File

SYSTEMS
ENGINEERING
SERVICES

D R A F T M A N U A L

TABULAR DESCRIPTIVE LANGUAGE

January 15, 1962
Technical Report No. 2A2

Thomas B. Glans
Burton Grad

INTERNATIONAL BUSINESS MACHINES CORPORATION
White Plains, New York

IBM Data Processing Division
200 Mamaroneck Avenue
White Plains, New York

January 17, 1962

To: Recipients of the Tabular Descriptive Language Manual

Subject: Corrections and Additions

The manual that you received is incomplete in a number of ways. Included within this package are a cover page, preface, table of contents and an errata sheet containing a list of the errors we have found to date. The corrections are in the same order as the manual itself and therefore can be made easily.

Three sections of the manual have not yet been completed: Chapter 3 which includes two sample problems and their solutions, using the Tabular Descriptive Language; Chapter 2, Example 8, which defines the use of the input-output operations (input-output is described in Chapter 6); and finally, Chapter 7, which defines the way in which the various forms used with TDL are written.

A complete revision of the manual is currently under way. It is anticipated that this will be printed by the 26th of January. These revised manuals will include those portions not currently in your hands and will be sent to you as soon as available.

Your comments, suggestions and recommendations are most welcome and should be submitted through Mr. D. Hradecky, Education Planning, Endicott. Any forms that you require can also be obtained from Education Planning. I hope that you will find the Tabular Descriptive Language a useful tool for system development and documentation.

Thomas B. Glans

Thomas B. Glans
Systems Engineering Services

TBG:mam
Enclosures

PREFACE

There is increasing evidence to indicate that decision tables are a useful means of analyzing and portraying systems logic. With this in mind the Tabular Descriptive Language was developed to encourage experiments, using decision tables for a wide variety of business problems. The feedback from the users of the Tabular Descriptive Language will provide a reliable evaluation of the usefulness of decision tables in problem analysis and documentation.

The Tabular Descriptive Language is designed to permit systems engineers to clearly and effectively think through the systems design logic, record this logic, and then communicate it to other systems engineers or programmers. The primary purpose is man-to-man communication, as opposed to man-to-machine communication, for which programming languages are designed. Various levels of system description can be made using the Tabular Descriptive Language because of the hierarchical nature of the language, that is, operands need be specified only to the level of detail necessary for clear communication, and certain procedures need merely be referenced in the decision tables without portraying the explicit logic.

This initial version of the Tabular Descriptive Language will provide a foundation upon which future developments can be built as the needs for expansion are identified. Therefore, the evaluation, comments and recommendations of users will serve to guide future development, so that the full potential of an advanced systems language can be realized within the near future.

A word about this manual. It is directed primarily at people who will be experimenting with the language. Chapter 1 presents a quick sketch of the basic ideas of data processing and decision tables, along with the environment in which the Tabular Descriptive Language is to be used. Chapter 2 covers the essentials of the Language, using a set of graded examples to introduce a few new ideas at a time. There are also review questions and answers after each example. Chapter 3 contains two problems worked through in the Tabular Descriptive Language. Chapters 4, 5 and 6 are the reference portion. Chapter 4 defines the structure of the language, Chapter 5 the operands and Chapter 6 the decision table operations. Chapter 7, the Appendix, indicates the mechanics of filling out the various forms needed to describe a system.

We wish to thank Messrs. G. H. Bang, R. N. Barnes, W. L. Kelly, A. A. Lea, and W. M. Selden for their contributions and suggestions in defining the Tabular Descriptive Language, Mr. J. L. Mannasmith for developing the problem solutions of Chapter 3, Mr. L. Weinstein for his development of sample problems and overall review of the language, Miss S. Troy for developing sample problems in Chapter 2 and preparing teaching material and Mr. E. L. Blossom for his assistance in rewriting the material in the reference portion of the manual.

It is expected that improvements will be made to the language as suggestions are received from users. Since the Tabular Descriptive Language is for man-to-man communication there is no implication that

IBM will produce a processor to translate this language to any IBM data processing machine.

Thomas B. Glans

Burton Grad

January 17, 1962

TABLE OF CONTENTS

CHAPTER 4 - The Structure of the Language

CHAPTER 5 - Operands

CHAPTER 6 - Decision Table Operations

TABULAR DESCRIPTIVE LANGUAGE
(Technical Report No. 2A2)

ERRATA #1 January 17, 1962

Page	Paragraph	Line	
1.7	1	12	...names of files , records...
1.8		5	Commonly understood names can be...
2.4		9	operand 1, operator__, operand 2
2.8	3	1	Using this form the complete...
2.10		12	...a quantity in the first
2.11		Q 2	What action__is...
		Q 3	EQ...operator
2.17		3	B + A → B
		4	B - A → B
		5	B * A → B
		6	B/A → B
2.18	2	1	Action row 8, in...
2.19	Ans.1		...go to table GETX
2.20	3	9	...action in row 5
2.21		6	...introduces
2.22		11	...range 22-24, the...
2.23	2	5	be preceded by...
2.24	Left	Ser. 4	... QUAN SOLD: CT * CUST DISC:WORK A
2.24	1	4	To refer to these fields the user must qualify the field name with the record name WORK A, e.g., QTY ON HAND: WORK A

Page	Paragraph	Line	
2.24	3	2	field, AMT EXT_, from...
2.25	3	4	For example, serial 6 of...
2.25	Fig. 5c	Ser. 4	QUAN SOLD N + 4
2.25	Last	4	...serial 5 of figure 5b,...
2.27		Q 4	delete question 4 and answer 4
2.28	4	4	...and GO TO table 016.
2.28	4	7	no identifying marks...
2.29		2	...LOC NO, DEPT, EMP NO, and EMP NAME are
2.31	Ans.	3	...of numeric valued operands with
	Ans.	5	...the bond purchase price is placed..
	Ans.	5	and the table is repeated.
2.32	2	10	DEPT EXP__ REPT.
		12	...is preceded by...
2.33	1	4	is preceded by...
2.35	2	13	...DEPT NO. From...
		16	...field (the right-most) is in...
	3	4	...is in printwheel _ position 64.
5.6	6	2	...as alphameric, the entire field is alphameric.
5.12	1	5	an alphameric literal is enclosed...
	3	2	...The value of an expression ____ is always...
5.21	2	8	All examples in single quotes (').
5.22		1	...ZERO_S
		6	...zero_s,...

Page	Paragraph	Line	
6.1		2	...to be described by the
6.3	4	2	...(2) High-order zeros are added
6.4	2	3	...is performed position by position from
6.6		7	...and these operands:
6.8		22	Value_ List Name
		23	Value_ List
6.12	4	5	Delete Record Name
6.13		15	...operands are discussed...
6.15		13	...be used with numeric or alphameric operands.
6.27		1	The first two forms shown are limited entry, the last four are
	Last		...the other six are...
6.32	Last		PER ASSOC letter FOR ASSOC letter PER ASSOC letter FOR ASSOC letter
6.46		4	X, ., or blank
		5	X, ., or blank
		7	X, ., or blank
		8	X, ., or blank
6.47		2	X, ., or blank
6.52		3	X, ., or blank
		16	X, ., or blank

SYSTEMS ENGINEERING SERVICES

PRELIMINARY MANUAL

TABULAR DESCRIPTIVE LANGUAGE

January 25, 1962
Technical Report No. 2A5

Thomas B. Glans
Burton Grad

INTERNATIONAL BUSINESS MACHINES CORPORATION
White Plains, New York

PREFACE

There is increasing evidence to indicate that decision tables are a useful means of analyzing and portraying systems logic. With this in mind the Tabular Descriptive Language was developed to encourage experiments using decision tables for a wide variety of business problems. The feedback from the users of the Tabular Descriptive Language will provide a reliable evaluation of the usefulness of decision tables in problem analysis and documentation.

The Tabular Descriptive Language is designed to permit systems engineers to clearly and effectively think through the systems design logic, record this logic, and then communicate it to other systems engineers or programmers. The primary purpose is man-to-man communication, as opposed to man-to-machine communication, for which programming languages are designed. Various levels of system description can be made using the Tabular Descriptive Language because of the hierarchical nature of the language, that is, operands need be specified only to the level of detail necessary for clear communication, and certain procedures need merely be referenced in the decision tables without portraying the explicit logic.

This initial version of the Tabular Descriptive Language will provide a foundation upon which future developments can be built as the needs for expansion are identified. Therefore, the evaluation, comments and recommendations of users will serve to guide future development, so that the full potential of an advanced systems language can be realized within the near future.

A word about this manual. It is directed primarily at people who will be experimenting with the language. Chapter 1 presents a quick sketch of the basic ideas of data processing and decision tables, along with the environment in which the Tabular Descriptive Language is to be used. Chapter 2 covers the essentials of the Language, using a set of graded examples to introduce a few new ideas at a time. There are also review questions and answers after each example. Chapter 3 contains two problems worked through in the Tabular Descriptive Language. Chapters 4, 5 and 6 are the reference portion. Chapter 4 defines the structure of the language, Chapter 5 the operands and Chapter 6 the decision table operations. Chapter 7, the Appendix, indicates the mechanics of filling out the various forms needed to describe a system.

We wish to thank Messrs. G. H. Bang, R. N. Barnes, W. L. Kelly, A. A. Lea, and W. M. Selden for their contributions and suggestions in defining the Tabular Descriptive Language, Mr. J. L. Mannasmith for developing the problem solutions of Chapter 3, Mr. L. Weinstein for his development of sample problems and overall review of the language, Miss S. Troy for developing sample problems in Chapter 2 and preparing teaching material, and Mr. E. L. Blossom for his assistance in rewriting the material in the reference portion of the manual.

It is expected that improvements will be made to the language as suggestions are received from users. Since the Tabular Descriptive Language is for man-to-man communication there is no implication that

IBM will produce a processor to translate this language for any IBM data processing machine.

Since this is a first attempt to formalize the language used for man-to-man communication there have been many technical difficulties. As a result, there may need to be further changes and improvements as feedback is received from users. Please note, therefore, that this is a preliminary manual which will need to be technically modified and rewritten after there has been adequate experience.

Thomas B. Glans

Burton Grad

January 17, 1962

TABLE OF CONTENTS

CHAPTER 4 - The Structure of the Language

CHAPTER 5 - Operands

CHAPTER 6 - Decision Table Operations

CHAPTER 7 - Appendix

CHAPTER I

A Tabular-Form Descriptive Language

Suppose you have been assigned to a team that is to set up a data processing system for some application in payroll, or inventory control, or utility billing, or insurance, or even in an area of science or engineering. What do you need to know about data processing in order to use the Tabular-form Descriptive Language (TDL) on such a job?

To begin with, we can mention a few areas of knowledge that are not needed. You need no knowledge of electronics. You need no knowledge of mathematics beyond high school algebra (unless, of course, the problem itself is mathematical). With the Tabular-form Descriptive Language you do not even need a detailed knowledge of how any particular computer system works. However, you do need to know certain facts about data processing, and eventually, if you work with the subject for a while, you will pick up certain detailed facts about particular computers - but you do not need these now. For now, the general ideas which you should have are discussed below.

A data processing system is composed functionally of five parts, as shown in Figure 1. The input section accepts information "from the outside," and converts it into the electronic form in which it is manipulated and stored internally. Externally, information is typically recorded on punched cards, punched paper tape, or magnetic tape. In some applications, printed characters can be read directly. Presently-used business machines cannot recognize handwriting or speech. The output section of a computer has the obvious function of converting from the internal representation to some convenient external form, such as printing, punched cards, magnetic tape, punched paper tape, or a variety of specialized media. Though the speeds of all

---

This introductory information is reprinted from the IBM Commercial Translator Manual; it also seems appropriate for this teaching text. "Tabular-form Descriptive Language" has been substituted for "Commercial Translator" throughout.

these devices are much greater than those of manual devices, they are still generally quite slow compared to speeds of internal electronic manipulation. The kind and number of input and output devices naturally depends on the particular machine and its application.

The storage section of a computer serves two important purposes. The obvious function is to hold the data on which we wish to operate. A function less obvious to the newcomer is to hold coded instructions which we place there to specify the procedure we wish to follow. A collection of such instructions is called a program. There are usually two types of storage. One type, though very fast, is of limited capacity and quite expensive; it is called main storage. What is frequently termed auxiliary storage can hold much more information, but is substantially slower.

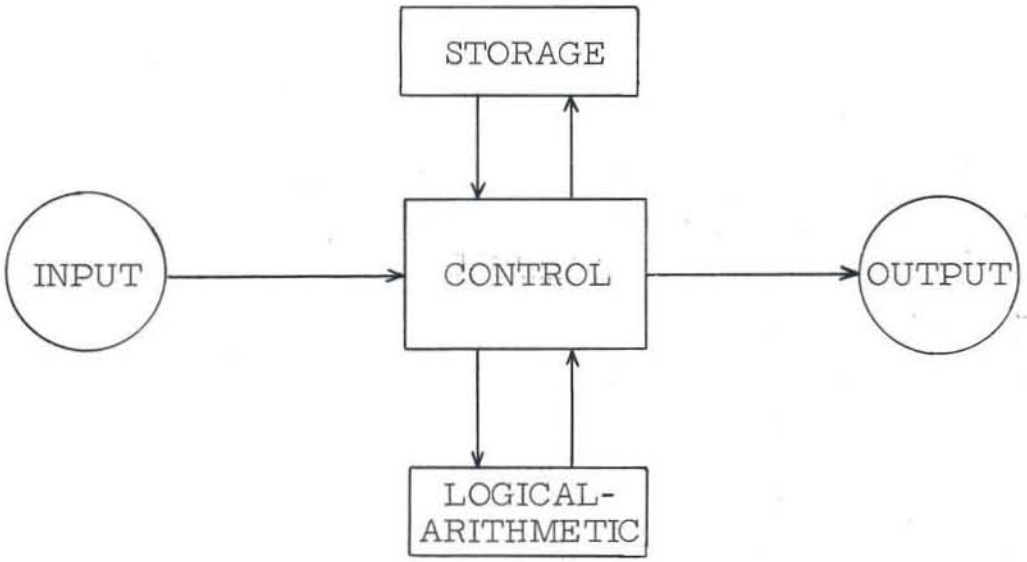


Figure 1. Schematic representation of the functional parts of a computer.

The last two sections of a computer are called the logical-arithmetic section and the control section. The actual data processing is done in the logical-arithmetic section, and the control section is needed to decode and interpret the instructions in storage.

A most important feature of modern data processing machines is the way instructions are held in main storage right along

with the data. For this reason we speak of a stored-program machine. The machine is therefore able to bring in new instructions when needed so that extremely large and complex problems can be solved.

The instructions which a data processing machine can execute naturally vary from one machine to another, but they can still be grouped into general categories. One group is used for arithmetic operations, another for making the elementary "decisions" of which a data processing machine is capable. Still another group covers input/output operations and a fourth group carries out miscellaneous control functions which are required because of the way the machine operates. Most individual operations are quite elementary, requiring a large number of them be combined properly in order to carry out a meaningful data processing task. This work, which follows the complete definition of the processing task, is called programming.

Data processing requires an extremely precise statement of the problem. We must not say "less than 30" if we mean "less than or equal to 30." There is no way we can say, "make sure the data looks reasonable;" if we want to check the validity of data, we must specify exactly what tests are to be made on it.

With electronic data processing, we are required to detail our procedures in advance to a degree not found in other methods. If we were asking a clerk to do a job, we might end by saying, "and if you run into anything you don't know how to handle, call me and we'll figure out what to do." In order to do a similar thing with a data processing machine, it is necessary first to define precisely what constitutes an exception, and then to write a procedure to handle it.

Decision Tables

Of the various activities that go into setting up a data processing procedure for a computer, the hardest is obtaining this precise definition of exactly what is to be done under all combinations of circumstances. Every step must be specified. The conditions under which each action

is to be performed must be enumerated. The exceptions to normal processing must be identified. Necessary sequences of operations must be precisely indicated.

Determining what is required of the computer system is called analysis; deciding just how to go about meeting these requirements is the area of system design; communicating the chosen procedures to the computer is called programming. In each of these areas a language is needed for defining the data processing procedures. Ideally, a language form or structure should be suitable both for man-to-man documentation, and for man-to-machine communication.

Many languages are used for these purposes. Procedures are often communicated to the machine in a form closely resembling the language of its own instructions, but this imposes a heavy and unnecessary burden on the person writing the procedures, since human language and machine language are quite different. Flow charts are widely used for man-to-man communication, but they can become confusing in complex situations, and the flow chart language is not suitable for communication with the machine. Logical equations are sometimes used, but they do not display relationships in as graphical a form as we might wish, and furthermore most system designers do not find them to be a comfortable form of expression.

Decision Tables seem to hold promise of meeting all of these needs of a computer language. Decision tables provide a graphical

representation of complex decision procedures in a way that is easy to visualize and understand. They show alternatives and exceptions much more explicitly than other languages. They present relationships among variables clearly. They show the necessary sequences of conditions and actions in an unambiguous manner. The language of decision tables can be used equally effectively for system analysis, procedure design, and computer coding. Thus, a computer procedure written as a set of decision tables is, to a large extent, its own documentation.

There is a growing body of experience to indicate that these claims are justified. Those who have used decision tables for man-to-machine work say that programming is much faster and that program checkout time is significantly reduced. They say that the use of tables leads to greater accuracy and completeness in problem formulation. They say that program maintenance is simpler, and that a program written in tabular form is indeed a powerful communication and documentation device.

For all these reasons it has seemed worth while to develop a Tabular-form Descriptive Language; i.e., a language using decision table format to structure the communication of procedures and systems solutions.

To our knowledge this is the first attempt to describe a man-to-man language with the kind of precision that computer languages possess,

while retaining the ability to describe the system logic rather than the details of a particular solution.

The Language Concept

In developing such a formal language there are three major considerations: Purpose, Structure and Content.

Purpose: The purpose of TDL is to permit systems engineers to clearly and effectively think through the system design logic, record this logic, and then communicate it to other systems engineers or programmers. It is not a machine-level or programmer-level language and it is not intended to be implemented automatically. TDL is expected to provide a clear statement of the solution logic so that a good programmer can prepare an efficient program. TDL must be usable at various levels of detail for all types of business data processing. It must, therefore, have features suitable to the various kinds of problems found in business systems: report preparation, decision making, arithmetic processes, file maintenance, etc.

With this broad a purpose, it is evident that the initial TDL can only establish a foundation; extensive work will be required to add extra features as they are identified, and clarify aspects for situations not currently visualized. Much, therefore, depends on those who use TDL at this experimental stage. Their evaluation, comments, and recommendations will serve to guide future development so that the full potential of an advanced systems language can be realized.

Structure and Content: The structure of the language sets up two classes of information:

(1) Operand Descriptions to enable the systems engineer to define the meaning and use of various field names, constants, etc.; and

(2) Decision tables to describe the data transformations.

While operand descriptions are common to many current machine-oriented languages like Autocoder and COBOL, the attempt here has been to allow a quite simple and natural way for identifying and specifying operand names only to the level of detail absolutely necessary for clear communication. The only real requirement is that names of files, records, fields, etc. be identified to avoid their inadvertent reuse for other operands. The balance of the descriptive material is only needed for actual communication to the programmer who has to layout and code the job for a particular machine.

To indicate how the idea and nature of the operand descriptions permit a high degree of flexibility and convenience, let's review a few of the features.

- Virtually any convenient name can be used for an item of data. Common names can be used in many records and distinguished by simply indicating the record name.

- Names can represent discrete functions of one or two arguments — such as, airplane fare between two points.

- Names can represent the value of complex arithmetic formulas.

- Names can represent Boolean-algebra type logical expressions.
- Names can represent lines of information resulting from putting together the values from different fields.
- Commonly accepted words can be used without definition, such as Branch Office Manager's name, current month, etc.
- Work fields which can be created and changed at will do not require formal definition.

The second basic element of the structure of the language is the one of decision tables. To understand this, let's look at the simple table shown in Figure 2.

	Rule 1	Rule 2		Rule 30
age	25-34	25-34		≥65
health	"excellent"	"excellent"		"poor"
section of country	"east"	"west"		"west"
rate per $1000	1.57	1.72		5.92
policy limit	200,000	200,000		20,000

Figure 2. An example of a decision table.

The first decision rule can be paraphrased: If age is between 25 and 34 inclusive, and health is excellent, and section of country is east, then rate per thousand dollars is 1.57 and policy limit is 200,000. The underlined words are implied by the table layout. The quote marks

are used to differentiate non-numeric values from variable names. The other rules are alternatives to this one, so that logically, it doesn't matter which rule is examined first; only one rule can be satisfied in a single pass through this decision table.

The information in Figure 2 is shown in an exploded view in Figure 4, to show more clearly the parts of a table and the terms that are used to describe them. The double lines serve as demarcation: CONDITIONS are shown above the horizontal double line, ACTIONS below; the STUB is to the left of the vertical double line, ENTRIES are to the right. Each vertical combination of conditions and actions is called a RULE. The essential nomenclature is completed by adding at the top of the table a title section, called a TABLE HEADER, and by adding a RULE HEADER over the entries.

Tables may be used in a slightly different way to state decision logic, as shown in Figure 3.

Rule No.	1	2	3	4
credit limit ok	Y	N	N	N
pay experience favorable		Y	N	N
special clearance obtained			Y	N
approve order	X	X	X	
return order to sales				X

Figure 3. A limited entry table

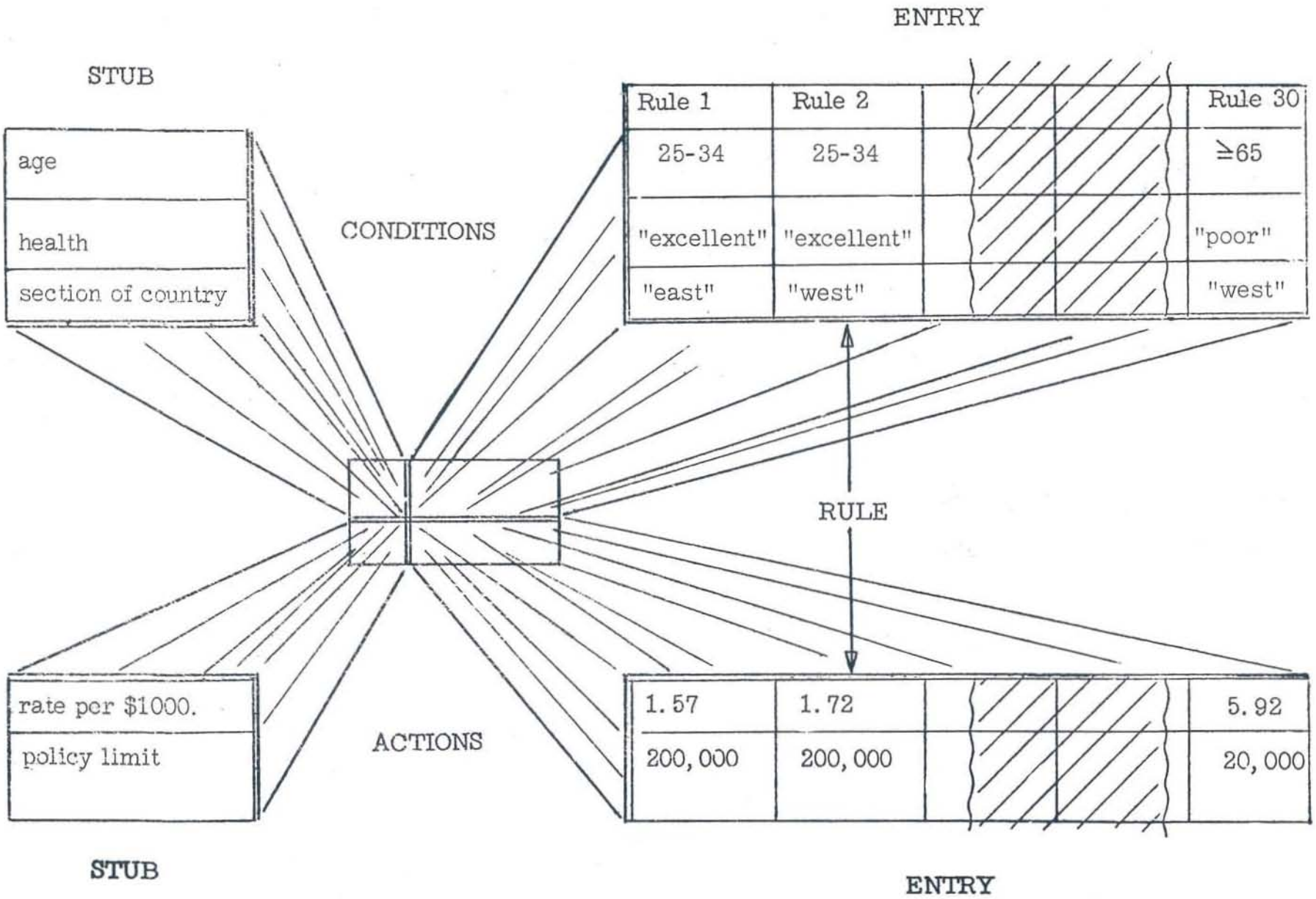


Figure 4. Exploded view of the table in Figure 2.

The first rule (columns 1 and 2) is read: If credit limit ok, then approve order. Again, the underlined words are implied by the form. You may note here that the form of the individual condition or action is somewhat different. In a limited entry condition or action, the entire condition or action must be written in the stub; the entry is limited to asserting, reversing, or ignoring a condition or executing an action. In contrast, an extended entry form (as in Figure 2) has part of the condition or action extended directly into the entry. Both forms may be used within one table, but any one condition or action row must be of just one form.

You may note from this example that the basic concept of a single rule in a table is quite straightforward, being based on the "if. . . then. . ." relationship. If A = B, and C is greater than 5, and . . . then assign the value 7 to X, and GO TO Table 10. If the conditions in rule 1 are not met, then try rule 2. If none of the rules succeeds, then something is wrong. If we have considered all significant possibilities, and we want to indicate, "Go ahead anyhow with a special routine," then we indicate "ELSE" in the last rule; otherwise, we provide for an automatic error routine. The flow chart in Figure 5 shows schematically the way in which a table is "executed." In practice the actual solution technique may vary, but the logical result remains the same.

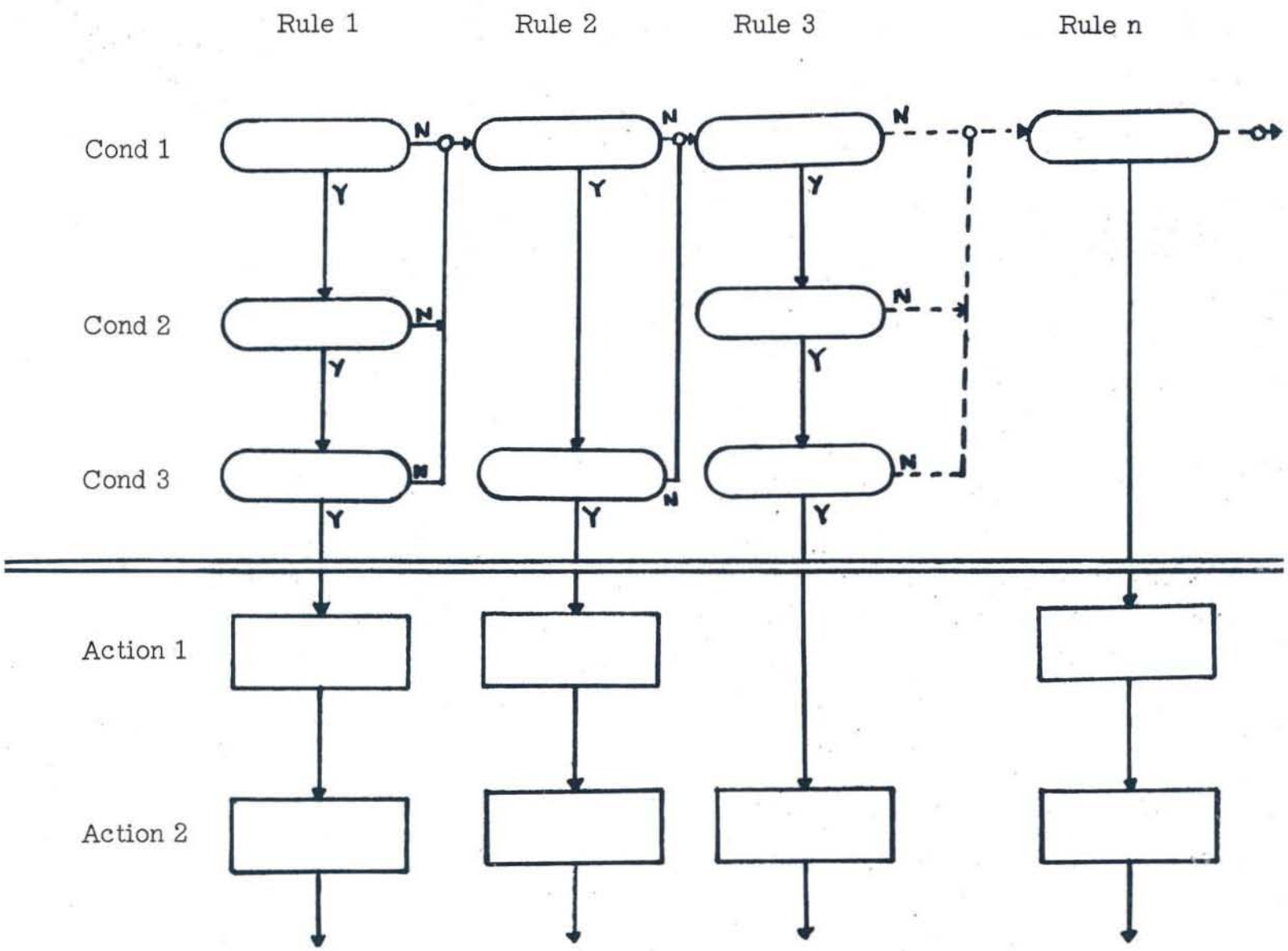


Figure 5. Schematic representation of the sequence of tests and actions in executing a table.

For user convenience decision tables may be referred to by name or identification code. Groups of tables are referenced by a procedure name. Until the job is actually turned over for programming many of these tables and procedures may remain undefined. They are simply an indication that such a routine will be supplied. This also offers the opportunity of using named tables to define a particular routine which is used many times.

Use of TDL

The general principles of decision tables have been applied to the development of a Tabular-form Descriptive Language (TDL) for man-to-man communication. As such, TDL is not a processable language, but it is intended to be precise and comprehensive. Certain operations which only have meaning for computer coding have been omitted and will have to be added by the programmer. The primary goal of TDL is to replace flow charts and narrative description of problem solutions and decision logic. It is not intended to replace block diagrams used to show physical relations or essentially pure referential processing work. But where there are significant conditions and branches, and where there are complex decisions involving many factors TDL permits more rapid, accurate, effective solution formulation and description with attendant savings in computer programming and machine space and running time.

TDL uses decision tables to describe the decision logic involved in a systems problem. The tabular form is essential to decision tables.

It provides a means of briefly, unequivocally and completely describing the requirements of an information processing system in a language which is easy to write and read. Because the language is also rigorously defined and the forms on which the tables are described have an effective structure, brevity is possible. The language content and structure enable the user to describe complex, logical solutions with a relatively small amount of writing. The nature of decision tables induce the user to make complete descriptions of the information processing requirements. As a corollary to this advantage is the fact that what is written in tabular language may be readily checked for completeness and correctness.

Who then will use TDL? Primarily the systems engineer. The systems engineer performs the function of systems analysis and development. While he may later act as a programmer, when he is doing systems analysis he is concerned with the system on a level above that of programming a computer. The system with which a systems engineer is concerned is some segment of a company's total information processing. It is the function of the systems engineer to analyze the requirements and to develop the necessary detail to describe the information processing system.

While no one seems to know just how to go about solving a particular data processing problem, the following steps always seem to come up:

1. Describe input and output fields, records, and files;
2. Describe printed reports, card and tape content;
3. Define constants, expressions, lists; and
4. Determine transformation logic to indicate preparation of outputs as a function of inputs.

There may be substantial iteration in these steps, but they have worked well enough in the past to serve as an organizing device for the Tabular-form Descriptive Language.

The systems engineer will describe the data which is input to or output from the system and the processing of the data which must be accomplished. Data processing consists variously of recording data without change, validating data to show that it is correct, operating on data based on certain conditions, and manipulating data arithmetically. It is obvious that a description of such processing does not require explicit computer instructions or computer hardware or that a particular computer be used; it's only necessary that the systems engineer describe fully what the data is and what is to be done to the data, in a language which is as close as possible to the language in which he thinks about the problem and its solution.

As the systems engineer determines what the inputs and outputs of the system which he is developing will be he may describe these using the operand description facilities of the Tabular-form Descriptive Language. As the systems engineer determines what processing must be done on the information in his system, he will describe this using decision tables.

Summary

This then is a general introduction to TDL, a Tabular-form Descriptive Language. Intended for precise man-to-man communication it is based on separating the operand definition from the transformation logic. The transformation logic is expressed in a tabular form for convenience, completeness, and clarity.

The balance of the TDL manual explains the details of the language and the forms used for describing systems solution in the language.

The next two chapters present a teaching text description of the language with a number of sample problems. The remainder of the manual provides a reference description of the language elements and forms.

CHAPTER 2

Using the Tabular Descriptive Language

The purpose of language is effective communication. Consider:

John is big

as a language statement. It satisfies the requirements of a communication cycle in that a completely defined thought is transmitted by symbols over a medium (paper) to a receiver (the reader) who understands it.

For convenience, the format should be standardized, although the symbolism may vary. Now consider:

Jean est gros

which has the same meaning as the earlier statement, except that the symbolism is slightly different. The format, however, remains exactly the same in that the leftmost something is related to the rightmost something by the something in the middle.

The Tabular Descriptive Language (TDL) likewise promotes effective communication from sender to receiver by conveying defined thoughts in symbolized language statements through a standard format. This chapter is addressed to an explanation and description of the TDL standard format.

In treating the subject, the examples and text will simply be illustrative of its potential, since a complete analysis of all the applications,

TYPE CODE	EQ	1								2							3							
SET MODEL CODE	EQ	'	C	N	Y	T	B	L	'	'	H	R	D	T	O	P	'	'	S	T	A	W	G	N	'

Figure 1. Decision Table, Example 1.

situations, and problem solving approaches which the language is capable of handling is too extensive for coverage here. Consequently, this chapter should be read thoroughly to acquire a general understanding of language techniques before proceeding to the more rigorous treatment of the language itself in later chapters.

Concepts of the language are introduced through a series of examples which grow progressively more difficult; the associated narrative points out and emphasizes the rules of format. Periodically a group of questions is inserted which permit the reader to test his understanding of the material he has just studied. In this manner, the reader may proceed to succeeding pages at his own pace.

Example 1 (all figures will appear on left facing pages and will be repeated to coincide with the narrative) is an excerpt from one form used in TDL, showing a simplified version of a decision table. It might have applied to a situation where it was necessary to convert code classifications of automobile styles to an understandable abbreviation. The table reads, in part:

IF the type code of the car body	i	s	1
THEN the model	i	s	a convertible

or

IF the type code of the car body	i	s	2
THEN the model	i	s	a hardtop

etc.

TYPE CODE	EQ	1								2							3							
SET MODEL CODE	EQ	'	C	N	Y	T	B	L	'	'	H	R	D	T	O	P	'	'	S	T	A	W	G	N	'

Figure 1. Decision Table, Example 1.

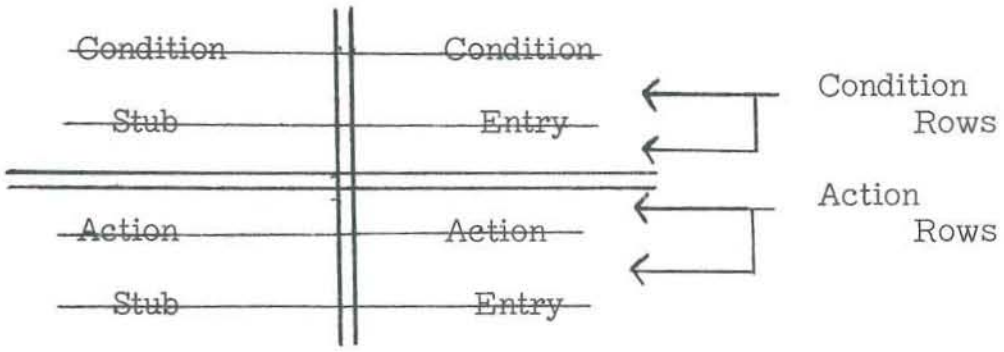
Each of these statements is known as a rule, which indicates the relationship between a condition and an action in tabular language.

IF a condition or set of conditions is satisfied,

THEN the corresponding action(s) are executed.

A rule further is composed of statements located in the 4 major areas of a decision table: condition stub, condition entry, action stub, and action entry, each one separated from the others by horizontal and vertical double lines.

Decision Table Structure



This sketch locates and identifies those major areas in a decision table where information may be assigned for later reference.

The number of rules included in a decision table is limited only by the requirements of the system. The condition stub and the action stub are common to all the rules, but condition and action entries vary from one rule to the next. In example 1, there is only one condition and one action for each of three distinct rules. In general, a decision table will involve a series of conditions with a related series of actions for each rule.

TYPE CODE	EQ	1								2							3							
SET MODEL CODE	EQ	'	C	N	Y	T	B	L	'	'	H	R	D	T	O	P	'	'	S	T	A	W	G	N	'

Figure 1. Decision Table, Example 1.

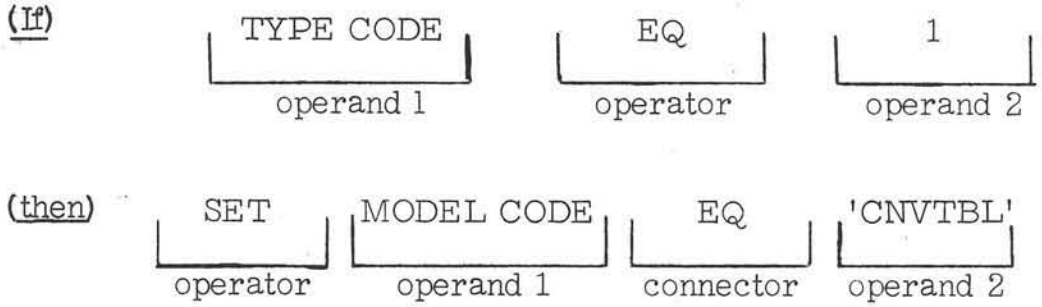
Each statement in a decision table is constructed from a series of components of two different types: operators and operands. Each has a distinct function depending upon the context of the statement. The method of their relationship in a condition row for a single rule is:

operand 1, operator, operand 2

The relationship in an action row is:

operator, operand 1, connector, operand 2.

To show this relationship, rule 1 (rules are numbered consecutively starting with the first entry column after the vertical double line) is written as follows:



Operands in TDL are of many forms. All have the common property of representing something and of being the subject or object of some statement. Consequently, they can be considered the equivalent of the noun in English grammar rules. With the same reasoning, the operator may be assumed to take the place of the verb. A condition statement, then, can be considered in the form:

noun (operand) verb (operator) noun (operand)

In rule 1, TYPE CODE and MODEL are operands which are the names of certain fields. These operands are known as field names and

TYPE CODE	EQ	1								2								3							
SET MODEL CODE	EQ	'	C	N	Y	T	B	L	'	'	H	R	D	T	O	P	'	'	S	T	A	W	G	N	'

Figure 1. Decision Table, Example 1.

represent data variables. These operands come under the general category of named operands, and identify information that is peculiar to the system. In the same rule, the symbols 1 and 'CNVTBL' are also operands. However, these operands are known as literals since they represent literally that which they say, and are classified under the general category of self-describing operands. These may be used universally for all systems without additional definition since they directly communicate the value of the operand.

The distinction between a named operand and a self-describing operand is important, since one implies a variable value and the other a fixed value. Therefore, a method must be devised for differentiating between them. This is accomplished by the restriction that a named operand must start with a letter. Literals, when they are alphameric (as 'CNVTBL') are enclosed in quotation marks, so that they are not mistaken for named operands. Numeric literals need not be enclosed in quotation marks as they cannot be mistaken for field names.

In this table, two kinds of operators are displayed, one each in condition and action statements:

- The first, written as EQ, is referred to as a condition operator. Functionally, it relates the field name TYPE CODE to the numeric literals, 1, 2, or 3 in the 3 rules and therefore acts as a relational operator.
- The second, recognized by the SET... EQ... notation is an action statement. It performs the function of des-

TYPE CODE	EQ	1								2							3							
SET MODEL CODE	EQ	'	C	N	Y	T	B	L	'	'	H	R	D	T	O	P	'	'	S	T	A	W	G	N	'

Figure 1. Decision Table, Example 1.

cribing actions to be followed if the conditions of a rule are satisfied. For this reason, it is more specifically known as an assignment operator.

In this case, the action is to transfer one of the abbreviations ('CNVTBL', 'HRDTOP', 'STAWGN') to the field named MODEL CODE, depending upon the rule satisfied.

The manner in which a decision table is manipulated consists of investigating individual rules until the conditions for a rule are satisfied. Either rule 1 may be satisfied or rule 2, or rule 3, but not more than one at a particular time.

QUESTIONS

1. How do you read the 3 rules in Example 1 using English grammar?
2. What happens if rule 3 is considered before rule 2 in Example 1?
3. What is the difference between a literal and a field name? How is the difference shown in a table?
4. What does the action statement SET...EQ... call for in this example?

