Form F20-8102

IBM Data Processing Techniques

Analysis

Decision Tables–A Systems Analysis and Documentation Technique

The basic concepts of decision tables and a minimum set of conventions for their use in systems analysis, procedure design, and documentation are described. The purpose of such tables is to provide information in a concise format that is easy to read and understand. The tabular approach is used to express complex decision logic in a manner that encourages the analyst to reduce a problem to its simplest form by arranging and presenting logical alternatives under various conditions. The resultant tables present a system's logic in a concise manner that is easy to visualize and grasp. While the concepts in the text are presented on a level for comprehension by students in basic computer courses, the techniques are applicable at all levels of sophistication by everyone in a data processing environment.

The decision table (Form no. X28-1630) is a convenience in standardizing documentation.

Form F20-8102

IBM Data Processing Techniques

Analysis

Decision Tables–A Systems Analysis and Documentation Technique

The basic concepts of decision tables and a minimum set of conventions for their use in systems analysis, procedure design, and documentation are described. The purpose of such tables is to provide information in a concise format that is easy to read and understand. The tabular approach is used to express complex decision logic in a manner that encourages the analyst to reduce a problem to its simplest form by arranging and presenting logical alternatives under various conditions. The resultant tables present a system's logic in a concise manner that is easy to visualize and grasp. While the concepts in the text are presented on a level for comprehension by students in basic computer courses, the techniques are applicable at all levels of sophistication by everyone in a data processing environment.

The decision table (Form no. X28-1630) is a convenience in standardizing documentation.

Note

While the concepts in this text are presented on a level for comprehension by students in basic computer courses, the techniques are applicable at all levels of sophistication by the systems analyst, the programmer and all those in a data processing environment for developing, implementing and documenting systems.

Copies of this and other IBM publications can be obtained through IBM Branch Offices. Address comments concerning the contents of this publication to IBM, Technical Publications Department, 112 East Post Road, White Plains, N. Y. 10601

INTRODUCTION

The use of data processing equipment has focused attention upon the necessity for an orderly representation of information flow from elementary decisions to final actions. The sequence in which operations are to be executed must be precisely stated and the exceptions to normal processing must be identified. Most operations are so elementary that a large number of them must be combined properly in order to carry out a meaningful data processing task. The data itself, as well as the operations to be performed upon it, are the information which constitutes the system.

Decision tables are a means of bringing together and presenting this related information to express complex decision logic in a way that is easy to visualize and follow. By presenting logical alternative courses of action under various combinations of conditions, a decision table enables the analyst to think through a problem and present its solution effectively. He is encouraged to reduce the documentation to its simplest form. The basic objective of decision tables is to arrange and present a system's logic in such a way that its meaning can readily be grasped.

The term "data processing" has become associated with all recordkeeping, decision-making and problem-solving operations in business, mathematics and the sciences. In each of these fields, decision tables can be used independently of, or to complement, flow charts and block diagrams.

Decision tables can be used effectively for system analysis, procedure design and documentation. Their use expedites and simplifies the time-consuming functions of problem definition and system analysis. Once the system is established, it is easy to maintain, and the documentation is easy to change.

The value of decision tables is independent of the equipment used, and the tables may, in fact, be used by people with little knowledge of equipment. An understanding of the tables shown in this text can, in itself, provide the student with an insight into the procedures followed in using computers. The masters of the art of data processing, on the other hand, are provided with a different approach to a problem and its solution than that given by other methods.

This manual describes a minimum set of decision table conventions to be used in defining systems or portions of systems. The principles involved can lead to standardization by the individual user, which in turn can lead to standardization within the data processing community.

This description of the structure and interpretation of decision tables should demonstrate the value of the technique, since the decision tables speak for themselves, as indeed all documentation systems are intended to do.

BASIC CONCEPTS OF DECISION TABLES

Tables of Information

Tables as a means of displaying information are used for such diverse purposes as railroad timetables, stock market quotations, logarithms and racetrack results. An example from a United States Government publication on tabular presentation techniques is shown:

Simple form; additive box and stub.

Division	Total number of farms	Under 30 acres	30 to 49 acres	50 to 99 acres	100 to 179 acres	180 to 259 acres	260 to 499 acres	500 to 999 acres	1,000 acres and over
United States	6,096,799	1,519,373	767,289	1,291,048	1,309,741	486,336	459,787	262,684	100,531
New England	135,190	41,674	14,662	29,442	29,688	10,976	7,836	1,874	378
Middle Atlantic	348,100	84,573	34,650	91,677	97,449	28,158	14,424	1,646	309
East North Central	1,006,085	164,996	104,780	268,023	295,824	105,299	58,601	6,895	877
West North Central	1,000,574	100,321	62,900	153,998	323,965	156,696	198,183	67,887	27,744
South Atlantic	1,019,451	319,359	173,968	263,000	164,778	53,270	37,394	11,367	4,217
East South Central	1,023,349	379,072	181,360	232,463	148,731	52,755	29,348	7,732	2,830
West South Central	964,370	260,663	138,939	190,166	194,101	64,941	71,448	28,618	19,474
Mountain	233,497	67,152	18,758	26,833	34,440	12,786	30,663	27,557	33,268
Pacific	276,173	131,553	35,356	35,348	26,964	10,543	14,900	10,109	11,321

Figure 1. A Table of Information

The primary purpose of such tabular presentation is to provide information in a format that is concise yet easy to read and to understand.

Similarly, some orderly arrangement of data is necessary for the logical solution of any complex problem. At first glance, for example, the puzzle below does not lend itself to easy solution:

> Andy dislikes the catcher. Ed's sister is engaged to the second baseman. The center fielder is taller than the right fielder. Harry and the third baseman live in the same building. Paul and Allen each won $20 from the pitcher at pinochle. Ed and the outfielders play poker during their free time. The pitcher's wife is the third baseman's sister. All the battery and infield, except Allen, Harry and Andy, are shorter than Sam. Paul, Andy and the shortstop lost $50 each at the racetrack. Paul, Harry, Bill and the catcher took a trouncing from the second baseman at pool. Sam is undergoing a divorce suit. The catcher and the third baseman each have two children. Ed, Paul, Jerry, the right-fielder and the center fielder are bachelors. The others are married. The shortstop, the third baseman and Bill each cleaned up $100 betting on the fight. One of the outfielders is either Mike or Andy. Jerry is taller than Bill. Mike is shorter than Bill. Each of them is heavier than the third baseman. With these facts determine the names of the men playing the various positions on the baseball team.

The puzzle is, however, far less complex than it appears when the problem is broken down into a series of steps based on the facts which determine the solution. For example, the first sentence of the puzzle implies that Andy is not the catcher; the second sentence implies that Ed is not the second

baseman; the fourth sentence, that Harry is not third baseman; the fifth, that neither Paul nor Allen is pitcher. These negative facts may be recorded as follows:

	P	C	1st	2nd	3rd	SS	LF	RF	CF
Andy		N							
Ed				N					
Harry					N				
Paul	N								
Allen	N								
Sam									
Bill									
Jerry									
Mike									

Figure 2. Data Organization

The recording of all such implied facts leads to the puzzle's solution:

	P	C	1st	2nd	3rd	SS	LF	RF	CF
Andy	N	N	N	N	(Y)	N	N	N	N
Ed	N	N	N	N	N	(Y)	N	N	N
Harry	(Y)	N	N	N	N	N	N	N	N
Paul	N	N	(Y)	N	N	N	N	N	N
Allen	N	(Y)	N	N	N	N	N	N	N
Sam	N	N	N	N	N	N	(Y)	N	N
Bill	N	N	N	N	N	N	N	N	(Y)
Jerry	N	N	N	(Y)	N	N	N	N	N
Mike	N	N	N	N	N	N	N	(Y)	N

Figure 3. Baseball Puzzle Solution

The implications of the narrative description have been abstracted to solve the problem using a tabular format for organization. This format has permitted an orderly and rapid solution of the problem.

A more familiar use of tabular techniques is shown in Figure 4.

