CA Datacom®/DB

Reporting Facility Guide

Version 14.02

ca®
technologies

CA Technologies Product References

This document references the following CA products:

- CA Datacom®/DB
- CA Datacom® CICS Services
- CA Datacom® Datadictionary™
- CA Datacom® SQL
- CA Datacom® VSAM Transparency
- CA Dataquery™ for CA Datacom® (CA Dataquery)
- CA Common Services for z/OS

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: Introduction

The CA Datacom/DB Reporting Facility is a versatile, easy to use, file retrieval and report generating system that allows you to access and report on information contained in CA Datacom/DB databases and in conventional files.

Reporting Facility Features

A part of CA Datacom/DB, the Reporting Facility operates as a batch job under the control of either a z/OS or z/VSE operating system. The free-form Reporting Facility language eliminates the need to learn a complex computer language and the requirements of entering words in a specific column. Designed for end-user efficiency, the English-like commands allow for efficient execution of reporting tasks.

In addition to generating reports, the Reporting Facility has the ability to create output files and can be used as a file conversion tool. Reporting Facility features include the following:

- Input to the Reporting Facility requires no forms. The commands are easy to use, yet the system is compatible with other high-level computer languages such as COBOL and PL/I.
- The system handles all data manipulation, sorting, and report cosmetics without programmer intervention.
- You establish the contents of the report and the Reporting Facility does the rest.
- You can override the Reporting Facility automatic formatting to customize a report.
- You can print fields defined as SQL TIMESTAMP using the Timestamp DEF statement.
- The Reporting Facility supplies full logic capability for selectively retrieving records and printing data.
- The Reporting Facility provides full arithmetic computation capability and data movement and alignment.
- The system handles all report headings, dating, and page numbering automatically.
- The system provides the ability to access and relate multiple input files in a run.
- By coding a single program, up to 63 individual reports can be generated in a single pass of the input data.
- The Reporting Facility provides a library, with full maintenance capabilities, for storage of frequently used input statements.

- You can direct output to a DISK file instead of generating a report.
- The Reporting Facility executes at I/O speed.
- The Reporting Facility is fully integrated into the complete CA Datacom product line and can accept input from CA Datacom/DB, CA Datacom Datadictionary, or CA Dataquery. As part of the CA Datacom environment, the system uses the power of CA Datacom/DB design to provide either random or sequential processing of data.

In your Reporting Facility program, you can specify the content and characteristics of a report, the operations you want to take place on the data, and which data you want represented in the report. There are numerous commands available that allow you to tailor a report to your particular needs. The following commands are common to most Reporting Facility programs:

USER

Identifies the heading to be used on all pages

FILE

Defines the input file

DEFINE

Defines the input record fields

REPORT

Specifies each report within a run

SELECT

Specifies which records you wish to select for the report

PRINT

Specifies the format of the detail lines

END

Ends the Reporting Facility program

The Reporting Facility provides integration with both CA and non-CA software, and is therefore useful in the Application Development Center and the Information Center. The Reporting Facility is flexible and not limited to CA Datacom/DB tables for input data. It can access data from many sources, including IMS-DL/I databases, VSAM files, or standard sequential files. It can also utilize user-defined access methods.

Modes of Operation

The Reporting Facility uses four specialized modes of operation:

- Primary
- Secondary
- Write-Only
- Library Maintenance

You can enter the operating modes either *explicitly*, by the commands or options you select, or *implicitly*, by a decision automatically made when certain conditions are recognized.

Once a particular mode of operation has been selected, it remains in effect for the duration of the run. You cannot mix operating modes in a single run.

Both the Primary and Secondary modes of operation are designed for tasks related to generating a report.

Primary

The Primary mode provides all of the facilities of report generation in a single run. It is the most frequently used mode of operation and is the best suited for generalized reporting functions. In Primary mode, you can produce multiple reports with a single pass of an input data file. Each report can be unique in the sequence of data and the selection of the data. Primary mode is very powerful, but it requires the most resources.

Secondary

Secondary mode provides the facilities for producing a single report. This is the most efficient of the report generating modes and should be used whenever possible because it uses minimum resources. You can enter Secondary mode, which has no sort or control cards, by specifying OPTION SORT=NONE in your Reporting Facility program. The Reporting Facility can choose to enter Secondary mode itself if it determines that you are generating a report in the same sequence as the primary input file.

Write-Only

Use the Write-Only mode to create a sequential disk file as output instead of generating a report.

Library Maintenance

The Library Maintenance mode enables you to catalog frequently used input statements or groups of statements for later use.

For more information about operating modes, see Reporting Facility Modes of Operation (see page 105).

Error Handling

When error conditions occur during a Reporting Facility run, it is likely that one of the following basic problems is responsible:

- Invalid use of the facilities, commands, or parameters through human error (for example, keypunch errors or miscoded statements in the source program)
- General misunderstanding and misuse of the basic concepts and facilities provided by the Reporting Facility
- Improper Job Control Language (JCL) specification
- Environmental errors (such as unedited input data or insufficient core storage)

The Reporting Facility documents each error as it occurs and provides tools for solving complicated problems. When errors occur, see the *CA Datacom/DB Message Reference Guide* for solutions.

The functions that the Reporting Facility performs while processing a request are *compilation* and *execution*. The input commands are processed sequentially and compiled. Then, when it is possible to continue, the system performs the processes it was instructed to execute. During compilation, the Reporting Facility generates diagnostic messages as a result of comprehensive syntax and lexical analysis performed on the input source statements.

In most cases, the Reporting Facility discovers and prints an error message at the point in the program where it occurs. However, more complex error conditions may not be discovered until the entire source statement, or even the entire source program, has been analyzed. In this situation, the message may not print next to the error.

A complete summary of the total number of errors or warnings issued prints at the end of the compilation phase. Even if OPTION LIST OFF is specified, all compiler diagnostic messages print.

After the compilation phase is completed and free of errors, the execution phase can begin. If an abend occurs during the execution phase, a message prints and the run terminates.

In some circumstances, the Reporting Facility can correct an error and continue processing. For example, if a numeric field does not contain valid numeric data, a program check interruption and an abend normally occurs. However, the Reporting Facility can fix the numeric field and continue processing.

Disclaimer

The sample code, JCL, and reports provided in this guide are intended for use as reference aids only. No warranty of any kind is made as to the completeness or correctness of the exact samples in your specific installation environment. If you are planning to use any of the samples provided in this guide, be sure to adjust them for your site standards and use.

Terminology

Some terms in this manual are being used in a special context.

z/OS

The term *region* is used throughout this manual to represent *address space* for z/OS environments.

z/VSE

The term *region* is used throughout this manual to represent *partition* for z/VSE environments. A *DDname* equates to a *filename* and a *data set* equates to a *file-ID*.

Sample Report Headers

The report headers for the sample reports contained in this guide are shown here. Report headers have the following format:

```
Date: mm/dd/ccyy    *******************************************************************************        Page:     n
                    *                                  CA Datacom/DB                              *
Time: hh.mm.ss      *                                 General Utility                             *      Version: nn.n
                    *                   Copyright © 1990-2011 CA. All rights reserved.            *
Base:     dbid      ******************************************************************************* Directory:    name
```

Base:

The *dbid* is the DATACOM-ID (DBID) of the database (base) in use when the report was executed.

Note: Base does not appear in the header if it is not appropriate for the report that was generated or not known at the time the report is produced.

Date:

The date when the report was executed is shown in the format *mm/dd/ccyy*:

mm

month

dd

day

cc

century

yy

year

Directory:

The *name* is the internal name of the Directory (CXX), assigned with the INIT CXX function that was in use when the report was executed and if known at the time the report was produced.

Note: Directory does not appear in the header if it is not appropriate for the report that was generated.

Page:

The *n* is the page number of the report.

Time:

The time when the report was assembled is shown in the format *hh.mm.ss*:

hh

hour

mm

minutes

ss

seconds

Version:

The version of CA Datacom/DB being executed when the report was executed is shown in the format *nn.n,* for example, Version: 14.0.

Reading Syntax Diagrams

Syntax diagrams are used to illustrate the format of statements and some basic language elements. Read syntax diagrams from left to right and top to bottom.

The following terminology, symbols, and concepts are used in syntax diagrams:

- Keywords appear in uppercase letters, for example, COMMAND or PARM. These words must be entered exactly as shown.
- Variables appear in italicized lowercase letters, for example, *variable*.
- Required keywords and variables appear on a main line.
- Optional keywords and variables appear below a main line.
- Default keywords and variables appear above a main line.
- Double arrowheads pointing to the right indicate the beginning of a statement.
- Double arrowheads pointing to each other indicate the end of a statement.
- Single arrowheads pointing to the right indicate a portion of a statement, or that the statement continues in another diagram.
- Punctuation marks or arithmetic symbols that are shown with a keyword or variable must be entered as part of the statement or command. Punctuation marks and arithmetic symbols can include the following:
  - addition (+)
  - comma (,)
  - division (/)
  - equal sign (=)
  - greater than symbol (>)
  - less than symbol (<)
  - multiplication (*)
  - not sign (¬)
  - parenthesis, close ())
  - parenthesis, open (()
  - period (.)
  - subtraction (-)

Statement Without Parameters

The following is a diagram of a statement without parameters:

```
▶▶─ COMMAND ──────────────────────────────────────────────────────────▶◀
```

For this statement, you must write the following:

```
COMMAND
```

Statement with Required Parameters

Required parameters appear on the same horizontal line, the main path of the diagram, as the command or statement. The parameters must be separated by one or more blanks.

The following is a diagram of a statement with required parameters:

```
▶▶───COMMAND──PARM1──PARM2──────────────────────────────────────▶◀
```

If the word variable is a valid entry, you must write the following:

```
COMMAND (PARM1) PARM2='variable'
```

Delimiters Around Parameters

Delimiters, such as parentheses, around parameters or clauses must be included.

```
▶▶─ COMMAND ─ (PARM1) ─ PARM2='variable' ─────────────────────────▶◀
```

If the word variable is a valid entry, you must write the following:

```
COMMAND (PARM1) PARM2='variable'
```

Choice of Required Parameters

When you see a vertical list of parameters as shown in the following example, you must choose one of the parameters. This indicates that one entry is required, and only one of the displayed parameters is allowed in the statement.

```
▶▶─ COMMAND ─┬─ PARM1 ─┬──────────────────────────────────────────▶◀
             ├─ PARM2 ─┤
             └─ PARM3 ─┘
```

You can choose one of the parameters from the vertical list, such as in the following examples:

```
COMMAND PARM1
COMMAND PARM2
COMMAND PARM3
```

Default Value for a Required Parameter

When a required parameter in a syntax diagram has a default value, the default value appears on the main line with a left-facing arrowhead, and it indicates the value for the parameter if the command is not specified. If you specify the command, you must code the parameter and specify one of the displayed values.

```
►►─ COMMAND ─ PARM1= ─┬─ YES ◄ ─┬─ PARM2 ──────────────────────────────►◄
                      └─ NO ────┘
```

If you specify the command, you must write one of the following:

```
COMMAND PARM1=NO PARM2
COMMAND PARM1=YES PARM2
```

Optional Parameter

A single optional parameter appears below the horizontal line that marks the main path.

```
►►─ COMMAND ─┬───────────────┬─────────────────────────────────────────►◄
             └─ PARAMETER ───┘
```

You can choose (or not) to use the optional parameter, as shown in the following examples:

```
COMMAND
COMMAND PARAMETER
```

Repeatable Variable Parameter

In some statements, you can specify a single parameter more than once. A repeat symbol indicates that you can specify multiple parameters.

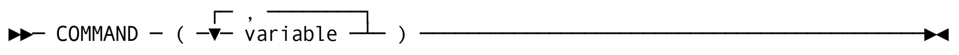
```
             ┌──────────────┐
►►─ COMMAND ─▼─ variable ───┴──────────────────────────────────────────►◄
```

In the preceding diagram, the word variable is in lowercase italics, indicating that it is a value you supply, but it is also on the main path, which means that you are required to specify at least one entry. The repeat symbol indicates that you can specify a parameter more than once. Assume that you have three values named VALUEX, VALUEY, and VALUEZ for the variable. The following are some of the statements you might write:

```
COMMAND VALUEX
COMMAND VALUEX VALUEY
COMMAND VALUEX VALUEX VALUEZ
```

Separator with Repeatable Variable and Delimiter

If the repeat symbol contains punctuation such as a comma, you must separate multiple parameters with the punctuation. The following diagram includes the repeat symbol, a comma, and parentheses:

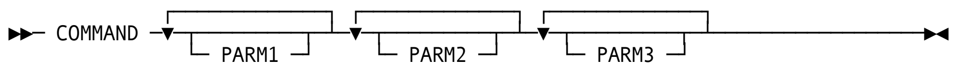


In the preceding diagram, the word variable is in lowercase italics, indicating that it is a value you supply. It is also on the main path, which means that you must specify at least one entry. The repeat symbol indicates that you can specify more than one variable and that you must separate the entries with commas. The parentheses indicate that the group of entries must be enclosed within parentheses. Assume that you have three values named VALUEA, VALUEB, and VALUEC for the variable. The following are some of the statements you can write:

```
COMMAND (VALUEC)
COMMAND (VALUEB,VALUEC)
COMMAND (VALUEB,VALUEA)
COMMAND (VALUEA,VALUEB,VALUEC)
```

Optional Repeatable Parameters

The following diagram shows a list of parameters with the repeat symbol:

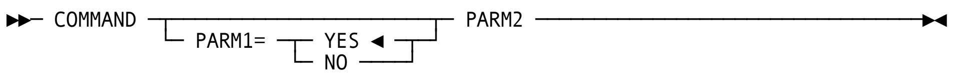


The following are some of the statements you can write:

```
COMMAND PARM1
COMMAND PARM1 PARM2 PARM3
COMMAND PARM1 PARM1 PARM3
```

Default Value for a Parameter

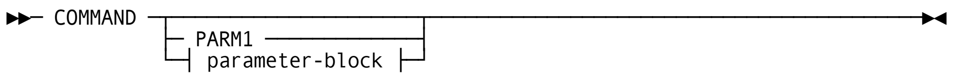
The placement of YES in the following diagram indicates that it is the default value for the parameter. If you do not include the parameter when you write the statement, the result is the same as if you had actually specified the parameter with the default value.



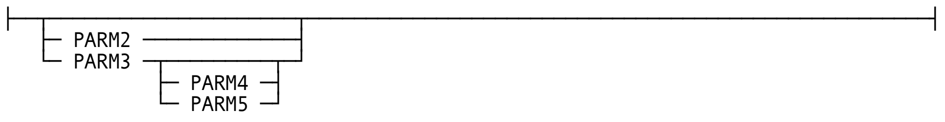
For this command, COMMAND PARM2 is the equivalent of COMMAND PARM1=YES PARM2.

Variables Representing Several Parameters

In some syntax diagrams, a set of several parameters is represented by a single reference.



Expansion of parameter-block



The *parameter-block* can be displayed in a separate syntax diagram.

Choices you can make from this syntax diagram therefore include, but are not limited to, the following:

```
COMMAND PARM1
COMMAND PARM3
COMMAND PARM3 PARM4
```

Note: Before you can specify PARM4 or PARM5 in this command, you must specify PARM3.

JCL Requirements

Guidelines to assist you in preparing your JCL are provided in this manual. The sample code provided in this document is intended for use as a reference aid only and no warranty of any kind is made as to completeness or correctness for your specific installation.

Samples for JCL and programs are provided in the install library (in z/OS, the default name for this library is CABDMAC). In z/VSE, sample PROCs are provided that allow you to make use of parameter substitution. You can copy and modify these samples for your specific requirements.

Any JOB statements should be coded to your site standards and specifications. All data set names and library names should be specified with the correct names for the installation at your site. In many examples, a REGION= or SIZE= parameter is displayed in an EXEC statement. The value displayed should be adequate in most instances, but you can adjust the value to your specific needs.

The libraries listed for searching must include the following in the order shown:

1. User libraries you may have defined for specially assembled and linked tables, such as DBMSTLST, DBSIDPR, DDSRTLM, DQSYSTBL, or User Requirements Tables (CUSLIB)
2. CA Datacom base libraries: CA Datacom/DB, CA Datacom Datadictionary, CA Dataquery, SQL
3. CA IPC libraries
4. CA Common Services (formerly known as CA Common Services for z/OS) base libraries
5. Libraries for additional products, such as CA Datacom CICS Services, CA Datacom VSAM Transparency, CA Ideal, and so on

CA Dataquery users also need the following libraries and data sets for the following specific functions:

- The z/OS data set DQOUT or the z/VSE data set DQOUTD is used only if the DQBATCH execution uses the EXPORT function.
- In z/OS, running deferred queries with separate JCL members in batch requires, in addition to the SYSIN statement DEFER, the inclusion of a DD statement for the internal reader used by VPE. This DD statement should be:

  ```
  //IRDR DD    SYSOUT=(A,INTRDR)
  ```

Chapter 2: Components of a Reporting Facility Program

In coding a CA Datacom/DB Reporting Facility program, the functions you need to perform determine the sequence in which you submit the output source statements. Therefore, you must submit the commands in a specific sequence.

The Reporting Facility program is divided into seven distinct functional areas:

- Run Control
- Input File Definition
- General Storage Area (GSA) Definition
- Data Manipulation/File Reading
- Report Definition
- Output File Definition
- Library Maintenance

Some of the functional groups of commands are unique to a particular mode of operation, while some are common to all types of programs.

Command Summary

Certain commands apply to each area of the seven functional areas in a Reporting Facility program. A general description of each type of Reporting Facility command is provided next. For a more information about each command, see Reporting Facility Commands (see page 147).

Run Control

Run Control commands set the general parameters used throughout the run to direct the compilation processes.

CALL

Instructs the Reporting Facility compiler to retrieve prestored input statements from the Reporting Facility call library

EJECT

Controls the physical spacing of the compiler SKIP1,2,3 listing

END

Terminates the source input stream

NOTE

Enables you to insert comments into the source program

OPTION

Overrides, for a single processing run, the default options specified at system installation time, or provides parameters which influence the mode of operation

USER

Provides for generating a report identification line as the first line of every report

Input File Definition

Input File Definition commands describe the physical characteristics and attributes of the data files input to the Reporting Facility program.

FILE <INPUT>

Identifies and describes an input data file and its physical characteristics and attributes

COPYDD

Instructs the Reporting Facility compiler to retrieve the record/element structure from CA Datacom Datadictionary

DEFINE

Defines a single addressable unit, scalar variable, or array element that resides in the data record of the previously defined file

Note the following:

- An *array* is a scalar variable that occurs multiple times, in contiguous locations, within the GSA or the input record area.
- A *scalar variable* is a single data occurrence located either in the General Storage Area (GSA) or the input record area. You must assign attributes describing the type of occurrence and give it a location referencing the field name associated with it. Scalar variables are the smallest addressable unit in a Reporting Facility program.

General Storage Area Definition

The General Storage Area (GSA) has a single command which the system creates automatically. It is basically a scratch pad used to define standard literals or tables of data.

DEFINE

Defines a single addressable unit, scalar variable, or array element that resides in the GSA

The GSA can be equated to a COBOL Working Storage section. The system also uses the GSA for immediate storage of special work areas during compilation.

Data Manipulation and File Reading

Data Manipulation and File Reading commands enable you to access input files and define operations to be performed on the data prior to output selection. Many functions can be performed when coding this section of the program, including data movement, arithmetic calculations, data translation or decoding, input file access, and direction of the basic program flow. The Data Manipulation/File Reading commands are: ADD, COMPUTE, CONT, CONTINUE, DECODE, DECR, DECREMENT, DIVIDE, ENDPROC, GET, GOTO, INCR, ,INCREMENT, MOVE, MULTIPLY, PERFOR, PROC, REDUCE, SET, SUBTRACT, and UP.

Report Definition

Report Definition commands describe the content, sequence, and format of each report produced in a single run.

REPORT

Identifies and describes a single report to be generated

SELECT

Conditionally selects a record, or set of records, to be included in an individual report

SET/COMPUTE

Directs specialized data movement or arithmetic calculations unique to a single report

CONTROL

Specifies the sequence in which the data is to be presented in the report, or those fields to be designated as control break fields during report printing

PRINT

Defines the format of a single line to appear on the output report

Output File Definition

These commands enable you to select the content and format of the output. You must code the output file definition (Write-Only mode, as the last area in the Reporting Facility program). Use the FORMAT command to specify the contents of each logical record to be created. Two basic commands direct output file definition:

SELECT

Conditionally selects a record or set of records to be included as part of the output data set

FORMAT

Defines the format of each output record to be created

Library Maintenance

Library Maintenance commands direct processing when maintenance is performed on the Reporting Facility call library. You cannot perform Library Maintenance in conjunction with report generation or output file creation. The invocation of Library Maintenance mode using the run control options area dedicates the entire run to performing library functions. The Library Maintenance commands are:

CONDENSE

Initiates reorganization of the Reporting Facility call library

CREATE

Initiates the preformatting procedures of the physical library extent prior to any use of the library

DELETE

Deletes a specific member from the library

DISPLAY

Displays specified contents of the library, including indexes or specific members

LIBRARY

Instructs the Reporting Facility to enter the Library Maintenance mode

LOAD

Identifies a specific member to be placed in the library for later retrieval

The remainder of this chapter discusses the seven functional areas in more detail.

Run Control

Run Control commands can appear anywhere within the Reporting Facility program and have the following restrictions:

- When generating a report in Primary or Secondary modes, the USER command must be either the first command in the input stream or follow the first OPTION command in the input stream.

  Note: The USER command is not required when operating in the Write-Only or Library Maintenance modes.

- Compiler control commands (EJECT, SKIP1,2,3, and NOTE) can appear anywhere in the input stream *except* in Library Maintenance mode.
- The OPTION command can appear anywhere in the program, in any operating mode. It is recommended, however, that you place it at the beginning of the input stream.
- The CALL command can appear anywhere in the program, regardless of operating mode. In Library Maintenance mode, however, the CALL command is recognized only when it is part of a specific member to be cataloged.

Run Control commands enable you to analyze the external requirements of a particular program and establish the working environment as soon as possible during the program development cycle. This reduces the programming effort and produces mode efficient programs.

The following is the basic diagram of the Run Control commands within a program:

1	**Run Control**
2A	Input File Definition
2B	GSA Definition
3	Data Manipulation/File Reading
4A,4B	Report Definition/Output File Definition

Input File Definition

Input File Definition commands appear immediately after the run control area in the program. The FILE, DEFINE, or COPYDD commands define each program input file.

The first file defined in any run is known as the primary file. The system automatically generates the commands to access the primary file unless you override them or the file is a CA Datacom/DB table or IMS-DL/I file.

You must explicitly access secondary files, but you can define and access any number of input data sets in a single run. When defining fields that reside in the input records, define only those fields required for processing.

The following is a basic diagram of the Input File Definition commands within a program:

1	Run Control
2A	**Input File Definition**
2B	GSA Definition
3	Data Manipulation/File Reading
4A,4B	Report Definition/Output File Definition

General Storage Area (GSA) Definition

GSA Definition commands appear immediately following all input file definition commands to ensure program continuity.

Use the GSA to define standard literals or tables of data for the program. Size of the GSA is limited only by the amount of core storage available. The Reporting Facility can also use this area internally for intermediate storage of work areas during compilation.

The following is a basic diagram of the GSA definition commands within a program:

1	Run Control
2A	Input File Definition
2B	**GSA Definition**
3	Data Manipulation/File Reading
4A,4B	Report Definition/Output File Definition

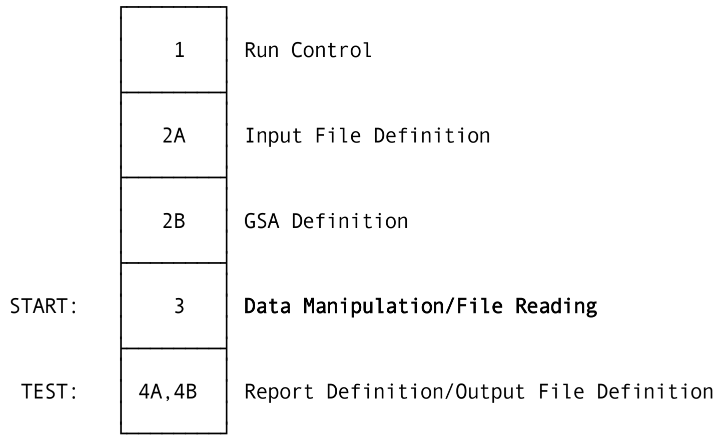
Data Manipulation/File Reading

Data Manipulation and File Reading commands must appear after the Input File Definition or GSA Definition commands, but before the Report Specification or Output File Definition commands. This section of the program is responsible for all activity to be performed prior to a sort, the printing of a single line on the report, or the output of a single record.

The Reporting Facility recognizes the first command in this area as the *first procedural keyword*. The system works in a cyclical manner when operating in Primary, Secondary, or Write Only modes. In other words, when a complete input record set must be made available, you can either ignore the set completely, or pass it through the selection criteria specified in the SELECT commands in either the report definition groups or the subsequent output file definition section.

The predefined label START is logically attached to the first procedural keyword in this section of the program. A branch to START ignores the current record set and reads a new record set *without* passing the record set through the SELECT criteria. Another predefined label, TEST, is automatically assigned by the system and is logically attached to the first SELECT command. Encountering a branch to TEST in the program indicates that the current record set will be tested against the SELECT criteria. When the system completes its analysis of the record set, it takes an automatic branch to START to begin reading the next record set. The combination of these two predefined labels provides multiple options for selecting input data.

The following is a basic diagram of the Data Manipulation and File Reading commands within a program:


1
Run Control
2A
Input File Definition
2B
GSA Definition
START:
3
Data Manipulation/File Reading
TEST:
4A,4B
Report Definition/Output File Definition

Report Definition

Report Definition commands must appear as the final group of commands in the input stream. In Primary mode, you can code up to 63 different report definitions. In Secondary mode, you can code only one report definition. Each definition describes the content and format of an individual report.

The following is a basic diagram of the Report Definition commands within a program:

	1	Run Control
	2A	Input File Definition
	2B	GSA Definition
START:	3	Data Manipulation/File Reading
TEST:	4A	**Report Definition**

Output File Definition

The Output File Definition is the last section in the Reporting Facility program. You can define a single file in a particular run. Use the FORMAT command to specify the contents of each logical record to be created.

The following is a basic diagram of the Output File Definition commands within a program:

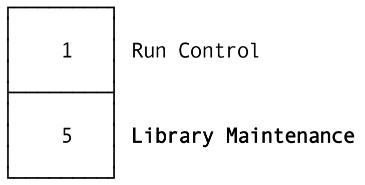
	1	Run Control
	2A	Input File Definition
	2B	GSA Definition
START:	3	Data Manipulation/File Reading
TEST:	4B	**Output File Definition**

Library Maintenance

You cannot perform Library Maintenance in conjunction with report generation or output file creation.

Using the Run Options section to invoke Library Maintenance mode dedicates the entire run to performing library functions.

The following is a basic diagram of the Library Maintenance commands within a program:



Chapter 3: Reporting Facility Rules for Coding

While no coding forms are required, there are rules you must follow to code a Reporting Facility program.

Rules for coding the following are discussed in this chapter:

- Reporting Facility programs
- Character set
- Reserved words
- Predefined field names
- Alphanumeric literals
- Numeric constants
- Hexadecimal literals
- Bit masks
- User-defined field names
- Labels
- File names
- Logical expressions
- Arithmetic expressions
- Call library member names
- Data format
- Data movement and alignment
- Data editing

Reporting Facility Programs

A Reporting Facility statement consists of an English-like command and its associated qualifying parameters. Each statement specifies a particular function or operation to be performed. Enter the statements free-form in card-image format. You may separate each entry with any number of spaces.

If you code totally blank lines, the Reporting Facility ignores them. You can overflow the program statements onto any number of consecutive input lines, but the following rules must apply:

- Only columns 1—72 of any input line can contain source information. The system uses columns 73—80 for sequence numbers.
- Command words, field names, numeric constants, and literals enclosed in apostrophes must be specified in one statement and cannot span from one card-image to another.

To improve program readability, use multiple input statements when coding individual commands and their associated parameters.

For example, the command

```
DECODE DD INTO ALPHA-DAY 1='MON' 2='TUE' 3='WED' 4='THU'
                         5='FRI' 6='SAT' ELSE 'SUN'
```

could be coded as follows to improve readability and appearance of the program:

```
DECODE DD INTO ALPHA-DAY 1='MON'
                         2='TUE'
                         3='WED'
                         4='THU'
                         5='FRI'
                         6='SAT'
                         ELSE 'SUN'
```

The Reporting Facility is parameter driven and commands are interpreted sequentially. Therefore, the sequence in which you present the commands to the Reporting Facility is very important.

Character Set

The following characters from a standard keyboard are recognized by the Reporting Facility in commands and parameters:

0—9

Specifies digits

A—Z

Specifies uppercase letters

+

Specifies a plus sign

–

Specifies a minus sign, dash, or hyphen

_

Specifies an underscore

(space)

Specifies a blank, or space

,

Specifies a comma

.

Specifies a period or decimal

>

Specifies greater than

<

Specifies less than

&

Specifies an ampersand

'

Specifies an apostrophe or single quote

;

Specifies a semicolon

*****

Specifies an asterisk

/

Specifies a slash

=

Specifies an equal sign

:

Specifies a colon

#

Specifies a hash sign

(

Specifies a left parenthesis

)

Specifies a right parenthesis

@

Specifies an at sign

$

Specifies a dollar sign

These characters have special meanings when used in an arithmetic expression:

+

Specifies addition

/

Specifies division

*****

Specifies multiplication

-

Subtraction

The following characters take on special meanings when used in logical expressions:

>

Specifies greater than

<

Specifies less than

=

Specifies equal

Note the following:

- In any expression or parameter that allows the use of an = (equal sign), you can replace the equal sign with its alphabetical equivalent, EQ.
- When using Format 1 of the DEFINE command or a range specification in a logical expression, you can replace the - (dash) with its alphabetic equivalent, THRU or TO.

Reserved Words

Since the CA Datacom/DB Reporting Facility uses a command-driven language, the system must reserve words or phrases for use by the compiler. The Reporting Facility recognizes certain words or phrases as requests for specific functions. Do not use any of the following reserved words as user-defined labels, field names, file names, or parameters.

The words in bold type in the following lists are Reporting Facility commands. All other words are parameters or reserved field names and can be used in other ways.