TYPE CODE	'EQ	1							2							3							
SET MODEL CODE	EQ	'	C	N	Y	T	B	L	'	'	H	R	D	T	O	P	'	'	S	T	A	W	G	N	'

Figure 1. Decision Table, Example 1.

ANSWERS

1. Rule 1: If the type code of the car body is 1, then the model is a convertible.

   Rule 2: If the type code of the car body is 2, then the model is a hardtop.

   Rule 3: If the type code of the car body is 3, then the model is a station wagon.

2. There is no change in the results. The rules may be investigated in any order, regardless of position in the table.

3. The field name represents a variable that changes according to the information placed in it. A literal is its own value. Alphameric literal names, which could be mistaken for field names are differentiated in the table by enclosing them within quotation marks (' ').

4. The SET...EQ... action moves an abbreviation, which is a literal that describes the car body, to the field named MODEL CODE.

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAG
PROXY REPORT	—		HOLT	1/2/62	

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2 / OP 02 OPERAND 2 / OP 03 OPERAND 2
			01 02 03 04 05 06 07 08 / 09 10 11 12 13 14 15 16 / 17 18 19 20 21 22 23 24 25
** 005	SEL STKVALUE GRPS	OPEN	
1	TYPE STKHLDR CODE	EQ	01 / 02 / 03
2	SHARES	GR 100000	Y / / N
3	SET TYPE STKHLDER	EQ	'INDIV' / 'BANK' / 'BROKER'
4	MOVE STKVALUE	TO	INDIVIDUAL STOCK AMOUNT / BANK STOCK AMOUNT / BROKER STOCK AMOUNT
5	GO TO	KEY REPORT	X / X / X

REMARKS:

Figure 2. Decision Table, Example 2.

Example 2 shows the complete TDL Decision Table form. Here, the situation deals with a proxy report, especially that part dealing with the selection of stockholder groups.

The heading, SYSTEM NAME, is filled in with the particular name, PROXY REPORT. A system description consists of a group of decision tables, each of which is assigned a name and an identification. The entire system description defines a complete information processing task. A decision table defines a segment of that analysis. In this case, the TABLE NAME is SEL STKVALUE GRPS and the TABLE IDENTIFICATION (usually a 3 digit number) is 005, but the table is a part of the overall system called PROXY REPORT.

A system description, composed of many decision tables, must have some method of indicating the sequence of going from one table to the next. This is performed through the use of the operator GO TO in the last action row. The operand for this operator is the table name KEY REPORT. (The table identification could also have been used as the operand.) It is assumed that some previous table or tables have been completed for this system and the last action of the previous table considered was GO TO SEL STKVALUE GRPS, the name of this table. Likewise, after this table is completed, the last action called for is a transfer of control to the table named KEY REPORT. At the termination of the processing, transfer of control is made to a table named STOP.

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAG
PROXY REPORT	—		HOLT	1/2/62	

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2 / OP 02 OPERAND 2 / OP 03 OPERAND 2
			01 02 03 04 05 06 07 08 / 09 10 11 12 13 14 15 16 / 17 18 19 20 21 22 23 24 25
** 005	SEL STKVALUE GRPS	OPEN	
1	TYPE STKHLDR CODE	EQ	01 / 02 / 03
2	SHARES	GR 100000	Y / / N
3	SET TYPE STKHLDER	EQ	'INDIV' / 'BANK' / 'BROKER'
4	MOVE STKVALUE	TO	INDIVIDUAL STOCK AMOUNT / BANK STOCK AMOUNT / BROKER STOCK AMOUNT
5	GO TO	KEY REPORT	X / X / X

REMARKS:

Figure 2. Decision Table, Example 2.

This table is classified as an open table. All open tables must be entered by a GO TO operator and must contain a GO TO operator as the last action of the table, the only exception to this rule being the STOP table.

This example illustrates another difference in format as compared to Example 1. In the previous example, the second operand of all conditions and actions were written in the entry portion of each rule, a fact that classifies it as an extended entry table.

Where the complete condition is written in the stub of the table, and the affirmation or negation of the condition is indicated by a Y or an N in the rule column, the table is referred to as limited entry. It is possible that a rule does not require the condition. In this case, the column has no entry. For example, the second condition row for the three rules of this table could be roughly translated:

Rule 1. the number of shares is more than 100000

Rule 2. (regardless of the number of shares)

Rule 3. the number of shares is not more than 100000

The limited entry format may also apply to actions. In this case, the complete action is written in the action stub, and is to be executed if an X is placed in the column for that rule. If the action is not pertinent to a rule (not to be executed), the entry for that rule is left blank. In this example, all three rules require the action GO TO KEY REPORT. Since

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAG
PROXY REPORT	—		HOLT	1/2/62	

TABLE ID / SER	TABLE NAME / OPER	OPERAND 1	TABLE TYPE / OP	OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2
					01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24 25
** 005	SEL STKVALUE GRPS		OPEN				
1		TYPE STKHLDR CODE	EQ		01	02	03
2		SHARES	GR	100000	Y		N
3	SET TYPE STKHLDER		EQ		'INDIV'	'BANK'	'BROKER'
4	MOVE STKVALUE		TO		INDIVIDUAL STOCK AMOUNT	BANK STOCK AMOUNT	BROKER STOCK AMOUNT
5	GO TO		KEY REPORT		X	X	X

REMARKS:

Figure 2. Decision Table, Example 2.

this action applies for all three rules, an X is written in the action entry for each rule. To qualify as either limited or extended entry, a decision table must follow the arrangement rules for each consistently.

A third tabular format is known as mixed entry. Example 2 is a mixed entry table. That is, some of the conditions and actions are in extended entry form while others are in limited entry form. The only restriction in the use of mixed entry is that a single condition row or action row must be consistent. It must be either all limited entry or all extended entry. The two cannot be intermixed in one condition or action row.

The condition part of rule 1 asks: ... "Is the TYPE STKHLDR CODE equal to 01 and are the number of shares greater than (GR) 100000?"... The operator GR is another relational operator. In order for the second condition to be satisfied, the question "Are the number of shares owned greater than 100000?" must be answered yes in rule 1. The full list of relational operators, with their abbreviations, is given below:

LR less than (lesser)	LE less than or equal to
GR greater than	GE greater than or equal to
EQ equal to	NE not equal to

These relational operators could be expressed symbolically (e.g., >, =), but the alphabetic notation will be followed throughout this manual.

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAG
PROXY REPORT	—		HOLT	1/2/62	

TABLE ID / SER	TABLE NAME / OPER OPERAND 1	TABLE TYPE / OP OPERAND 2	FREQUENCY / OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2
			01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24 25
** C005	SEL STKVALUE GRPS	OPEN			
1	TYPE STKHLDR CODE	EQ	01	02	03
2	SHARES	GR 100000	Y		N
3	SET TYPE STKHLDER	EQ	'INDIV'	'BANK'	'BROKER'
4	MOVE STKVALUE	TO	INDIVIDUAL STOCK AMOUNT	BANK STOCK AMOUNT	BROKER STOCK AMOUNT
5	GO TO	KEY REPORT	X	X	X

REMARKS:

Figure 2. Decision Table, Example 2.

Rule 2 reads:

IF the type stockholder code is 2

THEN the type of stockholder is a bank

and

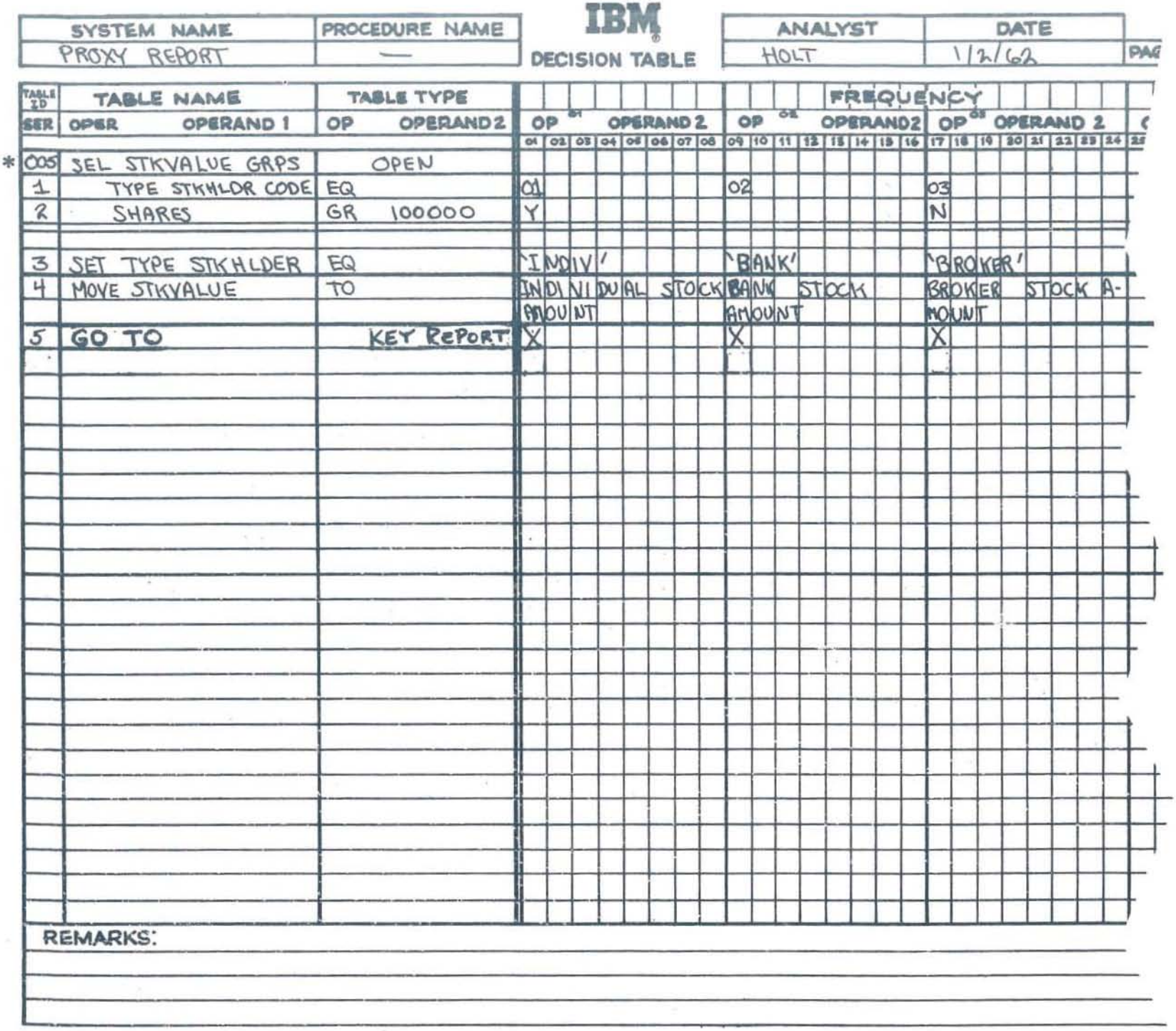
move the stock value to bank stock amount

and

the table KEY REPORT is entered.

This rule introduces the MOVE operator with the connective TO which works as an assignment operator just like the SET...EQ... action. It moves the quantity specified by the first operand (STKVALUE) to the field named in the second operand (BANK STOCK AMOUNT). The differences between the SET and MOVE assignment operators may be stated as follows: SET...EQ... is used when one of several quantities in the second operand field is to be transferred to the first; MOVE... TO... is used when a quantity in the first operand field is to be transferred to one of several fields named by the second operand.

In the action row containing the MOVE operator, the entries are INDIVIDUAL STOCK AMOUNT, BANK STOCK AMOUNT, BROKER STOCK AMOUNT. If the names do not conveniently fit a line, they may be hyphenated or broken where most convenient. They are then continued on the second line of the rule, as illustrated in this example.



IBM DECISION TABLE

SYSTEM NAME	PROCEDURE NAME	ANALYST	DATE
PROXY REPORT	—	HOLT	1/2/62

TABLE ID / SER	TABLE NAME / OPER OPERAND 1	TABLE TYPE / OP OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2
** 005	SEL STKVALUE GRPS	OPEN			
1	TYPE STKHLDR CODE	EQ	01	02	03
2	SHARES	GR 100000	Y		N
3	SET TYPE STKHLDER	EQ	'INDIV'	'BANK'	'BROKER'
4	MOVE STKVALUE	TO	INDIVIDUAL STOCK AMOUNT	BANK STOCK AMOUNT	BROKER STOCK AMOUNT
5	GO TO	KEY REPORT	X	X	X

REMARKS:

Figure 2. Decision Table, Example 2.

QUESTIONS

1. Read rule 3
2. What action is called for by the MOVE...TO... action in rule 3?
3. How does the action of the MOVE...TO... compare with the action of the SET...EQ...?
4. In rule 1, if the number of shares is equal to 100000, are the conditions of the rule satisfied? Why or why not?
5. What are the requirements of an open table?

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAGE
PROXY REPORT	—		HOLT	1/2/62	

TABLE ID / SER	TABLE NAME / OPER / OPERAND 1	TABLE TYPE / OP / OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2
			01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24 25
** 005	SEL STKVALUE GRPS	OPEN			
1	TYPE STKHLDR CODE	EQ	01	02	03
2	SHARES	GR 100000	Y		N
3	SET TYPE STKHLDER	EQ	'INDIV'	'BANK'	'BROKER'
4	MOVE STKVALUE	TO	INDIVIDUAL STOCK AMOUNT	BANK STOCK AMOUNT	BROKER STOCK AMOUNT
5	GO TO	KEY REPORT	X	X	X

REMARKS:

Figure 2. Decision Table, Example 2.

ANSWERS

1. If the type number of the stockholder code is 03, and if the number of shares is not greater than 100000, then the type of stockholder is a broker, and broker stock amount receives the stock value, and another table called KEY REPORT is entered.
2. The stock value is placed in the data field broker stock amount.
3. Their action is opposite. MOVE transfers a quantity from the first field to one of several fields named by the second operand, while SET transfers quantities from one of the possible second operand fields to a field named by the first operand.
4. No. The condition used with the relational operator is that the number of shares must be greater than 100000. Equal to this number does not satisfy the condition of the rule.
5. An open table is entered by a GO TO operator, and must conclude with a GO TO operator.

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY																											
USER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2			
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 005	SALES EXTENSION	OPEN																														
1	INVENTORY TRANSACTIONS	IS END OF FILE	N								N								N								Y					
2	ITEM NO: SALES	EQ ITEM NO: INVEN	Y								Y								N													
3	QTY SOLD	LE QTY ON HAND	Y								N																					
4	SET SALES QTY SHIP	EQ			QTY SOLD								QTY ON HAND																			
5	SET QTY BACKORDER										EQ		QTY SOLD																			
											-		QTY ON HAND																			
6	SET QTY ISSUE		+		QTY SOLD						+		QTY ON HAND																			
7	MOVE UNIT PRICE	TO SALES AMT	X								X																					
	* SALES QTY SHIP																															
8	SET QTY ON HAND		-		QTY SOLD						EQ		ZERO																			
9	GOTO		TRANS IN								TRANS IN								GETX								SHIPMENT RELEASE					

Figure 3a. Decision Table, Example 3.

Example 3 is concerned with an inventory and billing system. It applies to that part of the system which processes sales transactions against inventory, checks the quantity sold against the inventory record to establish the quantity to be shipped or back ordered, applies a price to the quantity shipped to extend the sales amount, and updates the inventory record by the quantity shipped in the sales transaction.

The first condition of the first rule is:

IS INVENTORY TRANSACTIONS AT END OF FILE . . . ?

This illustrates the use of a state conditional operator and a state term. The state conditional operator relates a state term to an operand name of something that varies. There are two such operators, IS and NT (is NoT). In this example, the first operand is the file name, INVENTORY TRANSACTIONS, and the operand which is related to it by the operator, IS, is the state term END OF FILE. This entry indicates that when the answer is yes, there are no more records available. The END OF FILE operand may also be abbreviated EOF.

Prior to discussing the remainder of this example, it is necessary to digress and discuss some of the types of named operands: files, records, and fields.

- A file is a collection of one or more records and has three characteristics:
  (1) medium - e.g., card, disk, etc.
  (2) directional type - input, output, retained, intermediate.
  (3) access type - serial, random

TABLE ID / SER	TABLE NAME / OPER	OPERAND 1	TABLE TYPE / OP	OPERAND 2	FREQUENCY
** 005	SALES EXTENSION		OPEN		
1	INVENTORY TRANSACTIONS		IS END OF FILE		N / U / N / Y
2	ITEM NO: SALES		EQ	ITEM NO: INVEN	Y / Y / N / Y
3	QTY SOLD		LE	QTY ON HAND	Y / N
4	SET SALES QTY SHIP		EQ	QTY SOLD	QTY ON HAND
5	SET QTY BACK ORDER				EQ QTY SOLD / − QTY ON HAND
6	SET QTY ISSUE		+	QTY SOLD	+ QTY ON HAND
7	MOVE UNIT PRICE		TO	SALES AMT	X / X
	* SALES QTY SHIP				
8	SET QTY ON HAND		−	QTY SOLD	EQ ZERO
9	GOTO		TRANS IN		TRANS IN / GETX / SHIPMENT RELEASE

Figure 3a. Decision Table, Example 3.

SYSTEM NAME: INVENTORY, BILLING

IBM

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY TRANSACTIONS	IN	SER-CARD	SALES	—	8000	10,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	QTY SOLD	N	+	4				1						
3	SALES QTY SHIP	N	+	4				1						
4	QTY BACK ORDER	N	+	4				1						
5	SALES AMT	N	U	4.2				1						

Figure 3b. File Description, Example 3.

SYSTEM NAME: INVENTORY, BILLING

IBM

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY MASTER	RETAINED	RAN-DISK	INVENTORY	INVEN	4,000	5,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	UNIT PRICE	N	+	3,2				1						
3	OLD BAL ON HAND	N	+	5				1						
4	QTY RECEIPTS	N	+	5				1						
5	QTY ISSUES	N	+	5				1						
6	QTY ON HAND	N	+	5				1						
7	QTY ON ORDER	N	+	5				1						
8	QTY MINIMUM	N	+	5				1						
9	ITEM DESCR	A		20	✓			1						

Figure 3c. File Description, Example 3.

- A record is a collection of one or more fields. It assumes the attributes of the file to which it belongs. A File Description form should be completed for every type of record, listing all the fields which make up the record.
- A field is a named piece of information with value and can be part of a record.

File names, record names, and field names are operands in a decision table. On figure 3b, the file description of a record of the file named INVENTORY TRANSACTIONS is shown. This form describes the record, SALES, of that file. The record is composed of five fields. The file name, INVENTORY TRANSACTIONS, the record name, SALES, and the field names, ITEM NO, QTY SOLD, SALES QTY SHIP, QTY BACK ORDER, and SALES AMT, may all be used as operands.

In figure 3c, a file description is shown for the record, INVENTORY, which is part of the file, INVENTORY MASTER. This record has nine fields. The headings for figures 3b and 3c are explained as follows:

System Name: The name of the system of which the file description is a part. In this example, both forms are part of the system INVENTORY, BILLING. The decision table is also part of this system.

File Name: The name of the file, as described above.

TABLE ID SER	TABLE NAME OPER	OPERAND 1	TABLE TYPE OP	OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
** 005	SALES EXTENSION		OPEN					
1	INVENTORY TRANSACTIONS		IS END OF FILE		N	W	N	Y
2	ITEM NO: SALES		EQ	ITEM NO:INVEN	Y	Y	N	
3	QTY SOLD		LE	QTY ON HAND	Y	N		
4	SET SALES QTY SHIP		EQ		QTY SOLD	QTY ON HAND		
5	SET QTY BACK ORDER					EQ QTY SOLD		
						- QTY ON HAND		
6	SET QTY ISSUE				+ QTY SOLD	+ QTY ON HAND		
7	MOVE UNIT PRICE		TO	SALES AMT	X	X		
	* SALES QTY SHIP							
8	SET QTY ON HAND				- QTY SOLD	EQ ZERO		
9	GOTO				TRANS IN	TRANS IN	GETX	SHIPMENT RELEASE

Figure 3a. Decision Table, Example 3.

IBM

SYSTEM NAME: INVENTORY, BILLING

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY TRANSACTIONS	IN	SER-CARD	SALES	—	8000	10,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	QTY SOLD	N	+	4				1						
3	SALES QTY SHIP	N	+	4				1						
4	QTY BACK ORDER	N	+	4				1						
5	SALES AMT	N	U	4.2				1						

Figure 3b. File Description, Example 3.

IBM

SYSTEM NAME: INVENTORY, BILLING

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY MASTER	RETAINED	RAN-DISK	INVENTORY	INVEN	4,000	5,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	UNIT PRICE	N	+	3,2				1						
3	OLD BAL ON HAND	N	+	5				1						
4	QTY RECEIPTS	N	+	5				1						
5	QTY ISSUES	N	+	5				1						
6	QTY ON HAND	N	+	5				1						
7	QTY ON ORDER	N	+	5				1						
8	QTY MINIMUM	N	+	5				1						
9	ITEM DESCR	A		20	✓			1						

Figure 3c. File Description, Example 3.

Dir. Type: One file is IN, the other RETAINED.

Medium: The medium on which the file is contained. INVENTORY TRANSACTIONS is a serial card file, and INVENTORY MASTER is a random disk file.

Record Name: The name of that record which is described by the form. In these cases, SALES and INVENTORY.

Abb: This entry allows for the abbreviation of long record names. Such abbreviations may be used as an operand rather than the full record name.

Average Records: The average number of the record described on the sheet.

Peak Records: The peak number of the record on the sheet.

After this information is inserted, the field names for the record described are then listed in the field name column. The fields need not be described in any particular order. For each field, the specification must be made as to

(1) whether the field is alphameric or numeric (A or N),

(2) the sign possibilities for numeric fields (+, -, E, U)

TABLE ID / SER	TABLE NAME / OPER	OPERAND 1	TABLE TYPE / OP	OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
** 005	SALES EXTENSION		OPEN					
1	INVENTORY TRANSACTIONS		IS END OF FILE		N	U	N	Y
2	ITEM NO: SALES		EQ	ITEM NO: INVEN	Y	Y	N	
3	QTY SOLD		LE	QTY ON HAND	Y	N		
4	SET SALES QTY SHIP		EQ		QTY SOLD	QTY ON HAND		
5	SET QTY BACK ORDER					EQ QTY SOLD		
						− QTY ON HAND		
6	SET QTY ISSUE				+ QTY SOLD	+ QTY ON HAND		
7	MOVE UNIT PRICE		TO SALES AMT		X	X		
	* SALES QTY SHIP							
8	SET QTY ON HAND				− QTY SOLD	EQ ZERO		
9	GOTO				TRANS IN	TRANS IN	GETX	SHIPMENT RELEASE

Figure 3a. Decision Table, Example 3.

SYSTEM NAME	IBM FILE DESCRIPTION	ANALYST	DATE	PAGE OF PAGES
INVENTORY, BILLING		FRANCIS	1/2/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY TRANSACTIONS	IN	SER-CARD	SALES	—	8000	10,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	QTY SOLD	N	+	4				1						
3	SALES QTY SHIP	N	+	4				1						
4	QTY BACK ORDER	N	+	4				1						
5	SALES AMT	N	U	4.2				1						

Figure 3b. File Description, Example 3.

SYSTEM NAME	IBM FILE DESCRIPTION	ANALYST	DATE	PAGE OF PAGES
INVENTORY, BILLING		FRANCIS	1/2/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY MASTER	RETAINED	RAN-DISK	INVENTORY	INVEN	4,000	5,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	UNIT PRICE	N	+	3,2				1						
3	OLD BAL ON HAND	N	+	5				1						
4	QTY RECEIPTS	N	+	5				1						
5	QTY ISSUES	N	+	5				1						
6	QTY ON HAND	N	+	5				1						
7	QTY ON ORDER	N	+	5				1						
8	QTY MINIMUM	N	+	5				1						
9	ITEM DESCR	A		20	✓			1						

Figure 3c. File Description, Example 3.

(3) the maximum total length for that field (and whether the field is of fixed length or variable length) and finally,

(4) the maximum number of times that each field will appear on each record.

Serial 5 on figure 3b illustrates these points. The field name is SALES AMT which is a numeric value (N) of length 4.2. The number 4.2 indicates four integer and two decimal positions. If only an integer value is specified, then it represents the total number of characters. The field name SALES AMT occurs only one time per record.

Serial 9 on figure 3c also amplifies the discussion. The field name is ITEM DESCR, which has an alphameric value. Therefore, no sign possibilities need be noted. The maximum length of the field is 20, but it is variable. This field occurs only one time per record.

The second condition of the first rule of the decision table asks: Is the ITEM NO of the SALES record the same as the ITEM NO of the INVENTORY record...? This is a case where field names are qualified by their respective record names. In other words, the field name is given (ITEM NO) and then followed by a colon (:) and the record name (SALES). The field names are qualified because each of the two types of records contain identical field names. If the field names of a record are unique (never used in any other part of the system) then the name may be used without any qualification. For example, the third condition of rule 1 is:

TABLE ID / SER	TABLE NAME / OPER	OPERAND 1	TABLE TYPE / OP	OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
** 005	SALES EXTENSION		OPEN					
1	INVENTORY TRANSACTIONS		IS END OF FILE		N	U	N	Y
2	ITEM NO: SALES		EQ	ITEM NO:INVEN	Y	Y	N	
3	QTY SOLD		LE	QTY ON HAND	Y	N		
4	SET SALES QTY SHIP		EQ		QTY SOLD	QTY ON HAND		
5	SET QTY BACKORDER					EQ QTY SOLD – QTY ON HAND		
6	SET QTY ISSUE				+ QTY SOLD	+ QTY ON HAND		
7	MOVE UNIT PRICE		TO SALES AMT		X	X		
	* SALES QTY SHIP							
8	SET QTY ON HAND				– QTY SOLD	EQ ZERO		
9	GOTO				TRANS IN	TRANS IN	GETX	SHIPMENT RELEASE

Figure 3a. Decision Table, Example 3.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
INVENTORY, BILLING	FRANCIS	1/2/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY TRANSACTIONS	IN	SER-CARD	SALES	–	8000	10,000

SER NO	FIELD NAME	ALPHA/NUM	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC NO.	ASSOC SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	QTY SOLD	N	+	4				1						
3	SALES QTY SHIP	N	+	4				1						
4	QTY BACK ORDER	N	+	4				1						
5	SALES AMT	N	U	4.2				1						

Figure 3b. File Description, Example 3.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
INVENTORY, BILLING	FRANCIS	1/2/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY MASTER	RETAINED	RAN-DISK	INVENTORY	INVEN	4,000	5,000

SER NO	FIELD NAME	ALPHA/NUM	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC NO.	ASSOC SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	UNIT PRICE	N	+	3.2				1						
3	OLD BAL ON HAND	N	+	5				1						
4	QTY RECEIPTS	N	+	5				1						
5	QTY ISSUES	N	+	5				1						
6	QTY ON HAND	N	+	5				1						
7	QTY ON ORDER	N	+	5				1						
8	QTY MINIMUM	N	+	5				1						
9	ITEM DESCR	A		20	✓			1						

Figure 3c. File Description, Example 3.

QTY SOLD is less than or equal to the QTY ON HAND...

Each of these operands is a field of the records SALES and INVENTORY, respectively. However, they are unique within the system and need not be qualified. Since ITEM NO is a field name in both records and therefore lacks uniqueness, it has to be qualified by being linked to the name of the record where it is contained.

The decision table in figure 3 illustrates the use of the SET and MOVE operators in two operand arithmetic and in multi-operand arithmetic. Action row 6 for rules 1 and 2 is stated as follows:

rule 1: SET the Quantity of Issue equal to the Quantity of Issue plus the Quantity Sold...

rule 2: SET the Quantity of Issue equal to the Quantity of Issue plus the Quantity on Hand...

This is the SET operator with two operand arithmetic. Two operand arithmetic may be used for addition, subtraction, multiplication, and division. The results of these arithmetic operations for two fields, A and B, are summarized below:

OPERATOR	OPERAND 1	CONNECTOR	OPERAND 2	Result is
SET	A	+	B	A ← A + B
SET	A	-	B	A ← A - B
SET	A	*	B	A ← A * B
SET	A	/	B	A ← A / B

TABLE ID / SER	TABLE NAME / OPER	OPERAND 1	TABLE TYPE / OP	OPERAND 2	FREQUENCY: OP 01 OPERAND 2	OP 02 OPERAND 2	OP 01 OPERAND 2	OP 04 OPERAND 2
** 005	SALES EXTENSION		OPEN					
1	INVENTORY TRANSACTIONS		IS END OF FILE		N	W	N	Y
2	ITEM NO: SALES		EQ	ITEM NO:INVEN	Y	Y	N	
3	QTY SOLD		LE	QTY ON HAND	Y	N		
4	SET SALES QTY SHIP		EQ		QTY SOLD	QTY ON HAND		
5	SET QTY BACK ORDER					EQ QTY SOLD – QTY ON HAND		
6	SET QTY ISSUE				+ QTY SOLD	+ QTY ON HAND		
7	MOVE UNIT PRICE * SALES QTY SHIP		TO	SALES AMT	X	X		
8	SET QTY ON HAND				– QTY SOLD	EQ ZERO		
9	GOTO				TRANS IN	TRANS IN	GETX	SHIPMENT RELEASE

Figure 3a. Decision Table, Example 3.

IBM

SYSTEM NAME: INVENTORY, BILLING

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY TRANSACTIONS	IN	SER-CARD	SALES	–	8000	10,000

SER NO	FIELD NAME	NUMERIC / ALPHA	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	TO ASSOC	OCCURRENCES	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	QTY SOLD	N	+	4				1						
3	SALES QTY SHIP	N	+	4				1						
4	QTY BACK ORDER	N	+	4				1						
5	SALES AMT	N	V	4.2				1						

Figure 3b. File Description, Example 3.

IBM

SYSTEM NAME: INVENTORY, BILLING

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY MASTER	RETAINED	RAN-DISK	INVENTORY	INVEN	4,000	5,000

SER NO	FIELD NAME	NUMERIC / ALPHA	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	TO ASSOC	OCCURRENCES	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	UNIT PRICE	N	+	3.2				1						
3	OLD BAL ON HAND	N	+	5				1						
4	QTY RECEIPTS	N	+	5				1						
5	QTY ISSUES	N	+	5				1						
6	QTY ON HAND	N	+	5				1						
7	QTY ON ORDER	N	+	5				1						
8	QTY MINIMUM	N	+	5				1						
9	ITEM DESCR	A		20	✓			1						

Figure 3c. File Description, Example 3.

A is always the receiving field. B is added to, subtracted from, multiplied by, or divided into A before the result is placed in A. A similar display could be constructed for MOVE:

OPERATOR	OPERAND 1	CONNECTOR	OPERAND 2	Result is
MOVE	A	+	B	B + A → B
MOVE	A	-	B	B - A → B
MOVE	A	*	B	B * A → B
MOVE	A	/	B	B / A → B

B is always the receiving field. A is added to, subtracted from, multiplied by or divided into B before the result is placed in B. Two operand arithmetic may be used in either limited entry or extended entry form.

Action row 7 of rules 1 and 2 illustrates the MOVE operator in multi-operand arithmetic. That part of rule 1 reads:

...MOVE the UNIT PRICE times the SALES QuanTitY SHIP to SALES AMounT... The results of multi-operand arithmetic for three fields, A, B, and C, are summarized below.

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY																													
USER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2					
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 005	SALES EXTENSION	OPEN																														
1	INVENTORY TRANSACTIONS	IS END OF FILE	N								N								N								Y					
2	ITEM NO: SALES	EQ ITEM NO:INVEN	Y								Y								N													
3	QTY SOLD	LE QTY ON HAND	Y								N																					
4	SET SALES QTY SHIP	EQ			QTY SOLD								QTY ON HAND																			
5	SET QTY BACK ORDER										EQ		QTY SOLD																			
											-		QTY ON HAND																			
6	SET QTY ISSUE		+		QTY SOLD						+		QTY ON HAND																			
7	MOVE UNIT PRICE	TO SALES AMT	X								X																					
	* SALES QTY SHIP																															
8	SET QTY ON HAND		-		QTY SOLD						EQ		ZERO																			
9	GOTO		TRANS IN								TRANS IN								GETX								SHIPMENT RELEASE					

Figure 3a. Decision Table, Example 3.

IBM

SYSTEM NAME: INVENTORY, BILLING

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY TRANSACTIONS	IN	SER-CARD	SALES	—	8000	10,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	ASSOC SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	QTY SOLD	N	+	4				1						
3	SALES QTY SHIP	N	+	4				1						
4	QTY BACK ORDER	N	+	4				1						
5	SALES AMT	N	U	4.2				1						

Figure 3b. File Description, Example 3.

IBM

SYSTEM NAME: INVENTORY, BILLING

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY MASTER	RETAINED	RAN-DISK	INVENTORY	INVEN	4,000	5,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	ASSOC SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	UNIT PRICE	N	+	3.2				1						
3	OLD BAL ON HAND	N	+	5				1						
4	QTY RECEIPTS	N	+	5				1						
5	QTY ISSUES	N	+	5				1						
6	QTY ON HAND	N	+	5				1						
7	QTY ON ORDER	N	+	5				1						
8	QTY MINIMUM	N	+	5				1						
9	ITEM DESCR	A		20	✓			1						

Figure 3c. File Description, Example 3.

OPERATOR	OPERAND 1	CONNECTOR	OPERAND 2	Result is
MOVE	A	TO	C	A + B → C
	+ B			
MOVE	A	TO	C	A - B → C
	- B			
MOVE	A	TO	C	A * B → C
	* B			
MOVE	A	TO	C	A / B → C
	/ B			

OPERATOR	OPERAND 1	CONNECTOR	OPERAND 2	Result is
SET	A	EQ	B	A ← B + C
		+	C	
SET	A	EQ	B	A ← B - C
		-	C	
SET	A	EQ	B	A ← B * C
		*	C	
SET	A	EQ	B	A ← B / C
		/	C	

TABLE ID / SER	TABLE NAME / OPER	OPERAND 1	TABLE TYPE / OP	OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
** 005	SALES EXTENSION		OPEN					
1	INVENTORY TRANSACTIONS		IS END OF FILE		N	Y	N	Y
2	ITEM NO: SALES		EQ	ITEM NO: INVEN	Y	Y	N	
3	QTY SOLD		LE	QTY ON HAND	Y	N		
4	SET SALES QTY SHIP		EQ		QTY SOLD	QTY ON HAND		
5	SET QTY BACK ORDER					EQ QTY SOLD		
						− QTY ON HAND		
6	SET QTY ISSUE				+ QTY SOLD	+ QTY ON HAND		
7	MOVE UNIT PRICE		TO	SALES AMT	X	X		
	* SALES QTY SHIP							
8	SET QTY ON HAND				− QTY SOLD	EQ ZERO		
9	GOTO				TRANS IN	TRANS IN	GETX	SHIPMENT RELEASE

Figure 3a. Decision Table, Example 3.

SYSTEM NAME: INVENTORY, BILLING

IBM

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY TRANSACTIONS	IN	SER-CARD	SALES	—	8000	10,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	ASSOC INCR	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	QTY SOLD	N	+	4				1						
3	SALES QTY SHIP	N	+	4				1						
4	QTY BACK ORDER	N	+	4				1						
5	SALES AMT	N	U	4.2				1						

Figure 3b. File Description, Example 3.

SYSTEM NAME: INVENTORY, BILLING

IBM

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY MASTER	RETAINED	RAN-DISK	INVENTORY	INVEN	4,000	5,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	ASSOC INCR	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	UNIT PRICE	N	+	3.2				1						
3	OLD BAL ON HAND	N	+	5				1						
4	QTY RECEIPTS	N	+	5				1						
5	QTY ISSUES	N	+	5				1						
6	QTY ON HAND	N	+	5				1						
7	QTY ON ORDER	N	+	5				1						
8	QTY MINIMUM	N	+	5				1						
9	ITEM DESCR	A		20	✓			1						

Figure 3c. File Description, Example 3.

More than three operands can be used in multi-operand arithmetic.

MOVE	A	TO	D	means	A * B * C → D
	*B				
	*C				
SET	A	EQ	B		A ← B + C + D + E
		+	C		
		+	D		
		+	E		

Two division connectives cannot be used in a single action, although addition, subtraction and multiplication connectives can be used more than once.

Action row 8 in rule 2 illustrates the use of a connotative term. This is an operand whose value is directly implied by the meaning of its symbols. This action of rule 2 is read:

... SET the QUANTITY ON HAND to ZERO...

Here, ZERO is a connotative term which represents a string of zeros. The number of zeros is determined by the size of the field which is to be set to zero. In this case QTY ON HAND is a five position field, so five zeros are moved to QTY ON HAND. Other connotative terms are: BLANK, NINE, ALL 'X'. In the last term, X represents any symbol or combination of symbols. For example, ALL '7' would produce 7777...., ALL 'ABC' would produce ABCABCABC...

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY																													
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2							
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 005	SALES EXTENSION	OPEN																														
1	INVENTORY TRANSACTIONS	13 END OF FILE	N								N								N								Y					
2	ITEM NO: SALES	EQ ITEM NO: INVEN	Y								Y								N													
3	QTY SOLD	LE QTY ON HAND	Y								N																					
4	SET SALES QTY SHIP	EQ			QTY SOLD								QTY ON HAND																			
5	SET QTY BACK ORDER										EQ		QTY SOLD																			
											-		QTY ON HAND																			
6	SET QTY ISSUE		+		QTY SOLD						+		QTY ON HAND																			
7	MOVE UNIT PRICE	TO SALES AMT	X								X																					
	* SALES QTY SHIP																															
8	SET QTY ON HAND		-		QTY SOLD						EQ		ZERO																			
9	GOTO		TRANS IN								TRANS IN								GETX								SHIPMENT RELEASE						

Figure 3a. Decision Table, Example 3.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
INVENTORY, BILLING	FRANCIS	1/2/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY TRANSACTIONS	IN	SER-CARD	SALES	—	8000	10,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	ASSOC SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	QTY SOLD	N	+	4				1						
3	SALES QTY SHIP	N	+	4				1						
4	QTY BACK ORDER	N	+	4				1						
5	SALES AMT	N	U	4.2				1						

Figure 3b. File Description, Example 3.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
INVENTORY, BILLING	FRANCIS	1/2/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY MASTER	RETAINED	RAN-DISK	INVENTORY	INVEN	4,000	5,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	ASSOC SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	UNIT PRICE	N	+	3,2				1						
3	OLD BAL ON HAND	N	+	5				1						
4	QTY RECEIPTS	N	+	5				1						
5	QTY ISSUES	N	+	5				1						
6	QTY ON HAND	N	+	5				1						
7	QTY ON ORDER	N	+	5				1						
8	QTY MINIMUM	N	+	5				1						
9	ITEM DESCR	A		20	✓			1						

Figure 3c. File Description, Example 3.

QUESTIONS

1. Read the 3rd and 4th rules.
2. What are the relationships among files, records, and fields?
3. How may a field be uniquely qualified?
4. What is the meaning of the following?

   a)
   ```
   MOVE  A   TO  D
         + B
         + C
   ```

   b)
   ```
   SET   A   EQ  B
             +   C
   ```

5. What is the meaning of the statement:

   ```
   SET   A   EQ  NINES
   ```

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY																													
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2							
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 005	SALES EXTENSION	OPEN																														
1	INVENTORY TRANSACTIONS	IS END OF FILE	N								Y								N								Y						
2	ITEM NO: SALES	EQ ITEM NO:INVEN	Y								Y								N													
3	QTY SOLD	LE QTY ON HAND	Y								N																					
4	SET SALES QTY SHIP	EQ		QTY SOLD								QTY ON HAND																				
5	SET QTY BACK ORDER										EQ	QTY SOLD																				
											-	QTY ON HAND																				
6	SET QTY ISSUE		+	QTY SOLD							+	QTY ON HAND																				
7	MOVE UNIT PRICE	TO SALES AMT	X								X																					
	* SALES QTY SHIP																															
8	SET QTY ON HAND		-	QTY SOLD							EQ	ZERO																				
9	GOTO		TRANS IN								TRANS IN								GETX								SHIPMENT RELEASE						

Figure 3a. Decision Table, Example 3.

IBM

SYSTEM NAME: INVENTORY, BILLING

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY TRANSACTIONS	IN	SER-CARD	SALES	—	8000	10,000

SER NO	FIELD NAME	RECORD NUMBER	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	ASSOC SEGMENT	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	QTY SOLD	N	+	4				1						
3	SALES QTY SHIP	N	+	4				1						
4	QTY BACK ORDER	N	+	4				1						
5	SALES AMT	N	U	4.2				1						

Figure 3b. File Description, Example 3.

IBM

SYSTEM NAME: INVENTORY, BILLING

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 1/2/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVENTORY MASTER	RETAINED	RAN-DISK	INVENTORY	INVEN	4,000	5,000

SER NO	FIELD NAME	RECORD NUMBER	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	ASSOC SEGMENT	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	7				1						
2	UNIT PRICE	N	+	3.2				1						
3	OLD BAL ON HAND	N	+	5				1						
4	QTY RECEIPTS	N	+	5				1						
5	QTY ISSUES	N	+	5				1						
6	QTY ON HAND	N	+	5				1						
7	QTY ON ORDER	N	+	5				1						
8	QTY MINIMUM	N	+	5				1						
9	ITEM DESCR	A		20	✓			1						

Figure 3c. File Description, Example 3.

ANSWERS

1. Rule 3: If it is not the end of the transaction file, and the item number of the sales record is not the same as the item number of the inventory record, then go to table GETX.

   Rule 4: If there are no more records in the transaction file, then go to the table named SHIPMENT RELEASE.

