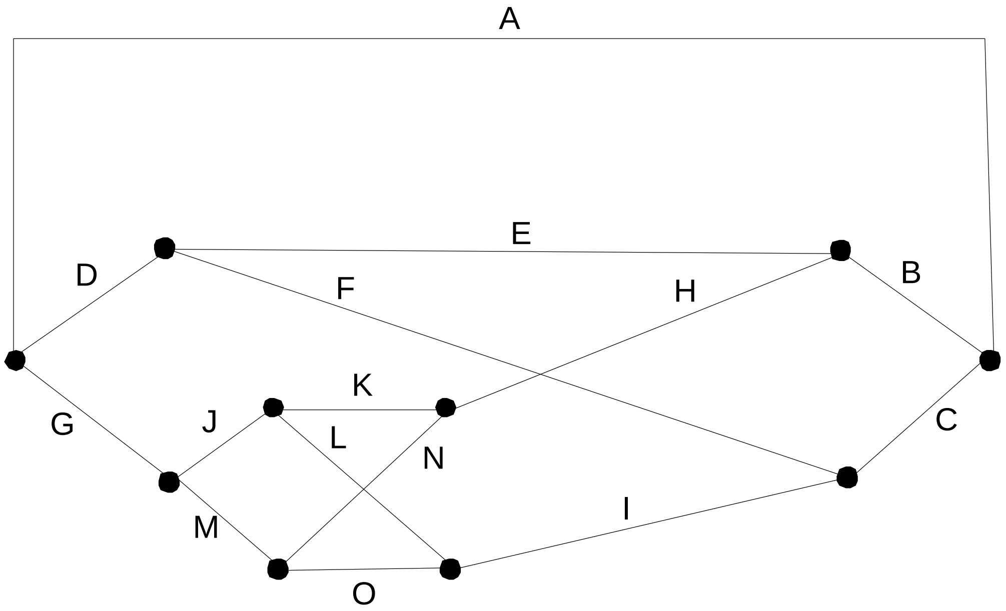
Assigning capacities to the unpublished cells

In section I-E, I showed how a table with a hierarchial row structure can be converted into a network. Assume we have the following table with its corresponding network.

	Column Total	C_1	C_2
Row Total	950 A	490 B	460 C
R_1	450 D	290 E	160 F
R_2	500 G	200 H	300 I
R_{21}	100 J	0 K	0 L
R_{22}	400 M	0 N	0 O

Row $R_2 = R_{21} + R_{22}$

The values in cells K and N should add to equal the value in cell H, but cells K and N are zero because the input file did not have records for these cells.



As explained in section I-E, if cell H is a primary suppression, a closed path must be constructed to protect the cell, and the closed path must include either cell K or cell N. In order to flow units through the closed path, either cell K or cell N must have a non-zero capacity.

I decided to make the capacities of the unpublished cells equal to the largest value they could possibly have. Of course, a cell can not have a value larger than its row or column totals. These are the capacities I assigned to the four unpublished cells.

Capacity of cell K = 100
Capacity of cell N = 200
Capacity of cell L = 100
Capacity of cell O = 300

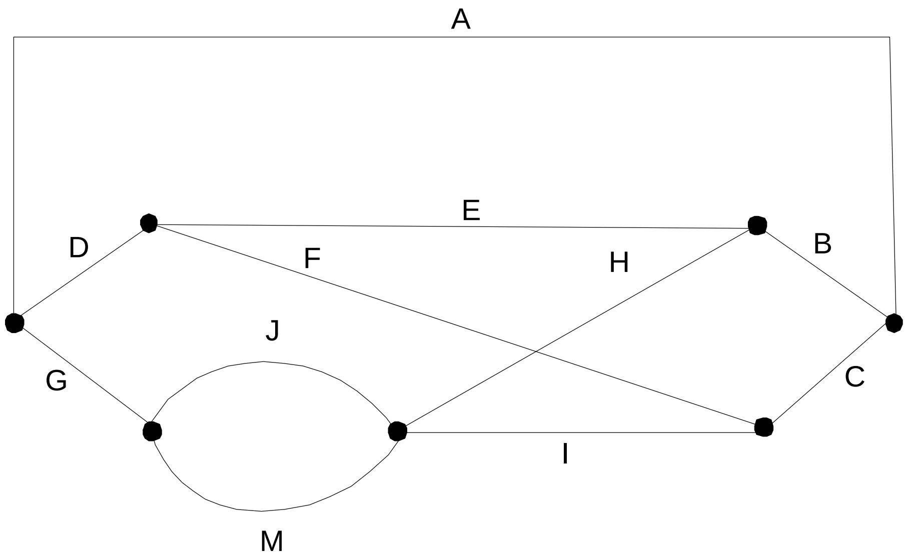
With these capacities assigned to the unpublished cells, the program can use the cells to form a closed path which protects the primary suppression in cell H.

Removing unpublished cells from the table

The unpublished cells in the previous table were grouped together, as often happens in practice. For example, in many applications the rows refer to SIC codes. The first column may have data at the county level, and the other columns may have the data for the places in the county. It is common to have certain SIC codes only published at the county level. For those SIC codes, the input file would not even have data for the place-level cells.

In cases like this, I think it is valid to remove the unpublished cells from the table. To be more specific, the arcs corresponding to the unpublished cells are removed from the network by collapsing the nodes at the ends of the arcs into a single node. When the arcs for cells K, L, N, and O are removed, the network takes on the following structure.

In this network, the arc for cell H connects to the arc for cell I, so a primary suppression in cell H can be protected by suppressing cells I, F, and E. Also, a primary suppression in cell M could be



protected by only suppressing cell J. As you can see from the network, arcs M and J form a closed path all by themselves.

With the arcs for the unpublished cells removed, the network is smaller. There are fewer arcs and nodes, which helps the Minimal Cost Flow subroutine run much faster, especially for larger tables.

Conclusion

As I said before, both techniques are used when a table has unpublished cells that have no data records on the input file. Usually there are rows in which every cell is unpublished or in which only the first cell in the row is published. In both of these cases, the unpublished cells can be removed from the table.

On a few occasions, I have seen rows that had only a few unpublished cells. I think these cases occurred because of errors in the input files, but nevertheless, my program had to deal with them. The unpublished cells were given non-zero capacities so the primary suppressions could be protected, and the program continued on its way, never looking back.

SECTION II-I: Residual Disclosures

I mentioned residual disclosures several times earlier in the documentation, so I thought I should explain what they are. If they were considered important enough for Bobby Russell to include them in a memo he wrote on May 27, 1987, they must be worth describing in detail.

In general, residual disclosures may occur when we are not able to perform disclosure analysis on an entire data set in a single table. We often use different column relations to divide our data set into separate tables, and we sometimes even have to divide the rows into groups. This may lead to two types of residual disclosures.

1) Residual disclosures for rows

Assume we have a table with a hierarchical row structure. For clarity, I will only show the number of respondents and value for one column of the table.

	Number of Respondents	Value	Suppression Flag
Total	17	5000	
R_1	11	2000	
R_{11}	10	1200	
R_{12}	1	800	P
R_2	6	3000	
R_{21}	5	2000	
R_{22}	1	1000	P

If we could not process this entire table at one time, we could break it into sub-tables and use MCF to choose the complementary suppressions within each sub-table in the following manner:

Sub-table 1:

	Number of Respondents	Value	Suppression Flag
R_1	11	2000	C
R_{11}	10	1200	
R_{12}	1	800	P

Row R_1 was chosen to complement the primary suppression in row R_{12}. You would think the program would have chosen row R_{11} instead, but let's assume that R_1 was a better choice when the entire 2-D sub-table was run through MCF.

We then process the next sub-table.

Sub-table 2:

	Number of Respondents	Value	Suppression Flag
Total	17	5000	
R_1	11	2000	C
R_2	6	3000	C

Row R_2 is chosen to protect the complementary suppression in row R_1.

Finally, the last sub-table is checked for disclosures.

Sub-table 3:

	Number of Respondents	Value	Suppression Flag
R_2	6	3000	C
R_{21}	5	2000	
R_{22}	1	1000	P

Since row R_2 was suppressed previously, this table has no disclosures.

This is the complete table with the suppressed values removed. Both primary suppressions appear to be protected.

	Number of Respondents	Value
Total	17	5000
R_1	11	C
R_{11}	10	1200
R_{12}	1	P
R_2	6	C
R_{21}	5	2000
R_{22}	1	P

When you look at this table as a whole, you can see the respondents in the two primary suppressions are not protected. The respondent in row R_{12} knows his value is 800, so he can tell the value of row R_1 is 2000. Then he can derive the value of 3000 for row R_2, and then determine the value of row R_{22} is 1000. Therefore, the respondent in row R_{12} knows the value of the respondent in row R_{22} is 1000. This is called a residual disclosure.

This would never have happened if we had run the entire table through MCF, because the cell in row R_{12} would not have had enough capacity to fully protect the cell in row R_{22}. The program would have been forced to suppress additional cells.

2) Residual disclosures among columns

Residual disclosures can also occur in the column relations just like they occur in the row relations. For example, if we are processing an MSA-to-county table, we may have a primary suppression in county A and we may select county B as a complementary suppression. Within county B we may suppress a place level cell to protect the county level suppression. In theory, the place level cell must have enough capacity to protect the original primary suppression in county A. If the cell in county A had only one respondent and if the place in county B also had only one respondent, then we may be disclosing the data for both respondents.

3) Residual geographic disclosures

This type of disclosure can occur when the columns refer to geographic areas and we do not check all possible ways the geographic areas are related. For example, assume we have data for the counties, places, and cross-over places within a state. Assume the geographic areas have the following additive relations. A 'P' refers to a primary suppression, and a 'C' refers to a complementary suppression.

		(C)		(C)
State	=	County 1	+	County 2
(C)				(C)
County 1	=	Place A	+	Place B (part 1)
(C)		(P)		(C)
County 2	=	Place E	+	Place B (part 2)
		(C)		(C)
Place B	=	Place B (part 1)+		Place B (part 2)

The primary suppression in Place E appears to be protected because every relation has at least two cells suppressed, but if we consider the state-to-place relation, it is clear the primary suppression is disclosed.

$$\text{State} = \text{Place A} + \text{Place B} + \overset{\text{(P)}}{\text{Place E}}$$

This could have been avoided if we had checked this relation for disclosures.

This example was created by Alan Saalfeld, and I included it mainly because it is so ingenious. A similar example could be developed using a division, states within the division, MSAs, and cross-over MSAs.

CHAPTER III: THREE DIMENSIONAL (3-D) DISCLOSURE ANALYSIS

SECTION III-A: General Description

When a table has cells defined by three variables and includes summary data for each variable, we consider the table to be three-dimensional. For example, County Business Patterns publishes the total payroll of firms by state, SIC code, and employee size class. One cell may contain the total payroll of firms with 10-19 employees for SIC 52 in Ohio. If this cell is a primary suppression, it must not be published. Since this cell is included in three different summary totals, a number of other cells must also be suppressed to protect it.

In the 2-D tables, a closed path of cells has to be suppressed to protect a primary suppression. This closed path could be a simple rectangle, or it may have a more elaborate structure. Each cell in the closed path must have enough capacity to protect the primary suppression. Otherwise, additional closed paths will be identified and more cells will be suppressed.

I'm not exactly sure how these ideas extend to 3-D tables. If we can construct a cube or closed box where the primary suppression is at one corner and the cells at the other corners have enough capacity to protect the primary suppression, then the primary suppression will be protected if the cells at the corners are all suppressed. This is analogous to the rectangle of suppressed cells in the 2-D tables. In a simple case like this we can understand how a 2-D rectangle is similar to a 3-D closed box, but I don't understand what type of 3-D structure would be similar to some of the more unusual 2-D closed paths.