ABORT	COBOL	DBID	DUMP
ADD	COMPILE	DD	DUTCH
ALL	**COMPUTE**	DDDBID	**EJECT**
AND	CONDENSE	**DECODE**	ELSE
ARABIC	**CONT**	**DECR**	**END**
ASM	**CONTINUE**	**DECREMENT**	END-OF-FILE
ASSEMBLER	**CONTROL**	**DEF**	ENDPROC
B	**COPYDD**	**DEFINE**	ENGLISH
PROC	CREATE	DELETE	ENT
BLANK	CURRDATE	DETAIL	ENTITY
BLOCK	CURRLDATE	DISK(S)	EOJ
BLOCKED	CURRENT-DATE	DISPLAY	EQ
BSUB	CURRENT-TIME	**DIVIDE**	EQUAL
BY	CURRTIME	DLI	ESUB
C	D	DOUBLE	EXCLUDE
CALL	DANISH	DROUT	F
CARD	DASH	DRVSAM	**FILE**
CC	DATACOM	DRWORK	FILL
COB	DBCOMM	DT	FINNISH

FIRST	LINES	PICTURE	START
FIXED	LIST	PLI	**SUBTRACT**
FORMAT	LOAD	PL1	SWEDISH
FRENCH	LOW-VALUE(S)	PORTUGUESE	SYSnnn
FROM	LT	**PRINT**	T
DQF	LTE	PRINTER	TAG
GERMAN	MAP	PROD	TAPE
GET	MM	PROG	TEST
GIVING	MN	PRTEXIT	THRU
GO	**MOVE**	PUNCH	TIMES
GOTO	**MULTIPLY**	QSEQ	TO
GE	N	RECORD(S)	TOTAL(S)
GSA	NAME	RECORD-FOUND	TRACE
GT	NE	**REDUCE**	TRIPLE
GTE	NEWPAGE	REPORT	UNLOAD
HDG	NO	REWIND	**UP**
HEADING	NO-RECORD-FOUN	ROUND	USE
HIGH-VALUE(S)	D	**SELECT**	**USER**
HIST	NONE	SEQUENTIAL	USING
HR	NORWEGIAN	**SET**	VAL
INCR	NOT	SIGN	VALUE
INCREMENT	**NOTE**	SKIP	VARIABLE
INDEX	OCCURS	**SKIP1**	VSAM
INDEXED	OFF	**SKIP2**	WHEN
INDEXERR	OMIT	**SKIP3**	WHERE
INPUT	ON	SORT	WRITE
INTO	ONLY	SORTIN	X
ITALIAN	**OPTION**	SORTIN1	XC
JAPANESE	OPTPRT	SORTWK01	XD
KEY	OR	SORTWK1	XY
KOREAN	OTHERWISE	SPACE(S)	YES
LABEL	OUTLIM	SPACING	YY
LANG	P	SPANISH	ZERO(S)
LAST	PAGE(S)	SPARE	0
LE	**PERFORM**	SS	
LIBRARY	PIC	STANDARD	

Predefined Field Names

In an effort to standardize certain functions and provide some basic options, the Reporting Facility comes with a set of predefined fields for general use. You can reference these fields as if you had coded the DEFINE commands in the program.

The following fields can be printed, interrogated against, used in arithmetic expressions, accumulated, or used as sort keys.

BLANK

A single alphanumeric character defined with an initial value of X'40'.

CURRDATE

An 8-character alphanumeric field that contains the current date in the format MM/DD/YY. This field is updated only once, at the beginning of the Reporting Facility run.

CURRENT-DATE

A synonym for CURRDATE described previously. It resides in the same location and contains the same data as CURRDATE.

CURRTIME

An 8-character alphanumeric field that contains the current time in the format HH:MM:SS. This field is updated each time a primary input record is read by the Reporting Facility.

Note: If you override the automatic read, it is updated each time the record selection processing cycle begins (for example, GOTO START).

CURRENT-TIME

A synonym for CURRTIME described previously. It resides in the same location and contains the same data as CURRTIME.

MM

Numeric representation of the month (2-character alphanumeric field).

dd

Numeric representation of the day (2-character alphanumeric field).

YY

Numeric representation of the year (2-character alphanumeric field).

hr

A 2-character alphanumeric field. It is the numeric representation of the hour. This field redefines positions 1—2 of the CURRENT-TIME field.

MN

A 2-character alphanumeric field. It is the numeric representation of the minute. It redefines positions 4—5 of the CURRENT-TIME field.

ss

A 2-character alphanumeric field. It is the numeric representation of the second. It redefines positions 7—8 of the CURRENT-TIME field.

END-OF-FILE

A 1-character alphanumeric field with an initial value of E, (X'C5'). It is most commonly used in conjunction with the GET command.

NO-RECORD-FOUND

A 1-character alphanumeric field with an initial value of N, (X'D5'). It is most commonly used in conjunction with the GET command.

RECORD-FOUND

A 1-character alphanumeric field with an initial value of Y, (X'E8'). It is most commonly used in conjunction with the GET command.

QSEQ

A 5-byte packed decimal field that is automatically incremented by 1 each time a primary input record is read. If you override the automatic read, the Reporting Facility updates it each time the record selection processing cycle begins.

QSEQ is defined as containing a maximum of nine integers, with no decimal places. Therefore, the maximum number it can contain is nine 9s (999999999). This field is commonly used:

- To limit the number of primary input records read in order to test the Reporting Facility program before the entire file is searched
- To ensure that when producing multiple reports, the data for a particular report is presented in the same sequence as it is read

SPACE

A single alphanumeric character defined with an initial value of X'40'.

TAG

A 1-character alphanumeric field used internally by the SELECT command for the conditional printing or accumulation of fields within a record.

ZERO

A 1-byte packed decimal field with a value of zero. Use it to initialize fields, or as a first operand in a logical expression.

Alphanumeric Literals

When you code a CA Datacom/DB Reporting Facility program, you may need to establish a constant value by coding an alphanumeric literal. An alphanumeric literal can contain any character defined in the Reporting Facility character set.

Enclose all alphanumeric literals within apostrophes. When apostrophes are included as part of the literal, they must be coded in pairs, each pair generating a single value in the resulting literal.

The Reporting Facility automatically calculates the net length of the literal. Literals are no longer than one line of source code. They cannot be continued to another line. For example, the maximum length of a single literal is 70 characters.

The following table illustrates some rules to follow in coding alphanumeric literals:

Coded String	**Net Length**	**Precision**
'12345'	5	12345
'COMPANY''S'	9	COMPANY'S
'''DALLAS'''	8	'DALLAS'
'''''TEXAS'''''	9	''TEXAS''
''''	1	'
'DALLAS TEXAS'	12	DALLAS TEXAS
'AMOUNT-FIELD'	25	AMOUNT-FIELD
'009.21-'	7	009.21
'A'	1	A
'A B C D E F'	12	A B C D E F
ABC	**	INVALID
'DALLAS	**	INVALID
DALLAS'	**	INVALID

You can code alphanumeric literals on these commands:

CALL	FILE	OPTION
COMPUTE	FORMAT	PRINT
CONTROL	GET	REPORT
DECODE	GOTO	SELECT
DEFINE	MOVE	SET
		USER

Numeric Constants

When writing a Reporting Facility program, you may need to establish a constant numeric value by coding a numeric constant. Numeric constants can be preceded by a sign (+ or -) and must consist of digits in the range 0 to 9. You can code up to 15 digits in any single numeric constant. You can place a decimal point anywhere within the constant.

The constant is assumed to be positive if no sign is specified. The precision and length of the constant are automatically calculated and internally stored in packed decimal format.

The following table illustrates rules for coding numeric constants:

Coded String	Net Length	Precision
123.45	3	(3.2)
+.6234	3	(0.4)
-146	2	(3.0)
-000.0007	4	(3.4)
1234567890	6	(10.0)
98765.43219	6	(5.5)
0	1	(1.0)
-	1	(1.0)
+1	1	(1.0)
1	1	(1.0)
2.00000000	5	(1.8)
+123456789012345	8	(15.0)
.123456789012345	8	(0.15)
-98765432101.33	7	(11.2)

You can code numeric constants on these commands:

ADD	DIVIDE	PRINT
CALL	GET	REPORT
COMPUTE	GOTO	SELECT
DECODE	INCREMENT	SET
DECREMENT	MOVE	SUBTRACT
DEFINE	MULTIPLY	

Hexadecimal Literals

When you need to initialize a field or variable to a hexadecimal value, you can code a hexadecimal literal in the following format:

```
►►─ X'digits' ──────────────────────────────────────────────►◄
```

'digits'

Are a combination of valid hexadecimal digits (0—9, A—F) representing the value of a single byte. You must provide an even number of hexadecimal digits.

Since a hexadecimal literal must be contained on one source line in a Reporting Facility program, the maximum number of bytes to be represented in a single literal is 34.

The following table illustrates uses of hexadecimal literals:

Coded String	Net Length	Resultant Literal
X'00'	1	low-value
X'0123450C'	4	a packed number
X'C1C2C3F1F2F3'	6	ABC123
X'FFFFFFFF'	4	high-values
X'F1F2F3F4'	5	1234
X'40C140C240C340C4'	8	A B C D
X'4040'	2	spaces
X'F0F0F94BF2F160'	7	009.21-
X'505B5C6C7D'	5	&$*%:
X'0G'	*	INVALID
X'000'	*	INVALID
'FF'	*	INVALID
X00FF	*	INVALID

Code hexadecimal literals only on Format 2 DEFINE or MOVE commands.

Bit Masks

In coding a Reporting Facility program, you may need to test the contents of a single byte against a mask. When using a mask in a logical expression, code it in the following format:

```
▶▶─ B'testbits' ──────────────────────────────────────────▶◀
```

'*testbits*'

Specifies the mask to be tested. It is comprised of one to eight 1s, 0s, or Xs. Positioning in the mask corresponds to bits 0—7, from left to right, of the byte to be tested against.

- **1** in the mask means that the corresponding bit in the string will be tested to be ON.
- **0** in the mask means that the corresponding bit in the string will be tested to be OFF.
- **X** in the mask means that the corresponding bit in the string is NOT TO BE TESTED.

Note the following:

- Reference bit masks only in logical expressions on GOTO or SELECT commands. You must define the first operand in the logical expression as a bit string.
- The Reporting Facility compiler requires at least one bit position to be tested in a mask specification, that is, at least one 1 or 0 must be specified in the mask.
- You can never print a bit string field. You can only test it against a mask.
- Compare bit string fields with a bit mask only in an EQUAL or NOT EQUAL relationship.
- Consider the bit mask test to be true if the tested byte conforms to the bit mask pattern; otherwise, a false condition exists.
- The system assumes that bit positions not specified on the right side of the mask are not to be tested.

- The following table shows both valid and invalid bit mask comparisons based on these commands:

  ```
  DEFINE BIT-STRING 27 S
  GOTO START WHEN (BIT-STRING EQ mask)
  ```

Mask in Command	Hex Value	Condition
B'11000000'	X'CO' (11000000)	true
B'11101100'	X'FF' (11111111)	false
B'00000000'	X'00' (00000000)	true
B'11X1000X'	X'F1' (11110001)	true
B'1'	X'82' (10000010)	true
B'X0000000'	X'01' (00000001)	false
B'1110'	X'F0' (11110000)	false
B'X0'	X'40' (01000000)	false
B'00'	X'20' (00100000)	true
B'XXX'	. .	INVALID
B'02'	. .	INVALID
'01'	. .	INVALID
B011	. .	INVALID

User-Defined Field Names

To access data at a specific location within an input record or the General Storage Area (GSA), assign a symbolic name to that location. When referencing storage reserved as part of an input record field, each field name must be defined with the DEFINE command. You do not need to account for filler entries.

Define fields in the GSA in one of the following ways:

- Explicitly, with the DEFINE command
- Implicitly, by the procedural command itself (COMPUTE, SET, or DECODE)

The field name assigned to any specific location must follow these basic rules:

- Field names must begin with a character in the range of A—Z and all other characters must be in the range of A—Z, 0—9, or be a - (dash) or an _ (underscore). The Reporting Facility allows no other special characters in a field name specification.
- The last character of a field name must not be a (-) dash or an _ (underscore).
- The field name cannot exceed 72 characters and is terminated by a blank.

Valid Field Names

- TRANSACTION-CODE
- YEAR-TO-DATE-TAXES-1990
- YTDCOMM
- THIS-IS-ALSO-A-VALID-FIELDNAME-IN-A-PROGRAM

Invalid Field Names

The following entries result in the described error conditions:

123

First character is not A—Z.

FIELD#1

Special character is present.

INPUT

Field name is a Reporting Facility reserved word.

TYPE-CODE-

Last character is a dash.

PAY_DAY_

Last character is an underscore.

Field Types

You can assign field names to two types of fields in a Reporting Facility program.

- Scalar variables
- Arrays

Scalar Variable

A scalar variable is a single data occurrence located in either the General Storage Area (GSA) or the input record area. You must assign attributes describing the type of occurrence, giving it a location that references the field name associated with it. Scalar variables are the smallest addressable unit in a Reporting Facility program.

The primary method of defining a scalar variable is with the DEFINE command, but you can also define scalar variables using the implicit field definition facilities supplied by the SET, COMPUTE, or DECODE commands.

When defining a scalar variable, assign specific attributes describing the type of variable. You must define a scalar variable as one of the following:

- Numeric packed decimal
- Numeric zoned decimal
- Numeric binary
- Alphanumeric
- Bit string

You can assign any of the above attributes to a variable defined in an input record area. Variables defined as residing in the GSA can be defined as being numeric or alphanumeric only.

Reference to the newly assigned field name results in the alternate type of the occurrence being processed. For example, define and initialize a numeric zoned decimal variable in the GSA as shown:

```
DEFINE ALPHANUMERIC-FIELD(7)=' '
DEFINE ZONED-DECIMAL-FIELD EQ ALPHANUMERIC-FIELD 1-7 N
   .
   .
   .
MOVE ZERO TO ZONED-DECIMAL-FIELD
```

Array

An array is a scalar variable that occurs multiple times, in contiguous locations, within the GSA or the input record area. Arrays can be defined to the Reporting Facility only by the DEFINE command.

The same restrictions and attribute options apply to arrays as to scalar variables. However, when an array element is referenced in a procedural statement, you must supply an INDEX value to indicate which occurrence of the array is being referenced. Code an array element reference in the following format:

```
►►─ fieldname(indexval) ──────────────────────────────────────►◄
```

fieldname

Specifies the symbolic name assigned to the array.

indexval

Specifies the element within the array that is to be referenced. You can code it as either:

- an absolute index specification that specifies the element within the array to be referenced
- a relative index specification that contains the index value to identify the specific element within the array to be referenced

Note the following:

- The Reporting Facility supports only single dimensional arrays. The system flags as an error any condition encountered that would require multiple levels of indexing.
- If an array element is fully or partially redefined, the new field is also considered an array and you must index it when it is referenced.
- You can redefine and specify a scalar variable as an array however, the entire calculated length of the array must not exceed the original dimensions of the scalar variable.
- Most of the Reporting Facility commands support indexing an array element, but each command can impose its own special requirements or limitations.
- When you refer to an array element using an absolute index specification, the array element is treated internally as though it were simply a scalar variable.

  When you use a relative index specification, however, the Reporting Facility calculates the address of the desired element each time it is referenced during execution. This can cause the entire array to be written to the internal Hit File and overuse your resources. The Hit File is the file where the Reporting Facility temporarily stores the report as it selects records.

Field Name Qualification

Define scalar variables and arrays as being either within the input record area or the General Storage Area (GSA). Each storage location is associated with a specific input file or a location in the GSA. You may assign duplicate field names if the duplicates are not in the same input file or the GSA.

Reference to the qualified field is in the following format:

Field Name Qualification

```
▶▶─ qualifier.fieldname ─────────────────────────────────────▶◀
```

qualifier

The unique file name associated with the FILE command where the field resides, or GSA if the field resides in the General Storage Area.

fieldname

The symbolic name assigned to the scalar variable or array element to be referenced.

The following demonstrates common uses and benefits of field name qualification:

```
MASTER:  FILE   DATACOM   RECORD EQ 250            File 1 definition
          DEFINE DB-COMMAND        1-5 X
           .
         DEFINE AMOUNT-FIELD 246-250 P4
           .
PAYROLL: FILE   DATACOM   RECORD EQ 100            File 2 definition
         DEFINE DB-COMMAND        1-5 X
           .
         DEFINE AMOUNT-FIELD  96-100 P4
           .
         DEFINE AMOUNT-FIELD(7.4)=0                 GSA Field
           .
         MOVE   'GETIT' TO MASTER.DB-COMMAND        Program body
           .
         MOVE   'REDKG' TO PAYROLL.DB-COMMAND
           .
         COMPUTE GSA.AMOUNT-FIELD EQ
                 (MASTER.AMOUNT-FIELD + PAYROLL.AMOUNT-FIELD
```

Note the following:

- If the compiler encounters a field name reference requiring qualification but a qualifier is not specified, an error results. You can specify a qualifier even when qualification is not required.
- The compiler assigns qualifiers automatically.
- Using field name qualification gives programs continuity and makes them easier to read.

Labels

The Reporting Facility provides the ability to attach a label to certain commands. You can conditionally or unconditionally branch to the commands.

This technique directs the program flow and determines which commands will be executed and under what circumstances they will be executed. For the Reporting Facility to recognize your intentions, you must follow certain label coding rules:

- Apply labels only to these commands:

ADD	DECREMENT	INCREMENT
PROC	DIVIDE	MOVE
COMPUTE	ENDPROC	MULTIPLY
CONTINUE	GET	SET
DECODE	GOTO	SUBTRACT

- Labels must begin with an alphabetic character (A—Z) and all other characters must be in the range of A—Z, 0—9, or be a – (dash) or an _ (underscore). No other special characters are allowed in a label specification.
- The last character of the label must not be a – (dash) or an _ (underscore).
- Labels cannot exceed 71 characters and must be terminated by a colon.
- The Reporting Facility automatically defines four labels that take on special meaning. They are ABORT, EOJ, START, and TEST. You can branch to these labels, but you cannot define them.
- If you use duplicate labels in a single run, an error condition occurs.

Valid Labels

- THIS-IS-A-SAMPLE-OF-A-VALID-LABEL-IN-A-PROGRAM:
- THIS_IS_TOO:
- THISISTOO:
- PROG123:
- ABC:

Invalid Labels

The following entries result in the described error conditions:

123:

First character is not A—Z.

THIS-IS-AN-INVALID-LABEL

Terminating colon is missing.

SO_THIS_IS_#1:

Special character is present.

INVALID-LABEL-SPEC-:

Last character before the colon is a dash.

START:

Predefined, reserved label.

File Names

When defining input data files or tables to a Reporting Facility program using the FILE command, you must assign a unique name to each file or table. This name has two functions:

- To identify the file or table that will be processed
- To provide a means to tie a specific file or table name to JCL in the Reporting Facility execution stream that defines the physical characteristics of the file or table to the host operating system

Follow these rules when coding input file or table names:

- File names can be attached only to FILE or INPUT commands.
- File names must begin with a character in the range of A—Z and all other characters must be in the range of A—Z, 0—9, or be a - (dash) or an _ (underscore). No other special characters are allowed.
- File names must be terminated by a single colon.
- The length of any file name cannot exceed seven characters in a z/VSE environment or eight characters in a z/OS environment.
- The Reporting Facility automatically generates a field with the same name as the file name to be used in conjunction with the GET and GOTO commands.
- If duplicate file names in a single Reporting Facility run are used, an error condition occurs.
- For disk or tape input files, the file name must match the DDNAME in a z/OS environment or the DLBL/TLBL name in a z/VSE environment.

Valid File or Table Name

- IN-FILE:
- DQF:
- CARDFLE:
- DISKIN:

Invalid File or Table Name

The following entries result in the described error conditions:

123:

First character is not A—Z.

FILENAME-TOO-LONG:

Name exceeds maximum length.

INFILE

Terminating colon is missing.

FILE#1:

Special character is present.

INPUT:

File name is a reserved word.

Logical Expressions

When you need to perform processes based on a set of predefined conditions, you can code these conditions using a logical expression.

A logical expression consists of one or more elementary conditions. Each elementary condition can be connected by an AND or an OR connector to form compound conditions. Use parentheses to indicate the order in which conditions or compound conditions are to be evaluated. The AND connector cannot be used with the PERFORM command.

Code an elementary condition as follows:

```
►►─┬──────────┬─ operand-A ─ relop ─ operand-B ─┬─────────┬──────────────►◄
   ├─ WHERE ─┤                                  └─ logop ─┘
   └─ WHEN ──┘
```

WHERE or WHEN

Separators between elementary conditions (used to improve statement readability). WHEN and WHERE can be used interchangeably and are never required operands.

operand-A

The first operand of the logical expression. It must be the field name of a predefined field.

relop

Relational operator used to specify the type of comparison to be performed between operand-A and operand-B. The following illustrates all possible relational operators for a logical expression:

=

Equal to

>

Greater than

<

Less than

EQ

Equal to

NE

Not equal to

GT

Greater than

LT

Less than

GE, GTE

Greater than or equal to

LE, LTE

Less than or equal to

Any of these operators may be preceded by the logical NOT operator to cause inversion of the relation. For example, specifying the relation NOT LT (not less than) implies a comparison for greater than or equal to (GTE).

operand-B

The comparator of the expression that is, the field name, constant, or mask to be compared with the value in the field specified by operand-A. Code the operand in any of the following formats:

- Previously defined field name
- Numeric constant
- Alphanumeric literal
- Bit mask
- Range of values (or a range test)

If specifying a range test, you do not need to specify the equal sign (or its equivalent). Any other relational operator is flagged as an error. Precede the range of values with at least one scalar variable or one numeric literal; otherwise, the Reporting Facility returns error messages. Code range tests as follows:

```
►►─┬─ numeric literal ──┬─ (low-range ─┬─ ─┬─ high-range) ──────────────►◄
   └─ scalar variable. ─┘              ├─ THRU ─┤
                                       └─ TO ───┘
```

logop

Logical connector specifying the relationship of the condition with previous conditions in the predicate set. Valid values are AND and OR. When you omit the logical operator and code multiple conditions, the system assumes OR.

Note the following:

- Parentheses indicate the order in which compound conditions are evaluated. The logical evaluation begins with the innermost pair of parentheses and proceeds to the outermost pair.
- The AND operator takes precedence over the OR operator. Adjacent AND conditions are evaluated prior to OR conditions. Like operators are processed from left to right.
- If a series of elementary conditions are connected by an OR and operand-A is identical in all conditions, you can omit both the OR operator and the field name in operand-A. For better readability, insert commas between the conditions.

If operand-B is not a literal or numeric, you cannot use the abbreviated form:

```
FIELDX EQ FIELDA or FIELDX EQ FIELDB
```

The following logical expressions yield the same results:

```
FIELDX EQ 1 OR FIELDX EQ 3 OR FIELDX (5-7) OR FIELDX GT 10
FIELDX EQ 1 3 (5-7) GT 10
```

The following table shows valid combinations of operands for condition tests:

X

X,alphanumeric literal

N,P,B

N,P,B,numeric constant

S

Bit mask

When you use field names in any of the operands of a logical expression, you can code any valid field name, including scalar variables, array elements, and field name qualifications.

Note: An equal (EQ) compare of two unequal length fields will result in the shorter field being padded with blanks before the compare.

Arithmetic Expressions

An arithmetic expression in a Reporting Facility program allows you to specify one or more arithmetic operations to be performed on specified data elements.

The terms involved in the expression can be numeric fields or numeric constants connected by arithmetic operators. Evaluation of the expression is carried out according to a specified order to yield a single numeric result.

Define numeric fields in the expression as either packed decimal, zoned decimal, or binary. The Reporting Facility automatically converts all components of the expression to packed decimal format, performs the calculation, and converts the final result to the specified format of the result field.

The types of arithmetic operators that are valid in an arithmetic expression are:

+

(addition)

-

(subtraction)

*****

(multiplication)

/

(division)

When using the subtraction operator (-), code a space on each side of the operator so the Reporting Facility compiler can distinguish between an arithmetic operator and a dash in a data name. Code any other operators contiguously in the expression.

You can also use parentheses to indicate the hierarchy of operation within an arithmetic expression. When the order is not completely specified, the Reporting Facility assumes that the order of operation is:

- Unary plus and unary minus
- Multiplication and division
- Addition and subtraction
- Like operators are processed from left to right

Coding parentheses in an expression, helps define the hierarchy of operation and improve readability of the expression. Each method of coding the expression yields the same result:

```
COMPUTE  resultfield = A+B/C - D/C*F+G*C/H+I
COMPUTE  resultfield = A+(B/C)-(D/C*F)+(G*C)/H+I
```

Call Library Member Names

A call library member is a group of statements that you want to save in the Reporting Facility call library. Load the statements to the call library and name the member using the LOAD command.You cannot refer to a library member until it has been loaded.

Code the member name according to these standards:

- Assign library member names using LOAD commands only.
- Maximum length of member names is eight characters.
- The first character of the member name must be in the range A—Z. All other characters must be in the range A—Z or 0—9. No special characters are allowed.

Valid Names

- PAYROLL
- CARDFILE
- PMF001
- X

Invalid Names

The following entries result in the described error conditions:

123:

First character is not A—Z.

MEMBERNAMETOOLONG

Name exceeds maximum length.

FILE#1

Special character is present.

DISPLAY

Name is a reserved word.

User-Module Names

When Reporting Facility standard facilities are insufficient to process a specific request, you may need to supply an accessible alternate processing module. The modules can perform different functions, but naming conventions are the same.

Follow these conventions when assigning a user-module name:

- The first character of the user-module name must be alphabetic (A—Z) and all other characters must be in the range of A—Z or 0—9. No special characters are allowed.
- Module names have a maximum length of eight characters.
- The Reporting Facility reserves some special module names for internal use. These modules generally begin with DR. Avoid beginning your module names with DR since unpredictable results occur if any of the Reporting Facility special modules are overlaid.

Valid User-Module Names

- DBALTURT
- ACCESS02
- PRTXIT
- X

Invalid User-Module Names

The following entries result in the described error conditions:

123:

First character is not A—Z.

MODULENAMETOOLONG

Name exceeds maximum length.

db#1

Special character is present.

INPUT

Name is a reserved word.

Data Format

The Reporting Facility allows you to define and process numeric and alphanumeric data within a program. Follow these guidelines:

- Define numeric fields in an input record area as packed decimal, zoned decimal, or binary. The maximum length for a zoned decimal field is 15 digits. For a packed decimal field, the maximum length is 8 bytes. For a binary field, it is 4 bytes.
- Define numeric fields in the General Storage Area (GSA) as packed decimal only.
- Define GSA fields with an initial value. If you code no initial value, the system automatically sets alphanumeric fields to blanks and numeric fields to zeros.
- Allocate storage to the input record area fields according to the type and precision of the field, but do not give them an initial value. The values of input record area fields are unpredictable prior to the first read of the input file.

Note: Numeric precision is the number of integers and the number of decimals, separated by a decimal point. For alphabetic fields, the precision is the length of the field, in bytes.

Data Movement and Alignment

The Reporting Facility can effect the movement and reformatting of data based on the internal data representation of the fields involved. It also allows the three types of numeric fields to be completely interchangeable.

Numeric fields of any type can be involved in an arithmetic or movement operation. The Reporting Facility handles all data conversion automatically. The system performs all internal arithmetic operations in packed decimal format and converts the final result to the format of the receiving field just before the operation is complete.

The system automatically aligns numeric fields or constants involved in arithmetic calculations by the decimal point before the operation takes place. It is not necessary to specify fields having equal numbers of integers or decimal places. The Reporting Facility pads or truncates as appropriate.

You can specify alphanumeric data fields with a length of 1 to 256 bytes. Data transfer between alphanumeric fields begins with the leftmost byte and continues until all characters are transferred, or until the last byte of the receiving field is filled.

Data Editing

The Reporting Facility uses a standard method of printing data, whether printing the data from an input file or from the General Storage Area (GSA). The system can print alphanumeric and numeric data. The system cannot print bit strings.

Alphanumeric data always prints in character format and, therefore, requires as many print positions as the length of the field, in bytes. The system does not edit alphanumeric data.

Default Numeric Editing

Numeric data, by default, is printed using a standard method of editing. The ordering of the data into its final edited form, and the calculation of the number of print positions required to contain it, are based upon the length and precision of the field to be printed.

Numeric fields, defined in the GSA by a DEFINE command or defined implicitly with a precision in a SET, COMPUTE, or DECODE command, adopt the specified dimensions in the translation to their final edited form. Numeric fields defined in an input record use the field type specified in the DEFINE command to calculate the number of digits, integers, and the location of the decimal point.

The number of digits that can be contained within the field types N (zoned decimal) and P (packed decimal) are calculated as follows:

- N = number of bytes
- P = (number of bytes * 2) -1

For fields defined as B (binary), the number of digits to be printed is calculated as follows:

- 1-byte binary = 3 digits
- 2-byte binary = 5 digits
- 3-byte binary = 7 digits
- 4-byte binary = 10 digits

The system assumes that all numeric fields to be printed have a plus or minus sign. Therefore, the rightmost position in this final edited form is reserved for a sign. The system suffixes positive or zero values with a single blank and negative values with a minus sign.

If the value contains both integers and decimals, the system prints both, separated by a decimal point. If the value contains no decimals, the rightmost integer prints adjacent to the sign position. If the value contains only decimals (no integers), a decimal point and a single zero print immediately to the left of the numeric value.

The calculation to determine the total number of characters required to print a numeric field can be summarized as follows:

- If decimals = 0, then characters = integers + 1
- If integers = 0, then characters = decimals + 3
- Otherwise, characters = integers + decimals + 2

Edit Pattern Specification

Sometimes it is necessary to specify the printing of numeric data items using different editing rules from those supplied by the Reporting Facility. If this is the case, supply a PICTURE clause on the DEFINE command used to define the numeric item. The PICTURE clause is valid only on the DEFINE command and restricts edit pattern specification to explicitly defined fields.

Implicitly defined numeric fields always print using the default printing techniques discussed in the previous section. The edit mask in the PICTURE clause is specified as an alphanumeric literal enclosed in apostrophes, and the field must be defined as either N (zoned decimal) or P (packed decimal).

COBOL-like edit patterns can be coded, although some restrictions apply. The following editing features are supported:

- Fixed insertion editing
- Floating insertion editing
- Zero suppression/replacement editing
- Sign editing

Each edit pattern must be composed of specific characters so the type of editing to be performed can be determined. Valid characters in an edit pattern are described as follows:

9

Specifies a numeric digit position that is always replaced by the corresponding digit from the source field, with no zero suppression implied.

Z

Specifies a numeric digit position that is replaced by a digit from the source field if a significant source digit (nonzero) is encountered; otherwise, the position is set to a blank (X'40').