2. A file is a collection of one or more records.

   A record is a collection of one or more fields.

3. By specifying the field name, placing a colon after it, and then specifying the particular record name.

4. a) This is multi-operand arithmetic with the MOVE operator. The final result is that

   D will contain A + B + C

   b) This is multi-operand arithmetic with the SET operator. The final result is that

   A will be replaced by B + C

5. The field A is to be replaced with the value 9999.....; the number of nines will be determined by the size of A.

TABLE NO. / USER	TABLE NAME: OPER	OPERAND 1	TABLE TYPE: OP	OPERAND 2	FREQUENCY: OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
					01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32
** 037	DEDUCTION VALID		OPEN		60 35 E			
1	TYPE DED CODE		EQ DED CODE LIST		Y Y			
2	TYPE DED CODE		EQ 13 22-24		N Y			
3	BAL AMT		GE DED AMT		Y			
4	GO TO		PAYROLL CALC		X X			
5	GO TO		ERROR		X			

Figure 4a. Decision Table, Example 4.

SYSTEM NAME	IBM	ANALYST	DATE	
PAYROLL	REFERENCE OPERAND DESCRIPTION	HOLT	1/3/62	PAGE OF PAGES

REF NO	NAME	TYPE	VALUE
1	DED CODE LIST	V LIST	3, 7, 13, 18, 22-26

Figure 4b. Reference Operand Description, Example 4.

Example 4 covers the part of a payroll system which checks to see if a certain type of deduction is valid.

The decision table figure 4a is written entirely in limited entry form. In preparing decision tables for programmers the systems engineer should make an estimate of the frequency that a rule will be executed. For this example, the analyst has estimated the conditions for rule 1 will be satisfied 60% of the time and 35% of the time for rule 2. This information is recorded at the top of the column of entries for each rule in either the extended entry or the limited entry table.

In the first 3 examples of this chapter, it is assumed that the several rules exhaust all possible combinations of conditions which can ever occur in these tables. Therefore, failure to satisfy at least one set of conditions indicates that there is an error in logic or in data. However, occasions will arise when the table is not intended to explicitly describe all possible combinations of conditions, or where it is impractical or impossible to do so. In this event, some method must be provided to recognize unaccounted for conditions without inferring that an error in logic or data exists. This is covered by the special unconditional rule ELSE (E), which provides for action control when none of the other rules can be executed. In example 4, the ELSE rule will occur about 5% of the time (100 - 60 - 35 = 5). This rule will execute action row 5 which provides for an error table to be entered.

TABLE NO. / SER	TABLE NAME / OPER OPERAND 1	TABLE TYPE / OP OPERAND 2	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 037	DEDUCTION VALID	OPEN	60	35	E																													
1	TYPE DED CODE	EQ DED CODE LIST	Y	Y																														
2	TYPE DED CODE	EQ 13	N	Y																														
		22-24																																
3	BAL AMT	GE DED AMT		Y																														
4	GO TO	PAYROLL CALC	X	X																														
5	GO TO	ERROR			X																													

Figure 4a. Decision Table, Example 4.

IBM

REFERENCE OPERAND DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	HOLT	1/3/62	

SER NO	NAME	TYPE	VALUE
1	DED CODE LIST	V LIST	3, 7, 13, 18, 22-26

Figure 4b. Reference Operand Description, Example 4.

On the limited entry table, the ELSE rule is indicated by the placement of an E in the column where the frequency of the rule is usually written; in a mixed or extended entry table, the word ELSE is written in the appropriate place for the rule to which it is assigned. The ELSE rule is always considered last when "examining" a table. There can, of course, be no condition entries in an ELSE rule.

Rule 1 of this example is read as follows:

If the type of deduction code is on the deduction code list,

and

the type of deduction code is not 13, 22, 23, or 24,

then proceed to payroll calculation table.

With this rule, two kinds of value list operands are displayed simultaneously. The first DED CODE LIST, is a value list name which represents a specific collection of one or more values or ranges of values. It is not a field, nor does it appear on a record or file. It may only be used in a conditional statement and must be defined on a Reference Operand Description form (figure 4b) before being used.

The Reference Operand Description form is used for the description of constants, expressions, reference lists, and value lists. In figure 4b, the system name, which is the same as the system name for the decision table, is placed in the upper left-hand corner. Each listing is then assigned an individual number. For this example, there is only one

TABLE NAME		TABLE TYPE		FREQUENCY																											
USER	OPER	OPERAND 1	OP	OPERAND 2	OP	OPERAND 2								OP	OPERAND 2								OP	OPERAND 2							
					01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 037	DEDUCTION VALID		OPEN		6	0	3	5	E																									
1	TYPE DED CODE		EQ DED CODE LIST		Y		Y																											
2	TYPE DED CODE		EQ 13 22-24		N		Y																											
3	BAL AMT		GE DED AMT				Y																											
4	GO TO		PAYROLL CALC		X		X																											
5	GO TO		ERROR					X																										

Figure 4a. Decision Table, Example 4.

SYSTEM NAME	IBM	ANALYST	DATE	
PAYROLL	REFERENCE OPERAND DESCRIPTION	HOLT	1/3/62	PAGE OF PAGES

REF NO	NAME	TYPE	VALUE
1	DED CODE LIST	V LIST	3, 7, 13, 18, 22-26

Figure 4b. Reference Operand Description, Example 4.

listing, the DED CODE LIST. The type entry is V LIST, and the values are as shown. Each value is either a single entry value (e.g., 3, 7, etc.) or a range of values for which the top and bottom limits are stated (e.g., 22-26). All individual values or ranges of values are separated by a comma which denotes an exclusive or associated with the value list. In other words, the value is 3, or 7, or... or 22-26. Consequently, to satisfy the first condition of rule 1, the code number must be one of the values, or within the range of values given in the list.

The second kind of value list operand is demonstrated in the second condition of rule 1:

```
TYPE DED CODE      EQ      13
                           22-24
```

The operand value list (13, 22-24) reveals that a list of literal values can be used directly as the operand in a condition statement without specifying the list in a Reference Operand Description form. A range (22-24) is permitted, just as with the value list name.

If TYPE DED CODE is 13 in example 4, the condition of row 2, rule 1 is not satisfied (N in the entry) and rule 2 is satisfied (a Y in the entry).

TABLE NUMBER	TABLE NAME	TABLE TYPE	FREQUENCY
	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2 / OP 02 OPERAND 2 / OP 03 OPERAND 2 / OP 04 OPERAND 2
** 037	DEDUCTION VALID	OPEN	E
1	TYPE DED CODE	EQ DED CODE LIST	Y Y
2	TYPE DED CODE	EQ 13 22-24	N Y
3	BAL AMT	GE DED AMT	Y
4	GO TO	PAYROLL CALC	X X
5	GO TO	ERROR	X

Figure 4a. Decision Table, Example 4.

SYSTEM NAME	IBM	ANALYST	DATE	
PAYROLL	REFERENCE OPERAND DESCRIPTION	HOLT	1/3/62	PAGE OF PAGES

REF NO	NAME	TYPE	VALUE
1	DED CODE LIST	V LIST	3, 7, 13, 18, 22-26

Figure 4b. Reference Operand Description, Example 4.

QUESTIONS

1. How are a list of values indicated in TDL?
2. What do the rules of this table indicate as far as frequency of execution is concerned?
3. What is the effect of the ELSE rule?
4. If the logic required that a code type not be in the range 72 - 103, how could this be handled?
5. If a value list has a range of 0 - 53, how may a code type be investigated to ascertain that it is not in the range of 12 - 26, but that it is on the list?

TABLE NAME		TABLE TYPE		FREQUENCY																											
ENTR	OPER	OPERAND 1	OP	OPERAND 2	OP 01		OPERAND 2						OP 02		OPERAND 2								OP 03		OPERAND 2							
					01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 037	DEDUCTION VALID		OPEN																															
1	TYPE DED CODE		EQ DED CODE LIST		Y		Y																											
2	TYPE DED CODE		EQ 13 22-24		N		Y																											
3	BAL AMT		GE DED AMT				Y																											
4	GO TO		PAYROLL CALC		X		X																											
5	GO TO		ERROR				X																											

Figure 4a. Decision Table, Example 4.

SYSTEM NAME	IBM	ANALYST	DATE	
PAYROLL	REFERENCE OPERAND DESCRIPTION	HOLT	1/3/62	PAGE OF PAGES

REF NO	NAME	TYPE	VALUE
1	DED CODE LIST	V LIST	3, 7, 13, 18, 22-26

Figure 4b. Reference Operand Description, Example 4.

ANSWERS

1. By giving the list of values a name, and describing them as a list on a Reference Operand Description form, or by introducing a list of literal values directly in the table.

2. The analyst who prepared this decisive table has estimated that rule 1 will be executed almost twice as often as rule 2, but the two rules still do not cover all the possibilities that may occur. In 5% of the cases, rule 3 will be executed.

3. The conditions for this rule are automatically satisfied and the actions executed. The ELSE rule is considered only when the conditions of none of the other rules are satisfied.

4. A range literal might be used, as there is no obvious requirement that the code types need be part of a defined list. Therefore, a statement as:

   TYPE CODE NE 72-103

   satisfies the requirement.

5. The reference operand form list has the value list described on it. Therefore, the conditions may be stated in two steps:

   1) Is the type code on the value list?

   2) Is the type code in the range 0 - 11?

   27 - 53?

   OR

   1) Is the type code on the value list?

   2) Is the type code not in the range 12 - 26?

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY																														
USER	OPER	OPERAND 1	OP	OPERAND 2	OP 01			OPERAND 2					OP 02								OPERAND 2							OP 03				OPERAND 2							OP 04				OPERAND 2						
					01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 009	EXTEND DETAIL		OPEN		005	010	020																											
1	CUST NO: CUST HEDNG		EQ CUST NO: WORK A		Y	Y	N																											
2	DISCOUNT (ITEM NO: CUST TRAN, QUAN SOLD: CUST TRAN)		IS PRESENT		Y	N	·																											
3	MOVE CUST TRAN		TO CUST INV		X	X	·																											
4	SET AMT EXT: CUST INV		EQ UNIT PRICE: CT		X	X	·																											
			* QUAN SOLD: CT																															
			* CUST DISC: WORK A																															
5	SET AMT EXT: CUST INV		* DISCOUNT (ITEM NO: CT QUAN SOLD: CT)		X	·	·																											
6	SET WORK TOTAL		+ AMT EXT: CUST INV		X	X	·																											
7	WRITE		CUST INV		X	X	·																											
8	MOVE WORK TOTAL		TO AMT BILL TOT: TOTAL BILLING		·	·	X																											
9	WRITE		TOTAL BILLING		·	·	X																											
10	MOVE 0		TO WORK TOTAL		·	·	X																											
11	MOVE CUST HEDNG		TO WORK A		·	·	X																											
12	GO TO		GET REC		X	X	X																											

Figure 5a. Decision Table, Example 5.

Example 5 is part of the procedure Invoice Preparation in the system called Customer Billing, and is concerned with the calculation of an amount for extended sales, the calculation of a discount value for the extended sales amount, and the writing of a customer invoice.

Condition row 1 in rule 1, reads:

... Is the customer number on the record named customer heading the same as the customer number on work record A... ?

WORK A is an illustration of a work operand, which is used for the temporary holding of either records or fields. The names of these operands must always be preceded by the word WORK.

A work operand always takes on the characteristics of the field or record which is placed in it, except for the name. It is assumed that there is sufficient space to contain all work records and work fields. The work operand will retain the value last placed in it, until something else is placed in it.

In this example, a record has been previously placed in the work record. A field in this record (not shown) is now being compared for correspondence with a field value of another record abbreviated CUST HEDNG (also not shown). Action row 11 of rule 3, demonstrates the placing of this record in the work record in preparation for the next pass through this table. This portion of rule 3 reads as follows:

TABLE NO. USER	TABLE NAME OPER	OPERAND 1	TABLE TYPE OP	OPERAND 2	FREQUENCY OP	OPERAND 2	OP	OPERAND 2	OP	OPERAND 2
					01 02 03	04 05 06 07 08	09 10 11 12 13 14 15 16		17 18 19 20 21 22 23 24		25 26 27 28 29 30 31 32	
** 000	EXTEND	DETAIL	OPEN		30 50 20							
1	CUST NO: CUST HEDNG		EQ CUST NO: WORK A		Y Y N							
2	DISCOUNT (ITEM NO: CUST TRAN, QUAN SOLD: CUST TRAN)		IS PRESENT		Y N •							
3	MOVE	CUST TRAN	TO CUST INV		X X •							
4	SET AMT EXT: CUST INV		EQ UNIT PRICE: CT * QUAN SOLD: CT * CUST DISC.: WORK A		X X •							
5	SET AMT EXT: CUST INV		* DISCOUNT (ITEM NO: CT QUAN SOLD: CT)		X • •							
6	SET WORK TOTAL		+ AMT EXT: CUST INV		X X •							
7	WRITE		CUST INV		X X •							
8	MOVE	WORK TOTAL	TO AMT BILL TOT: TOTAL BILLING		• • X							
9	WRITE		TOTAL BILLING		• • X							
10	MOVE	0	TO WORK TOTAL		• • X							
11	MOVE	CUST HEDNG	TO WORK A		• • X							
12	GO TO		GET REC		X X X							

Figure 5a. Decision Table, Example 5.

...move the record named customer heading to the work record named A... When the contents of a record are moved into a work record, the work record contains all the fields, and also the same field names as they appeared in the original record. The record then exists in identical form in two different places. To refer to these fields, the user must qualify the field name with the record name, WORK A, e. g., QTY ON HAND: WORK A.

A work operand may be either a work record or a work field.

Action row 6 of rule 1 is read as follows:

...the work field, WORK TOTAL, is incremented by the value of the field, AMT EXT, from the record named CUST INV...

A value has previously been placed in the work field. It is now replaced by this value plus some other value. The remainder of the work record does not change as a result of this action.

The movement of one record to another is different. When records are assigned to other records, only those fields with corresponding names are assigned. To illustrate this, consider two records A and B. They are defined:

Record A, contains fields named Q, R, S, T, U, V, and W.

Record B, contains fields named Q, R, X, Y, Z, S, and T.

With the operator MOVE in the statement

MOVE A TO B,

the values of corresponding fields, Q, R, S, and T of record A, are moved

TABLE NO. / SEQ	TABLE NAME / OPER OPERAND 1	TABLE TYPE / OP OPERAND 2	01	02	03
* * 009	EXTEND DETAIL	OPEN	50	50	20
1	CUST NO: CUST HEDNG	EQ CUST NO: WORK A	Y	Y	N
2	DISCOUNT (ITEM NO: CUST TRAN, QUAN SOLD: CUST TRAN)	IS PRESENT	Y	N	•
3	MOVE CUST TRAN	TO CUST INV	X	X	•
4	SET AMT EXT: CUST INV	EQ UNIT PRICE: CT * QUAN SOLD: CT * CUST DISC: WORK A	X	X	•
5	SET AMT EXT: CUST INV	* DISCOUNT (ITEM NO: CT QUAN SOLD: CT)	X	•	•
6	SET WORK TOTAL	+ AMT EXT: CUST INV	X	X	•
7	WRITE	CUST INV	X	X	•
8	MOVE WORK TOTAL	TO AMT BILL TOT: TOTAL BILLING	•	•	X
9	WRITE	TOTAL BILLING	•	•	X
10	MOVE 0	TO WORK TOTAL	•	•	X
11	MOVE CUST HEDNG	TO WORK A	•	•	X
12	GO TO	GET REC	X	X	X

Figure 5a. Decision Table, Example 5.

to record B. Fields X, Y, and Z of record B are unchanged. The original record A is also unchanged. A similar operation can be performed with the SET operator. Note that the corresponding fields of A and B are moved regardless of their position on the record. This illustrates the fact that a file description is made up without regard to the positions of the fields. Action row 3, 8, and 11 illustrate assignment of records:

Condition row 2 (on figure 5a) of rule 1 reads:

... Is the discount (defined as a function of the item number and quantity of that item sold according to a record named CUST TRAN) present... ?

This condition row introduces an operand which has not been previously described, a reference list function. A reference list serves a purpose similar to that of a matrix or table of values. Each reference list function (like discount) is a function of either one or two arguments (two in our example: item number and quantity sold). These arguments may be considered as the vertical and horizontal axes of a tabular array, as shown below:

	ITEM NOS.	QUANTITY		
		Col. 1	Col. 2	Col. 3
		1-30	31-100	101-1000
Row 1	00000 -22040	+.98	+.95	+.88
Row 2	50000 -79999	+.97	+.94	+.87

TABLE ID / SER	TABLE NAME / OPER OPERAND 1	TABLE TYPE / OP OPERAND 2	01	02	03
** 000	EXTEND DETAIL	OPEN	30	50	20
1	CUST NO: CUST HEDNG	EQ CUST NO: WORK A	Y	Y	N
2	DISCOUNT (ITEM NO: CUST TRAN, QUAN SOLD: CUST TRAN)	IS PRESENT	Y	N	·
3	MOVE CUST TRAN	TO CUST INV	X	X	·
4	SET AMT EXT: CUST INV	EQ UNIT PRICE: CT * QUAN SOLD: CT * CUST DISC: WORK A	X	X	·
5	SET AMT EXT: CUST INV	* DISCOUNT (ITEM NO: CT QUAN SOLD: CT)	X	·	·
6	SET WORK TOTAL	+ AMT EXT: CUST INV	X	X	·
7	WRITE	CUST INV	X	X	·
8	MOVE WORK TOTAL	TO AMT BILL TOT: TOTAL BILLING	·	·	X
9	WRITE	TOTAL BILLING	·	·	X
10	MOVE 0	TO WORK TOTAL	·	·	X
11	MOVE CUST HEDNG	TO WORK A	·	·	X
12	GO TO	GET REC	X	X	X

Figure 5a. Decision Table, Example 5.

IBM

REFERENCE OPERAND DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
CUSTOMER BILLING	FRANCIS	12/29/61	

SER NO	NAME	TYPE	VALUE		
1	QUANTITY DISCOUNTS	REF LIST	ITEM RANGE	QUAN RANGE	DISCOUNT
	DISCOUNT (ITEM RANGE, QUAN RANGE)	FUNC NAME (2,3)	00000 - 22040	1 - 30	+ .98
			00000 - 22040	31 - 100	+ .95
			00000 - 22040	101 - 1000	+ .88
			50000 - 79999	1 - 30	+ .97
			50000 - 79999	31 - 100	+ .94
			50000 - 79999	101 - 1000	+ .87

Figure 5b. Reference Operand Description, Example 5.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
CUSTOMER BILLING	FRANCIS	12/28/61	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
CUST DATA	IN	SER-CARD	CUST TRAN	CT	1,000	3,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENGTH MAX	LENGTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	CUST NO	N	U	6				1			2			
2	TYPE REC CO	N	U	2				1			1			
3	ITEM NO	N	U	5				1						
4	QUAN SOLD	N	+	4				1						
5	UNIT PRICE	N	+	2.2				1						

Figure 5c. File Description, Example 5.

This tabular array can be described as a reference list and written on a Reference Operand Description form, shown partially on figure 5b. Here, the two axes of the original table are expressed in list form. For example, serial 6 of figure 5b expresses the information contained in row 2, column 2 of the matrix above. The quantity range is 31 thru 100, the range of item numbers 50000 thru 79999, and the function of these two arguments, discount quantity, is +.94.

When a range of values is used as an argument in a reference list, both the top and bottom limits of the range should be given. Arguments may be single entry constant values, ranges of values, or the names of value lists. If the name of a value list is given, the values represented by that name should be separately defined on a Reference Operand Description form as noted in example 4.

Reference lists cannot be modified by the actions of a decision table.

Using condition row 2 as the illustration, specific entries within a list are referred to in this form:

Reference list function name (first argument (variable , second argument variable)

DISCOUNT (ITEM NO: CUST TRAN, QUAN SOLD: CUST TRAN)

Here variables 1 and 2 are field names on the input record CUST TRAN. They are not unique names and consequently are qualified in the normal manner. If they were unique names, the reference of condition row 2 would be:

DISCOUNT (ITEM NO, QUAN SOLD).

TABLE NAME	TABLE TYPE	FREQUENCY		
SER / OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2 / OP 04 OPERAND 2
009 EXTEND DETAIL	OPEN	30 50 20		
1 CUST NO: CUST HEDNG	EQ CUST NO: WORK A	Y Y N		
2 DISCOUNT (ITEM NO: CUST TRAN, QUAN SOLD: CUST TRAN)	IS PRESENT	Y N ·		
3 MOVE CUST TRAN	TO CUST INV	X X ·		
4 SET AMT EXT: CUST INV	EQ UNIT PRICE: CT * QUAN SOLD: CT * CUST DISC: WORK A	X X ·		
5 SET AMT EXT: CUST INV	* DISCOUNT (ITEM NO: CT QUAN SOLD: CT)	X · ·		
6 SET WORK TOTAL	+ AMT EXT: CUST INV	X X ·		
7 WRITE	CUST INV	X X ·		
8 MOVE WORK TOTAL	TO AMT BILL TOT: TOTAL BILLING	· · X		
9 WRITE	TOTAL BILLING	· · X		
10 MOVE 0	TO WORK TOTAL	· · X		
11 MOVE CUST HEDNG	TO WORK A	· · X		
12 GO TO	GET REC	X X X		

Figure 5a. Decision Table, Example 5.

SYSTEM NAME: CUSTOMER BILLING

IBM

REFERENCE OPERAND DESCRIPTION

ANALYST: FRANCIS | DATE: 12/29/61 | PAGE OF PAGES

SER NO	NAME	TYPE	VALUE		
1	QUANTITY DISCOUNTS	REF LIST	ITEM RANGE	QUAN RANGE	DISCOUNT
	DISCOUNT (ITEM RANGE QUAN RANGE)	FUNC NAME (2,3)	00000 - 22040	1 - 30	+.98
			00000 - 22040	31 - 100	+.95
			00000 - 22040	101 - 1000	+.88
			50000 - 79999	1 - 30	+.97
			50000 - 79999	31 - 100	+.94
			50000 - 79999	101 - 1000	+.87

Figure 5b. Reference Operand Description, Example 5.

SYSTEM NAME: CUSTOMER BILLING

IBM

FILE DESCRIPTION

ANALYST: FRANCIS | DATE: 12/28/61 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABS	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
CUST DATA	IN	SER-CARD	CUST TRAN	CT	1,000	3,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ADDR TO	OCCURRENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	CUST NO	N	U	6				1			2			
2	TYPE REC CD	N	U	2				1			1			
3	ITEM NO	N	U	5				1						
4	QUAN SOLD	N	+	4				1						
5	UNIT PRICE	N	+	2.2				1						

Figure 5c. File Description, Example 5.

To record information externally, an input/output operator WRITE is used, as noted in action row 9. The operand (e.g., TOTAL BILLING) always appears in the second operand position of the action. For limited entry tables like Figure 5a, this would be in the action stub; for extended entry, it is located in the entry area of the rule.

The remainder of the decision table shown in Figure 5a is self-explanatory, except for a new notation in the condition and action entries. This is the placement of a period (.) in the entry for a rule when that condition or action entry is not pertinent. This is a convenience which signifies that the analyst has considered the action or condition of that row, and has decided that it does not apply to the rule. A period has exactly the same meaning as the blank entry.

QUESTIONS

1. What is a work operand? What are its characteristics?
2. What happens when one record is moved to another record?
3. How is a specific entry of a reference list referred to in a decision table?
4. What is the operation specified in action row 4 of the decision table shown in Figure 5a?
5. What is the operation specified in action row 10 of the decision table shown in Figure 5a?

TABLE ID / SER	TABLE NAME: OPER / OPERAND 1	TABLE TYPE: OP / OPERAND 2	01	02	03
* 009	EXTEND DETAIL	OPEN	30	50	20
1	CUST NO: CUST HEDNG	EQ CUST NO: WORK A	Y	Y	N
2	DISCOUNT (ITEM NO: CUST TRAN, QUAN SOLD: CUST TRAN)	IS PRESENT	Y	N	·
3	MOVE CUST TRAN	TO CUST INV	X	X	·
4	SET AMT EXT: CUST INV	EQ UNIT PRICE: CT * QUAN SOLD: CT * CUST DISC: WORK A	X	X	·
5	SET AMT EXT: CUST INV	* DISCOUNT (ITEM NO: CT QUAN SOLD: CT)	X	·	·
6	SET WORK TOTAL	+ AMT EXT: CUST INV	X	X	·
7	WRITE	CUST INV	X	X	·
8	MOVE WORK TOTAL	TO AMT BILL TOT: TOTAL BILLING	·	·	X
9	WRITE	TOTAL BILLING	·	·	X
10	MOVE 0	TO WORK TOTAL	·	·	X
11	MOVE CUST HEDNG	TO WORK A	·	·	X
12	GO TO	GET REC	X	X	X

Figure 5a. Decision Table, Example 5.

IBM
REFERENCE OPERAND DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
CUSTOMER BILLING	FRANCIS	12/28/61	

SER NO	NAME	TYPE	VALUE		
1	QUANTITY DISCOUNTS	REF LIST	ITEM RANGE	QUAN RANGE	DISCOUNT
	DISCOUNT (ITEM RANGE, QUAN RANGE)	FUNC NAME (2,3)	00000 - 22040	1 - 30	+.98
			00000 - 22040	31 - 100	+.95
			00000 - 22040	101 - 1000	+.88
			50000 - 79999	1 - 30	+.97
			50000 - 79999	31 - 100	+.94
			50000 - 79999	101 - 1000	+.87

Figure 5b. Reference Operand Description, Example 5.

IBM
FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
CUSTOMER BILLING	FRANCIS	12/28/61	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
CUST DATA	IN	SER-CARD	CUST TRAN	CT	1,000	3,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENGTH MAX	LENGTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	CUST NO	N	U	6				1			2			
2	TYPE REC. CD	N	U	2				1			1			
3	ITEM NO	N	U	5				1						
4	QUAN SOLD	N	+	4				1						
5	UNIT PRICE	N	+	2.2				1						

Figure 5c. File Description, Example 5.

ANSWERS

1. A work operand is an operand used for the temporary storage of either records or fields. It has the characteristics of the last item assigned to it.

2. An assignment operator used with records specifies the movement of corresponding fields of the sending record to the receiving record. The only exception to this operation is the movement of a record to a work record. In this case, the entire record is moved to the work record. The work record is then a carbon copy of the sending record.

3. A specific entry of a reference list is referred by writing the function name, and enclosing in parentheses the variables which correspond to arguments. If necessary, the variables are qualified in the usual manner.

4. This is an example of multi-operand arithmetic using the SET operator with arithmetic connectors. The operation performed is:

   The unit price, from the record CT, is multiplied by the value of the quantity sold from the same record, which in turn is multiplied by the value of the customer discount. The customer discount was derived previously from the reference list and moved to the WORK A record.

5. This is an example of moving a literal to a work field. Since the work field takes on the value of the field which is last placed there, WORK TOTAL now contains the single digit zero.

TABLE NUMBER	TABLE NAME	TABLE TYPE	FREQUENCY
	OPER	OPERAND 1 / OP	OPERAND 2 / OP 01 OPERAND 2 / OP 02 OPERAND 2 / OP 03 OPERAND 2 / OP 04 OPERAND 2
** 011	DED ANALYSIS	OPEN	99 / 13 / 35 / 10
1	CODE: BS	EQ	'A' / 'B' / 'S' / 'A'
2	SET EMP IDENTIF	EQ EMP	X / X / X
3	SET BOND BAL: PURCHASE	EQ BOND BAL: BS + BOND DED	X / X / X
4	SET STK BAL: PURCHASE	EQ STK BAL: BS + STK DED	X / X
5	DO		012 / 013 / 012
6	DO		013
7	GO TO		025 / 016 / 016 / 016
** 012	BOND PURCHASE TEST	CLOSED	35 / ELSE
1	BOND BAL: PURCHASE		GE BOND PURCHASE PRICE
2	SET BOND PRICE	EQ	BOND PURCHASE PRICE
3	SET CODE: PURCHASE	EQ	'B'
4	SET BOND BAL: PURCHASE	-	BOND PURCHASE PRICE
5	DO	PREPARE P.O.	X
6	DO	012	X
** 013	STK PURCHASE TEST	CLOSED	BOE
1	STK BAL: PURCHASE	GE MKT VALUE	Y
2	SET STK PRICE	EQ MKT VALUE	X
3	SET STK BAL: PURCHASE	- MKT VALUE	X
4	SET CODE: PURCHASE	EQ 'S'	X
5	DO	PREPARE P.O.	X
6	DO	013	X

REMARKS: EMP SE JOIN(LOC NO: BS, DEPT: BS, EMP NO: BS, EMP NAME: BS)
MKT VALUE AE (DAYS HIGH PRICE + DAYS LOW PRICE)/2

Figure 6a. Decision Tables, Example 6.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
DEDUCTION	IN	SER-CARD	BOND STOCK DED	BS	5000	7000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSUM	ASSUM TO	REDEFINES	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP NAME	A		20	✓			1						
2	EMP NO	N	U	6				1						
3	BOND BAL	N	+	3,2				1						
4	BOND DED	N	+	3,2				1						
5	BOND PURCHASE PRICE	N	+	3,2				1						
6	STK BAL	N	+	4,2				1						
7	STK DED	N	+	3,2				1						
8	CODE	A		1				1						
9	DEPT	N	U	3				1						
10	LOC NO	N	U	3				1						

Figure 6b. File Description 1, Example 6.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
CUR. DED BAL	OUT	SER-CARD	PURCHASE		1000	1500

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSUM	ASSUM TO	REDEFINES	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP IDENTIF	A		3,2				1						
2	BOND BAL	N	+	3,2		0	.5	1						
3	BOND PRICE	N	+	3,2		0	5	1						
4	STK BAL	N	+	4,2		0	5	1						
5	STK PRICE	N	+	3,2		0	.5	1						
6	CODE	A		1				1						

Figure 6c. File Description 2, Example 6.

Example 6 treats a stock and bond purchase procedure, which should be easy to follow upon careful examination of the three tables.

The situation described in tables 011, 012, and 013 of figure 6a, is a deduction computation for stocks and bonds. It is assumed that prior to the execution of these tables, a record has been read that contains a code to indicate employee participation as follows:

N	Neither bond nor stock
B	Bond
S	Stock
A	Bond and stock

The record also contains the various fields shown on the File Description form, figure 6b.

Table 011 of figure 6a tests for participation in either or both plans. Rule 2 reads:

If The employee is buying bonds

then various employee information is placed in a field named employee identification

and

his previous bond balance plus the amount of this bond deduction is placed in the field named bond balance

and

DO table 012

and

GO TO table 016

	TABLE ID / SER	TABLE NAME / OPER OPERAND 1	TABLE TYPE / OP OPERAND 2	01 OP	01 OPERAND 2	02 OP	02 OPERAND 2	03 OP	03 OPERAND 2	04 OP	04 OPERAND 2
**	011	DED ANALYSIS	OPEN	40		15		35		10	
	1	CODE: BS	EQ	'N'		'B'		'S'		'A'	
	2	SET EMP IDENTIF	EQ EMP			X		X		X	
	3	SET BOND BAL: PURCHASE	EQ BOND BAL: BS + BOND DED			X				X	
	4	SET STK BAL: PURCHASE	EQ STK BAL: BS + STK DED					X		X	
	5	DO				012		013		012	
	6	DO								013	
	7	GO TO		023		016		016		016	
**	012	BOND PURCHASE TEST	CLOSED	35		ELSE					
	1	BOND BAL: PURCHASE		GE	BOND PURCHASE PRICE						
	2	SET BOND PRICE	EQ		BOND PURCHASE PRICE						
	3	SET CODE: PURCHASE	EQ		'B'						
	4	SET BOND BAL: PURCHASE	-		BOND PURCHASE PRICE						
	5	DO	PREPARE P.O.	X							
	6	DO	012	X							
**	013	STK PURCHASE TEST	CLOSED	30	E						
	1	STK BAL: PURCHASE	GE MKT VALUE	Y							
	2	SET STK PRICE	EQ MKT VALUE	X							
	3	SET STK BAL: PURCHASE	- MKT VALUE	X							
	4	SET CODE: PURCHASE	EQ 'S'	X							
	5	DO	PREPARE P.O.	X							
	6	DO	013	X							

REMARKS: EMP SE JOIN(LOC NO: BS, DEPT: BS, EMP NO: BS, EMP NAME: BS)
MKT VALUE AE (DAYS HIGH PRICE + DAYS LOW PRICE)/2

Figure 6a. Decision Tables, Example 6.

IBM
FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
DEDUCTION	IN	SER-CARD	BOND STOCK DED	BS	5000	7000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP NAME	A		20	✓			1						
2	EMP NO	N	U	6				1						
3	BOND BAL	N	+	3.2				1						
4	BOND DED	N	+	3.2				1						
5	BOND PURCHASE PRICE	N	+	3.2				1						
6	STK BAL	N	+	4.2				1						
7	STK DED	N	+	3.2				1						
8	CODE	A		1				1						
9	DEPT	N	U	3				1						
10	LOC NO	N	U	3				1						

Figure 6b. File Description 1, Example 6.

IBM
FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
CUR DED BAL	OUT	SER-CARD	PURCHASE		1000	1500

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP IDENTIF	A		3.2				1						
2	BOND BAL	N	+	3.2		0	.5	1						
3	BOND PRICE	N	+	3.2		0	.5	1						
4	STK BAL	N	+	4.2		0	.5	1						
5	STK PRICE	N	+	3.2		0	.5	1						
6	CODE	A		1				1						

Figure 6c. File Description 2, Example 6.

In the remarks section of the decision table it is noted that the EMP is a symbolic expression name (denoted by SE). In the table, this name has no identifying marks to indicate that it is such an expression. A symbolic expression relates various symbolic values much like an arithmetic expression relates numeric values. JOIN is the only operator; it describes the stringing together of two or more fields to form a new field. Whenever expressions like this are used in decision tables, they are immediately evaluated, and re-evaluated every time they appear.

In this example, the fields LOC NO, DEPT, EMP NO, and EMP NAME are qualified as being part of the record (see figure 6b) which is abbreviated BS. Since these are separate fields, it is not possible to say whether or not they are physically adjacent to one another. In order to be able to use them as a single value, the analyst describes the joining of the appropriate fields. If the expression is used only on a single Decision Table form, it need only be defined in the remarks, but if the expression is used throughout the system, it should be defined on the Reference Operand Description form. In either case, it is defined by writing the expression name, the type SE (symbolic expression), and the operator JOIN. The fields to be joined are then placed in parenthesis after the operator. As usual, the fields are qualified if they are not unique.

The JOIN operator may be used to form symbolic expressions by the joining of fields, literals, constants, descriptive terms, work field names, and reference list function names.

TABLE ID SER	TABLE NAME OPER	OPERAND 1	TABLE TYPE OP	OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
** 011	DED ANALYSIS		OPEN		40	15	35	10
1	CODE: BS		EQ		'M'	'B'	'S'	'A'
2	SET EMP IDENTIF		EQ EMP			X	X	X
3	SET BOND BAL: PURCHASE		EQ BOND BAL: BS + BOND DED			X		X
4	SET STK BAL: PURCHASE		EQ STK BAL: BS + STK DED				X	X
5	DO					012	013	012
6	DO							013
7	GO TO				02 5	016	016	016
** 012	BOND PURCHASE TEST		CLOSED		35	ELSE		
1	BOND BAL: PURCHASE				GE	BOND PURCHASE PRICE		
2	SET BOND PRICE		EQ			BOND PURCHASE PRICE		
3	SET CODE: PURCHASE		EQ			'B'		
4	SET BOND BAL: PURCHASE		-			BOND PURCHASE PRICE		
5	DO		PREPARE P.O.		X			
6	DO		012		X			
** 013	STK PURCHASE TEST		CLOSED		30	E		
1	STK BAL: PURCHASE		GE MKT VALUE		Y			
2	SET STK PRICE		EQ MKT VALUE		X			
3	SET STK BAL: PURCHASE		- MKT VALUE		X			
4	SET CODE: PURCHASE		EQ 'S'		X			
5	DO		PREPARE P.O.		X			
6	DO		013		X			

REMARKS: EMP SE JOIN(LOC NO: BS, DEPT: BS, EMP NO: BS, EMP NAME: BS)
MKT VALUE AE (DAYS HIGH PRICE + DAYS LOW PRICE)/2

Figure 6a. Decision Tables, Example 6.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
DEDUCTION	IN	SER-CARD	BOND STOCK DED	BS	5000	7000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP NAME	A		20	✓			1						
2	EMP NO	N	U	6				1						
3	BOND BAL	N	+	3.2				1						
4	BOND DED	N	+	3.2				1						
5	BOND PURCHASE PRICE	N	+	3.2				1						
6	STK BAL	N	+	4.2				1						
7	STK DED	N	+	3.2				1						
8	CODE	A		1				1						
9	DEPT	N	U	3				1						
10	LOC NO	N	U	3				1						

Figure 6b. File Description 1, Example 6.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
CUR DED BAL	OUT	SER-CARD	PURCHASE		1000	1500

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP IDENTIF	A		3.2				1						
2	BOND BAL	N	+	3.2		0	.5	1						
3	BOND PRICE	N	+	3.2		0	5	1						
4	STK BAL	N	+	4.2		0	5	1						
5	STK PRICE	N	+	3.2		0	.5	1						
6	CODE	A		1				1						

Figure 6c. File Description 2, Example 6.

The next to last action of rule 2 in table 011 is: DO 012. DO is a new operator which calls for executing the table specified, and then returning to the action which follows the DO in the particular rule. This corresponds closely to the linkage with a closed subroutine in normal programming. It relinquishes control to the identified table, and then returns when that table has been executed.

The type of table which is entered by a DO operator is a closed table. If a closed table contains a GO TO operator the return will be broken for that rule. Both open and closed tables may contain DO operators.

Rules pertaining to open and closed tables can be summarized in this chart:

	open table	closed table
entered by	GO TO	DO
exited by	GO TO	(automatic return to calling table)
may contain	DO, and GO TO	DO

The use of the automatic return for a closed table is shown in table 012. This table is entered by the DO operator of rules 2 or 4 of table 011. There are two rules for table 012; rule 1 is:

If the bond balance is greater than or equal to (GE) the bond purchase price,

then the bond purchase price is placed in the bond price, a 'B' is

TABLE NUMBER	TABLE NAME	TABLE TYPE	FREQUENCY			
USER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
* * 011	DED ANALYSIS	OPEN	40	15	35	10
1	CODE : BS	EQ	'W'	'B'	'S'	'A'
2	SET EMP IDENTIF	EQ EMP		X	X	X
3	SET BOND BAL: PURCHASE	EQ BOND BAL : BS + BOND DED		X		X
4	SET STK BAL: PURCHASE	EQ STK BAL : BS + STK DED			X	X
5	DO			012	013	012
6	DO					013
7	GO TO		025	016	016	016
* * 012	BOND PURCHASE TEST	CLOSED	35	ELSE		
1	BOND BAL: PURCHASE		GE	BOND PURCHASE PRICE		
2	SET BOND PRICE	EQ		BOND PURCHASE PRICE		
3	SET CODE: PURCHASE	EQ		'B'		
4	SET BOND BAL: PURCHASE	-		BOND PURCHASE PRICE		
5	DO	PREPARE P.O.	X			
6	DO	012	X			
* * 013	STK PURCHASE TEST	CLOSED	30	E		
1	STK BAL: PURCHASE	GE MKT VALUE	Y			
2	SET STK PRICE	EQ MKT VALUE	X			
3	SET STK BAL: PURCHASE	- MKT VALUE	X			
4	SET CODE : PURCHASE	EQ 'S'	X			
5	DO	PREPARE P.O.	X			
6	DO	013	X			

REMARKS: EMP SE JOIN(LCC NO: BS, DEPT: BS, EMP NO: BS, EMP NAME: BS)
MKT VALUE AE (DAYS HIGH PRICE + DAYS LOW PRICE)/2

Figure 6a. Decision Tables, Example 6.

IBM
FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
DEDUCTION	IN	SER. CARD	BOND STOCK DED	BS	5000	7000

SER NO	FIELD NAME	NUMERIC/ALPHA	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP NAME	A		20	✓			1						
2	EMP NO	N	U	6				1						
3	BOND BAL	N	+	3.2				1						
4	BOND DED	N	+	3.2				1						
5	BOND PURCHASE PRICE	N	+	3.2				1						
6	STK BAL	N	+	4.2				1						
7	STK DED	N	+	3.2				1						
8	CODE	A		1				1						
9	DEPT	N	U	3				1						
10	LCC NO	N	U	5				1						

Figure 6b. File Description 1, Example 6.

IBM
FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
CUR DED BAL	OUT	SER-CARD	PURCHASE		1000	1500

SER NO	FIELD NAME	NUMERIC/ALPHA	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP IDENTIF	A		3.2				1						
2	BOND BAL	N	+	3.2		0	.5	1						
3	BOND PRICE	N	+	3.2		0	5	1						
4	STK BAL	N	+	4.2		0	5	1						
5	STK PRICE	N	+	3.2		0	.5	1						
6	CODE	A		1				1						

Figure 6c. File Description 2, Example 6.

placed in the purchase code, subtract the purchase price from the bond balance, prepare a purchase order, and do this table again.

The second time the table is entered, this time by its own DO operator, the actions of rule 1 are executed again if the condition is still satisfied. When the bond balance is not greater than or equal to the bond purchase price, then the conditions of the rule are not satisfied, and the ELSE rule is called. The ELSE rule has no actions, so the table is complete and control is returned to the original table (table 011), and to the action following the first DO action.

Rule 3 of table 011 specifies the logic for a man who participates in the stock purchase plan only, and rule 4 covers the employee who is in both plans.