The "if . . . then" relationship is inherent in the table showing geographic areas and farm acreage, in the baseball puzzle, and in the income tax table. The tables shown thus far, however, have all been tables of information dealing with specific subject matter from which facts may be abstracted, such as "Your tax is . . ." In contrast, decision tables are concerned with, for example, all the records on a payroll, or the part numbers on an inventory list, or any of myriad files, for each of which some action or series of actions may be required. While decision tables are to be contrasted with tables of information, the "if . . . then" relationship is a significant feature of both types.

In summary, on a decision table, the "if" conditions, instead of providing an answer (as in the puzzle), cause some action or actions to be taken: for example, <u>if</u> an employee worked overtime, <u>then</u> compute his overtime pay. The format of a decision table separates conditions from actions. As in the puzzle, the format in which notations are made facilitates the analysis of the problem.

TAX TABLE FOR INCOMES UNDER $5,000

If your total income (item 9 on your return) is $5,000 or more, use Tax Computation Schedule on page 3 instead of this Tax Table

To find your tax read down income columns until you find the line covering the total income shown as item 9. Then read across to appropriate column headed by number corresponding to number of exemptions claimed on item 15. Enter tax as item 10.

If your total income is—		And the number of exemptions is—			If your total income is—		And the number of exemptions is—						
							1	2		3			
At least	But less than	1	2	3 If 4 or more there is no tax	At least	But less than	And you are— Single or a married person filing separately	And you are— Single or a married person filing separately	And you are— A married couple filing jointly	And you are— Single or a married person filing separately	And you are— A married couple filing jointly	4	5	6	7 If 8 or more there is no tax
		Your tax is—					Your tax is—								
$0	$675	$0	$0	$0	$2,325	$2,350	$301	$181	$181	$61	$61	$0	$0	$0	$0
675	700	4	0	0	2,350	2,375	305	185	185	65	65	0	0	0	0
700	725	8	0	0	2,375	2,400	310	190	190	70	70	0	0	0	0
725	750	13	0	0	2,400	2,425	314	194	194			0	0	0	
		17				2,450	319	199							
2,175	2,200	274	154	34	4,700					500	491				11
2,200	2,225	278	158	38	4,750	4,800	773	641	620	509	500	380	280	140	20
2,225	2,250	283	163	43	4,800	4,850	783	651	629	519	509	389	289	149	29
2,250	2,275	287	167	47	4,850	4,900	793	661	638	529	518	398	298	158	38
2,275	2,300	292	172	52	4,900	4,950	803	671	647	539	527	407	307	167	47
2,300	2,325	296	176	56	4,950	5,000	813	681	656	549	536	416	316	176	56

Figure 4.

SITUATION 1

To begin to see the structure of a decision table consisting of conditions separated from actions, consider an aspect of an airlines reservation system.

Expressed in decision table format, the problem might appear as stated below, where each column to the right of the vertical double line is regarded as a rule. A single rule at a time is examined, reading from top to bottom. Y and N mean yes and no, and X indicates execute. Actions to be taken for each rule are expressed in rows beneath the horizontal double line.

Rule No. →	1	2	3	4
REQUEST IS FOR	1st CLASS	1st CLASS	TOURIST	TOURIST
1STCLASS AVAILABLE	Y	N		
TOURIST AVAILABLE			Y	N
ISSUE	1st CLASS		TOURIST	
PLACE ON WAIT LIST		X		X

Situation 1. Airlines Reservations

To read this table, examine a single rule at a time in conjunction with the statements to the left. No matter how many alternative rules exist, only one rule can be satisfied in a single pass through the decision table. The first decision rule (the column numbered 1) can then be paraphrased: <u>If</u> the request is for a first-class reservation <u>and if</u> a first-class seat is available, <u>then</u> issue a first-class ticket. The underlined words are implied by the table layout.

The second rule says that <u>if</u> a first-class ticket is requested <u>and</u> there are no first-class seats available, <u>then</u> place the passenger's name on a wait list.

The third and fourth rules apply to tourist class. Note that where conditions are not to be considered, no entry is made in the rules column. The table considers each condition in terms of yes, no or "don't care".

The information in the above table is shown in an exploded view in the following figure to indicate the terms that are used to describe the various parts of a table.

The horizontal and vertical double lines serve as demarcation: conditions are shown above the horizontal double line, actions below; the portion to the left of the vertical double line is called the "stub", and entries are to the right. Each vertical combination of conditions and actions is called a decision rule.

There are three types of tables: limited entry, extended entry and mixed entry. This table serves to illustrate the format of a mixed entry table — that is, it combines limited and extended entries. In limited entry form, the entire condition or action is written in the stub, and the entry is limited to showing, for each case whether the particular condition is true, false or not pertinent (Y, N or blank), and whether a particular action should be performed (X or blank). In the table shown here, the second and third condition rows and the second action row are expressed in limited entry form.

In contrast, an extended entry expresses a part of the condition or action in the entry side of the table, as in the first condition row and the first action row of the table shown.

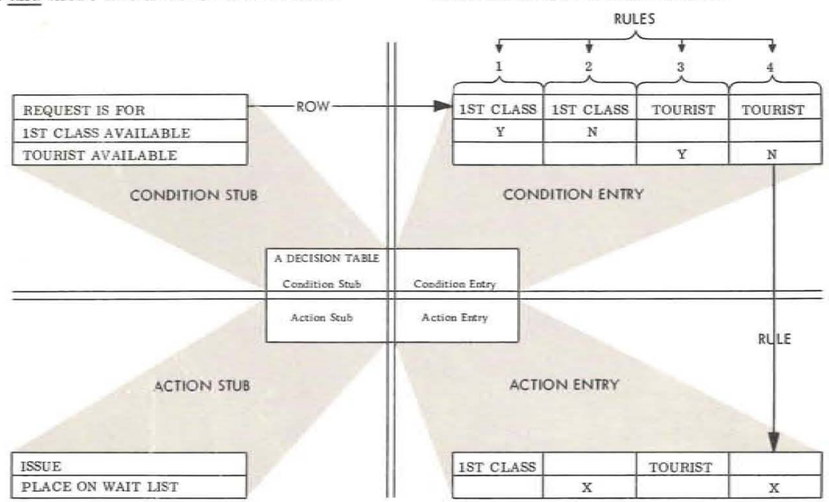


Figure 5. Exploded View of Table of Situation 1

Note that this table, being partly limited entry and partly extended entry, is a mixed entry table. Expressed entirely in limited entry form, the table would read:

Rule No. →	1	2	3	4
REQUEST IS 1ST CLASS	Y	Y		
REQUEST IS TOURIST			Y	Y
1ST CLASS AVAILABLE	Y	N		
TOURIST AVAILABLE			Y	N
ISSUE 1ST CLASS	X			
ISSUE TOURIST			X	
PLACE ON WAIT LIST		X		X

Situation 1. Airlines Reservations Expressed in Limited Entry

The original mixed entry table is shown again for contrast.

Rule No. →	1	2	3	4
REQUEST IS FOR	1st CLASS	1st CLASS	TOURIST	TOURIST
1st CLASS AVAILABLE	Y	N		
TOURIST AVAILABLE			Y	N
ISSUE	1st CLASS		TOURIST	
PLACE ON WAIT LIST		X		X

Situation 1. Airlines Reservations

In summary, a decision table defines all conditions (the prerequisites for an action) and separates them from all actions. Further, it relates given conditions to the appropriate actions, with a column of entries which form a rule. Alternative conditions that result in other actions constitute other rules which are written side by side. Tables may be limited, extended or mixed entry form.

SITUATION 2

Situation 1 has shown the format and terminology used to describe the parts of a decision table. Situation 2 again considers an aspect of airlines reservations, but in this problem another condition is added to provide the passenger with alternate class reservations, if he request them when no space is available in the class he chooses. The first rule states: If the request is for first-class (condition 1), and first-class space is available (condition 3), then issue a first-class ticket (action 1), and subtract 1 from the first-class seats available (action 3).

In rule 2 the request is for first-class, but no first-class seats are available. Tourist space is open and the passenger will accept tourist accommodations. The actions call for the issue of a tourist ticket and the subtraction of 1 from the number of tourist seats available.