An earlier section of the documentation explained how we convert a 2-D table into a network and use the Minimal Cost Flow (MCF) subroutine to choose the complementary suppressions. This technique is of no help in three dimensions, because a 3-D table cannot be converted into a network.

The best mathematical theory for 3-D disclosure analysis is integer programming using all of the additive relations among the cells in the 3-D table. Integer programming solutions can only be implemented for very small tables, hence the technique is impractical for our applications. The best known approximating method that can be implemented on a computer for medium-sized problems uses a linear programming subroutine like XMP to choose the complementary suppressions that give the least total cost of cells suppressed. Jim Fagan and Laura Zayatz wrote such a program. I wrote a disclosure analysis program that created the 3-D tables and identified the primary suppressions but, instead of using the MCF subroutine, I used their program to choose the complementary suppressions. This system worked very well on smaller tables, but it took over four hours of CPU time to process a table with 1500 cells. In some Economic Census applications, we have tables much larger than this. It was never clear how large of a table could even be run with XMP because a lot of memory is needed to store all of the linear relations. Therefore, we have to admit that, at this point, we do not have a good computer program to perform 3-D disclosure analysis on tables of any size.

In October, 1991 I decided to write a 3-D disclosure analysis program with the same methodology Bob Hemmig used for the 1987 Economic Census. I knew this program would not do a theoretically correct job of disclosures analysis, but I hoped it would give the respondents at least some protection and would run within a reasonable amount of computer time. Before explaining this program, I should define a few terms.

Terminology

The first two dimensions in a 3-D table are called the row and column, and the third dimension is called the level. We usually consider the rows and columns to be the horizontal dimensions, and the level to be the vertical dimension. A group of cells with the same row and column but with different levels is called a vertical shaft, or simply a shaft. A group of cells with different rows and columns but the same value for the level is called a horizontal 2-D table. For example, if a 3-D table has 4 rows, 5 columns, and 10 levels, we have 10 horizontal 2-D tables (one for each level) and 20 vertical shafts (one for each row and column combination).

3-D Disclosure Analysis Using MCF

I want to first discuss a simple technique for 2-D disclosure analysis. If we were not able to convert a 2-D table into a network and use MCF to select the complementary suppressions, we could probably do a decent job of disclosure analysis by just checking each row and column for disclosures. If enough cells were suppressed to protect the primary suppressions within each row and column, we could assume there were no disclosures in the table as a whole. An example given early in the documentation shows how this procedure can allow disclosures because it does not guarantee that a closed path of suppressions exists.

Despite its theoretical shortcomings, I decided to use a similar technique for the 3-D disclosure analysis. Each horizontal 2-D table is checked for disclosures, and cells are suppressed to make sure every primary suppression is protected. Then each vertical shaft is checked, and additional complementary suppressions are chosen when necessary. If any new cells are suppressed when the shafts are checked, the 2-D horizontal tables must be re-checked to make sure those new suppressions are protected. Of course, if more new cells are suppressed when the 2-D horizontal tables are re-checked, the vertical shafts containing those cells must be re-checked for disclosures. This process continues until all of the cells appear to be protected. If it goes on too long, with the horizontal tables and the shafts being checked and re-checked, the program has a shortcut that makes the process finish quicker. Since the program uses the MCF subroutines to process the 2-D horizontal tables, we call this 3-D disclosure analysis program the "MCF version."

This program seems to run within a reasonable amount of computer time, but it has two main drawbacks. Since the horizontal tables and the vertical shafts are checked separately, the program is not able to find a good overall combination of cells to protect a primary suppression with a minimal total cost. For example, the program may suppress a small cell when the horizontal tables are checked, but this cell may cause a large complementary suppression within its vertical shaft. The program makes a valiant effort to avoid suppressing cells that will require additional suppressions, but it still oversuppresses in many cases.

On the other hand, the program can undersuppress and may leave some primary suppressions totally unprotected. Even though a cell may appear to be protected within its horizontal table and vertical shaft, it is sometimes possible to derive the exact value for that cell by using some mathematical calculations. Remember the example in the first part of the documentation where we calculated the value of a cell that seemed to be protected? A similar thing can be done in 3-D tables. For sure, a 3-D example would be more complicated and you might think a data user would never go to such lengths to determine the value for one of our suppressed cells, but there are linear programming computer programs that could be used to make the job much easier.

Laura Zayatz has written a program to identify the cells left under-protected by the disclosure analysis program. Her program uses the XMP subroutine and, as can be expected, runs much slower than the disclosure analysis program. After her program identifies the under-protected cells, the analysts are able to choose additional suppressions to guarantee that all respondents are protected.

In my opinion, doing the 3-D disclosure analysis separately for the 2-D horizontal tables and the vertical shafts has another advantage besides just running fast--it is easier for the analyst to understand what happened during the run. For each primary suppression, the XMP subroutine creates the entire three dimensional suppression pattern in one fell swoop, and it can be hard to understand why it selected certain cells. It is difficult for us to visualize 3-D tables, let alone a 3-D suppression pattern. Since the MCF version only operates on a
2-D table or a one-dimensional shaft, it is easier for us to understand why each cell was suppressed. The analysts are already adept at reviewing 2-D tables where MCF selected the complementary suppressions, and these skills can be carried over to the 2-D horizontal tables.

SECTION III-B: The 3-D Disclosure Analysis Program

The 3-D disclosure analysis program is very similar to the 2-D program. The main difference is the subroutine which checks for disclosures in the vertical shafts.

For each column relation, the program reads in data from the index file and creates a 3-D table. Each 2-D horizontal table is then converted into a network and checked for disclosures. The technique used to check for disclosures in the 2-D horizontal table is almost identical to the one used in the 2-D disclosure analysis program. The main difference is that the costs are adjusted to avoid suppressing cells that would cause new suppressions in their vertical shafts.

The costs are adjusted in the following manner. Actually, the program adjusts the cell values and then uses the adjusted value to compute the cost. Before selecting an initial suppression and using MCF, the program checks each cell in the horizontal table to see if the shaft containing that cell contains other suppressions. If the shaft has no other suppressions, then suppressing the cell would require at least one other cell in the shaft to be suppressed. Therefore, the value of the cell is increased so the cell will later be assigned a higher cost.

On the other hand, if the shaft only contains one suppression or if the suppressed cells in the shaft do not protect each other, the shaft needs extra suppressions. The value of the cell is decreased so the program will give it a lower cost.

If the shaft already contains suppressions, then it may not make any difference whether we suppress the cell or not. If the existing suppressions in the shaft protect each other and if they have enough capacity to protect the cell if it were suppressed, then the value of the cell is not adjusted. The cost will depend on the true cell value.

The logic used in the computer program is a little more involved, but I don't have the ambition to describe it in detail. If you read the code in the program and the comments that go with it, you might be able to figure it out.

The important thing to understand is that the cell values are adjusted before we start to check the initial suppressions to see if they are protected. The adjusted values are used to determine the costs when the 2-D horizontal table is checked for disclosures.

After all of the 2-D horizontal tables are processed, the program checks each vertical shaft to make sure the values of the suppressed cells cannot be estimated from the values of the published cells. That is, we make sure there are no disclosures in the vertical shafts.

Since the shafts are only one-dimensional, you would think it should be relatively easy to check for disclosures and choose complementary suppressions, but it is almost as hard as checking a 2-D table for disclosures. I even use a procedure very much like the one used in the 2-D disclosure analysis program - the shaft is converted into a network, an initial suppression is chosen, costs are assigned to each cell in the shaft, and MCF is used to select the complementary suppressions. If new cells are suppressed in the shaft, the costs are reversed and MCF is re-run to release some of the newly suppressed cells.

Before the costs are assigned, the values of the cells in the shaft are adjusted. For each cell in the shaft, we examine the row and column of the 2-D horizontal table that contains that cell. If the row and column have no suppressed cells, then suppressing this cell would cause the need for at least two additional suppressions. In this case, the value of the cell is increased. As you might expect, the logic in the computer program is more complicated, but at least you understand the general idea.

If new cells are suppressed in the vertical shafts, the 2-D horizontal tables must be re-checked for disclosures. Of course, we do not need to check every suppressed cell in the 2-D table. We only have to make sure the new suppressions are protected within the 2-D table. After the 2-D horizontal tables are re-checked, the shafts are again checked for disclosures. This process repeats until no disclosures are found in the shafts.

After the disclosure analysis is done for this column relation, the index file is updated. The rest of the column relations are processed, and backtracking is done if necessary. This completes the 3-D disclosure analysis.

SECTION III-C: Running the Program

The command procedure to run the 3-D disclosure analysis and display programs is very similar to the command procedure used for the 2-D programs. The only difference is the extra input parameter that specifies the number of levels in the 3-D table.