*****

Specifies a numeric field position that is replaced by a digit from the source field if a significant source digit is encountered; otherwise, the position is set to an asterisk (X'5C'). This gives the ability to "asterisk protect" leading nonsignificant digits in a printed field.

$ or €

Specifies a numeric digit position that is replaced by a digit from the source field if a significant source digit is encountered. Leading, nonsignificant positions are set to blank and the dollar sign or euro currency symbol is printed to the left of the first significant digit. This enables you to "float" the currency symbol with a printed field.

You can also specify the currency symbol as the first character in the edit pattern, along with other "replacement" characters. This causes a single currency symbol to print in a fixed location. Then normal editing continues.

Note: To change from a dollar sign to a euro currency symbol, specify the hex code 9F in the PICTURE clause.

+

Specifies that a plus sign is to be printed when the numeric item being edited is positive. When the numeric item is negative, a blank prints. You must code this as the rightmost character in the edit pattern.

-

Specifies that a minus sign is to be printed when the numeric item being edited is negative. When the numeric item is positive, a blank prints. A minus sign, or dash, can also be used as a separator. (An example is a social security number.)

DB

Performs like the plus sign, with the exception that DB is printed in the two rightmost positions when the numeric item is positive.

CR

Performs like the minus sign, with the exception that CR is printed in the two rightmost positions when the numeric item is negative.

.

Specifies that a decimal point is to print in the corresponding print position. Only one decimal point is allowed in a single edit pattern.

,

Specifies that a comma is to be printed in the corresponding print position. Any number of commas can be coded in a single edit pattern.

/

Specifies that a slash is to be printed in the corresponding print position. Any number of slashes can be coded in a single edit pattern.

The following table illustrates some practical uses of the edit pattern specification:

EDIT PATTERN	VALUE/PRECISION	EDITED RESULT
PIC 'ZZZZ9.99999'	0000012345 (5.5)	0.12345
PIC '$$,$$9.99-'	1234567 (5.2)	$12,345.67
PIC '$**,***.**'	0034567 (5.2)	$**,345.67
PIC '999'	123 (3.0)	123
PIC '999-'	123-(2.1)	123-
PIC 'Z9'	03406 (3.2)	6
PIC '999'	000 (3.0)	000
PIC 'ZZZ'	000 (3.0)	blank
PIC '$$$9.99CR'	12345-(3.2)	$123.45CR
PIC '$$$9.99DB'	12345 (3.2)	$123.45DB
PIC 'ZZZZZ9'	00123-(4.1)	123
PICTURE '999CR'	123 (3.0)	123
PICTURE 'ZZZZ9'	00000 (1.4)	0

Note the following:

- The total number of replacement characters in the edit pattern cannot exceed the total number of digits in the field, or 15, whichever is smaller.
- When the number of replacement characters in the edit pattern is less than the number of digits in the field, editing starts with the rightmost digit and proceeds left.
- When the edited field is printed, the Reporting Facility uses the length of the edit pattern, not the field, to calculate the maximum size of the print window.
- When an edited field is accumulated, the Reporting Facility edits the accumulator using the same edit pattern, inserting two significant digits on the left.

Chapter 4: CA Datacom/DB Reporting Facility Tutorial

This chapter describes how to use the Reporting Facility, primarily by examining a simple program and the report it generates. Later, you will have the opportunity to test your programming skills by writing a program.

Although not all of the commands, facilities, and functions are discussed in this tutorial section, it can be used as a starting point for learning the Reporting Facility. The descriptions of the commands are not comprehensive, but are simplified to demonstrate the basics of the program. For more detailed information about commands, consult the Command Reference section in this guide.

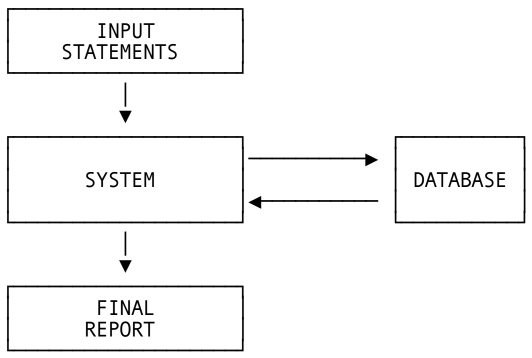
The data accessed to produce the sample report is stored in CA Datacom/DB, and it is assumed that you have a basic knowledge of CA Datacom/DB. However, many of the concepts used to access the CA Datacom/DB table can be applied to using the Reporting Facility with standard files. Later in this manual, you will be given the opportunity to write a program that accesses a standard file.

Reporting Facility

The Reporting Facility is an easy-to-use file retrieval and report generating system. Communication with the system is through the Reporting Facility language. This language is made up of simple English-like input statements used to access a database and establish the contents and format of the report.

You must code these statements in a prescribed manner and then submit them with the appropriate Job Control Language (JCL) statements. Some of the statements direct the program to access specified data from the database.The Reporting Facility reads the statements and produces the desired report. For the JCL statements required to execute the program under z/OS and z/VS, see Sample Reporting Facility JCL Streams (see page 139).

The generalized interactions between the input statements, the system, the database, and the final report are illustrated below:



Input Statements

Each input statement (line of coding) consists of a command and its parameters. The statements tell the system what data to access from the database and how the report is to appear. Since the Reporting Facility reacts to one line of coding at a time, reading from top to bottom, the order in which the statements are coded is important. In general, code your individual statements as follows:

1. Specify a heading with the USER command.
2. Define the tables with the appropriate INPUT command format.
3. Define the fields with the appropriate DEFINE command format.
4. Move data with the MOVE command if accessing a CA Datacom/DB table.
5. Start reading the table with the GET command.
6. Direct program flow with the GOTO command.
7. Manipulate the data with one of the following commands:
   ```
   COMPUTE          SUBTRACT
   DECODE           MULTIPLY
   SET              DIVIDE
   ADD
   ```
8. Specify the report with the following commands:
   ```
   REPORT           CONTROL
   SELECT           PRINT
   ```
9. End the program with the END command.

Program and Its Output

The sample report in this guide is created by the Reporting Facility accessing data in an imaginary CA Datacom/DB table, PERSONEL table (PMF). The printed report is illustrated next followed by an illustration of the input statements or coding required to produce that report.

The following report lists all employees who live in the state of Texas and includes each employee name, identification number, city, and zip code. This report is sorted according to the city in which the employee resides.

Printed Report

```
                                XYZ COMPANY, INC.

 01  NOV  08          EMPLOYEE SUMMARY - TEXAS            PAGE   1

                                ID                             ZIP
EMPLOYEE NAME                 NUMBER            CITY           CODE

LUTHER GARY                   00009             DALLAS         75243
WALKER FRANK                  00016             DALLAS         75243
PATTERSON AL                  00018             DALLAS         75243
EVERS DANNY                   00030             DALLAS         75243
     .                          .                 .              .
     .                          .                 .              .
     .                          .                 .              .
CHURCH PHILLIP                00105             HOUSTON        77506
ABEL PHILIP                   00115             HOUSTON        77506
NEELY ROY                     00123             HOUSTON        77506
DIETER RODNEY                 00130             HOUSTON        77506
END OF REPORT
```

Unless intentionally altered with the appropriate commands and parameters, all reports have the following characteristics:

- Reports are centered on the page.
- Headings are centered over the report.
- Report date and page numbers are included.
- Fields are properly spaced.
- Subtotal and total lines are labeled.

Reporting Facility Input Statements

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM RECORD EQ 375 NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

This tutorial gives you the opportunity to practice writing Reporting Facility programs, and explains the input statements and the printed report generated by those statements.

The explanation of each statement includes the relevant general format of the command and the specific coding required to produce the printed report. For an explanation of the complete general format of the command and all of its parameters, see Reporting Facility Commands (see page 147).

Coding a Heading

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375 NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

The USER statement produces the first line of heading on every page of the report. This statement is required and is the first input statement in the program, unless the OPTION command is used.

The USER command has the following format:

```
►►─ USER ─ 'report-heading' ──────────────────────────────────────────►◄
```

In the sample program, the USER statement looks like this:

```
USER 'XYZ COMPANY, INC.'
```

The word USER is coded as shown, followed by the literal value designated as the first heading line on each page of the report. You must enclose the literal with apostrophes.

The sample USER statement tells the Reporting Facility that the first heading line on each page of the report will be XYZ COMPANY, INC.

Defining a Table

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

After the USER statement, the input table must be defined to the Reporting Facility with the INPUT statement.

For a CA Datacom/DB table the INPUT command has the following format:

```
►►─ filename: ─ INPUT ─ DATACOM ─ RECORD ─ EQ ─ xxx ─ NAME ─ EQ ─ xxx ──►

►─ DBID ─ EQ ─ xxx ─┬────────────────────┬─┬──────────────────┬──────────►
                    └─ SIDNAME=xxxxxxxx ─┘ └─ DBIDUSER=9999 ─┘

►─┬─────────────────┬──────────────────────────────────────────────────►◄
  └─ DBIDMIF=9999 ─┘
```

The sample program INPUT statement looks like this:

```
PERSONEL:  INPUT DATACOM RECORD EQ 375 NAME EQ PMF DBID EQ 001
```

PMF is the three-character CA Datacom/DB name for the table that is accessed from CA Datacom/DB and used in the report. 001 is the numeric database ID of the same table.

The RECORD EQ xxx parameter is used by the Reporting Facility to define the total length, in characters, of all the elements to be read from this database table. When calculating the length for a CA Datacom/DB table, add 300 characters to the total length of all the elements to supply the space needed to communicate with CA Datacom/DB.

Coding the RECORD EQ xxx parameter requires an understanding of table definition and input record fields as discussed in the next section.

The INPUT statement tells the Reporting Facility:

- The name of the table for this run is PERSONEL.
- PERSONEL is a CA Datacom/DB table.
- The total length of all the elements to be retrieved from the table (PERSONEL), plus 300, is 375 characters.
- The database table name is PMF.
- The database ID number is 001

The following parameters are used to request data from multiple MUFS and are optional:

- SIDNAME=xxxxxxxx - The xxxxxxxx is the load module name of the SID of the MUF you want to access. SIDNAME defaults to DBSIDPR.
- DBIDUSER=9999 - The 9999 is the DBID that you use in the Reporting Facility program with DBIDMUF. These parameters are only needed if you are using the same DBID in one DR program from the two different MUFs.
- DBIDMUF=9999 - The 9999 is the DBID that MUF knows about.

Note: Code and link the DBSIDPR macro as xxxxxxxx matching the SIDNAME=parameter coded. For more information about coding the DBSIDPR macro, see the *CA Datacom Database and System Administration Guide*.

Defining Input Record Fields

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND    001-005   X
DEFINE PERSONEL-KEY        006-010   X
DEFINE PERSONEL-ELMLIST    191-201   X
DEFINE PERSONEL-NUMBER     301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME       306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY       354-368   X 'CITY'
DEFINE PERSONEL-STATE      369-370   X
DEFINE PERSONEL-ZIP-CODE   371-375   X 'ZIP' 'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

The DEFINE statements assign symbolic names to fields and describe the fields' characteristics so the data can be referenced by commands within the Reporting Facility program.

The DEFINE statements are coded immediately following the INPUT statement and are divided into two areas, the Communications Area and the Data Area, when you are working with a CA Datacom/DB table.

Note: Standard files do not require a Communications Area in the program.

Communications Area

The Communications Area is the first 300 bytes of storage following the INPUT command. It is made up of DEFINE statements which provide the space where the CA Datacom/DB commands, key names, key values, and element list are passed to CA Datacom/DB. Each DEFINE statement defines one field in the area.

The three DEFINE statements coded to make up the Communications Area in the sample program are shown following:

```
DEFINE PERSONEL-COMMAND       001-005 X
DEFINE PERSONEL-KEY           006-010 X
DEFINE PERSONEL-ELMLIST       191-201 X
```

Organization of the Communications Area can be summarized as follows:

CA Datacom/DB Command	5	characters
Key Name	5	characters
Key Value	180	characters
Element List	101	characters
Filler	9	characters
TOTAL	300	characters

The order in which the above fields of the Communications Area are coded is always the same. Although the first 300 positions of the table are always reserved for the Communications Area, not all of the positions must be used.

For example, notice in the three DEFINE statements coded to make up the Communications Area in the sample program that the CA Datacom/DB command (PERSONEL-COMMAND) and the key name (PERSONEL-KEY) occupy the first five and second five positions, respectively, of the Communications Area. However, no key value was specified, so none of the 180 positions (11—190) reserved for the key value are allocated.

Positions 191—291 (101 characters) are reserved for the CA Datacom/DB Element list, but only 11 positions have been allocated in this case. Although the last nine positions (292—300) of the Communications Area are used as filler, do not specify these positions in the coding.

Allocating space within the first 300 positions is further explained in the following discussion.

Code the general DEFINE command in the Communications Area as follows:

```
►►─ DEFINE ─ fieldname ─ start-end ─ X ─────────────────────────────────────►◄
```

In the first DEFINE statement in the Communications Area, PERSONEL-COMMAND is the user-supplied name of the field where the CA Datacom/DB command will be moved using the MOVE command (explained later).

```
DEFINE PERSONEL-COMMAND      001-005 X
DEFINE PERSONEL-KEY          006-010 X
DEFINE PERSONEL-ELMLIST      191-201 X
```

The start-end parameter indicates, in characters, the relative starting and ending position of the field within the record of the input table. Since the organization of the Communications Area requires that the CA Datacom/DB command occupy the first five positions of that area, start-end is replaced with 1—5.

The X parameter indicates that the field named PERSONEL-COMMAND contains alphanumeric data. See DEFINE Command (see page 169) for the complete DEFINE command format and for other parameters that could have been entered here.

The first DEFINE statement tells the Reporting Facility:

- The name of the field that contains the CA Datacom/DB command is PERSONEL-COMMAND.
- The field named PERSONEL-COMMAND is five characters long.
- The field named PERSONEL-COMMAND contains alphanumeric data.

In the second DEFINE statement, PERSONEL-KEY is the user-supplied name of the field where the CA Datacom/DB key name will be moved using the MOVE command.

```
DEFINE PERSONEL-COMMAND      001-005 X
DEFINE PERSONEL-KEY          006-010 X
DEFINE PERSONEL-ELMLIST      191-201 X
```

Since the organization of the Communications Area requires that the CA Datacom/DB key name occupy the second five positions of that area, start-end is replaced with 6—10.

X indicates that the field named PERSONEL-KEY contains alphanumeric data. Positions 11—190 (180 characters) of the Communications Area are used for specifying the key values, although no key value was specified in the sample program. Use the key value to tell CA Datacom/DB where to start accessing data.

For example, one of the key values associated with the key name EMPNO is the employee number 23100. To start accessing data associated with employee number 23100, assign space for the key value starting at position 11 in the Communications Area. Then use the MOVE command to move 23100 to the allocated space.

No key value field is defined in the sample program because the CA Datacom/DB command GETIT in the first MOVE statement indicates that the records will be retrieved sequentially, starting at the beginning of the table. However, the 180 characters reserved for the key value cannot be allocated for any other purposes, even if no key value is specified.

In the third DEFINE statement, PERSONEL-ELMLIST is the user-supplied name of the field where the CA Datacom/DB element list will be moved.

```
DEFINE PERSONEL-COMMAND      001-005 X
DEFINE PERSONEL-KEY          006-010 X
DEFINE PERSONEL-ELMLIST      191-201 X
```

Positions 191—285 (95 characters) of the Communications Area are used for the element list. Each record in the database has from 1 to 255 elements associated with it. Up to 15 elements can be read at one time with this Communications Area.

Not all of the positions (191—285) must be used. The size of the field to contain the element is determined by the command used and the number of desired elements to be retrieved. These elements should not overlap.

Each element name requires five bytes (ADEMP in this example; shorter element names are padded with blanks) followed by a one-byte security code (blank in this example). CA Datacom/DB detects the end of the list when it finds five blanks instead of another element name. Since this example uses only one element and a blank security code, the element list field only needs to be five characters long.

Note: LOCXX commands do not retrieve data and do not use the element list.

In the sample program, there is only one element, ADEMP, designated in the third MOVE statement. This statement allows CA Datacom/DB to access the data associated with the element ADEMP, and to bring the correct data for the report into the Reporting Facility program.

X indicates that the field PERSONEL-ELMLIST contains alphanumeric data.

Data Area

Code the Data Area DEFINE statements after the Communications Area DEFINE statements when working with a CA Datacom/DB table. Otherwise, code the Data Area DEFINE statements after the INPUT statement.

The primary function of the DEFINE statements in the Data Area is to provide space for the accessed data associated with the element, but column headings can also be assigned to the data. The accessed data is moved to the Data Area beginning with position 301.

Code the five DEFINE statements that make up the Data Area in the sample program as follows:

```
DEFINE PERSONEL-NUMBER      301-305  X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME        306-329  X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY        354-368  X 'CITY'
DEFINE PERSONEL-STATE       369-370  X
DEFINE PERSONEL-ZIP-CODE    371-375  X 'ZIP' 'CODE'
```

In these statements, five fields of the element ADEMP have names assigned to them. The fields are the PERSONEL-NUMBER field, PERSONEL-NAME field, PERSONEL-CITY field, PERSONEL-STATE field and PERSONEL-ZIP-CODE field. Space has been allocated for the data in each field. The space allocated for the fields reflects their relative starting and ending positions in the element.

Each line of coding also specifies the type of data the named fields are to contain, and some lines specify column headings for the named fields. The following is the general DEFINE command format for coding the Data Area:

```
►►─ DEFINE ─ fieldname ─ start-end ─ X ─┬─────────────┬──────────────────────►◄
                                         └─ 'heading' ─┘
```

In the first DEFINE statement in the Data Area, PERSONEL-NUMBER is the user-supplied name of the field where the five-character employee identification number is to be moved.

```
DEFINE PERSONEL-NUMBER      301-305  X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME        306-329  X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY        354-368  X 'CITY'
DEFINE PERSONEL-STATE       369-370  X
DEFINE PERSONEL-ZIP-CODE    371-375  X 'ZIP' 'CODE'
```

Since the employee ID number occupies the first five positions of the element, the start-end parameter is 301—305, followed by X, indicating the field is to contain alphanumeric data.

Beginning with this DEFINE statement, an optional heading is coded for some element fields. In the previous (highlighted) line of coding, the field named PERSONEL-NUMBER has been given the column heading ID NUMBER.

Notice in this statement how the placement of the blanks and apostrophes around ID resulted in the centering of ID over NUMBER in the printed report. Remember to place apostrophes correctly if the command format specifies them.

The first DEFINE statement in the Data Area of the sample tells the Reporting Facility that the employee ID numbers are in the field named PERSONEL-NUMBER. PERSONEL-NUMBER is five characters long, contains alphanumeric data and has the column heading **ID NUMBER.**

In the second DEFINE statement, PERSONEL-NAME is the user-supplied name of the field where you move the 24-character employee name.

```
DEFINE PERSONEL-NUMBER      301-305  X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME        306-329  X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY        354-368  X 'CITY'
DEFINE PERSONEL-STATE       369-370  X
DEFINE PERSONEL-ZIP-CODE    371-375  X 'ZIP' 'CODE'
```

Since the employee name field is 24 characters long and occupies positions 6—29 of the element, the start-end parameter is 306-329, followed by X, indicating the field contains alphanumeric data. The field PERSONEL-NAME field has the heading **EMPLOYEE NAME**.

In the third DEFINE statement, PERSONEL-CITY is the user-supplied name of the field indicating the employee's city of residence. **CITY** is the designated heading name.

```
DEFINE PERSONEL-NUMBER      301-305  X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME        306-329  X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY        354-368  X 'CITY'
DEFINE PERSONEL-STATE       369-370  X
DEFINE PERSONEL-ZIP-CODE    371-375  X 'ZIP' 'CODE'
```

Since the city field is 15 characters long and occupies positions 54—68 of the element, the start-end parameter is 354-368, followed by X, indicating the field is to contain alphanumeric data.

Positions 30—53 of the element are not defined. You must assume these positions contain data that is not relevant to processing this report. However, these positions must be included in the RECORD EQ xxx parameter when calculating the total length of the element. In the fourth DEFINE statement, PERSONEL-STATE is the user-supplied name of the field indicating employee's state of residence.

```
DEFINE PERSONEL-NUMBER      301-305  X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME        306-329  X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY        354-368  X 'CITY'
DEFINE PERSONEL-STATE       369-370  X
DEFINE PERSONEL-ZIP-CODE    371-375  X 'ZIP' 'CODE'
```

Since the state field is two characters long, the start-end parameter is 369-370, followed by X, indicating that the field contains alphanumeric data.

For this field, no column heading was designated in the DEFINE statement and it will not be printed in the report. It is used only to control which records appear in the report. For more information, see PRINT Command (see page 221).

In the fifth DEFINE statement, PERSONEL-ZIP-CODE is the user-supplied name of the field indicating the employee's zip code.

```
DEFINE PERSONEL-NUMBER      301-305  X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME        306-329  X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY        354-368  X 'CITY'
DEFINE PERSONEL-STATE       369-370  X
DEFINE PERSONEL-ZIP-CODE    371-375  X 'ZIP' 'CODE'
```

Since the zip code field is five characters long, the start-end parameter is 371—375, followed by X, indicating the field is to contain alphanumeric data. The field named PERSONEL-ZIP-CODE has been given the heading ZIP CODE.

With the previous explanation of the Communications and Data Areas, it is now possible to code the RECORD EQ xxx parameter in the input statement. Recall that the RECORD EQ xxx parameter is used to determine the total length (in characters) of all the elements to be read from this database table, plus 300 characters for the Communications Area. Since the fields that make up the element ADEMP occupy a total of 75 positions, RECORD EQ 375.

The total length of the element includes positions 30—53 of the element even though the data in those positions is not included in the processing.

Moving Data

```
 USER 'XYZ COMPANY, INC.'
 PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
 DEFINE PERSONEL-COMMAND      001-005   X
 DEFINE PERSONEL-KEY          006-010   X
 DEFINE PERSONEL-ELMLIST      191-201   X
 DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
 DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
 DEFINE PERSONEL-CITY         354-368   X 'CITY'
 DEFINE PERSONEL-STATE        369-370   X
 DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
 MOVE 'GETIT' TO PERSONEL-COMMAND
 MOVE 'EMPNO' TO PERSONEL-KEY
 MOVE 'ADEMP' TO PERSONEL-ELMLIST
 GET PERSONEL
 GOTO EOJ WHEN PERSONEL EQ 'E'
 REPORT 'EMPLOYEE SUMMARY - TEXAS'
 SELECT PERSONEL-STATE EQ 'TX'
 CONTROL PERSONEL-CITY
 PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
      PERSONEL-ZIP-CODE
END
```

MOVE statements are coded after the DEFINE statements if you are working with CA Datacom/DB. The MOVE statements move literal strings to fields within the Communications Area.

The literal strings in the sample program are the appropriate CA Datacom/DB command, key name, and list that had locations assigned to them in the Communications Area.

If this program were accessing a non-CA Datacom/DB file, the DEFINE statements would be followed by a GET statement. In the sample program, however, the MOVE statement must come before the GET statement. You cannot tell the Reporting Facility to start processing records (GET) if the CA Datacom/DB command, key name, and element have not already been moved (MOVE) to the locations assigned them. The CA Datacom/DB command, key name, and element contain information vital to processing the table.

Code the general MOVE command in the following format:

```
▶▶─ MOVE ─ 'literal-string' ─ TO ─ fieldname ──────────────────────────▶◀
```

In the first MOVE statement of the sample program, code the word MOVE as shown. PERSONEL-COMMAND is the user-supplied name of the five-character field where the literal string will be moved.

```
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
```

Code the literal string first, then the keyword TO followed by the field name. The CA Datacom/DB command GETIT is the literal string, and must be enclosed by apostrophes. The GETIT command indicates that the table will be searched sequentially, starting with the first record in the table.

The first MOVE statement in the sample program tells the Reporting Facility to move a literal string GETIT, which represents the CA Datacom/DB READ command, to the field named PERSONEL-COMMAND.

The GETIT command specifies that CA Datacom/DB is to read the table in sequential order. Also available is a table positioning command, GSETL, that indicates a starting point for the first read of the table. When the command specified during the first read of the table is the GETIT command, the system automatically issues a GSETL command to position the table at the first record.

In the second MOVE statement, PERSONEL-KEY is the name of the five-character field where the literal string will be moved.

```
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
```

The literal string is coded first, then the keyword TO followed by the field name. The literal string is the CA Datacom/DB key name EMPNO as defined in the CA Datacom/DB Directory. It must be enclosed within apostrophes. The key name EMPNO indicates that the table will be searched using the employee ID number as the key.

The second MOVE statement tells the Reporting Facility to move the literal string EMPNO to the field named PERSONEL-KEY.

In the third MOVE statement, PERSONEL-ELMLIST is the user-supplied name of the 11-character field where the literal string will be moved.

```
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
```

The literal string is coded first, then the keyword TO followed by the field name. Although the literal string ADEMP is only 5 characters long, the field to which it is moved is 11 characters long.

When you are moving alphanumeric literals and the literal is too short, the receiving field is padded with blanks. If this is the case, the trailing blanks are used to satisfy the CA Datacom/DB requirement for the element list terminator. The data associated with the element ADEMP is accessed by CA Datacom/DB to produce the report.

The third MOVE statement tells the Reporting Facility to move the five-character literal string ADEMP to the field named PERSONEL-ELMLIST.

Retrieving a Record

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

Code the GET statement after the MOVE statements. When the GET statement is processed during program execution for a CA Datacom/DB table, the system constructs a CA Datacom/DB request area in the previously set up command area. This request area is used to access the next record.

The CA Datacom/DB command you moved to PERSONEL-COMMAND prior to the GET determines the type of read that is performed. GETIT indicates sequential processing and REDKG indicates a random read using a key value previously provided in the command area. When a standard file is read, the GET statement always returns the next sequential record. Code the general GET command in the following format:

```
►►─ GET ─ filename ──────────────────────────────────────────────►◄
```

The GET statement for the sample program looks like this:

```
GET PERSONEL
```

The word GET is coded as shown. The file name should be the name of the previously defined input file in the INPUT statement.

The GET statement tells the Reporting Facility to get a record in the table named PERSONEL.

Transferring Control

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

The GOTO statement transfers control to a previously defined label statement.

Code the general GOTO command in the following format:

```
►►─ GOTO ─ label ─┬──────────────────────┬──────────────────────────────►◄
                  └─ logical expression ─┘
```

The GOTO statement in the sample program looks like this:

```
GOTO EOJ WHEN PERSONEL EQ 'E'
```

The word GOTO is coded as shown. EOJ is a predefined procedural label to which the program branches when the end of the table is reached. At that time, the program terminates all table reading and produces a report containing only the records selected up to the point of termination.

PERSONEL EQ 'E' is the optional logical expression. A logical expression specifies the logical constraint that must be satisfied prior to performing the branch. In the Reporting Facility, logical expressions may consist of one or more elementary conditions. The format of an elementary condition is:

```
field1 operator field2
```

field1

A previously defined field name

operator

Any of the following:

EQ or =

Equal to

NE or NOT =

Not equal to

GT or >

Greater than

LTE or NOT >

Not greater than

LT or <

Less than

GTE or NOT <

Not less than

field2

One of the following:

- A previously defined field name
- A numeric constant
- An alphanumeric literal
- A bitmask
- A range of values

In the GOTO statement, PERSONEL is the predefined field name and 'E' is the alphanumeric literal.

The GOTO statement tells the Reporting Facility to branch to the end-of-job procedure when it finds the alphanumeric value, 'E', in the return code field, PERSONEL.

Coding a Second Heading

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

The REPORT command specifies the second heading line that can be used to uniquely identify a report.

Code the general REPORT command in the following format:

```
►►─ REPORT ─ 'heading literal' ──────────────────────────────────────►◄
```

The REPORT command for the sample program looks like this:

```
REPORT 'EMPLOYEE SUMMARY - TEXAS'
```

The word REPORT is coded as shown, followed by the title of the desired second heading. The second line of heading is centered on the line directly below the heading line specified by the USER command. The heading literal must be enclosed within apostrophes.

This REPORT command tells the Reporting Facility that the second heading line on each page of the report will be EMPLOYEE SUMMARY - TEXAS.

Selecting Records for Processing

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

The SELECT statement specifies which records you are selecting for processing.

Code the general SELECT command in the following format:

```
►►─ SELECT ─ logical expression ──────────────────────────────────►◄
```

The SELECT statement for the sample program looks like this:

```
SELECT PERSONEL-STATE EQ 'TX'
```

The word SELECT is coded as shown, followed by the logical expression PERSONEL-STATE EQ 'TX'. The logical expression specifies the logical constraint which must be satisfied before you select the record for processing.

PERSONEL-STATE EQ 'TX' conforms to the format of an elementary condition as previously discussed in the GOTO command. PERSONEL-STATE is the defined field name and 'TX' is the alphanumeric literal.

The SELECT statement tells the Reporting Facility to select for processing only records for the state of Texas. The purpose of this statement is to present the state field for screening.

Controlling Printing Sequence

```
USER 'XYZ COMPANY, INC.'
PERSONEL: INPUT DATACOM  RECORD EQ 375  NAME EQ PMF  DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

The CONTROL statement specifies the hierarchical sequence in which the data is presented or specifies the control-break fields. The sample program is concerned only with specifying the sequence in which the report is printed.

The following is the general CONTROL command format relevant to coding the CONTROL statement:

```
►►─ CONTROL ─ fieldname ──────────────────────────────────────────────────►◄
```

The CONTROL statement for the sample program looks like this:

```
CONTROL PERSONEL-CITY
```

The word CONTROL is coded as shown, followed by the name of the field (PERSONEL-CITY) that will be the sort sequence field.

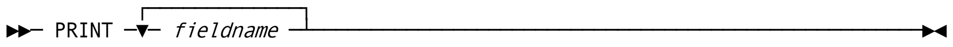
The CONTROL statement in this example tells the Reporting Facility to sort the report in ascending sequence by city.

Printing Report Fields

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

The PRINT statement specifies which fields will be printed on the report and their format. You may also use the PRINT statement to specify the location of each field within a detail line. (The sample program does not demonstrate this.)

Code the general PRINT command in the following format:



The PRINT statement for the sample program looks like this:

```
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
```

The word PRINT is coded as shown, followed by the names of the fields to appear on the report.

Notice in the following report that the contents of the fields named in the PRINT statement are printed on the report under the appropriate column headings and that the fields are properly spaced across the page automatically.

```
                             XYZ COMPANY, INC.