Note that rules 1 through 4 all cover first-class requests and rules 5 through 8 cover tourist requests. The format of the table has led the analyst to an examination of all the possibilities for the given set of conditions.

In rule 1, no entries are made for the second, fourth and fifth condition rows, because these conditions are not applicable. If the request is for first-class and first-class space is available, there is no reason to consider whether tourist class is acceptable or whether tourist space is available.

Notice that not only is the combination of entries for each rule different from that for all other rules, but also that there is no combination of entries that satisfies the conditions of more than one rule. This is a decision table convention: the conditions should be set up so that only one rule can be carried out each time a table is examined. This means that the

Rule No. →	1	2	3	4	5	6	7	8
1ST CLASS REQUEST	Y	Y	Y	Y				
TOURIST REQUEST					Y	Y	Y	Y
1ST CLASS OPEN	Y	N	N	N		Y	N	
TOURIST OPEN		Y	N		Y	N	N	N
ALTERNATE CLASS ACCEPTABLE		Y	Y	N		Y	Y	N
ISSUE 1ST CLASS TICKET	X					X		
ISSUE TOURIST TICKET		X			X			
SUBTRACT 1 FROM 1ST CLASS AVAIL	X					X		
SUBTRACT 1 FROM TOURIST AVAIL		X			X			
PLACE ON TOURIST WAIT LIST			X				X	X
PLACE ON 1ST CLASS WAIT LIST			X	X			X	

Situation 2. Airlines Reservations with alternate reservations

rules are true alternatives, so that they may be examined in any order. If a first-class ticket is requested, for example, and first-class space is available, rule 8 may be examined first, and the rule will fail. Similarly, each rule except rule 1 will fail.

Notice further that each detail of the decision-making process has been explicitly stated and every condition considered in a direct way. The decision table format has definitely simplified the problem analysis and clearly expressed the relationships involved.

The table could have been shortened by combining conditions 1 and 2 as below:

	1	2	3	4	5	6	7	8
1ST CLASS REQUEST	Y	Y	Y	Y	N	N	N	N

Since, for this case, there are only two classes, the negative entry asserts that the request is for tourist class.

SITUATION 3

Situation 2, with the table expressed entirely in limited entry form, has shown the power of decision tables in expressing complex decision logic. Situation 3, like situation 1, is a mixed entry table containing both limited and extended entry rows. The versatility of expression allowed within the table format further increases the table's power as a communication medium, as demonstrated in the present situation:

> In setting price, the determining factors are (1) whether the sale is retail or wholesale, (2) the quantity ordered and (3) the distance of the shipment.

The table is written on a standard decision table form (X28-1630). The vertical double line separating the stub and entry is predrawn, but had more space been required for the stub or for the entry, the user could simply have drawn in his own double line, ignoring the original double line. The double horizontal line separating the conditions from the actions should be drawn by the user.

The first line within the table gives the table name. Tables may be named and/or numbered, and the name or number is used in other tables or within the table itself to reference the table.

Rule 1 is satisfied when the customer is a wholesaler but the quantity ordered is less than (LT) 10. Distance is not considered when the quantity is less than 10, and no discount is allowed. The first action row (row 5) for this rule says to record the gross price so that it will be available for use by another table. The last action row (row 7) references another table named "Prepare Invoice". This table utilizes the net price developed by the current table.

There is a decision table convention (ignored in the previous situations to simplify the tables) that each rule must specify where to go next. In the table shown, "where to go next" is the same for each rule — it is indicated in the last, and limited entry action row with "Go To". In other tables the "where to go next" might be different for each rule. One rule, for example, might specify some other table, while another rule might specify a repetition of the same table.

When different rules specify different tables in extended entry form, the words "Go To" could appear in the stub, and the next table for each rule could be listed in the entry portion of the table, as shown below:

Stub	Rule 1	Rule 2
GO TO	PREPARE INVOICE	PRICE COMPUTATION

In limited entry form, a series of action rows could be added, each referencing a different table, and the appropriate X's inserted in the entry section, as below:

Stub	Rule 1	Rule 2
GO TO PREPARE INVOICE	X	
GO TO PRICE COMPUTATION		X

Rule	1	2	3	4
PRICE COMPUTATION				
1 CUSTOMER TYPE IS	WHOLESALE	WHOLESALE	WHOLESALE	RETAIL
2 QUANTITY ORDERED	LT 10	GE 10	GE 10	
3 DISTANCE IN MILES		LT 50	GE 50	
4 SET DISCOUNT EQ		.15	.10	
5 SET NET PRICE EQ	GROSS	GROSS*(1.00-DISCOUNT)	GROSS*(1.00-DISCOUNT)	GROSS
6 MOVE DISCOUNT*100 TO DISCOUNTPERCENT		X	X	
7 GO TO PREPARE INVOICE	X	X	X	X

Remarks: LT=LESS THAN
GE=GREATER THAN OR EQUAL TO
The asterisk * is used here to denote multiplication.

Situation 3. Price Computation

The ability to reference other tables allows the problem to be divided into logical segments and makes provisions for multi-level structure in problem analysis. At the highest level the overall system is analyzed by referencing lower-level tables. These tables, in turn, can express the logic of the problem to the degree of detail necessary at that level; these tables may, in turn, call upon lower-level tables with even more detailed conditions and actions. Alternatively, a flow chart may be used to show the overall structure, supported at lower level by decision tables.

In the present situation, rule 2 states that the customer is entitled to wholesale rates; that the quantity ordered is greater than or equal to (GE) 10, and that a 15% discount is to be applied. The net price is computed by multiplying the gross price by 100% less the discount determined. In other words, the net price here is computed by multiplying the gross price by 85%. Action rows 5 and 6 cause the net price and the discount percentage to be stored for use by the prepare invoice routine.

Rule 3 is similar to rule 2 except that the delivery distance exceeds 50 miles and the discount is reduced to 10%.

In rule 4 no discount is allowed on retail sales regardless of quantity or delivery distance.

In this table it would be illogical to execute the actions in any other order except the order in which they are written. This, again, is a decision table convention: actions are executed in the order in which they are written.

Note that mnemonics (LT and GE) and symbols (*) have been used in the table. The vocabulary and grammar of the language used within the decision table are left to the discretion of the user. It is his responsibility to insure that the language used in describing the system will be understandable to those using the documentation.

As in the examples given, a condition must be answerable by either yes or no. Some typical condition statements are listed below:

Transaction is a change item
FICA is greater than 4800
Department number equals 276, 302, 736, or 914
Net amount is positive
Switch 1 is on
End of detail file
Start

In like manner, actions may be expressed in any language which will connote proper meaning to the reader. Examples:

Add FICA total to FICA total
Do gross-to-net procedure
Read master record
Write detail line
Go to update procedure
Go to Table A
Set A equal to B+C-D
A=B+C-D

The decision table form (X28-1630) is shown in its entirety on page 20.

SITUATION 4

In each of the previous situations a problem has been stated, and the text has then presented a decision table followed by an explanation of the table. Situation 4, on the other hand, is one approach to the writing of a decision table.

The procedure the systems analyst follows in preparing a decision table may be simply to identify conditions and actions of the problem statement as he encounters them. Conditions are placed on the top half of the form; actions on the bottom half.

Consider the following problem narrative:

> When the quantity ordered for a particular item does not exceed the order limit and the credit approval is "OK", move the quantity-ordered amount to the quantity-shipped field; then go to a table to prepare a shipment release. Of course, there must be a sufficient quantity on hand to fill the order.
>
> When the quantity ordered exceeds the order limit, go to a table named "Order Reject". Do the same if the credit approval is not "OK".
>
> Occasionally, the quantity ordered does not exceed the order limit, credit approval is "OK", but there is insufficient quantity on hand to fill the order. In this case, go to a table named "Back Order".

Note that this is not written with all conditions first, prefixed by "if", and with all actions following, prefixed by "then". The narrative was written casually with conditions and actions scrambled, much like the original baseball problem. Words like "when" and "occasionally" are used instead of the more precise "if". Such ambiguity is typical of most narratives. For illustrative purposes the problem is restated below with a solid line under conditions and a broken line under actions.