An example of a command procedure is given in the next three pages.

```
$!
$!          This command procedure can be used to run the 3-D disclosure analysis program.
$!
$!           The file that contains the data for the tables is converted into an 'index' file
$!
$convert/fdl=disclose-record.fdl      input-table.dat     input-file4.dat
$!
$!          Delete the output files created by the disclosure analysis program.
$!
$delete   output-file2l.dat;*
$delete   output-file22.dat;*
$delete   output-file23.dat;*
$delete   output-file24.dat;*
$delete   output-file25.dat;*
$delete   output-file26.dat;*
$delete   output-file27.dat;*
$delete   output-file28.dat;*
$delete   output-file29.dat;*
$delete   output-file30.dat;*
$delete   output-file3l.dat;*
$delete   output-file32.dat;*
$delete   output-file33.dat;*
$delete   output-file34.dat;*
$delete   output-file35.dat;*
$delete   output-file36.dat;*
$delete   output-file37.dat;*
$delete   output-file38.dat;*
$delete   output-file39.dat;*
$delete   output-file40.dat;*
$delete   output-file4l.dat;*
$delete   output-file42.dat;*
$delete   output-file43.dat;*
$delete   output-file44.dat;*
$delete   output-file45.dat;*
$!
$delete   output-file50.dat;*
$delete   output-file5l.dat;*
$delete   output-file55.dat;*
$delete   output-file56.dat;*
$!
$!
$!          Assign the input and output files and run the disclosure analysis program.
$!
$assign  input-file4.dat                          for009           ! the index file
$assign  input-file2.dat                          for0l0           ! the row relations
$assign  input-file3.dat                          for0ll           ! the column relations
$assign  input-file1.dat                          for0l2           ! list of valid row numbers
$!
$assign  output-file2l.dat                        for02l           ! 25 files that may be used
$assign  output-file22.dat                        for022           ! to print the tables formed
$assign  output-file23.dat                        for023           ! during the run
$assign  output-file24.dat                        for024                      .
$assign  output-file25.dat                        for025                      .
$assign  output-file26.dat                        for026                      .
$assign  output-file27.dat                        for027
$assign  output-file28.dat                        for028
$assign  output-file29.dat                        for029
```

```
$assign  output-file30.dat                         for030
$assign  output-file31.dat                         for031
$assign  output-file32.dat                         for032
$assign  output-file33.dat                         for033
$assign  output-file34.dat                         for034
$assign  output-file35.dat                         for035
$assign  output-file36.dat                         for036
$assign  output-file37.dat                         for037
$assign  output-file38.dat                         for038
$assign  output-file39.dat                         for039
$assign  output-file40.dat                         for040
$assign  output-file41.dat                         for041
$assign  output-file42.dat                         for042
$assign  output-file43.dat                         for043
$assign  output-file44.dat                         for044
$assign  output-file45.dat                         for045
$!
$assign  output-file50.dat                         for050          ! intermediate calculations.
$assign  output-file51.dat                         for051          ! short version of file 50.
$!
$assign  input-file53.dat                          for053          ! these files are needed if
$assign  input-file54.dat                          for054          ! this is a re-run.
$!
$assign  output-file55.dat                         for055          ! these files may be used if
$assign  output-file56.dat                         for056          ! this job is a re-run later.
$!
$fort/lis     disclose-3d-mcf
$fort/lis     mcfsub
$link         disclose-3d-mcf,mcfsub
$run          disclose-3d-mcf
WHOLESALE TRADE               identifier for the run (20 characters)
1             print opt:    0=tallies, 1=tables, 2=shaft details, 3=more details
03            number of levels in the third dimension
000 000       first and last column relations to process (000 = process all)
PP%           primary suppression rule:  PP% (PP=percent protection)
FIRST         'FIRST'  = first run,  'RERUN'  = re-run if first did not finish
$!
$!
$!      Delete the output print files created by the display program.
$!
$delete   print-file21.dat;*
$delete   print-file22.dat;*
$delete   print-file23.dat;*
$delete   print-file24.dat;*
$delete   print-file25.dat;*
$delete   print-file26.dat;*
$delete   print-file27.dat;*
$delete   print-file28.dat;*
$delete   print-file29.dat;*
$delete   print-file30.dat;*
$delete   print-file31.dat;*
$delete   print-file32.dat;*
$delete   print-file33.dat;*
$delete   print-file34.dat;*
$delete   print-file35.dat;*
$delete   print-file36.dat;*
$delete   print-file37.dat;*
$delete   print-file38.dat;*
$delete   print-file39.dat;*
$delete   print-file40.dat;*
$delete   print-file4l.dat;*
$delete   print-file42.dat;*
$delete   print-file43.dat;*
```

```
$delete   print-file44.dat;*
$delete   print-file45.dat;*
$!
$!
$!
$!        Assign the input and output print files and run the display program.
$!
$assign  input-file4.dat                       for009        ! the 'index' file
$assign  input-file2.dat                       for0l0        ! the row relations
$assign  input-file3.dat                       for0ll        ! the column relations
$assign  input-file1.dat                       for0l2        ! list of valid row numbers
$!
$assign  geog-pub-file.dat                     for013        ! the Geog Publication File
$!                                                           ! (index file with the Geog
$assign  state-names.dat                       for014        ! the state names.
$!
$assign  print-file2l.dat                      for02l        ! these are the output
$assign  print-file22.dat                      for022        ! print files.
$assign  print-file23.dat                      for023                .
$assign  print-file24.dat                      for024                .
$assign  print-file25.dat                      for025                .
$assign  print-file26.dat                      for026                .
$assign  print-file27.dat                      for027
$assign  print-file28.dat                      for028
$assign  print-file29.dat                      for029
$assign  print-file30.dat                      for030
$assign  print-file31.dat                      for031
$assign  print-file32.dat                      for032
$assign  print-file33.dat                      for033
$assign  print-file34.dat                      for034
$assign  print-file35.dat                      for035
$assign  print-file36.dat                      for036
$assign  print-file37.dat                      for037
$assign  print-file38.dat                      for038
$assign  print-file39.dat                      for039
$assign  print-file40.dat                      for040
$assign  print-file41.dat                      for041
$assign  print-file42.dat                      for042
$assign  print-file43.dat                      for043
$assign  print-file44.dat                      for044
$assign  print-file45.dat                      for045
$!
$fort/lis      print-3d
$link          print-3d
$run           print-3d
WHOLESALE TRADE                identifier for the run (20 characters)
03             the number of levels in the third dimension
000 000        first and last column relations to process (000 = print all)
$!
```

SECTION III-D: The Input Files

The input data files for the 3-D disclosure analysis program are the same as they were for the 2-D program, but there is a new input parameter which gives the number of levels in the third dimension.

The 3-D program also has a parameter which defines the amount of printing that will be produced on output files 21 through 45. If the value of this parameter is 0, only the final tallies are printed. A value of 1 causes the tables to be printed after each column relation is processed. If the parameter has a value of 2, the program prints more details about the selection of complementary suppressions within the shafts. With this printing, the user should be able to understand exactly why each suppression was chosen when the shafts were checked for disclosures. If the parameter has a value of 3, the program displays the suppression patterns in each 2-D horizontal table, just like it did in the 2-D program.

SECTION III-E: The Output Files

The 3-D disclosure analysis program creates basically the same output files as the 2-D program, with a couple of small differences. As explained in the previous section, output file 21 may contain information showing how cells are suppressed in the vertical shafts.

The biggest changes are in output files 50 and 51, which have four new fields on the output records. When the 2-D horizontal tables are checked for disclosure, the program writes records to file 50 that are identical to the types of records written by the 2-D program, except for the addition of these four fields. When the vertical shafts are processed, the level number in characters 24:26 is not needed, and the suppression number in characters 27:30 is not defined. The same changes apply to the records written to output file 51.

This is the record layout for output files 50 and 51.

<u>Character</u>

2:6 The File 50 suppression counter. This counts the number of suppression patterns written to File 50. For each initial suppression, you will note a number of cells with the same File 50 suppression counter. These are all of the cells used to protect that initial suppression.

7:10 The column relation counter. This is the value stored in the variable CHINDX in the disclosure analysis program. It is a counter for the number of column relations processed during the run. For example, if there are 30 column relations to process, the first relation we backtrack has CHINDX=31.

11:14 The column relation number.

The next two fields have data only if the rows are divided into groups. If no groups are used, the fields are blank.

15:17 The row group counter. For each column relation, the program keeps a counter for the number of row groups processed. This is stored in the variable GPINDX in the program.

18:20 The row group number.

21:23 A counter for the number of times we check and re-check the horizontal tables and vertical shafts.

24:26 The level number of the 2-D horizontal table.

27:30 A number that indicates the order in which the suppressions were checked within a table.

31:36 The row number of the initial suppression being protected.

37:42 The column number of the initial suppression.

43:45 The level of the initial suppression.

46:55 The value of the initial suppression.

56:65 The required protection for the initial suppression.

66:66 A character that identifies the preference, publication, or stratum code of the complementary suppression. This is the same identifier used in the tables printed on files 21 through 45.

67:67 The suppression flags for the complementary suppressions that appear in the tables printed on files 21 through 45.

68:73 The row number of the complementary suppression.

74:79 The column number of the complementary suppression.

80:82 The level of the complementary suppression.

83:92 The value of the complementary suppression.

93:102 The capacity of the complementary suppression to protect the initial suppression.

103:112 The number of units flowing through the complementary suppression. This can be used to determine the importance of this cell in protecting the initial suppression. For example, if the initial suppression requires 1000 units of protection and this cell carries a flow of 50, the user can tell this cell is not very important. However, the cell may be very important in protecting other primary suppressions.

113:119 The File 50 counter when this cell was assigned its maximum protection or, in other words, when the cell carried its greatest flow. This field can be very useful. For example, near the end of the run we may be protecting a primary suppression with a value of 1000, and it may require a protection of 900. The user may want to know why the protection is so high. This field on File 50 may indicate the maximum flow for the primary suppression occurred when the File 50 counter was 147. If the suppression pattern for that counter is examined, it should indicate that the primary suppression carried a flow of 900 to protect another initial suppression.

CHAPTER IV: CREATING THE GEOGRAPHIC RELATIONS

Introduction

This part of the documentation explains the procedure to create the geographic relations, which specify how certain geographic areas can be combined to equal other areas. For example, these relations specify how counties add to states, places add to counties, and PMSAs add to CMSAs. There are other relations that are harder to define, such as the Balance of MSAs within a county and the Non-MSA Balance of County being combined to equal a Pseudo-Remainder of County.

These relations are needed for the Business and Industry Censuses, where the rows of the tables refer to SIC codes and the columns refer to geographic areas. As you know, the disclosure analysis programs need input files that describe the additive relations among the columns. When the columns refer to geographic areas, these additive column relations are called geographic relations.

Creating the geographic relations is a very tedious operation that requires great attention to detail. I decided to break the task into small pieces because I thought it would be easier to detect errors and verify the results. You will probably be surprised to see how many computer programs and intermediate data files are involved in the whole process, but I just thought it would be better to do it this way. Stephenie Syer in EPD had to create similar files for the Business Census tabulations but, as I understand, she used only one program to create all of the additive relations. As the old saying goes, there are a lot of ways to skin a cat, even though very few of us have had the pleasure of seeing even one way in action.

In the latter part of 1992, Hoa Nguyen wrote several programs to compare our geographic relations with those created by EPD. This work was extremely valuable. She found errors in both sets of relations and also uncovered errors in the input file from Geography Division.

It may not seem cost effective having both us and EPD producing basically the same geographic relationships, but I liked the idea because it helped to compare our output files to theirs. The errors in our files were very devious, and they could have easily gone undetected until the disclosure analysis programs were run in production and the analysts reviewed the final tables.

SECTION IV-A: General Description of the System of Programs

The information needed to create the geographic relations comes from a file produced by Geography Division. This file is commonly called the "Stub File", and it contains one record for each geographic area. A description and record layout of the file are given in the next section of the documentation.

From our point of view, the Stub File has a few drawbacks. It does not have records for the state parts of cross-over MSAs, the Non-MSA portions of states, or the Pseudo Remainders of Counties. Also the records are very long and hard to read on a printout. For these reasons, we wrote a program GEO-PUB-92 to reformat the file and create the extra records needed for disclosure analysis. The reformatted file is called the Geographic Publication File, and it is described in detail in a later section of the documentation. On this file, each record is assigned a unique 6-digit Geographic Control Number, which will eventually be used as a column number in our tables. A geographic relation will specify how the data for a group of geographic control numbers can be added to equal the data for another geographic control number, which is equivalent to combining geographic areas.

The next program SPLIT-GEO-92 uses the Geographic Publication File to form eleven separate files. These files are actually subsets of the Geographic Publication File, and they make it easier for us to create the eleven types of geographic relations which are listed below.

Type 1: A state is a sum of counties.

Type 2: A county is a sum of places. A place may be a Balance of County or a Pseudo Remainder of County.

Type 3: A place that crosses county boundaries is a sum of its place parts.

Type 4: The state part of a CMSA is a sum of PMSAs or state parts of PMSAs.

Type 5: Outside of New England, an MSA or PMSA is a sum of counties.

Type 6: In New England, an MSA or PMSA is a sum of places. One of these places may be a Balance of MSA in a county.

Type 7: A CMSA is a sum of PMSAs.

Type 8: The state parts of a cross-over metropolitan area (CMSA, MSA, or PMSA) add to the entire metropolitan area.

Type 9: A state is a sum of MSAs, state parts of cross-over MSAs, PMSAs, state parts of cross-over PMSAs, and the Non-MSA portion of the state.

Type 10: In New England states, the Pseudo Remainder of a County is the Balance of County plus any Balance of MSA parts of the county.