 01  NOV  08             EMPLOYEE SUMMARY - TEXAS              PAGE  1

                               ID                                ZIP
EMPLOYEE NAME                NUMBER            CITY             CODE

LUTHER GARY                  00009             DALLAS           75243
WALKER FRANK                 00016             DALLAS           75243
PATTERSON AL                 00018             DALLAS           75243
EVERS DANNY                  00030             DALLAS           75243
     .                         .                 .                .
     .                         .                 .                .
     .                         .                 .                .
CHURCH PHILLIP               00105             HOUSTON          77506
ABEL PHILIP                  00115             HOUSTON          77506
NEELY ROY                    00123             HOUSTON          77506
DIETER RODNEY                00130             HOUSTON          77506
END OF REPORT
```

As previously discussed, the Reporting Facility uses the name of the field specified in the PRINT statement for the column heading if no heading was specified by a DEFINE statement in the Data Area.

The order in which the column headings and field contents appear is determined by the order in which you enter the names of the fields in the PRINT statement. Therefore, the contents of the field named PERSONEL-NAME are printed first under the column heading EMPLOYEE NAME, followed by the contents of the field named PERSONEL-NUMBER under the column heading ID NUMBER, and so on.

The PRINT statement tells the Reporting Facility:

- The contents of the PERSONEL-NAME, PERSONEL-NUMBER, PERSONEL-CITY, and PERSONEL-ZIP-CODE fields will be printed on the report under the column headings specified for those fields in the Data Area DEFINE statements.
- The order of the printed fields and column headings (left to right) will be PERSONEL-NAME, PERSONEL-NUMBER, PERSONEL-CITY, PERSONEL-ZIP-CODE.

Ending a Program

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID  ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

The last line of coding in a program is the END command. It has no parameters and indicates that the final Reporting Facility statement has been processed.

Writing a Program (CA Datacom/DB Tables)

You have now thoroughly examined the sample program and may test your skills by writing a simple CA Datacom/DB Reporting Facility program. This section presents the requirements for a program to test your understanding of the Reporting Facility concepts.

Before writing the program, become familiar with these guidelines for coding input statements.

- Although a single statement may exceed one input line, the statements should be on separate lines, if possible, to make the program more readable.
- Keywords, commands, field names, numeric constants, and alphanumeric literals may not exceed one input line.
- The entries on a particular line of coding may be separated from each other by any number of spaces.
- The statements are entered free-form, requiring no coding forms.

- When coding a program, remember that the Reporting Facility reads input.
- When entering code at a terminal, each line on the screen is considered to be one card.
- Only the first 71 positions of each line are available for coding input statements.

Write the program to produce a report in the following format:

```
                       XYZ COMPANY, INC.

  01  NOV  08           PAYROLL REPORT                        PAGE  1

 DEPT    EMPLOYEE   EMPLOYEE                 REGULAR         OVERTIME
 NUMBER  NUMBER     NAME                     EARNINGS        EARNINGS

 XXX     XXXX       XXXXXXXXXXXXXXXXXXXX       XXX.XX          XXX.XX
  .         .                .                      .               .
  .         .                .                      .               .
  .         .                .                      .               .
 XXX     XXXX       XXXXXXXXXXXXXXXXXXXX       XXX.XX          XXX.XX
```

This format indicates that the program will generate a report that includes:

- Specified headings
- Department number
- Employee number
- Employee name
- Regular earnings
- Overtime earnings

You will need the following information to write the program:

- The accessed file is a CA Datacom/DB table.
- The table name is PAY and resides in database 001.
- The table is to be sequentially searched, starting with the first record.
- All of the fields defined with DEFINE statements are alphanumeric, except for PAYROLL-REGEARN and PAYROLL-OVTEARN, which are zoned decimal.
- The key name is IDNUM.
- The element name is PAYRC.
- The department number field is named PAYROLL-DEPTNUM and occupies positions 1—3 of the element.
- The employee number field is named PAYROLL-EMPNUM and occupies positions 4—7 of the element.

- The employee name field is named PAYROLL-NAME and occupies positions 8—27 of the element.
- The regular earnings field is named PAYROLL-REGEARN and occupies positions 28—32 of the element.
- The overtime earnings field is named PAYROLL-OVTEARN and occupies positions 33—37 of the element.
- The field to hold the CA Datacom/DB command is named PAYROLL-COMMAND.
- The field to hold the CA Datacom/DB key name is named PAYROLL-KEY.
- The field to hold the CA Datacom/DB element is named PAYROLL-ELMLIST.
- The report is to be sorted in ascending sequence by department number.

After you have written the program, look at the following solution for the correct input statements and an explanation of those statements.

Solution

```
USER 'XYZ COMPANY, INC.'
PAYROLL:  INPUT DATACOM  RECORD EQ 337  NAME EQ PAY DBID EQ 001
DEFINE PAYROLL-COMMAND  001-005   X
DEFINE PAYROLL-KEY      006-010   X
DEFINE PAYROLL-ELMLIST  191-201   X
DEFINE PAYROLL-DEPTNUM  301-303   X ' DEPT ' 'NUMBER'
DEFINE PAYROLL-EMPNUM   304-307   X  'EMPLOYEE' 'NUMBER'
DEFINE PAYROLL-NAME     308-327   X  'NAME'
DEFINE PAYROLL-REGEARN  328-332  N2 'REGULAR' 'EARNINGS'
DEFINE PAYROLL-OVTEARN  333-337  N2 'OVERTIME' 'EARNINGS'
MOVE 'GETIT' TO PAYROLL-COMMAND
MOVE 'IDNUM' TO PAYROLL-KEY
MOVE 'PAYRC' TO PAYROLL-ELMLIST
GET PAYROLL
GOTO EOJ WHEN PAYROLL EQ 'E'
REPORT 'PAYROLL REPORT'
SELECT ALL
CONTROL PAYROLL-DEPTNUM
PRINT PAYROLL-DEPTNUM PAYROLL-EMPNUM PAYROLL-NAME
     PAYROLL-REGEARN PAYROLL-OVTEARN
END
```

The following is a line-by-line explanation of the above payroll program.

USER Statement

In the first statement, the USER command provides the first line of heading on each page of the report.

```
USER 'XYZ COMPANY, INC.'
```

INPUT Statement

In the second statement, the INPUT command defines the table to the Reporting Facility.

```
PAYROLL:  INPUT DATACOM RECORD EQ 337 NAME EQ PAY DBID EQ 001
```

Communications Area

The next three DEFINE statements make up the Communications Area.

- The first DEFINE statement in the Communications Area allocates space for the CA Datacom/DB command.

  ```
  DEFINE PAYROLL-COMMAND    001-005 X
  ```

- The second DEFINE statement in the Communications Area allocates space for the CA Datacom/DB key name.

  ```
  DEFINE PAYROLL-KEY      006-010 X
  ```

- The third DEFINE statement in the Communications Area allocates space for the CA Datacom/DB element.

  ```
  DEFINE PAYROLL-ELMLIST    191-201 X
  ```

Data Area

The next five DEFINE statements make up the Data Area.

- In the first and second DEFINE statements in the Data Area, the PAYROLL-DEPTNUM and PAYROLL-EMPNUM fields are defined and assigned column headings.

  ```
  DEFINE PAYROLL-DEPTNUM    301-303 X ' DEPT ' 'NUMBER'
  DEFINE PAYROLL-EMPNUM     304-307 X 'EMPLOYEE' ' NUMBER '
  ```

- The third DEFINE statement in the Data Area defines the NAME field.

  ```
  DEFINE PAYROLL-NAME        308-327   X   'NAME'
  ```

- In the fourth and fifth DEFINE statements in the Data Area, the PAYROLL-REGEARN and PAYROLL-OVTEARN fields are defined and assigned column headings.

  ```
  DEFINE REGEARN    328-332 N2 'REGULAR' 'EARNINGS'
  DEFINE OVTEARN    333-337 N2 'OVERTIME' 'EARNINGS'
  ```

  Notice the N2 parameter in both of the above statements. N indicates that the fields named PAYROLL-REGEARN and PAYROLL-OVTEARN contain zoned decimal data. The number of implied decimal places is indicated by the integer 2.

Move Statements

- The first MOVE statement moves the CA Datacom/DB command GETIT to the PAYROLL-COMMAND field in the Communications Area. GETIT specifies a sequential search of the table.

  ```
  MOVE 'GETIT' TO PAY-COMMAND
  ```

- The second MOVE command moves the CA Datacom/DB key name IDNUM to the PAYROLL-KEY field in the Communications Area. IDNUM indicates that the table is to be searched by using the employee ID number as the key.

  ```
  MOVE 'IDNUM' TO PAYROLL-KEY
  ```

- The third MOVE statement moves the CA Datacom/DB element, PAYRC, to the PAYROLL-ELMLIST in the Communications Area.

  ```
  MOVE 'PAYRC' TO PAYROLL-ELMLIST
  ```

GET Statement

The GET statement tells the Reporting Facility to return a record from the table named PAYROLL.

```
GET PAYROLL
```

GOTO Statement

The GOTO statement specifies a branch to the end-of-job procedure when the Reporting Facility determines the end of the table.

```
GOTO EOJ PAYROLL EQ 'E'
```

REPORT Statement

The REPORT command specifies that the second heading line on each page of the report is to be PAYROLL REPORT.

```
REPORT 'PAYROLL REPORT'
```

SELECT Statement

The SELECT statement specifies that all the records in the table are to be selected for processing.

```
SELECT ALL
```

CONTROL Statement

The CONTROL statement specifies that the report is to be sorted by department number.

```
CONTROL PAYROLL-DEPTNUM
```

PRINT Statement

The PRINT statement specifies which fields are to be printed and the order in which they are to be printed.

```
PRINT PAYROLL-DEPTNUM PAYROLL-EMPNUM PAYROLL-NAME
       PAYROLL-REGEARN PAYROLL-OVTEARN
```

END Command

The END command signals the Reporting Facility that the last statement has been processed.

Writing a Program (Standard Files)

If you use the Reporting Facility to access standard files, practice writing a program that accesses a standard file. This section presents the requirements of a report and then provides the program that generated it. The report generated by the program you write has the following format:

```
                             XYZ COMPANY, INC.

01    NOV   08               EARNINGS REPORT                   PAGE   1


DEPT    EMPLOYEE  EMPLOYEE                         REGULAR    OVERTIME
NUMBER  NUMBER    NAME                             EARNINGS   EARNINGS


XXX     XXXX      XXXXXXXXXXXXXXXXXXXX               XXX.XX     XXX.XX
.       .         .                                       .          .
.       .         .                                       .          .
.       .         .                                       .          .
XXX     XXXX      XXXXXXXXXXXXXXXXXXXX               XXX.XX     XXX.XX


                                                               -------

GRAND TOTAL                                                   XXXXX.XX

                                                              --------
```

This format indicates that the program will generate a report that includes:

- Specified headings
- Department number
- Employee number
- Employee name
- Regular earnings
- Overtime earnings
- Grand total for overtime earnings

The following are the position and contents of the input record fields:

1-3

Department number

4-7

Employee number

8-27

Employee name

28-29

Hours

30-33

Current rate (9.999)

34-38

Regular earnings (999.99)

39-43

Overtime earnings (999.99)

In writing the program, make the following assumptions:

- The accessed file is a standard sequential disk file.
- The file name is PAYFILE.
- Each record on the file is 43 bytes long.
- Each block of data on the disk file contains 50 records.
- The department number field is PAYFILE-DEPTNUM.
- The employee number field is PAYFILE-EMPNUM.
- The employee name field is PAYFILE-NAME.
- The regular earnings field is named PAYFILE-REGEARN.
- The overtime earnings field is named PAYFILE-OVTEARN.
- The report is sorted in ascending sequence by department number.

Note: You may also assume that all of the fields defined with a DEFINE statement are alphanumeric, except the regular earnings and overtime earnings fields, which are numeric with two implied decimal places.

Remember that you would provide the requirements (as shown previously) in a real situation. Study the command formats and note which parameters are commands or reserved words and which parameters you must supply.

There are three significant differences between the earnings report program in this section and the earnings report program in the previous section.

1. The program in this section requires a standard file definition.
2. There is no Communications Area in this program because the file is standard.
3. The program in this section requires a grand total for the overtime earnings field. The grand total is specified with the PRINT command.

After you have written the program, check the following solution to find the correct input statements and their explanations. .

Solution

```
USER 'XYZ COMPANY, INC.'
PAYFILE:  INPUT  DISK  BLOCK EQ 2150  FIXED RECORD EQ 43
DEFINE PAYFILE-DEPTNUM        001-003  X ' DEPT ' 'NUMBER'
DEFINE PAYFILE-EMPNUM         004-007  X 'EMPLOYEE' 'NUMBER' 'NUMBER'
DEFINE PAYFILE-NAME           008-027  X 'EMPLOYEE' '  NAME  '
DEFINE PAYFILE-REGEARN        034-038 N2 'REGULAR' 'EARNINGS'
DEFINE PAYFILE-OVTEARN        039-043 N2 'OVERTIME' 'EARNINGS'
GET PAYFILE
GOTO EOJ WHEN PAYFILE EQ 'E'
REPORT 'EARNINGS REPORT'
SELECT ALL
CONTROL PAYFILE-DEPTNUM
PRINT PAYFILE-DEPTNUM PAYFILE-EMPNUM PAYFILE-NAME
     PAYFILE-REGEARN (PAYFILE-OVTEARN)
END
```

Following is a line-by-line explanation of the earnings report program.

- In the first statement, the USER command is used to provide the first line of heading on each page of the report.

  ```
  USER 'XYZ COMPANY, INC.'
  ```

- In the second statement, the INPUT command is used to define the file to the Reporting Facility The BLOCK EQ parameter is calculated by multiplying 43 x 50.

  ```
  PAYFILE:  INPUT DISK BLOCK EQ 2150 FIXED RECORD EQ 43
  ```

- In the third, fourth, and fifth statements, the PAYFILE-DEPTNUM, PAYFILE-EMPNUM, and PAYFILE-NAME fields are defined and assigned column headings.

  ```
  DEFINE PAYFILE-DEPTNUM       001-003 X ' DEPT ' 'NUMBER'
  DEFINE PAYFILE-EMPNUM        004-007 X 'EMPLOYEE' ' NUMBER '
  DEFINE PAYFILE-NAME          008-027 X 'EMPLOYEE' '  NAME  '
  ```

- In the sixth and seventh statements, the PAYFILE-REGEARN and PAYFILE-OVTEARN fields are defined and assigned column headings.

  ```
  DEFINE PAYFILE-REGEARN    034-038 N2 'REGULAR'  'EARNINGS'
  DEFINE PAYFILE-OVTEARN    039-043 N2 'OVERTIME' 'EARNINGS'
  ```

  Notice the N2 parameter in both of the above statements. N indicates the fields named PAYFILE-REGEARN and PAYFILE-OVTEARN contain zoned decimal data. The number of implied decimal places is indicated by the integer 2.

- In the eighth statement, the GET command is used to get a record from PAYFILE.

  ```
  GET PAYFILE
  ```

  The GET statement following the DEFINE statement is a significant deviation from the previous programs in this guide and requires some explanation.

As previously discussed, the general order of entry for individual statements is as follows:

- Specify a heading with the USER command.
- Define the files with the INPUT command.
- Define the fields with the appropriate DEFINE command format.
- Move data with the MOVE command if accessing a CA Datacom/DB file.
- Start reading the file with the GET command.
- Direct program flow with the GOTO command.
- Manipulate the data with one of the following commands:

```
COMPUTE              MULTIPLY
DECODE               DIVIDE
SET                  MOVE
ADD                  INCREMENT
SUBTRACT             DECREMENT
```

- Specify the report with the following commands:

```
REPORT                 CONTROL
SELECT                 PRINT
```

- End the program with the END command.

Note: Since this program accesses a standard file and not a CA Datacom/DB file, no MOVE statements are required.

In this earnings report program, the GET and GOTO statements were not really necessary because the Reporting Facility would have automatically provided those two statements after the DEFINE statements. The system automatically provides GET and GOTO statements for the first file defined. See GET Command (see page 202) for more information about the GET command.

In the two previous programs in this guide, you specified the GET and GOTO statements. If they had not been specified, the Reporting Facility would have automatically generated them following the last DEFINE statement. This would produce undesirable results.

When a GET statement is found that specifies the name of the first file defined, the automatic GET and GOTO statements are not generated.

■ In the ninth statement, the GOTO command is used to direct the Reporting Facility to the end-of-file procedure when it finishes reading the file. The GOTO statement is not mandatory for this program.

```
GOTO EOJ PAYFILE EQ 'E'
```

■ In the tenth statement, the REPORT command is used to specify the second line of heading.

```
REPORT 'EARNINGS REPORT'
```

- In the eleventh statement, the SELECT command is used to indicate which records are to be selected for processing.

  ```
  SELECT ALL
  ```

- In the twelfth statement, the CONTROL command is used to specify how the report is to be sorted.

  ```
  CONTROL PAYFILE-DEPTNUM
  ```

- In the thirteenth statement, the PRINT command is used to specify which fields are to appear in the report, the order in which the fields are to be printed, and which fields are to have a total.

  ```
  PRINT PAYFILE-DEPTNUM PAYFILE-EMPNUM PAYFILE-NAME
                PAYFILE-REGEARN (PAYFILE-OVTEARN)
  ```

  PAYFILE-OVTEARN is enclosed in parentheses. This indicates that the values in the PAYFILE-OVTEARN field are to be accumulated and printed as a final total at the end of the report.

- The END command terminates the Reporting Facility program.

Writing a Program (Multiple CA Datacom/DB Tables)

In this section, you have the opportunity to further expand your Reporting Facility programming skills by writing a program that accesses more than one table.

The tables to be accessed are the sample CA Datacom/DB PERSONEL table and PAYROLL table. As with previous sample programs, all the information needed to write the program is provided.

Write the program to generate a report with this format:

```
                                 XYZ COMPANY, INC.

 09   JAN   08                   PAYROLL REPORT                    PAGE  1


EMPLOYEE                                        CURRENT      YEAR-TO-DATE
NUMBER    NAME                        STATUS    PAY RATE            WAGES


XXXXX     XXXXXXXXXXXXXXXXXXXXXX        X        99999.99        $$$,$$9.99
 .         .                            .             .                  .
 .         .                            .             .                  .
 .         .                            .             .                  .
XXXXX     XXXXXXXXXXXXXXXXXXXXXX        X        99999.99        $$$,$$9.99

                                                                 ----------

GRAND TOTAL                                                    $$$$$,$$9.99
```

The previous example indicates that the program will generate a report that includes:

- Specified headings
- Employee number
- Employee name
- Employee status (whether the employee is salaried or paid hourly)
- Current pay rate
- Year-to-date wages, with the indicated edit pattern
- Grand total for year-to-date wages

Note: Assume that the status field in the record contains either S (salaried) or H (hourly).

You can make the following assumptions in writing this program:

- The accessed tables are both CA Datacom/DB tables.
- The table name for the PERSONEL table is PMF. It resides in database 001.
- The table name for the PAYROLL table is PAY, and it resides in database 001.
- Total length of the PERSONEL table element, is 75 characters.
- The total length of the PAYROLL table element, is 39 characters.
- The PERSONEL table is searched sequentially, starting with the first record.
- The PAYROLL table is matched one-to-one with a record in the PERSONEL table. (Hint: The employee number is a common key to both tables.)
- All fields defined with the DEFINE statements are alphanumeric, except the current pay rate and year-to-date wages fields, which are numeric. The current pay rate field has three implied decimal places, and the year-to-date wages field has two implied decimal places.
- The field to hold the PERSONEL table CA Datacom/DB command is PERSONEL-COMMAND.
- The field to hold the PERSONEL table CA Datacom/DB key name is PERSONEL-KEY.
- The field to hold the PERSONEL table CA Datacom/DB element is PERSONEL-ELMLIST.
- The employee number field is PERSONEL-EMPLOYEENUM and occupies positions 1—5 of the PERSONEL table element.
- The employee name field is PERSONEL-NAME and occupies positions 6—29 of the PERSONEL table element.
- The field to hold the PAYROLL table CA Datacom/DB command is PAYROLL-COMMAND.

- The field to hold the PAYROLL table CA Datacom/DB key name is PAYROLL-KEY.
- The field to hold the PAYROLL table CA Datacom/DB key value is PAYROLL-VALUE.
- The field to hold the PAYROLL table CA Datacom/DB element is PAYROLL-ELMLIST.
- The employee code field is PAYROLL-CODE in position 6 of the PAYROLL table element. The code field contains an A (active) or an I (inactive).
- The status field is PAYROLL-STATUS and occupies position 7 of the PAYROLL table element.
- The current pay rate field is PAYROLL-CURRENTRATE and occupies positions 8—15 of the PAYROLL table element.
- The year-to-date wages field is PAYROLL-YTDWAGES and occupies positions 16—23 of the PAYROLL table element.
- The key name for the PERSONEL table and PAYROLL table is EMPNO (employee number).
- The element name for the PERSONEL table is IDEMP.
- The element name for the PAYROLL table is PAYRC.
- Code the program so that, if there is no match of employee number in the PERSONEL and PAYROLL tables (return code N for PAYROLL GET statement), neither record is processed, but another PERSONEL record is read.
- Select for processing only those records that have an A (active) in the code field, but do not print it in the report.
- The report is to be sorted in ascending sequence by employee number.

Keep in mind that you would supply the above requirements in a real situation. Study the command formats and note which parameters are commands or reserved words and which parameters are user-supplied.

Note: This should be the general structure of the program:

- USER statement
- First INPUT statement
- DEFINE statements for first table
- Second INPUT statement
- DEFINE statements for second table
- MOVE statements for first table
- GET statement for first table
- GOTO statement for first table
- MOVE statements for second table
- GET statement for second table

- GOTO statement for second table
- REPORT command
- SELECT statements
- CONTROL statement
- PRINT statement
- END statement

After you have written the program, check the following solution to find the correct input statements and their explanation.

Solution

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375  NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND     001-005 X
DEFINE PERSONEL-KEY         006-010 X
DEFINE PERSONEL-ELMLIST     191-201 X
DEFINE PERSONEL-EMPLOYEENUM 301-305 X 'EMPLOYEE' ' NUMBER '
DEFINE PERSONEL-NAME        306-329 X 'NAME'
PAYROLL:  INPUT DATACOM  RECORD EQ 339  NAME EQ PAY DBID EQ 001
DEFINE PAYROLL-COMMAND      001-005 X
DEFINE PAYROLL-KEY          006-010 X
DEFINE PAYROLL-VALUE        011-015 X
DEFINE PAYROLL-ELMLIST      191-201 X
DEFINE PAYROLL-CODE             306 X
DEFINE PAYROLL-STATUS           307 X
DEFINE PAYROLL-CURRENTRATE  308-315 N3 'CURRENT' ' PAY RATE '
DEFINE PAYROLL-YTDWAGES     316-323 N2
       'YEAR-TO-DATE' '    WAGES' PIC '$$$,$$9.99'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'IDEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
MOVE 'REDKG' TO PAYROLL-COMMAND
MOVE 'EMPNO' TO PAYROLL-KEY
MOVE PERSONEL-EMPLOYEENUM TO PAYROLL-VALUE
MOVE 'PAYRC' TO PAYROLL-ELMLIST
GET PAYROLL
GOTO START WHEN PAYROLL EQ 'N'
REPORT 'PAYROLL REPORT'
SELECT PAYROLL-CODE EQ 'A'
CONTROL PERSONEL-EMPLOYEENUM
PRINT PERSONEL-EMPLOYEENUM PERSONEL-NAME PAYROLL-STATUS
   PAYROLL-CURRENTRATE (PAYROLL-YTDWAGES)
END
```

The following is a line-by-line explanation of the above report.

- In the first statement, the USER command is used to provide the first line of heading on each page of the report.

```
USER 'XYZ COMPANY, INC.'
```

- In the second statement, the INPUT command is used to define the first table to the Reporting Facility. Notice the RECORD EQ parameter. Although you only took data from two of the fields (employee number and name) in the element, you must include the total length of all the elements read from this database table, plus 300 characters for the CA Datacom/DB interface.

```
PERSONEL:  INPUT DATACOM RECORD EQ 375 NAME EQ PMF DBID EQ 001
```

- The next five statements define these fields: PERSONEL-COMMAND, PERSONEL-KEY, PERSONEL-ELMLIST, PERSONEL-EMPLOYEENUM, PERSONEL-NAME. The PERSONEL-EMPLOYEENUM field is assigned an alternate column heading. These statements make up the Communications Area and the Data Area for the PERSONEL table.

```
DEFINE PERSONEL-COMMAND      001-005 X
DEFINE PERSONEL-KEY          006-010 X
DEFINE PERSONEL-ELMLIST      191-201 X
DEFINE PERSONEL-EMPLOYEENUM  301-305 X 'EMPLOYEE' ' NUMBER '
DEFINE PERSONEL-NAME         306-329 X 'NAME'
```

- In the eighth statement, the INPUT command defines the second table to the Reporting Facility. Notice the RECORD EQ parameter. You took data from only the first 23 bytes, but you must include the total length of all elements read from this database table plus 300 characters for the CA Datacom/DB interface.

```
PAYROLL:  INPUT DATACOM RECORD EQ 339 NAME EQ PAY DBID EQ 001
```

- In the next eight statements define these fields: PAYROLL-COMMAND, PAYROLL-KEY, PAYROLL-VALUE, PAYROLL-ELMLIST, PAYROLL-CODE, PAYROLL-STATUS, PAYROLL-CURRENTRATE, PAYROLL-YTDWAGES. The PAYROLL-CURRENTRATE and PAYROLL-YTDWAGES fields are assigned alternate column headings. These statements comprise the Communications Area and Data Area for the PAYROLL table. Notice the way in which the edit pattern is specified for the year-to-date wages field.

```
DEFINE PAYROLL-COMMAND      001-005 X
DEFINE PAYROLL-KEY          006-010 X
DEFINE PAYROLL-VALUE        011-015 X
DEFINE PAYROLL-ELMLIST      191-201 X
DEFINE PAYROLL-CODE             306 X
DEFINE PAYROLL-STATUS           307 X
DEFINE PAYROLL-CURRENTRATE  308-315 N3 ' CURRENT ' 'PAY RATE'
DEFINE PAYROLL-YTDWAGES     316-323 N2
     'YEAR-TO-DATE' '  WAGES' PIC '$$$,$$9.99'
```

Following the DEFINE statements for the second table are the MOVE statements for the first table.

- The first MOVE statement moves the CA Datacom/DB command GETIT to the CA Datacom/DB command field in the Communications Area. GETIT specifies a sequential search of the PERSONEL table, beginning with the first record.

  ```
  MOVE 'GETIT' TO PERSONEL-COMMAND
  ```

- The second MOVE statement moves the CA Datacom/DB key name EMPNO to the CA Datacom/DB key name field in the Communications Area. EMPNO indicates that the table is to be searched by using the employee identification number as the key.

  ```
  MOVE 'EMPNO' TO PERSONEL-KEY
  ```

- The third MOVE statement moves the CA Datacom/DB element IDEMP to the CA Datacom/DB element list field in the Communications Area. Although the literal string IDEMP has only 5 characters, the field to which it is moved is 11 characters long. When alphanumeric literals are moved, the receiving field is padded with blanks if the literal is too short. The trailing blanks satisfy the CA Datacom/DB requirement for the element list terminator.

  ```
  MOVE 'IDEMP' TO PERSONEL-ELMLIST
  ```

- The GET statement after the first MOVE statement tells the Reporting Facility to return a record from the first table (PERSONEL).

  ```
  GET PERSONEL
  ```

- The GOTO statement specifies a branch to the end-of-job procedure when the program determines that there are no more records in the first table (PERSONEL) to process.

  ```
  GOTO EOJ PERSONEL EQ 'E'
  ```

After the GOTO statement for the first table, the MOVE statements for the second table (PAYROLL) are coded.

- The first MOVE statement for the second table moves the CA Datacom/DB command REDKG to the CA Datacom/DB command field in the Communications Area of that table. REDKG specifies a random search of the second table.

  ```
  MOVE 'REDKG' TO PAYROLL-COMMAND
  ```

- The next MOVE statement moves the CA Datacom/DB key name EMPNO to the CA Datacom/DB key name field in the Communications Area of the second table. The key name EMPNO indicates that the second table, like the first table, is to be searched by using the employee identification number as the key.

  ```
  MOVE 'EMPNO' TO PAYROLL-KEY
  ```

- The next MOVE statement moves the employee number that was identified in the PERSONEL-EMPLOYEENUM field in the PERSONEL table to the CA Datacom/DB key value field in the Communications Area of the PAYROLL table. The employee number tells the program where to start accessing data in the second table. Since a random search was specified for the second table with the CA Datacom/DB command REDKG, the Reporting Facility must be told where to start accessing the data associated with this table.

  ```
  MOVE PERSONEL-EMPLOYEENUM TO PAYROLL-VALUE
  ```

- The last MOVE statement moves the CA Datacom/DB element PAYRC to the CA Datacom/DB element list field in the Communications Area of the PAYROLL table.

  ```
  MOVE 'PAYRC' TO PAYROLL-ELMLIST
  ```

- The next statement is the GET statement for the second table. This statement tells the Reporting Facility to return a record from the PAYROLL table.

  ```
  GET PAYROLL
  ```

The next line of coding is the GOTO statement for the second table. Remember that you were instructed to code the program so that if there were no match of employee numbers in the PERSONEL record and the PAYROLL record, neither record would be selected for printing.

- The GOTO statement below tells the Reporting Facility to go to START (first procedural statement) if the GET statement for the PAYROLL table has a return code of 'N'.

  ```
  GOTO START PAYROLL EQ 'N'
  ```

- The REPORT command specifies that the second heading line on each page of the report is to be PAYROLL REPORT.

  ```
  REPORT 'PAYROLL REPORT'
  ```

- The SELECT statement specifies that only those records that have an A (active) in the field named PAYROLL-CODE will be selected for processing.

  ```
  SELECT PAYROLL-CODE EQ 'A'
  ```

- The CONTROL statement specifies that the report is to be sorted in ascending order by employee number. The 'A' parameter was not necessary. The Reporting Facility defaults to an ascending sort if the parameter is not included.

  ```
  CONTROL PERSONEL-EMPLOYEENUM
  ```

- The PRINT statement specifies which fields are to be printed, the order in which they are to be printed, column headings for the PERSONEL-NAME and PAYROLL-CODE fields and a total for the PAYROLL-YTDWAGES field.

  ```
  PRINT PERSONEL-EMPLOYEENUM PERSONEL-NAME PAYROLL-STATUS
               PAYROLL-CURRENTRATE (PAYROLL-YTDWAGES)
  ```

- The END command signals that the last statement has been processed.