Table 013 differs from table 012 in that the current market value of the stock is used in place of the fixed purchase price of the bond. The current market value is developed through the use of an arithmetic expression (AE) explained at the base of the form in the remarks section, as was the symbolic expression. An arithmetic expression is a description of a combination of operands and arithmetic operators which has a numeric value when evaluated. The operands can be numeric fields, numeric constants, numeric literals, numeric work fields and reference list function names. The five operators used in arithmetic expressions are:

+ addition, - subtraction, * multiplication, / division, ** exponentiation.

TABLE ID	TABLE NAME	TABLE TYPE		FREQUENCY			
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
011	DED ANALYSIS	OPEN	40	15	35	10
1	CODE: BS	EQ	'W'	'B'	'S'	'A'
2	SET EMP IDENTIF	EQ EMP		X	X	X
3	SET BOND BAL: PURCHASE	EQ BOND BAL: BS + BOND DED		X		X
4	SET STK BAL: PURCHASE	EQ STK BAL: BS + STK DED			X	X
5	DO			01 2	01 3	01 2
6	DO					01 3
7	GO TO		02 3	01 6	01 6	01 6
012	BOND PURCHASE TEST	CLOSED	35	ELSE		
1	BOND BAL: PURCHASE		GE	BOND PURCHASE PRICE		
2	SET BOND PRICE	EQ		BOND PURCHASE PRICE		
3	SET CODE: PURCHASE	EQ		'B'		
4	SET BOND BAL: PURCHASE	-		BOND PURCHASE PRICE		
5	DO	PREPARE P.O.	X			
6	DO	012	X			
013	STK PURCHASE TEST	CLOSED	30	E		
1	STK BAL: PURCHASE	GE MKT VALUE	Y			
2	SET STK PRICE	EQ MKT VALUE	X			
3	SET STK BAL: PURCHASE	- MKT VALUE	X			
4	SET CODE: PURCHASE	EQ 'S'	X			
5	DO	PREPARE P.O.	X			
6	DO	013	X			

REMARKS: EMP SE JOIN(LOC NO: BS, DEPT: BS, EMP NO: BS, EMP NAME: BS)
MKT VALUE AE (DAYS HIGH PRICE + DAYS LOW PRICE)/2

Figure 6a. Decision Tables, Example 6.

IBM

SYSTEM NAME: PAYROLL

FILE DESCRIPTION

ANALYST: JONES | DATE: 1/3/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABS	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
DEDUCTION	IN	SER-CARD	BOND STOCK DED	BS	5000	7000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP NAME	A		20	✓			1						
2	EMP NO	N	U	6				1						
3	BOND BAL	N	+	3 2				1						
4	BOND DED	N	+	3 2				1						
5	BOND PURCHASE PRICE	N	+	3 2				1						
6	STK BAL	N	+	4 2				1						
7	STK DED	N	+	3 2				1						
8	CODE	A		1				1						
9	DEPT	N	U	3				1						
10	LOC NO	N	U	3				1						

Figure 6b. File Description 1, Example 6.

IBM

SYSTEM NAME: PAYROLL

FILE DESCRIPTION

ANALYST: JONES | DATE: 1/3/62 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABS	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
CUR DED BAL	OUT	SER-CARD	PURCHASE		1000	1500

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP IDENTIF	A		3,2				1						
2	BOND BAL	N	+	3,2		0	.5	1						
3	BOND PRICE	N	+	3,2		0	5	1						
4	STK BAL	N	+	4,2		0	5	1						
5	STK PRICE	N	+	3,2		0	.5	1						
6	CODE	A		1				1						

Figure 6c. File Description 2, Example 6.

If an arithmetic expression is described in the remarks section of a table, it is a local arithmetic expression. When it is used throughout the system, it should be described on a Reference Operand Description form. In figure 6a, the local arithmetic expression is the market stock value. It is defined as: The day's high price for the stock, plus the day's low price for the stock; the sum divided by two.

QUESTIONS

1. What are the differences between an open table and a closed table?
2. What is the definition of a symbolic expression? How is it used?
3. What is the definition of an arithmetic expression? Where is it defined in TDL?
4. What is the sequence of actions for rule 4 of table 011 of this example?
5. In table 012, read both rules.

	SER	TABLE NAME OPER	OPERAND 1	TABLE TYPE OP	OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
* *	011	DED ANALYSIS		OPEN		40	15	35	10
	1	CODE : BS		EQ		'N'	'B'	'S'	'A'
	2	SET EMP IDENTIF		EQ EMP			X	X	X
	3	SET BOND BAL: PURCHASE		EQ BOND BAL : BS + BOND DED			X		X
	4	SET STK BAL: PURCHASE		EQ STK BAL : BS + STK DED				X	X
	5	DO					012	013	012
	6	DO							013
	7	GO TO				025	016	016	016
* *	012	BOND PURCHASE TEST		CLOSED		35	ELSE		
	1	BOND BAL : PURCHASE				GE	BOND PURCHASE PRICE		
	2	SET BOND PRICE		EQ			BOND PURCHASE PRICE		
	3	SET CODE : PURCHASE		EQ			'B'		
	4	SET BOND BAL: PURCHASE		-			BOND PURCHASE PRICE		
	5	DO		PREPARE P.O.		X			
	6	DO		012		X			
* *	013	STK PURCHASE TEST		CLOSED		30	E		
	1	STK BAL : PURCHASE		GE MKT VALUE		Y			
	2	SET STK PRICE		EQ MKT VALUE		X			
	3	SET STK BAL: PURCHASE		- MKT VALUE		X			
	4	SET CODE : PURCHASE		EQ 'S'		X			
	5	DO		PREPARE P.O.		X			
	6	DO		013		X			

REMARKS: EMP SE JOIN(LOC NO: BS, DEPT: BS, EMP NO: BS, EMP NAME: BS)
MKT VALUE AE (DAYS HIGH PRICE + DAYS LOW PRICE)/2

Figure 6a. Decision Tables, Example 6.

IBM
FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
DEDUCTION	IN	SER-CARD	BOND STOCK DED	BS	5000	7000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP NAME	A		20	✓			1						
2	EMP NO	N	U	6				1						
3	BOND BAL	N	+	3.2				1						
4	BOND DED	N	+	3.2				1						
5	BOND PURCHASE PRICE	N	+	3.2				1						
6	STK BAL	N	+	4.2				1						
7	STK DED	N	+	3.2				1						
8	CODE	A		1				1						
9	DEPT	N	U	3				1						
10	LOC NO	N	U	3				1						

Figure 6b. File Description 1, Example 6.

IBM
FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
PAYROLL	JONES	1/3/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
CUR DED BAL	OUT	SER-CARD	PURCHASE		1000	1500

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EMP IDENTIF	A		3.2				1						
2	BOND BAL	N	+	3.2		0	.5	1						
3	BOND PRICE	N	+	3.2		0	5	1						
4	STK BAL	N	+	4.2		0	5	1						
5	STK PRICE	N	+	3.2		0	.5	1						
6	CODE	A		1				1						

Figure 6c. File Description 2, Example 6.

ANSWERS

1. An open table is entered by a GO TO operator and terminates with a GO TO operator. The closed table is entered by a DO operator and returns automatically to the action after the DO action of the same rule of the table from which it was called.

2. A symbolic expression causes a stringing together of two or more fields to form a new field.

3. An arithmetic expression is a combination of numeric operands with arithmetic operators. This expression has a numeric value when evaluated. It may be defined either in the Remarks section of the tables or on a Reference Operand Description form.

4. If the conditions of the rule are satisfied, the actions are carried out to the DO action. The first DO action calls table 012, and the return is to rule 4 for the next DO action. This DO action calls table 013; return is to rule 4 again for execution of the GO TO 016 action.

5. a) If the bond balance is greater than or equal to the bond purchase price, then the bond purchase price is placed in the bond price, a 'B' is placed in the code of the purchase record, the bond purchase price is subtracted from the bond balance, a purchase order is prepared, and the table is repeated.

   b) If the condition of the first rule is not satisfied, the ELSE rule is called, the table is completed and return is to the next action of the rule in the table from which it came.

SYSTEM EXPENSE DIST. ANALYST S.P.M. DATE 11/28/61 PAGE__OF____

FILE NAME
DEPT EXP REPT
8 CONSTANT

```
                MONTHLY EXPENSE DISTRIBUTION REPORT    REP DATE    PAGE
                                                       AAA NN NNNN NNNN

                DEPT      ACCT      INVOICE      INVOICE      INVOICE
                 NO        NO         NO           DATE        AMOUNT

2 DEPT DETAIL-V NNNN      NNNNN     AA-NNNN      ZN ZN NN    ZZ,ZZZ.NNCR


2 LEDGER TOT-V  *** GEN LEDGER NN                           Z,ZZZ,ZZZ.NNCR


2 END REPORT-V                                              LAST PAGE
4 CONSTANT
```

Figure 7a. Report Layout Sheet, Example 7.

The rules of format for preparing reports in Tabular Descriptive Language are described in the context of their application to the following forms:

(1) Report Layout sheet, Figure 7a

(2) File Description forms, Figures 7b, d, and e

(3) Decision Tables, Figure 7c.

Example 7 involves the compilation and printing of a Monthly Expense Distribution Report in the system EXPENSE DIST. The collection of records needed to make up the report are indicated in the left hand margin of the Report Layout sheet. In this case, the records LEDGER TOT and DEPT DETAIL are part of the file DEPT EXP REPT, which represents a complete report description.

Information at the very top of the layout is filled in by the analyst:

SYSTEM NAME
ANALYST (name)
DATE (the sheet is prepared)
PAGE (number of number)

MONTHLY EXPENSE DISTRIBUTION REPORT, DEPT NO, ACCT NO, INVOICE NO, INVOICE DATE, and INVOICE AMOUNT are headings. Constant information that is to be printed in the same location on every page of the report is identified by the record name CONSTANT and so marked in the left hand margin of the Report Layout sheet. Every page will carry this exact same information, printed in the same position,

SYSTEM EXPENSE DIST. ANALYST S.P.M. DATE 11/28/61 PAGE__OF____

FILE NAME
DEPT EXP REPT
8 CONSTANT

```
                MONTHLY EXPENSE DISTRIBUTION REPORT     REP DATE    PAGE
                                                        AAA NN NNNN NNNN

                DEPT        ACCT        INVOICE         INVOICE     INVOICE
                 NO          NO          NO              DATE        AMOUNT
2 DEPT DETAIL-V NNNN        NNNNN       AA-NNNN         ZN ZN NN    ZZ,ZZZ.NNCR


2 LEDGER TOT-V  *** GEN LEDGER NN                                   Z,ZZZ,ZZZ.NNCR
```

2 END REPORT-V LAST PAGE

4 CONSTANT

Figure 7a. Report Layout Sheet, Example 7.

automatically and without instructions. CONSTANT is prefixed by a number (e.g., 8) to state the number of lines that comprise the record. No File Description form is required, since no supplemental information is needed.

The representative date and page number fields are considered part of this record even though their values will change; the programmer has the responsibility for inserting sequential page numbers and the current date on each page.

Record names (e.g., DEPT DETAIL) are suffixed with a V or F which designate vertical placement of the final printed line on the form. V means that printing may occur at any position on the page, except where the space has been pre-empted by the higher priority CONSTANT. F indicates that the position of the final printed line is fixed at that vertical position on the page. Since the term CONSTANT denotes fixed position, no suffix is needed.

The number preceding record name indicates the number of lines required for that record. Reading from the Report Layout, DEPT DETAIL and LEDGER TOT each require two lines. Here, even though just one line is printed, the blank line will be skipped before the next record is written. The number includes the actual number of printed lines as well as the lines not to be printed.

Any record described by the analyst may be composed entirely of pre-assigned information or some combination of pre-assigned and variable information.

SYSTEM EXPENSE DIST. ANALYST S.P.M. DATE 11/28/61 PAGE__OF____

FILE NAME
DEPT EXP REPT
8 CONSTANT

MONTHLY EXPENSE DISTRIBUTION REPORT REP DATE AAA NN NNNN PAGE NNNN

	DEPT NO	ACCT NO	INVOICE NO	INVOICE DATE	INVOICE AMOUNT
2 DEPT DETAIL-V	NNNN	NNNNN	AA-NNNN	ZN ZN NN	ZZ,ZZZ.NNCR
2 LEDGER TOT-V	*** GEN LEDGER NN				Z,ZZZ,ZZZ.NNCR
2 END REPORT-V				LAST PAGE	

4 CONSTANT

Figure 7a. Report Layout Sheet, Example 7.

The record END REPORT is an example of pre-assigned information. This is a record consisting of two lines, one of which is blank and the other contains the words LAST PAGE. The LAST PAGE designation will print at least five lines from the bottom (four CONSTANT). On the Report Layout sheet, the instruction in the left margin specifies exactly what information will appear and its precise position vertically.

Variable information on a record is specified by indicating where it is to appear within the record with the use of special editing characters. Some of these are described below, but others are given in the reference section of the manual.

CHARACTER	MEANING AND USE
N	An N indicates that the character position will always contain a numeric character.
.	A decimal or period will occur where indicated.
A	An A means that the character position can contain any alphameric character.
Z	The character Z specifies zero suppression of the indicated characters. This is the process of replacing unwanted left-hand zeros by blanks.
,	A comma will be inserted only if the value is large enough to have non-zeros to the left of it.
CR	The symbol CR appears at the right end of the field description and occupies two character positions. When the value of the field is negative, this symbol will be placed at the right end of the item.

SYSTEM NAME: EXPENSE DIST

IBM
FILE DESCRIPTION

ANALYST: J.P.M. | DATE: 11/28/61 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
DEPT EXP REPT	OUT	SER-PRINT	DEPT DETAIL	DD	18,000	25,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENGTH MAX	LENGTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	DEPT NO	N	U	4				1				1	13	
2	ACCT NO	N	U	5				1				1	25	
3	GEN LEDGER NO	N	U	2				1				1	22	ACCT NO
4	SUB LEDGER NO	N	U	3				1				1	25	ACCT NO
5	INV NO TYPE	A		2				1				1	32	
6	INV NO	N	U	4				1				1	37	
7	INV DATE	N	U	6				1				1	51	
8	INV MO	N	U	2				1				1	45	INV DATE
9	INV DAY	N	U	2				1				1	48	INV DATE
10	INV YR	N	U	2				1				1	51	INV DATE
11	INV AMT	N	E	5 2				1				1	64	

Figure 7b. File Description 1, Example 7.

SYSTEM NAME: EXPENSE DIST.

IBM
FILE DESCRIPTION

ANALYST: JPM | DATE: 11/28/61 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
EXPENSE FILE	IN	SER-MAG TAPE	EXPENSE DETAIL	ED	18,000	25,000

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENGTH MAX	LENGTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	DEPT NO	N	U	4				1			3			
2	ACCT NO	N	U	5				1						
3	GEN LEDGER NO	N	U	2				1			1	1	2	ACCT NO
4	SUB LEDGER NO	N	U	3				1			2	3	5	ACCT NO
5	INV NO TYPE	A		2				1						
6	INV NO	N	U	4				1						
7	INV AMT	N	E	5.2				1						
8	INV DATE	N	U	6				1						
9	INV MO	N	U	2				1				1	2	INV DATE
10	INV DAY	N	U	2				1				3	4	INV DATE
11	INV YR	N	U	2				1				5	6	INV DATE

Figure 7d. File Description 2, Example 7.

SYSTEM NAME: EXPENSE DIST.

IBM
FILE DESCRIPTION

ANALYST: JPM | DATE: 11/28/61 | PAGE OF PAGES

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
DEPTEXP REPT	OUT	SER-PRINT	LEDGER TOT	LT	70	99

SER NO	FIELD NAME	ALPHA/NUMERIC	SIGN	LENGTH MAX	LENGTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	GEN LEDGER NO	N	U	2				1				1	22	
2	INV TOT LEDG	N	E	7.2				1				1	64	

Figure 7e. File Description 3, Example 7.

All of these editing characters are displayed in Figure 7a. For example, the record DEPT DETAIL consists of several fields. The first of these is a four character numeric field, the second a five character numeric field, etc.

A File Description form (Figure 7b for DEPT DETAIL output, 7d for EXPENSE DETAIL input, and 7e for LEDGER TOT) is prepared for each record in the file to report positions with field names. Pre-assigned information cannot be included through the File Description. For each field, the position of the character located farthest to the right is specified in the END column of RELATIVE LOCATION; the BEGIN column indicates which line of the record contains the field. If there may be confusion, the beginning position may also be shown:

INVOICE NO 1, 34 37

The first field of Figure 7b to be printed is DEPT NO. From the File Description, this field is a numeric, unsigned field of four characters, printed on the first line of the record, with the last character in position 13. Another example is shown with the field INV AMT. This is a numeric field, which can be either positive or negative, and has a length of five integer and two decimal characters. The field is written on the first line of the record and the character farthest to the right is in position 64. On the Report Layout sheet INVOICE AMOUNT has a field length of 11 characters counting commas, periods, and the final CR. This difference in length between the Report Layout sheet and the File

	TABLE NAME	TABLE TYPE	FREQUENCY																													
	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2					
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 005	GET INPUT	OPEN	99	1																												
1	EXPENSE FILE	IS EOF	N	Y																												
2	WRITE	LEDGER TOT		X																												
3	WRITE	END REPORT		X																												
4	GO TO	STOP		X																												
5	READ	EXPENSE FILE	X																													
6	GO TO	REPT PREP	X																													
** 006	REPT PREP	OPEN	99	1																												
1	GEN LEDGER NO: ED	EQ GEN LEDGER NO: LT	Y	N																												
2	WRITE	LEDGER TOT		X																												
3	MOVE EXPENSE DETAIL	TO LEDGER TOT		X																												
4	MOVE EXPENSE DETAIL	TO DEPT DETAIL	X	X																												
5	WRITE	DEPT DETAIL	X	X																												
6	SET INV TOT LEDG	+ INV AMT: ED	X																													
7	GO TO	GET INPUT	X	X																												

Figure 7c. Decision Tables, Example 7.

Description sheet (5.2) indicates that editing characters such as commas and decimal points, are not counted in the total length of the field.

The decision tables shown in Figure 7c reveal the operation of writing the report described in the previous text. Rule 1 of table 006 is a typical example of the sequence of operations. The condition of rule 1 requires that the value of the fields GEN LEDGER NO of records ED and LT are identical. When such is the case, the record EXPENSE DETAIL is moved to the record DEPT DETAIL. This is the output record which is to be printed on the output report. The next action of rule 1 is to actually write the lines for DEPT DETAIL. The remainder of the decision table shows the sequence of making up and printing the output record.

QUESTIONS

1. What are the requirements for the description of the output records on the File Description forms?
2. What are the meanings of the characters Z, A, and N on the Report Layout sheet?
3. What is a CONSTANT record?
4. On the Report Layout sheet, how is a record identified?
5. Read rule 2 of table 005.

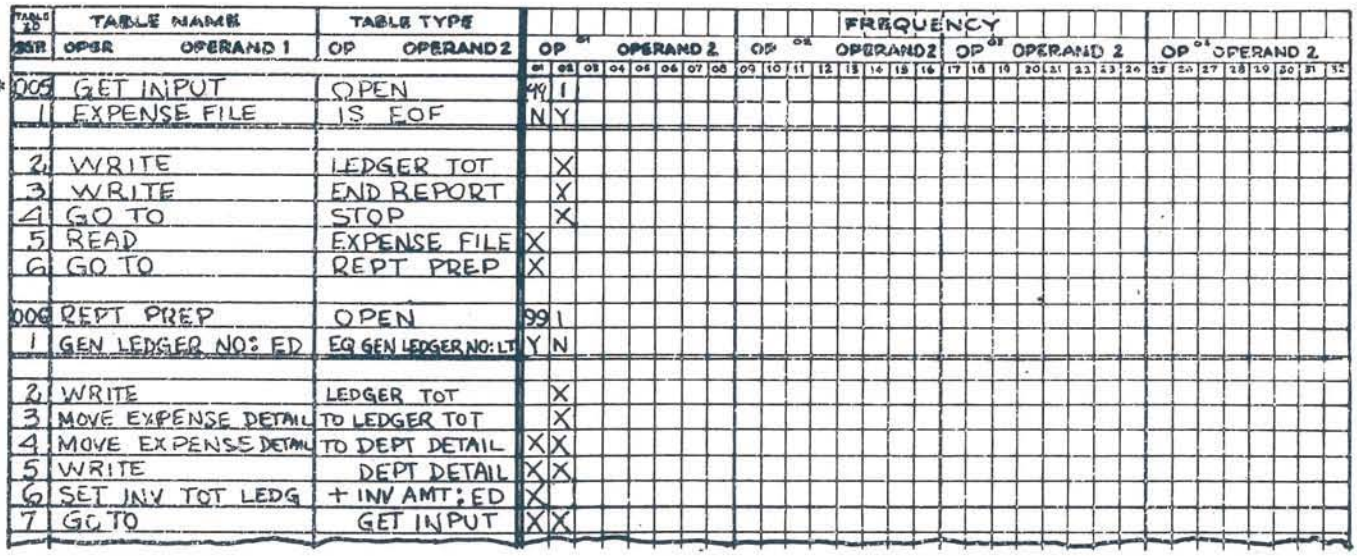


	TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY																											
	RSR	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2			
				01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
**	005	GET INPUT	OPEN	99	1																												
	1	EXPENSE FILE	IS EOF	N	Y																												
	2	WRITE	LEDGER TOT		X																												
	3	WRITE	END REPORT		X																												
	4	GO TO	STOP		X																												
	5	READ	EXPENSE FILE	X																													
	6	GO TO	REPT PREP	X																													
**	006	REPT PREP	OPEN	99	1																												
	1	GEN LEDGER NO: ED	EQ GEN LEDGER NO: LT	Y	N																												
	2	WRITE	LEDGER TOT		X																												
	3	MOVE EXPENSE DETAIL	TO LEDGER TOT		X																												
	4	MOVE EXPENSE DETAIL	TO DEPT DETAIL	X	X																												
	5	WRITE	DEPT DETAIL	X	X																												
	6	SET INV TOT LEDG	+ INV AMT: ED	X																													
	7	GO TO	GET INPUT	X	X																												

Figure 7c. Decision Tables, Example 7.

ANSWERS

1. The output records are described on the File Description forms and should contain all the variable fields of the record, plus a description in the column labelled RELATIVE LOCATION. This description must describe the line of the record in which the field is written, and the position of the right hand character of the field.

2. Z, A, and N are used for editing. The characters A and N specify alphameric and numeric data respectively, while the Z indicates zero suppression.

3. A CONSTANT record is one which describes those parts of a report which reappear in exactly the same position from page to page with the same information.

4. A record name is indicated with a prefix and a suffix in the left column on the first line where its format is described. The prefix indicates the number of lines to the record, and the suffix indicates whether the record is in a fixed vertical position or if it may vary in vertical position.

5. If EXPENSE FILE is at end of file, write the ledger total, write the end report, and go to the STOP table.

IBM DECISION TABLE

SYSTEM NAME	PROCEDURE NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	INVOICE DETAIL	FRANCIS	1/4/62	

	TABLE ID / SER	TABLE NAME / OPER	OPERAND 1	TABLE TYPE / OP	OPERAND 2	FREQUENCY OP	OPERAND 2	OP	OPERAND 2	OP	OPERAND 2	OP	OPERAND 2
**	071	INVOICE		OPEN-FIRST									
		DO		NEXT ITEM	X								
		SET LOC ADD		EQ ITEM FORMULA	X								
		DO		NEXT INVEN	X								
		SET QTY ON HAND		- QTY SOLD	X								
		MOVE PRICE EXT		TO NET PRICE	X								
		MOVE ITEM		TO ITEM LINE	X								
		DO		INVOICE HEADER	X								
		GO TO		071	X								
**	072	NEXT ITEM		CLOSED	1	99							
		ITEM CARD		IS EOF	Y	N							
		READ		ITEM CARD		X							
		DO		STOP	X								
**	073	NEXT INVEN		CLOSED	50	50							
		ITEM NO: ITEM		EQ ITEM NO MASTER	Y	N							
		SET LOC ADD		+ 1		X							
		READ		INVEN MASTER PER LOC ADD		X							
		DO		073		X							
**	074	INVOICE HEADER		CLOSED	90	10							
		CUST NO. ITEM		EQ CUST NO: HEADER	Y	N							
		WRITE		ITEM LINE	X								
		DO		TOTAL CALL		X							
		DO		PROCEDURE HEADER PREP		X							
		DO		074		X							

REMARKS:

Figure 8a. Decision Table, Example 8.

IBM FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	FRANCIS	1/4/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
ITEM CARD	IN	SER-CARD	ITEM-1		4500	6000

SER NO	FIELD NAME	CHARACTER / NUMERIC	SIGN	LENGTH MAX	LENGTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	CARD CODE	N	U	1				1						
2	CUST NO	N	U	5				1			1			
3	ITEM NO	N	U	6				1						
4	QTY SOLD	N	+	3				1						
5	UNIT PRICE	N	+	4.2				1						

Figure 8b. File Description 1, Example 8.

IBM FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	FRANCIS	1/4/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVEN MASTER	RETAINED	RAN-DESK	MASTER		10,000	12,000

SER NO	FIELD NAME	CHARACTER / NUMERIC	SIGN	LENGTH MAX	LENGTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	6				1						
2	CLASS OF ITEM	A		3				1						
3	STOCK LOC	A		6				1						
4	DESCRIPTION	A		25	✓			1						
5	QTY ON HAND	N	+	6				1						

Figure 8c. File Description 2, Example 8.

Example 8 demonstrates a number of approaches for reading and writing records in files and explains the various ways of correcting files, depending on their classification. The procedure outlined in this example is part of the system BILLING and includes (1) reading item cards in a file, (2) getting an inventory master, and (3) proceeding with invoice preparation.

The first action in the open table 071 shown in Figure 8a is:

DO NEXT ITEM

which is the title of table 072. Control is temporarily transferred to this table through the DO operator.

Table 071 differs from earlier samples in that no conditions have been written into its structure. Whenever this happens, actions are executed upon entry from another table, and the arrangement is referred to as an unconditional table.

In closed table 072, the first action:

ITEM CARD IS EOF

tests whether the item card file is at the end of the file. This action is used only with a special kind of file, one in which records appear one after the other as in a deck of punched cards or a magnetic tape. When a file is to be treated in this way, it is known as a serial file.

A detailed summary of the serial file ITEM CARD is supplied in the File Description form, Figure 8b. Across the header essential

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAGE OF PAGES
BILLING	INVOICE DETAIL		FRANCIS	1/4/62	

	TABLE ID / SER	TABLE NAME / OPER OPERAND 1	TABLE TYPE / OP OPERAND 2	01	02
**	071	INVOICE	OPEN-FIRST		
		DO	NEXT ITEM	X	
		SET LOC ADD	EQ ITEM FORMULA	X	
		DO	NEXT INVEN	X	
		SET QTY ON HAND	- QTY SOLD	X	
		MOVE PRICE EXT	TO NET PRICE	X	
		MOVE ITEM	TO ITEM LINE	X	
		DO	INVOICE HEADER	X	
		GO TO	071	X	
**	072	NEXT ITEM	CLOSED	1	99
		ITEM CARD	IS EOF	Y	N
		READ	ITEM CARD		X
		DO	STOP	X	
**	073	NEXT INVEN	CLOSED	50	50
		ITEM NO: ITEM	EQ ITEM NO MASTER	Y	N
		SET LOC ADD	+ 1		X
		READ	INVEN MASTER PER LOC ADD		X
		DO	073		X
**	074	INVOICE HEADER	CLOSED	90	10
		CUST NO. ITEM	EQ CUST NO: HEADER	Y	N
		WRITE	ITEM LINE	X	
		DO	TOTAL CALL		X
		DO	PROCEDURE HEADER PREP		X
		DO	074		X

REMARKS:

Figure 8a. Decision Table, Example 8.

SYSTEM NAME	IBM FILE DESCRIPTION	ANALYST	DATE	PAGE OF PAGES
BILLING		FRANCIS	1/4/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	PEAK
ITEM CARD	IN	SER-CARD	ITEM		4500	6000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	AVG	MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	END	VALUE/REMARKS
1	CARD CODE	N	U	1				1						
2	CUST NO	N	U	5				1			1			
3	ITEM NO	N	U	6				1						
4	QTY SOLD	N	+	3				1						
5	UNIT PRICE	N	+	4,2				1						

Figure 8b. File Description 1, Example 8.

SYSTEM NAME	IBM FILE DESCRIPTION	ANALYST	DATE	PAGE OF PAGES
BILLING		FRANCIS	1/4/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	PEAK
INVEN MASTER	RETAINED	RAN-DESK	MASTER		10,000	12,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	AVG	MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	END	VALUE/REMARKS
1	ITEM NO	N	U	6				1						
2	CLASS OF ITEM	A		3				1						
3	STOCK LOC	A		6				1						
4	DESCRIPTION	A		25	✓			1						
5	QTY ON HAND	N	+	6				1						

Figure 8c. File Description 2, Example 8.

descriptive data concerning the file is entered:

File Name - ITEM CARD
Dir. Type - INPUT
Medium - SER-CARD
Record Name - ITEM
etc.

The EOF (END OF FILE) notation looks ahead to see whether there are any more records to process before reaching the end of the file. Y signifies there is no next record; if this condition is not checked at this point, and further operations are called for (e.g., READ ITEM CARD), they will be ignored if the end of file has been reached. An N denotes there is at least one more record.

Table 072, rule 2 calls for the execution of: READ ITEM CARD which arranges for the storage in memory of values for the 5 fields in the record as they are described in Figure 8b. When the EOF is reached, the DO STOP action provides for the completion of certain closing routines.

The operator READ is always used for providing input from a serial file. The operand is always a file name, and the next record in the file is read into memory.

Control now returns to table 071 to get the inventory record for the item number on the item card. In this cycle from the initial DO action in table 071, through the steps in table 072 and back to 071 again, table 071 assumes the role of master coordinator for several closed tables. Basically, it calls on these tables for a series of actions, then asks for

IBM DECISION TABLE

SYSTEM NAME	PROCEDURE NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	INVOICE DETAIL	FRANCIS	1/4/62	

	TABLE ID / SER	TABLE NAME / OPER / OPERAND 1	TABLE TYPE / OP / OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
**	071	INVOICE	OPEN-FIRST				
		DO	NEXT ITEM	X			
		SET LOC ADD	EQ ITEM FORMULA	X			
		DO	NEXT INVEN	X			
		SET QTY ON HAND	- QTY SOLD	X			
		MOVE PRICE EXT	TO NET PRICE	X			
		MOVE ITEM	TO ITEM LINE	X			
		DO	INVOICE HEADER	X			
		GO TO	071	X			
**	072	NEXT ITEM	CLOSED	1	99		
		ITEM CARD	IS EOF	Y	N		
		READ	ITEM CARD		X		
		DO	STOP	X			
**	073	NEXT INVEN	CLOSED	50	50		
		ITEM NO: ITEM	EQ ITEM NO: MASTER	Y	N		
		SET LOC ADD	+ 1		X		
		READ	INVEN MASTER PER LOC ADD		X		
		DO	073		X		
**	074	INVOICE HEADER	CLOSED	90	10		
		CUST NO: ITEM	EQ CUST NO: HEADER	Y	N		
		WRITE	ITEM LINE	X			
		DO	TOTAL CALC		X		
		DO	PROCEDURE HEADER PREP		X		
		DO	074		X		

REMARKS:

Figure 8a. Decision Table, Example 8.

IBM FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	FRANCIS	1/4/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
ITEM CARD	IN	SER-CARD	ITEM-1		4500	6000

SER NO	FIELD NAME	ALPHA NUMERIC	BIN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSC	ASSC TO	SIGNIFICANT	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	CARD CODE	N	U	1				1						
2	CUST NO	N	U	5				1			1			
3	ITEM NO	N	U	6				1						
4	QTY SOLD	N	H	3				1						
5	UNIT PRICE	N	F	4.2				1						

Figure 8b. File Description 1, Example 8.

IBM FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	FRANCIS	1/4/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVEN MASTER	RETAINED	RAN-DESK	MASTER		10,000	12,000

SER NO	FIELD NAME	ALPHA NUMERIC	BIN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSC	ASSC TO	SIGNIFICANT	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	H	6				1						
2	CLASS OF ITEM	A		3				1						
3	STOCK LOC	A		6				1						
4	DESCRIPTION	A		25	✓			1						
5	QTY ON HAND	N	+	6				1						

Figure 8c. File Description 2, Example 8.

return of control to issue further instructions. The cycle may be repeated as often as necessary, depending on the complexity of the procedure.

To get the inventory record for the item number on the item card, the file INVEN MASTER is required. An examination of its File Description form in Figure 8c reveals that this file is on disk and is a retained random disk file. As such, it requires different treatment.

A retained file is one which is used over and over in a system; it exists as an input at some time during the processing cycle and still exists at the end of that cycle.

In a random file, as differentiated from a serial file, reading and writing are performed on the basis of location or information content. A random file has no beginning or end, and each record is viewed as it turns up in the loop. New data may be added to a record while it is in memory, but this new information is superimposed on the previously recorded data, so the older data is lost unless explicitly recorded elsewhere. This provides a way to correct the file automatically.

For the current example, information will be obtained by address. The field LOC ADD is given a value from an arithmetic expression called ITEM FORMULA.

The DO NEXT INVEN action calls for table 073; the first condition:

ITEM NO: ITEM EQ ITEM NO: MASTER

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAGE OF PAGES
BILLING	INVOICE DETAIL		FRANCIS	1/4/62	

	TABLE ID SER	TABLE NAME OPER OPERAND 1	TABLE TYPE OP OPERAND 2	01	02
**	071	INVOICE	OPEN-FIRST		
		DO	NEXT ITEM	X	
		SET LOC ADD	EQ ITEM FORMULA	X	
		DO	NEXT INVEN	X	
		SET QTY ON HAND	- QTY SOLD	X	
		MOVE PRICE EXT	TO NET PRICE	X	
		MOVE ITEM	TO ITEM LINE	X	
		DO	INVOICE HEADER	X	
		GO TO	071	X	
**	072	NEXT ITEM	CLOSED	1	99
		ITEM CARD	IS EOF	Y	N
		READ	ITEM CARD		X
		DO	STOP	X	
**	073	NEXT INVEN	CLOSED	50	50
		ITEM NO: ITEM	EQ ITEM NO MASTER	Y	N
		SET LOC ADD	+ 1		X
		READ	INVEN MASTER PER LOC ADD		X
		DO	073		X
**	074	INVOICE HEADER	CLOSED	90	10
		CUST NO. ITEM	EQ CUST NO: HEADER	Y	N
		WRITE	ITEM LINE	X	
		DO	TOTAL CALL		X
		DO	PROCEDURE HEADER PREP		X
		DO	074		X

REMARKS:

Figure 8a. Decision Table, Example 8.

SYSTEM NAME	IBM FILE DESCRIPTION	ANALYST	DATE	PAGE OF PAGES
BILLING		FRANCIS	1/4/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
ITEM CARD	IN	SER-CARD	ITEM		4500	6000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	CARD CODE	N	U	1				1						
2	CUST NO	N	U	5				1			1			
3	ITEM NO	N	U	6				1						
4	QTY SOLD	N	+	3				1						
5	UNIT PRICE	N	+	42				1						

Figure 8b. File Description 1, Example 8.

SYSTEM NAME	IBM FILE DESCRIPTION	ANALYST	DATE	PAGE OF PAGES
BILLING		FRANCIS	1/4/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVEN MASTER	RETAINED	RAN - DESK	MASTER		10,000	12,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	6				1						
2	CLASS OF ITEM	A		3				1						
3	STOCK LOC	A		6				1						
4	DESCRIPTION	A		25	✓			1						
5	QTY ON HAND	N	+	6				1						

Figure 8c. File Description 2, Example 8.

This is unlikely to be true the first time through, unless the same item number shows up twice in a row for a single customer; the N path (rule 2) will probably be satisfied first. In rule 2, LOC ADD is incremented by 1, then the action READ INVEN MASTER PER LOC ADD is executed. Table 073 is repeated again. When the Y condition is satisfied, control returns to table 071. Incrementing of LOC ADD is simply one way of chaining if the first address is incorrect.

The READ operator for a random file is somewhat different in construction than in a serial file:

	Operand 2	Connector	Location Name
READ	_____	PER	_____

Here the operator obtains the record at the location. If no record is there, the READ is not executed.

By definition, a random retained file is automatically updated with modifications at the same location.

In table 071, a number of actions now take place:

1. Quantity on hand is decremented (reduced) by the quantity sold.
2. Price extension (an arithmetic formula name) is moved to net price.
3. The record item is moved to the record item line.

After these actions, a DO INVOICE HEADER appears, which is once more the signal to call on a closed table, this time 074, for execution.

IBM

DECISION TABLE

SYSTEM NAME	PROCEDURE NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	INVOICE DETAIL	FRANCIS	1/4/62	

	TABLE ID / SER	TABLE NAME / OPER / OPERAND 1	TABLE TYPE / OP / OPERAND 2	OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
				FREQUENCY			
				01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32
**	071	INVOICE	OPEN-FIRST				
		DO	NEXT ITEM	X			
		SET LOC ADD	EQ ITEM FORMULA	X			
		DO	NEXT INVEN	X			
		SET QTY ON HAND	- QTY SOLD	X			
		MOVE PRICE EXT	TO NET PRICE	X			
		MOVE ITEM	TO ITEM LINE	X			
		DO	INVOICE HEADER	X			
		GO TO	071	X			
**	072	NEXT ITEM	CLOSED	1	99		
		ITEM CARD	IS EOF	Y	N		
		READ	ITEM CARD		X		
		DO	STOP	X			
**	073	NEXT INVEN	CLOSED	50	50		
		ITEM NO: ITEM	EQ ITEM NO MASTER	Y	N		
		SET LOC ADD	+ 1		X		
		READ	INVEN MASTER PER LOC ADD		X		
		DO	073		X		
**	074	INVOICE HEADER	CLOSED	90	10		
		CUST NO: ITEM	EQ CUST NO: HEADER	Y	N		
		WRITE	ITEM LINE	X			
		DO	TOTAL CALC		X		
		DO	PROCEDURE HEADER PREP		X		
		DO	074		X		

REMARKS:

Figure 8a. Decision Table, Example 8.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	FRANCIS	1/4/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
ITEM CARD	IN	SER-CARD	ITEM-1		4500	6000

SER NO	FIELD NAME	NUMERIC / ALPHA	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	TO ASSOC	SUBSCRIPTS	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE / REMARKS
1	CARD CODE	N	U	1				1						
2	CUST NO	N	U	5				1			1			
3	ITEM NO	N	U	6				1						
4	QTY SOLD	N	+	3				1						
5	UNIT PRICE	N	+	4.2				1						

Figure 8b. File Description 1, Example 8.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	FRANCIS	1/4/62	

FILE NAME	DIR TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVEN MASTER	RETAINED	RAN-DESK	MASTER		19,000	12,000

SER NO	FIELD NAME	NUMERIC / ALPHA	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	TO ASSOC	SUBSCRIPTS	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE / REMARKS
1	ITEM NO	N	U	6				1						
2	CLASS OF ITEM	A		3				1						
3	STOCK LOC.	A		6				1						
4	DESCRIPTION	A		25	✓			1						
5	QTY ON HAND	N	+	6				1						

Figure 8c. File Description 2, Example 8.

This table checks to see if the customer number on the item record is the same as the customer number on the header. If they are, then an item line is written (ITEM LINE is a record name on a serial file which is appropriately described on a Record Layout sheet and a File Description form). If they are not equal, then these actions are executed:

DO TOTAL CALC

DO PROCEDURE HEADER PREP

DO 074 (i.e., repeat this table)

The next time through the table the condition will be satisfied and the action WRITE ITEM LINE is executed.

Control then reverts to table 071 to continue the master cycle. The WRITE actions, designating output, are essentially the same in application whether for random or serial files except on random files a location must be specified.

The DO PROCEDURE HEADER PREP action brings up a procedure name for the first time. A procedure is one or more decision tables which together form a distinct and logical segment of an information processing system. The procedure name must start with an alphameric character and be unique. In this series of tables, for example, the procedure name is INVOICE DETAIL. The first table in a procedure must be marked as FIRST under TABLE TYPE as in table 071.

IBM

DECISION TABLE

SYSTEM NAME	PROCEDURE NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	INVOICE DETAIL	FRANCIS	1/4/62	

	TABLE ID / SER	TABLE NAME / OPER / OPERAND 1	TABLE TYPE / OP / OPERAND 2	FREQUENCY: OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
**	071	INVOICE	OPEN-FIRST				
		DO	NEXT ITEM	X			
		SET LOC ADD	EQ ITEM FORMULA	X			
		DO	NEXT INVEN	X			
		SET QTY ON HAND	- QTY SOLD	X			
		MOVE PRICE EXT	TO NET PRICE	X			
		MOVE ITEM	TO ITEM LINE	X			
		DO	INVOICE HEADER	X			
		GO TO	071	X			
**	072	NEXT ITEM	CLOSED	1	99		
		ITEM CARD	35 EOF	Y	N		
		READ	ITEM CARD		X		
		DO	STOP	X			
**	073	NEXT INVEN	CLOSED	50	50		
		ITEM NO: ITEM	EQ ITEM NO: MASTER	Y	N		
		SET LOC ADD	+ 1		X		
		READ	INVEN MASTER PER LOC ADD		X		
		DO	073		X		
**	074	INVOICE HEADER	CLOSED	90	10		
		CUST NO: ITEM	EQ CUST NO: HEADER	Y	N		
		WRITE	ITEM LINE	X			
		DO	TOTAL CALL		X		
		DO	PROCEDURE HEADER PREP		X		
		DO	074		X		

REMARKS:

Figure 8a. Decision Table, Example 8.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	FRANCIS	1/4/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
ITEM CARD	IN	SER-CARD	ITEM-1		4500	6000

SER NO	FIELD NAME	ALPHA/NUM	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	KEY	KEY NO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	CARD CODE	N	U	1				1						
2	CUST NO	N	U	5				1			1			
3	ITEM NO	N	U	6				1						
4	QTY SOLD	N	+	3				1						
5	UNIT PRICE	N	+	4.2				1						

Figure 8b. File Description 1, Example 8.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES
BILLING	FRANCIS	1/4/62	

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INVEN MASTER	RETAINED	RAN-DESK	MASTER		10,000	12,000

SER NO	FIELD NAME	ALPHA/NUM	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	KEY	KEY NO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	ITEM NO	N	U	6				1						
2	CLASS OF ITEM	A		3				1						
3	STOCK LOC	A		6				1						
4	DESCRIPTION	A		25	✓			1						
5	QTY ON HAND	N	+	6				1						

Figure 8c. File Description 2, Example 8.