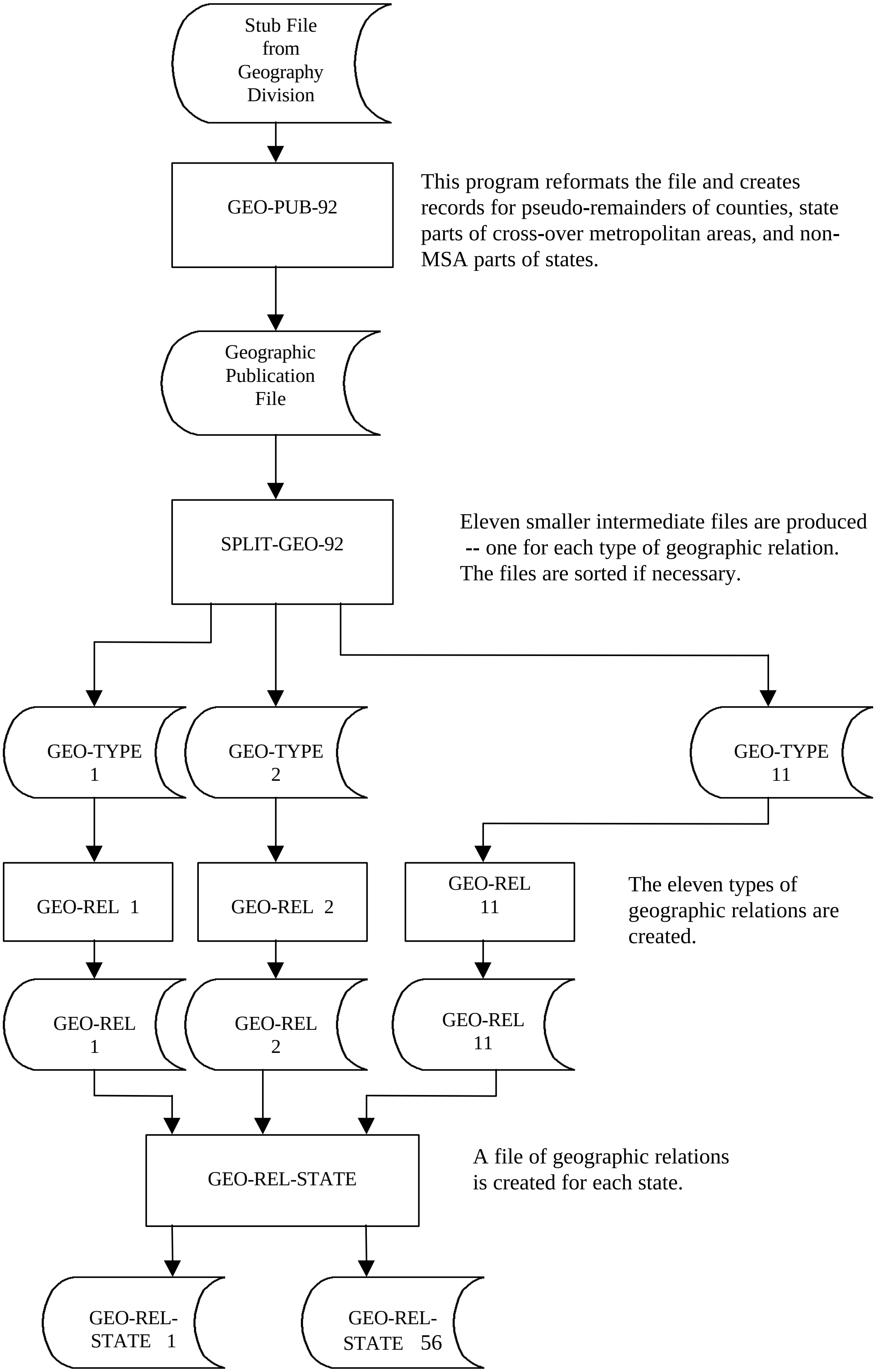
Type 11: A consolidated city is a sum of the places in the consolidated city.

Some of these eleven files need to be sorted so it will be easier to create the geographic relations. After this is done, there are eleven separate programs that use these files to form each type of geographic relation. The records for the relations have codes that tell which states are affected by the relation. Most relations only pertain to one state, but some relations, like the ones for cross-over MSAs, affect more than one state.

After the relations are formed, there is a small program GEO-REL-STATE which can extract all of the relations for a particular state. This program will produce one file of geographic relations for each state. This completes the system of programs.

The flow chart on the next page should make things more clear. Listings of the intermediate files and their corresponding geographic relations are given later in the documentation. I went to a lot of trouble to prepare these listings, so you had better appreciate them.

Creating The Geographic Relations



This program reformats the file and creates records for pseudo-remainders of counties, state parts of cross-over metropolitan areas, and non-MSA parts of states.

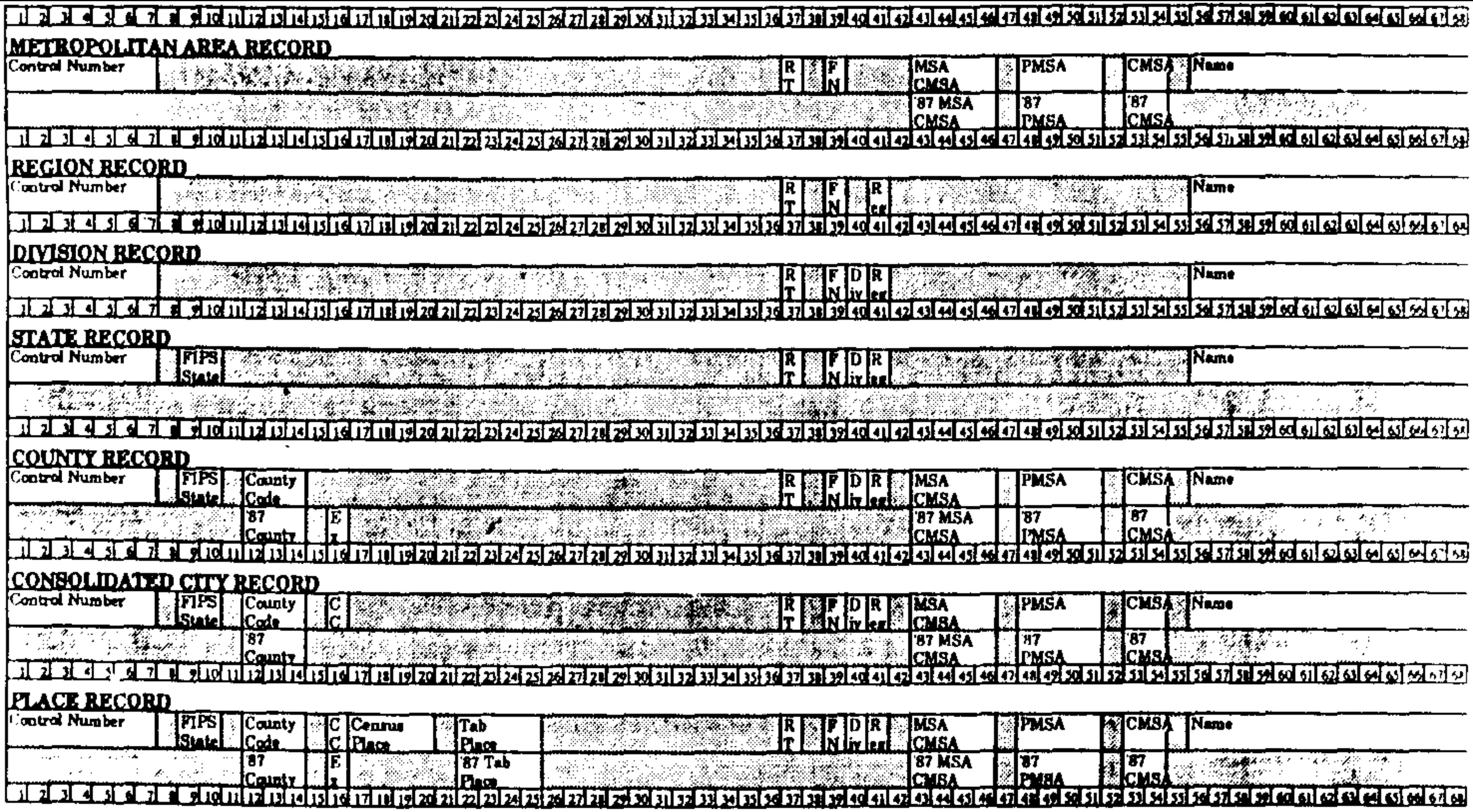
Eleven smaller intermediate files are produced -- one for each type of geographic relation. The files are sorted if necessary.

The eleven types of geographic relations are created.

A file of geographic relations is created for each state.

SECTION IV-B: The Stub File from Geography Division

The Stub File will be created in the Summer of 1993, and Geography Division will undoubtedly provide complete documentation for the file. In 1992 they gave us a preliminary test file, and the record layout for that file is given on the next two pages. There are two records for each geographic area. In my opinion, a listing of this file is difficult to read, which is the main reason we decided to reformat it.



METROPOLITAN AREA RECORD

Control Number	R T	F N	MSA CMSA	PMSA	CMSA	Name
			'87 MSA CMSA	'87 PMSA	'87 CMSA	

REGION RECORD

Control Number	R T	F N	R eg	Name

DIVISION RECORD

Control Number	R T	F N	D iv	R eg	Name

STATE RECORD

Control Number	FIPS State	R T	F N	D iv	R eg	Name

COUNTY RECORD

Control Number	FIPS State	County Code		R T	F N	D iv	R eg	MSA CMSA	PMSA	CMSA	Name
		'87 County	E z					'87 MSA CMSA	'87 PMSA	'87 CMSA	