Chapter 5: Reporting Facility Modes of Operation

This chapter discusses each of the four modes of operation, how and why specific modes are entered, and what resources are necessary when a particular mode is invoked. The four modes of operation are:

- Primary
- Secondary
- Write-Only
- Library Maintenance

Any operating mode is accessible through a single execution of the Reporting Facility. The mode of operation for any particular run is determined at runtime. The system enters a mode in one of two ways:

- Explicitly, directed by your selection of a basic processing option
- Implicitly, directed by the Reporting Facility after interpreting the source commands and determining the most efficient utilization of resources

After you enter any of the operating modes, the Reporting Facility is dedicated to that activity or processing method for the duration of the run.

Primary Mode

The Primary mode is the most frequently entered mode of operation and is best suited for report generation. It provides methods for resequencing input data or generating multiple reports.

There are four main processes performed in Primary mode:

Compilation/Code Generation

The Reporting Facility compiler analyzes the input source commands and generates machine code to perform the specified function. The system stores machine code, with the GSA, on the DRWORK file to be recalled later and used for execution.

Record Selection/Data Manipulation

The system recalls and executes the portion of the machine code on the DRWORK file responsible for performing presort functions. This includes the reading of input files, conditional record selection, and presort data movement, if necessary. As each of the input record sets are selected for processing, the system generates one or more records on the internal Hit File. Each record contains only the data required to print the final reports and the necessary sort control information.

Sort

After all input record sets have been processed, the Reporting Facility enters end-of-job processing, closes the Hit File, and invokes the sort program.

Report Generation

The system recalls and executes the machine code responsible for report printing. The system generates this machine code to read the sorted Hit File and produce the output report.

In Primary mode, a single pass of the input data files can produce multiple reports, with each report printed in a different sequence and containing different data.

The following data sets are required when executing the Reporting Facility in Primary mode:

- DRWORK
- SORTIN/SORTIN1
- SORTLIB *(z/OS)*
- SORTOUT
- SORTWK01-06/SORTWK1-8
- SYSIN *(z/OS)*
- SYSPRINT *(z/OS)*

Secondary Mode

Secondary mode is also suited to the functions involved in report generation, but is more efficient than Primary mode. Each request is processed faster and the system uses minimum resources.

The system enters this mode of operation either *explicitly*, by specifying the OPTION SORT=NONE parameter in the Reporting Facility source program, or *implicitly*, by the Reporting Facility system where it is possible to establish from the source commands that the system is to generate a single report in the same sequence as the primary input file.

Two main processes are performed in this mode.

Compilation/Code Generation

The Reporting Facility compiler analyzes the input source commands and generates the machine code to perform the specified function. The system stores the machine code, with the GSA, on the DRWORK file for later recall and execution.

Record Selection/Data Manipulation/Report Printing

The system recalls and executes all code generated on the DRWORK file. As each input record set is processed or selected, it prints directly on the output report. When END-OF-FILE is reached on the input data, final totals print and the run terminates. Since no intermediate sort is necessary, the system does not generate a Hit File.

These data sets are required when executing in Secondary mode:

- DRWORK
- SYSIN *(z/OS)*
- SYSPRINT *(z/OS)*

Write-Only Mode

The Write-Only mode allows creation of a single sequential output file rather than report generation. Enter this mode only by specifying the OPTION WRITE ONLY command in the Reporting Facility program.

You can perform two main processes in this mode.

Compilation/Code Generation

The Reporting Facility compiler analyzes the input source commands and generates the machine code needed to perform the specified function. The system stores the machine code, with the GSA, on the DRWORK file for later recall and execution.

Record Selection/Data Manipulation/Output File Creation

The system recalls and executes all of the code on the DRWORK file. As each input record set processes, the system analyzes the appropriate selection criteria, reformats the data, and creates a single output record in a format you defined.

Data sorting is not available in this mode. When input data reaches END-OF-FILE, the output file closes and the run immediately terminates.

These data sets are required when executing in Write-Only mode:

- DROUT
- DRWORK

- SYSIN *(z/OS)*
- SYSPRINT *(z/OS)*

Coding a Heading

```
USER 'XYZ COMPANY, INC.'
PERSONEL:  INPUT DATACOM  RECORD EQ 375 NAME EQ PMF DBID EQ 001
DEFINE PERSONEL-COMMAND      001-005   X
DEFINE PERSONEL-KEY          006-010   X
DEFINE PERSONEL-ELMLIST      191-201   X
DEFINE PERSONEL-NUMBER       301-305   X '  ID ' 'NUMBER'
DEFINE PERSONEL-NAME         306-329   X 'EMPLOYEE NAME'
DEFINE PERSONEL-CITY         354-368   X 'CITY'
DEFINE PERSONEL-STATE        369-370   X
DEFINE PERSONEL-ZIP-CODE     371-375   X 'ZIP'   'CODE'
MOVE 'GETIT' TO PERSONEL-COMMAND
MOVE 'EMPNO' TO PERSONEL-KEY
MOVE 'ADEMP' TO PERSONEL-ELMLIST
GET PERSONEL
GOTO EOJ WHEN PERSONEL EQ 'E'
REPORT 'EMPLOYEE SUMMARY - TEXAS'
SELECT PERSONEL-STATE EQ 'TX'
CONTROL PERSONEL-CITY
PRINT PERSONEL-NAME PERSONEL-NUMBER PERSONEL-CITY
     PERSONEL-ZIP-CODE
END
```

The USER statement produces the first line of heading on every page of the report. This statement is required and is the first input statement in the program, unless the OPTION command is used.

The USER command has the following format:

```
►►─ USER ─ 'report-heading' ──────────────────────────────────────►◄
```

In the sample program, the USER statement looks like this:

```
USER 'XYZ COMPANY, INC.'
```

The word USER is coded as shown, followed by the literal value designated as the first heading line on each page of the report. You must enclose the literal with apostrophes.

The sample USER statement tells the Reporting Facility that the first heading line on each page of the report will be XYZ COMPANY, INC.

Library Maintenance Mode

The Library Maintenance mode provides the only means for maintaining the Reporting Facility call library. Using this library enables you to catalog frequently used statements or groups of statements for later retrieval.

You can enter Library Maintenance mode *only* by specifying the LIBRARY command. If you specify the LIBRARY command, it must be the first command in the input source stream.

The run stream consists of a series of specialized Library Maintenance commands. The commands and their basic functions are:

LIBRARY

Instructs the Reporting Facility to enter the Library Maintenance mode. This must be the first command in the input stream or must immediately follow the first OPTION command.

CREATE

Preformats the entire extent when used before you use any Reporting Facility call library. Perform this function only once, usually during initial installation of the Reporting Facility.

LOAD

Specifies that any source statements that follow will be cataloged in the library with a unique name. The data portion of the member to be cataloged can be terminated by another Library Maintenance command, an END command, or end-of-data on the input stream. Use this facility to add new members to the library or to update existing members.

CONDENSE

Allows the periodic reorganization of the library after considerable DELETE/LOAD activity. The Reporting Facility automatically condenses the library when less than 10 percent of the total library area remains available for use. But it never physically deletes a member from the library until after performing a CONDENSE function.

DELETE

Marks the specified member for deletion from the library. The system deletes the member when the library is condensed, but the member is inaccessible until the condense takes place. Space on the library is not available for reuse until a reorganization takes place.

DISPLAY

Allows you to display various items in the library on the system printer, or punch specific members to an alternate data set.

These data sets are required to execute in the Library Maintenance mode:

- DRLIB
- DRWORK
- SYSIN *(z/OS)*
- SYSPRINT *(z/OS)*

Chapter 6: Analyzing Reporting Facility Output

The Reporting Facility is designed to be user friendly. The output source listings are ordered in logical sequence and contain enough information for you to analyze the internal and external requirements for each program.

Print these listings with each execution of the system to provide adequate documentation for future reference to the program. Direct the printing of this system information with the OPTION command.

OPTION LIST ON MAP

Designates that you wish to print all system information, including the source listing and diagnostics, LIST ON, and internal information about the run environment, MAP.

OPTION LIST OFF

Specifies that only the output reports are to be printed.

Many of the items printed can be used to analyze the processing and the resource requirements. These items help you and CA Support debug the Reporting Facility program when problems occur. The following is a detailed look at each item you can print prior to executing the program.

Source Listing

The source listing is the first item printed if OPTION LIST ON is specified. It contains two sections:

- Source statement listing
- Compilation summary

Each section is discussed in the following pages. Samples are provided.

Source Statement Listing

The source statement listing is a sequential list of the Reporting Facility commands in the order in which they were input to the system. This provides a hardcopy of the program logic. This portion of the output listing is printed in a standard format, and includes the following information:

- Corporate name and logo
- Current date and time of the compilation
- Operating system and version level information
- Source statements with unique sequential line numbers
- Page numbers

Unique sequential line numbers allow error or warning messages to specify which command or command line is in error.

Note: For more information, see the *CA Datacom/DB Message Reference Guide*.

EJECT, SKIP1, SKIP2, or SKIP3 commands in the source program can control the spacing on this portion of the output listing.

Compiler Summary

The compiler summary immediately follows the END command, whether you code it or it is generated automatically by the system.

A summary of all errors or warnings issued during the course of the compilation is printed. This includes the start and end times of the compilation phase. Length of the compilation phase is determined by subtracting the start time from the stop time.

Following is a sample of the first page of the report. For an example of the report header (not shown here), see Sample Report Headers).

```
┌──┐
| 1|         OPTION LIST ON MAP
| 2|         USER   'XYZ INC.'
| 3|         CALL   CARDFILE
|  |
| 4|  NOTE   CUSTOMER/SALESMAN DATA FILE
| 5|  INPT:    FILE   CARD
| 6|         DEFINE NAME                1 TO  2 X
| 7|         DEFINE CUSTOMER-NUMBER     3 TO  5 X  'CUSTOMER'  'NUMBER'
| 8|         DEFINE CITY                      6 X
| 9|         DEFINE STATE               7 TO  8 X
|10|         DEFINE ZIP-CODE            9 TO 13 X  'ZIP'  'CODE'
|11|         DEFINE CUSTOMER-SALE-ID   14 TO 15 X  'SALESMAN'  'ID'
|12|         DEFINE CREDIT-LIMIT       16 TO 23 N2 'CREDIT'  'LIMIT'
|13|                                               PIC '$$$,$$9.99'
|14|         DEFINE CURRENT-BALANCE    24 TO 30 N2 'CURRENT'  'BALANCE'
|15|                                               PIC '$$,$$9.99'              Source Listing
|16|         DEFINE SALESMAN-NAME      32 TO 39 X  'SALESMAN'  'NAME'
|17|         DEFINE SALESMAN-ID        40 TO 41 X  'SALESMAN'  '   ID   '
|18|         DEFINE YTD-SALES          42 TO 49 N2 'YEAR-TO-DATE'  'SALES'
|19|                                               PIC '$$$,$$9.99'
|20|         DEFINE BRANCH-ID          50 TO 52 X  'BRANCH'  '   ID   '
|22|         DEF    UNUSED-CREDIT(6.2)=0       'UNUSED' 'CREDIT'
|23|                                           PIC '***,**9.99-'
|24|         CALL   DECODE2 USING STATE ST
|25|  NOTE   STATE CODE TRANSLATION TABLE
|26|         DECODE  :01 INTO :02
|27|                 'TX' EQ 'TEXAS'
|28|                 'TN' EQ 'TENNESSEE'
|29|                 'NC' EQ 'NORTH CAROLINA'
|30|                 'GA' EQ 'GEORGIA'
|31|                 'OK' EQ 'OKLAHOMA'
|32|                 'NY' EQ 'NEW YORK'
|33|                 'OH' EQ 'OHIO'
|34|                 'NJ' EQ 'NEW JERSEY'
|35|                 'CT' EQ 'CONNETICUT'
|36|                 'IL' EQ 'ILLINOIS'
|37|                 'PN' EQ 'PENNSYLVANIA'
|38|                 'MI' EQ 'MICHIGAN'
|39|                 'CO' EQ 'COLORADO'
|40|                 'WA' EQ 'WASHINGTON'
|41|                 'CA' EQ 'CALIFORNIA'
|42|                          ELSE  'UNKNOWN'   'STATE'  'NAME'
└──┘
 ▲  Internally assigned sequence numbers
```

Following is a sample of the second page of the report.

```
    44           CALL   DECODE3 USING NAME CUSTOMER-NAME
    44
    45   NOTE   CUSTOMER NAME TRANSLATION TABLE
    46           DECODE :01 INTO :02
    47                   '01' EQ 'HIGH-ROLLING INVESTMENT '
    48                   '02' EQ 'SOUTHERN FRIED FOODS    '
    49                   '03' EQ 'LEGAL TOBACCO CO.       '
    50                   '04' EQ 'SOUTHERN PINE INDUSTRIES'
    51                   '05' EQ 'BARONIAL OIL CO.        '
    52                   '06' EQ 'COSMOPOLITAN FASHIONS   '
    53                   '07' EQ 'HEAVY METAL MACHINERY   '
    54                   '08' EQ 'AIRPORT SERVICES CORP.  '
    55                   '09' EQ 'STOLID INSURANCE CORP.  '
    56                   '10' EQ 'INLAND GRAIN TERMINALS  '
    57                   '11' EQ 'STEEL CURTAIN STEEL INC.'
    58                   '12' EQ 'PERFECT BEARING CORP.   '
    59                   '13' EQ 'MOUNTAIN STATES MINING  '
    60                   '14' EQ 'NORTHWEST PLYWOOD MILLS '
    61                   '15' EQ 'ORIENTAL TRADING CO.    '
    62                   '16' EQ 'WEST COAST LIFESTYLES   '
    63                            OTHERWISE ' '  'NAME'
    64           CALL   DECODE4 USING CITY CUSTOMER-CITY
    64
    65   NOTE   CITY TRANSLATION TABLE
    66           DECODE :01 INTO :02
    67                   'A' EQ 'DALLAS'
    68                   'B' EQ 'MEMPHIS'
    69                   'C' EQ 'CHARLOTTE'
    70                   'D' EQ 'ATLANTA'
    71                   'E' EQ 'OKLAHOMA CITY'
    72                   'F' EQ 'NEW YORK'
    73                   'G' EQ 'CLEVELAND'
    74                   'H' EQ 'NEWARK'
    75                   'I' EQ 'HARTFORD'
    76                   'J' EQ 'CHICAGO'
    77                   'K' EQ 'PITTSBURGH'
    78                   'L' EQ 'DETROIT'
    79                   'M' EQ 'DENVER'
    80                   'N' EQ 'SEATTLE'
    81                   'O' EQ 'SAN FRANCISCO'
    82                   'P' EQ 'LOS ANGELES'    ELSE ' '  'CITY'
```

Following is a sample of the third page of the report.

```
   84          SUBTRACT CURRENT-BALANCE FROM CREDIT-LIMIT
   85                   GIVING UNUSED-CREDIT

   87          REPORT  'XYZ SAMPLE REPORT 01'
   88          SELECT  ALL
   89          CONTROL QSEQ
   90          PRINT   CUSTOMER-NAME CUSTOMER-SALE-ID CUSTOMER-CITY STATE
   91                  ZIP-CODE CREDIT-LIMIT SALESMAN-ID CURRENT-BALANCE

   93          REPORT  'XYZ SAMPLE REPORT 02'
   94          SELECT  ALL
   95          CONTROL NAME
   96          PRINT   DOUBLE SPACING CUSTOMER-NAME CUSTOMER-NUMBER
   97                  CUSTOMER-CITY ST ZIP-CODE
   98                  CUSTOMER-SALE-ID CREDIT-LIMIT

  100          REPORT  'XYZ SAMPLE REPORT 03'
  101          SELECT  'A' (UNUSED-CREDIT GT 0)
  102          SELECT  ALL
  103          CONTROL ST CITY
  104          PRINT   @20 CUSTOMER-NAME 20 CUSTOMER-NUMBER 3 CREDIT-LIMIT
  105                  3 (CURRENT-BALANCE) 3 (A;UNUSED-CREDIT)
  106          PRINT   @CUSTOMER-NAME CUSTOMER-CITY 3 ST 3 ZIP-CODE

  108          REPORT  'XYZ SAMPLE REPORT 04'
  109          SELECT  ALL
  110          CONTROL NAME
  111          PRINT   CUSTOMER-NAME CUSTOMER-NUMBER CUSTOMER-CITY
  112                  ST ZIP-CODE CUSTOMER-SALE-ID  CREDIT-LIMIT
          ┌───┐
  113     │END│
          └───┘
           ▲   Automatically generated when necessary

                                  ┌──────────────────┐
           COMPILE PHASE COMPLETED -│NO ERRORS FOUND   │   ◄  Error/warning summary
                                  │NO WARNINGS ISSUED│
                                  └──────────────────┘
          ┌───────────────────────────┐
          │START 16:54:54 - STOP 16:54:57│ ◄  Compiler start/stop time:  Elapsed time:  3 seconds
          └───────────────────────────┘
```

MAP Listing

The MAP Listing, printed by specifying OPTION MAP, summarizes the internal and external system requirements necessary to process a specific request. You can determine the input and output requirements for a specific run by analyzing the MAP Listing.

Four items are printed on the map:

- Compile diagnostics
- Field name cross-reference listing
- Hit File layout
- Report description

Compile Diagnostics

The Compile Diagnostics is divided into five units:

1. The File Summary displays general information regarding both input and output files for the run, including blocking factors, record sizes, and block sizes for each file. It includes sort key information if more than one report is to be generated or a CONTROL statement is coded.
2. The General Storage Area Summary (GSA) lists each of the eight types of GSA entries added to the GSA during compilation, and the number of entries required in each category. The system also uses this area as a cross-reference to the Trace Facility.
3. The Field Names Table Summary (FNT) lists the four types of field name entries required in the run, including labels, scalar variables, internal file description blocks and arrays, and the number of entries required in each category.
4. The Function Specification Table Summary (FST) lists the 16 types of FST entries required in the run. The Reporting Facility uses FSTs internally to generate machine code to perform specific functions.
5. The Memory Use Summary displays the estimated core storage requirements for the internal compilation work areas. This does not include core storage required for the execution of the program, such as input buffers, machine code, and external software.

Following is a sample of the fourth page of the report. For an example of the report header (not shown here), see Sample Report Headers).

```
──────────────────────────────────┐
I/P  00001 X 00080 = 00080   INPT│     ◄Input file
──────────────────────────────────┘
O/P  00048 X 00096 = 04608   HITFILE
─────────────────────────────────────────┐
SRT  00005 - 00021 = 00016   .SORT KEY INFO.│  ◄Sort key generated on Hit File
─────────────────────────────────────────┘   (start position.end position.length)
GSA  00000001919 (00000000193 ENTRIES)
    ┌────────────────────────────────────────────────┐
    │GSA 1 = 00000000002 NUMERIC CONSTANT ...... CON│
    │GSA 2 = 00000000136 QUOTED LITERAL ........ LIT│
    │GSA 3 = 00000000001 INTERMEDIATE RESULT ... RES│
    │GSA 4 = 00000000049 UNQUOTED LITERAL ...... INT│    ◄General Storage Area utilization
    │GSA 5 = 00000000002 ACCUMULATOR ........... ACC│
    │GSA 6 = 00000000002 VALUE ................. VAL│
    │GSA 7 = 00000000000 STRING ................ STR│
    │GSA 8 = 00000000001 HEX LITERAL ........... HEX│
    └────────────────────────────────────────────────┘
FNT  00000000820 (00000000046 ENTRIES)
    ┌────────────────────────────────────────────────┐
    │FNT 0 = 00000000005 LABELS ................  01│
    │FNT 1 = 00000000040 SCALAR VARIABLES ......  01│    ◄Field Name Table
    │FNT 2 = 00000000001 FILE DESC. BLOCKS .....  02│
    │FNT 3 = 00000000000 ARRAYS ................  03│
    └────────────────────────────────────────────────┘
FST  00000009828 (00000000351 ENTRIES)
    ┌────────────────────────────────────────────────┐
    │FST 0 = 00000000004 INTERNAL ONLY ......... -02│
    │FST 1 = 00000000005 USER/REPORT ...........  01│
    │FST 2 = 00000000012 INPUT AREA VARIABLE ...  02│
    │FST 3 = 00000000191 GSA VARIABLE ..........  03│
    │FST 4 = 00000000002 DATA MOVEMENT .........  04│
    │FST 5 = 00000000050 DECODE ................  05│
    │FST 6 = 00000000005 SEQUENCE AND CONTROL ..  06│
    │FST 7 = 00000000002 CONDITION TEST ........  07│    ◄Function Specification Table
    │FST 8 = 00000000006 SELECTION .............  08│
    │FST 9 = 00000000035 PRINT LAYOUT ..........  09│
    │FST A = 00000000030 HIT RECORD EXTENSION ..  0A│
    │FST B = 00000000001 GET RECORD ............  0B│
    │FST C = 00000000000 CONTROL BREAK CALC ....  0C│
    │FST D = 00000000000 UNDEFINED .............  0D│
    │FST E = 00000000000 ARRAY DEF/REL. INDEX ..  0E│
    │FST F = 00000000008 DUMMY .................  0F│
    └────────────────────────────────────────────────┘
────────────────────────────────────────────────────┐
GETMAIN REQUIREMENT FOR GSA/FNT/FST ... APPROX. 013K│     ◄Core result analysis
────────────────────────────────────────────────────┘
```

Field Name Cross-Reference Listing

The Field Name Cross Reference Listing shows each user-defined and internally defined data element and field name in ascending sorted sequence. It indicates the origin of the data element, and whether the field resides in the General Storage Area or in one of the input records.

The listing also shows the internally assigned qualifier next to the origin for each of the data elements. This is not the qualifier used in the source statements, but is an internally assigned number used to show the difference in the fields.

For example, fields defined in the GSA always have an internal qualifier of 00, where fields defined in an input record are assigned internal qualifiers of 01 through nn, where nn represents the total number of input files defined in the run.

An X under the HIT column denotes that the field will be included in the Hit File.

An X under the ARY column indicates that the field specified is an array.

The rightmost columns, entitled REPORTS, indicate the specific report in which the field is referenced. For instance, if a particular field was referenced in both the first and third report groups, a 1 and a 3 would appear in this column next to the field name.

Following is a sample of the fifth page of the report. For an example of the report header (not shown here), see Sample Report Headers).

```
 ------ FIELD NAME ------ ORIGIN  QUAL  HIT |ARY| --------------------------------- REPORTS ---------------------------------

                                             ▲ Array indication

 |ABORT:             |      |*LABEL*||    |
 |BLANK              |      |* GSA *||(00)|
 |BRANCH-ID          |      |INPT   ||(01)|Field to be placed in Hit File
 |CITY               |      |INPT   ||(01)|  ▼
 |                   |      |       ||    |
 |CREDIT-LIMIT       |      |INPT   ||(01)| |X|        |1  2  3  4|    ◄Report number of reference to field
 |CURRDATE           |      |* GSA *||(00)| | |        |          |
 |CURRENT-BALANCE    |      |INPT   ||(01)| |X|        |1  3      |
 |CURRENT-DATE       |      |* GSA *||(00)| | |        |          |
 |CURRENT-TIME       |      |* GSA *||(00)| | |        |          |
 |CURRTIME           |      |* GSA *||(00)| | |        |          |
 |CUSTOMER-CITY      |      |* GSA *||(00)| |X|        |1  2  3  4|
 |CUSTOMER-NAME      |      |* GSA *||(00)| |X|        |1  2  3  4|
 |CUSTOMER-NUMBER    |      |INPT   ||(01)| |X|        |2  3  4   |
 |CUSTOMER-SALE-ID   |      |INPT   ||(01)| |X|        |1  2  4   |
 |DD                 |      |* GSA *||(00)| | |        |          |
 |END-OF-FILE        |      |* GSA *||(00)| | |        |          |
 |EOJ:               |      |*LABEL*||    | | |        |          |
 |HR                 |      |* GSA *||(00)| | |        |          |
 |INPT               |      |* GSA *||(00)| | |        |          |
 |INPT:              |      |*LABEL*||    | | |        |          |    ◄Field name cross-reference
 |MM                 |      |* GSA *||(00)| | |        |          |
 |MN                 |      |* GSA *||(00)| | |        |          |
 |NAME               |      |INPT   ||(01)| | |        |          |
 |NO-RECORD-FOUND    |      |* GSA *||(00)| | |        |          |
 |QSEQ               |      |* GSA *||(00)| | |        |          |
 |RECORD-FOUND       |      |* GSA *||(00)| | |        |          |
 |SALESMAN-ID        |      |INPT   ||(01)| |X|        |1         |
 |SALESMAN-NAME      |      |INPT   ||(01)| | |        |          |
 |SPACE              |      |* GSA *||(00)| | |        |          |
 |SPACES             |      |* GSA *||(00)| | |        |          |
 |SS                 |      |* GSA *||(00)| | |        |          |
 |ST                 |      |* GSA *||(00)| |X|        |2  3  4   |
 |START:             |      |*LABEL*||    | | |        |          |
 |STATE              |      |INPT   ||(01)| |X|        |1         |
 |TAG                |      |* GSA *||(00)| | |        |          |
 |TEST:              |      |*LABEL*||    | | |        |          |
 |UNUSED-CREDIT      |      |* GSA *||(00)| |X|        |3         |
 |XD                 |      |* GSA *||(00)| | |        |          |
 |XM                 |      |* GSA *||(00)| | |        |          |
 |XY                 |      |* GSA *||(00)| | |        |          |
 |YTD-SALES          |      |INPT   ||(01)| | |        |          |
 |YY                 |      |* GSA *||(00)| | |        |          |
 |ZERO               |      |* GSA *||(00)| | |        |          |
 |ZEROS              |      |* GSA *||(00)| | |        |          |
 |ZIP-CODE           |      |INPT   ||(01)| |X|        |1  2  3  4|

     ▲                       ▲         ▲
Actual field names         Internally  Physical location of field
(user-defined,             assigned
system-defined)            qualifier
```

Hit File Layout

The Hit File Layout includes detailed file and record information for the internally generated Hit File. This layout represents a COBOL Data Division definition and shows how the Reporting Facility expects the definition to look if the file is input to the Reporting Facility.

This technique is used to show the basic content of the file for illustration purposes. The record information includes the following:

- RDW
- Tag character
- Sort key
- Variable length user fields

RDW

The Record Descriptor Word standard for any variable format file. It always occupies the first four bytes of the record.

tag character

A one-character field containing the tag character assigned to the record when it was selected by a SELECT command.

sort key

Includes the report number and the internal sort key necessary after analyzing all of the CONTROL commands in the program. This allows the system to produce multiple reports in the same run with different sorting requirements for each.

variable length user fields

Represent multiple occurring groups of data, each pertaining to a specific report. Each record report group is the same length. Each variable segment listed describes the unique data necessary to print a single report.

Following is a sample of the sixth page of the report. For an example of the report header (not shown here), see Sample Report Headers).

```
           Assumed disk device type of Hit File
                                                        ┌─────────┐
                                                   ▶    |(3350   )|......H I T   F I L E   L A Y O U T......
                                                        └─────────┘
     FD  DROUT                                      DROUT:  FILE DISK SEQUENTIAL VARIABLE RECORD=        100 BLOCK=       4612

                                           ┌───────────────┐
           BLOCK CONTAINS         |4612 CHARACTERS|   ◀Automatically calculated BLOCK/RECORD size
                                           └───────────────┘
           RECORD CONTAINS          100 CHARACTERS
           RECORDING MODE V
           DATA RECORD IS HITFILE.

     01  HITFILE.    ▼ Common to all Hit File records
        ┌─────────────────────┐
        |02 RDW                |             PICTURE X(4)                DEFINE RDW                             1 -     4 X         4
        |02 TAG-CHARACTER      |             PICTURE X                   DEFINE TAG-CHARACTER                   5 -     5 X
        |02 SORT-KEY.          |
        |   03 REPORT-NUMBER   |             PICTURE B                   DEFINE REPORT-NUMBER                   6 -     6 B
        |   03 USER-SORT-KEY   |             PICTURE X(15)               DEFINE USER-SORT-KEY                   7 -    21 X       2
        └─────────────────────┘

         02 REPORT-01.                       XYZ SAMPLE REPORT 01

  Fields unique to Report No. 1
            ┌────────────────────────┐
     ▶      |03 00.CUSTOMER-NAME      |      PICTURE X(24).              DEFINE 00.CUSTOMER-NAME               22 -    45 X
            |03 01.CUSTOMER-SALE-ID   |      PICTURE XX                  DEFINE 01.CUSTOMER-SALE-ID            46 -    47 X
            |03 00.CUSTOMER-CITY      |      PICTURE X(13)               DEFINE 00.CUSTOMER-CITY               48 -    60 X
            |03 01.STATE              |      PICTURE XX                  DEFINE 01.STATE                       61 -    62 X
            |03 01.ZIP-CODE           |      PICTURE X(5)                DEFINE 01.ZIP-CODE                    63 -    67 X       6
            |03 01.CREDIT-LIMIT       |      PICTURE 9(6)V99             DEFINE 01.CREDIT-LIMIT                68 -    75 N2      7
            |03 01.SALESMAN-ID        |      PICTURE XX                  DEFINE 01.SALESMAN-ID                 76 -    77 X       7
            |03 01.CURRENT-BALANCE    |      PICTURE 9(5)V99             DEFINE 01.CURRENT-BALANCE             78 -    84 N2
            └────────────────────────┘

         02 REPORT-02.                       XYZ SAMPLE REPORT 02

  Fields unique to Report No. 2
            ┌────────────────────────┐
     ▶      |03 00.CUSTOMER-NAME      |      PICTURE X(24)               DEFINE 00.CUSTOMER-NAME               22 -    45 X
            |03 01.CUSTOMER-NUMBER    |      PICTURE XXX                 DEFINE 01.CUSTOMER-NUMBER             46 -    48 X       4
            |03 00.CUSTOMER-CITY      |      PICTURE X(13)               DEFINE 00.CUSTOMER-CITY               49 -    61 X       6
            |03 00.ST                 |      PICTURE X(14)               DEFINE 00.ST                          62 -    75 X       7
            |03 01.ZIP-CODE           |      PICTURE X(5)                DEFINE 01.ZIP-CODE                    76 -    80 X       8
            |03 01.CUSTOMER-SALE-ID   |      PICTURE XX                  DEFINE 01.CUSTOMER-SALE-ID            81 -    82 X       8
            |03 01.CREDIT-LIMIT       |      PICTURE 9(6)V99             DEFINE 01.CREDIT-LIMIT                83 -    90 N2
            └────────────────────────┘

         02 REPORT-03.                       XYZ SAMPLE REPORT 03

  Fields unique to Report No. 3
            ┌────────────────────────┐
     ▶      |03 00.CUSTOMER-NAME      |      PICTURE X(24)               DEFINE 00.CUSTOMER-NAME               22 -    45 X
            |03 01.CUSTOMER-NUMBER    |      PICTURE XXX                 DEFINE 01.CUSTOMER-NUMBER             46 -    48 X
            |03 01.CREDIT-LIMIT       |      PICTURE 9(6)V99             DEFINE 01.CREDIT-LIMIT                49 -    56 N2
            |03 01.CURRENT-BALANCE    |      PICTURE 9(5)V99             DEFINE 01.CURRENT-BALANCE             57 -    63 N2
            |03 00.UNUSED-CREDIT      |      PICTURE S9(7)V99     COMP-3 DEFINE 00.UNUSED-CREDIT               64 -    68 P2
            |03 00.CUSTOMER-CITY      |      PICTURE X(13)               DEFINE 00.CUSTOMER-CITY               69 -    81 X
            |03 00.ST                 |      PICTURE X(14)               DEFINE 00.ST                          82 -    95 X
            |03 01.ZIP-CODE           |      PICTURE X(5)                DEFINE 01.ZIP-CODE                    96 -   100 X
            └────────────────────────┘
```

Report Description

The Report Description contains information concerning the appearance of each output report.