> When the quantity ordered for a particular item does not exceed the order limit and the credit approval is "OK", move the quantity-ordered amount to the quantity-shipped field, then go to a table to prepare a shipment release. Of course, there must be a sufficient quantity on hand to fill the order.
>
> When the quantity ordered exceeds the order limit, go to a table named "Order Reject". Do the same if the credit approval is not "OK".
>
> Occasionally, the quantity ordered does not exceed the order limit, credit approval is "OK", but there is insufficient quantity on hand to fill the order. In this case, go to a table named "Back Order".

A count shows eight conditions and five actions for the problem.

C1 QTY ORDERED IS LESS THAN OR EQUAL TO ORDER LIMIT
C2 CREDIT APPROVAL IS "OK"
C3 QTY ON HAND IS GREATER THAN OR EQUAL TO QTY ORDERED
C4 QTY ORDERED IS GREATER THAN ORDER LIMIT
C5 CREDIT APPROVAL IS NOT "OK"
C6 QTY ORDERED IS LESS THAN OR EQUAL TO ORDER LIMIT
C7 CREDIT APPROVAL IS "OK"
C8 QTY ON HAND IS LESS THAN QTY ORDERED

A1 MOVE QTY ORDERED TO QTY SHIP
A2 GO TO SHIP RELEASE
A3 GO TO ORDER REJECT
A4 GO TO ORDER REJECT
A5 GO TO BACK ORDER

Notice that C1 and C2 are identical to C6 and C7, and that A3 is identical to A4. This occurs because a narrative describes rules one after another (serially). Thus two sets of two conditions common to two rules, appear in the narrative. On the other hand, a decision table aligns rules side by side (parallel). Thus a condition or action common to several rules need appear only once. Furthermore, C5 is not necessary since it is the negative of C2. Similarly C4 and C8 are the negatives of C1 and C3. Negative entries after the positive statements of C1, C2 and C3 cover the other cases.

The next step in preparing the table might be to identify and consolidate similar rows. C1, C6 and C4 are combined. C2, C5 and C7 are combined. Finally, C3 and C8 are combined.

In the action half of the decision table form, A3 and A4 are combined. After consolidation there are only three condition rows and four action rows.

Condition Stub

QTY ORDERED IS LESS THAN OR EQUAL TO ORDER LIMIT
CREDIT APPROVAL IS "OK"
QTY ON HAND IS GREATER THAN OR EQUAL TO QTY ORDERED

Action Stub

MOVE QTY ORDERED TO QTY SHIP
GO TO PREPARE SHIP RELEASE
GO TO ORDER REJECT
GO TO BACK ORDER

The stub portion of the table is now completed. In order to fill out the entry portions, the analyst must determine the rules expressed in the narrative. In this example, the first paragraph describes a single rule. The analyst enters the appropriate Y, N or X in the entry portions for rule 1. The second paragraph contains two rules. The analyst enters the appropriate Y, N, or X in the entry portion for rules 2 and 3. Finally, the last paragraph becomes rule 4 of the decision table. The final result is shown below:

	Rule	1	2	3	4
	OPEN				
1	QTY ORDER LE ORDER LIMIT	Y	N	Y	Y
2	CREDIT APPROVAL IS "OK"	Y		N	Y
3	QTY ON HAND GR QTY ORDERED	Y			N
4	MOVE QTY ORDERED TO QTY SHIP	X			
5	GO TO PREPARE SHIP RELEASE	X			
6	GO TO ORDER REJECT		X	X	
7	GO TO BACK ORDER				X

Situation 4. Order Approval, Reject or Back Order

A comparison of this decision table with the narrative statement of the problem makes it evident that the table is more concise, while pointing up cause-and-effect relationships and emphasizing logical alternatives. The resultant table is in limited entry form. If there are many rules in a table, the use of limited entry usually permits more rules on a single decision table form than is possible with extended entries.

In general, limited entries compress a table horizontally while extended entries compress a table vertically. The table shown, for example, could be compressed vertically by combining the three "Go To" action statements into one extended entry row:

	Rule 1	Rule 2	Rule 3	Rule 4
GO TO	PREPSHIPREL	ORDERREJECT	ORDERREJECT	BACKORDER

A compromise using rows of both types — that is, a mixed entry table — is often useful as in situation 1 (airlines reservations).

Aside from the terms used to describe tables according to the type of entries made, there are two other table types classified by the control they have on the sequence of execution or flow from table to table. Just as actions within a table are executed in the order in which they are stated, there must be similar control over the order in which tables are executed.

Within a set of tables, it may be desired for one table to permanently pass control to another table. This is accomplished by using appropriate terminology such as "Go To". On the other hand, it is often desirable within one table to execute another table and then return to the original table so that remaining actions can be executed. A notation such as "Do" can be used to accomplish this. For example, in a table it may be desirable to call other tables to perform certain functions such as determine net price, compute shipping charges, and prepare invoice. Each of these tables can be called and executed by actions in this calling table, for example — "Do determine net price". The tables to be executed by actions in another table with an implied return to the calling table are designated "closed" tables and may be compared with closed subroutines. Tables which are entered by a "Go To" action are designated "open" tables, and no return is implied.

Figure 6 shows within the numbered circles the sequence of execution of four tables.

Each of the open tables states where to go next. The closed table is referenced with a "Do" command in tables 002 and 004. When the closed table is referenced by table 002, it returns control to the next action row of table 002; when it is referenced by table 004, it returns control to the next action of table 004.

Not only may more than one table reference a closed table, but frequently more than one rule within a given table references the same closed table.

Closed tables make it unnecessary to repeat conditions and actions common to more than one table or more than one rule within a table by abstracting the conditions and actions to form a closed table. Further, they permit re-entry to the table which referenced them without starting at the top of the table — the normal entry point.

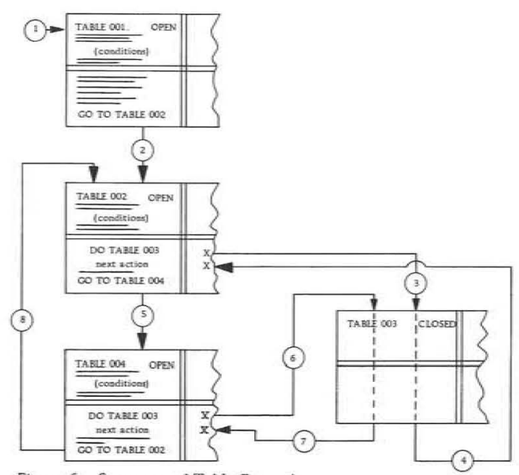


Figure 6. Sequence of Table Execution

SITUATION 5

The analyst should check to see that rules are not omitted. In extended entry, for example, in an insurance application where each policy type forms a rule, the analyst should check to see that all policy types issued by the company are included. Similarly, he should check to see that all the logical situations are covered. If, for instance, one rule states "greater than" and another rule states "less than", the analyst should note that the "equal to" situation has not been covered and should then determine either that the "equal to" situation is pertinent to one of the existing rules or that another rule should be added.

In limited entry form, for many tables simple arithmetic indicates whether every possibility has been considered. Each entry for a single condition has two possible values: yes or no (a blank covers both). Therefore, to determine the maximum number of possible rules, two is multiplied by itself as many times as there are conditions. Given three conditions, for example, this number is 2^3 or 8. The analyst knows before the table is written that he must examine eight possible combinations.

In the following situation there are three conditions and eight rules to cover all possible combinations:

> Three files of undetermined length are to be examined. As each record is read in, it is checked to see whether it signals the end of the particular file. When one file or any combination of the files is exhausted, a specific series of actions is required, depending on which file or combination has reached its end.

The table below provides for all end-of-file possibilities.

A	EQ	EOF	N	N	N	N	Y	Y	Y	Y
B	EQ	EOF	N	N	Y	Y	N	N	Y	Y
C	EQ	EOF	N	Y	N	Y	N	Y	N	Y
GO TO			R1	R2	R3	R4	R5	R6	R7	R8

Situation 5. End-of-File Test

SITUATION 6

The logic in a decision making process for a typical function of data processing equipment is documented with a decision table describing a merging function.