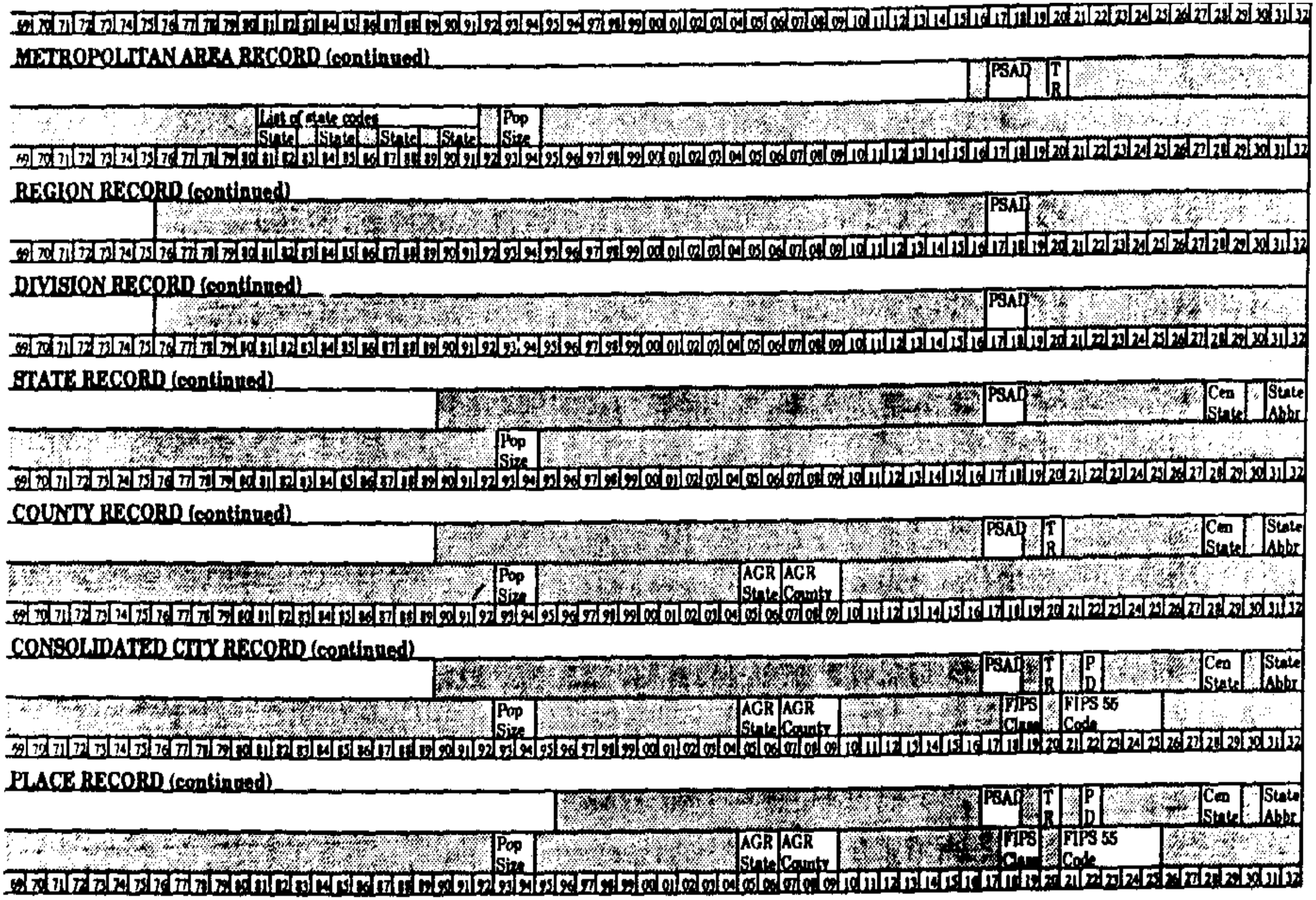
CONSOLIDATED CITY RECORD

Control Number	FIPS State	County Code	C C	R T	F N	D iv	R eg	MSA CMSA	PMSA	CMSA	Name
		'87 County						'87 MSA CMSA	'87 PMSA	'87 CMSA	

PLACE RECORD

Control Number	FIPS State	County Code	C C	Census Place	Tab Place	R T	F N	D iv	R eg	MSA CMSA	PMSA	CMSA	Name
		'87 County	E z		'87 Tab Place					'87 MSA CMSA	'87 PMSA	'87 CMSA	

Sort by FIPS state, FIPS county, consolidated city, tabulation place, record type, division, region, MSA/CMSA, PMSA.



METROPOLITAN AREA RECORD (continued)

PSAD | T R

List of state codes: State | State | State | State — Pop Size

REGION RECORD (continued)

PSAD

DIVISION RECORD (continued)

PSAD

STATE RECORD (continued)

PSAD | Cen State | State Abbr

Pop Size

COUNTY RECORD (continued)

PSAD | T R | Cen State | State Abbr

Pop Size | AGR State | AGR County

CONSOLIDATED CITY RECORD (continued)

PSAD | T R | P D | Cen State | State Abbr

Pop Size | AGR State | AGR County | FIPS Class | FIPS 55 Code

PLACE RECORD (continued)

PSAD | T R | P D | Cen State | State Abbr

Pop Size | AGR State | AGR County | FIPS Class | FIPS 55 Code

SECTION IV-C: The Geographic Publication File

For the 1992 Economic Censuses, we plan to create a Geographic Publication File that has one record for each publication area, including such areas as the Non-MSA portion of a state and Pseudo Remainder of County. This section of the documentation describes the information that will be on the file and explains the procedure to create the file.

The Input File from Geography Division

Geography Division will prepare a Stub File that identifies the geographic areas in the U.S.

The file has these four shortcomings:

1) It has no records for the Non-MSA portion of a state. We need a Non-MSA record for each state except New Jersey.

2) It has only one record for each CMSA, PMSA, or MSA. If a metropolitan area crosses a state line, we need a record for each state part of the area.

3) It has no Pseudo Remainder of County records for the New England counties. A Pseudo Remainder of County is a sum of the Balance of County and the Balance of MSA portions of the county.

4) The records have CMSA/MSA codes and PMSA codes. We need to convert them into a single 4-digit MSA code.

Creating the Geographic Publication File

We have written a computer program GEO-PUB-92 to read the Stub File and re-format the records. The program also creates the Non-MSA and Pseudo Remainder of County records. These are the details of the logic:

1) For each state (except New Jersey), create a Non-MSA record

2) For each metropolitan area, (CMSA, MSA, or PMSA) that crosses state lines, create a record for each state part of the metropolitan area and put the state code on the record.

3) The record for a complete CMSA, MSA, or PMSA has no state code, even if it is contained entirely within a state.

4) Define a new MSA code. This equals the MSA code for the MSAs, and the PMSA code for PMSAs.

5) If a New England county has a Balance of MSA record (9000 < Place code < 9990), create a Pseudo Remainder of County record for the county. The place code will be 9991.

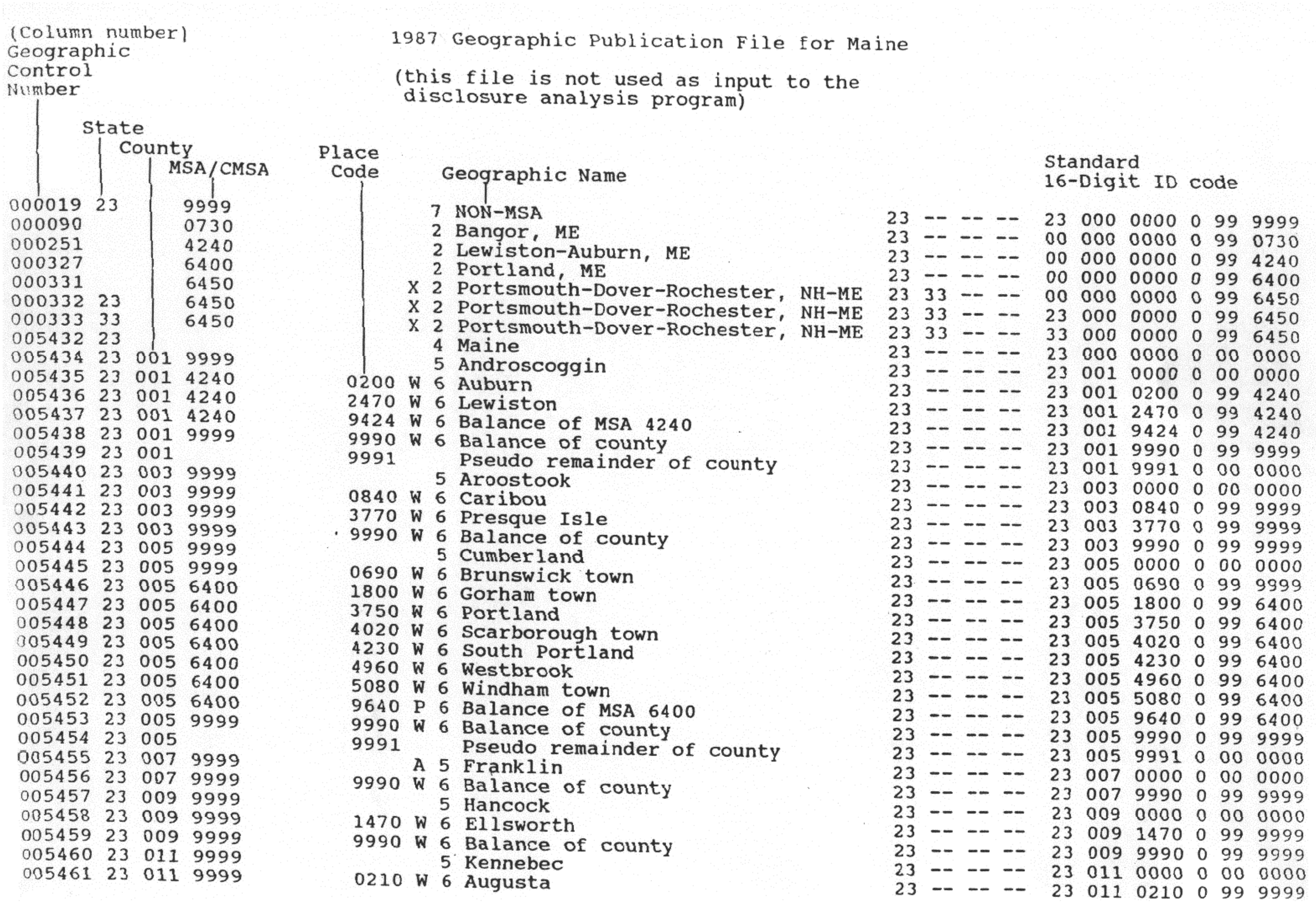
6) A Geographic Control Number is assigned sequentially to each record.

7) The standard 16-digit geographic ID code is put at the end of the record. To make the file easier to read, the fields in the 16-digit code are separated. Some examples of the code are given at the end of this section.

The record layout of the Geographic Publication File is:

Character	2:7	Geographic Control Number	
	9:10	State code	
	12:14	County code	
	16:19	MSA/CMSA code	
	21:22	CSA code	
	24:27	PMSA code	
	29:32	Place code	
	34:34	Territorial Relationship Flag	
	36:36	Record Type	
	38:73	Name of geographic area	
The states	75:76	which may be	
affected by	78:79	this record.	
If the	81:82	record is a cross-over MSA,	
each state	84:85	affected by the MSA is included.	
	88:89	State code	
	91:93	County code	
The	95:98	Place code	Standard
16-Digit	100:100	Consolidated city code	ID code
	102:103	CSA code	
	105:108	New MSA code	