The fields associated with each line of the report are listed along with their internally assigned qualifiers from the field name cross-reference. The following are included for each field:

- The line on which the data is to be printed
- A complete description of the print window required to print the field
- The tag character under which the field is conditionally printed
- Print positions within the current printer width that define the print window

Following is a sample of the seventh page of the report. For an example of the report header (not shown here), see Sample Report Headers).

```
                                                                                                     ▼
 Sequentially assigned report number
             ▼                                   Print window size calculation input
            ┌──┐                                                 ▼
 REPORT     │01│FIELD NAME                     HDR1   HDR2    DTL     ACC    TOTAL        PICTURE          TAG   FROM   TO
            └──┘
                ┌───┐┌────────────────────┐  ┌───────────────────────────────────┐   ┌─────────────┐         ┌────────────┐
 LINE   01      │00.││CUSTOMER-NAME       │  │  4            24              24  │   │             │         │   1     24 │
                │01.││CUSTOMER-SALE-ID    │  │  8      2      2               8  │   │             │         │  31     38 │
                │00.││CUSTOMER-CITY       │  │  4            13              13  │   │             │         │  46     58 │
                │01.││STATE               │  │  5             2               5  │   │             │         │  66     70 │
                │01.││ZIP-CODE            │  │  3      4      5               5  │   │             │         │  78     82 │
                │01.││CREDIT-LIMIT        │  │  6      5     11              11  │   │ $$$,$$9.99  │         │  90    100 │
                │01.││SALESMAN-ID         │  │  8      8      2               8  │   │             │         │ 108    115 │
                │01.││CURRENT-BALANCE     │  │  7      7     10              10  │   │  $$,$$9.99  │         │ 123    132 │
                └───┘└────────────────────┘  └───────────────────────────────────┘   └─────────────┘         └────────────┘
                  ▲            ▲                                                            ▲
    Internally          Field name                                                    Edit pattern
    assigned            printed                                                       (PIC clause
    qualifier                                                                         or default)
```

Following is a sample of the eighth page of the report.

```
            ┌──┐
 REPORT     │02│FIELD NAME                       HDR1  HDR2  DTL    ACC    TOTAL        PICTURE            TAG   FROM   TO
            └──┘
             ▲Second report description

 LINE   01     00.CUSTOMER-NAME                   4           24            24                                   1     24
               01.CUSTOMER-NUMBER                 8     6      3             8                                  33     40
               00.CUSTOMER-CITY                   4           13            13                                  49     61
               00.ST                              5     4     14            14                                  70     83
               01.ZIP-CODE                        3     4      5             5                                  92     96
               01.CUSTOMER-SALE-ID                8     2      2             8                                 105    112
               01.CREDIT-LIMIT                    6     5     11            11         $$$,$$9.99              122    132
```

Following is a sample of the ninth page of the report.

```
       ┌──┐
REPORT │03│FIELD NAME                 HDR1  HDR2  DTL   ACC   TOTAL       PICTURE           TAG   FROM   TO
       └──┘
        ▲Third report description

LINE   01   00.CUSTOMER-NAME              4           24          24                               20    43
            01.CUSTOMER-NUMBER            8     6      3           8                               64    71
            01.CREDIT-LIMIT               6     5     11          11         $$$,$$9.99            75    85
            01.CURRENT-BALANCE            7     7     10    12    12          $$,$$9.99            89   100
                                                                                         ┌─┐
            00.UNUSED-CREDIT              6     6     11    13    13        ***,**9.99-  │A│   104   116
                                                                                         └─┘
                                                                                         ▲Tag Character

LINE   02   00.CUSTOMER-CITY                          13          13                               20    32
            00.ST                                     14          14                               36    49
            01.ZIP-CODE                                5           5                               53    57
```

Following is a sample of the tenth page of the report.

```
       ┌──┐
REPORT │04│FIELD NAME                HDR1  HDR2  DTL   ACC   TOTAL       PICTURE           TAG   FROM   TO
       └──┘
        ▲Fourth report description

LINE   01   00.CUSTOMER-NAME              4           24          24                                1    24
            01.CUSTOMER-NUMBER            8     6      3           8                               33    40
            00.CUSTOMER-CITY              4           13          13                               49    61
            00.ST                         5     4     14          14                               70    83
            01.ZIP-CODE                   3     4      5           5                               92    96
            01.CUSTOMER-SALE-ID           8     2      2           8                              105   112
            01.CREDIT-LIMIT               6     5     11          11         $$$,$$9.99           122   132
```

Run Diagnostics

The Run Diagnostics page prints after the selection phase of the program is performed. It displays statistics showing what has taken place during the execution.

```
RUN DIAGNOSTICS


          INDEX VIOLATIONS       00000000000
          PROGRAM CHECKS         00000000000
          PRIMARY I/P RECORDS    00000000016
          HITS FOR REPORT   1  - 00000000016
          HITS FOR REPORT   2  - 00000000016
          HITS FOR REPORT   3  - 00000000016
          HITS FOR REPORT   4  - 00000000016
          TOTAL RECORDS SELECTED 00000000064
          RECS DROPPED BY USER   00000000000
```

INDEX VIOLATIONS

INDEX VIOLATIONS indicates the number of times an index violation occurred during the reference to an indexed entry in an array. An index violation is the condition in which a relative reference is made to an array element. When an index violation occurs, processing is continued based on the setting of the OPTION INDEXERR= parameter.

PROGRAM CHECKS

Reflects the number of internal program check interruptions that have occurred and were fixed by the Reporting Facility.

PRIMARY I/P RECORDS

Shows the number of input records from the Primary file.

HITS FOR REPORT (1 through n)

Specifies a series of counters showing the number of records selected for each unique report defined in the run.

TOTAL RECORDS Selected

Specifies the total number of records created on the internally generated Hit File.

RECS DROPPED BY USER

Specifies the total number of records dropped prior to being tested against the selection criteria, that is, the number of times a GOTO START command was encountered during the processing run. This diagnostic is reported only when records have been dropped.

Chapter 7: Accessing CA Datacom/DB Tables with the Reporting Facility

The Reporting Facility interacts with CA Datacom/DB in three instances:

- When a CA Datacom/DB table is defined in the program
- When a CA Dataquery extract file is defined in the program
- When a COPYDD command is invoked and CA Datacom Datadictionary must be loaded

  Note: CA Datacom Datadictionary tables reside on a CA Datacom/DB database.

Interaction between CA Datacom Datadictionary and CA Datacom/DB occurs internally and is not discussed here.

In the other instances, you may be required to perform some preparatory functions to establish the environment in which CA Datacom/DB will be used. This chapter describes these functions.

User Requirements Table Processing

When you access a CA Datacom/DB table in a Reporting Facility program, a User Requirements Table is automatically supplied by the Reporting Facility. Select the option at compilation time using the DBCOMM= parameter of the OPTION command in the following command format:

```
►►─ OPTION ─ DBCOMM= ─┬─ phasename ─┬┬─────────┬┬───────────────┬────────────►◄
                      └─ AUTO ──────┘└ DBPRI= ┘└ DBSEQBUF(s)= ┘
```

AUTO

Specifies that the Reporting Facility will automatically generate the User Requirements Table entries.

phasename

Specifies the name of a precataloged User Requirements Table that contains entries for each CA Datacom/DB table to be accessed in a single run. The User Requirements Table is loaded from the system library and is used for all database access.

When the User Requirements Table is generated automatically, the Reporting Facility chooses specific default values to define the CA Datacom/DB environment. The system creates a single entry in the table, one for each table defined in the program.

- Supply the table name and database ID by using the FILE command.
- Assume that the CA Datacom/DB Multi-User Facility (MUF) is being used.
- Random and sequential access is allowed.
- File synonyms are allowed.
- Update is not allowed.
- Two sequential buffers are reserved for each table that is, DBSEQBUFS=2.

When the options and parameters chosen by the Reporting Facility are not sufficient to describe the CA Datacom/DB environment, use OPTION DBCOMM=phasename. This allows you to preassemble the User Requirements Table using any available options that specifically pertain to accessing the database tables.

z/OS Environment

In a z/OS operating environment, make your User Requirements Table available by using the following two procedures:

User Requirements Table Assembly

```
//jobname    See the note above and JCL Requirements.
//STEP1      EXEC ASMFC,PARM.ASM='NODECK,LOAD'
//ASM.SYSLIB DD   DSN=maclib name,DISP=SHR
//ASM.SYSGO  DD   DSN=objlib name(TEMP),DISP=SHR
//ASM.SYSIN  DD   *
         DBURSTR
         DBURTBL
            .
            .        URT macro parameters
            .
         DBUREND
          END
/*
//
```

Link-Edit to Load Library

```
//jobname    See the note above and JCL Requirements.
//STEP1        EXEC LKED
//LKED.SYSLIN  DD DDNAME=SYSIN
//LKED.SYSLMOD DD DSN=loadlib name,DISP=SHR
//LKED.OBJLIB  DD DSN=objlib name,DISP=SHR
//LKED.SYSIN   DD *
  INCLUDE OBJLIB(TEMP)
  NAME phasename(R)
/*
//
```

Note: For more information about macro parameters, see the *CA Datacom/DB Database and System Administration Guide*.

z/VSE Environment

In a z/VSE operating environment, follow these procedures:

User Requirements Table Assembly

```
* $$ JOB ...          See the note above and JCL Requirements.
* $$ LST ...
// JOB name
// OPTION DECK
// EXEC ASSEMBLY
      PUNCH ' CATALR phasename,1.0'
      DBURSTR
      DBURTBL
         .
         .       URT macro parameters
         .
      DBUREND
      END
/*
/&
* $$ EOJ
```

Link-Edit to Core Image Library

```
 * $$ JOB ...          See the note above and JCL Requirements.
 * $$ LST ...
 // JOB name
 // OPTION CATAL
    ACTION NOAUTO
    PHASE phasename,+0
    INCLUDE phasename
 // EXEC LNKEDT
 /*
 /&
 * $$ EOJ
```

Note: For more information about macro parameters, see the *CA Datacom/DB Database and System Administration Guide*.

Accessing Dynamic System Tables

- Access is limited to Dynamic System Tables in the following circumstances;
  - Excludes tables starting with SQL_ that are only accessible using SQL
  - RAAT commands REDKX, REDKG, and REDNX only
  - The MUFNAME field or DIRNAME field is forced to the MUF being used and the CXX being used
  - With the CA Datacom fix for COPYDD, you have access to simple fields for fields defined as SQL DATE, TIME, and TIMESTAMP

For a sample of Dynamic System Tables, see the Dynamic System Tables Sample (see page 294).

Note: For more information about excluded SQL tables, see the *CA Datacom/DB System Tables Reference Guide*.

Chapter 8: Accessing Standard Files with Reporting Facility

This chapter discusses accessing sequential files.

The GET command processes during the Reporting Facility compilation and a logic flow develops providing access to the input files.

A one-character alphanumeric field contains the return code from the GET command. You can reference the field by the same name as that specified as the file name on the FILE or INPUT command.

Sequential Files

The following file definition illustrates how to access a sequential file:

```
FILE1: FILE TAPE RECORD=100 BLOCK=1000 KEY=A,B,C
       DEFINE A 1-4 N
       DEFINE B 12-13 X
       DEFINE C 10-11 N
       DEFINE D 27-30 P2
```

In the discussions that follow, assume this file is in ascending sequence of field C within field B within field A.

GET Command

A GET command presents the next logical record from the file. Once a return code of E is posted, no further reading is done. The return code from the GET command is Y if a record has been retrieved; otherwise, the return code is E.

Full Key Argument

A GET command with a full key argument retrieves the next logical record that conforms to the logical expression supplied. Since the logical expression is considered to be a key argument, retrieval is on the basis of reading through the file until a match is found. This is an example of a logical expression:

```
FILE001: FILE TAPE RECORD=100  BLOCK=1000 KEY=A,B,C
          DEFINE FILEKEY 1 - N
         DEFINE A      1 - 4 N
         DEFINE B      5 - 6 N
         DEFINE C      7 - 9 N
           .
           .
           .
         DEFINE FILEKEY 1 - 9

         GET FILE001 FILEKEY = 123456789
```

If the key in a record is found to be lower than the argument, further records are read. If a record is found to be higher than the argument, the key argument has not been satisfied.

If the file contains duplicate key values, you must clear the key field after each read with a return code of Y to ensure that the next occurrence of the duplicate key will be read.

On the first GET command for this file, a record is read. On all subsequent GETs, the previous record is examined before another record is read. Once an END-OF-FILE return code E is posted, no further reading is done.

The return code from the GET command is Y if a record has been retrieved. The return code is N if no record with this key has been found or the record has already been passed. The return code is E if END-OF-FILE has been reached.

Full Key and Non-Key Argument

A GET command with full key as well as additional non-key arguments in the logical expression develops the same logical flow as the previous example, with the addition of the non-key argument comparison when a match is found.

The record is retrieved only if both the key and the non-key arguments are satisfied. If the non-key arguments are not satisfied, the system reads and examines further records until either a match is found or the record is found to be higher than the key argument.

The return code from the GET command is Y (RECORD-FOUND) if a record has been retrieved. Otherwise, it is N (NO-RECORD-FOUND) or E (END-OF-FILE).

Non-Key Argument

A GET command with one or more non-key arguments in the logical expression reads each record from the file until the non-key argument is satisfied. On the first GET command for this file, a record is read; on all subsequent GETs, the previous record is examined before another record is read.

Once an uppercase return code E is posted, no further reading is done. The return code from the GET command is Y (RECORD-FOUND) if a record has been retrieved. Otherwise, it is E (END-OF-FILE).

Chapter 9: Accessing IMS-DL/I Files with Reporting Facility

The Reporting Facility can be executed with a DL/I or DL/I-ENTRY database. A generalized DL/I user-module interface program is provided by CA. If DL/I support was coded during installation, the DL/I interface program is already cataloged in your Core Image Library or Load Library.

The GET command accesses the database files. The return code posted by DL/I as a result of the GET command is contained in a two-byte alphanumeric field within the 80-byte Communications Area for the DL/I file.

The DL/I interface sets up the appropriate DL/I-RETRIEVAL call based on the function code and the optional Segment Search Argument (SSA) passed to the module from the Reporting Facility program definition. The function code and the SSA, if applied, are checked for validity before the call is made to DL/I.

The Reporting Facility also searches the PCB address to ensure that a PCB is available if a database name has been supplied. Any error detected causes the Reporting Facility run to abort and an explanatory message to be issued.

Since the parameters are analyzed and verified prior to each call to DL/I, it is possible to access multiple databases using the same file definition. This can be accomplished by modifying the database name in the communication region.

The entire PSB is searched in an attempt to locate the proper PCB entry.

Successful execution of the DL/I call returns the contents of the requested database segment to the file I/O area in bytes 256 through n. In addition, 80 bytes of PCB information returned by the DL/I call are placed in the Communications Area. Any unsuccessful call to DL/I causes the Reporting Facility run to abort and an explanatory message to be issued.

Restrictions:

When working with a DL/I or DL/I-ENTRY database, these restrictions apply:

- Only one DL/I file definition per execution
- If defined, DL/I file must be the first file defined

- Only GET calls supported (first character of function code is G)
- Follow IBM rules for coding an SSA. Structure a qualified SSA as follows:

```
(key/sequence field name = search argument)
```

Note: In the SSA shown previously, the search argument is the comparative value against which the specified key/sequence field is to be checked and must therefore be the same length and format as the key/sequence field.

Processing Notes

You can specify only one DL/I file definition statement per Reporting Facility execution and it must be the primary file. Multiple databases can be in the primary file. You can access multiple databases using the single DL/I file definition, although you must use the same I/O area and Communications Area.

When you specify a DL/I file in a Reporting Facility run, the system accesses the database. Then, at end-of-job, control passes to the DL/I nucleus rather than the operating system.

In a typical Reporting Facility-DL/I run, the Reporting Facility program definition you set up establishes, prior to issuing the GET command, the function code, the number of SSAs supplied, the segment name, and the optional SSAs in bytes 1—255 of the I/O area.

The status code (PCBSC) returned from the DL/I call is evaluated by the Reporting Facility program. The Reporting Facility posts the return code.

Communications Area Use

In addition to the control information you specified in bytes 1—255 of the I/O area, the Reporting Facility automatically sets up an 80-byte Communications Area. This Communications Area returns the PCB information on completion of a GET command.

Information returned to the Reporting Facility by the 80-byte Communications Area has the following format:

Field Name	Position	Description
PCBPROPT	1—4	Processing option
PCBSC	5—6	Status code
PCBDBDNM	7—14	Database name
PCBLEV	15—16	Segment level feedback

Field Name	Position	Description
PCBSEGNM	17—24	Segment name feedback
PCBLFBK	25—28	Length of key feedback area (fullword binary)
PCBFBK	29—30	Key feedback area

Use the area with the return code from the GET command to determine the status of the I/O area. Reference the 80-byte Communications Area by the keyword DL1 and reference the return code by the file name.

Execution Requirements for DL/I (z/OS)

DL/I controls the execution of the Reporting Facility with DL/I in a z/OS environment. The EXEC statement in the job stream specifies the DL/I initialization module (DFSRRC00) rather than DRREPORT. On the PARM statement, specify the DRREPORT application program and the PSB it uses.

The following is an example of the JCL necessary to execute the Reporting Facility with DL/I (z/OS):

```
// jobname JOB ...
// stepname EXEC PGM=DFSRRC00,PARM=(DL1,DRREPORT,psbname...)
  .
  .                    Other required JCL statements,
  .                    Reporting Facility program definition statements,
  .                    Input file, if any
  .
/*
//
```

Code the format of the parameter as shown. The psbname is the name by which PSB was generated and is referenced by the application program DRREPORT. You can specify other parameters necessary for execution after the parameters discussed previously.

Execution Requirements for DL/I (z/VSE)

Executing the Reporting Facility is under the control of DL/I. The EXEC command in the job stream specifies the DL/I initialization module (DLZRRC00). You must specify the application program and the PSB it uses on a parameter statement immediately following the EXEC statement. Parameter statement format is:

Execution Requirements for DL/I (z/VSE)

```
►►─ DLI,DRREPORT,psbname,buffer ──────────────────────────────────────►◄
```

,psbname

The name by which the PSB was generated and is referenced by the application program, DRREPORT.

,buffer

The number of data subpools to reserve for executing JCL.

Several execution-time functions are controlled by the UPSI byte. If all bits are zero, omit the UPSI statement. This is required only if any of the following must be set ON or OFF:

0

Must always be zero

1—4

Not applicable

5

0= Storage dump on ABEND if bit 7=0
1= No storage dump on ABEND

6

0= DL/I system log tape active
1= DL/I tape log function inactive

7

0= Set exit linkage to DL/I for ABEND
1= Set exit (STXIT) inactive

Example

Note: Use the following as a guide to prepare your JCL. The JCL statements are for example only. Lowercase letters in a statement indicate a value you must supply. Code all statements to your site and installation standards.

```
// JOB ...
// UPSI 00000010
  .
  .                   Other required JCL statements
  .
// EXEC DLZRRC00,SIZE=nnnk
DL1,DRREPORT,psbname,buffer
  .
  .                   Reporting Facility program definition statements,
  .                   Input file, if any
/*
/&
```

Execution Requirements for DL/I-ENTRY

The Reporting Facility program, with the DL/I interface, executes as a subprogram of DL/I-ENTRY, passing control to a standard entry point specified in the application program. You must link edit the application program with DL/I-ENTRY and the appropriate PSB.

The following sample job stream can be used to link and execute the Reporting Facility with DL/I-ENTRY:

```
// JOB ...                    Link edit application program
// OPTION CATAL
   PHASE username,*           Any valid phase name
    INCLUDE IS08XXXI
    INCLUDE IS08Z03G          Modules cataloged to
    INCLUDE IS08Z06B          the relocatable library
    INCLUDE IS08Z07I          during installation of
    INCLUDE IS08Z10B          RPT Facility
    INCLUDE IJIFAZRZ
    INCLUDE DRDLIMOD
    INCLUDE DQDNUCA           DL/I-ENTRY nucleus/interface module
    INCLUDE psbname           Name of PSB referenced by this program
    ENTRY DQENT
    LBLTYP NSD(n)

// EXEC LNKEDT
/&
```

After performing the above link-edit, execute the Reporting Facility as follows:

```
// JOB ...
.
.                                  Other required JCL statements
.
// EXEC phasename,SIZE=nnnk        Same phase name as on above link edit
.
.                                  Reporting Facility program definition
.                                  statements,
.                                  Input file, if any
.
/*
/&
```

Chapter 10: Sample Reporting Facility JCL Streams

This chapter includes the JCL requirements for each mode of operation in the Reporting Facility system. Variations are possible, but the basic JCL must be intact to ensure successful execution of the prescribed functions.

Note: The samples on the following pages illustrate only the standard system data sets that are required. In Primary, Secondary, or Write-Only modes, additional JCL may be necessary to define the optional user input files.

z/OS Requirements

Multiple report generation (sort required):

Primary Mode

Note: Use the following as a guide to prepare your JCL. The JCL statements are for example only. Lowercase letters in a statement indicate a value you must supply. Code all statements to your site and installation standards.

```
//jobname    See the note above and JCL Requirements.
//        EXEC PGM=DRREPORT
//STEPLIB    See the note above and JCL Requirements.
//SYSPRINT  DD   SYSOUT=*
//SYSOUT    DD   SYSOUT=*
//DRWORK    DD   UNIT=SYSDA,SPACE=(TRK,10)
//SORTLIB   DD   DSN=SYS1.SORTLIB,DISP=SHR
//SORTIN    DD   DSN=&.&HITFILE.,DISP=(NEW,PASS),
//               UNIT=SYSDA,SPACE=(CYL,2)
//SORTOUT   DD   DSN=*.SORTIN,UNIT=SYSDA,SPACE=(CYL,2)
//SORTWK01  DD   UNIT=SYSDA,SPACE=(CYL,2,,CONTIG)
//SORTWK02  DD   UNIT=SYSDA,SPACE=(CYL,2,,CONTIG)
//SORTWK03  DD   UNIT=SYSDA,SPACE=(CYL,2,,CONTIG)
//SYSIN     DD   *
    .

    .                            Reporting Facility Source Commands
    .
END
    .
    .                            Optional Input File
    .
/*
//
```

If a CALL command is present in the source input stream, insert this JCL:

```
//DRLIB DD     DSN=dr.library.dsn,DISP=SHR
```

Single report generation (no sort allowed):

Secondary Mode

Note: Use the following as a guide to prepare your JCL. The JCL statements are for example only. Lowercase letters in a statement indicate a value you must supply. Code all statements to your site and installation standards.

```
//jobname    See the note above and JCL Requirements.
//        EXEC PGM=DRREPORT
//STEPLIB    See the note above and JCL Requirements.
//SYSPRINT DD   SYSOUT=*
//SYSOUT   DD   SYSOUT=*
//DRWORK   DD   UNIT=SYSDA,SPACE=(TRK,10)
//SYSIN    DD   *
     .
     .                           Reporting Facility Source Commands
     .
  END
     .
     .                           Optional Input File
     .
/*
//
```

If a CALL command is present in the source input stream, insert this JCL:

```
//DRLIB DD DSN=dr.library.dsn,DISP=SHR
```

Single output file creation (no sort allowed):

Write-Only Mode

Note: Use the following as a guide to prepare your JCL. The JCL statements are for example only. Lowercase letters in a statement indicate a value you must supply. Code all statements to your site and installation standards.

```
//jobname    See the note above and JCL Requirements.
//        EXEC PGM=DRREPORT
//STEPLIB    See the note above and JCL Requirements.
//SYSPRINT DD   SYSOUT=*
//SYSOUT   DD   SYSOUT=*
//DRWORK   DD   UNIT=SYSDA,SPACE=(TRK,10)
//DROUT    DD   DSN=output.data.set,DISP=(NEW,CATLG),
//              UNIT=SYSDA,SPACE=(TRK,10)
//SYSIN    DD   *
     .
     .                           Reporting Facility Source Commands
     .
  END
     .
     .                           Optional Input File
     .
/*
//
```

If a CALL command is present in the source input stream, insert this JCL:

```
//DRLIB      DD DSN=dr.library.dsn,DISP=SHR 1
```

Library Maintenance Mode

Note: Use the following as a guide to prepare your JCL.The JCL statements are for example only. Lowercase letters in a statement indicate a value you must supply. Code all statements to your site and installation standards.

```
//jobname    See the note above and JCL Requirements.
//       EXEC PGM=DRREPORT
//STEPLIB    See the note above and JCL Requirements.
//SYSPRINT DD   SYSOUT=*
//SYSOUT   DD   SYSOUT=*
//DRWORK   DD   UNIT=SYSDA,SPACE=(TRK,10)
//DRLIB    DD   DSN=dr.library.dsn,DISP=OLD
//SYSIN    DD   *
     .
     .                              Reporting Facility Library
     .
     .                              Maintenance Command Stream
  END
/*
//
```

z/VSE Requirements

Multiple report generation (sort required):

Primary Mode

Note: Use the following as a guide to prepare your JCL. The JCL statements are for example only. Lowercase letters in a statement indicate a value you must supply. Code all statements to your site and installation standards.

```
* $$ JOB ...           See the note above and JCL Requirements.
* $$ LST ...
// JOB name
// ASSGN SYS011,DISK,VOL=volser,SHR
// ASSGN SYS001,DISK,VOL=volser,SHR
// ASSGN SYS002,DISK,VOL=volser,SHR
// ASSGN SYS003,DISK,VOL=volser,SHR
// DLBL DRWORK,'data.set.name',0,DA
// EXTENT SYS011,volser,1,0,nnnn,nn
// DLBL SORTOUT,'data.set.name',0
// EXTENT SYS001,volser,1,0,nnnn,nn
// DLBL SORTIN1,'data.set.name',0
// EXTENT SYS002,volser,1,0,nnnn,nn
// DLBL SORTWK1,'data.set.name',0
// EXTENT SYS003,volser,1,0,nnnn,nn
// EXEC DRREPORT
    .
    .                            Reporting Facility Source Commands
    .
  END
    .
    .                            Optional Input File
    .
/*
//
* $$ EOJ
```

If a CALL command is present in the source input stream, insert this JCL:

```
// ASSGN SYS010,DISK,volser,SHR
// DLBL DRLIB,'data.set.name',0,DA
// EXTENT SYS010,volser,1,0,nnnn,nn
```

Single report generation (no sort allowed):

Secondary Mode

Note: Use the following as a guide to prepare your JCL. The JCL statements are for example only. Lowercase letters in a statement indicate a value you must supply. Code all statements to your site and installation standards.

```
* $$ JOB ...           See the note above and JCL Requirements.
* $$ LST ...
// JOB name
// ASSGN SYS011,DISK,VOL=volser,SHR
// DLBL DRWORK,'data.set.name',0,DA
// EXTENT SYS011,volser,1,0,nnnn,nn
// EXEC DRREPORT
     .
     .                          Reporting Facility Source Commands
     .
 END
     .
     .                                Optional Input File
     .
/*
/&
* $$ EOJ
```

If a CALL command is present in the source input stream, insert this JCL:

```
// ASSGN SYS010,DISK,volser,SHR
// DLBL DRLIB,'data.set.name',0,DA
// EXTENT SYS010,volser,1,0,nnnn,nn
```

Single output file creation (no sort allowed):

Write-Only Mode

Note: Use the following as a guide to prepare your JCL. The JCL statements are for example only. Lowercase letters in a statement indicate a value you must supply. Code all statements to your site and installation standards.

```
* $$ JOB ...           See the note above and JCL Requirements.
* $$ LST ...
// JOB name
// ASSGN SYS011,DISK,VOL=volser,SHR
// ASSGN SYSnnn,DISK,VOL=volser,SHR
// DLBL DRWORK,'data.set.name',0,DA
// EXTENT SYS011,volser,1,0,nnnn,nn
// DLBL DROUT,'data.set.name',0,SD
// EXTENT SYSnnn,volser,1,0,nnnn,n
// EXEC DRREPORT
     .
     .                           Reporting Facility Source Commands
     .
 END
     .
     .                           Optional Input File
     .
/*
/&
* $$ EOJ
```

If a CALL command is present in the source input stream, insert this JCL:

```
// ASSGN SYS010,DISK,volser,SHR
// DLBL DRLIB,'data.set.name',0,DA
// EXTENT SYS010,volser,1,0,nnnn,nn
```

Library Maintenance Mode

Note: Use the following as a guide to prepare your JCL. The JCL statements are for example only. Lowercase letters in a statement indicate a value you must supply. Code all statements to your site and installation standards.

```
* $$ JOB ...           See the note above and JCL Requirements.
* $$ LST ...
// JOB name
// ASSGN SYS011,DISK,VOL=volser,SHR
// ASSGN SYS010,DISK,VOL-volser,SHR
// DLBL DRWORK,'data.set.name',0,DA
// EXTENT SYS011,volser,1,0,nnnn,nn
// DLBL DRLIB,'data.set.name',0,DA
// EXTENT SYS010,volser,1,0,nnnn,nn
// EXEC DRREPORT
     .
     .                               Reporting Facility Library
     .
     .                               Maintenance Commands
 END
/*
/&
* $$ EOJ
```

Chapter 11: Reporting Facility Commands

This chapter provides a comprehensive description of all commands available in the Reporting Facility. The commands are arranged in alphabetical sequence for ease of reference.

For the purposes of this guide, the Reporting Facility program has been divided into seven functional areas.