QUESTIONS

1. What is the significance of EOF for files? Is it equally applicable to random and serial files?
2. How does the READ operator differ in construction between serial and random files?
3. What does DO PROCEDURE accomplish?
4. In what ways is an unconditional table different from other tables?
5. What is the relationship of table 071 to 072, 073 and 074? Trace the pattern of movement from one table to another.

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	
BILLING	INVOICE DETAIL		FRANCIS	1/4/62	PAGE OF PAGES

	TABLE ID SER	TABLE NAME OPER OPERAND 1	TABLE TYPE OP OPERAND 2	01	02
**	071	INVOICE	OPEN-FIRST		
		DO	NEXT ITEM	X	
		SET LOC ADD	EQ ITEM FORMULA	X	
		DO	NEXT INVEN	X	
		SET QTY ON HAND	- QTY SOLD	X	
		MOVE PRICE EXT	TO NET PRICE	X	
		MOVE ITEM	TO ITEM LINE	X	
		DO	INVOICE HEADER	X	
		GO TO	071	X	
*R	072	NEXT ITEM	CLOSED	1	99
		ITEM CARD	IS EOF	Y	N
		READ	ITEM CARD		X
		DO	STOP	X	
**	073	NEXT INVEN	CLOSED	50	50
		ITEM NO: ITEM	EQ ITEM NO MASTER	Y	N
		SET LOC ADD	+ 1		X
		READ	INVEN MASTER PER LOC ADD		X
		DO	073		X
**	074	INVOICE HEADER	CLOSED	90	10
		CUST NO. ITEM	EQ CUST NO: HEADER	Y	N
		WRITE	ITEM LINE	X	
		DO	TOTAL CALL		X
		DO	PROCEDURE		X
			HEADER PREP		
		DO	074		X

REMARKS:

Figure 8a. Decision Table, Example 8.

SYSTEM NAME	IBM FILE DESCRIPTION	ANALYST	DATE	
BILLING		FRANCIS	1/4/62	PAGE OF PAGES

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	PEAK
ITEM CARD	IN	SER-CARD	ITEM		4500	6000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	VAR	OCCURRENCES MIN	AVG	MAX				RELATIVE LOCATION BEGIN	END	VALUE/REMARKS
1	CARD CODE	N	U	1				1						
2	CUST NO	N	U	5				1			1			
3	ITEM NO	N	U	6				1						
4	QTY SOLD	N	+	3				1						
5	UNIT PRICE	N	+	4.2				1						

Figure 8b. File Description 1, Example 8.

SYSTEM NAME	IBM FILE DESCRIPTION	ANALYST	DATE	
BILLING		FRANCIS	1/4/62	PAGE OF PAGES

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	PEAK
INVEN MASTER	RETAINED	RAN - DESK	MASTER		10,000	12,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	VAR	OCCURRENCES MIN	AVG	MAX				RELATIVE LOCATION BEGIN	END	VALUE/REMARKS
1	ITEM NO	N	U	6				1						
2	CLASS OF ITEM	A		3				1						
3	STOCK LOC	A		6				1						
4	DESCRIPTION	A		25	✓			1						
5	QTY ON HAND	N	+	6				1						

Figure 8c. File Description 2, Example 8.

ANSWERS

1. EOF (END OF FILE) has meaning only for serial files. A random file, which acts like an endless conveyor belt, has no end of file.

2. For a serial file, the form is:

   READ file name

   For a random file, the form is:

   READ file name PER location name

   Specification of location is necessary for a random file.

3. The DO PROCEDURE, as used here, avoids specifying the procedure at this time. The DO PROCEDURE normally states that another series of tables will be called upon to carry out a separate procedure within the system.

4. An unconditional table has no conditions. The table is entered from another table and actions are carried out as displayed.

5. The looping pattern for the four tables in INVOICE DETAIL:

   TABLE 071 to 072

   072 to 071

   071 action

   071 to 073

   073 to 073, etc.

   073 to 071

   071 action

   071 action

   071 action

   071 to 074

   074 to 074, etc.

   074 to 071

   etc.

CHAPTER 3

Sample Systems Descriptions

Two systems are described using the Tabular Descriptive Language. The first, CREDIT RESERVE, is a complete description. It is written in the fullest detail possible using the Tabular Descriptive Language. The second system, POLICY FILE MAINT., is a partial system description. This example illustrates the hierarchical use of TDL by showing various levels of detail. The example may be taken to represent the status of a system description at a point in time prior to the completion of the documentation of the detailed analysis.

Example 1

SYSTEM:

Credit Reserve

PURPOSE:

The purpose of the Credit Reserve System is to establish adequate reserves for insurance issued to lendors (banks, credit unions, etc.) in the amount of and for the duration of loans which they have made.

SCOPE:

This system deals only with those policies for which the full premium is paid in advance.

DESCRIPTION:

There are two types of loans:

1) Declining Balance Loans - Loans for which there is periodic repayment during the term of the loan.

2) Level Balance Loans - Loans which are repayed in one lump sum at the end of the term of the loan.

In the case of a Declining loan, the liability of the insurance company decreases during the term of the loan. The liability of the insurance company remains fixed, however, during the term of a Level loan.

Reserves are established and maintained for each policy. Policies are issued to lendors. Since a lendor may make many different loans, which he wants insured under one policy, loans whose term and expiration date are the same must be summarized and amortized together. When file maintenance is done, this function will be performed and the output report "Monthly Reserve List" is produced. A policy may cover loans which are of both Level and Declining. On the output report, totals will be produced for each type of loan within a policy and for each policy.

Input "Activity" records will contain information on additional loans to be covered by a policy, and information on new policies.

SYSTEM NAME	ANALYST	DATE	
CREDIT RESERVE	M.K.W.	1/18/62	PAGE OF PAGES

IBM
FILE DESCRIPTION

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS	
					AVERAGE	PEAK
ACTIVITY FILE	IN	SER MAG TAPE	ACTIVITY	ACT	10,000	16,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	POLICY NO	N	U	7				1			1			
2	LOAN CODE	N	U	1				1			2			
3	EXP DATE	N	U	3				1			3			
4	TERM	N	U	2				1			4			
5	AMT INS CHGE	N	F	7.2				1						

REMARKS:

SYSTEM NAME	ANALYST	DATE	
CREDIT RESERVE	M.K.W.	1/18/62	PAGE OF PAGES

IBM
FILE DESCRIPTION

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
MASTER FILE	RETAINED	SER MAG TAPE	MASTER	MF	50,000	70,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	POLICY NO	N	U	7				1			1			
2	LOAN CODE	N	U	1				1			2			
3	EXP DATE	N	U	3				1			3			
4	TERM	N	U	2				1			4			
5	AMT INS	N	E	7.2				1						

REMARKS:

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

IBM 1403 SPACING CHART

6 Lines per inch

LINE DESCRIPTION

FIELD HEADINGS/WORD MARKS

SYSTEM: CREDIT RESERVE

ANALYST: M.K.W.

DATE: 1/18/62

PAGE: 3 OF

GLUE

FILE NAME: MONTHLY RES LIST

5 CONSTANT

1 LOAN DET-V

1 LEV TOT F

1 DEC TOT-F

1 POL. TOT-F

N CONSTANT

POLICY NO

LOAN CODE

EXP DATE

TERM

AMOUNT ORIGINAL

AMOUNT CURRENT

AMOUNT IN FORCE

AMOUNT RESERVE

VNNNNNN N

NNN

ZN

ZZZZZZZZZ

-ZZZZZZZZZ

ZZZZZZZZN

-ZZZZZZZZN

LEVEL

DECLINING

POLICY

SYSTEM NAME	ANALYST	DATE	
CREDIT RESERVE	M.K.W.	1/18/62	PAGE 1 OF 4 PAGES

IBM

FILE DESCRIPTION

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
MONTHLY RES LIST	OUT	SER PRINTED MAT	LOAN DET	LD	50,000	70,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	TO ASSOC	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EXP DATE	N	U	3				1				1	24	
2	TERM	N	U	2				1				1	29	
3	AMT ORIG	N	U	7.2				1				1	42	
4	AMT CUR	N	E	7.2				1				1	54	
5	AMT RES	N	E	7.2				1				1	77	
6	AMT IN FORCE	N	U	7.2				1				1	65	
7	POLICY NO	N	U	7				1				1	13	
8	LOAN CODE	N	U	1				1				1	15	

REMARKS:

SYSTEM NAME	ANALYST	DATE	
CREDIT RESERVE	M.K.W.	1/18/62	PAGE 2 OF 4 PAGES

IBM
FILE DESCRIPTION

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
MONTHLY RES LIST	OUT	SER PRINTED MAT	LEV TOT	LT	2000	2400

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	NO. ASSOC	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	AMT ORIG TOT	N	U	7.2				1				1	72	
2	AMT CUR TOT	N	E	7.2				1				1	54	
3	AMT RES TOT	N	E	7.2				1				1	77	
4	AMT IN FOR TOT	N	U	7.2				1				1	65	

REMARKS:

SYSTEM NAME	ANALYST	DATE	
CREDIT RESERVE	M.K.W.	1/18/62	PAGE 3 OF 4 PAGES

IBM

FILE DESCRIPTION

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS	
					AVERAGE	PEAK
MONTHLY RES LIST	OUT	SER. PRINTED MAT	DEC TOT	DT	2000	2400

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	AMT ORIG TOT	N	U	7.2				1				1	42	
2	AMT CUR TOT	N	E	7.2				1				1	54	
3	AMT RES TOT	N	E	7.2				1				1	77	
4	AMT IN FOR TOT	N	U	7.2				1				1	65	

REMARKS:

SYSTEM NAME	ANALYST	DATE	
CREDIT RESERVE	M.K.W.	1/18/62	PAGE 4 OF 4 PAGES

IBM
FILE DESCRIPTION

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
MONTHLY RES LIST	OUT	SER PRINTED MAT	POL TOT	PT	1000	1200

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	AMT ORIG TOT	N	U	7.2				1				1	42	
2	AMT CUR. TOT	N	E	7.2				1				1	54	
3	AMT RES TOT	N	E	7.2				1				1	77	
4	AMT IN FOR TOT	N	U	7.2				1				1	65	

REMARKS:

SYSTEM NAME	ANALYST	DATE	
CREDIT RESERVE	M.K.W.	1/18/62.	PAGE OF PAGES

IBM
FILE DESCRIPTION

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
INPUT HOLD	INTER	SER. CORE	HOLD REC	HR	1	1

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	EXP DATE	N	U	3				1						
2	TERM	N	U	2				1						
3	AMT ORIG	N	U	7.2				1						
4	AMT CUR	N	E	7.2				1						
5	AMT RES	N	E	7.2				1						
6	AMT IN FORCE	N	U	7.2				1						
7	POLICY NO	N	U	7				1						
8	LOAN CODE	N	U	1				1						
9	AMT INS	N	E	7.2				1						

REMARKS:

SYSTEM NAME	IBM	ANALYST	DATE	
CREDIT RESERVE	REFERENCE OPERAND DESCRIPTION	M.K.W.	1/18/62	PAGE OF PAGES

SER NO	NAME	TYPE	VALUE
1	LEVEL	CONS	1
2	DECLINING	CONS	0
3	CONTROL ACTIVITY	SE	(POLICY NO: ACT, LOAN CODE: ACT, EXP DATE: ACT, TERM: ACT)
4	CONTROL MF	SE	(POLICY NO: MF, LOAN CODE: MF, EXP DATE: MF, TERM: MF)
5	CONTROL HOLD	SE	(POLICY NO: HR, LOAN CODE: HR, EXP DATE: HR, TERM: HR)

SYSTEM NAME	PROCEDURE NAME
CREDIT RESERVE	

IBM®
DECISION TABLE

ANALYST	DATE	
M.K.W.	1/18/62.	PAGE OF PAGES

TABLE ID / SER	TABLE NAME / OPER	OPERAND 1	TABLE TYPE / OP	OPERAND 2	FREQUENCY OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2							
					01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 001	START		OPEN																																	
1	READ		ACTIVITY FILE		X																															
2	READ		MASTER FILE		X																															
3	GO TO		002		X																															
** 002	INIT HOLD REC		OPEN		5								95																							
1	CONTROL ACTIVITY		LR CONTROL MF		Y								N																							
2	SET HOLD REC		EQ			ACTIVITY								MASTER																							
3	MOVE AMT INS CHGE: ACT		TO			AMT CUR: HR								AMT ORIG: HR																							
4	MOVE AMT INS CHGE: ACT		TO AMT INS: HR		X																															
5	READ					ACTIVITY FILE								MASTER FILE																							
6	GO TO		003		X								X																							

REMARKS:

SYSTEM NAME	PROCEDURE NAME
CREDIT RESERVE	

IBM
DECISION TABLE

ANALYST	DATE	
M.K.W.	1/18/62	PAGE 1 OF 2 PAGES

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY			
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
			01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32
** 001	CREATE OUTPUT	OPEN	10	43	43	2
1	CONTROL HOLD	EQ CONTROL ACTIVITY	Y	N	N	N
2	LOAN CODE: HOLD REC	EQ LEVEL		N	Y	N
3	EXP DATE: HOLD REC	LE #CURRENT DATE#		N	N	Y
4	SET AMT CUR: HR	+AMT INS CHGE: ACT	X			
5	SET AMT INS: HR	+AMT INS CHGE: ACT	X			
6	READ	ACTIVITY FILE	X			
7	SET AMT IN FORCE: HR	EQ		DECL IN FOR DECL RES	LEV IN FOR LEV RES	DECL IN FOR DECL RES
8	SET AMT RES: HR	EQ				
9	SET LOAN DET	EQ HOLD REC		X	X	X
10	WRITE	LOAN DET		X	X	X
11	MOVE AMT ORIG: HR	+		AMT ORIG TOT: DT	AMT ORIG TOT: LT	AMT ORIG TOT: DT
12	MOVE AMT CUR: HR	+		AMT CUR TOT: DT	AMT CUR TOT: LT	AMT CUR TOT: DT
13	MOVE AMT IN FORCE: HR	+		AMT IN FOR TOT: DT	AMT IN FOR TOT: LT	AMT IN FOR TOT: DT
14	MOVE AMT RES: HR	+		AMT RES TOT: DT	AMT RES TOT: LT	AMT RES TOT: DT
15	SET AMT ORIG: HR	AMT CUR: HR		X	X	X
16	MOVE HOLD REC	TO MASTER		X	X	
17	WRITE	MASTER		X	X	
18	GO TO		003	004	004	004

REMARKS: DECL IN FOR AE AMT ORIG: HR * (EXP DATE: HR - #CURRENT DATE#) / TERM: HR
DECL RES AE AMT IN FORCE: HR * (EXP DATE: HR - #CURRENT DATE#) * .000375
LEV IN FOR AE AMT ORIG: HR + AMT CUR: HR
LEV RES AE AMT IN FORCE: HR * (EXP DATE: HR - #CURRENT DATE# - .5) * .00075

SYSTEM NAME	PROCEDURE NAME
CREDIT RESERVE	

IBM®

DECISION TABLE

ANALYST	DATE	
M.K.W.	1/18/62.	PAGE 1 OF 2 PAGES

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY																											
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2							
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 004	NEXT INPUT	OPEN	5								92								1								1							
1	CONTROL ACTIVITY	LR CONTROL MF	Y								N																							
2	MASTER FILE	IS EOF	N								N								Y								N							
3	ACTIVITY FILE	IS EOF	N								N								N								Y							
4	SET AMT ORIG : HR	EQ		ZERO								AMT INS: MF								ZERO								AMT INS: MF							
5	SET AMT CUR; HR	EQ		AMT INS CHGE: ACT								ZERO								AMT INS CHGE: ACT								ZERO							
6	SET AMT INS: HR	EQ AMT INS CHGE: ACT	X																X															
7	SET HOLD REC	EQ		ACTIVITY								MASTER								ACTIVITY								MASTER							
8	READ			ACTIVITY FILE								MASTER FILE								ACTIVITY FILE								MASTER FILE							
9	DO	005	X								X								X								X							
10	GO TO			003								003								003								003							

REMARKS:

SYSTEM NAME		ANALYST	DATE	
CREDIT RESERVE	DECISION TABLE CONTINUATION	M.K.W.	1/15/62	PAGE 2 OF 2 PAGES

TABLE ID			FREQUENCY			
SER	OP 05 OPERAND 2	OP OPERAND 2	OP OPERAND 2	OP OPERAND 2	OP OPERAND 2	OP OPERAND 2
** 003	2					
1	N					
2	Y					
3	Y					
4						
5						
6						
7	LEV IN FOR					
8	LEV RES					
9	X					
10	X					
11	AMT ORIG TOT:					
	LT					
12	AMT CUR TOT: LT					
13	AMT IN FOR					
	TOT: LT					
14	AMT RES TOT: LT					
15	X					
16						
17						
18	004					

REMARKS:

SYSTEM NAME	PROCEDURE NAME
CREDIT RESERVE	

IBM
DECISION TABLE

ANALYST	DATE	
M.K.W.	1/18/62	PAGE OF PAGES

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY																											
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2							
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 005	PREP TOTALS	CLOSED	10								10								80														
1	POLICY NO: HR	= POLICY NO: LD	Y								Y								N														
2	LOAN CODE: HR	= LOAN CODE: LD	Y								N																						
3	SET AMT ORIG TOT: PT	+									AMT ORIG TOT: LT								AMT ORIG TOT: PT														
4	SET AMT CUR TOT: PT	+									AMT CUR TOT: LT								AMT CUR TOT: PT														
5	SET AMT INTOR TOT: PT	+									AMT INTOR TOT: LT								AMT INTOR TOT: PT														
6	SET AMT RES TOT: PT	+									AMT RES TOT: LT								AMT RES TOT: PT														
7	WRITE	LEV TOT									X								X														
8	DO	CLEAR LT									X								X														
9	WRITE	DEL TOT																	X														
10	DO	CLEAR DC																	X														
11	WRITE	POL TOT																	X														
12	DO	CLEAR PT																	X														

REMARKS:

SYSTEM NAME		ANALYST	DATE	
CREDIT RESERVE	DECISION TABLE CONTINUATION	M.K.W.	1/18/62	PAGE 2 OF 2 PAGES

TABLE ID				FREQUENCY			
SER	OP OPERAND 2	OP OPERAND 2	OP OPERAND 2	OP OPERAND 2	OP OPERAND 2	OP OPERAND 2
** 004	1					
1						
2	Y					
3	Y					
4						
5						
6						
7						
8						
9						
10	STOP					

REMARKS:

SYSTEM NAME	PROCEDURE NAME
CREDIT RESERVE	

IBM

DECISION TABLE

ANALYST	DATE	PAGE OF PAGES
M.K.W.	1/18/62.	

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY																											
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2			
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 006	STOP	OPEN	50	50																												
1	Policy No: PT	EQ ZERO	Y	N																												
2	MOVE ZERO	TO POLICY NO: PT		X																												
3	DO	005		X																												
** 007	CLEAR LT	CLOSED																														
1	SET AMT ORIG TOT: LT	EQ ZERO	X																													
2	SET AMT CUR TOT: LT	EQ ZERO	X																													
3	SET AMT RES TOT: LT	EQ ZERO	X																													
4	SET AMT IN FOR TOT: LT	EQ ZERO	X																													
** 008	CLEAR DC	CLOSED																														
1	SET AMT ORIG TOT: DC	EQ ZERO	X																													
2	SET AMT CUR TOT: DC	EQ ZERO	X																													
3	SET AMT RES TOT: DC	EQ ZERO	X																													
4	SET AMT IN FOR TOT: DC	EQ ZERO	X																													
** 009	CLEAR PT	CLOSED																														
1	SET AMT ORIG TOT: PT	EQ ZERO	X																													
2	SET AMT CUR TOT: PT	EQ ZERO	X																													
3	SET AMT RES TOT: PT	EQ ZERO	X																													
4	SET AMT IN FOR TOT: PT	EQ ZERO	X																													

REMARKS:

Example 2

SYSTEM:

Policy File Maintenance

PURPOSE:

The purpose of this system is to perform all file maintenance functions required for the POLICY Master File.

SCOPE:

This system describes the file maintenance of the POLICY Master File for all policies issued by the company.

DESCRIPTION:

For any record in the POLICY Master File, file maintenance may be required for one or both of the following reasons:

1) There are input transactions affecting master records.
2) Data in the index record satisfies conditions which establish a need for certain processing to be performed, e.g., billing date, anniversary, nonpayment.

In addition to an updated POLICY Master File, the system will produce other outputs for other systems. All files are not described in this example.

As stated earlier, this is not a complete system description. The documentation shows the various levels of decisions, but only a single path is covered at the next lower level. After the Housekeeping procedure, Table 002 determines the various types of activity,

Table 032 shows the next level for decision for one type of activity (Automatic changes) by determining the class of change (Billing, Late Pay, Nonpayment, etc.), Table 050 shows the logic for one class (Nonpayment). Tables 082 and 083 define the Procedure to determine Cash Value called for by the DO PROCEDURE CASH VALUE action of Serial 7 Table 050.

Light shading is used where subsequent tables will further define the decision logic.

SYSTEM NAME	ANALYST	DATE	
POLICY FILE MAINT	L. J. T.	1/18/62	PAGE OF PAGES

IBM FILE DESCRIPTION

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS AVERAGE	NO OF RECORDS PEAK
POLICY MF	RETAINED	RAND DISK	POLICY INFO	PI	200,000	275,000

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	POLICY NO	N	U	6				1						
2	POLICY TYPE	N	U	1				1						
3	PLAN CODE	N	U	3				1						
4	INT RATE	N	U	1								1	1	PLAN CODE
5	MAT AGE	N	U	2								2	3	PLAN CODE
6	ISS AGE	N	U	2				1						
7	SEX	N	U	1				1						
8	NAME LAST INS	A		8				1						
9	LAST TRAN DATE	N	U	5				1						
10	DAY LAST TRAN	N	U	2								1	2	LAST TRAN DATE
11	MO LAST TRAN	N	U	2								3	4	LAST TRAN DATE
12	YR LAST TRAN	N	U	1								5	5	LAST TRAN DATE
13	AMT INS	N	U	6				1						
14	SUB STD CD	A		1				1						
15	REINS CD	A		1				1						
16	DATE ISSUE	N	U	6				1						
17	MO DATE ISS	N	U	2								3	4	DATE ISSUE
18	YR DATE ISS	N	U	2								5	6	DATE ISSUE
19	PAID TO DATE	N	U	4				1						
20	MO PAID TO	N	U	2								1	2	PAID TO DATE
21	YR PAID TO	N	U	2								3	4	PAID TO DATE
22	STATUS	N	U	2				1						
23	NON FORF STAT	N	U	1				1						
24	A	N		5				1						
								1						

REMARKS:

SYSTEM NAME	ANALYST	DATE	
POLICY FILE MAINT	L. J. T.	1/18/62	PAGE OF PAGES

IBM

FILE DESCRIPTION

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS	
					AVERAGE	PEAK
CASH VAL DIV RATE	RETAINED	RAND DISK	CASH VAL	CV	150	300

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH MAX	LENTH VAR	OCCURRENCES MIN	OCCURRENCES AVG	OCCURRENCES MAX	ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION BEGIN	RELATIVE LOCATION END	VALUE/REMARKS
1	PLAN CODE	N	U	3				1	A					
2	ISS AGE	N	U	2				1	A					
3	MAT AGE	N	U	2				1	A					
4	POL DUR	N	U	2.		99	99	99	B	A				
5	DIV FACTOR	N	U	0.5		99	99	99	B					
6	CASH VAL FACT	N	U	1.5		99	99	99	B					
7	PD CODE	N	U	1				1	A					

REMARKS:

SYSTEM NAME	IBM	ANALYST	DATE	
POLICY FILE MAINT	REFERENCE OPERAND DESCRIPTION	L. J. T.	1/18/62	PAGE OF PAGES

SER NO	NAME	TYPE	VALUE
1	EXCEPTION LIST	LIST	152, 271, 272, 276, 281-284, 371, 372, 376

SYSTEM NAME	PROCEDURE NAME
POLICY FILE MAINT	

IBM®
DECISION TABLE

ANALYST	DATE	PAGE OF PAGES
L. J. T.	1/18/62	

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY			
RER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
			01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32
** 001	START	OPEN				
1	DO	PROCEDURE HOUSEKEEPING	X			
2	GO TO	ACTIVITY DETERMINATION	X			
** 002	ACTIVITY DETERMINATION	OPEN	S	10	85	2
1	#CURRENT DATE#	EQ HIT 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	X
4	GO TO		AUTOMATIC CHANGES	EXTERNAL CHANGES	NEXT POLICY	PRIORITY

REMARKS:

SYSTEM NAME	PROCEDURE NAME
POLICY FILE MAINT	

IBM®
DECISION TABLE

ANALYST	DATE	
L.J.T.	1/18/62	PAGE OF PAGES

TABLE ID SER	TABLE NAME OPER OPERAND 1	TABLE TYPE OP OPERAND 2	FREQUENCY OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2							
			01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 032	AUTOMATIC CHANGES	OPEN	88	2	2	2	5	E																										
1	STATUS	IS BILLING	Y																															
2	STATUS	IS LATE PAY		Y																														
3	STATUS	IS NON PAYMENT			X																													
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	NON PAYMENT ROUTINE			X																													
9	GO TO	ANNIVERSARY ROUTINE				X																												
10	GO TO	CHANGE ROUTINE					X																											
11	GO TO	ERROR ROUTINE						X																										

REMARKS: NON PAYMENT LOG EXP PAID TO DATE <= (#CURRENT DATE#-59) AND PLAN CODE: PL /=
EXCEPTION LIST

SYSTEM NAME	PROCEDURE NAME
POLICY FILE MAINT	

IBM
DECISION TABLE

ANALYST	DATE	
L. J. T.	1/18/62	PAGE 1 OF 2 PAGES

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY			
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
			01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32
* 050	NON PAYMENT ROUTINE	OPEN	3	3	20	15
1	WORK POLICY DURATION	LR 1	Y	N	N	N
2	NON FORF STAT: PI	EQ AUTOMATIC PREMIUM LOAN		Y	N	N
3	PLAN CODE: PI				LR 100	EQ 100
4	YR DATE ISS: PI	LR 35				Y
5	SET WORK MO PD	EQ MO PAID TO		X	X	X
6	SET WORK YR PD	EQ YR PAID TO			X	X
7	DO	PROCEDURE CASH VALUE			X	X
8	DO	ATTAINED AGE CALC		X		
9	GO TO		LAPSE	MAKE AUTOMATIC LOAN	INFORMATION SHEET	PRL TERM

REMARKS:

SYSTEM NAME		ANALYST	DATE	
POLICY FILE MAINT	DECISION TABLE CONTINUATION	L. J. T.	1/18/62	PAGE 1 OF 2 PAGES

TABLE ID / SER	OP 05 OPERAND 2	OP 06 OPERAND 2	FREQUENCY OP 07 OPERAND 2	OP OPERAND 2	OP OPERAND 2	OP OPERAND 2
** 050	34	5	20			
1	N	N	N			
2	N	N	N			
3	EQ 100	EQ 101-159 161-259 261-376	EQ 160 260			
4	N					
5	X		X			
6	X		X			
7	X		X			
8						
9	APL TERM	LAPSE	INFORMATION SHEET			

REMARKS:

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAGE OF PAGES
POLICY FILE MAINT	CASH VALUE		L. J. T.	1/18/62	

TABLE ID / SER	TABLE NAME / OPER OPERAND 1	TABLE TYPE / OP OPERAND 2	FREQUENCY OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
			01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32
082	DATE DIFF CALC	CLOSED FIRST	40	45	10	5
1	WORK MO PD	VS: MO DATE	LR	GR	EQ	
		ISS: PI				
3	STATUS: PI	EQ FULL PD	N	N	N	Y
4	SET WORK MO	EQ	+12	WORK MO PD	00	
			-MO DATE ISS: PI	-MO DATE ISS: PI		
			+WORK MO PD			
5	SET WORK YR	EQ	-1	WORK YR PD	WORK YR PD	
			-YR DATE ISS: PI	-YR DATE ISS: PI	-YR DATE ISS: PI	
			+WORK YR PD			
6	SET WORK YR X	EQ WORK YR + 1	X	X		
7	READ	CASH VAL DIV RATE PER LOC CONTROL	X	X	X	
8	DO	083 PER ASSOC B	X	X	X	
9	SET AMT POL CSH		EQ INTERP	EQ INTERP	EQ WORK FACT 1 * AMT INS: PI	
10	DO	FULL PD ROUT				X

REMARKS: INTERP AE((WORK FACT 2 - WORK FACT 1) * WORK MO/12 + WORK FACT 1) * AMT INS: PI

SYSTEM NAME	PROCEDURE NAME
POLICY FILE MAINT	CASH VALUE

IBM DECISION TABLE

ANALYST	DATE	PAGE OF PAGES
L. J. T.	1/18/62	

TABLE ID / SER	TABLE NAME OPER	OPERAND 1	TABLE TYPE OP	OPERAND 2	FREQUENCY OP 01 OPERAND 2								OP 02 OPERAND 2								OP 03 OPERAND 2								OP 04 OPERAND 2							
					01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
** 083	GET FACTORS		CLOSED		10	90																												
1	WORK MO		EQ. ZERO		Y	N																												
2	WORK YR		EQ POL DUR		Y																													
3	WORK YR X		EQ POL DUR			Y																												
4	SET WORK FACT 1		EQ CASH VAL FACT		X																													
5	SET WORK FACT 2		EQ CASH VAL FACT			X																												

REMARKS:

CHAPTER 4

The Structure of the Language

Chapters 4, 5, and 6 are the reference portion of this manual. The structure of the Tabular Descriptive Language is described in this chapter. Chapter 5 describes the formation and use of operands; and Chapter 6 describes the operators used in the TDL system.

The Tabular Descriptive Language allows systems to be developed and described in a convenient manner. As a documentation device it provides a rigorous statement of the logic of an information processing system that can be easily understood by others. It is a tool for the systems analyst, and as such is capable of describing complex decision logic in a concise and unambiguous way. The systems analyst describes systems at various levels of detail; thus, the language must be hierarchical in nature, permitting both high-level and detailed descriptions of a system. The Tabular Descriptive Language has been designed to serve these functions.

Any language requires a vocabulary and a grammar. The vocabulary of the Tabular Descriptive Language is the operands and operators, together with the character set used to form them. The grammar is the rules used to form logical statements of conditions and actions. The construction (vocabulary and grammar) is described.

Formation of Operands

A set of characters is used to construct words. These words are in turn used to construct operand names, operators, indicate particular values, etc. The characters used in TDL are based on the character set of the IBM 1401 and 1410. The collating sequence is covered in Chapter 7.

Alphabetic	(A - Z)
Numeric	(0 - 9)
Punctuation	blank . ( ) ; , : ! ?
Other Special Characters	(+ $ * - / % = " ' ¢ # ¤ < ǂ ∆ @ > √ ǂ)

Zero is assumed to be an unsigned value; thus zero is neither positive nor negative for numeric comparison.

Names

Names are used to identify the various operands used in the system. With a few exceptions (see below), names are developed by the systems analyst to represent the elements upon which the operators work, (e. g., files, records, fields, expressions, lists, etc.). Names are constructed as follows:

The first character must be alphabetic; the characters following

the first can be alphabetic, numeric, any of these special characters (# $ % ¢ . @ ▲), or a blank. The blank is represented by an underscore (_) when its meaning might be confused. Names can be of any length convenient to the user. For example:

SALES $	GROSS_PAY
GROSS %	SALARY
YEAR TO DATE	INVENTORY AMOUNT 1961 AVG.

The same name can be used for fields of data appearing in different records or files by appending a record name to insure that no ambiguity exists. Certain other names may also be qualified as explained in Chapter 5. When a name is qualified at a higher level, the name is followed by a colon (:) and then the qualifying name. When names are unique, no such qualification is needed. Examples of qualified names:

GROSS AMOUNT : INVENTORY

GROSS AMOUNT : TRANSACTION

YEAR TO DATE AMOUNT : TOTAL LINE

NUMBER_OF_SEATS : INPUT SEGMENT

Special Names

In addition to names developed by the user to identify operands, another class of operands, which is essentially self-descriptive, is used.

These have special significance.

Literals - The user may state the value itself in a decision table rather than a name which represents the value. Numeric and alphameric literals are permitted. Numeric literals start with a numeral (0 - 9), a sign (+ , -) or a decimal point (.) and are thus distinguishable from names, which start with a letter. Alphameric literals must be enclosed in quotation marks to differentiate them from operand names. Examples of numeric and alphameric literals are:

+3.1416

'SEDAN'

14

'OUT_OF_STOCK'

.03125

-.03125

'123 - 24/649'

Connotative Terms - A word which connotes its value.

BLANK, ZERO, NINE, ALL 'any alphameric literal'

State Terms - A word which indicates a particular state of a variable.

For example:

POSITIVE, NEGATIVE, NUMERIC, ALPHABETIC

PRESENT, END OF FILE, STATUS

Descriptive Terms - A word which describes its value

CURRENT DATE #, # BRANCH OFFICE ADDRESS

None of these special names need to be described in the operand description portion of the system documentation. Their names alone are sufficient to identify them. These self-describing operands are defined fully in Chapter 5.

Operator Names

There are various operators used in forming statements in TDL. These names are preselected and have a defined meaning. Examples are:

EQ

NE

MOVE

SET

READ

WRITE

SORT

Formation of Statements

Two types of statements are made in decision tables -- condition statements and action statements. These statements are formed by the proper placement of operators (generally verbs) and operands (nouns).

The construction of a condition statement is:

	Operand 1	Operator	Operand 2
Examples:	YEAR TO DATE FICA	GE	150.00
	TOTAL	IS	POSITIVE
	ID : MASTER	EQ	ID : TRANSACTION
	DEDUCTIONS	EQ	BLANK
	STATUS	IS	NON PAYMENT

The construction of an action statement is:

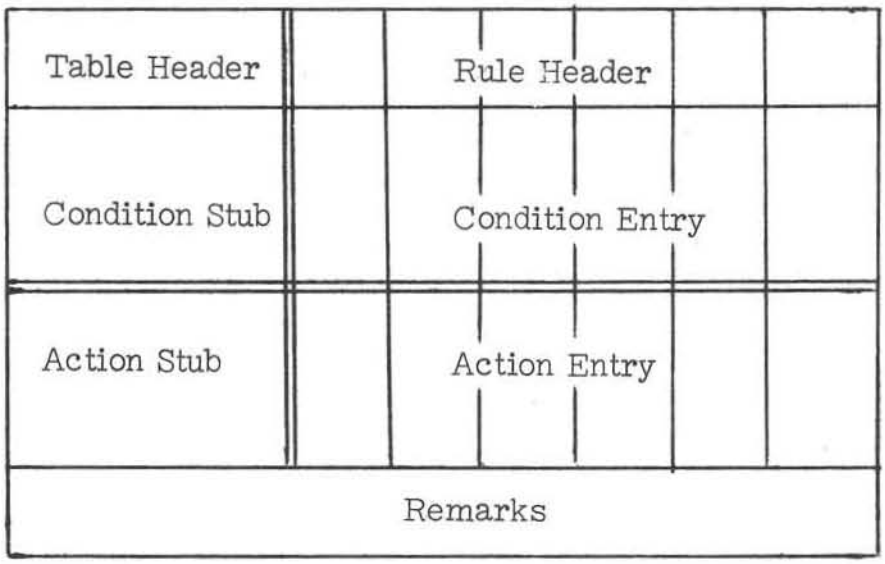
	Operator	Operand 1	Connector	Operand 2

or

	Operator	Operand 2
Examples:	MOVE AMOUNT TO TOTAL	
	SET YEAR TO DATE FICA + FICA	
	READ	MASTER
	DO	PROCEDURE GROSS TO NET

Formation of Decision Tables

A decision table is formed from sets of condition and action statements. The elements which make up a decision table are discussed with reference to this diagram.



A condition consists of a condition stub with one horizontally associated condition entry. An action consists of an action stub with one horizontally associated action entry. A decision rule consists of the conditions and actions associated vertically in an entry column in conjunction with the stub. A decision rule can be characterized as (where the underlined words are implied):

> If condition 1 is true and condition 2 is true and.....
> condition n is true then execute action 1 and then execute
> action 2 and then....execute action m.

A specific combination of conditions associated with a decision rule must be logically unique. Thus, decision rules can be examined in any order since only one rule can be satisfied for one pass through the table.

Figure 4-1 is a sample decision table form. The various elements will be briefly explained. There is a more detailed description in Chapter 7.

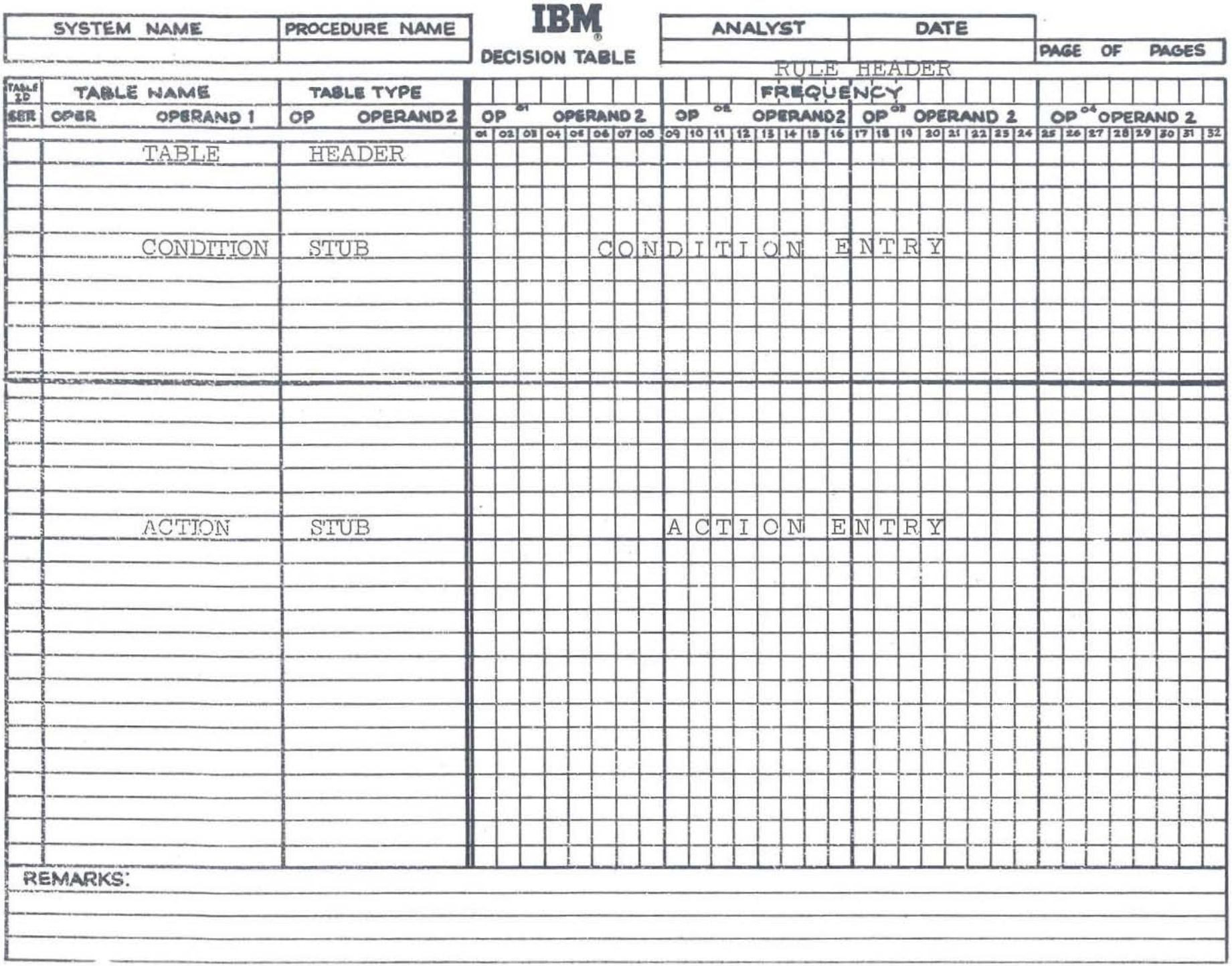

SYSTEM NAME
PROCEDURE NAME
IBM
DECISION TABLE
ANALYST
DATE
PAGE OF PAGES
TABLE ID
TABLE NAME
TABLE TYPE
RULE HEADER
FREQUENCY
OPER
OPERAND 1
OP
OPERAND 2
OP 01 OPERAND 2
OP 02 OPERAND 2
OP 03 OPERAND 2
OP 04 OPERAND 2
01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
TABLE HEADER
CONDITION STUB
CONDITION ENTRY
ACTION STUB
ACTION ENTRY
REMARKS:

Figure 4-1

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAGE OF PAGES

RULE HEADER

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY			
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
			01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32
* *	TABLE	HEADER				
	CONDITION	STUB	CONDITION ENTRY			
	ACTION	STUB		ACTION ENTRY		

REMARKS:

Figure 4-1

Table Header: The table is identified by both a name and an identification code. These may be referred to in the same or other decision tables. The header is also used to indicate whether the current table is "open" or "closed" (described later in this chapter).