> Three decks of punched cards are arranged in numerically ascending sequence. The three decks are to be combined in a single sequence with all cards bearing the same number from any of the decks placed together.

In a manual operation, the first card from each of the decks would be compared and the lowest card of the three selected. Removing the card selected from one deck and placing it in the merged deck would reveal the next card in the deck from which it was taken. This newly exposed card would then be compared with the two cards already exposed. Again the lowest of the three would be selected. Where duplications occurred, they would be assembled together; that is, both or all three cards would be selected if they had the same number.

It would also be desirable, after reading a new card, to compare the number of this card with the last card filed. If the card just read has a lower number than the last card placed in the merged deck, then the cards in the input deck were not in ascending sequence as assumed. This is, of course, an error condition.

Analyzing this merging and checking operation for machine processing makes it apparent that three records must be read first. After this initial reading, only one record at a time will be read, depending on the file from which a record was selected.

To return to the initial conditions, when the first three records are read and the lowest one selected, there is no previous record in the merged file with which to compare the record to be filed. Therefore, in machine processing, provisions must be made not only for the repetitive operations, but for these initial conditions of reading three records instead of one, and for the fact that there is no previous card with which to compare the first record selected. All these conditions are provided for in the tables shown on page 13.

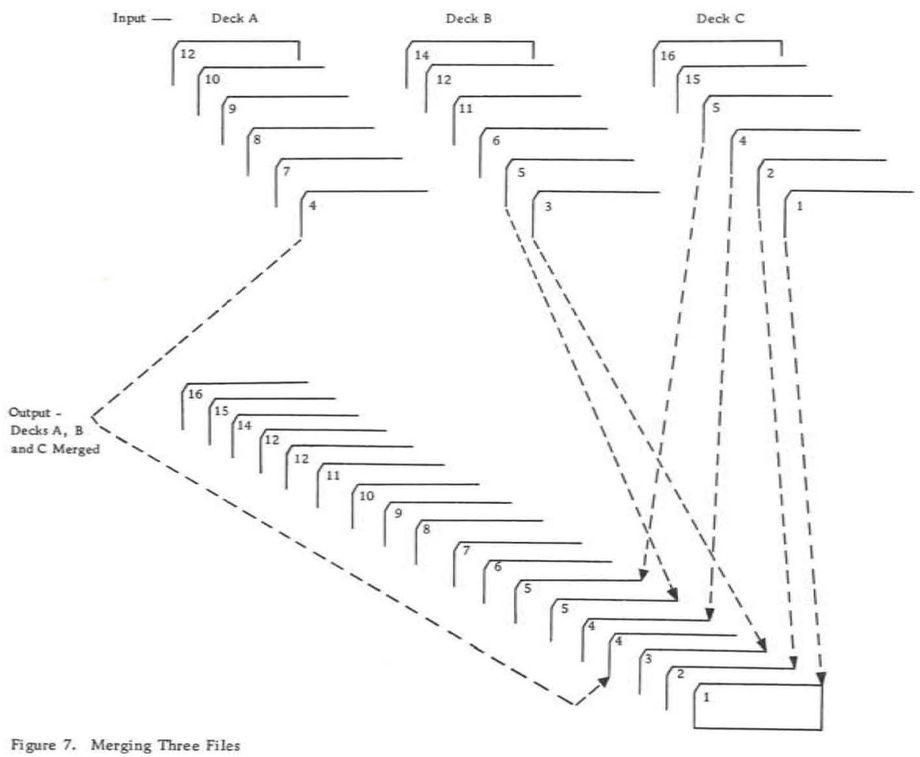


Figure 7. Merging Three Files

Since more than one decision table has been written on a single form, the beginning of each table has been indicated by a number in its heading and cross-hatching in the row reference area next to the table header information — the table name and/or number and the rule designations.

Table 1 has only one rule and no conditions: all the rows (1 through 5) are actions. This is indicated by the double line above row 1. Such a table is called an "unconditional" table. As in all decision tables, the actions are to be executed in the order in which they are written. Table 1 is executed only once. Before the repetitive operations of table 2 can be performed, table 1 must be executed.

The first condition row of table 2 checks to see whether the area designated "New Record" is less than the area designated "Test Area." On the first pass through table 2, one card has been read from each of the decks and both test areas have been set to zero by table 1. Condition row 1 will determine that the "New Record" identification is not less than "Test Area" which is filled with zeros; so that rule 4 will not be satisfied. This test performed by condition row 1 is meaningful only on succeeding passes through the table.

Rules 1, 2 and 3 in combination with condition rows 2, 3 and 4 are the crux of the whole operation. Rule 1 says that if A is less than or equal to B, and A is less than or equal to C, then select (specified as "write" in action row 6) record A. Before recording A as the lowest number in the sequence, however, action row 5 sets the test area to the identification number on record A. Row 6 writes this record. Row 7 then reads a new record from file A. The test area designated "New Record" is then set to the identification number of the record just read in. On the next pass through the table a check is made in row 1 to see whether the new record is lower than the previously selected record. This is the check to make sure that the records in the input are in sequence.

The actual merge abstracted from the overall operations appears as below:

	Rule No.	1	2	3
2.	A LE B	Y	N	
3.	B LE C		Y	N
4.	A LE C	Y		N
6.	WRITE	RECORD A	RECORD B	RECORD C
7.	READ	RECORD A	RECORD B	RECORD C
8.	GO TO TABLE 2	X	X	X

Figure 8. Abstracted Merge

There are three conditions, and the analyst will have noted that the maximum number of possible rules is 2^3 or 8. As he makes these entries, however, he observes two things: (1) certain rules are illogical because they assert that a number is both greater than and less than another number, and (2) several rules result in the same action and the conditions are such that the elimination of entries consolidates the table.*

To return to the overall operation of table 2, if condition 1 is satisfied, then the new record has a lower number than the previous record written, and this represents an out-of-sequence number in the input. The action specified for rule 4 is action row 10, which says "Go To error table". The error table would be another table that has been set up to take care of this situation.

Following the table header in table 2, an unexplained line states "Frequency" and has an entry of 0 for rule 4, 25 for rules 1 and 2, and 50 for rule 3. These entries total 100%, representing complete execution of the table. When it can be estimated how often each rule will be satisfied, such an entry may be made. For rule 4 it is anticipated that an out-of-sequence item will occur less than .5% of the time; hence an entry of 0 for frequency is made. File C is known to be approximately equal to the size of Files A and B combined. Files A and B are known to be approximately equal. These entries of 25 for rules 1 and 2, and the entry of 50 for rule 3 indicate this. Frequency information is useful to the programmer in writing efficient coding from the tables.

To summarize the merge as stated in tables 1 and 2, table 1 is executed only once and has no conditions. Table 1 sets up the initial conditions for table 2 which will be repeated for each record in the three files. Provisions are made to check to see that the input is in sequence. The table is not complete, however, since the end-of-file conditions (as shown in situation 5) would have to be considered.

*A full explanation of methods for checking tables for completeness is beyond the scope of this manual. It should be noted, however, that in some cases, such as the end-of-file situation, an arithmetic check is easily made. For other tables it becomes a complex computation, and is more easily checked by a logical examination.

	Rule	1			
	TABLE 1 OPEN				
1	READ RECORD A	X			
2	READ RECORD B	X			
3	READ RECORD C	X			
4	SET TEST AREA EQ ZEROS	X			
5	SET NEW RECORD EQ ZEROS	X			
6	GO TO TABLE 2	X			
	TABLE 2 Rule / OPEN	1	2	3	4
	FREQUENCY	25	25	50	0
1	NEW RECORD LR TEST AREA	N	N	N	Y
2	A LE B	Y	N		
3	B LE C		Y	N	
4	A LE C	Y		N	
5	SET TEST AREA TO	A	B	C	
6	WRITE	RECORD A	RECORD B	RECORD C	
7	READ	RECORD A	RECORD B	RECORD C	
8	SET NEW RECORD TO	A	B	C	
9	GO TO TABLE 2	X	X	X	
10	GO TO ERROR TABLE				Y

Situation 6. Merging Three Files

SITUATION 7

In phase 1 of an inventory control situation, unsorted transaction records for various items are received from a number of branch offices. The branch makes a count of the number of records, and this numerical count becomes another record which is sent with the transaction records to the central office.