Run Control

Area 1

Input File Definition

Area 2A

General Storage Area Definition

Area 2B

Data Manipulation/File Reading

Area 3

Report Definition

Area 4A

Output File Definition

Area 4B

Library Maintenance

Area 5

Each area has been assigned an ID (1, 2A, 2B,...).These reference IDs are used throughout this chapter to indicate the program areas to which a particular command applies.

For a discussion about the seven functional areas of a program, see Components of a Reporting Facility Program (see page 23).

ADD Command

(Area 3)

The ADD command algebraically increments a predefined numeric field by a specified numeric constant or the contents of another predefined numeric field.

The format of the ADD command is as follows:

```
►►─┬────────────┬─ ADD ─┬─ fieldname1 ───────┬─┬─ AND ─┬─ fieldname2 ──────►
   └─ label: ───┘       └─ numeric constant ─┘ └─ TO ──┘

►─┬──────────────────────────────┬─┬───────────┬─────────────────────────────►◄
  └─┬────────────┬─ fieldname3 ──┘ └─ ROUND ───┘
    └─ GIVING ───┘
```

label:

Specifies an identifying label that allows a GOTO branch to be made to this statement.

fieldname1

Specifies the name given a predefined numeric field whose value is to be added to the numeric value in *fieldname2.* Predefine the field name as zoned decimal, packed decimal, or binary.

fieldname2

Specifies the name given a predefined numeric field whose value is to be used as the base for the addition of two variables. Predefine the field name as zoned decimal, packed decimal, or binary.

numeric constant

Specifies an absolute numeric constant to add to the contents of *fieldname2*. It can be any valid numeric literal.

fieldname3

Optionally specifies a predefined numeric field in which the sum is placed. Predefine the field name as either zoned decimal or packed decimal.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

Example

```
          Predefined field name:     FIELD1     FIELD2     FIELD3     INDEX
                 Numeric precision:       3.2        1.4        3.4       2.0
         Initial value of field:    (123.45)   (5.4321)    (zero)      (3)

   ADD 1 TO FIELD1                   124.45         nc         nc        nc
   ADD FIELD2 TO FIELD3                  nc         nc     5.4321        nc
   ADD FIELD1 TO FIELD2
        GIVING FIELD3                    nc         nc   128.8821        nc
   ADD -123 TO FIELD1                  0.45         nc         nc        nc
   ADD 10 TO FIELD2
        GIVING FIELD3                    nc         nc    15.4321        nc
   ADD FIELD1 TO FIELD1              246.90         nc         nc        nc
   ADD INDEX TO FIELD3                   nc         nc     3.0000        nc
   ADD FIELD1 TO INDEX FIELD2            nc     6.4500         nc        nc
   ADD INDEX TO INDEX                    nc         nc         nc         6
   ADD -1 TO INDEX                       nc         nc         nc         2
   ADD +66 TO INDEX                      nc         nc         nc        69
   ADD .333 TO FIELD2
        GIVING FIELD1 ROUND            5.77         nc         nc        nc
```

Note the following:

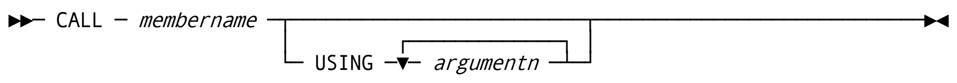
- The nc means no change in the original contents of the field.
- All significant digits of each of the operands take place in the addition, but the final result is aligned based on the DEFINE command options coded for the result field. In other words, significant digits or decimals can be lost in the final sum.
- The keywords TO and GIVING are optional.

CALL Command

(Areas 1, 2A, 2B, 3, 4A, 4B)

The CALL command enables you to dynamically include prestored input statements from the Reporting Facility call library at compilation time. This makes it unnecessary to recode long and complex groups of command sequences that are common to more than one Reporting Facility program. The Reporting Facility compiler fully edits each command within the library member each time it is included in the program.

The format of the CALL command is as follows:



membername

Specifies the name of the member contained in the Reporting Facility library to be included at the current point in the Reporting Facility source program.

argumentn ... argument30

Specifies from 1 to 30 arguments to be passed to the library member and substituted during inclusion. Each argument must be an alphanumeric literal, a numeric constant, or a valid field name. Terminate individual arguments with a blank.

Note the following:

- Library members can contain CALL commands which, in turn, can also contain CALL commands, up to nine levels in depth. After processing the entire member, the Reporting Facility returns to reading the input source statements.
- Specify library members with substitution codes identified as :01 to :30. The system substitutes the parameters following the reserved word USING on the CALL command on the basis of their position within the list of arguments. The first argument substitutes for each occurrence of the code :01, the second argument substitutes for each occurrence of the code :02, and so forth, up to the maximum specified.
- Do not specify Reporting Facility keywords and labels as USING clause arguments.
- You can pass substitution codes to be resolved by runtime USING arguments from level to level within a CALL by specifying subordinate CALL commands using the appropriate substitution codes.
- An overall maximum of 30 substitution codes, consisting of a maximum of 200 bytes of data, can remain active at any one time.
- Both the CALL command and the expanded member print on the compile listing if OPTION LIST ON is specified, but the system does not convert substitution codes within the member on the output compile listing.

Example

Assume that the following member was cataloged to the call library prior to its use in a Reporting Facility program:

```
LOAD  PCALC
DECODE :01 INTO :02 'A' EQ 1.72
                    'B' EQ 1.22
                    'C' EQ 1.63
                    'D' EQ 2.01
                    'E' EQ 2.43
                    OTHERWISE 2.00
COMPUTE :03 EQ ((:02 + :04)/100)
```

The following CALL command, **CALL PCALC USING CLASS RATE DISCOUNT 18.7**, would result as if the following statements were coded:

```
DECODE CLASS INTO RATE 'A' EQ 1.72
                           'B' EQ 1.22
                       'C' EQ 1.63
                       'D' EQ 2.01
                       'E' EQ 2.43
                       OTHERWISE 2.00
COMPUTE DISCOUNT EQ ((RATE + 18.7)/100)
```

COMPUTE Command

The COMPUTE command performs one or more arithmetic calculations in a specified order to yield a single numeric result. To provide maximum flexibility, the Reporting Facility supports two unique formats of the COMPUTE command.

- COMPUTE during record selection
- COMPUTE during report printing

The function of each type of command is basically the same, but format and placement of the different commands within the program vary slightly.

COMPUTE During Record Selection

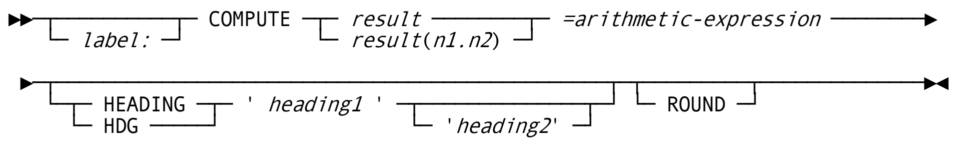
(Area 3)

Use this format of the COMPUTE command during record selection to specify one or more arithmetic operations to be performed on the specified numeric fields or constants and to place the result in the specified field.

You do not need to predefine the result field; it can be implicitly defined in the COMPUTE command itself with the result(n1,n2) parameter. However, you should predefine all fields referenced in procedural commands to minimize the margin of error.

Computation during record selection implies that the operation is performed prior to the sort and prior to the printing of reports. Therefore, the system uses fields from both the input buffer and the GSA as component parts of the arithmetic expression. When the computed value is then referenced by a field name in a subsequent report definition group, the system directs the value to the Hit File for processing.

The coding requirements for the COMPUTE during record selection command are:



label:

Specifies an optional identifying label that allows a GOTO branch to be made to this statement.

result

Specifies the name given a predefined numeric field in which the result of the computation is placed. Predefine the field name as zoned decimal, packed decimal, or binary.

result(n1.n2)

Specifies a user-supplied field name, coded using standard conventions, where the result of the computation is placed. This parameter implicitly defines the field to contain the result without predefining it.The field is generated in packed decimal format and resides in the GSA.

When the precision clause (n1.n2) does not follow the field name specification, the Reporting Facility automatically calculates the precision of the result based on the individual components of the arithmetic expression. The *n1* and *n2* indicate that the field will contain *n1* integers and n2 decimals.The value of n1+n2 must be greater than 0 and not exceed 15. When the field contains no integers, *n1* must be 0. When no decimals are necessary, n2 must be 0.

=arithmetic-expression

Specifies the arithmetic operations to be performed. The terms involved in the expression can be either numeric fields or constants, connected with arithmetic operators.

HEADING

Specifies an optional separator used to denote that the literals to follow are to be interpreted as headings. Heading specifications are valid only when the result field of the computation is being implicitly defined. HDG can be used in place of HEADING in the command.

'heading 1' 'heading 2'

Enables you to assign one or two lines of heading to the result field when the field is implicitly defined in the COMPUTE command. The lines of heading appear as column headings if the field is printed in a report. If the parameter is omitted and the result field is implicitly defined, the Reporting Facility automatically assigns the field name as heading line 1, with no second line of heading.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

Example

```
COMPUTE RESULT-FIELD EQ FIELDA + FIELDB
COMPUTE RESULT-FIELD(7.2) EQ FIELDA + .005 HDG 'AMOUNT'
COMPUTE FIELDA=((FIELDA+FIELDB)/FIELDC)*2
COMPUTE INDEX EQ INDEX+1
COMPUTE INDEX EQ INDEX - 1 ROUND
COMPUTE FIELDB=(((FIELDA + -2.367)*4) - (3*FIELDC)+.005)
```

Note: This format of the COMPUTE command cannot be specified in a report definition group.

COMPUTE During Report Printing

(Area 4A)

This format of the COMPUTE command, used during report printing, functions in much the same manner as the COMPUTE command during record selection. The intention is to calculate a numeric result based on an arithmetic expression. The basic differences are the timing of the execution of the calculation and the types of numeric fields that can be processed.

Since the system performs the calculations during the printing of the report, this format of the command can only appear within the report definition group. In addition, the computation performed pertains only to the particular report in which it is coded.

Calculations can be specified to be performed at detail time, total time, or both. When performing calculations at detail time, input to the expression can be from the GSA and from the Hit File. When performing calculations at total time, input can also be made from the field accumulators. This facility allows calculations to be made on the accumulated values of specific fields.

The format of the COMPUTE during report printing command is:

```
►►─ COMPUTE ─┬─ (D) ──┬─┬ result ──────────┬ =arithmetic-expression ────────►
             ├─ (T) ──┤ └ result(n1.n2) ───┘
             └─ (DT) ─┘

►─┬───────────────────────────────────────────┬─┬──────────┬──────────────────►◄
  └─┬─ HEADING ─┬─ ' heading1 ''heading2' ───┘ └─ ROUND ──┘
    └─ HDG ─────┘
```

D

Specifies that the computation is to be performed at detail time during printing of the report.

T

Specifies that the computation is to be performed at total time, that is, when a control break occurs.

DT

Specifies that the computation is to be performed at both detail time and total time.

result

Specifies the name given a predefined numeric field in which the result of the computation is placed. The field name should be defined as packed decimal only.

result(n1.n2)

Specifies a user-supplied field name, coded in accordance with standard conventions, in which the result of the computation is to be placed. Use of this parameter implicitly defines the field to contain the result without predefining the field.

When the precision clause (n1.n2) does not follow the field name specification, the Reporting Facility automatically calculates the precision of the result based on the individual components of the arithmetic expression.

When coded, *n1* and *n2* indicate that the field is to contain *n1* integers and n2 decimals, respectively. The value of n1+n2 must be greater than 0 and must not exceed 15. If you accumulate this field, the value must not exceed 15 or truncation will occur. When the field contains no integers, *n1* must be 0. When no decimals are necessary, n2 must be 0.

=arithmetic-expression

Specifies the arithmetic operations to be performed. The terms involved in the expression can be either numeric fields or constants, connected with arithmetic operators.

HEADING

Specifies an optional separator used to denote that the literals to follow are to be interpreted as headings. Heading specifications are valid only when the result field of the computation is being implicitly defined. HDG can be used in place of HEADING in the command.

'heading 1' 'heading 2'

Enables you to assign one or two lines of heading to the result field when the field is implicitly defined in the COMPUTE command. The lines of heading appear as column headings if the field is printed in a report. If the parameter is omitted and the result field is implicitly defined, the Reporting Facility automatically assigns the field name as heading line 1, with no second line of heading.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

Note the following:

- Since the calculation is performed just prior to report printing, fields referenced in the General Storage Area (GSA) contain their final presort value. The processing sequence within the Reporting Facility is:

  1. Read and process all input files.
  2. Sort.
  3. Print the reports.

  The system stabilizes the GSA at the end of the record selection phase. You can reference or modify the GSA by this type of command only after you perform the sort.

- The result field of the COMPUTE (D) or (T) can subsequently be referenced in another special COMPUTE or SET command or in a PRINT command. In each case, the current GSA value of the field is used. No reference to the Hit File or the presort GSA values is made.

- A field name specified within the arithmetic expression causes a unique reference to either a field in the Hit File or the GSA. The following are special coding considerations:

  - Fields in arithmetic expressions enclosed in parentheses indicate that the accumulated value is to be used in the calculation. If the accumulation is not specified, the Reporting Facility assigns zero to the field if it is used in a total time calculation.

- If you specify a literal in the arithmetic expression, you must prefix it with an ampersand (&).
- To use a GSA field in an arithmetic expression, you must first determine whether the calculation uses values from the HIT File (for example, EXTENSION = PRICE * QUANTITY). A column based calculation uses a GSA field that has been accumulated during program execution (ADD YTD-WAGES to TOTAL-WAGES), that references the final GSA value (COMPUTE(D) PERCENTAGE-OF-TOTAL = YTD-WAGES / &TOTAL-WAGES), and that the report definition includes a CONTROL statement even if no sort was actually required.

Example

```
COMPUTE(D) RESULT-FIELD EQ INPUT-FIELD * &100 ROUND 'HEADING1'

COMPUTE(T) RESULT-FIELD EQ (INPUT-FIELD) * &100

COMPUTE(DT) AVERAGE(7.2) EQ (((HOURS) / (CONTACTS)) + &.005) ROUND
```

CONDENSE Command

(Area 5)

The CONDENSE command initiates library reorganization procedures to remove logically deleted entries from the Reporting Facility call library and to reclaim any unused space. Perform regular reorganization procedures when considerable DELETE or LOAD activity is anticipated. The Reporting Facility automatically condenses the remaining library when less than 10 percent of the total library area is available for new entries.

The format of the CONDENSE command is:

```
►►─ CONDENSE ──────────────────────────────────────────────────────►◄
```

Note the following:

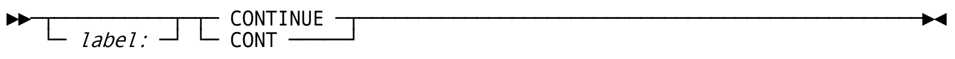
- The system restricts the use of the CONDENSE command to Library Maintenance mode only.
- The CONDENSE command can be one of multiple Library Maintenance commands in a single Reporting Facility run.

CONTINUE Command

(Area 3)

The CONTINUE command attaches a label to a nonprocedural statement. It does not move data, perform arithmetic calculations, or any other functions. CONT can be used in place of CONTINUE in a command.

The format of the CONTINUE command is as follows:



label:

Specifies an optional identifying label that allows a GOTO branch to be made to this statement.

A branch to a CONTINUE command shifts control to the procedural statement immediately following the CONTINUE command.

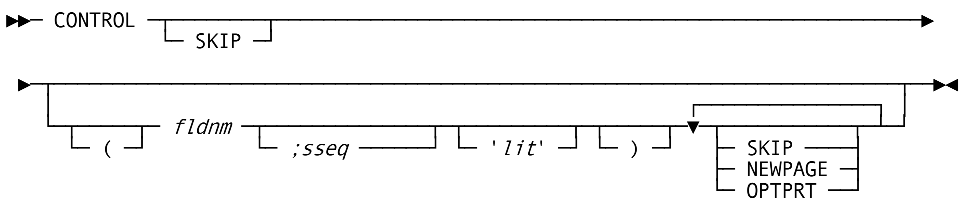
CONTROL Command

(Area 4A)

The CONTROL command provides two distinct functions:

- It specifies the sequence in which the data is to be presented for a particular report.
- It specifies those fields to be designated as control break fields for a particular report. A control break field is a user-defined data field. When the value of the data field changes from one input record to the next, a control break occurs.

The format of the CONTROL command is:



( )

Parentheses surrounding the field and associated parameters indicate that the field is a control break field and a sequence field. The parentheses must be paired.

fldnm

The field name specifies that the named field is to be a sort sequence field used to determine the final sequence of the output report for which this CONTROL command pertains.

;sseq

The sort sequence specifies that the individual field has a sort sequence. It is separated from the field name with a semicolon.

;A

Specifies that the field has an ascending sort sequence.

;D

Specifies that the field has a descending sort sequence.

'lit'

The literal specifies an alphanumeric literal to be used as the control break legend. You can specify a control break legend for any or all control break fields.

SKIP

Specifies one of two optional functions:

- When the SKIP parameter immediately follows the CONTROL command, the printer skips to a new page at the end of the report before printing the GRAND TOTAL or END OF REPORT lines.
- When the SKIP parameter follows a control break field, it indicates that the report is to continue on a new page after completing the printing of totals for the current level of the specified control break field.

NEWPAGE

Has the same function as SKIP when specified after a control break field, but also resets the page number on the output report to 1 when it causes a page eject on a control break.

OPTPRT

OPTPRT following a control break field indicates the control break field will print only when it changes, occurs after a heading break, or occurs at control total time.

Note the following:

- You must code fields on the CONTROL command from left to right, representing major to minor sorting sequence.
- You can intermix sequence-only fields and sequence/control break fields. The total combined size of all fields on the CONTROL command cannot exceed 256 characters. You can specify up to 15 levels of control breaks.
- If you omit the CONTROL command for a single report run (Secondary mode), no sort is performed and the data prints in the order in which the records are presented to the Reporting Facility. When you omit the CONTROL command from a report group in a multiple report run, the Reporting Facility generates a CONTROL command such that the data for that report prints in the order in which the records are presented to the Reporting Facility.
- Fields enclosed in parentheses cause control breaks to occur even when no sort is performed.
- The system permits only one SKIP or NEWPAGE parameter.
- When printing TOTALS ONLY, the system ignores any SKIP or NEWPAGE parameter at the lowest level of control break. Otherwise, since no detail lines are printed, each total line would appear on a new page.
- The specification of a sort sequence parameter (;A or ;D) applies to the immediately preceding field name only. If you require descending sequence for multiple fields, repeat ;D for each field.
- Field names specified on the CONTROL command can be of any type (except a bit string) and all rules of field name definition apply. In other words, field names can specify scalar variables or array elements, and field name qualification is supported where appropriate.
- Control break legends (literals) can be specified only for control break fields.
- The CONTROL statement must come before any COMPUTE or SET command or erroneous results can occur.

COPYDD Command

(Area 2A)

The COPY command provides an interface between the Reporting Facility and CA Datacom Datadictionary to achieve consistent definition of frequently referenced elements stored on the CA Datacom Datadictionary. You can use this command during the compile phase to dynamically include one element definition directly behind that COPYDD command.

The Reporting Facility assigns attributes and calculates the displacement of each field presented by CA Datacom Datadictionary. The system can then reference these fields in other Reporting Facility commands by specifying the generated field names on the DEFINE commands.

You can reference element definitions that belong to either CA Datacom/DB tables or non-database files (VSAM or tape). When multiple occurrences of the same element definition are contained in the dictionary, you can specify the version number of the definition desired (001 through 999) or the CA Datacom Datadictionary defined status of the element (PROD, TEST, or HIST).

Additionally, you can instruct the Reporting Facility to generate DEFINE commands with specific field names contained within the CA Datacom Datadictionary, that is, the Assembler or COBOL labels or the CA Datacom Datadictionary entity-occurrence names.

The format of the COPYDD command is:

```
►►─ COPYDD ─┬─ record-name ─┬─ element-name ─┬─ version ─┬──────────────────►
            └─ table-name ──┘                └─ status ──┘

►─┬──────────────────────┬──────────────────────────────────────────────────►◄
  └─ USE ─ label-type ───┘
```

record-name* or *table-name

Specifies the CA Datacom Datadictionary entity-occurrence name of the record or table to which the element being requested is associated. It is 1—32 characters long.

element-name

Specifies the CA Datacom Datadictionary entity-occurrence name of the element for which definition is requested. It is 1—32 characters long.

Note: For partitioned tables, you must use the table and element names for the full parent table, not one of the child (table partition) tables.

version

Indicates a version number of the element to be accessed. Valid entries are 1—999.

status

Indicates the generic CA Datacom Datadictionary status of the element to be accessed. The system uses the first entity-occurrence found with a matching status. Valid entries are PROD, TEST, and HIST.

label type

Indicates which label CA Datacom Datadictionary is to use when generating field names on the Reporting Facility DEFINE commands. The default is COMPILER.

The following list contains all valid label entries:

Assembler Label

ASSEMBLER
ASM

PL/I Label

PL1
PLI

Compiler Label

COMPILER
COM
COBOL
COB

Entity Label

ENTITY
ENT

Note the following:

- You can use multiple COPYDD commands to define a single file, one per element being accessed.
- When you are defining elements for a CA Datacom/DB table, the physical order of the COPYDD commands must correspond to the order of the CA Datacom/DB element list specified when a GET command is issued to the file.
- COPYDD commands cannot appear in cataloged library members.
- A FILE command, DEFINE command, or another COPYDD command must precede a COPYDD command.

Timestamp DEF Statement

CA Datacom Datadictionary and CA Datacom Reporting Facility supports printing individual fields that make up an SQL TIMESTAMP.

When the Reporting Facility requests a view of a table containing an SQL TIMESTAMP column, CA Datacom Datadictionary provides the SQL TIMESTAMP with a breakdown of simple fields that can be individually processed by the Reporting Facility. Currently, the Reporting Facility can only process the SQL TIMESTAMP as a single, binary-formatted field with a length of ten.

For example, the following SQL DDL statement defines a table containing an SQL TIMESTAMP (Column named MESSAGE_DATE):

```
CREATE TABLE SHORT_MESSAGE   (MESSAGE_ID      INTEGER,
                              RECIPIENT_NAME  CHAR(32),
                              MESSAGE_TEXT    CHAR(100),
                              MESSAGE_DATE    TIMESTAMP,
                              SENDER_NAME     CHAR(32)
                             );
```

A Reporting Facility definition generated for this table by CA Datacom Datadictionary looks like the following before Version 14.0:

```
LOAD        SHORTMSG
DEF SYSUSR-SHORT_MESSAGE            301-483 X
DEF MESSAGE_ID-NULL                 301-301 X
DEF MESSAGE_ID                      302-305 B
DEF RECIPIENT_NAME-NULL             306-306 X
DEF RECIPIENT_NAME                  307-338 X
DEF MESSAGE_TEXT-NULL               339-339 X
DEF MESSAGE_TEXT                    340-439 X
DEF MESSAGE_DATE-NULL               440-440 X
DEF MESSAGE_DATE                    441-450 B
DEF SENDER_NAME-NULL                451-451 X
DEF SENDER_NAME                     452-483 X
```

The same table definition now displays as follows:

```
LOAD        SHORTMSG
DEF SYSUSR-SHORT_MESSAGE            301-483 X
DEF MESSAGE_ID-NULL                 301-301 X
DEF MESSAGE_ID                      302-305 B
DEF RECIPIENT_NAME-NULL             306-306 X
DEF RECIPIENT_NAME                  307-338 X
DEF MESSAGE_TEXT-NULL               339-339 X
DEF MESSAGE_TEXT                    340-439 X
DEF MESSAGE_DATE-NULL                 440-440 X
DEF MESSAGE_DATE                      441-450 B
DEF MESSAGE_DATE-CC                   441-441 B
DEF MESSAGE_DATE-YY                   442-442 B
DEF MESSAGE_DATE-MM                   443-443 B
DEF MESSAGE_DATE-DD                   444-444 B
DEF MESSAGE_DATE-HH                   445-445 B
DEF MESSAGE_DATE-MI                   446-446 B
DEF MESSAGE_DATE-SS                   447-447 B
DEF MESSAGE_DATE-TS                   448-448 B
DEF MESSAGE_DATE-HS                   449-449 B
DEF MESSAGE_DATE-MS                   450-450 B
DEF SENDER_NAME-NULL                  451-451 X
DEF SENDER_NAME                       452-483 X
```

Each byte of the SQL TIMESTAMP is named using the name selected in the COPYDD statement or the –UTL LANGUAGE statement and includes one of the following three-character suffixes:

-CC	Century
-YY	Year
-MM	Month
-DD	Day of the month
-HH	Hour of the day
-MI	Minute of the hour
-SS	Second of the minute
-TS	Tenths of a second
-HS	Hundredths of a second
-MS	Thousandths of a second

Note: This functionality is only invoked for the definitions generated if the requested language is DR. A COBOL, PL1, or assembler copybook for the same table would not be expanded.

CREATE Command

(Area 5)

The CREATE command is valid only when processing in Library Maintenance mode. Use the CREATE command to preformat the library prior to first-time use of the Reporting Facility call library.

Invoke this command after installation of the Reporting Facility and never use it again unless the entire library file is to be re-created.

The format of the CREATE command is as follows:

```
►►─ CREATE ─ password ──────────────────────────────────────────►◄
```

password

Specifies the password, selected at product modification time, that identifies an authorized person to perform the CREATE function. It can be an alphanumeric string, not enclosed in apostrophes, up to eight characters in length.

Note the following:

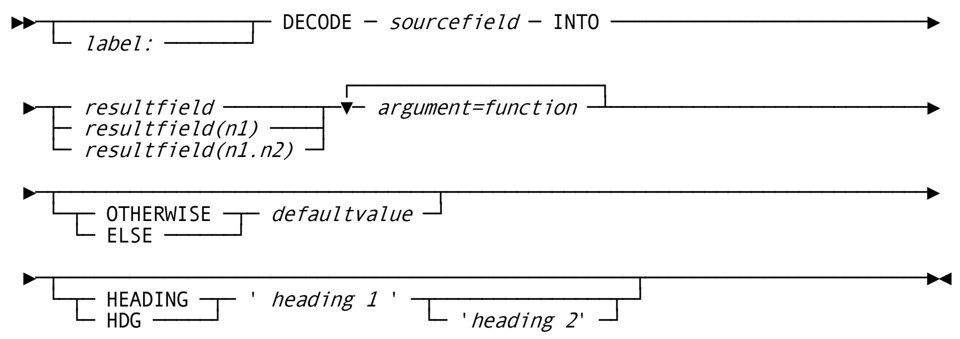
- Use of this command destroys any existing entries in the Reporting Facility call library.
- The CREATE command must immediately follow the LIBRARY command. The Reporting Facility allows only one CREATE command per run.
- If you move the Reporting Facility library file from one physical location or device to another, you must first preformat the new location using the CREATE command. Then, key and reload the old library data into the new library using standard library maintenance procedures.

DECODE Command

(Area 3)

The DECODE command translates a specified set of values from a source field into a different set of values in a result field.

The format of the DECODE command is as follows:



label:

Specifies an optional identifying label that allows a GOTO branch to be made to this statement.

sourcefield

Specifies the predefined field whose range of values are to be translated. The field can reside either in the input record area or the GSA, but you must predefine it.

resultfield

Specifies the name of the field into which the translated value is to be placed on completion of the DECODE sequence. When this field has not been predefined, The Reporting Facility generates the field automatically in the GSA.

resultfield(n1)

Specifies implicit definition of the result field as alphanumeric, *n1* bytes long. The result field must not be a predefined field. The result field resides on the GSA.

resultfield(n1.n2)

Specifies implicit definition of the result field as numeric, containing *n1* integers and *n2* decimals. The result field must not be a predefined field and resides in the GSA in packed decimal format.

argument

Specifies the value to be tested against the contents of the source field. Code the value as either a numeric constant or an alphanumeric literal, based on the attributes assigned to the source field.

=function

Specifies the value to be placed in the result field when the contents of the source field match the contents of the corresponding argument. The value must be coded as either a numeric constant or an alphanumeric literal based on the attributes assigned to the result field.

defaultvalue

Specifies the value to be placed in the result field after the system has exhausted the entire list of arguments and found no match. When coded, it must be either a numeric constant or an alphanumeric literal based on the attributes assigned to the result field.

HEADING

Specifies an optional separator used to denote that subsequent literals are to be interpreted as headings. Heading specifications are valid only when you are implicitly defining the result field of the transaction. HDG can be used in place of HEADING.

'heading 1' 'heading 2'

Enables you to assign one or two heading lines to the result field. You implicitly define the field in the DECODE command. Heading lines appear as column headings if the field prints in a report.

If the parameter is omitted and the result field is implicitly defined, the Reporting Facility automatically assigns the field name as heading line 1, with no second heading line.

Note the following:

- You can code up to a maximum of 50 argument and function specifications in a single DECODE command. At the time of execution, the system tests the value contained in the source field against the arguments, starting with the first one and proceeding sequentially through the entire list.
- When the result field is implicitly defined in the DECODE command and the precision is not specified, the Reporting Facility automatically calculates the size of the result field based on the size and precision of the largest function coded.
- All arguments specified in a single DECODE command must be of the same broad type, either numeric or alphanumeric, and must match the attributes of the source field. All functions must also be of the same broad type and match the attributes of the result field.
- You can code both the source field and the result field as a scalar variable or an array element.

Example

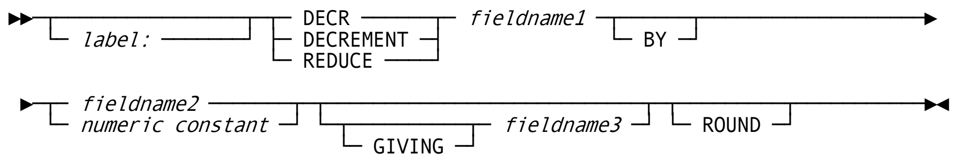
```
 DECODE CODE INTO GENDER                        Predefined result field
         1 EQ 'MALE'
        2 EQ 'FEMALE'
 DECODE TRANSACTION-CODE INTO TCODE(23)         Alphanumeric implicit
        'BF' EQ 'BALANCE BROUGHT FORWARD'         field definition
        'CK' EQ 'CHECK WRITTEN'
        'SC' EQ 'SERVICE CHARGE'
        'DP' EQ 'DEPOSIT'
        ELSE   'UNKNOWN TRANSACTION'
        'TYPE OF'  'TRANSACTION'
DECODE SOMETHING INTO SOMETHING-ELSE(2.3)       Numeric implicit
        'ABCDEF' EQ +12.345                       field definition
        'GHIJKL' EQ  -67.89
        'MNOPQR' EQ      0
        OTHERWISE      9.9
```

DECREMENT Command

(Area 3)

The DECREMENT command algebraically decrements the contents of a predefined numeric field by a specified numeric value or the contents of another predefined numeric field. DECR or REDUCE can be used in place of DECREMENT.