More than a single table can be written on a page--each table being given a unique name and identification. Two asterisks are placed in the margin next to the table header to indicate the start of a new table.

Condition Stub: The condition stub may contain the entire condition statement (limited entry form), or it may contain just the first operand or the first operand and the operator (extended entry form).

Condition Entry: The condition entry may contain the second operand or it may contain the second operand and the operator (extended entry form). In other cases the condition entry may contain either Y, N, . or blank (limited entry form). Y is to be interpreted as an assertion of the condition statement made in the stub. N is interpreted as an assertion of the negative of the condition stated in the stub. For example, if the condition stub says A = B and there is an N in the associated condition entry, then we are examining the assertion that A does not equal B. The . or blank can be interpreted two ways. The basic interpretation is that this condition is not pertinent. The second interpretation is that this condition is impossible. In the last decision rule, all the condition entries may have periods or blanks, indicating an unconditional rule. The word ELSE is written in the rule header area. The actions in that particular rule are

executed if no other rule is satisfied.

Action Stub: The action stub can contain the entire statement, just the operator, the operator and first operand, or the operator, the first operand and the connector.

Action Entry: The action entry can contain the second operand, or the connector and the second operand, or the only operand, if it is a one-operand statement. In other cases it contains an X, which means "execute", or a blank or period (.) which means ignore or do not execute (limited entry form). Actions are executed in the order written. A GO TO action must be the last action for every rule, except in closed decision tables and the special decision table named STOP.

Rule Header: For each decision rule there is a header which indicates specifically the rule number and may show its frequency of occurrence. Rule number does not specify the order of consideration of the rules, but rather is simply a means for identifying each rule. Frequency indicates the relative number of times a rule will be executed. As such, it need not be exact, but suggests a possible ordering for most efficient solution of the decision table. Frequency is used to indicate the per cent of time that a particular set of conditions is likely to succeed. This means that the sum of all the frequencies should add up to 100.

Remarks: The remarks area provides an opportunity for making local definitions and descriptions; for example, arithmetic, logical and symbolic expressions can be defined when they apply only to the decision tables on that page. Comments to aid others in understanding the description are also useful.

Formation of System Description

An entire system is described by two types of operand description forms, standard printer spacing charts, and decision table forms. The File Description form describes files, records, and fields. The Reference Operand Description form describes expressions (arithmetic, symbolic and logical), value and reference lists, and constants. The printer chart specifies printed report formats. The Decision Table form (and a continuation form) specify the logic of the system. The amount of detail in preparing these descriptions depends upon the level of system specification desired.

Each type of record is defined on a separate File Description. The form and content of the File Description and Reference Operand sheets is described in Chapters 5 and 7; sample sheets are included at the end of this chapter (figures 4-2, 4-3).

A description of the system logic is given through the use of many decision tables, suitably connected to demonstrate logical flow. Two special decision tables are used (START, STOP) for beginning and ending the processing. In addition, there are two table formats used (limited entry tables and extended or mixed entry tables), and two major classes of tables (open, closed).

Special Decision Tables

Two special decision tables always appear in each system documented by the systems analyst. The first table to be executed in a system is labeled as the START table. The word "START" will be used for the table

name. The last table to be executed within a system is labeled as the STOP table. Both the START and STOP tables are OPEN tables. There is no action operator with the name "STOP." If the systems analyst wishes to stop all processing, he can do so by transferring control with a GO TO action to the STOP table.

The START and STOP tables can be in limited or extended entry form.

Limited and Extended Entry Tables

In a limited entry table all operators and operands for both condition and action statements are written in the stub area (left of the vertical double line). The condition entries are Y for yes, N for no, and period or blank for not pertinent. Action entries are X for execute, and period or blank for do not execute. Since the entry is limited to these items, the narrow rule columns are used to specify the combinations of conditions and actions. Thus, a maximum of 32 rules can appear on a Decision Table Sheet.

In extended entry tables, the condition entry can contain the second operand or the operator and the second operand. The action entry can contain the second operand or the connector and the second operand. In an extended entry table the wide rule areas (each composed of 8 narrow columns) are used. Thus, only four rules can appear on a Decision Table sheet. If more rules are needed a Decision Table Continuation sheet is used for the additional rules. The extended entry table form may include both limited and extended statements (in different rows). These are called mixed entry tables.

Open and Closed Tables

An OPEN table is one to which control is transferred by means of a GO TO action. A CLOSED table is entered only by a DO action which also controls the return to the originating table. An OPEN decision table may contain one or more DO action operators, but the last action of each rule should be a GO TO. A CLOSED decision table may contain one or more DO action operators, but generally will have no GO TO action operators. After the CLOSED decision table has been executed, control is automatically returned to the originating table, to the action statement following the DO action operator which called the CLOSED table. If a CLOSED table has a GO TO in any rule this will effectively destroy the return path for that rule. The table header specifies whether a table is CLOSED or OPEN.

The following chart summarizes this information on OPEN and CLOSED decision tables.

OPEN AND CLOSED TABLES

Table Type	May Contain	Entered By	Must Contain
Open	DO, GO TO	GO TO	* GO TO
Closed	DO, GO TO	DO	

* As last statement of each rule, except STOP Table.

This chapter has described the structure of the Tabular Descriptive Language, its vocabulary and grammar. The details of particular operand and operator specifications are given in the next two chapters along with additional information and illustrations of how they are used.

IBM

FILE DESCRIPTION

SYSTEM NAME	ANALYST	DATE	PAGE OF PAGES

FILE NAME	DIR. TYPE	MEDIUM	RECORD NAME	ABB	NO OF RECORDS	
					AVERAGE	PEAK

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH		OCCURRENCES			ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION		VALUE/REMARKS
				MAX	VAR	MIN	AVG	MAX				BEGIN	END	

REMARKS:

Figure 4-2

SYSTEM NAME		IBM REFERENCE OPERAND DESCRIPTION	ANALYST	DATE	PAGE OF PAGES

SER NO	NAME	TYPE	VALUE

Figure 4-3

CHAPTER 5

Operands

In order to document a logical solution to a problem, a means must be provided to identify and refer to data associated with that problem. In the Tabular Descriptive Language this means if provided by operands. An operand serves a function similar to that of a noun in an ordinary English language sentence. Operands, used in conjunction with operators (see Chapter 6), allow the various conditions and desired actions associated with the problem to be expressed in a simple, concise form.

Types of Operands: Operands in TDL are divided into two major classifications: Named Operands and Self-Describing Operands. Named Operands identify data associated with a particular information processing system. Their names do not indicate actual values, and therefore, they must be defined in the system in which they are used. Self-Describing Operands do not need to be described for each system and can be used universally, as actual values are specified.

Named Operands are classified in four major types: Information, Reference, Work, and System Segmenting Operands. The Operands used in the TDL System are as follows:

Named Operands

Information Operands

File
Record
Field

Reference Operands

Reference List Function
Value List
Arithmetic Expression
Logical Expression
Symbolic Expression
Constant

Work Operands

Work Record
Work Field

System Segmenting Operands

Procedure
Decision Table

Self-Describing Operands

Literals
Connotative Terms
State Terms
Descriptive Terms
Value List
Range

Operand Description Sheets: Three forms are used to describe operands: File Description, Report Description, and Reference Operand Description. The File Description form describes files, records, and fields within records. The Report Description is a

standard printer layout sheet that describes both fixed and variable information appearing on a printed report. The Reference Operand Description describes reference lists, value lists, all types of expressions, and constants. Work operands and System Segmenting Operands are defined by their use in decision tables.

Operand Names: The name of an operand is composed of a group of characters. The first character of a name must be alphabetic, but the remaining characters can be any of the letters (A - Z), the numerals (0 - 9), or the special characters (#, $, @, %, ¢, Δ, decimal point, or a blank). A name should be selected that is meaningful for the data described. The length of a name is not limited, but should be kept to a minimum without sacrificing readability or comprehension. Here are examples of operand names:

STOCK NO.

QUAN ON HAND

DISCOUNT %

DEPT NO.

T 127

There are special rules for naming work operands and self-describing operands.

INFORMATION OPERANDS

Files

A file is a uniquely named group of one or more types of records. It has two other fundamental properties that must be described: medium and directional type. Medium is the method of recording information within a file. Examples are magnetic tape, paper tape, punched card, printed report, disk, inquiry station, core memory, transmission line, etc. While the medium of a file must be described, no specific input/output or storage device is named. For example, the medium of a file can be indicated as punched card, but a particular card reading is not specified.

A file can be specified as having one of eight directional types: serial-input, serial-output, serial-intermediate, serial-retained, random-input, random-output, random-intermediate, or random-retained. The term "serial" indicates that only the next record of a file is available (e.g., tape or card). "Input" and "output" indicate whether the information comes into the system or is provided to other systems. "Intermediate" indicates that a file is used exclusively within a system, and is not retained for future use (a temporary file). Retained indicates that a file is available initially, but must be maintained for future operations either inside or outside the system (e.g., a master file).

The system description does not imply that the actual program must keep files physically separate, or that a file cannot be split into two or more files. For example, a systems analyst can indicate that the medium for a file is magnetic tape. The programmer can place this file, and one or more other similar files, on the same reel of magnetic tape; or he can split the file, placing some of its records on one reel of tape, and the remaining records on another reel of tape. This need not concern the systems analyst, nor will it invalidate the system description.

For each input, retained, or intermediate file it is assumed that there is a single area in memory for each file large enough to contain the largest record of a file. For output files, it is assumed that an area has been reserved in memory that is sufficient to store one of each of the record types at the same time.

Records

A record is a uniquely named group of fields which belong to a file. The systems analyst's description of a record should not be considered to be a record layout, for he is describing the content of the record, not its format. A record may be read into or written from a system by use of the READ and WRITE operators (see Chapter 6).

Each record in a file is described on a separate File Description form. Thus, the description of a file is a collection of record descriptions. Printed reports, which are considered to be files, are described on a Report Description form, in addition to a File Description form.

However, the order in which fields appear on a line of a Report Description form is the order in which they are to appear on the actual report. The record corresponds to one or more printed lines. For any record except lines on a printed report, the programmer can arrange the fields of a record in the most efficient order.

All fields should be explicitly noted in describing a record. Because only one record from each input file is available at any one time, a means is provided to save such records for subsequent use. Either a work record can be created or an intermediate file can be used. When identical field names and descriptions apply to two different records, the File Description of one record can refer to the other record which has already been defined.

Fields

A field is a named item of information that has a value and is part of a record. A sub-field is a part os a field. Fields and sub-fields have the same characteristics and are both normally referred to as fields.

Because a field has value, the relative position of its characters is significant. A field value can be all numeric, all alphabetic, all special characters, or a combination of these. A field in either an intermediate or output record can have a pre-assigned value. The systems analyst must indicate this pre-assigned value.

When describing a field, the systems analyst should indicate the maximum size it can attain and whether its length can vary. If a

numeric field contains a decimal part, the systems analyst should describe its maximum decimal length.

The sign of a pre-assigned numeric field should be indicated. Also, the sign possibilities should be indicated for all other numeric fields. The systems analyst should indicate that a numeric field will always be zero or greater (+), zero or less (-), either positive or negative (E for either), or will be unsigned (U).

There should be no inconsistency between a field and any sub-fields which are part of it. A sub-field can be defined as part of a numeric field that is signed and contains integral and decimal parts. If this is done, the sub-field must be a continuous part of the field, and must have the same sign possibilities as the field. If a sub-field contains integer and decimal parts, its decimal position must be consistent with the position of the decimal point in the field.

The same field can occur more than once in a signle record. Such a multiple-occurring field can be assigned a single name, permitting the systems analyst to describe, just once in a decision table, the processing for each separate occurrence of the field.

Association Groups: To facilitate the preparation of decision tables for multiple occurrence fields, all fields on the record having a logical relationship to each other can be grouped and identified by an Association Group code.

FIELD NAME	ALPHA NUMERIC	SIGN	LENTH		OCCURRENCES			ASSOC
			MAX	VAR	MIN	AVG	MAX	
PART NO	A		07				01	
DESCRIPTION	A		25				01	
LOCATION CODE	N	U	02		01	05	10	A
LOCATION NAME	A		15		01	05	10	A
CLASS CODE	N	U	01		01	01	03	B
QTY ON HAND	N	U	06		01	01	03	B
PRICE CODE	N	U	01		01	01	04	C
UNIT PRICE	N	+	04.2		01	01	04	C

In the preceding illustration there are three Association Group codes. CLASS CODE and QTY ON HAND are in the same group (B) because quantity on hand is always recorded for each class code.

Associated To: In addition, Association Group codes are utilized to indicate the logical hierarchy of the individual Association Groups to each other.

FIELD NAME	ALPHA NUMERIC	SIGN	LENTH		OCCURRENCES			ASSOC	ASSOC TO
			MAX	VAR	MIN	AVG	MAX		
PART NO	A		07				01		
DESCRIPTION	A		25				01		
LOCATION CODE	N	U	02		01	05	10	A	
LOCATION NAME	A		15		01	05	10	A	
CLASS CODE	N	U	01		01	01	03	B	A
QTY ON HAND	N	U	06		01	01	03	B	
PRICE CODE	N	U	01		01	01	04	C	
UNIT PRICE	N	+	04.2		01	01	04	C	

The group of fields in Association Group B are recorded a maximum 3 times for each group of fields in Association Group A. Similarly, the fields in Group A occur a maximum of 10 times for each record.

Sub-Fields: If a systems analyst wishes to refer to only part of a field, he must establish a sub-field and name it according to the rules for naming operands. For example, the six-digit, unsigned numeric field DATE can be divided into four sub-fields. One sub-field, defined as occupying the two left-most positions of the field, can be called DAY. A second sub-field MONTH can be defined as the third and fourth positions of the field. A third sub-field YEAR can be defined as the fifth and sixth positions of the field. A fourth sub-field MONTH AND YEAR DATE can be defined as the third through the sixth characters of the field.

One or more sub-fields can be specified for a field by indicating the positions of the field that form each sub-field. To do this, the positions of the field are numbered from left to right beginning with the number 1. To specify a sub-field, the positions of the field that form the sub-field are indicated in the File Description.

Even though one or more sub-fields are specified for a field, sub-fields for all parts of the field are not required. For example, only the first four positions of a ten-position field might be specified as a sub-field.

Sub-field names are unique within a record type. A sub-field can occur only once within a field. Thus, a field name need not be added to a sub-field name to qualify it. However, it may be necessary to append

a record name to a sub-field name when that sub-field occurs in more than one type of record.

Sub-fields can overlap one another within a field. A sub-field, however, cannot be part of one field and part of another field. A field can be an entire record and be divided into as many levels of sub-fields as needed.

REFERENCE OPERANDS

Reference operands have values defined through an operand description. These values will be present or will be computed at the time the operand is referred to in a decision table. Reference operands are described on a Reference Operand Description form.

Reference List Functions

A reference list is similar to a matrix or table of values. Each reference list function is related to one or two arguments. These arguments may be thought of as the vertical and horizontal axes of a tabular array. A reference list provides the systems engineer with a convenient form for describing a discrete set of values as functions of other values. An insurance rate table can be a reference list, with the function, rate, being dependent on the values of the two arguments, age and health.

Each reference list function value must be an explicit constant value. The argument values may be constants, ranges, or even names of value lists. Alphameric values are enclosed within quotation marks.

Numeric values do not require quotes. If a range is used as an argument, the upper and lower limits of the range are separated by a hyphen. If any part of a range (upper or lower limit) is alphameric both values must be enclosed within quotes. The collating sequence specified for TDL determines whether a particular value falls within the range. Reference lists cannot be modified by action within decision tables.

To refer to a specific entry within a reference list this form is used:

Reference List Function Name (argument 1, argument 2).

The reference list function name is the name of the function on the list. Argument 1 and argument 2 are designations of operands which may match the argement values of the list. Thus, a request for the rate for age 32 in the eastern section of the country can be specified as: RATE (32, "EAST"). If the rate to be determined is for the age and section of the country on an input record called APPLICATION, the form would be:

RATE (AGE: APPLICATION, SECTION OF COUNTRY: APPLICATION)

Value Lists

A value list is a group of two or more values or ranges of values that are identified by a single unique name. For example, a list of cities in which a company has warehouses can be called WAREHOUSE CITY. A field of an input record, with the name SHIPPING CITY, can be compared to the cities identified by the name WAREHOUSE CITY by the condition statement: SHIPPING CITY EQ WAREHOUSE CITY. If the specific value

of the input field is "Albuquerque", the value list named WAREHOUSE CITY is examined to determine whether it contains the value "Albuquerque". A value list can only be used in condition statements; it has no use in action statements.

The values or ranges of values appearing on a value list can be either numeric or alphameric. Each value or range of values is separated by a comma. The upper and lower limits of a range are separated by a hyphen. A numeric value can be written directly but an alphameric value is enclosed within quotation marks. For consistency, if one part of a range is alphameric, then both parts should be in quotes. When ranges of values appear on a value list, the collating sequence determines whether any specific value falls within one of the ranges specified.

An example of a value list is:

+4, -6.2, "RR" - "3A"

A condition statement comparing a field with this value list will result in the reply "yes" if the field value is +4, -6.2, or in the range from RR through 3A.

Arithmetic Expressions

An arithmetic expression is a named operand whose value is an arithmetic function of one or more operands. The value of an expression is always calculated at the time of its use in a decision table using current values of the operands forming the expression. Each time the expression name appears in a decision table, its value is recalculated. An

arithmetic expression must be given a unique name which will represent a numeric value. There is no practical limit to the number of operands and operators that may appear in an arithmetic expression. These numeric operands can be fields, constants, literals, reference list functions, descriptive and work fields. The five operators used in the expression are:

+ Addition

- Subtraction

* Multiplication

/ Division

** Exponentiation

In addition to these five basic operators used in arithmetic expressions, special mathematical functions like SIN, COS, TAN, SUM, NEG, POS, and ABS may be used. These are explained in Chapter 7.

Some examples of arithmetic expressions follow:

> C is the expression name for A + B, indicating that the sum of operands A and B produce the value represented by operand C.
>
> D represents A + B / C, indicating that the result of adding operand A to the quotient of B and C is the value represented by operand D.
>
> C represents A ** B, indicating that the value of A to the Bth power is represented by operand C.

An arithmetic expression is evaluated from left to right, using this hierarchy of the five basic operators: first--exponentiation, second--

multiplication and division, and third--addition and subtraction. The second example shows the quotient would be formed first, then the sum.

Parentheses can be used to establish the order and effect of any operator when the normal hierarchy is not desired. Operations within parentheses are always evaluated first. If there are parentheses within parentheses, the inner most operations are carried out first, then the next until effectively the parentheses are cleared and the normal hierarchy applies. An equal number of left and right parentheses must be used. As indicated above -- A + B / C would indicate that B is first divided by C and then added to A -- however, (A + B) / C would cause A to be added to B and the sum divided by C. All operands, operators, functions, and parentheses can be separated by one or more blanks to make an expression more readable.

If an arithmetic expression is only to be used for one decision table it is called a local arithmetic expression and may be described in the remarks portion of that table. Its name need only be unique for that decision table. Local arithmetic expressions have the same characteristics as other arithmetic expressions.

Logical Expressions

A Logical expression is an ordered combination of operands and operators that can be evaluated to be either true or false. Any operand, except the name of another expression (arithmetic, logical, or symbolic), can be used in a logical expression.

The operators that can be used in a logical expression are:

1. The three Logical Operators: AND, OR and NOT
2. The six Relational Operators:

   LR < lesser than

   EQ = equal to

   GR > greater than

   LE <= lesser than or equal to

   NE /= not equal to

   GE >= greater than or equal to
3. The five Arithmetic Operators: + - * / **
4. The two State Operators: IS and NT

Examples of logical expressions are:

MARITAL STATUS = 1 OR MARITAL STATUS = 2 OR

MARITAL STATUS = 4 - 7

PAY EXPERIENCE = "OK" AND BALANCE LE 10000.00

The logical operator OR is interpreted to mean either or both; that is, it is inclusive. The operator AND means that the two conditions joined by the AND must both be true for the logical expression to be true. The operator NOT means the negation of a condition. For example, NOT (A AND B) means if either A is not true or B is not true then the expression is satisfied. The expression NOT (A OR B) means both A and B must be not true for the expression to be satisfied.

Parentheses can be used effectively to indicate meaning and improve readability of logical expressions. If parentheses are used, an equal number of left and right parentheses are required. The hierarchy of the logical operators is: AND then OR then NOT. For example, A AND B OR C is interpreted as (A AND B) OR C.

Great caution should be exercised in the use of logical expressions since they can readily lead to errors. It is suggested that decision tables normally be used instead of such expressions. Even where a logical expression seems necessary or convenient, it may be helpful to work out the logic with a decision table. Local logical expressions may be used through definition in the remarks portion of a decision table.

Symbolic Expressions

A symbolic expression is a means for operating on values with other than arithmetic operators. The Reference Operand Description can be used to specify that ITEM IDENTIFICATION is the name for JOIN (LINE CODE, STOCK NUMBER), which means that the value of ITEM IDENTIFICATION is determined by joining the value of LINE CODE and STOCK NUMBER, in that order. Any number of fields, constants, literals, and reference list functions can be joined in a symbolic expression; the operands are written after the operator JOIN within parentheses in the order in which the values are to be put together; the operands are separated by commas. Local symbolic expressions may be described in the remarks portion of a decision table.

Constants

A constant is a uniquely named operand whose value cannot be changed by operation of the system. There is no limit to the size of a constant. Constants are described on a Reference Operand Description sheet. A constant may represent a signed or unsigned integer or decimal number or an alphameric value.

WORK OPERANDS

Work operands are used for the temporary storage of a value or a set of values (fields or records). Each work operand must have a different name but all of them are preceded by the letters WORK. No formal definition is required for either of the two types of work operands: work records and work fields.

Other than its name, a work operand assumes all the characteristics of any operand that is placed in it; these characteristics remain unchanged until a new operand is placed in it. Thus, each work operand always contains that which was placed in it most recently by an action in a decision table. For example, at one time a work operand may contain a ten-character numeric field; at a later time it may contain a fifteen-character alphameric field. If a systems engineer desires to delete the contents of a particular work operand, he may do so by placing blanks in that work operand.

Care should be exercised by the systems engineer in the use of work operands. He must be careful not to destroy data that will be required at a later time.

It is assumed that there will be sufficient space to contain the contents of all work records and work fields.

Work Records

A work record will contain as many fields as the record which is moved into it. When the contents of a record are moved into a work record, the work record will contain all the field values (and names) associated with the original record. These fields will have the same names as when they appeared in the original record. For example, if a record named STOCK STATUS is moved into a work record named WORK RECORD 1, each field in the work record gets the same name except it must be qualified by WORK RECORD 1. The systems engineer may refer to a field as: STOCK NUMBER: WORK RECORD 1. If later TRANSACTION is moved to WORK RECORD 1 then the fields in TRANSACTION can be referenced by their name qualified by WORK RECORD 1.

Work Fields

Each work field will assume the characteristics and the value of what is moved to it. These characteristics and value will persist until something else is placed in the work field. Each work field name will be unique and begin with the letters WORK. Work fields are not defined in any operand description.

SYSTEM SEGMENTING OPERANDS

Two types of operands are defined on Decision Table sheets and need not be defined elsewhere. These operand names are procedure names and decision table names or identifications. Their primary purpose is

to allow a system to be segmented into logical portions. These tables and procedures can then be referred to from other decision tables to provide sequence control.

Procedure Names

A procedure is a group of one or more decision tables that form a logical segment of an information processing system. The procedure need not be developed to refer to it in a decision table. This permits a system specification to be developed at a level above detail analysis. A payroll system could call for procedures to determine FICA deduction, Federal Withholding Tax, etc., assuming that the detailed procedures would be specified at a later date.

The first table of a procedure is indicated by writing FIRST in the Table Type section of the Decision Table form. A procedure may be open or closed depending upon how it will be used in the system. A procedure name must begin with a letter and must always be preceded by the word PROCEDURE, when it is referred to. No two procedure names can be identical. A procedure name should be indicative of the functions performed in the decision tables which comprise it.

Decision Table Names

A decision table is a logical association of conditions and actions. Since it usually takes many decision tables to make up a system, it is necessary to be able to refer from one decision table to another. This is done by using a decision table name or a decision table identification.

A decision table name is assigned using the rules for name construction and must be unique within a system. Decision table identifications are three characters in length with the first character numeric; the other two characters may be either numeric or alphabetic. Each decision table will be assigned a different identification.

SELF-DESCRIBING OPERANDS

Self-describing operands can be used universally without being specifically defined for a particular system. There are six types of self-describing operands: literals, connotative terms, state terms, descriptive terms, value lists, and ranges. These permit direct consideration of factor values instead of creating names and then referring to their values indirectly. They are specially useful in handling relationships or standards which are not apt to change.

Literals

A literal designates its own value rather than being the name of a value. There are two types of literals: numeric literals and alphameric literals.

A numeric literal may be a signed or unsigned fixed point number or a signed or unsigned floating point number. There are no restrictions on the length of a literal.

A fixed point number may consist of all integers, which may be preceded by a sign. Unsigned values are assumed positive if used in arithmetic. Examples of integers are:

+ 42

- 689

7254

A fixed point number may also consist of integer and decimal digits. It may be preceded by a plus or minus sign; unsigned, the value is assumed positive. A decimal point is shown between the integer and decimal part of the number. Examples are:

+ 42.6

- 68.253

0.567

.793

A floating point literal is constructed like a fixed point literal, except that it also contains an exponent, representing a power of ten. This exponent can be indicated as shown in these examples:

Floating Point Notation	Interpretation
14F + 7	14×10^{7}
11.7F - 3	11.7×10^{-3}
+ 14.269F21	$+ 14.269 \times 10^{21}$
- 11F - 15	$- 11 \times 10^{-15}$

An alphameric literal consists of any combination of numeric, alphabetic, or special characters. To differentiate it from a name or number it is enclosed within quotation marks. A blank is a value character, designated by an underline. A single blank between two sets of

characters can be left as a space. Quote marks in literals may lead to confusion so these should be handled by constant names and definitions. Examples of alphameric literals are:

'JANE DOE___SECRETARY'

'+ - * / **'

'369A4'

'+ 12.57'

Connotative Terms

A connotative term is one whose value is directly implied by its name. The following is a list of the acceptable connotative terms:

ZERO or ZEROS

BLANK or BLANKS

NINE or NINES

ALL 'any alphameric literal'

The first three connotative terms represent respectively one or more zeros, one or more blanks, and one or more nines. The connotative term ALL is used to represent repetitions of the same character or group of characters. The alphameric literal may be any character or group of characters; it is enclosed in quotes. ALL '7' would represent one or more 7's. ALL 'ABC' would represent one or more repetitions of ABC, such as ABCABCAB.

State Terms

A state term represents a characteristic or class of values. The

state terms that are included in TDL are:

POSITIVE or POS

NEGATIVE or NEG

NUMERIC or NUM

ALPHABETIC or ALPHA

STATUS

PRESENT

END OF FILE or EOF

The state term POSITIVE relates to all numeric values greater than zero, NEGATIVE to all numeric values lesser than zero. NUMERIC pertains to a field containing only numbers or blanks; ALPHABETIC fields may contain only letters or blanks.

The state term STATUS can be related only to a logical expression name. It indicates that the value of the associated logical expression is true or not true depending upon the operator used.

Descriptive Terms

A descriptive term indicates by means of its commonly understood meaning the value represented by that name. It is always enclosed within number symbols (#). Examples are:

CURRENT YEAR

MONTH END CLOSING DATE

Descriptive terms represent values which change. In this sense they differ from connotative terms whose values are always the same. A

descriptive term is not defined by the systems engineer since he can assume that its correct value will be provided by the programmer where required.

Value List

A value list consists of two or more literals or ranges. A value list can be used directly in a decision table as an operand. The same rules apply to noting values in a value list as when used in a Reference Operand Description. However, for convenience, each value may be put on a separate line without commas in between; similarly a range may be put on a single line.

An example:

13

15

17 - 27

42

Another example:

"MIA"

"SFO"

"LAG"

"LAX"

"JOL"

Range

A range is similar in use to a value list, except that it contains but

a single range of literal values. The upper and lower limits of the range are separated by a hyphen. Ranges are indicated as a distinct type of self-describing operand because, unlike most value lists, they can be written on a single line of a decision table. The rules for writing each individual literal (upper or lower limit) of a range operand are the same as those for writing any other type of literal except that if one boundary of the range is alphameric both must be enclosed in quotes. The range is considered inclusive at both ends.

CHAPTER 6

Decision Table Operations

Operands are the nouns of a language; operators are the verbs. Operators combined with operands permit dynamic procedures to be described by the systems analyst.

Operators are generally intended to work at a system analysis level rather than at a programming level. Thus, certain assumptions and statements will be made concerning automatic processes that may in fact be quite complex for a particular computer.

Only actions can change the values of operands. These actions are usually taken only if specified conditions are satisfied. Two major types of operations provide for these basic capabilities: testing a set of conditions and executing resultant actions.

Conditions -- to interrogate values and states

Actions -- to change values, provide input/output and control sequence

Because pyramiding of all conditions within a single decision table makes a system description unwieldy, a means is provided to describe systems using many tables. Action operators are available to control the sequence of table execution.

Conditions

Conditions can be determined to be either true or false. Conditions are the means for asking questions; as such they can be used to control

the actions which are to be executed.

A condition is true or false, depending on its operator and the value of its operands. Each individual condition should be stated as simply as possible to avoid inaccuracy and difficulty of interpretation; complex interactions among conditions are shown through position in a decision table. Those within the same rule bear a logical AND relationship to one another. Individual rules express an OR relationship.

There are two major categories of conditions: Value Relational and State Relational. Each of these is discussed under the headings:

OPERATORS

OPERANDS

COMPARISON RULES

USAGE

MECHANICS

Value Relational Conditions

All value relational conditions are constructed as follows:

form:	Operand 1	Condition Operator	Operand 2
example:	QUANTITY ON HAND	LE	1000

read as: IF QUANTITY ON HAND is less than or equal to 1000

OPERATORS

The following operators may be used in a value relational condition:

Symbolic	Mnemonic	Read As
=	EQ	is EQual to
>	GR	is GreateR than
>=	GE	is Greater than or Equal to
<	LR	is LesseR than
<=	LE	is Lesser than or Equal to
/=	NE	is Not Equal to

An operator can be written in symbolic or mnemonic representation.

OPERANDS

These operands can be used either as operand 1 or operand 2 of a value relational condition:

Field Name
Reference List Function Name
Value List Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Connotative Term
Descriptive Term
Value List
Range

COMPARISON RULES

Since an operand may represent either a numeric or an alphameric value of different lengths it is necessary to state what happens under various circumstances. Value relational conditions usually require that both operands be numeric or both be alphameric.

Numeric: If both operands are numeric, comparison follows normal algebraic rules: (1) Decimal points are aligned, (2) High-order zeros are added to the shorter integer operand, (3) Low-order zeros are added to the shorter decimal operand, (4) All negative values are less than zero, (5) All positive values are greater than zero, (6) For negative values larger absolute magnitude is considered smaller value, (7) Comparisons are permitted between fixed and floating point numbers by considering floating point number to be converted to fixed point notation, (8) Integers are treated as decimal numbers with the decimal point to the right of the least significant position.

Alphameric: If both operands are alphameric, the comparison follows usual practice: (1) Line up left-most positions, (2) Add low order blanks to shorter operand (3) Comparison is performed position by position from left to right, using character collating sequence, (4) The first unequal character position determines the result of the comparison.

Mixed: Comparisons between numeric and alphameric operands are permitted and are interpreted on this basis: (1) The rules of

alphameric comparison apply. The numeric value is considered an alphameric operand, (2) The sign and decimal point (if any) of the numeric value are not considered in the comparison, (3) Low order blanks are added to the shorter operand, (4) Floating point numbers are converted to decimal notation before comparison.

USAGE

Special combination rules apply to the use of the operands and operators in value relational conditions.

Field Name: A Field Name can be used with any value relational operator and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Connotative Term
Descriptive Term

A Field Name may be used only with the EQ and NE operators and these operands:

Value List Name
Value List
Range

Reference List Function Name: A Reference List Function Name can be used with any other value relational operator and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Connotative Term
Descriptive Term

It can be used only with EQ and NE operators and these operands:

Value List Name
Value List
Range

If the arguments of a Reference List do not appear in the reference list the comparison will fail regardless of what value it is compared with. For example,

PAY ('CARPENTER', 7) LE 4.50

and PAY ('CARPENTER', 7) GR 4.50

will both fail if the Reference List skips from ('CARPENTER', 6) to ('CARPENTER', 8).

Value List Name: A Value List Name can be used only with the EQ and NE operators and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Connotative Term
Descriptive Term

The EQ comparison is considered true if any item in the Value List matches the value of the other operand. The NE comparison is considered true if no item in the Value List matches the value of the other operand.

Arithmetic Expression Name: An Arithmetic Expression Name can be used with any value relational operator and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Connotative Term
Descriptive Term

It can be used only with EQ and NE operators and these operands:

Value List Name
Value List
Range

The value of the arithmetic expression is determined dynamically at the point of usage and redetermined each time it is used. Since an arithmetic expression always develops a numeric value, this should be considered when selecting the other operand. If it is alphameric, the results may be in error.

Symbolic Expression Name: A Symbolic Expression Name can be used with any value relational operator and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Connotative Term
Descriptive Term

It can be used only with EQ and NE and these operands:

Value List Name
Value List
Range

The value of a symbolic expression is determined dynamically at the point at which it is used and redetermined each time it is used. Since the value can be either an unsigned integer or an alphameric value, the comparison rules are determined by the other operand. When compared with another symbolic expression, the rules for alphameric comparison will apply.

Constant Name: A Constant Name can be used with any value relational operator and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Work Field Name
Descriptive Term

It can be used only with EQ and NE with a Value List Name.

Work Field Name: A Work Field Name can be used with any value relational operator and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Connotative Term
Descriptive Term

It can be used only with EQ and NE and these operands:

Value List Name
Value List
Range

A Work Field can only take on value that has been assigned to it; therefore, a comparison with a Work Field Name which has never been assigned any value will be treated as an undefined comparison and will fail every type of test.

Literal: A Literal can be used with any value relational operator and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Work Field Name
Descriptive Term

It can be used only with EQ and NE with a Value List Name. The value type is interpreted as written. An unsigned integer is handled in accordance with the other operand's type.

Connotative Term: A Connotative Term can be used with any value relational operator and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name (unsigned integer only)
Symbolic Expression Name
Work Field Name
Descriptive Term

It can be used only with EQ and NE with a Value List Name. The length and value type are interpreted by the other operand.

Descriptive Term: A Descriptive Term can be used with any value relational operator and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Connotative Term
Descriptive Term

It can be used only with EQ and NE and these operands:

Value List Name
Value List
Range

A Descriptive Term takes on the form of the operand to which it is compared. For example,

MONTHLY CLOSING DATE # EQ 12/31/61

or # MONTHLY CLOSING DATE # EQ DEC 31, 1961

are both acceptable.

Value List and Range: A Value List or Range can be used directly in the decision table only with the EQ and NE operators and these operands:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Work Field Name
Descriptive Term

The EQ comparison for a Value List is true if any item in the Value List matches the value of the other operand. The NE comparison is true if no item in the Value List matches the value of the other operand.

The EQ comparison for a Range is true if the other operand is greater than or equal to the lower bound of the Range while less than or equal to the upper bound. Unsigned integer ranges are interpreted as alphameric or numeric, depending upon the other operand value type.

MECHANICS

A value relational condition can be expressed in either limited or extended entry form. These are the permitted arrangements:

STUB		ENTRY
Operand 1	EQ, = Operand 2 NE, /= GR, >= LR, < GE, >= LE, <=	Y, N, . or blank
Operand 1	EQ, = NE, /= GR, >= LR, < GE, >= LE, <=	Operand 2, . or blank
Operand 1		EQ, = Operand 2, . or blank NE, /= GR, >= LR, < GE, >= LE, <=
Operand 1	VS Operand 2	EQ, = , . or blank NE, /= GR, >= LR, < GE, >= LE, <=

The first form shown is limited entry with Y meaning yes (or true), N meaning no (or false) and ". " or blank meaning not pertinent or impossible. The second form is extended entry using a common value relational operator for all rules expressed in the entry columns. The third form is extended entry with different relational operators for each rule. In both of these ". " or blank means not pertinent. The fourth form is a special type of limited entry where VS (interpreted as Versus) is used to represent the specific value relational operator

in each rule. Each rule is read as though the operator in that rule were substituted for the VS.

State Relational Conditions

All state relational conditions are constructed as follows:

form:	Operand 1	Condition operator	Operand 2
example:	AVAILABLE BALANCE	IS	POSITIVE
read as:	IF AVAILABLE BALANCE IS POSITIVE		

OPERATORS

Either of two state relational operators can be used in a state relational condition:

Mnemonic	Read as
IS	IS
NT	is NoT

There are no symbolic representations for these operators. The mnemonic representatives should be used.

OPERANDS

In a state relational condition, operands 1 and 2 may be interchanged, but readability is often impaired. In general, operand 2 should be the state term, however, operand 1 or operand 2 can be any of these operands:

File Name
Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Work Field Name

When Operand 2 is one of these state terms:

POSITIVE
NEGATIVE
NUMERIC
ALPHABETIC
PRESENT
END OF FILE or EOF

Operand 1 can be the State Term STATUS if Operand 2 is a Logical Expression Name.

COMPARISON RULES

Comparison rules are determined by the particular State Term used in the state relational condition. Generally, an incorrect match of value class (e.g., alphameric value against POSITIVE) results in failing the test.

USAGE

The rules for usage of the various operands are discussed for each permissible State Term. Multiple occurring Field Names may be used wherever a Field Name can be used except that they must only appear in a closed decision table under association control from a preceding DO statement.

Positive: The State Term POSITIVE can only be used with numeric valued operands. An operand is POSITIVE if its algebraic value is greater than zero. The following operands can be used when they represent a numeric value:

Field Name (per Operand Description)

Reference List Function Name (per Operand Description)

Arithmetic Expression Name

Symbolic Expression Name (if composed only of numeric valued operands)

Work Field Name (per description or characteristics of previous field value which entered the work field)

All unsigned numeric values except zero will be considered POSITIVE. If a Reference List Function Name is not defined for the arguments indicated, the comparison will fail.

Negative: The State Term NEGATIVE can only be used with numeric valued operands. An operand is NEGATIVE if its algebraic value is less than zero. The following operands can be used when they represent a numeric value:

Field Name (per Operand Description)

Reference List Function Name (per Operand Description)

Arithmetic Expression Name

Symbolic Expression Name (if composed only of numeric valued operands)

Work Field Name (per description or characteristics of previous field value which entered the work field)

Unsigned numeric values are considered positive and will fail a test for NEGATIVE. If a Reference List Function Name is not defined for the arguments indicated, the comparison will fail.

Numeric: The State Term NUMERIC can be used with operands defined as numeric or alphameric. An operand is NUMERIC if all of its characters are from the set 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, or blank. The following operands can be used:

Field Name
Reference List Function Name
Symbolic Expression Name
Work Field Name

If a Reference List Function Name is not defined for the arguments indicated, the comparison will fail.

Alphabetic: The State Term ALPHABETIC can only be used with alphamerically defined operands. An operand is ALPHABETIC if all of its characters are letters (A, B, C, . . . , X, Y, Z) or blank. The following operands can be used where they represent alphameric values.

Field Name (per Operand Description)

Reference List Function Name (per Operand Description)

Symbolic Expression Name (if composed only of alphameric valued operands)

Work Field Name (per description or characteristics of previous field values which entered the work field)

Status: The State Term STATUS can be used with a Logical Expression Name only. STATUS is written in the Operand 1 position.

The logical expression name is written as Operand 2. For example:

STATUS IS MARRIED

The comparison is satisfied if the value of the logical expression MARRIED is true. It is not satisfied if the value of the logical expression is false.

Present: The State Term PRESENT can be used with a Reference List Function Name only. If the function is defined for the arguments given, then the function is considered PRESENT. If the function is not defined for these arguments, the function is not considered PRESENT. PRESENT can be used to prevent missing factors in a reference list from destroying a legitimate calculation; it is also useful to check existence of special circumstances. For example,

SPECIAL FARE ('BOSTON', 'NEWARK') IS PRESENT

might be used to determine if a regular mileage-based fare should be used or if there is a special fare agreement.

End of File: The State Term END OF FILE can be used with a File Name only. File name IS END OF FILE is satisfied when there are no more records available on the file. If another record is available, the test will fail. This State condition can be used for serial files only. Also, END OF FILE (can be abbreviated EOF) is only pertinent to input, retained and intermediate files when they are being treated as input to the system. For example,

TRANSACTION IS EOF

will only be satisfied when there are no more records in the TRANSACTION file.