At the central office, the transaction records, along with the batch slips from each office, are put on tape. The number of transactions from each branch is checked by the system against the batch slip.

The primary purpose of phase 1 of the system is to sort all records into ascending sequence by inventory number in blocks of 27. No consideration is made of the branch from which the record originated, after the records are counted. After sorting the records in blocks of 27, a standard output routine not described here will write the records for further processing.

Sorting applications involve a wide range of machines, problems and special considerations. Highly efficient sorting programs are available from IBM, and the accompanying table is presented to show decision table techniques and some of the general considerations involved in data processing. It was designed as a teaching aid rather than as an example of fully sophisticated sorting.

In the table shown, each record is checked to see whether it is a batch slip, and after the batch slip is checked and discarded, items from another branch are added to fill the block to 27.

It is unlikely that the items in the last batch will be an exact multiple of 27. In this case, the sort area is filled with 9s to identify the short record for the next phase when further processing is done.

Two tables are used to describe the process. As previously discussed, tables spell out detail to the degree necessary for communications at a specified level. Phase 1 is an analysis of the overall operation. It references a table named "Sort", which describes procedures and references other tables not shown (action rows 8, 13, 17, 18, 19, 20 and 21).

When it is desirable to specify more detail than the scope within which a table is written, a lower-level table is referenced and the details specified in the lower table.

The controlling routine for the sort is phase 1, which has, as its first rule, a condition to determine whether the operation is beginning. For this start condition, the first action specified is the closed routine "Initialize". Before the actual processing of data, operations pertaining to the peripheral equipment and to internal controls must be performed.

	Rule	1	2	3	4	5	6	7	8
PHASE 1	OPEN								
1	START	Y	N	N	N	N	N	N	
2	INPUT EMPTY		N	N	N	N	Y	Y	
3	ITEM EQ BATCH SLIP		N	N	Y	Y			
4	BATCH SLIP EQ COUNT				Y	N			
5	SORT AREA FULL		N	Y					
6	END-OF-FILE						N	Y	
7	END OF PHASE 1						N	N	Y
8	DO INITIALIZE	X							
9	MOVE ITEM TO SORT AREA		X						
10	ADD 1 TO COUNT		X						
11	SET COUNT EQ ZERO	X			X				
12	READ TAPE	X					X		
13	DO SORT AREA FILL WITH 9'S							X	
14	SET END OF PHASE 1 INDICATOR							X	
15	SET N EQ 1			X				X	
16	SET SWITCH EQ A			X				X	
17	DO CLOSE ROUTINE								X
18	GO TO SORT			X				X	
19	GO TO ERROR TABLE				X				
20	GO TO PHASE 1	X	X		X		X		
21	GO TO PHASE 2								X

Situation 7. Phase 1 of Sort

	Rule	1	2	3	4
SORT	OPEN				
1	N LESS THAN 27	Y	N	Y	N
2	SWITCH SETTING EQ	A	A	B	B
3	MERGE RECORD #N WITH (N+1) AND (N+2)	X			
4	MERGE RECORD (#N THRU N+2) WITH (N+3 THRU N+5) AND (#N+6 THRU N+8)			X	
5	MERGE RECORD (#1 THRU 9) WITH (#10 THRU 18) AND (#19 THRU 27)				X
6	SET N EQ N+3	X			
7	SET N EQ N+9			X	
8	SET SWITCH EQ B		X		
9	SET N EQ 1		X		
10	MOVE 27 ITEMS TO OUTPUT				X
11	GO TO SORT	X	X	X	
12	GO TO PHASE 1				X

Situation 7. Sort

Such requirements would be spelled out in the "Initialize" table. The next action sets a counter to zero, so that in subsequent rules a count may be properly incremented. After a portion of tape is read, the final action is "Go To Phase 1", so that the table will be re-executed.

On the next pass through the table, rule 2 is satisfied, since this is not the start, the read operation has filled the input area, and the first item will not be a batch slip, nor will the sort area be filled.

Note that the first condition of rule 2 has a negative entry (rules 3, 4, 5, 6 and 7 also have this negative entry). Since this test is made in rule 1, it may not be obvious why a negative entry is made in these rules. The entry is necessary because rules may be considered in any order. Rule 2 can be examined first, without regard to rule 1, and if no entry for condition row 1 were made, then this condition would not be considered. If this were the start and some rule other than rule 1 were satisfied because the start condition was ignored, the table would become meaningless.

The actions for rule 2 call for the movement of one item to the sort area and for adding 1 to "Count" (set to zero by rule 1). The final actions calls for a re-entry to this table. Normally, rule 2 would be repeatedly executed until 27 items were accumulated in the sort area. Condition row 5 determines when the sort area is filled, and rule 3 is then satisfied. The actions for rule 3 call for setting N EQ 1, setting a switch equal to A, and for the transfer to the sort table. The settings are necessary for the execution of the sort table. The sort table, being an open table, takes control and, after sequencing 27 items, returns control to phase 1.

In the fourth and fifth rules of phase 1, a batch slip is encountered. Condition row 4 checks the count on the batch slip against the count made as the items were transferred from the input area to the sort area. If the counts are equal, then rule 4 is satisfied, the counter is reset to zero, and the table is re-entered. When the count on the batch slip does not match the transfer count, then control is transferred to an error table.

In rule 6 the input area is empty and it is not the end of the file or the end of this phase of the processing, and the only actions called for are for reading the tape and re-entering the table.

In rule 7 the input area is empty and the entire file has been read, but the processing for this phase is not complete. The actions call for filling the remaining storage locations in the sort area with 9s, and setting an indicator to show that after this rule is executed, this phase of the processing will be over. Rule 7 also calls for the setting of controls for the sort, and then for a transfer to the sort table to perform the sort on any remaining items in the sort area as well as the 9s added by action row 13. After the sort is completed, the action of row 12 in the sort table returns the program to phase 1. After rule 7 is executed, the only rule which can be satisfied is rule 8, because rule 7 has set the end-of-phase-1 indicator. The actions for rule 8 call for executing the "close routine", which performs whatever operations are necessary to complete the run, such as rewinding tapes, and finally for the entry of phase 2, which is not shown.

The sort table, like most sorting routines, is based on a succession of merges such as the one shown in situation 6. In this sort table, three records are merged, then three more, and the merge is repeated until nine strings of three records each are accumulated:

3	1	3	7	6	13	11	1	3
4	2	5	11	6	14	13	3	8
9	8	10	14	10	16	15	5	15

The first three strings from the group of nine are then merged, then the next three strings, and then the last three strings; so that the result is three strings of nine records:

1	6	1
2	6	3
3	7	3
3	10	5
4	11	8
5	13	11
8	14	13
9	14	15
10	16	15

The three strings of nine are then merged as the last step, forming one string of 27 records.

SAMPLE PROBLEM USING DECISION TABLES (SITUATION 8)

The basic ideas of decision table usage have been illustrated with examples which show some aspect of a data processing operation. Such tables may be used in conjunction with flow charts and block diagrams. Decision tables may also be employed to describe an entire data processing system as well as a portion of the system.

At the highest level the overall logic of the system is described, and references are made to lower-level decision tables which describe more completely the activities to be accomplished and the conditions under which more detailed actions are to be performed.

The problem presented here shows various levels of decision table documentation, but it is not complete, since only a single path is covered at the next lower level of the multi-level structure.

> The purpose of this system is to perform all file maintenance functions required for the policy master file in an insurance application. An index file is used to locate records within the policy master file. There is an input file of current activity records. In addition to updating the master file to reflect current activities, there is a daily date check. Each record contains the next date on which some processing for that record is required, and the file is examined to see whether the current date matches the date when some activity such as billing is required for that record. In addition to an updated policy master file, the system will produce other outputs for other systems.