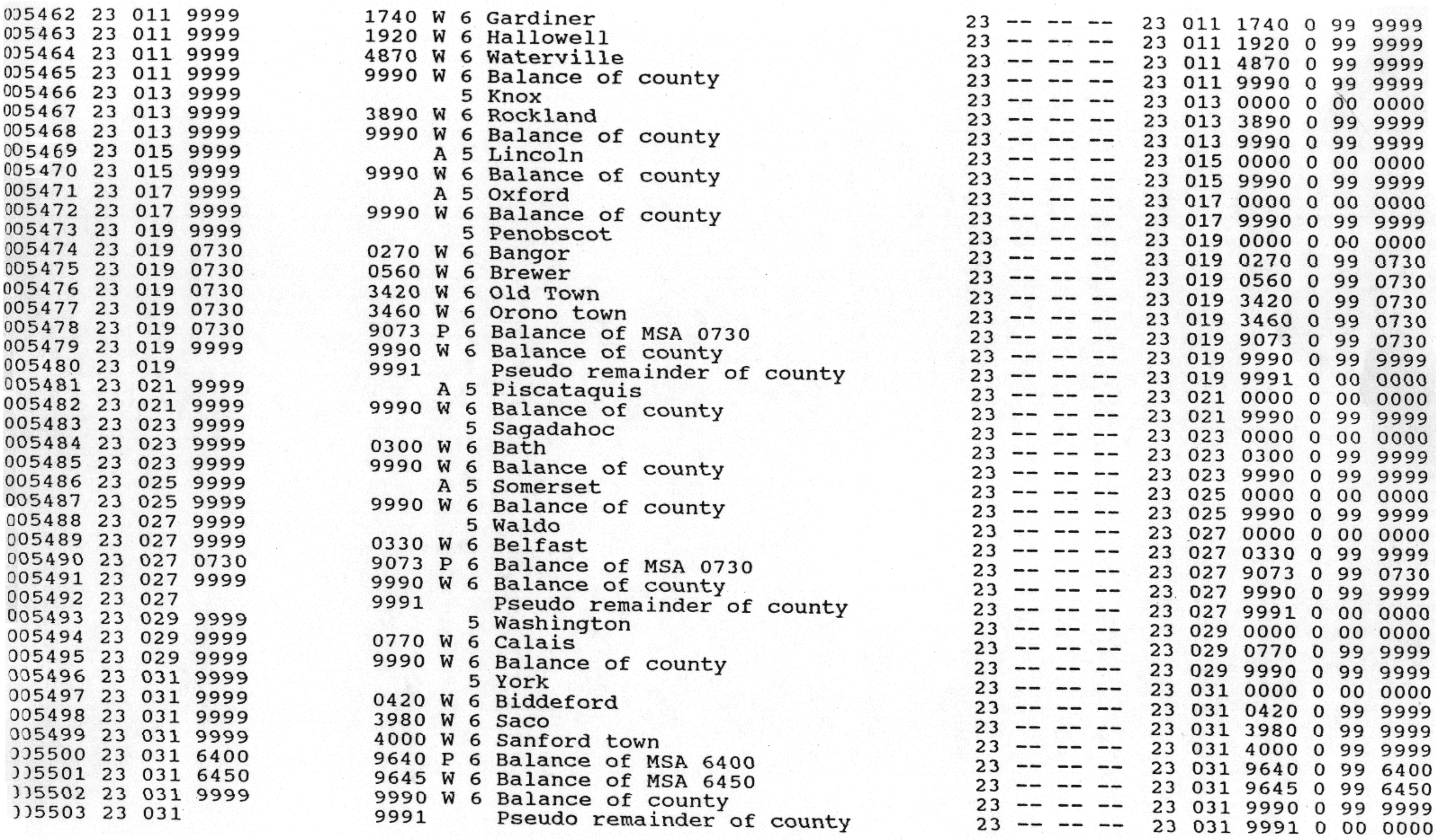
The following two pages contain a listing of a Geographic Publication File for Maine.



```
(Column number)                   1987 Geographic Publication File for Maine
Geographic
Control                           (this file is not used as input to the
Number                             disclosure analysis program)

       State
          County
              MSA/CMSA       Place                                                     Standard
                             Code      Geographic Name                                 16-Digit ID code

000019 23     9999         7 NON-MSA                            23 -- -- --  23 000 0000 0 99 9999
000090        0730         2 Bangor, ME                         23 -- -- --  00 000 0000 0 99 0730
000251        4240         2 Lewiston-Auburn, ME                23 -- -- --  00 000 0000 0 99 4240
000327        6400         2 Portland, ME                       23 -- -- --  00 000 0000 0 99 6400
000331        6450       X 2 Portsmouth-Dover-Rochester, NH-ME  23 33 -- --  00 000 0000 0 99 6450
000332 23     6450       X 2 Portsmouth-Dover-Rochester, NH-ME  23 33 -- --  23 000 0000 0 99 6450
000333 33     6450       X 2 Portsmouth-Dover-Rochester, NH-ME  23 33 -- --  33 000 0000 0 99 6450
005432 23                  4 Maine                              23 -- -- --  23 000 0000 0 00 0000
005434 23 001 9999         5 Androscoggin                       23 -- -- --  23 001 0000 0 00 0000
005435 23 001 4240  0200 W 6 Auburn                             23 -- -- --  23 001 0200 0 99 4240
005436 23 001 4240  2470 W 6 Lewiston                           23 -- -- --  23 001 2470 0 99 4240
005437 23 001 4240  9424 W 6 Balance of MSA 4240                23 -- -- --  23 001 9424 0 99 4240
005438 23 001 9999  9990 W 6 Balance of county                  23 -- -- --  23 001 9990 0 99 9999
005439 23 001       9991     Pseudo remainder of county         23 -- -- --  23 001 9991 0 00 0000
005440 23 003 9999         5 Aroostook                          23 -- -- --  23 003 0000 0 00 0000
005441 23 003 9999  0840 W 6 Caribou                            23 -- -- --  23 003 0840 0 99 9999
005442 23 003 9999  3770 W 6 Presque Isle                       23 -- -- --  23 003 3770 0 99 9999
005443 23 003 9999  9990 W 6 Balance of county                  23 -- -- --  23 003 9990 0 99 9999
005444 23 005 9999         5 Cumberland                         23 -- -- --  23 005 0000 0 00 0000
005445 23 005 9999  0690 W 6 Brunswick town                     23 -- -- --  23 005 0690 0 99 9999
005446 23 005 6400  1800 W 6 Gorham town                        23 -- -- --  23 005 1800 0 99 6400
005447 23 005 6400  3750 W 6 Portland                           23 -- -- --  23 005 3750 0 99 6400
005448 23 005 6400  4020 W 6 Scarborough town                   23 -- -- --  23 005 4020 0 99 6400
005449 23 005 6400  4230 W 6 South Portland                     23 -- -- --  23 005 4230 0 99 6400
005450 23 005 6400  4960 W 6 Westbrook                          23 -- -- --  23 005 4960 0 99 6400
005451 23 005 6400  5080 W 6 Windham town                       23 -- -- --  23 005 5080 0 99 6400
005452 23 005 6400  9640 P 6 Balance of MSA 6400                23 -- -- --  23 005 9640 0 99 6400
005453 23 005 9999  9990 W 6 Balance of county                  23 -- -- --  23 005 9990 0 99 9999
005454 23 005       9991     Pseudo remainder of county         23 -- -- --  23 005 9991 0 00 0000
005455 23 007 9999       A 5 Franklin                           23 -- -- --  23 007 0000 0 00 0000
005456 23 007 9999  9990 W 6 Balance of county                  23 -- -- --  23 007 9990 0 99 9999
005457 23 009 9999         5 Hancock                            23 -- -- --  23 009 0000 0 00 0000
005458 23 009 9999  1470 W 6 Ellsworth                          23 -- -- --  23 009 1470 0 99 9999
005459 23 009 9999  9990 W 6 Balance of county                  23 -- -- --  23 009 9990 0 99 9999
005460 23 011 9999         5 Kennebec                           23 -- -- --  23 011 0000 0 00 0000
005461 23 011 9999  0210 W 6 Augusta                            23 -- -- --  23 011 0210 0 99 9999
```



```
005462 23 011 9999     1740 W 6 Balance                        23 -- -- --   23 011 1740 0 99 9999
```

Please note these features of the File:

1) There are three records pertaining to MSA 6450. The last two records were created when the Geography Division Stub File was reformatted.

   Geog Control Number = 331 (the entire MSA)
   Geog Control Number = 332 (the Maine portion of the MSA)
   Geog Control Number = 333 (the New Hampshire portion of the MSA)

2) There is a Maine Non-MSA record (Geog Control Number = 19)

3) A county has a Balance of MSA record, then there is a Pseudo Remainder of County record, and:

   a) County = Place 1 + Place 2 + ... + Pseudo Remainder of County
   b) Pseudo Remainder of County = Balance of County + Balance of MSA1 + Balance of MSA2 +

The 16-digit geographic ID code

Bill Wester told me that he planned to use a 16-digit ID code to uniquely identify each geographic area. We discussed this in late 1991, and I learned that he wanted the code to be more complex than I had envisioned. For example, I thought the code for a county record should only have a meaningful state and county, and the other fields should be zero. He wanted the code to include the CSA and MSA if the county was outside of New England. My way was simpler, but his definition of the code was more informative, and I saw no good reason to not do it his way. I made up the following examples to show how the code is defined for various types of geographic areas.

CC = Consolidated City

State	County	Place	CC	CSA	MSA	
09	000	0000	0	00	0000	State record for Connecticut
						County record:
39	145	0000	0	99	9999	Scioto County, Ohio (Non-MSA)
39	151	0000	0	99	1320	Stark County, Ohio (in an MSA)
39	153	0000	0	28	0080	Summit County, Ohio (in a PMSA)
09	013	0000	0	00	0000	Tolland Cty., Connecticut (New England)
						Place record:
39	145	3530	0	99	9999	Portsmouth, Ohio (Non-MSA)
39	151	0705	0	99	1320	Canton, Ohio (in an MSA)
39	153	0035	0	28	0080	Akron, Ohio (in a PMSA)
09	013	1190	0	99	9999	Mansfield town, Connecticut (Non-MSA in New England)
23	005	5080	0	99	6400	Windham town, Maine (in a New England MSA)
09	013	2160	0	41	3280	Stafford Springs, Connecticut (in a New England PMSA)
						Balance of County:
23	029	9990	0	99	9999	Balance of Washington County, Maine (In New England, the Balance of County is always Non-MSA)
39	145	9990	0	99	9999	Balance of Scioto County, Ohio (Non-MSA)
39	151	9990	0	99	1320	Balance of Stark County, Ohio (in an MSA)
39	153	9990	0	28	0080	Balance of Summit County, Ohio (in a PMSA)

State	County	Place	CC	CSA	MSA	
23	031	9991	0	00	0000	Pseudo Remainder of York County, Maine
23	031	9645	0	99	6450	Balance of MSA 6450 in York County, Maine
09	003	9328	0	41	3280	Balance of PMSA 3280 in Hartford County, Connecticut
01	000	0500	0	00	0000	Cross-over place: Dothan, Alabama
						<u>County part of a cross-over place:</u>
01	069	0500	0	99	2180	Part of Dothan in Houston County, Alabama (in an MSA)
01	057	1840	0	99	9999	Part of Winfield in Fayette County, Alabama (Non-MSA)
17	031	1420	0	14	1600	Part of Deerfield in Cook County, Illinois (in a PMSA)
00	000	0000	0	56	0000	CSA contained in a state: Miami-Fort Lauderdale, Florida
00	000	0000	0	21	0000	Cross-over CSA: Cincinnati-Hamilton OH-KY-IND
39	000	0000	0	21	0000	State part of cross-over CSA: the Ohio portion of CSA 21
00	000	0000	0	99	9260	MSA contained in a state: Yakima, Washington
00	000	0000	0	99	4920	Cross-over MSA: Memphis, TN-AR-MS
28	000	0000	0	99	4920	State part of cross-over MSA: Mississippi portion of Memphis MSA
00	000	0000	0	07	1120	PMSA contained in a state: Boston, Massachusetts
00	000	0000	0	07	4560	Cross-over PMSA: Lowell, MA-NH
33	000	0000	0	07	4560	State part of cross-over PMSA: New Hampshire portion of Lowell PMSA
28	888	0000	0	99	9999	Offshore area of Mississippi (state 28)
12	031	0000	1	99	3600	Jacksonville consolidated city
12	031	1465	2	99	3600	Neptune Beach, inside the Jacksonville consolidated city
00	000	0000	0	02	0000	Midwest Region
00	000	0000	0	03	0005	South Atlantic Division
00	000	0000	0	00	0000	United States of America

SECTION IV-D: An Example of a Geographic Relation

After we create the Geographic Publication File with one record for each publication area, we need to form the set of geographic relations that define how the areas add up. Each record on the Geographic Publication File has a unique Geographic Control Number, and a geographic relation defines how the data for a group of control numbers can be added to equal the data for another control number.

For example, the following Geographic Control Numbers were assigned:

5483 = Sagadahoc County
5484 = the city of Bath
5485 = Balance of Sagadahoc County

The corresponding geographic relation states that:

5483 = 5484 + 5485

SECTION IV-E: The Intermediate Files and the Geographic Relations

The Geographic Publication File is divided into eleven smaller intermediate files that make it easier to create the geographic relations. In this section of the documentation, I will give some partial listings of the intermediate files and the geographic relations that are created from them. There is one intermediate file for each type of geographic relation, and they have the same record layout as the Geographic Publication File. The record layout for the geographic relations is given below. It is basically the same as the layout for the column relations defined in Section II-D.