MECHANICS

A state relational condition can be expressed in either limited or extended entry form. These are the permitted arrangements:

STUB		ENTRY
Operand 1	IS POSITIVE NT NEGATIVE NUMERIC ALPHABETIC PRESENT END OF FILE EOF	Y, N, . or blank
STATUS	IS Operand 2 NT (Logical Expression Name)	Y, N, . or blank
Operand 1	IS NT	POSITIVE ,. or blank NEGATIVE NUMERIC ALPHABETIC PRESENT END OF FILE EOF
STATUS	IS NT	Operand 2 ,. or blank (Logical Expression Name)
Operand 1		IS POSITIVE ,. or blank NT NEGATIVE NUMERIC ALPHABETIC PRESENT END OF FILE EOF
STATUS		IS Operand 2 ,. or blank NT (Logical Expression Name)

The first two forms shown are limited entry. The remaining forms are extended entry.

Actions

Actions change values, specify input/output, and control the sequence within a set of decision tables. Within a single table, each action is executed in the order written, and must be logically completed before a subsequent action is initiated.

Actions are stated as commands, and are indicated by their position in a decision table. The three major categories of actions are: assignment, sequence control, and input/output. These are discussed under the headings:

OPERATORS

OPERANDS

SPECIAL RULES

USAGE

MECHANICS

Assignment Actions

The construction of an assignment action is:

form:	Action operator	Operand 1	Action Connector	Operand 2
example:	MOVE	EMPLOYEE NAME	TO	CHECK NAME

read as: Assign the value of employee name to the field called check name.

OPERATORS

The following operators can be used in assignment actions:

MOVE

SET

With MOVE these action connectors can be used:

Symbolic	Mnemonic	Read as
→	TO	inTO
+	PL	PLus operand 2 into operand 2
-	SU	SUbtracted from operand 2 into operand 2
*	MU	MUltiplied by operand 2 into operand 2
/	DI	DIvided into operand 2 into operand 2

With MOVE, Operand 1 specifies the sending value and Operand 2 specifies the receiving area.

With SET these action connectors can be used:

Symbolic	Mnemonic	Read as
←	EQ	EQual to
+	PL	PLus operand 2 into operand 1
-	SU	SUbtract operand 2 into operand 1
*	MU	MUltiplied by operand 2 into operand 1
/	DI	DIvided by operand 2 into operand 1

With SET, Operand 1 specifies the receiving area and Operand 2 specifies the sending value. With both MOVE and SET, either the

symbolic or mnemonic representations of the action connectors can be used.

OPERANDS

These operands can be used as the sending operand (Operand 1 with MOVE or Operand 2 with SET).

Record Name
Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Record Name
Work Field Name
Literal
Connotative Term
Descriptive Term
Special Arithmetic Expression

These operands can be used as the receiving operand (Operand 2 with MOVE and Operand 1 with SET).

Record Name
Field Name
Work Record Name
Work Field Name

SPECIAL RULES

Since an operand may represent either numeric or alphameric values of different lengths, it is necessary to state what happens under various circumstances. Usually, both operands are numeric or both alphabetic.

Numeric: If both operands are numeric, the assignment of value is controlled by the receiving operand. When both operands have the same decimal and integer length, the function of the MOVE and SET operators is apparent. If the sending and receiving operands have different decimal and integer lengths, the sending field is rounded from the right or zeros are added to the right until the decimal lengths correspond, and is truncated from the left or zeros are added to the left until the integer lengths correspond.

If the sending operand is in floating-point notation, but the receiving operand is fixed point, the sending operand is first converted to fixed point. When both operands are in floating-point notation, the floating point format is retained. If the number of positions in the exponent and/or mantissa are different, the rules for length stated for fixed point apply. If a fixed point operand is moved into a floating point operand, it is modified to retain the most significant positions and is then converted to floating point.

On arithmetic MOVE or SET actions, all sending operands are operated upon in fixed or floating as per receiving operand, and the result after the arithmetic operations is adjusted to fit the length and decimal restrictions of the receiving operand.

Alphameric: If both operands are alphameric, the assignment of value is controlled by the receiving operand. When operands are of the same length, the function of the MOVE and SET operators is apparent.

If the sending operand is shorter, it is left justified when moved into the receiving operand, and low-order blanks are added to fill the receiving operand. If the sending operand is longer before the MOVE, it is truncated from the right until its length is the same as the receiving operand.

Mixed: Alphameric valued operands can be moved into numeric valued operands only if the actual characters moved are all numeric. In this case, the sending operand is treated as an unsigned integer and the rules for numeric assignment are followed. Numeric valued operands which are unsigned integers can be moved into alphameric valued operands and the rules for alphameric assignment are followed.

When records are assigned to other records, only fields with corresponding names are moved. Each field moved follows the rules that apply for individual field assignment.

If the receiving operand is a report Field Name that is to be edited, it is assigned on a non-edited basis. The editing occurs when the WRITE command is executed.

USAGE

A number of special rules apply to the selection of specific operands to be used with various connectors. Therefore, specific usage is discussed under four categories: Individual value assignment, corresponding value assignment, two-operand arithmetic assignment, and

multiple-factor arithmetic assignment.

Individual Value Assignment: This is the basic operation which assigns one value to a Field Name or Work Field Name. The sending operand can be:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Connotative Term
Descriptive Term

The four forms are:

MOVE sending operand TO receiving operand
MOVE sending operand ⟶ receiving operand
SET receiving operand EQ sending operand
SET receiving operand ⟵ sending operand

If a Reference List Function Name is undefined for the argument values specified, then no assignment takes place; that is, the receiving operand is not changed.

Assignment to a Work Field Name makes the work field take on all the properties of the sending field, except its name.

The sending operand is never affected by the assignment.

Corresponding Value Assignment: This operation assigns values whose Field Names correspond from one record to another. The sending operand and the receiving operand must be a Record Name.

The four forms are:

MOVE sending operand TO receiving operand
MOVE sending operand ⟶ receiving operand
SET receiving operand EQ sending operand
SET receiving operand ⟵ sending operand

The only fields assigned are those whose names are the same in both records. All rules for individual value assignment apply. The sending record is not affected by the assignment.

Note: If the receiving operand is a Work Record, all fields in the sending record are assigned to the work record and their original names are retained. The original names can be used, but they must be qualified by the Work Record Name.

Two-Operand Arithmetic Assignment: This operation provides for two-operand arithmetic when a specific receiving operand is to be increased, decreased, etc. The sending operand can be:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Descriptive Term

The receiving operand can be:

Field Name
Work Field Name

MOVE A + B means A plus B into B
MOVE A - B means B minus A into B
MOVE A * B means A times B into B
MOVE A / B means B divided by A into B

SET A + B means A plus B into A
SET A - B means A minus B into A
SET A * B means A times B into A
SET A / B means A divided by B into A

Multiple Factor Arithmetic Assignment: This operation provides greater facility in arithmetic operations without using arithmetic

expression names. The receiving operand must be a Field Name or a Work Field Name. The sending operand can only be a Special Arithmetic Expression.

The forms are:

MOVE sending operand TO receiving operand
MOVE sending operand → receiving operand
SET receiving operand EQ sending operand
SET receiving operand ← sending operand

The factors in a special arithmetic expression can be:

Field Name
Reference List Function Name
Arithmetic Expression Name
Symbolic Expression Name
Constant Name
Work Field Name
Literal
Descriptive Term

The receiving operand can be:

Field Name
Work Field Name

Any reasonable number of factors can be used with special arithmetic connectors between each of the factors. The rules are:

(1) Connector is + (plus)
(2) Connector is - (minus)
(3) Connector is * (times)
(4) Connector is / (divided by)--only a single division permitted
(5) Connector is ** (exponentiation)--only a single exponentiation permitted

Parentheses cannot be used. The computation takes place from top to bottom except that exponentiation always takes place first, then multiplication and division, then addition and subtraction.

MECHANICS

An assignment action can be expressed in either limited or extended entry form. The following arrangements are permitted for two-operand arithmetic.

STUB				ENTRY
MOVE	Operand 1	TO → + - * /	Operand 2	X , . or blank
SET	Operand 1	EQ ← + - * /	Operand 2	X , . or blank
MOVE	Operand 1	TO → + - * /		Operand 2 , . or blank
SET	Operand 1	EQ ← + - * /		Operand 2 , . or blank
MOVE	Operand 1			TO Operand 2 , . or blank → + - * /
SET	Operand 1			EQ Operand 2 , . or blank ← + - * /

The first two forms shown are limited entry, the others are extended entry.

With multiple factor arithmetic, the sending operand is more complex, but the assignment operations are limited to a basic MOVE sending operand TO receiving operand, or SET receiving operand EQ sending operand. These can also appear in limited or extended entry form as shown.

STUB				ENTRY
MOVE + - ⋮ ±	factor a factor b factor c factor n	TO	Operand 2	X ,. or blank
SET	Operand 1	EQ + - ⋮ ±	factor a factor b factor c factor n	X ,. or blank
MOVE * *	factor a factor b factor c	TO		Operand 2 ,. or blank
SET	Operand 1	EQ		factor a ,. or blank * factor b * factor c
MOVE /	factor a factor b			TO Operand 2 ,. or blank
SET	Operand 1			EQ factor a ,. or blank / factor b
MOVE **	factor a factor b			TO Operand 2 ,. or blank
SET	Operand 1			EQ factor a ,. or blank ** factor b

The first two forms are limited entry, the other six are extended entry.

Sequence Control Actions

Sequence control actions are constructed as follows:

form:	Action Operator	Operand 2
example:	GO TO	UPDATE

read as: Transfer control to the decision table called UPDATE.

Notice that there is only one operand, and this always appears in the Operand 2 position (see Mechanics).

OPERATORS

There are two sequence control operators:

GO TO

DO

These operators are used to transfer control to a specified decision table or procedure. GO TO transfers control permanently. DO transfers control, but control is later returned to the originating table.

OPERANDS

The following operands can be used:

PROCEDURE Procedure Name
Decision Table Name
Decision Table Identification

SPECIAL RULES

The last action in each rule of an open decision table except table STOP must be a GO TO. A table can GO TO or DO itself. All tables are entered at the beginning; therefore a specific rule or action cannot be selected by a GO TO or DO action.

USAGE

GO TO and DO provide the means for a complex logical problem to be subdivided into manageable segments. Theoretically, they can be eliminated by writing just one decision table for a job. However, experience indicates that effective system analysis and description occurs when the problem is subdivided into smaller parts.

GO TO Sequence Control Action: The GO TO action can only be the last action in a decision table rule. There must be a GO TO action in each rule of an open table, which transfers control to the table or procedure specified. A rule of a closed table may have a GO TO action but this is usually not the case, as the return path to the originating table is destroyed.

The three operands that can be used are:

PROCEDURE Procedure Name
Decision Table Name
Decision Table Identification

The first table of a procedure is entered when control is transferred via the GO TO action. The first table of a procedure is identified by

writing FIRST in the Table Type area of the Decision Table form. The initial table of the procedure must be an open table.

DO Sequence Control Action: A DO action can appear in either an open or a closed decision table. It always transfers control to a closed table or procedure, but retains the return path to the originating table. After carrying out the logic in the closed table referenced, control is transferred back to the originating table to the action following the DO. A table may DO itself; there may be any number of levels in a series of DO tables; that is, table 1 may DO table 2 which may DO table 3, etc. Eventually the DO path is retraced to the original table, unless there has been an intervening GO TO action which destroys the return path. Three operands may be used with DO:

PROCEDURE Procedure Name
Decision Table Name
Decision Table Identification

If a procedure is referenced, the first table in the procedure is entered and it must be a closed table. In fact, all tables in a procedure to be referred to by a DO must be closed. The FIRST table then may call upon other closed tables which make up the procedure. No GO TO actions may appear in the procedure unless it is desired to leave the procedure without returning.

Association control provides a means of handling multiple recurring fields or associations of fields.

Operand 2 can have either of two forms:

Decision Table Name or Identification

PER ASSOC I

PER ASSOC J

etc.

I and J represent non-related association codes indicated in the File Description for multiple recurring fields.

Decision Table Name or Identification

PER ASSOC I

PER ASSOC J

etc.

I and J represent related association codes indicated in the File Description for multiple recurring fields.

In the first form (PER ... PER), the association values indicated are all set to 1 to start with. The first time through the specified closed table, the first of the association fields is used for conditions and actions. After execution of the table, and when there are more occurrences of all the fields in the associations, association values are incremented by 1 (set equal to 2 the second time around). The next time through the same closed table, the second occurrence of the association fields is used. This continues until one of the associations runs out of recurring fields. This is a very powerful format allowing many independent fields or

groups of fields to be varied simultaneously. However, it can lead to errors if not closely analyzed. Association values can be directly analyzed in conditions in a table just as any other Field Name, but then, values cannot be changed by MOVE or SET. A table can be repeated for each value of the association until completion, but particular conditions can inhibit actions except under controlled circumstances.

The second form (PER ... FOR) provides hierarchy control of associations. If Assoc J varies within Assoc I, then J is incremented and exhausted for I = 1; I is then advanced to 2, J reset to 1, and the procedure continued until Assoc I is exhausted.

MECHANICS

The operand always appears in the Operand 2 position, which is the second column of the stub or the entry column. Procedure Name is always preceded by the word PROCEDURE. Since a Decision Table Name starts with a letter and a Decision Table Identification starts with a number, these can easily be differentiated. Either limited or extended entry form can be used. Illustrations are as follows:

	STUB	ENTRY
GO TO	PROCEDURE procedure name decision table name decision table identification	X, . or blank
DO	PROCEDURE procedure name decision table name decision table identification	X, . or blank
GO TO		PROCEDURE procedure name , . or blank decision table name decision table identification
DO		PROCEDURE procedure name , . or blank decision table name decision table identification
DO	decision table identification or name PER ASSOC letter	X, . or blank
DO	decision table identification or name PER ASSOC letter FOR ASSOC letter	X, . or blank
DO		decision table identification or name PER ASSOC letter , . or blank PER ASSOC letter
DO		decision table identification or name, . or blank PER ASSOC letter FOR ASSOC letter PER ASSOC letter FOR ASSOC letter

The first and third forms are limited entry and the second and fourth forms are extended entry.

Input/Output Actions

Of all the operations required in a descriptive language those relating to input and output are the most complex and cause the greatest difficulty. The reasons for this include:

a) Current input and output devices have certain limitations and conventions not directly comparable to manual filing and recording systems.

b) The concept of "memory" as different from input/output devices has no analogy in manual data processing.

c) Users seem less willing to accept constraints and artificial mechanics when dealing with records and files than with fields and character positions.

Thus, the input/output portion of any language is a compromise between unrestricted generality with resultant implementation inaccuracies and constrained mechanical manipulation with loss of logical clarity. The compromise chosen here considers the capabilities and structure of various input and output devices, but allows the user to think about these in the most convenient way. Although certain internal or memory records may be used for intermediate information, they are not involved in input/output actions.

Types of Files

Files are classified in two ways, by the function they serve, and by the type of access to them. Functionally, a file can be input only, output only, intermediate (input and output), and retained (input and output).

Two types of access are allowed for files, serial and random. For a specific file, the type of access depends largely on the medium on which the file is written. A serial file allows access to only the next physical record on the file, and is represented by many common media (e.g., punched cards, magnetic tape, printed reports, punched paper tape, and Teleprocessing lines). A random file allows access to any record of the file, but each such record must be specified. Typical media for random files are magnetic drum and disk. Under certain conditions, a specific medium can be used for both serial and random files. For example, a magnetic drum can be read serially by sequential addressing; and a magnetic tape can effectively be read randomly by searching for a specific record (rejecting all other records).

The input/output action operators, and the functional types of files with which they can be used, are discussed separately for serial and random files.

Serial Files

Consider a serial file as a roll of film, each frame holding one record. Only one frame can be viewed or exposed at a time.

An input file is a roll of exposed film. The roll is read serially from the first to the last frame.

An output file is a roll of unexposed film. A frame at a time can be exposed starting with the first frame. A whole series of frames (records), either wholly or partially ready for recording, can be available at a time.

An intermediate file is similar to an output file, but after a number of frames are exposed (and cut off), it can be used as an input. After use as an input, it can be saved, or it can be destroyed and a new roll given that same name. While being exposed, an intermediate file is treated just like an output; while being viewed, it is just like an input.

The retained file is somewhat more sophisticated. As each frame comes into viewing position, it is cut off the roll. It then can be added to a separate output roll or discarded. New frames can be added to the output roll or old frames modified before being re-exposed. After reading a retained file the input roll is destroyed and only the output roll retained. This in turn may be further read and written.

The only condition meaningful for a serial-type file is:

File Name IS END OF FILE

If END OF FILE is not checked and a READ is given beyond the last record, the READ operation does not occur.

Sequence is maintained for an input file as specified on the File Description. Input files cannot be sorted. An output file is presumed to be in the sequence specified for the data from which it was developed, unless a different sequence is specified through the SORT operator. This also applies to intermediate files. A retained file is presumed to be in the sequence specified and this sequence cannot be changed. If no sequence is specified, then none can be obtained operationally.

OPERANDS

Record names and file names are the principal operands used with the input/output operators for serial files. However, symbolic expression names and field names are used with the SORT operator.

OPERATORS

Four input/output operators are used with serial files:

READ provides for obtaining information from a file by making one record from that file available in memory for processing.

WRITE provides for recording information on a file by placing one record from memory in that file. The connector ON can be used with WRITE.

BEGIN enables one to start over again to READ or WRITE a particular file.

SORT rearranges the records on a file according to particular keys on those records.

READ (Serial File)

The READ operator is used in the form:

READ operand

The operand must be a file name. This action is used to read the next record of an input, intermediate, or retained file into a memory area that is large enough for the largest record of the file.

INPUT	OUTPUT	INTERMEDIATE	RETAINED	STUB		ENTRY
X		X	X	READ	File Name	X, . or blank
X		X	X	READ		File Name, . or blank

The first form shown is limited entry, and the second form is extended entry.

WRITE (Serial File)

The WRITE operator can be used in two forms:

WRITE operand

WRITE operand 1 ON operand 2

The first form requires that the operand be a record name for an output or intermediate file. It can be a record name or a file name for a retained

file. WRITE cannot be used with an input file. The second form requires that operand 1 be a record name from some other file (including internal files), and that operand 2 be a file name. This form can only be used for a retained file. A WRITE places the record selected in the next record area on the file. For output and intermediate files, there is one memory area for each type of record while these files are being written. There is only one common record area for each retained file.

INPUT	OUTPUT	INTERMEDIATE	RETAINED	STUB		ENTRY
	X	X	X	WRITE	Record Name	X, . or blank
			X	WRITE	File Name	X, . or blank
			X	WRITE Record Name	ON File Name	X, . or blank
			X	WRITE Record Name	ON	File Name, . or blank
			X	WRITE Record Name		ON File Name, . or blank
	X	X	X	WRITE		Record Name, . or blank
			X	WRITE		File Name, . or blank

The first three forms shown are limited entry. The remaining forms are extended entry.

BEGIN (Serial File)

The BEGIN operator is used in the form:

BEGIN operand

The operand must be a file name. This action can only be used with input, intermediate, and retained files. For an input BEGIN means "prepare to read the first record again". For an intermediate file it means "return to the first record position". If the next action on the file is to be a READ, the file is considered an input file; if the next action is a WRITE, it is considered an output file. For a retained file, BEGIN means "return to the first record of the newly constituted file". If BEGIN is used while processing a retained file, the portion of the file that has not been read is assumed to be destroyed.

INPUT	OUTPUT	INTERMEDIATE	RETAINED	STUB		ENTRY
X		X	X	BEGIN	File Name	X, . or blank
X		X	X	BEGIN		File Name, . or blank

The first form shown is limited entry. The second form is extended entry.

SORT (Serial File)

The SORT operator is used in the form:

SORT operand

The operand is:

file name

PER key name

The key name can be a symbolic expression name or a field name common to all records of that file. The file is always sorted into ascending order according to the defined collating sequence. The file name must be an output or intermediate file. An intermediate file must first be operated on by the BEGIN operator to insure that the file is ready for reading.

INPUT	OUTPUT	INTERMEDIATE	RETAINED	STUB		ENTRY
	X	X		SORT	File Name PER Symbolic Expression Name	X, . or blank
	X	X		SORT	File Name PER Field Name	X, . or blank
	X	X		SORT		File Name, . or blank PER Symbolic Expression Name
	X	X		SORT		File Name, . or blank PER Field Name

The first two forms shown are limited entry. The remaining forms are extended entry.

Random Files

Random files have the same functional classification as serial files:

Input -- read only

Output -- write only

Intermediate -- a temporary file

Retained -- a permanent file

Consider a random file as a group of film slides that are in a box containing a series of slots, each slot identified by a location. Each slot holds one slide (or frame). A slide must be raised to be examined; to raise another slide the one just raised must be lowered. Changes can be made to the raised slide, just as though it were only partially exposed. New slides can be inserted into empty slots, or be inserted into occupied slots by removing (and destroying) the previous slide in that slot. A particular slide can be destroyed by clearing the slot in which it is stored.

There is no meaning to END OF FILE for a random file because of the media usually used by such files.

OPERANDS

Record names and file names are the principal operands used with the input/output operators for random files. However, logical expression names and location names are used for controlling the READ and WRITE operators.

OPERATORS

Three input/output operators are used with random files:

READ provides for obtaining a specific record from a file.

WRITE provides for inserting additional records in a file. The connector ON can be used with WRITE.

CLEAR is used to remove unwanted records from a file.

READ (Random File)

The READ operator is used in the form:

READ operand

The operand can be either:

file name
PER location name

or

file name
PER logical expression name

READ file name

PER location name

obtains the record at the location specified by location name. If there is no record at that location, the read operation is not executed.

READ file name

PER logical expression name

obtains the first record for which the logical expression value is true. This operation is defined on a circular basis. It starts searching from

the location of the previous record that was read, and continues through the end of the physical file and around again to the same record. If there is no record that satisfies the conditions of the logical expression the READ does not take place.

READ can be used with input, intermediate, or retained files. There is just one memory area for the file. A READ temporarily removes the record from the file. For an input file, it is automatically returned in its original location just prior to the subsequent READ of that file. For an intermediate file and a retained file, the record is returned to its original location, but it may have been modified.

INPUT	OUTPUT	INTERMEDIATE	RETAINED	STUB	ENTRY
X		X	X	READ	File Name PER Location Name	X, . or blank
X		X	X	READ	File Name PER Logical Expression Name	X, . or blank
X		X	X	READ		File Name, . or blank PER Location Name
X		X	X	READ		File Name, . or blank PER Logical Expression Name

The first two forms shown are limited entry. The remaining forms are extended entry.

WRITE (Random Files)

The WRITE operator can be used in two forms:

```
WRITE                          operand

WRITE      operand 1 ON        operand 2
```

In the first form, the operand can only be a record name for an output file. For an intermediate or retained file, the operand can be a record name or a file name. The second form can be used only for intermediate and retained files. Operand 1 must be a record name, and operand 2 a file name.

The operand in the first form has this construction:

```
file name (or record name)
PER location name

file name (or record name)
PER logical expression name
```

The same construction is used for operand 2 in the second form, except that a record name cannot be used.

After WRITE the record in memory is not affected.

INPUT	OUTPUT	INTERMEDIATE	RETAINED	STUB		ENTRY
	X	X	X	WRITE	Record Name PER Location Name	X, . or blank
		X	X	WRITE	File Name PER Location Name	X, . or blank
	X	X	X	WRITE	Record Name PER Logical Expression Name	X, . or blank
		X	X	WRITE	File Name PER Logical Expression Name	X, . or blank
		X	X	WRITE Record Name	ON File Name PER Location Name	X, . or blank
		X	X	WRITE Record Name	ON File Name PER Logical Expression Name	X, . or blank
		X	X	WRITE Record Name	ON	File Name, . or blank PER Location Name
		X	X	WRITE Record Name	ON	File Name, . or blank PER Logical Expression Name
	X	X	X	WRITE		Record Name PER Location Name
		X	X	WRITE		File Name PER Location Name
	X	X	X	WRITE		Record Name PER Logical Expression Name
		X	X	WRITE		File Name PER Logical Expression Name
		X	X	WRITE Record Name		ON File Name, . or blank PER Location Name
		X	X	WRITE Record Name		ON File Name, . or blank PER Logical Expression Name

The first six forms shown are limited entry. The remaining forms are extended entry.

CLEAR (Random File)

The CLEAR operator is used in the form:

CLEAR operand

The operand is:

file name

CLEAR can be used only with intermediate or retained files to remove an unwanted record from the file. An intermediate file is assumed to be cleared prior to the start of the process. CLEAR prevents the last record read from being returned to the area from which it was read.

INPUT	OUTPUT	INTERMEDIATE	RETAINED	STUB		ENTRY
		X	X	CLEAR	File Name	X, . or blank
		X	X	CLEAR		File Name, . or blank

The first form shown is limited entry. The second form is extended entry.

CHAPTER 7

Appendix

The appendix provides detailed explanations on filling out the various forms plus compact summaries of certain features of the language (e.g., which operands may be used with which operators, effect of different value class on conditions and actions, etc.).

The categories are as follows:

File Description Sheet

Reference Operand Description Sheet

Decision Table Form

Report Layout Sheet

Operand Summary

Condition Summary

Action Summary

FILE DESCRIPTION SHEET (See Figure 7-1)

The File Description Sheet is used to describe files, records, and fields. Each type of record within a file is described on a separate File Description Sheet. In addition certain information about the file itself is placed on each sheet. The complete description of the file is contained on the several sheets used to describe the records which make up the file.

SYSTEM NAME	IBM	ANALYST	DATE	
(1)	FILE DESCRIPTION	(2)	(3)	PAGE (4) OF (5) PAGES

FILE NAME	DIR. TYPE	MEDIUM		RECORD NAME	ABB	NO OF RECORDS	
						AVERAGE	PEAK
(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)

SER NO	FIELD NAME	ALPHA NUMERIC	SIGN	LENTH		OCCURRENCES			ASSOC	ASSOC TO	SEQUENCE	RELATIVE LOCATION		VALUE / REMARKS
				MAX	VAR	MIN	AVG	MAX				BEGIN	END	
(14)	(15)	(16)	(17)	(18)	(19)	(20)	(21)	(22)	(23)	(24)	(25)	(26)	(27)	(28)

REMARKS: (29)

General Headings

At the top of the sheet the systems engineer will indicate:

(1) the system name

(2) his own name

(3) the date of preparation or revision

(4) a serial assignment of page number, and

(5) the total number of pages which should normally equal the number of records names in the file; it will be greater if any record takes more than one page.

File Description Headings

(6) The file name is shown

(7) For directional type the systems engineer will indicate:

IN, OUT, RET (for retained), or INTER (for intermediate). IN represents a file used only for input; OUT represents output only; retained (RET) represents a file retained for reuse; intermediate (INTER) represents a file originated and used by the system but not available for subsequent use.

Medium covers two entries. (8) The first relates to whether the file is to be considered in a serial (SER) or random (RAN) manner. Serial files can be read and written one record after the other without need to control location. Random files are read and written per location or content. Serial files have an explicit End of File indication after the last record; random files do not. Serial Files cannot be interrogated

without being read (except for End of File); random files can. Serial retained and intermediate files must be written to be preserved; random files do not.

(9) The second aspect of medium is related to the specific medium on which the file will appear. Suitable entries include simple abbreviations for:

Punched Card (PC)

8 Channel Punched Paper Tape (8 PAP TAPE)

Magnetic Tape (MAG TAPE)

8 1/2 x 11 Printed Continuous Form (8 1/2 x 11 CONT)

1301 Disk

Inquiry Station (INQ)

(10) A unique record name is inserted together with (11) a suitable abbreviation which must also be unique. Abbreviations are normally used for field name qualifiers to reduce writing. (12) The average number of this type of record in the file is noted and (13) the peak number expected. Other volume information may be shown under remarks (29).

Record Continuation

If there is a need to use more than one sheet for a single record the general heading (1 through 5) should be written, along with the file name (6) and (10) the record name with the letters CONT in parentheses; e.g., CUSTOMER RECORD (CONT).

Field Description

The fields which may appear on the record need not be described in any particular order. There is no implication that the reference order establishes relative position of these fields. Sub-fields do have positional implication as do fields on printed report records.

If all field descriptions (14-28) of a record are identical to another record in the same file or even a different file, enter the general heading (1-5) and the file description heading (6-13). (14) Enter Serial Number as 001 and (15) the words SAME AS followed by the record name of the File Description Sheet that contains the appropriate field descriptions.

SERIAL NUMBER (14) A serial number is assigned to each field or sub-field. Serial numbers will begin with 1 or 001 and continue as far as required. Serial numbers can begin with 1 for each record description since they are only unique within a record description. If additional lines are required for any field the serial number is not changed; it may therefore be omitted on the subsequent lines.

FIELD NAME (15) Under field name the name of each field will be inserted. While there is no limit on the length of a field name, it should, for convenience, be kept brief and mnemonic. Lengthy names may use more than one line with the break being shown by a hyphen if it is in the middle of a word. Names for sub-fields will also be placed in this column. Descriptive information about a field will be written on the first line for that field.

ALPHA/NUMERIC (16) In the alpha/numeric column the field is indicated as: N for numeric meaning that the value is always composed of the digits 0 through 9, A for alphameric meaning that any characters, including letters, numbers, and special symbols may be used; or F for floating point numbers. A numeric field may represent an integer or decimal and is acceptable for arithmetic manipulation; similarly for a floating point field. Alphameric fields while they may only have numeric characters in them at any given time are not suitable for arithmetic since neither sign nor decimal point are defined.

SIGN (17) Under the sign column will be indicated for each numeric and floating point field its sign possibilities. A field which is always zero or positive will have a plus sign (+). A field which is always zero or negative will be marked by a minus sign (-). If a field may be both positive or negative an E (for either) is inserted. Unsigned numeric fields, which are treated as positive in arithmetic are marked as U. Alphameric fields have the sign column left blank. The sign possibilities for floating point numbers will always relate to the mantissa.

LENGTH -- The systems engineer will indicate (18) under MAX the maximum length for the value of the field. For a number (either N or F) the maximum integer length will be noted, then a decimal point, and finally the maximum decimal length. The sum of these indicates the number of significant positions. For all numbers the sign, decimal point, plus the floating point notation (including the F and the sign and

integer which follows) will not be considered in determining the length of the field (e.g., +16 has a MAX of 2, -17.350 has a MAX of 2.3, +27.1F-3.5 has a MAX of 2.1). For an alphameric field its maximum length will be indicated. (19) If the length of a field may vary considerably (like in name and address) this will be indicated by placing a check mark in the VAR column.

OCCURRENCES -- Certain fields may occur more than once in a record. If a field always appears exactly once per record a 1 is placed under MAX (22). If it is sometimes absent then this is shown by placing a 0 in MIN (20) and the proper average under AVG (21). A field which always occurs a fixed number of times greater than one, will have that number inserted in all three columns. If the number of times that a field appears on a record varies, the systems engineer will indicate the minimum, average, and maximum number of times that the field occurs on the record. Sub-fields will be blank in the occurrence columns because they can occur only once per field. Occurrences are always in terms of the "association to" as explained next.

ASSOCIATION -- On multiple occurring fields there is a need to be able to keep track of which particular field is being handled and to know which other fields occur in association with this field. For these reasons an association code letter is inserted under (23) ASSOC to indicate mutually dependent fields. Association code is blank for sub-fields because they have the same association as the parent field. Fields which occur a

maximum of once per record may not belong to an association. Association codes may be any letter and need only be unique for a record since they can be qualified by record name when used. All associated fields in a record will carry the same association code.

This one level association scheme works conveniently for fields whose occurrences only vary per the basic record. Items per invoice might well be one association including item number, item name, and item price, while taxes per invoice might be another independent association incorporating tax type and tax amount for federal, state, and local taxes.

A more complex problem is how to handle associations which are nested. For instance, there might be many operations to make a particular product; each operation might call on many different fixtures and tools; certain fixtures and tools might call for special instructions. To take care of this, each association or field which doesn't vary directly with the record is related to the controlling association by noting the controlling association code under (24) ASSOC TO. The following example may clarify how these columns are used:

Field Name	Occurrences			Assoc.	Assoc. To
	Min.	Avg.	Max.		
Part Number			1		
Operation Number	1	5	25	A	
Operation Description	1	5	25	A	
Operation Time Standard	1	5	25	A	
Tool or Fixture Number	0	2	5	B	A
Tool Location	0	2	5	B	A
Special Instructions	0	.2	1	C	B

Occurrences for items in an association must be identical. This means that association A which occurs 1 to 25 times per the record consists of operation number, operation description, and operation time standard. Association B consists of two fields, tool or fixture number and tool location; these occur 0 to 5 times per operation (hence 0-125 times per record). Special instructions, Association C, is associated to and occurs 0 to 1 time for each tool (hence will occur 0 to 125 times per record).

SEQUENCE -- If a file is to be considered as ordered, the systems engineer will indicate in (25) SEQUENCE the fields on each record which are used to sequence the file and the relative order of these fields. Numbers are indicated in the column, 1 for major, 2 for intermediate, etc. If any records in a file are ordered all must be ordered. A common named field must be included in each record to indicate the overall sequence of the file. For input and retained files sequence may be used. If used for intermediate or output files this can be nothing but informative to the reader. The analyst must still indicate through a SORT operation the particular sequence he wishes if this is different from the sequence in which it was developed. Files which are not considered sequenced have this column blank; sequence may still be introduced operationally by using SORT for intermediate and output files. Additional sequencing information may be indicated in the remarks section (29).

RELATIVE LOCATION -- There are two uses for relative location:

to indicate the relation of a sub-field to its parent field, and

to show the line of the record and beginning and ending locations for fields on printed reports.

Each field which has sub-fields must have a length which doesn't vary (e.g., Length: VAR is blank). This allows positional reference to the field by the sub-fields. The sub-field has indicated the relative beginning position (of the referenced field) in (26) BEGIN and the ending position in (27) END. The referenced field name is noted in (28) the value/remarks area. An example may help explain:

	Length	Relative Location		
Field Name	Max.	Begin	End	Value/Remarks
DATE	8			
MONTH	2	1	2	DATE
DAY	2	4	5	DATE
YEAR	2	7	8	DATE

The other use of relative position handles actual print position on a printed report. With the Report Layout sheet filled in the analyst must relate each field name to the print positions on the Report Layout sheet and the line of the record. Hence (26) the line of the record is shown for each field and (27) the ending print position of the field. If there is a possibility of misunderstanding the beginning print position, enter the beginning print position in (26) BEGIN after the line number of the record and separated from the line number by a comma; e.g., 2, 42 47.

VALUE/REMARKS (28) The value/remarks column will be used for several purposes. If a file is an output or intermediate and the systems engineer wants to describe an actual value which a particular field will always have, he may do this by writing the value as a literal. If a sub-field is described on the line, the name of the field of which the sub-field is a part will be noted. If values are assigned to sub-fields, the value will follow the field name separated by a comma. This column may also be used by the systems engineer for any brief remarks which he desires to make about any field such as format, possible values, accuracy, etc.

REMARKS (29) At the bottom of the File Description sheet any remarks concerning the record or file may be made: information on sequencing; variations in record size or number of records; source of input or destination of outputs.

REFERENCE OPERAND DESCRIPTION SHEET (see Figure 7-2)

The Reference Operand Description Sheet is used to describe constants, expressions, value lists, and reference lists. None of these operands can have their values changed by the decision tables.

General Headings

At the top of each sheet will be indicated (1) the name of the system, (2) the name of the systems engineer or analyst, (3) the date the sheet is prepared or revised and (4) the page number. After completion, (5) the total number of pages of the Reference Operand Description sheets is shown for control purposes.

SYSTEM NAME	IBM	ANALYST	DATE	
(1)	REFERENCE OPERAND DESCRIPTION	(2)	(3)	PAGE (4) OF (5) PAGES

SER NO	NAME	TYPE	VALUE
(6)	(7)	(8)	(9)

FIGURE 7-2

Operand Description

(6) A different serial number is used for each operand. These should start with 1 and continue sequentially as far as needed. The serial number may be a 1 - 3 digit integer.

Because the use of the form varies considerably for different types of operands, the form will be discussed under an appropriate heading for that operand.

If an operand requires continuation beyond the initial page the general headings (1 - 5) should be repeated, the serial number (6) also repeated, and (7) the name of the operand with the letters CONT in parentheses written after it.

CONSTANTS

(7) The constant name will be written in the name column. Names can be continued to more than one line if needed; if this requires breaking in the middle of a word then a hyphen should be inserted. (8) Under type is written CONSTANT or CON. (9) In the value column will be indicated the value of the constant exactly as it is to be used. If the value is numeric it is to be shown directly as with any literal. Alphameric values are to be enclosed in quotes. There is no length limit for constants.

EXPRESSION

(7) The name assigned to an expression will be written in the name column. (8) Under type will be written a mnemonic code to indicate the

type of expression:

Mnemonic Code	Interpretation
ARITH EXP or AE	Arithmetic expression
SYMB EXP or SE	Symbolic expression
LOG EXP or LE	Logical expression

(9) The expressions themselves will be written in the value column, using as many lines as are required; an operand should not be hyphenated. An arithmetic expression will be written using the arithmetic operators, functions, parentheses, and operands as allowed. Individual factors in the arithmetic expression may be set off by blanks to aid readability.

(9) For symbolic expressions the value column will also contain the expression. The Symbolic Operator name (currently only JOIN is available) will be followed by a left parenthesis; then the operands whose values will be put together to create the value for the expression will be written. Each of these operands will be separated by commas and written in the order they are to appear in the resultant value. The expression is terminated with a right parenthesis.

Suppose IDENT NO were given as the symbolic expression name for the expression JOIN (DEPT NO, '-', DIV NO, ' / ', MAN NO). If the current values of the operands are:

DEPT NO = 726

DIV NO = 14

MAN NO = 123456

the value for IDENT NO would be 726-14/123456.

(9) A logical expression will be written in the value column using the operands, relational operators, logical operators, arithmetic operators, and parentheses required. Blanks will be used to insure readability. In constructing logical expressions note the following:

1) The simplest form consists of a two operand condition like MARITAL STATUS = 1
PAY AMOUNT = 24.00

2) Following generally the same rules for constructing conditions, each of these operands may be more complex. Note the use of THRU instead of the hyphen for ranges.

For example, BOARDING CITY EQ "ATL", "MIA", "LAG"
TAXABLE INCOME EQ 4000.00 THRU 7999.99

3) Arithmetic expressions may be introduced for either operand following the same rules as those for constructing arithmetic expressions. An example is:

CREDIT BALANCE + NEW ORDER AMOUNT GR
(CREDIT CLASS * CREDIT RATE) + CREDIT BASE

4) Similarly, special arithmetic <u>functions</u> may be used:

SUM (OPERATING INCOME, INVESTMENT INCOME,
TAX RETURNS) LE (# DOW JONES IND AVERAGE
CURRENT YEAR # / # DOW JONES IND AVERAGE
LAST YEAR #) * INVESTMENT BASE

Symbolic expressions may be used:

JOIN (NAME, ', ', RANK) = "JOSEPH JONES, 1ST LT"

5) Finally, in even more complex situations multiple conditions may need to be considered. This is done by using OR, AND, or NOT. NOT should be used very infrequently since there is a full set of relational operators. AND means both, and OR means either one or both. Examples include:

RATE (HEALTH, AGE) GR 4.25 AND POLICY
REQUEST AMOUNT GE 25000
STATUS IS SENIOR OR AGE GR 23

VALUE LIST

(7) The name of the value list will be inserted in the name column. (8) The analyst will write VALUE LIST or VLIST under type. (9) In the value column the literal values comprising the list will be noted separated by commas. Each entry may be a single entry value or a range. The upper and lower limits of a range will be separated by a hyphen; the range is considered inclusive at both ends. The range itself will be separated from other entries by a comma. Numeric values are listed directly while alphameric values are enclosed in quotes. If either the upper or lower limit of a range is alphameric, both are individually enclosed in quotes. Values may continue on succeeding lines. Values need not be in any sequence. Alphameric and numeric values may be mixed in a list.

NAME	TYPE	VALUE
RECEIVING CITY	VALUE LIST	1, 3, 7, 9-11, 13, 16, 87, 2
SHIPPING CODE	VALUE LIST	"ATL", "GEO", "NOL", "4A" - "4H", 181 - 189, "LAG"

REFERENCE LIST

(7) The name of the reference list is given under name. This will usually contain some word like table or matrix to indicate the nature of the operand. (8) Under type the words REF LIST or RLIST is entered.

(9) On the top line the name of the first argument is shown - this name need not be unique since it is never referenced. If no obvious name suggests itself then ARGUMENT 1 or a suitable abbreviation should be used. A column should be drawn with this argument name as the title. The second argument name should be entered also on the top line and a column drawn. Finally the function name should be entered; this must be a unique name (or will be when qualified by the reference list name). A third column is drawn. (7) On the second line under name the reference list function name is written with the two argument names in parentheses following it. (8) Under type FUNC NAME is entered followed by a left parenthesis, then the number of values of the first argument, a comma, then the number of values of the second argument, and a right parenthesis.

If the actual values are not to be inserted until later the different values of each argument may be shown in the appropriate columns with the note in the third column saying "all combinations" or something expressive of how long the list will be. If the entries are to be inserted, each legitimate combination of argument 1 and argument 2 should be made with the appropriate value for the function. Figure 7-3 gives an example of such a reference list filled in.