While the vocabulary and grammar of the language used within the table are left to the discretion of the user, some qualification or specification of names which could apply to more than one file or record is necessary. (A record is a collection of fields where fields contain a value or unit of information. A file is collection of one or more records associated with an input or output device.) The analyst chose to associate fields with records and files by qualifying the field name with the record or file name and separating the names with colons. The field "activity date" in row 1 of the activity determination table, for example, is associated with the file index.

The first table, numbered 001 and named "Start", is an unconditional table. The first action row uses the word "Do" to reference a closed routine entitled "Housekeeping procedure". This routine, which is not shown, opens and checks the files and does whatever else is necessary to begin the processing run. Being a closed routine, it causes an automatic return to the table which called it (table 001). The second action row uses the words "Go To" to enter the open table called "Activity Determination".

Table 002, named "Activity Determination", has as its first condition a check to see whether the current date coincides with a date from the record on which some activity is required. The second condition checks to see whether an input transaction item is applicable to this record. The four possible combinations of entries to these two conditions are shown as four rules.

To illustrate the single path followed in this explanation, an area is shaded to indicate that it is assumed that the current record satisfies rule 1. Such shading is not a decision table technique and is used here only as a teaching aid. Rule 1 says that the current date coincides with the date on the record on which some activity is required, but there is no input transaction applicable to this record. Two actions are indicated for rule 1: "Do set up procedure" (a closed table, not shown) and "Go To automatic changes", an open table expressed in limited entry form.

	Rule	1	2	3	4
001	START — OPEN				
1	DO HOUSEKEEPING PROCEDURE	X			
2	GO TO ACTIVITY DETERMINATION	X			
002	ACTIVITY DETERMINATION — OPEN	5	10	83	2
1	CURRENT DATE EQ ACTIVITY DATE: INDEX	Y	N	N	Y
2	POLICY NO: TRANS EQ POLICY NO: INDEX	N	Y	N	Y
3	DO SET UP PROCEDURE	X	X		X
4	GO TO	AUTOMATIC CHANGES	EXTERNAL CHANGES	NEXT POLICY	PRIORITY

Situation 8. Start and Activity Determination

Table 32, named "Automatic Changes", determines which of five types of automatic changes is required. Assume this particular record satisfies the condition of rule 3. The only action called for by rule 3 is "Go To nonpayment routine".

To the right of rule 5, an unexplained column has an entry "E". Decision tables provide two alternative means of handling situations in which provisions are not made for all possible combinations of conditions:

1. If it is expected that cases will arise fairly frequently which do not satisfy any of the rules, an "all others" or "else" rule can be provided as in this situation where the E indicates else.
2. Where failure to satisfy any of the rules represents an error in the logic of the table or in the data, provisions should be made to consider some error table. This can be noted in the table header.

	Rule	1	2	3	4	5
32	AUTOMATIC CHANGES — OPEN	8	8	2	3	2 E
1	STATUS IS BILLING	Y				
2	STATUS IS LATE PAY		Y			
3	STATUS IS NONPAYMENT			Y		
4	STATUS IS ANNIVERSARY				Y	
5	STATUS IS CHANGE					Y
6	GO TO BILLING ROUTINE	X				
7	GO TO LATE PAY ROUTINE		X			
8	GO TO NONPAYMENT ROUTINE			X		
9	GO TO ANNIVERSARY ROUTINE				X	
10	GO TO CHANGE ROUTINE					X
11	GO TO OTHER ACTIVITY					

Situation 8. Automatic Changes

Table 50, named "Nonpayment", is an open table. It determines, on the basis of two conditions, which of three other tables will be called. The first rule tests to see whether the policy has been in effect less than or equal to a year, and for such policies the only action specified in this table is to "Go To" the table named "Lapse". If rule 2 is satisfied, the policy has been in effect for more than a year and the policy status entitles the holder to an automatic loan.

If the third rule applies, the policy has been in effect for more than a year but it is not automatically entitled to a loan. Five actions are specified for this rule. Here, to demonstrate a closed table, a single action row (5) is traced.

	Rule	1	2	3
50	NONPAYMENT — OPEN			
1	WORK POLICY DURATION (YEARS)	LE 1	GR 1	GR 1
2	NONFORFEIT STATUS: PI		EQ AUTOMATIC PREMIUM LOAN	UN AUTOMATIC PREMIUM LOAN
3	SET WORK MONTH PAID EQ MONTH PAID		X	X
4	SET WORK YEAR PAID EQ YEAR PAID		X	X
5	DO DURATION CALCULATION			X
6	DO CASH VALUE PROCEDURE			X
7	DO ATTAINED AGE		X	
8	GO TO	LAPSE	MAKE AUTOMATIC LOAN	INFORMATION SHEET

Situation 8. Nonpayment

Duration calculation (Table 82), the closed table referenced in the previous table with a "Do" command, will, after execution, return program control to the table which referenced it. Table 82 determines the duration of the policy in the same manner that a person's age might be calculated to the month. The month of the last payment which was stored by the previous table is compared with the date of issue of the policy.

In rule 2 the months coincide, so that the two actions called for are to subtract the year of the date of issue from the year of the last payment. Zero is specified for months, since there are no extra months to be counted (as a person born in June 1930 is 32 years old in June 1962).

In rule 1 the last payment was made in some month after the anniversary of the policy. The number of months is computed by subtracting; for example, a policy on which a payment was made in August (the eighth month) which was issued in June of another year, would be two months older than the yearly calculation indicates.

In rule 3, the anniversary month has not been reached, one year is subtracted from the number of years of the policy's duration computed in row 2. A policy issued in June of 1955, for example, is not seven years old at the beginning of 1962 (1962 — 1955 = 7). It is not seven years old until June 1962; so that one year must be subtracted after the initial calculation. Again, the months must be computed. Here the months from the previous year, which have been paid, plus the months paid in the current year are computed by row 4.

Each record processed will satisfy one of the rules of the table. After execution of this table, the remaining actions of the table which called on it are executed — in this case row 6 of table 50, which states "Do cash value procedure". This table, which is not shown, would compute the cash value utilizing the duration time computed by the "Duration Calculation" table.

	Rule	1	2	3
50	NONPAYMENT	OPEN		
1	WORK POLICY DURATION (YEARS)	LE 1	GR 1	GR 1
2	NONFORFEIT STATUS: PI		EQ AUTOMATIC PREMIUM LOAN	UN AUTOMATIC PREMIUM LOAN
3	SET WORK MONTH PAID EQ MONTH PAID		X	X
4	SET WORK YEAR PAID EQ YEAR PAID		X	X
5	DO DURATION CALCULATION			X
6	DO CASH VALUE PROCEDURE			X
7	DO ATTAINED AGE		X	
8	GO TO	LAPSE	MAKE AUTOMATIC LOAN	INFORMATION SHEET

Situation 8. Nonpayment (repeated from previous page)

	Rule	1	2	3
82	DURATION CALCULATION	CLOSED		
1	MONTH PAID VS MONTH ISSUED	GR	EQ	LT
2	SET YEARS IN FORCE EQ YEAR PAID MINUS YEAR ISSUE	X	X	X
3	SUBTRACT 1 FROM YEARS IN FORCE			X
4	SET MONTHS IN FORCE EQ	MONTH PAID − MONTH ISSUED	ZERO	12 + MONTH PAID − MONTH ISSUED

Situation 8. Duration Calculation

REFERENCE

DECISION TABLE FORMAT

The four basic elements which constitute the structure of a decision table are shown in a schematic diagram below:

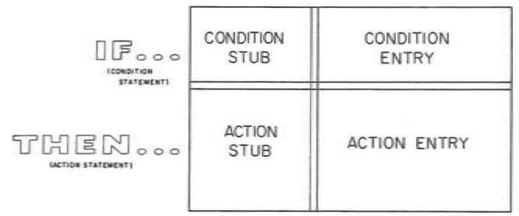


Figure 9. Basic Elements of a Decision Table

The double horizontal and vertical lines separate the four basic elements: Above the horizontal double line is the condition area; below, the action area. To the left of the vertical double line is the stub; to the right, the entry area.