Character	2:3 5:6 8:9 11:12	The states affected by this relation. A relation for a cross-over metropolitan area like the Philadelphia CMSA may affect as many as four states.
Character	14:15	The type of geographic relation. These types were defined in section IV-A.
Character	17:20	Relation number (assigned sequentially)
Character	23:23	The record count for the relation. Some relations are so long they need more than one record.
Character	25:93	The list of Geographic Control Numbers in the relation. Each number is stored in 6 digits, with a space in between. The first Geographic Control Number is a sum of all the rest.

Type 1 A state is a sum of counties.

This intermediate file contains state and county records. For example, these are the records for Delaware and Hawaii. The format is the same as the Geographic Publication File, except that the last part of the record is truncated because the page is too narrow.

Geog Control Number	State	County	MSA/CMSA	CSA	PMSA	Place	Record Type	Geographic Name
002064	10						4	Delaware
002068	10	001	9999				5	Kent
002073	10	003	6162	77	9160		5	New Castle
002081	10	005	9999				5	Sussex
002938	15						4	Hawaii
002940	15	001	9999				5	Hawaii
002944	15	003	3320				5	Honolulu
002975	15	007	9999				5	Kauai
002982	15	009	9999				5	Maui

The geographic relations show how the qeoqraphic control numbers for the counties add to the geographic control numbers for the state.

```
10 -- -- -- 01 0001        1 002064 002068 002073 002081
15 -- -- -- 01 0002        1 002938 002940 002944 002975 002982
```

Type 2: A county is a sum of places.

This intermediate file contains county and place records. A place record may also refer to a Balance of County or a Pseudo Remainder of County. These are some of the records for Maine.

```
005434 23  001 9999                          5 Androscoggin
005435 23  001 4240                0200      6 Auburn
005436 23  001 4240                2470      6 Lewiston
005439 23  001                     9991        Pseudo remainder of county
005440 23  003 9999                          5 Aroostook
005441 23  003 9999                0840      6 Caribou
005442 23  003 9999                3770      6 Presque Isle
005443 23  003 9999                9990      6 Balance of county
005496 23  031 9999                          5 York
005497 23  031 9999                0420      6 Biddeford
005498 23  031 9999                3980      6 Saco
005499 23  031 9999                4000      6 Sanford town
005503 23  031 9991                            Pseudo remainder of county
```

These are the geographic relations for this set of records

```
23 -- -- -- 02 0001 01 005434 005435 005436 005439
23 -- -- -- 02 0002 01 005440 005441 005442 005443
23 -- -- -- 02 0003 01 005496 005497 005498 005499 005503
```

Type 3: A place that crosses county boundaries is a sum of its parts within each county.

This intermediate file contains records for cross-over places and their parts within each county. These are some records for Illinois.

Note that the cross-over places have a blank county code. For a small place like Barrington Hills to intersect four counties in nothing short of amazing.

```
003126  17                           0050     6 Algonquin
003484  17   089   1602   14   0620  0050     6 Algonquin (part)
003580  17   111   1602   14   1600  0050     6 Algonquin (part)
003127  17                           0280     6 Aurora
003371  17   043   1602   14   1600  0280     6 Aurora (part)
003485  17   089   1602   14   0620  0280     6 Aurora (part)
003129  17                           0323     6 Barrington Hills
003227  17   031   1602   14   1600  0323     6 Barrington Hills (part)
003486  17   089   1602   14   0620  0323     6 Barrington Hills (part)
003521  17   097   1602   14   3965  0323     6 Barrington Hills (part)
003581  17   111   1602   14   1600  0323     6 Barrington Hills (part)
```

These are the geographic relations for this set of records.

```
17   -- -- --   03   0001   1   003126  003484        003580
17   -- -- --   03   0002   1   003127  003371        003485
17   -- -- --   03   0003   1   003129  003227        003486     003521     003581
```

Type 4: The state part of a CMSA is a sum of PMSAs and state parts of PMSAS.

If a CMSA crosses state lines, this intermediate file contains records for the state parts of the CMSA and the PMSAs inside of it. The file does not contain a record for the entire CMSA. These are some records for two cross-over CMSAS.

```
000129    17     1602     14               2 Chicago-Gary-Lake County, IL-IN-WI
000430    17     1602     14     0620      3 Aurora-Elgin, IL
000441    17     1602     14     1600      3 Chicago, IL
000462    17     1602     14     3690      3 Joliet, IL
000464    17     1602     14     3965      3 Lake County, IL
000130    18     1602     14               2 Chicago-Gary-Lake County, IL-IN-WI
000457    18     1602     14     2960      3 Gary-Hammond, IN
000131    55     1602     14               2 Chicago-Gary-Lake County, IL-IN-WI
000463    55     1602     14     3800      3 Kenosha, WI
000134    18     1642     21               2 Cincinnati-Hamilton, OH-KY-IN
000443    18     1642     21     1640      3 Cincinnati, OH-KY-IN
000135    21     1642     21               2 Cincinnati-Hamilton, OH-KY-IN
000444    21     1642     21     1640      3 Cincinnati, OH-KY-IN
000136    39     1642     21               2 Cincinnati-Hamilton, OH-KY-IN
000445    39     1642     21     1640      3 Cincinnati, OH-KY-IN
000458    39     1642     21     3200      3 Hamilton-Middletown, OH
```

These are the geographic relations for this set of records. Note that each of the relations affects three states. When files of geographic relations are created for each state, these relations will be in three of the files.

```
17 18  55 – 04        0001      1      000129      000430     000441     000462      000464
17 18  55 – 04        0002      1      000130      000457
17 18  55 – 04        0003      1      000131      000463
18 21  39 – 04        0004      1      000134      000443
18 21  39 – 04        0005      1      000135      000444
18 21  39 – 04        0006      1      000136      000445     000458
```

Type 5: Outside of New England, an MSA or PMSA is a sum of counties.

For the states outside of New England, this intermediate file contains records for MSAS, PMSAS, and counties. If an MSA or PMSA crosses state lines, the file only contains the records for the state parts.
The file also has records for the Non-MSA portions of the states. These are some records for an MSA in Alabama, the Non-MSA portion of Hawaii, and the two state parts of the St. Louis MSA.

```
000285               5240                              2  Montgomery, AL
000534   01   001    5240                              5  Autauga
000627   01   051    5240                              5  Elmore
000753   01   101    5240                              5  Montgomery
000011   15          9999                              7  NON-MSA
002940   15   001    9999                              5  Hawaii
002975   15   007    9999                              5  Kauai
002982   15   009    9999                              5  Maui
000356   17          7040                              2  St. Louis, MO-IL
003213   17   027    7040                              5  Clinton
003475   17   083    7040                              5  Jersey
003608   17   119    7040                              5  Madison
003642   17   133    7040                              5  Monroe
003703   17   163    7040                              5  St. Claire
000357   29          7040                              2  St. Louis, MO-IL
006996   29   071    7040                              5  Franklin
007051   29   099    7040                              5  Jefferson
007189   29   183    7040                              5  St. Charles
007207   29   189    7040                              5  St. Louis
007304   29   510    7040                              5  St. Louis*
```

These are the geographic relations for this set of records

```
01 -- -- -- 05    0001   1   000285   000534   000627   000753
15 -- -- -- 05    0002   1   000011   002940   002975   002982
17 -- -- -- 05    0003   1   000356   003213   003475   003608   003642   003703
29 -- -- -- 05    0004   1   000357   006996   007051   007189   007207   007304
```

Type 6 : For the New England states, an MSA or PMSA is a sum of places

This intermediate file contains records for MSAS, PMSAS, and places. A place record may refer to a Balance of County or a Balance of MSA. If an MSA or PMSA crosses state lines, the file only contains the records for the state parts. The file also has records for the Non-MSA portions of states. The records below are for an MSA in Maine, an MSA that crosses from Massachusetts to New Hampshire, and the Non-MSA portion of Rhode island.

```
000090              0730                               2 Bangor, ME
005474   23   019   0730                 0270          6 Bangor
005475   23   019   0730                 0560          6 Brewer
005476   23   019   0730                 3420          6 Old Town
005477   23   019   0730                 3460          6 Orono town
005478   23   019   0730                 9073          6 Balance of MSA 0730
005490   23   027   0730                 9073          6 Balance of MSA 0730
000467   33         1122   07   4160                   3 Lawrence-Haverhill, MA-NH
007766   33   015   1122   07   4160     0630          6 Derry town
007771   33   015   1122   07   4160     2280          6 Salem town
007772   33   015   1122   07   4160     9416          6 Balance of MSA 4160
```

```
000038    44              9999          7                          NON-MSA
010921    44     005      9999 0280     6                          Middletown town
010922    44     005      9999 0310     6                          Newport
010923    44     005      9999 0390     6                          Portsmouth town
010919    44     003      9999 9990     6                          Balance of county
010951    44     009      9999 9990     6                          Balance of county
```

These are the geographic relations for this set of records.

```
23 -- -- --   06   0001    1   000090    005474    005475 005476   005477      005478     005490
33 -- -- --   06   0002    1   000467    007766    007771 007772
44 -- -- --   06   0003    1   000038    010921    010922 010923   010919      010951
```

Type 7: A CMSA is a sum of PMSAs.

This intermediate file contains records for complete CMSAs and PMSAs If the CMSA or PMSA crosses state lines, the records for the state parts are not included on the file. These records are examples from three CMSAS.

```
000104                    1122   07                   2   Boston-Lawrence-Salem, MA-NH
000433                    1122   07   1120            3   Boston, MA
000438                    1122   07   1200            3   Brockton, MA
000465                    1122   07   4160            3   Lawrence-Haverhill, MA-NH
000470                    1122   07   4560            3   Lowell, MA-NH
000478                    1122   07   5350            3   Nashua, NH
000499                    1122   07   7090            3   Salem-Gloucester, MA
000257                    4472   49                   2   Los Angeles-Anaheim-Riverside, CA
000428                    4472   49   0360            3   Anaheim-Santa Ana, CA
000469                    4472   49   4480            3   Los Angeles-Long Beach, CA
000487                    4472   49   6000            3   Oxnard-Ventura, CA
000498                    4472   49   6780            3   Riverside-San Bernardino, CA
000328                    6442   79                   2   Portland-Vancouver, OR-WA
000495                    6442   79   6440            3   Portland, OR
000510                    6442   79   8725            3   Vancouver, WA
```

These are the geographic relations for this set of records

```
25  33 -- -- 07 0001  1       000104  000433        000438     000465      000470    000478     000499
06  -- -- -- 07 0002  1       000257  000428        000469     000487      000498
41  53 -- -- 07 0003  1       000328  000495        000510
```

Type 8: A metropolitan area (CMSA, MSA, or PMSA) that crosses state lines is a sum of its state parts.

This intermediate file contains records for the cross-over metropolitan areas and all of their state parts. These are records from the Cincinnati CMSA and the Washington MSA.

```
000133              1642  21                              2   Cincinnati-Hamilton, OH-KY-IN
000134     18       1642  21                              2   Cincinnati-Hamilton, OH-KY-IN
000135     21       1642  21                              2   Cincinnati-Hamilton, OH-KY-IN
000136     39       1642  21                              2   Cincinnati-Hamilton, OH-KY-IN
000442              1642  21     1640                     3   Cincinnati, OH-KY-IN
000443     18       1642  21     1640                     3   Cincinnati, OH-KY-IN
000444     21       1642  21     1640                     3   Cincinnati, OH-KY-IN
000445     39       1642  21     1640                     3   Cincinnati, OH-KY-IN
000407              8840                                  2   Washington, DC-MD-VA
```

000408	11	8840	2	Washington, DC-MD-VA
000409	24	8840	2	Washington, DC-MD-VA
000410	51	8840	2	Washington, DC-MD-VA

These are the geographic relations for this set of records. Note that each of these relations affects three states.