SYSTEM NAME	IBM® REFERENCE OPERAND DESCRIPTION (PARTIAL)	ANALYST	DATE	PAGE OF PAGES

SER NO	NAME	TYPE	VALUE		
	INS RATE MATRIX	REF LIST	AGE	SECTION OF COUNTRY	RATE
	RATE (AGE, SECTION OF COUNTRY)	FUNC NAME (6, 4)	0-9	"EAST"	0.20
			↓	"WEST"	0.20
			↓	"SOUTH"	0.15
			↓	"NORTH"	0.10
			10-19	"EAST"	0.35
			↓	"WEST"	0.30
			↓	"SOUTH"	0.28
			↓	"NORTH"	0.20
			20-29	"EAST"	0.55
			↓	"WEST"	0.50
			↓	"SOUTH"	0.45
			↓	"NORTH"	0.25
			30-39	"EAST"	0.80
			↓	"WEST"	0.80
			↓	"SOUTH"	0.60
			↓	"NORTH"	0.40
			40-49	"EAST"	1.10
			↓	"WEST"	1.05
			↓	"SOUTH"	0.95
			↓	"NORTH"	0.80
			50-59	"EAST"	1.55
			↓	"WEST"	1.50
			↓	"SOUTH"	1.30
			↓	"NORTH"	1.10

Figure 7-3

SUMMARY OF ENTRIES FOR NAME (7), TYPE (8) AND VALUE (9)

USE OF REFERENCE OPERAND DESCRIPTION	NAME (7)	TYPE (8)	VALUE (9)
Constant	Constant Name	CONSTANT or CON	Literal value of constant
Expressions	Arithmetic Expression Name	ARITH EXP or AE	Actual arithmetic expression
	Symbolic Expression Name	SYMB EXP or SE	Actual symbolic expression
	Logical Expression Name	LOG EXP or LE	Actual logical expression
Value List	Value List Name	VALUE LIST or V LIST	Literal values
Reference List	Reference List Name and Function Name (Argument 1, Argument 2)	REF LIST or R LIST and FUNC NAME (Count-Argument 1 values, Count-Argument 2 values)	Argument 1 Name and Argument 1 Values; Argument 2 Name and Argument 2 Values; Function Name and Function Values

An abbreviated version would look this way:

Name	Type	Value		
SPECIAL FARE TABLE	REF LIST	BOARDING CITY	DESTINATION	FARE
FARE (BOARDING CITY, DESTINATION CITY)	FUNC NAME (5, 3)	"ATLANTA" "BOSTON" "MONTREAL" "WASHINGTON" "MIAMI"	"NEW YORK" "PHILADELPHIA" "CHICAGO"	ALL COMBINATIONS

DECISION TABLE FORMS (See Figures 7-4 and 7-5)

Decision table forms are used to describe the procedure logic of the system. The Decision Table Continuation sheet is used for horizontal continuation of a decision table. Extended entry or mixed entry tables will often require continuation since only 4 rules are provided on the Decision Table sheet. Since limited entry tables have 32 rules provided no special continuation form has been prepared. It is recommended that a limited entry decision table generally not exceed 32 rules; rather, the original table should be split into 2 or more tables. However, the Continuation sheet can be used for limited entry tables by drawing the proper vertical lines.

The type of data entered on the Continuation sheet is the same as on the Decision Table sheet. Therefore, the index numbers (circled numbers on figures 7-4 and 7-5) are the same and the entries will not be discussed separately except where noted.

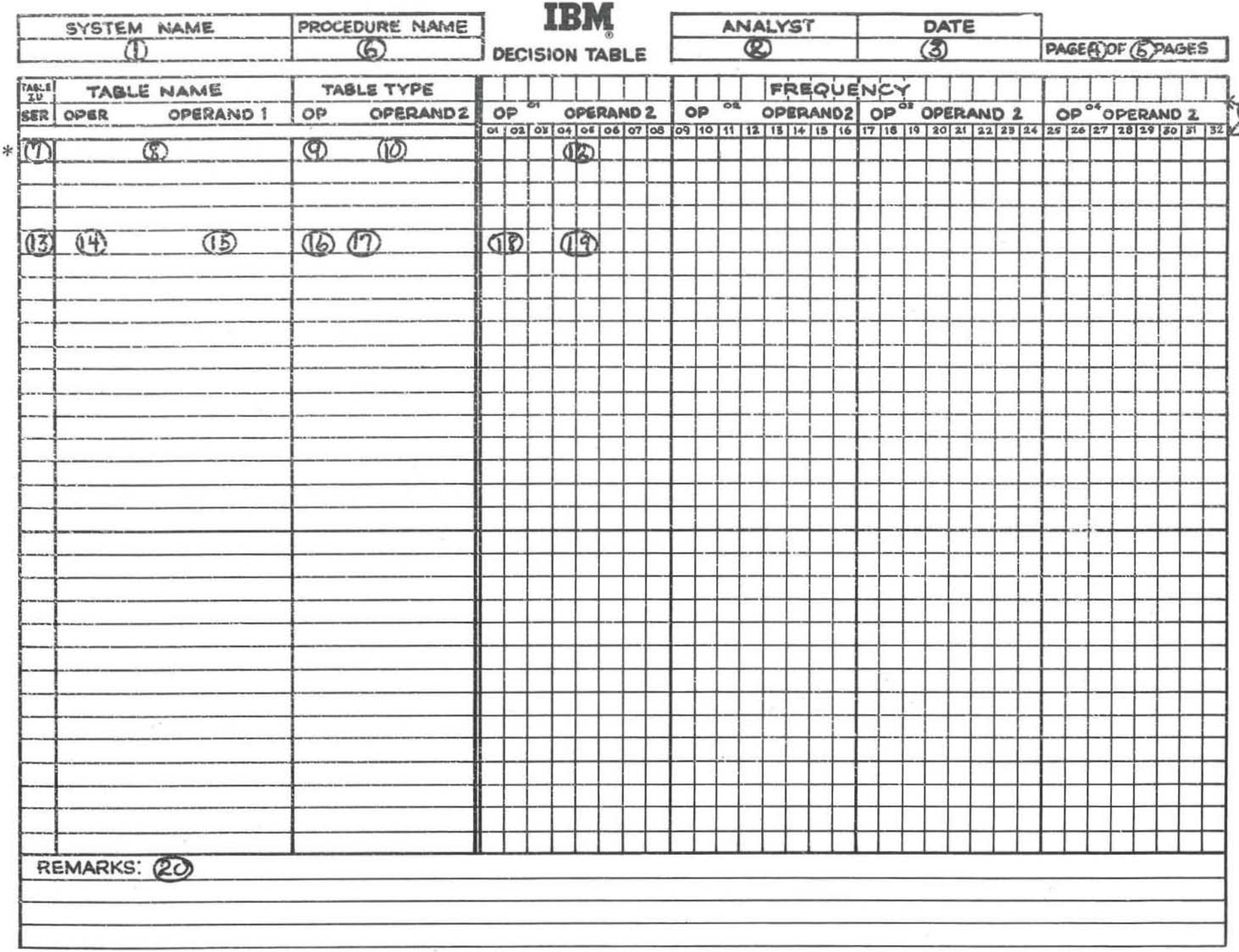

SYSTEM NAME
(1)
PROCEDURE NAME
(6)
IBM
DECISION TABLE
ANALYST
(2)
DATE
(3)
PAGE (4) OF (5) PAGES
TABLE ID
SER
TABLE NAME
OPER
OPERAND 1
TABLE TYPE
OP
OPERAND 2
FREQUENCY
OP 01 OPERAND 2
OP 02 OPERAND 2
OP 03 OPERAND 2
OP 04 OPERAND 2
01 02 03 04 05 06 07 08
09 10 11 12 13 14 15 16
17 18 19 20 21 22 23 24
25 26 27 28 29 30 31 32
(11)
** (7) (8) (9) (10) (12)
(13) (14) (15) (16) (17) (18) (19)
REMARKS: (20)

Figure 7-4

IBM

DECISION TABLE CONTINUATION

SYSTEM NAME	ANALYST	DATE	PAGE (4) OF (5) PAGES
(1)	(2)	(3)	

TABLE ID SER						FREQUENCY							
	OP	OPERAND 2	OP	OPERAND 2	OP	OPERAND 2	OP	OPERAND 2	OP	OPERAND 2	OP	OPERAND 2	(11)
** (7)		(12)											
(13)	(14)	(15)	(16)	(17)	(18)	(19)	(18)	(19)	(18)	(19)	(18)	(19)	

REMARKS:

Figure 7-5

General Heading

At the top of each sheet will be indicated: (1) the name of the system, (2) the name of the systems engineer, (3) the date the sheet is prepared or revised, (4) the page number, and (5) the total number of pages for this decision table. Each page of a decision table is numbered consecutively starting with 1. (6) On the Decision Table sheet enter the procedure name if the table is part of a procedure; otherwise leave blank. A single page may not contain tables from two different procedures, nor may it contain some tables which are part of a particular procedure and some which are not.

Decision Table Header

The first line of each decision table will be indicated by placing two asterisks (* *) in the left hand margin of the sheet.

On the first line of each decision table enter (7) the table identification, (8) the name of the table, (9) the table type (OPEN or CLOSED); if the decision table is the first table of a procedure (10) enter FIRST. The table identification is a 3 character field; the first position must be numeric to distinguish it from a table name, but the other two positions may be letters or numbers. The identification must be unique within the system as must table name.

If a table requires continuation to another sheet, the general headings (1-6) are inserted, as is (7) table identification and (8) table name; (9) CONT should be inserted under table type. More than one decision

table may be described on the same sheet by placing two asterisks on the margin of the new table header line. Page 2 is used for the second vertical page, then page 3, etc.

If horizontal continuation is needed to handle additional decision rules then a Decision Table Continuation sheet is used. The general headings are inserted (1-5) with the page number of the first horizontal continuation sheet following the page number of the last vertical decision table sheet. An illustration may help.

Dec Table Form page 1 of 9 Ser 1 - 10 rules 1 - 4	Continuation Form page 4 of 9 Ser 1 - 10 rules 5 - 10	Continuation Form page 7 of 9 Ser 1 - 10 rules 11 - 16
Dec Table Form page 2 of 9 Ser 11 - 20 rules 1 - 4	Continuation Form page 5 of 9 Ser 11 - 20 rules 5 - 10	Continuation Form page 8 of 9 Ser 11 - 20 rules 11 - 16
Dec Table Form page 3 of 9 Ser 21 - 30 rules 1 - 4	Continuation Form page 6 of 9 Ser 21 - 30 rules 5 - 10	Continuation Form page 9 of 9 Ser 21 - 30 rules 11 - 16

Figure 7 - 6

There are certain special tables. The initial table of the system is called START. Control will always begin at that point. START must be on open table. There must also be one STOP table per system. This provides for ending the procedure. The STOP table must also be an open table. In every procedure there must be a first table to which control

will be transferred when entering the procedure. The first table of a procedure must be open if the procedure is entered by a GO TO action. If the procedure is entered by a DO action then the FIRST table must be closed. Since the DO is interpreted as applying to the whole procedure the return outside the procedure is retained even if a GO TO is encountered. Hence, within a procedure, a GO TO in a closed table does not destroy the return path to the table which called the procedure.

Let's review the GO TO and DO actions relating to open and closed tables. Whenever a DO is encountered a return point is set up in a push down list to bring control back to the action following the DO action. If further DO actions are encountered additional return points are placed one on top of the other in the push down list. A DO can only call on a closed table. A closed table must have at least one decision rule with no GO TO action. If there is a GO TO in a closed table the entire contents of the push down list are destroyed (only within the current procedure, however). An open table may only be referenced by a GO TO action, and must have an explicit GO TO for each decision rule (except within a procedure when a decision rule without a GO TO will return control to the table calling on the procedure).

Decision Rule Header

(11) Rule numbers are preprinted on the Decision Table sheet. The rule number for extended entry form is at the top of the row; it is at the bottom of the row for limited entry form. If a Decision Table Continuation

sheet is used, the rule numbers must be entered in the same relative position as on the Decision Table sheet.

(12) Frequency information can be inserted on the first line of each rule by noting a 2 digit number to indicate the percent of times that each rule will be satisfied. The percents should add up to 100.

OPER OPERAND 1	OP OPERAND 2	RULE 01	RULE 02	
		80	20	← Frequency
CREDIT LIMIT	EQ 'LIMITED'	Y	N	

If 80% of the customers had limited credit, it would be expected that 80% of the time rule 01 would be satisfied. Rule 02 would then be satisfied 20% of the time. For extended entry form the percentage will be entered in the left-most part of the rule column.

If all possible combinations of conditions are not written and it is expected that the data to be processed will not always satisfy the explicit rules, an ELSE rule will be entered. The ELSE rule will be the first rule column to the right of the last written rule. For extended entry, enter (12) the word ELSE in the left-most part of the column. The letter E will be entered (12) for the ELSE rule on limited entry form. The frequency of the ELSE rule is presumed to be 100 minus the sum of the percentages of the other rules.

Table Body

The balance of the decision table is called the table body. The stub is separated from the entry by a preprinted, vertical double line. The condition statements will be separated from the action statements by drawing a double horizontal line after the last condition statement.

Condition Stub	Condition Entry
Action Stub	Action Entry

(13) Serial Number - A 1 to 3 digit serial number is assigned to each condition and action row in the table. Serial number is only unique for a single decision table. If a horizontal continuation is necessary, the serial numbers will be repeated for the corresponding rows of the continuation sheet. If multiple lines are required for a single statement, the serial number need not be indicated on the subsequent lines. A single heavy line will be drawn under each row when multiple lines are used. Conditions - For each condition row there will be (13) a serial number assigned. This number can be used for reference purposes, but does not indicate the order of examination. All condition serial numbers should be smaller than any action serial number. (15) The first operand name or value is shown under operand 1. If the name requires continuation it will be split where a blank actually occurs or else a hyphen will be used at the end of the operand 1 space. In limited entry form

(16) the operator and (17) the second operand will be shown in the stub except when VS (versus) is entered. (18) In the entry area on the first line of each condition enter Y (yes or true), N (no or false), period or blank (not pertinent or impossible) for each rule. If VS (versus) is entered in the stub, any relational operator may be shown in the condition entry, e.g., LE, GR, <, etc. These act just as though they were substituted for the VS.

In extended entry form the operator may appear in the stub (16) or the entry (18). The operator normally will appear in the stub area (16) if it is common for the row. If it varies, it must appear in the entry area (18). Operand 2 always appears in the entry area (19).

Actions - For all action rows, the action operator (14) will be entered in the stub. Operand 1 (15) is entered for action operators SET, MOVE and WRITE (when the WRITE operator is used with the connector ON).

	(14)	(15)	(16)	(17)
Limited entry example	WRITE	RECORD	ON	FILE

The other action operators do not require an operand 1.

The operators SET, MOVE and the two operand form of WRITE use connectors. The connectors are entered as shown in Figure 7-7.

In extended entry form operand 2 (19) is entered in the action entry area on the first line of the action statement. If the action is not to be taken, leave the entry area (18, 19) blank.

DESCRIPTION	OPERATOR	CONNECTORS	CONNECTOR LOCATION
LIMITED ENTRY			
Output	WRITE	ON	(16)
Value assignment	SET	EQ or ←	(16)
Two operand arithmetic	SET	+, -, * or /	(16)
Multiple operand arithmetic	SET	EQ or ←	(16)
		+, -, *, / or **	(16)
Value assignment	MOVE	TO or →	(16)
Two operand arithmetic	MOVE	+ - * or /	(16)
Multiple operand arithmetic	MOVE	TO or →	(16)
		+, -, *, / or **	(14)
EXTENDED ENTRY			
Output	WRITE	ON	(16) or (18)
Value assignment	SET	EQ or ←	(16) or (18)
Two operand arithmetic	SET	+, -, * or /	(16) or (18)
Multiple operand arithmetic	SET	EQ or ←	(16) or (18)
		+, -, *, / or **	(18)
Value assignment	MOVE	TO or →	(16) or (18)
Two operand arithmetic	MOVE	+, -, * or /	(16) or (18)
Multiple operand arithmetic	MOVE	TO or →	(16) or (18)
		+, -, *, / or **	(14)

Figure 7-7

An operand may be continued on as many lines as required. An operand should not be split between pages of the decision table. Either split at a blank or insert a hyphen.

(20) Remarks - At the bottom of the sheet, space is provided to permit comments to aid others in understanding the decision tables entered on the page, e.g., the desired length of the result of an arithmetic expression. If appropriate, reference may be made to the serial number of the condition or action. If more than one decision table is on the page the remarks should reference the table.

Any expression, (arithmetic, logical, or symbolic) that is referred to only by the decision tables on the page may be defined here instead of on the Reference Operand Description sheet. This is called a local expression. The name of the expression is written, then the type of expression (AE, LE, SE), and the expression itself following the same rules as noted for the Reference Operand Description sheet.

Appropriate local abbreviations may also be indicated under Remarks by indicating the abbreviation, then ABB, then the full name or value.

REPORT SPECIFICATION (See Figure 7-8)

Report format and content are described by using a Report Layout sheet (Figure 7-8) and one or more File Description sheets.

The Report Layout sheet will be used to specify:

1. The position of records and fields
2. The size of records and fields

IBM

INTERNATIONAL BUSINESS MACHINES CORPORATION

IBM 1403 SPACING CHART

6 Lines per inch

Form X24-6436-2

PRINTED IN U.S.A.

LINE DESCRIPTION

FIELD HEADINGS/WORD MARKS

SYSTEM NAME ①

ANALYST ②

DATE ③

PAGE ④ OF ⑤ PAGES

GLUE ⑥

⑦ ⑧ ⑨

0 1 2 3 4 5 6 7 8 9 10 11 12 13

Figure 7-8

DUAL-SPEED CARRIAGE

Punch only one channel per line. Holes in the same channel should be punched at least 8 lines apart.

MEASUREMENTS	FORM WIDTHS	COMMONLY USED FORM DEPTHS
Horizontal spacing 1/10" Vertical spacing 1/6"	Min. 3½" Max. 18¾"	3½", 3½", 3⅝", 5½", 6" 7", 7½", 8½", 11"

NOTE: This chart is subject to inaccuracies from variations in humidity. Dimensions of form should be calculated from measurements shown and not scaled from this chart.

3. The pre-assigned content of records

4. The editing of variable fields.

A separate File Description sheet will be prepared for each record of a report which contains one or more fields which will vary in value. The report is considered a file and the report name the file name. The File Description sheet is used to:

1. List and relate the names of the variable fields to the characters in the printer layout sheet which describe the position, size and editing of the fields.

2. Specify the unedited size of the variable fields.

The systems engineer will specify in decision tables the conditions for assigning different values to variable fields and the conditions for printing non-constant records. Many housekeeping type details are assumed to be taken care of automatically and hence need not be procedurally specified by the analyst.

REPORT LAYOUT SHEET (See Figure 7-8)

General Heading

(1) The system name, (2) analyst name, (3) date of preparation or revision, (4) page number, (5) total number of pages for the report, and (6) the report name (file name) are entered.

Figure 7-8 is one of many printer layout sheets that may be used for Report Layout. If another layout sheet is used, the general headings may appear elsewhere on the sheet.

A heavy rectangle is drawn on the printer layout sheet to show the relative size of the desired report page.

Record Heading

Each different line, or inseparable, continuous group of print lines, may be considered a record. When a record consists of more than one print line, all lines are printed when the WRITE action occurs for the record.

For operational control of variable spacing between printed lines, one or more blank lines may be designated as a record and written when desired.

Line Count (7) Enter a 1 or 2 digit number to indicate the number of print lines in the record. The line count will immediately precede the record name.

Record Name (8) The name of the record is entered on the first line that is used to describe the record. To continue a record name on a subsequent line, either break at a blank between words or enter a hyphen in the last character position. A record composed of fixed data fields and written in the same position on every page is called a constant record. All such records will be given the record name CONSTANT. A constant record may contain commonly understood variable fields. Presently Page Number and Date are the only variable fields that may be included in constant records. It is presumed that the programmer will supply the

proper page number and date on each page of the report. Other fields will be included if the need arises. Writing of CONSTANT records is not required. It is assumed that the programmer will make the necessary arrangements.

(9) Record Position

Records, when printed, may always appear in the same page position (Fixed) or they may appear anywhere on the page (Variable) except where CONSTANT records appear. To indicate print positioning, enter a code immediately after the record name, separated from the record name by a hyphen.

Code	Meaning
V	Variable - The record will be printed in the next available non-constant position.
F	Fixed - The record will be printed on the lines corresponding to the lines on which it is described.
2 Digit Number	Fixed - The record will start on the line specified by the numeric code. The code may not specify lines occupied by CONSTANT records.

Normally, the record which is to be fixed in position (Code F) will be described on the lines on which it is to be printed. When two or more fixed records must appear on the same or overlapping lines on different pages, the number code is used. If the fixed position on the current page has already been passed, it will be printed on the next page at the designated position.

(10) Record Description

A record may be composed of all variable value fields, all fixed value fields, or a combination of both. Except for pre-printed information, fields will be described in the body (10) of the Report Layout sheet. Pre-printed information will be shown by including a sample pre-printed form (or a picture thereof). The relative vertical and horizontal position for each field within a record will be shown on the Report Layout sheet.

Fixed Value Fields Fields that always have the same value are called fixed value fields. These will be indicated by writing their values in the proper positions on the Report Layout sheet and not referencing these positions in a File Description sheet.

Variable Value Fields Fields whose values change are called variable value fields. Their values must be supplied by the decision table procedure. Only variable value fields will be described on a File Description sheet.

If the same field can appear in a record more than once, it will be shown on the Report Layout sheet each time it can appear. The field will be described on the File Description sheet in the same manner as any other field except that all possible line numbers and print positions will be entered on separate lines in the Begin and End columns respectively. Variable value fields are shown by proper use of the symbols N or A; N representing numeric valued fields and A for alphameric.

Edited Fields Various characters are used on the Report Layout sheet to show the editing of variable value fields. The following describes the use of these characters.

There are four types of editing characters used on the Report Layout sheet:

1. Character Representation
2. Field Adjustment
3. Replacement
4. Insertion

When the field description on the File Description sheet (unedited field) differs from the edited field description on the Report Layout sheet (edited) field, the edited description will control. If the edited field is longer, zeros will be substituted for the missing high order integer and low order decimal positions, for numeric fields. Low order blanks will be added to alphameric fields. Decimal points will be aligned for numeric fields. If a decimal point is not specified, it is assumed to be immediately to the right of the right-most digit. For alphameric fields, if the edited field is shorter than the unedited field, the high order values will be printed and the extra low order positions dropped. For numeric fields, if the edited field is shorter, the high order decimal positions will be rounded and printed and the low order integer positions will be printed.

Character Representation

There are two characters used to indicate the character representation mode, A and N. A character will be printed for each of these, unless the character in the source field is blank.

CHARACTER	MEANING	UNEDITED VALUE	DESCRIPTION	PRINTED VALUE
N	the character position is always numeric (0-9) or blank	0164 03040 24	NNNN NNNN NNN	0164 3040 024
A	The character position may be any character (0-9, A-Z or a special symbol).	ABCD ABCD ABCD	AAAA AAA AAAAA	ABCD ABC ABCD_

Insertion Characters

Insertion characters add to the length of the printed field. They do not by themselves replace any character that would otherwise be printed. There are four types of insertions: special, floating, conditional, and normal.

Special - There is only one special insertion character. This is the decimal point (.).

CHARACTER	MEANING	UNEDITED VALUE	DESCRIPTION	PRINTED VALUE
.	Decimal point. For numeric fields the decimal points will be aligned. If a decimal point is not indicated, it is specified by the File Description sheet.	00123 00.123 00.127 .134	NNN N.NN N.NN N.NN	123 0.12 0.13 0.13

Floating Insertions - All floating insertion characters suppress lead zeros and lead blanks. The first floating insertion character will be placed next to the high order position of the edited field description and continued to the right for as many positions as are to be suppressed. On floating insertions, a minimum of two insertion characters must be indicated. These are three floating insertion characters:

$

+

-

CHARACTER	MEANING	UNEDITED VALUE	DESCRIPTION	PRINTED VALUE
$	Dollar sign. The dollar sign will be printed in the rightmost character position that is suppressed	123 012 _12 000 012 000	$$NN $$NN $$NN $$NN $$NN $$$$ $$$$	$123 $12 $12 $00 $_ _ $12
+	The plus sign causes a plus sign to be placed in the rightmost position in which suppression occurs if the field is not negative. A minus sign will be placed in the rightmost position in which suppression occurs if the field is negative.	+123 012 -001 000	++NN ++NN ++NN ++NN	+123 +12 -01 +00

CHARAC-TER	MEANING	UNEDITED VALUE	DESCRIP-TION	PRINTED VALUE
-	The minus sign causes a minus sign to be placed in the rightmost position in which suppression occurs providing the field is negative. If the field is <u>not</u> <u>negative</u>, insertion is not made made but lead zeros and blanks are suppressed.	-123 +123 -012 +012 -000	--NN --NN --NN --NN ----	-123 123 -12 12

<u>Conditional Insertions</u> - The conditional insertion characters are inserted in the exact position shown, only if one or more conditions are satisfied. There are five conditional insertion characters:

+

-

CR

DB

,

CHARACTER	MEANING	UNEDITED VALUE	DESCRIPTION	PRINTED VALUE
+	A plus sign in the high order or low order position of the edited field description will cause the placement of a minus sign in the specified location providing the field is negative. If the field is not negative a plus sign will be placed in the specified position.	-123 +123 123 -001	NNN+ +NNN NNN+ +NNN	123- +123 123+ -001
-	When a minus sign is the high order or low order position of the edited field description, it will cause the placement of a minus sign in the indicated position. If the field is not negative no character will be printed in that position	-123 +123 -123 -001 -001	-NNN -NNN NNN- -NNN NNN-	-123 123 123- -001 001-
CR	When these characters are the low order position of the edited field description, CR will be printed if the field is negative. If the field is not negative, the characters will not be printed.	+123 -123	NNNCR NNNCR	123 123CR

CHARAC-TER	MEANING	UNEDITED VALUE	DESCRIP-TION	PRINTED VALUE
DB	If the field is negative CR is printed; if the value is positive DB is printed; if the value is zero nothing is printed.	+123 -123 123	NNNDB NNNDB	123DB 123CR
,	A comma will be printed in the position indicated only if there are non-blank characters preceding.	1234 ---010 1030	NN,NNN NN,NNN NNN,NNN	1,234 010 1,030

Normal Insertions - Insertions will be made in the character position specified by the placement of the insertion character. Any numbers or special character may be inserted.

CHARAC-TER	MEANING	UNEDITED VALUE	DESCRIP-TION	PRINTED VALUE
0 - 9 Special Symbols	The character specified will be inserted exactly as shown in the field. It may not be the leftmost or rightmost character.	123062 123062 312.34	NN-NN-NN NN/NN/NN N@-$NN.NN	12-30-62 12/30/62 3@-$12.34

Replacement - Replacement characters are used to change the value of a field. They never add length to an edited field, because they will only replace existing characters. There are two types of replacement characters; Conditional and Normal.

Conditional - Conditional replacement characters specify the replacement of high order zeros or blanks of a field. There are two conditional replacement characters:

*

Z

CHARACTER	MEANING	UNEDITED VALUE	DESCRIPTION	PRINTED VALUE
*	Asterisk replacement. Replaces leading zeros or blanks of a field with asterisks.	12345 00123 00100 00000 01000 00000 --100 ---00 -----	***NN ***NN ***NN ***NN ***** ***** ***NN ***NN *****	12345 **123 **100 ***00 *1000 ***** **100 ***00 *****
Z	Suppress lead zeros for numeric fields ONLY. This character is used when it is desired to replace zeros that appear in the leftmost positions of a field with blanks.	12345 00123 00100 00000 00100 00000	ZZZNN ZZZNN ZZZNN ZZZNN ZZZZZ ZZZZZ	12345 123 100 00 100

Normal Replacement - Replacement characters replace the value of whatever character appears in the field with themselves. There are two normal replacement characters:

O (stands for zero)

B

CHARACTER	MEANING	UNEDITED VALUE	DESCRIPTION	PRINTED VALUE
O	Zero Replacement for numeric fields only. The placement of zero in an edited field will replace the corresponding character in the unedited field with a zero.	12345 12345 12345 12345	OOONN NNOOO OONN NNOO	OOO45 12000 OO45 23OO
B	Blank Replacement. The placing of B in a field is the same as zero except the character is replaced with a blank instead of a zero.	12345 12345 12345 12345 ABCDE ABCDE	BBBNN NNBBB BBNN NNBB BBBAA AAABB	___45 12___ ___45 23__ ___DE ABC__

Field Adjustment: - The field adjustment characters are used to specify the truncation or rounding of a field. Where they are written they specify the location of the rightmost character of the field. There are two field adjustment characters:

T

R

CHARAC-TER	MEANING	UNEDITED VALUE	DESCRIP-TION	PRINTED VALUE
T	Truncate. When the unedited value is longer than the edited description and the high order positions are to be printed, a T is placed in the right-most position. The fields will be left justified.	12345 112.345 12.3	NNNT NN.NT NNT	1234 12.34 12
R	Round for numeric fields only. Similar to Truncate except 5 is added to the right-most position to be dropped, the carry, if any, is made, and the field is truncated. The fields will be left justified.	12345 12345 123.45 456.78	NNNR NNR NNN.R NN.R	1235 123 123.5 56.8

If decimal points are aligned and the edited decimal position is shorter, rounding will occur unless truncating is specified.

OPERAND SUMMARY

The permissible operands are listed in Figure 7-9. The character set in proper collating sequence is listed in Figure 7-10. The character set listing includes the character names and information concerning the use of the various characters.

OPERAND	WHERE DEFINED	WHERE USED
Information Operands		
File	File Description Report Layout	READ, WRITE, SORT, BEGIN, CLEAR, State relational
Record		WRITE, MOVE, SET
Field		MOVE, SET, value relationals
Reference Operands		
Reference List	Reference Operand Description	MOVE, SET, relationals
Function		
Value List		value relationals
Constant		MOVE, SET, value relationals
Arithmetic Expression	Reference Operand Description	MOVE, SET, relationals
Logical Expression		MOVE, SET, relationals
Symbolic Expression	Remarks in Decision Table	MOVE, SET, relationals
Segmenting Operands		
Procedure	Decision Table Headers	GO TO, DO
Decision Table		GO TO, DO
Work Operands		
Work Record		WRITE, MOVE, SET
Work Field		MOVE, SET, relationals
Self-Describing Operands		
Literals		MOVE, SET, value relationals
Connotative Terms		MOVE, SET, value relationals
State Terms		State relationals
Descriptive Terms		MOVE, SET, value relationals
Value List		value relationals
Range		value relationals

Figure 7-9

CHARACTER SET in COLLATING SEQUENCE

Character	Name	Use
	Blank	other than first position in operand names; to set off operands from operators
.	Period or Decimal Point	to show decimal position in number
¤	Lozenge	
(	Left parenthesis	for delimiting portions of expressions
<	Less than	relational operator
≢	Group mark	
+	Plus sign	for addition; for positive numeric literals
$	Dollar sign	for editing on reports; for other than first position of operand names
*	Asterisk	for multiplication; for editing on reports
)	Right parenthesis	to delimit positions of expressions
;	Semicolon	
△	Delta	for other than first position of operand names
-	Minus sign or Hyphen	for subtraction; for continuation in the middle of a word; for negative numeric literals
/	Slash	for division
,	Comma	for editing on reports
%	Percent sign	for other than first position of operand names
=	Equal	relational operator; with SET operator

Figure 7-10

Character	Name	Use
'	Single quotation mark	to set off alphameric literals
"	Double quotation mark	
¢	Cents sign	for other than first position of operand names
#	Number sign	for other than first position of operand names; to set off descriptive terms
@	At sign	for other than first position of operand names
:	Colon	between qualified names
>	Greater than	relational operator
√‾	Radical	
?	Question Mark	
A to I	Alphabetic Characters	for operand names
!	Exclamation Point	
J to R	Alphabetic Characters	for operand names
≠	Record Mark	
S to Z	Alphabetic Characters	for operand names
0 to 9	Numeric Characters	for other than first position of operand names

Figure 7-10 (cont.)

Functions

Special mathematical functions may be used in arithmetic expressions. A function consists of a function name and, in parentheses, one or more

arguments. The arguments are field names, literals, arithmetic expressions, or functions.

If a multiple occurring field is used in an arithmetic expression, the arithmetic expression may be referred to only in a closed table. The rules that apply to the DO statement for handling multiple occurring fields are applicable as though the arithmetic expression is part of the closed decision table.

The following functions are explicitly defined with values as accurate as any other in the expression or as accurate as the receiving or comparing field in the decision table. In the case of comparing two arithmetic expressions each consisting only of functions, the result will be carried to the same number of significant positions as the argument value.

Mnemonic Code	Name	Form
SIN	Sine*	SIN (arg 1)
COS	Cosine*	COS (arg 1)
TAN	Tangent*	TAN (arg 1)

* all values of arguments are expressed in degrees

Examples:

SIN (ANGLE)

COS (RATE (HEALTH, AGE))

TAN (LENGTH 1 - LENGTH 2)

SIN (36.2)

COS (SIN (B))

TAN (WORK ANGLE)

Mnemonic Code	Name	Form
ABS	Absolute	ABS (arg 1)
NEG	Negative The sign of the argument will be reversed. A unary minus.	NEG (arg 1)

Examples:

ABS (QTY ORDERED - QTY DELIVERED)

A + NEG (B)

Mnemonic Code	Name	Form
SUM	Summation Only field names, work field names, literals, reference list function names, and functions may be used.	SUM (arg 1, arg 2, ..., arg n)

Examples:

SUM (A, B, C, D) The result is A+ B+ C + D

SUM (Sin (A), COS (B)) The result is SIN (A) + COS (B)

Other functions may be added later as experience indicates their need and value. For instance, these arithmetic functions are under consideration:

MAX (arg 1, arg 2, ..., arg n)
-- picks the largest value of the arguments.

MIN (arg 1, arg 2, ..., arg n)
-- picks the smallest value of the arguments.

AVG (arg 1, arg 2, ..., arg n)
-- adds the argument values and divides by the number of arguments.

REM (arg 1, arg 2)
-- the remainder after dividing argument 1 by argument 2.

INT (arg 1, arg 2)
-- the integer value from dividing argument 1 by argument 2.

Other logical and symbolic functions are also being examined. Recommendations as to the need for such functions is requested.

CONDITION SUMMARY

Certain operands may not be used together in condition statements. Also, certain of these operand pairs may only be used with the equal and not equal operators. The rules are summarized in Figure 7-11 with examples shown in Figure 7-12.

When the State Relational Operators (IS and NT) are used in a condition statement, one of the operands must be a Named Operand and the other must be a State Term. The permitted combinations are shown in Figure 7-13 with examples shown in Figure 7-14.

Operand Combinations with Value Relational Operators

If Operand 1 is / Operand 2 May Be	Field Name	Value List Name	Arith. Expr. Name	Symb. Expr. Name	Ref. List Func. Name	Const. Name	Work Field Name	Literal	Connot. Term	Desc. Term	Value List	Range
Field Name	X	X*	X	X	X	X	X	X	X	X	X*	X*
Value List Name	X*		X*	X*	X*	X*	X*	X*	X*	X*		
Arithmetic Expression Name	X	X*	X	X	X	X	X	X	X	X	X*	X*
Symbolic Expression Name	X	X*	X	X	X	X	X	X	X	X	X*	X*
Reference List Function Name	X	X*	X	X	X	X	X	X	X	X	X*	X*
Constant Name	X	X*	X	X	X		X			X		
Work Field Name	X	X*	X	X	X	X	X	X	X	X	X*	X*
Literal	X	X*	X	X	X		X			X		
Connotative Term	X	X*	X	X	X		X			X		
Descriptive Term	X	X*	X	X	X	X	X	X	X	X	X*	X*
Value List	X*		X*	X*	X*		X*			X*		
Range	X*		X*	X*	X*		X*			X*		

Note: X means any operator permitted.
X* means only the EQ and NE Value Relational Operators may be used.
Blank means that the operand pair is not legitimate.

Figure 7-11

SYSTEM NAME	PROCEDURE NAME	IBM® DECISION TABLE	ANALYST	DATE	PAGE OF PAGES

TABLE ID / SER	TABLE NAME / OPER OPERAND 1	TABLE TYPE / OP OPERAND 2	FREQUENCY: OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
			01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32
** 001	CITY CODE	EQ 04 06 09 13–19	Y N			
002	CITY CODE	EQ	13 – 19	27 – 30	41 – 57	
003	OPEN BALANCE		GR + 17 50	EQ + 17 50	LR + 17 50	
004	EMPLOYEE NAME	GR	"DAW"	"GLO"	"JOH"	
005	EMPLOYEE NAME	LE	"GLO"	"JOH"	"MIN"	
006	CUSTOMER IDENT	NE BLANK	Y N			
007	DATE REQUESTED: CUSTOMER ORDER	GR # CURRENT DATE #	Y . . N			
008	WORK ABC	LR OVERTIME PAY (JOB CLASS, HOURS WORKED)	N N . . Y			
009	FORMULA 172		LR 1275	.	LR 1275	
010	STRING HEADER		EQ "ABC-257"	EQ "BRL-373"	.	
011	DIAMETER: REQUIREMENT SPECIFICATION	LR PI	Y N			
012	BOARDING CITY	EQ SABER CITY	N N . Y			

REMARKS:

Figure 7-12

Operands with State Relational Operators

State Terms / Operand Names	POSITIVE	NEGATIVE	NUMERIC	ALPHABETIC	PRESENT	END OF FILE (EOF)	STATUS
File Name					X*	X**	
Field Name	X	X	X	X			
Reference List Function Name	X	X	X	X	X		
Arithmetic Expression Name	X	X					
Symbolic Expression Name	X	X	X	X			
Work Field Name	X	X	X	X			
Logical Expression Name							X

NOTE: In general, State Terms may be either operand 1 or operand 2. Readability is improved if the State Terms, except STATUS, are operand 2. STATUS reads best as operand 1.

X means the combination is permitted
blank means the combination is not permitted

* random only
** serial only

Figure 7-13

SYSTEM NAME	PROCEDURE NAME	IBM DECISION TABLE	ANALYST	DATE	PAGE OF PAGES

TABLE ID	TABLE NAME	TABLE TYPE	FREQUENCY			
SER	OPER OPERAND 1	OP OPERAND 2	OP 01 OPERAND 2	OP 02 OPERAND 2	OP 03 OPERAND 2	OP 04 OPERAND 2
			01 02 03 04 05 06 07 08	09 10 11 12 13 14 15 16	17 18 19 20 21 22 23 24	25 26 27 28 29 30 31 32
** 001	INVENTORY	IS PRESENT	Y N			
002	QUANTITY ON HAND	IS	POSITIVE	NEGATIVE		
003	IDENT CODE: MATERIAL LIST		IS NUMERIC	IS ALPHABETIC	•	
004	STATUS	IS	MARRIED	SINGLE	DIVORCED	
005	STATUS	IS SWITCH A ON	Y - N			
006	TRANSACTION	IS EOF	Y N			
007	WITHDRAWAL		IS EOF	NT EOF		

REMARKS:

Figure 7-14

File Conditions

There are two permissable condition statements that may contain a File Name as one of the operands. One statement pertains exclusively to serial type files, while the other pertains exclusively to random type files.

Serial Files - It is presumed that a serial file has a detectable end beyond which data is not recorded. The statement used to detect this condition has the following appearance:

file name { IS / NT } END OF FILE

The IS condition is true if the last record in the file has already been read. The NT condition is true if the last record in the file has not yet been read.

Random Files - Because of the characteristics of random files, it is sometimes necessary to determine if specific data is recorded in the file. The statement used to make this determination may have the following forms:

file name { IS / NT } PRESENT
PER logical expression name

file name { IS / NT } PRESENT
PER location name

file name { IS / NT } PRESENT
PER logical expression name
PER location name

The IS condition of the first form is true if any location is found whose

values satisfy the specified logical expression. If no location is found, the NT condition is true. The IS condition of the second form is true if the specified location exists, otherwise the NT condition is true.

The logical expression may be any logical expression permissable in TDL. The location name may be:

Field Name

Reference List Function Name

Arithmetic Expression Name

Symbolic Expression Name

Constant Name

Work Field Name

Literal

ACTION SUMMARY

The following action operators are used in TDL:

SET	WRITE
MOVE	BEGIN
DO	CLEAR
GO TO	SORT
READ	

With SET and MOVE only certain operands can be used. These are listed in Figure 7-15.

Operands with Assignment Operators

MOVE AND SET				
Sending Operand	Receiving Operand			
	Record Name	Field Name	Work Record Name	Work Field Name
Record Name	*		*	
Field Name		X		X
Reference List Function Name		X		X
Arithmetic Expression Name		X		X
Symbolic Expression Name		X		X
Constant Name		X		X
Work Record Name	*		*	
Work Field Name		X		X
Literal		X		X
Connotative Term	X	X	X	X
Descriptive Term		X		X

X = Individual value assignment.
* = Corresponding value assignment.

NOTE: With MOVE, operand 1 is sending operand, operand 2 is receiving operand.

With SET, operand 2 is sending operand, operand 1 is receiving operand.

Figure 7-15

DO and GO TO are used only with a Decision Table Name or identification numbers, or with a Procedure Name.

The use of input/output operators is summarized in Figure 7-16.

Input Output Operators

File Type	Serial				Random			
Directional Type	Input	Output	Inter.	Ret'd.	Input	Output	Inter.	Ret'd.
Sequence Controlled by File Description Sheet	X			X				
Number of Records Available in Memory	1	(a)	(b)	1	1	(a)	1	1
Record is Deleted By				(c)				(d)
SORT		X	X					
READ	X		X	X	X		X	X
WRITE - Record Name		X	X	X		X	X	X
WRITE - File Name				X			X	X
WRITE - Record Name ON File Name				X			X	X
CLEAR							X	X
BEGIN	X		X	X				

(a) 1 of each type

(b) 1 for READ, 1 of each type for WRITE

(c) READ no WRITE

(d) CLEAR

Figure 7-16