The relative position of each element should remain consistent — that is, the stubs are always to the left of the entries, and the conditions are always placed above the actions.

A table can be entered only at the top (a single entry point), but may have multiple exit points. Therefore, note that a table must be used in its entirety. There is no way of executing a selected rule isolated from the table.

Conditions and actions have an "if . . . then" relationship; no actions may appear in the condition area, no conditions may be indicated in the action area. No values may be changed by conditions.

In a limited entry condition or action, the entire condition or action must be written in the stub; the entry is used to show, for each case, whether the particular condition is true, false or not pertinent (Y, N, or blank), and whether a particular action should be performed (X or blank).

In extended entry format, part of the condition or action is extended into the entry.

Limited and extended entry form may be freely mixed within a table, but a single condition or action row must be in just one format.

A condition must be answerable by either a yes or no.

Actions are to be written in the order in which they are executed.

RULES WITHIN THE DECISION TABLE

A rule is one vertical column (or group of associated columns numbered as a single rule) of the entry portion of the table and is read downwards in conjunction with the stub.

For a rule where multiple conditions exist, all conditions must be satisfied before the actions are executed.

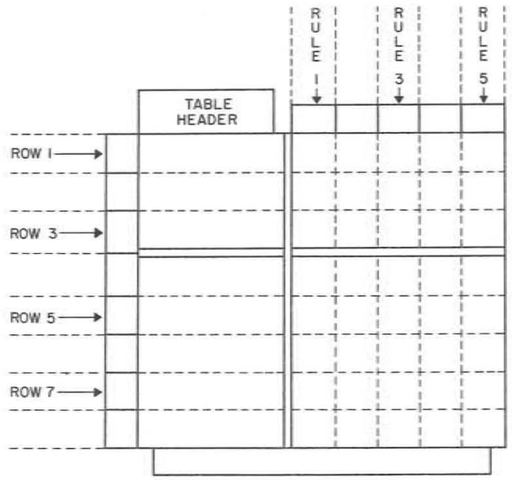


Figure 10. Further Elements of a Decision Table

Each decision table must indicate where to go next — to another table or to the table itself.

LANGUAGE OF THE DECISION TABLE

The vocabulary and grammar of the language used within the decision table are left to the discretion of the user. It is his responsibility to insure that the language used in describing the system will be understandable to those who use the documentation.

DECISION TABLE FORM

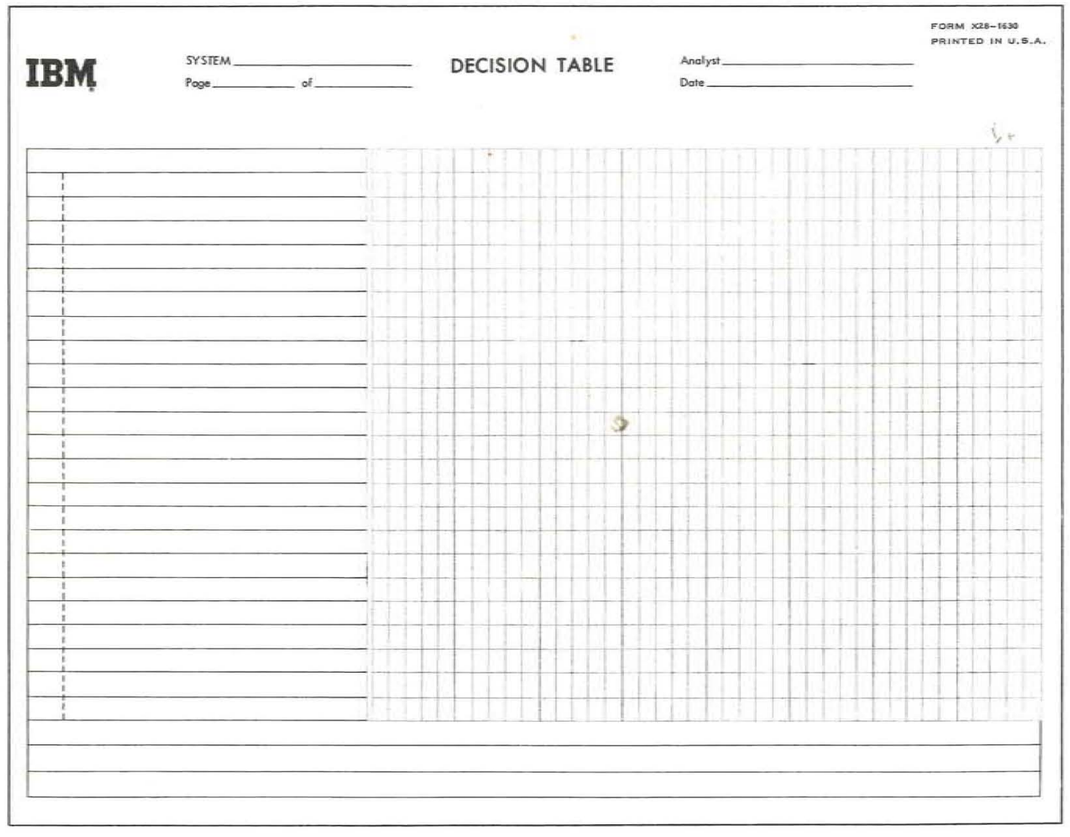
A decision table form (X28-1630) is available for those including decision tables in their documentation. The vertical lines in the entry area are printed in two weights. The lighter, close-together vertical lines are used to separate rules written entirely in limited entry form. The wider-spaced, slightly heavier vertical lines may be used to separate the rules for

extended entry form. If the user desires to have narrower or wider columns, he indicates this by drawing in his own vertical lines. Thus in a single table, space for one rule may be different from the space indicated for another rule. The preprinted lines will save the user time and bother when a uniform format is convenient.

In writing on the form, it is not necessary to allow only one character per square on the grid. The vertical lines may be ignored, so that more than one character may appear in a square or a single character may span two squares.

The vertical double line separating the stub and entry is predrawn, but may be ignored when the user draws in his own line. The user should draw the horizontal double line separating conditions from actions.

As a convenience to the user it is permissible to include more than one decision table on a single decision table form. The beginning of each table is indicated by separating it from preceding tables with blank lines and, if desired, cross-hatching the reference area next to the table header information. Table and rule reference information is supplied for each table just before the writing of the decision table proper. Multiple tables per page should be made in the vertical direction only — that is, the stubs of all tables on a single form should be aligned.



RECOMMENDED AND OPTIONAL USES OF THE VARIOUS ELEMENTS OF THE DECISION TABLE

R - Recommended
O - Optional

1. TABLE HEADER
   - R - Table number and/or table name.
   - O - Table dimensions, i.e., number of rules, conditions, or actions; next table to be considered; error table; comments; order of rules.

2. RULE REFERENCE
   - R - Rule number.
   - O - Rule name; frequency.

3. ROW REFERENCE
   - O - Row identification; footnote indication.

4. REMARKS
   - O - Comments, definitions, abbreviations, explanations, formulas.

5. CONDITION STUB
   - R - All or part of a condition statement, i.e., a logical question, or relational or state condition that is answerable by a yes or no.
   - O - The condition operators may be represented by symbols (=, $>$, $\leq$), mnemonics (EQ, LT, GT) or words (is equal to, is less than).

6. CONDITION ENTRY
   - R - Completion of the condition statement (extended entry) or Y, N, ., -, or blank (limited entry).
   - O - An "all others" or "else" rule to provide an unconditional rule to be taken when none of the other rules are satisfied.

7. ACTION STUB
   - R - All or part of an action statement. An explicit statement of where to go next for each rule, if not specified in the table header.
   - O - Any language may be used to specify the actions to be taken; i.e., input-output, sequence control or value assignments. Arithmetic expressions may be written using normal mathematical notation.

8. ACTION ENTRY
   - R - Completion of the action statement (extended entry), or X, ., -, or blank (limited entry).
   - O - Any language may be used to complete the action statement.

F20-8102 Printed in U. S. A.

IBM®
International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N. Y. 10601