18	21	39	--	08	0001	1	000133	000134	000135	000136
I8	21	39	--	08	0002	1	000442	000443	000444	000445
11	24	51	--	08	0003	1	000407	000408	000409	000410

Type 9: A state is a sum of MSAS, state parts of MSAS, PMSAS, state parts of PMSAS, and a Non-MSA total.

This intermediate file would contain these records for Maryland, New Hampshire, and Mississippi.

006208	24				3	Maryland
000089	24	0720	99	9999	2	Baltimore, MD
000176	24	1900	99	9999	2	Cumberland, MD-WV
000239	24	3180	99	9999	2	Hagerstown, MD
000500	24	8840	99	9999	2	Washington, DC-MD-VA
000390	24	6162	77	9160	2	Wilmington, DE-NJ-MD
000020	24			9999	7	NON-MSA
007595	28				3	Mississippi
000097	28	0920	99	9999	2	Biloxi-Gulfport, MS
000261	28	3560	99	9999	2	Jackson, MS
000318	28	4920	99	9999	2	Memphis, TN-AR-MS
000375	28	6025	99	9999	2	Pascagoula, MS
000024	28	9999			7	NON-MSA
008744	33				3	New Hampshire
000111	33	1122	07	4160	2	Lawrence-Haverhill, MA-NH
000114	33	1122	07	4560	2	Lowell, MA-NH
000310	33	4760	99	9999	2	Manchester, NH
000115	33	1122	07	5350	2	Nashua, NH
000407	33	6450	99	9999	2	Portsmouth-Dover-Rochester, NH-ME
000029	33	9999			7	NON-MSA

These are the geographic relations for this set of records.

24	00	00 – 09	0001	1	006208	000089	000176	000239	000500	000390	000020
28	00	00 – 09	0002	1	007595	000097	000261	000318	000375	000024	
33	00	00 – 09	0003	1	008744	000111	000114	000310	000115	000407	000029

Type 10: In New England states, a Psuedo Remainder of County is equal to the Balance of County plus any Balance of MSAs in the county.

This intermediate file will contain these records for Massachusetts.

005616	25	003				9991		Pseudo remainder of county
005614	25	003	6320			9632	6	Balance of MSA 6320
005615	25	003	9999			9990	6	Balance of county
005636	25	005				9991		Pseudo remainder of county
005632	25	005	1122	07	1120	9112	6	Balance of MSA 1120

005633	25	005	5400			9540	6	Balance of MSA 5400	
005634	25	005	6482	80	6060	9606	6	Balance of MSA 6060	
005635	25	005			9999	9990	6	Balance of county	
005661	25	009				9991		Pseudo remainder of county	
005658	25	009	1122	07	1120	9112	6	Balance of MSA	1120
005659	25	009	1122	07	4160	9416	6	Balance of MSA	4160
005660	25	009	1122	07	7090	9709	6	Balance of MSA	7090

These are the geographic relations for this set of records.

25	--	--	--	10	0001	1	005616	005614	005615		
25	--	--	--	10	0002	1	005636	005632	005633	005634	005635
25	--	--	--	10	0003	1	005661	005658	005659	005660	

Type 11: A consolidated city is the sum of the places within it.

In the partial listing of this intermediate file, the consolidated cities have a blank place code. On the Geographic Publication File they can be distinguished from the county records because they have a record type of 5.

002339	12	031	3600	99			5	Jacksonville
002340	12	031	3600	99	9999	0055	6	Atlantic Beach
002341	12	031	3600	99	9999	1002	6	Jacksonville Beach
002342	12	031	3600	99	9999	1003	6	Jacksonville city (balance)
002343	12	031	3600	99	9999	1465	6	Neptune Beach
004250	18	097	3480	99	9999		5	Indianapolis
004252	18	097	3480	99	9999	0580	6	Cumberland (part)
004253	18	097	3480	99	9999	1145	5	Indianapolis city (balance)
012845	47	037	5360	99	9999		5	Nashville-Davidson
012846	47	037	5360	99	9999	0105	6	Belle Meade
012847	47	037	5360	99	9999	0513	6	Forest Hills
012848	47	037	5360	99	9999	0575	6	Goodlettsville (part)
012849	47	037	5360	99	9999	1016	6	Nashville-Davidson (balance)
012850	47	037	5360	99	9999	1070	6	Oak Hill

These are the geographic relations for this set of records.

12	--	--	--	11	0001	1	002339	002340	002341	002342	002343	
18	--	--	--	11	0002	1	004250	004252	004253			
47	--	--	--	11	0003	1	012845	012846	012847	012848	012849	012850

CHAPTER V: Applications of the Disclosure Analysis Programs

Introduction

The first version of this documentation was written in the Fall of l992. By that time, the disclosure analysis programs had already been run in production for the l990 County Business Patterns. The programs were also being used to do the disclosure analysis for some special tables ECSD was preparing for the Small Business Administration. In this chapter of the documentation I will describe our work on these two projects. Learning about experiences may help you in planning future disclosure analysis applications.

SECTION V-A: County Business Patterns

This was a good project for us to test the disclosure analysis programs because it included both 2-D and 3-D tables, and some of them were quite large. We began to plan the system and develop the input files in January of 1992, and the production work was started in June. The project involved three divisions - EPD created the input files, SRD ran the disclosure analysis, ECSD reviewed the outputs, and EPD produced the final publication tables. The processing of the 2-D and 3-D tables is summarized in the remainder of this section.

<u>2-D Tables</u>

The County Business Patterns publications have tables that give employee payroll totals for 1137 SIC codes at the state and county level. This could be viewed as one large table for each state, where the rows refer to SIC codes and the columns refer to counties. This table would be absolutely huge for a state like Texas, but we could probably run it if we divided the rows into subgroups. Unfortunately, this method was not available in the program when we began getting the system ready for production in early 1992.

As an alternative, we decided to let the counties in the state be the rows in the table and to have the SIC codes serve as the columns. Each of the 248 additive SIC relations defined a column relation.

It worked out pretty well having the SIC codes as the columns of the tables. Most of the relations had less than ten columns, so we could fit those tables on one page of the printout. This made the tables easier for the analysts to review and verify.

Sometimes the published total for a group of SIC codes was greater than the sum of the published values for the individual codes. This occurred because some business establishments were included in the overall total but were not included in one of the component SIC codes. This destroyed the additivity in the SIC relations, but I made a few relatively easy modifications to the program to compensate for it.

EPD created some good test files in March, which allowed us to test the disclosure analysis programs well before production. The analysts in ECSD reviewed the output tables very thoroughly, and Jim Bowman detected a definite case of oversuppression. When a primary suppression had only one respondent, every other one-respondent cell was given a capacity of zero, which eventually caused the program to needlessly suppress other cells.

I made a small change in the program to increase the cell capacities of the one-respondent cells, and it resulted in about 9% less total value suppressed.

The production work was run during June and July, and most of it went very smoothly. We did one disclosure analysis run for each state. The smaller states ran in a couple minutes of computer time on a VAX 8530, but Texas needed 5 1/2 hours to complete. It took about 45 hours of computer CPU time to run all 50 states.

3-D Tables

The publications include tables that give the total payroll of business firms for nine different employment size classes. This data is provided for each of the 1137 SIC codes at the state and U.S. level. For example, the tables give the total payroll of retail hardware stores with 20-49 employees in Ohio.

This data was grouped into one large 3-D table. The table had 52 rows that referred to the states plus the District of Columbia and U.S. total, 1137 columns that corresponded to the SIC codes, and 10 levels in the third dimension which represented a total level plus the nine employee size classes. Where the disclosure analysis was run on this table, the additive SIC relations were used to define column relations, just like we did in the 2-D disclosure analysis.

The cells in the first level gave the total payroll of firms for each SIC code within each state. These cells also appeared in the 2-D tables, and if a cell was suppressed in the 2-D tables we had to make sure the cell was protected in the 3-D table as well. Just as important, we had to be careful to not add any new suppressions into this group of cells because we could not guarantee any new suppressions would be protected in the 2-D tables. In other words, the cells had to be frozen after the 2-D disclosure analysis was completed, and no new suppressions could be introduced into these cells when the 3-D program was run. This was accomplished by assigning a preference code of 9 to the input file records that represented these cells.

Before a 3-D input file could be formed, the 2-D disclosure analysis had to be run for each state. The records for the state totals were extracted from the 2-D output files, the preference codes were set to 9, and the records were inserted into the file that would be an input to the 3-D disclosure analysis program. The data for the nine employment size classes were also put into the same file.

After I corrected an embarrassing error in one of the input parameters, the 3-D disclosure analysis program ran smoothly in production. I was thrilled that it needed only 5 1/2 hours of CPU time.

Probably one of the main things to be gained from the County Business Patterns experience is to remind us that we all have to do our part for a project to be successful. The programmers in EPD did a good job preparing input files, we did a good job running the disclosure analysis, and the analysts in ECSD did a good job reviewing the output tables. None of us should feel that our work was of greater or lesser importance than the work of others. If any of us had not done our job properly, the whole project would have been a flop.

SECTION V-B: Small Business Administration

In the latter part of l992 we ran the disclosure analysis for some tables the Census Bureau was preparing for the Small Business Administration (SBA). Probably the most interesting part of this project is the way we had to restructure their tables to fit the constraints of the disclosure analysis program. All of their tables were three dimensional. The first table had 10 rows that added to a total, and had 9 columns with a fairly complicated hierarchical structure. There was one table for each state and the District of Columbia, and a table for the U.S. total.

To make this table fit into the disclosure analysis system, we decided to let the hierarchical columns become the rows, the states became the columns, and the rows became the levels in the third dimension. At first the analysts were puzzled that we had to go through such gyrations, but they caught on very quickly, and asked the programmers to give us data files in the exact format we requested. After we received the files, the disclosure analysis ran without any problems.

The second set of tables had the same hierarchical column structure, but there were 33 rows and their additive relations were not hierarchical. The tables were also produced for the states and U.S. total.

For the purpose of disclosure analysis, we again converted the columns into rows. The rows of the publication table became the columns of our disclosure analysis table, and each additive relation among the rows in the publication table became a column relation in our table. This left the states to be the third dimension, which meant our tables had 52 levels in the third dimension, more than we ever had previously.

When the programmers were preparing the input files, they were a little unhappy to learn they could not use a normal state code to define the level in the third dimension. The program expects the levels to have codes 1,2,3,... with level 1 being the total, so they had to convert all of their state codes. Maybe it was all for the best, because if my program had met the customer's needs too well, I might have been asked to teach a seminar on CQM.

When the disclosure analysis work was being planned, the analysts realized that many cells in the second set of tables were also in the first set of tables. This meant that any of these cells suppressed in the first set of tables had to be protected in the second set of tables. Just like we did in the CBP tables, these cells had to be frozen by assigning a preference code of 9 to their records on the input file. This was no problem for me because the disclosure analysis program was designed to handle cases like this, but I think it was an unwelcome surprise to the programmers who were creating the input files.

All of the data files for this project were created by the computer programmers and we had very little contact with them. On the County Business Patterns disclosure analysis, we worked closely with the programmers who were producing the data files, but on the SBA project we primarily dealt with the statisticians in ECSD, who passed on the information to the programmers within their division. In my opinion, this arrangement worked very well. It was probably harder for the statisticians because they had to be involved in all of the details, but it kept them fully informed.