The format of the DECREMENT commands follows:



label:

Specifies an optional identifying label that allows a GOTO branch to be made to this statement.

fieldname1

Specifies the name given a predefined numeric field whose value represents the minuend for the subtraction of two variables. Predefine the field name as zoned decimal, packed decimal, or binary.

fieldname2

Specifies the name given a predefined numeric field whose value is to be used as the subtrahend for the subtraction of two variables. Predefine the field name as zoned decimal, packed decimal, or binary.

numeric constant

Specifies an absolute numeric constant to be used as the subtrahend in the calculation. It can be any valid numeric literal.

fieldname3

Optionally specifies a predefined numeric field in which the difference is placed. Predefine the field name as zoned decimal, packed decimal, or binary.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

Example

```
           Predefined field name:     FIELD1     FIELD2     FIELD3     INDEX
                  Numeric precision:       3.2        1.4        3.4       2.0
          Initial value of field:    (123.45)   (5.4321)      (1)       (10)

DECREMENT INDEX BY 1                    nc         nc         nc         9
DECREMENT INDEX BY FIELD3               nc         nc         nc         9
DECREMENT FIELD1 BY FIELD2
     GIVING FIELD3                      nc         nc   118.0179        nc
DECR FIELD1 BY -1.5                 124.95         nc         nc        nc
DECREMENT INDEX BY FIELD2
     GIVING FIELD3                      nc         nc     4.5679        nc
REDUCE INDEX BY INDEX                   nc         nc         nc         0
REDUCE FIELD3 BY 10                     nc         nc    -9.0000        nc
DECREMENT FIELD3 INDEX INDEX            nc         nc         nc        -9
DECREMENT FIELD2 BY FIELD3              nc     4.4321         nc        nc
DECREMENT INDEX BY 2.93                 nc         nc         nc         7
DECR FIELD1 BY +.92                 122.53         nc         nc        nc
REDUCE FIELD2 BY .333                   nc     5.0991         nc        nc
```

Note the following:

- The nc means no change in the original contents of the field.
- All significant digits of each of the operands take place in the subtraction, but the final result is aligned based on the DEFINE command options coded for the result field.
- The keywords BY and GIVING are optional.

DEFINE Command

(Areas 2A, 2B)

The DEFINE command assigns a symbolic name to a field or array to be used by the Reporting Facility program. You can locate the field or array in one of the input records or in the internally generated GSA. You can also use the DEFINE (or DEF) command to redefine all or part of an existing field or array.

DEFINE Command Format 1

(Area 2A)

Format 1 of the DEFINE command is used to explicitly define a field or array within an input file and should immediately follow the FILE command to which the fields are being defined.

Sample DEFINE Command Format 1

```
►►─┬─ DEFINE ─┬─ fieldname ─┬─ n1 ─ THRU ─ n2 ─┬─ data-type ──────────────────►
   └─ DEF ────┘             └─ n1 ─────────────┘

►─┬──────────────────────────────────────────────────┬───────────────────────►
  └─┬─ HEADING ─┬─ ' heading 1 ' ─┬──────────────────┤
    └─ HDG ─────┘                 └─ 'heading 2' ──┘

►─┬─────────────────────────────────────┬─┬──────────────────────────┬───────►◄
  └─┬─ PICTURE ─┬─ ' edit ptn ' ─┘ └─ OCCURS ─ n ─┬──────────┤
    └─ PIC ─────┘                                 └─ TIMES ─┘
```

fieldname

Is a user-supplied name assigned to the location specified in the DEFINE command. When accompanied by an OCCURS *n* TIMES clause, this field name represents the name of each array element.

n1* THRU *n2

Indicates, in bytes, the relative starting and ending positions of the field in a record of the file. When the OCCURS *n* TIMES clause is used, *n1* THRU *n2* specifies the starting and ending positions of the first array element within the record. The array element occurs *n* times in contiguous locations.

TO or – (dash) can be used in place of THRU.

n1

Indicates that the field is one byte long and integer n1 is the position of the field within the record.

data-type

Describes the data type of the field. Use the following entries:

X

Indicates that the field contains alphanumeric data and can contain any EBCDIC character.

H

Works the same as data-type X for moves and compare operations but on a PRINT statement displays in HEX. For example, A12 would display as C1F1F2 on the print line. Maximum length is 62.

N(*d*)

Indicates that the field contains zoned decimal data. d indicates the number of implied decimal places.

P(*d*)

Indicates that the field contains packed decimal data. d indicates the number of implied decimal places.

B(*d*)

Indicates that the field contains binary data. d indicates the number of implied decimal places.

S

Indicates that the field contains bit string data and must be only one byte long.

HEADING of HDG

Specifies an optional separator indicating the literals that follow it are interpreted as headings.

'heading 1' 'heading 2'

Enables you to assign one or two heading lines to the field. Heading lines appear as column headings if the field is printed in a report. If you omit the parameter, the Reporting Facility assigns the field name as heading 1, with no second heading line.

PICTURE '*edit ptn*'

Assigns an edit mask to the field when it is printed, overriding the default editing. PIC is the short form of PICTURE.

OCCURS *n* TIMES

Indicates that this is an array definition and the array element being defined occurs *n* times in contiguous locations in the record.

Example

Assume the following for an input file. The sample DEFINE commands following the input file illustrate techniques for defining individual fields that are accessible to the Reporting Facility.

1-2

A 2-character transaction code

3-6

A 4-character ID number

7-13

A 7-character (5 integers, 2 decimals) zoned decimal AMOUNT field.

14-21

An 8-character DATE field in the format of MM/DD/YY

22

A 1-character indicator showing whether the transaction is cleared

23-42

A 20-character PAYEE field

43

A 1-character filler

44-75

A 32-character REASON field

```
FILE ...
DEFINE  TRANSACTION-CODE       1-2     X
DEFINE  ID-NUMBER              3-6     X      'CHECK'    'NUMBER'
DEFINE  AMOUNT                7-13    N2      'TRANSACTION'   'AMOUNT'
                                              PICTURE '$$,$$9.99-'
DEFINE ACTIVITY-DATE         14-21     X      'DATE  OF'  'ACTIVITY'
DEF    CLEARED                  22     X
DEF    PAYEE              23 TO 42     X
DEFINE REASON             44 THRU 75   X
```

DEFINE Command Format 2

(Area 2B)

Format 2 of the DEFINE command defines a field or array within the General Storage Area (GSA). The GSA, which the system creates automatically, is basically a scratch pad that you can use to define standard literals or tables of data for use in the program. (The GSA can be equated to a COBOL Working Storage section.) The system also uses the GSA for immediate storage of special work areas during the compilation process.

Sample DEFINE Command Format 2

```
►►─┬─ DEFINE ─┬─ fieldname ─┬────────────┬─ = ─┬─ num lit ──┬──────────────►
   └─ DEF ────┘             ├─ (n1.n2) ──┤     └─ 'string' ─┘
                            └─ (n1) ─────┘

►─┬───────────────────────────────────────────────┬───────────────────────►
  └──┬─ HEADING ─┬─ ' heading 1 ' ─┬──────────────┤
     └─ HDG ─────┘                 └─'heading 2'──┘

►─┬─────────────────────────────────────┬─────────────────────────────────►
  └──┬─ PICTURE ─┬─ ' edit ptn ' ───────┘
     └─ PIC ─────┘

►─┬───────────────────────────────────────────────────────────────┬───────►◄
  └─ OCCURS ─ n ─┬─────────┬─┤ VALUE and FILL Parameters ├────────┘
                 └─ TIMES ─┘
```

Expansion of VALUE and FILL Parameters

```
├─┬──────────────────────────────────────────┬────────────────────────────►
  └─ VALUE ──┬─ num lit 2-n ─────────────────┤
             ├─ 'string' 2-n ──────────────┤
             └─ X'hex literal' 2-n ────────┘

►─┬──────────────────────────────────────┬────────────────────────────────┤
  └─ FILL ──┬─ num lit-d ────────────────┤
            ├─ 'string'-d ─────────────┤
            └─ X'hex literal'-d ───────┘
```

fieldname

Is a user-supplied name in accordance with the Reporting Facility naming conventions that are assigned to the location specified in the DEFINE command. When accompanied by an OCCURS *n* TIMES clause, this field name represents the name of one array element.

n1.n2

Indicates that this is a packed decimal field that contains n1 integers and n2 decimals. The value (n1 + n2) cannot exceed 15.

n1

Indicates that the field is an alphanumeric field *n1* bytes long.

num lit 'string'

Assigns an initial numeric value to the field. If the field is part of an array, this is the initial value of the first element.

HEADING

Specifies an optional separator used to denote that subsequent literals should be interpreted as headings. It is used primarily in conjunction with the VALUE clause. HDG can be used in place of HEADING.

'heading 1' 'heading 2'

Enables you to assign one or two heading lines to the field. Heading lines appear as column headings if the field is printed in a report.

PICTURE 'edit ptn'

Assigns an edit mask to be used to edit the field when it is printed, overriding the Reporting Facility default editing. PIC can be used in place of PICTURE.

OCCURS *n* TIMES

Indicates that this is an array definition. The array element being defined occurs *n* times in contiguous locations in the GSA.

VALUE

Optional separator used to denote that the subsequent literals are to be interpreted as initial values for array elements *2-n*. It is only valid when used in conjunction with the OCCURS clause, and must immediately follow either the OCCURS or FILL clause. VAL can be used in place of VALUE.

num lit 2-n

When defining a numeric array, this parameter assigns an initial numeric value to elements 2 through n, where *n* is the number of occurrences specified in the OCCURS *n* TIMES clause.

'string' 2-n

When defining an alphanumeric array, this assigns an initial alphanumeric value to elements 2 through n, where *n* is the number of occurrences specified in the OCCURS *n* TIMES clause.

X'*hex literal*' 2-*n*

When defining an alphanumeric array, this assigns an initial hexadecimal value to elements 2 through n, where *n* is the number of occurrences specified in the OCCURS *n* TIMES clause.

FILL

The FILL clause initializes (places) the FILL literal in each of the remaining occurrences that were not initialized by the VALUE clause. It is valid only when used in conjunction with the OCCURS clause and must immediately follow either the OCCURS or VALUE clause.

num lit-d

When defining a numeric array, this parameter specifies the numeric FILL value to elements *2-n* when the initial values are not specified. d indicates the number of implied decimal places.

'string'-d

When defining an alphanumeric array, this parameter specifies the alphanumeric FILL value to elements *2-n* when the initial values are not specified.

X'*hex literal*'-d

When defining an alphanumeric array, this parameter specifies the hexadecimal FILL value to elements *2-n* when the initial values are not specified. The literal must contain an even number of valid hexadecimal digits. d indicates the number of implied decimal places.

DEFINE Command Format 3

(Areas 2A, 2B)

Format 3 of the DEFINE command enables you to redefine all or part of an existing field or array element. This command defines:

- All or part of a scalar variable as a sub-scalar variable
- All or part of a scalar variable as an array
- All or part of an array element as a sub-array element

Sample DEFINE Command Format 3

```
►►─┬─ DEFINE ─┬─ fieldname1=fieldname2 ─┬─ n1 ─ THRU ─ n2 ─┬─ data-type ───►
   └─ DEF ────┘                         └─ n1 ─────────────┘

►─┬───────────────────────────────────────────┬────────────────────────►
  └─┬─ HEADING ─┬─ ' heading 1 ' ─┬──────────────┬┘
    └─ HDG ─────┘                 └─ 'heading 2'─┘

►─┬──────────────────────────┬─┬──────────────────────────┬──────────►◄
  └─ PICTURE ─ 'edit ptn' ─┘ └─ OCCURS ─ n ─┬───────────┬┘
                                              └─ TIMES ─┘
```

fieldname1

Is a user-supplied name assigned to the location specified in the DEFINE command.

fieldname2

Is the predefined name of a field or element of an array which you must redefine.

n1* THRU *n2

Indicates, in bytes, the relative starting and ending positions, respectively, of the field within the predefined *fieldname2*.

TO or – (dash) can be used in place of THRU.

n1

Indicates that the field is one byte long and that integer *n1* is the position of the field.

data-type

Describes the data type of the field. Use the following entries:

X

Indicates that the field contains alphanumeric data and can contain any EBCDIC character.

H

Works the same as data-type X for moves and compare operations but on a PRINT statement displays in HEX. For example, A12 would display as C1F1F2 on the print line. Maximum length is 62.

N(d)

Indicates that the field contains zoned decimal data. d indicates the number of implied decimal places.

P(d)

Indicates that the field contains packed decimal data. d indicates the number of implied decimal places.

B(d)

Indicates that the field contains binary data. d indicates the number of implied decimal places.

S

Indicates that the field contains bit string data and must have a length of only one byte.

HEADING

Specifies an optional separator used to denote that the literals that follow are to be interpreted as headings. HDG can be used in place of HEADING.

'heading 1' 'heading 2'

Enables you to assign one or two heading lines to the field. The heading lines appear as column headings if the field is printed in a report.

PICTURE 'edit ptn'

Assigns an edit mask to be used to edit the field when it is printed, overriding the Reporting Facility default editing. PIC can be used in place of PICTURE in the command.

OCCURS *n* TIMES

Indicates that this is an array definition occurring *n* times in contiguous locations beginning with the first byte of *fieldname1*. Using the keyword TIMES is optional, but recommended for documentation purposes.

DELETE Command

(Area 5)

The DELETE command logically deletes a specified member from the Reporting Facility call library. The system does not physically delete the library member, but marks it for deletion later when library reorganization takes place (CONDENSE function). The space on the Reporting Facility library that is marked for deletion is not available for reuse until a reorganization takes place.

If a password is required for deletions at your site, the password must be supplied on the USER statement.

The format of the DELETE command is as follows:

```
►►─ DELETE ─ membername ──────────────────────────────────────────►◄
```

membername

Specifies the name of the member to delete from the call library.

Note the following:

- The system restricts use of the DELETE command to the Library Maintenance mode only.
- The DELETE command can be one of multiple Library Maintenance commands in a single Reporting Facility run.

DISPLAY Command

(Area 5)

The DISPLAY command prints or punches the specified contents of the Reporting Facility call library.

The format of the DISPLAY command is as follows:

```
►►─ DISPLAY ──┬─ ALL ──────────┬─┬─────────┬────────────────────────────────────►◄
              ├─ INDEX ────────┤ └─ PUNCH ─┘
              └─ member-name ──┘
```

ALL

Indicates that every member in the library is to be printed in its entirety. Each member to be listed begins on a new page.

INDEX

Causes a listing of all member names and their associated comment entries, if present. The member names are listed in the order in which they were added to the Reporting Facility library.

member-name

Causes the printing of only the member name specified.

You can use a generic name for the member name. A generic name consists of a maximum of seven characters and ends with a period. All members whose leftmost node of the member name matches the supplied generic name are listed in their entirety.

PUNCH

Specifies that each member is to be punched, together with its appropriate LOAD card, into 80-column card images on SYSPCH/SYSPUNCH.

Note the following:

- The system restricts use of the DISPLAY command to the Library Maintenance mode only.
- The DISPLAY command can be one of multiple Library Maintenance commands in a single Reporting Facility run.

DIVIDE Command

(Area 3)

The DIVIDE command algebraically divides a predefined numeric field by a specified numeric constant or the contents of another predefined numeric field.

The format of the DIVIDE command is as follows:

```
►►─┬──────────┬─ DIVIDE ─ fieldname1 ─┬──────┬─┬─ fieldname2 ──────┬──────────►
   └─ label: ─┘                       └─ BY ─┘ └─ numeric constant ─┘

►─┬─────────────────────────────┬─┬─────────┬──────────────────────────────►◄
  └─┬────────────┬─ fieldname3 ─┘ └─ ROUND ─┘
    └─ GIVING ─┘
```

label:

Specifies an optional identifying label that allows a GOTO branch to be made to this statement.

fieldname1

Specifies a predefined numeric field whose value represents the dividend for the division of two variables. Predefine the field name as zoned decimal, packed decimal, or binary.

fieldname2

Specifies the name given a predefined numeric field whose value is to be used as the divisor for the division of two variables. Predefine it as zoned decimal, packed decimal, or binary.

numeric constant

Specifies an absolute numeric constant to be used as the divisor in the calculation. It can be any valid numeric literal.

fieldname3

Specifies a predefined numeric field in which the quotient is placed. Predefine it as zoned decimal, packed decimal, or binary.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

Example

```
         Predefined fieldname:      FIELD1     FIELD2     FIELD3   INDEX
               Numeric precision:      3.2        1.4        3.4     2.0
       Initial value of field:     (246.90)   (6.0000)    (zero)    (32)

   DIVIDE FIELD1 BY 2               123.45         nc         nc      nc
   DIVIDE FIELD2 BY INDEX               nc     0.1875         nc      nc
   DIVIDE FIELD1 BY FIELD2
        GIVING FIELD3                   nc         nc    41.1500      nc
   DIVIDE FIELD1 BY -24             -10.28         nc         nc      nc
   DIVIDE FIELD2 BY 3
        GIVING FIELD3                   nc         nc     2.0000      nc
   DIVIDE FIELD1 BY FIELD1            1.00         nc         nc      nc
   DIVIDE FIELD3 BY INDEX               nc         nc     0.0000      nc
   DIVIDE FIELD1 INDEX FIELD2           nc     7.7156         nc      nc
   DIVIDE INDEX BY FIELD2               nc         nc         nc       5
   DIVIDE INDEX BY 16                   nc         nc         nc       2
   DIVIDE FIELD1 BY +.92            268.36         nc         nc      nc
   DIVIDE FIELD2 BY .333                nc     8.0180         nc      nc
```

Note the following:

- The nc means no change in the original contents of the field.
- All significant digits of each of the operands take place in the division, but the final result is aligned based on the DEFINE command options coded for the result field.
- The keywords BY and GIVING are optional.

EJECT Command

(Areas 1,2A,2B,3,4A,4B)

The EJECT command is a compiler control command, processed only if OPTION LIST ON is specified. That is, the Reporting Facility compile listing is to print. Each invocation of this command forces a new page on the compiler listing.

The format of the EJECT command is as follows:

```
►►─ EJECT ────────────────────────────────────────────────────────────►◄
```

Note the following:

- The EJECT command must be the only command specified on the entire line.
- Use this command in conjunction with the SKIP and NOTE commands to properly document each program.

END Command

(Areas 4A,4B,5)

The END command terminates the Reporting Facility program. This signals completion of the final Reporting Facility command. That is, it is the end of the compilation phase or Library Maintenance command sequence. Only one such END command can appear in any Reporting Facility execution.

The format of the END command is as follows:

```
►►─ END ─────────────────────────────────────────────────────────────────►◄
```

Note the following:

- If an END command is not found, the Reporting Facility generates one automatically.
- When an input file is being processed, the END command is required. The system places the command between the last Reporting Facility command and the first data file. This is because both Reporting Facility commands and input files are read from the same physical data set, that is, SYSIN or SYSIPT.

ENDPROC Command

(Area 3)

The ENDPROC command signals the Reporting Facility to terminate a subroutine. The command must be paired with a PROC command. Any valid Reporting Facility (Area 3) commands can be placed between the PROC and ENDPROC commands.

The format of the ENDPROC command is as follows:

```
►►─┬──────────┬─ ENDPROC ────────────────────────────────────────────────►◄
   └─ label: ─┘
```

label:

Specifies an optional identifying label that may be referenced in a GOTO command within the limits of the subroutine.

Note the following:

- The PROC and ENDPROC commands must be the last group of statements immediately before the REPORT section.

FILE Command

(Area 2A)

The FILE command defines the physical attributes of an input data file or table to the Reporting Facility program. An input file is a collection of data, stored in computer-readable format, that the Reporting Facility can access and manipulate. Files can reside on different physical devices and can require different access methods. INPUT can be used in place of FILE.

The types of files that the Reporting Facility currently supports are:

- CA Datacom/DB tables
- CA Dataquery DQF tables
- VSAM files
- Disk files (including sequential, fixed, and variable)
- Tape files (including fixed and variable)
- Card files
- IMS-DL/I database files
- User modules

Since each specific type of input file requires a special format of the FILE command, this guide discusses each type individually in the following pages.

Note the following:

- The first file or table defined to a Reporting Facility program is known as the primary file. The Reporting Facility automatically generates the commands necessary to access the primary file except for CA Datacom/DB tables or IMS-DL/I files. For these files, you must initialize the Communications Area before issuing the GET command.
- You can define any number of input files or tables in a single execution of the Reporting Facility. Code the secondary files in the same manner as the primary file. (You must code the commands necessary to access a secondary file.)
- The appropriate Format 1 DEFINE commands must follow each FILE command to define the individual fields. You need to define only those fields that are referenced in the Reporting Facility program.
- The system enforces certain restrictions on the type of input files being referenced. This manual documents these restrictions with the individual FILE commands.

CA Datacom/DB Table Definition

This format of the FILE command defines the physical attributes of a CA Datacom/DB input table. The command is coded as follows:

```
►►─ filename: ─┬─ INPUT ─┬─ DATACOM ─ RECORD=nnn ─┬──────────────────┬────────────►
               └─ FILE ──┘                         └─ NAME=dbname ─┘

►─┬──────────────┬─────────────────────────────────────────────────────────────►◄
  └─ DBID=dbid ─┘
```

filename:

Is a unique identifier within the Reporting Facility program. The Reporting Facility automatically generates a one-character alphanumeric field with the same name as the table name. It is used as a return code in conjunction with the GET command.

RECORD=*nnn*

Indicates the combined size of all the elements to be retrieved from the CA Datacom/DB table, plus an additional 300 characters used for communication with CA Datacom/DB. The system reserves the 300 characters of control information as the first 300 characters of the I/O record area.

The subsequent Format 1 DEFINE commands must define all data to be returned as being relative to position 301. The reserved 300-character area has a specific format, and you must complete it prior to any access to the database.

The following details the format of the I/O area as required by the CA Datacom/DB Interface program:

1-5

5-character CA Datacom/DB command

6-10

5-character key name by which access is requested

11-190

Maximum 180-character key value for specific key requests

191-285

Maximum 15-element list, specifying the CA Datacom/DB elements to be returned

286-292

7-character CA Datacom/DB record ID, in the format FFTTTRL

293-300

Filler, reserved for future use

301-nnn

Returned elements

NAME=*dbname*

Specifies the three-character CA Datacom/DB table name for the CA Datacom/DB table to be accessed. This parameter is optional. If omitted, the first three characters of the *filename* are used.

Note: For partitioned tables, the CA Datacom/DB Reporting Facilitysupports full parent tables orthe child (table partition) tables.

DBID=*dbid*

Specifies the numeric database ID in which the table to be accessed resides. If coded, it must specify a positive integer in the range 1—5000.

Note the following:

- The CA Datacom/DB Reporting Facility interface program performs all database table handling and the contents of the 300-character reserved portion of the I/O area directs processing.
- The CA Datacom/DB commands currently supported (in bytes 1-5 of the I/O area) are:

GSETL	LOCKL	LOCNK	REDKL
GETIT	LOCKX	LOCNX	REDKX
LOCBR	LOCKY	REDBR	REDNE
LOCKG	LOCNE	REDKG	REDNX

REDKY has been superseded by REDKX, REDLE is no longer required as a priming read, and REDID should not be used if the URI option is elected. The first GETITs will read the next record in key sequence unless and until another GSETL is encountered. The KEY= value in bytes 11—190 of the I/O area must accompany those commands that require a key value. Supply the key name in positions 6—10 of the I/O area for all CA Datacom/DB requests. It must match the key name defined in the Control File.

- Supply the element list in positions 191—285 of the I/O area prior to issuing a GET to the database table. You must initialize it according to normal CA Datacom/DB rules.

Example

```
 xxxxxs....
```

xxxxx

A five-character element name.

s

A one-character security code.

Both are defined in the Control File.

Note the following:

- Five consecutive blanks terminate the list.
- The internal record ID of the current record in the I/O buffer is always updated in positions 286-292 of the I/O buffer after each request to CA Datacom/DB is completed. The record ID is in the format FFTTTRL.
- When a GETIT command is the first command requested for a CA Datacom/DB table, the Reporting Facility automatically issues a GSETL command internally. This defaults to positioning at the front of the file.
- When an explicit GSETL command is encountered, the Reporting Facility first performs the GSETL, then performs an implied GETIT to retrieve the data record for the key value specified.
- When the system processes any LOC command, the Reporting Facility updates the appropriate values in the Communications Area of the I/O buffer, but does not return data to the I/O area.
- With the exception of sequential processing (GSETL/GETIT), you must follow the prerequisite hierarchy of commands.
- The system automatically generates a User Requirements Table (URT) that contains an entry for each CA Datacom/DB table to be accessed.
- Any unrecoverable errors that occur during access to a CA Datacom/DB table cause the Reporting Facility to stop processing at the time the error is detected. A standard message prints, showing the table on which the problem has occurred and the associated CA Datacom/DB return code.

CA Dataquery DQF Table Definition

This format of the FILE command describes the attributes of a CA Dataquery found table. Through techniques described in the CA Dataquery documentation, it is possible to perform an online query and store the extracted data as a member on the CA Dataquery found table. CA Dataquery supplies this function to allow batch applications access to data preselected from an online system such as CA Dataquery.

Remember that the CA Dataquery found table is a CA Datacom/DB table. The only way that the Reporting Facility is able to distinguish the difference between a standard CA Datacom/DB table and a CA Dataquery found table is that the first three characters of the table name of the CA Dataquery found table are DQF.

This FILE command is coded as follows:

```
►►─┬─ DQF ────────┬─┬─ INPUT ─┬─ DATACOM ─ ',parm1,parm2,parm3' ──────────►
   └─ DQF qual: ──┘ └─ FILE ──┘

 ►─ RECORD=nnn ─┬──────────────────┬─┬──────────────┬──────────────────────►◄
                └─ NAME=dbname ────┘ └─ DBID=dbid ──┘
```

DQF *qual:*

Is a unique file name within the Reporting Facility program. The characters DQF in positions 1—3 of the file name are required to identify this file as a CA Dataquery file. Use *qual* to provide a unique file name when multiple CA Dataquery tables are specified in a single Reporting Facility run.

A one-character alphanumeric field is automatically generated by the Reporting Facility with the same name as the file name (without the colon), and is used as a return code in conjunction with the GET command.

',parm1

Specifies the predefined, eight-character member name of the collection of data to be retrieved from the CA Dataquery table. It is assigned by the online CA Dataquery operator at the time the data is extracted.

,parm2

Specifies the CA Dataquery operator ID under which the member was created. This parameter is required and can contain a maximum of 32 characters.

,parm3'

Specifies the CA Dataquery password assigned when the member was created. This parameter is required only if a password was assigned when the collection was created. When coded, a maximum of nine characters can be supplied, and it must match the password assigned by the CA Dataquery operator.

RECORD=*nnn*

Specifies the total length of the data record to be retrieved from the DQF table. The additional 300 characters at the front of the I/O area, required for normal CA Datacom/DB table access, are not required for access to the CA Dataquery table.

NAME=*dbname*

Optionally specifies the three-character CA Datacom/DB table name needed to access the DQF table.

DBID=*dbid*

Specifies the numeric database ID in which the DQF table resides. When coded, it must specify a positive integer in the range 1—5000.

Note the following:

- For a more detailed description of the creation of the CA Dataquery found tables, see the CA Dataquery documentation on the EXTRACT function.
- Retrieve a record from the found table by issuing a GET command. The Reporting Facility allows only sequential access to the table. The Reporting Facility controls the access method internally and you issue no alternate CA Datacom/DB commands against the table.
- Although data collected from the online CA Dataquery system is a subset of a common CA Datacom/DB table, the Reporting Facility accesses the member as though it were a flat sequential file.
- You can access multiple extracted members in a single run of the Reporting Facility as long as you define and access each one separately.
- The database ID must be specified in DBID=*dbid* or in the DQF *qual*.

VSAM File Definition

This format of the FILE command defines to the Reporting Facility the attributes of a VSAM input file. It also accesses VSAM files organized as entry-sequenced data sets (ESDS), key-sequenced data sets (KSDS), or relative-record data sets (RRDS).

The VSAM File Definition command is coded as follows:

```
►►─ filename: ─┬─ INPUT ─┬─┬─ DRVSAM ─┬─┬──────────────────────┬──────────►
               └─ FILE ──┘ └─ VSAM ───┘ └─ 'defineparameters' ─┘

 ►─ RECORD=nnn ─────────────────────────────────────────────────────────►◄
```

filename:

Is a unique file name within the Reporting Facility program. The Reporting Facility automatically generates a one-character alphanumeric field with the same name as the file name (without the colon). It is used as a return code in conjunction with the GET command.

VSAM

Indicates that this is a VSAM input file. An 80-byte Communications Area is automatically allocated by the Reporting Facility to be used for communication between the VSAM interface program and the Reporting Facility. The field resides in the GSA and can be referenced by the field name VSAM. DRVSAM can be used in place of VSAM.

'defineparameters'

Specifies an initial value for the 80-byte Communications Area used to determine the VSAM file organization type.

RECORD=*nnn*

Specifies the length of the longest logical record in the VSAM file to be retrieved. This parameter is required and must be a positive numeric integer.

Note the following:

- The GET command accesses the VSAM input files. In the case of the primary file, the Reporting Facility accesses them automatically. A one-character alphanumeric field, referenced by the same name as the file name on the FILE command, contains the return code from the GET operation.

  During retrieval of records from a VSAM file, three conditions can occur:

  - RECORD-FOUND (return code Y)
  - END-OF-FILE (return code E)
  - NO-RECORD-FOUND (return code N)

- When the NO-RECORD-FOUND condition occurs, you must reinitialize the one-character alphanumeric return code to blank after the condition has been tested and prior to any subsequent operation.

- The Reporting Facility/VSAM interface program performs all VSAM file handling. The contents of the *'defineparameters'* (as specified on the FILE command for the file being accessed) directs processing.

- The maximum length of the '*defineparameters*' is 80 characters. The system analyzes its contents each time you perform a GET operation on the file.

ESDS Organization

Retrieval of records from VSAM files organized as ESDS is sequential only. Establish the '*defineparameters*' in the 80-byte Communications Area prior to any GET operation on the file.

Supply the following '*defineparameters*' for access to an ESDS VSAM file:

```
'ESDS,CAI,SEQ'
```

If the parameters CAI and SEQ are not specified, the Reporting Facility automatically supplies them as default values. Establish the Communications Area in one of the following two ways prior to access of the file:

- Specification of the '*defineparameters*' directly on the FILE command, providing an initial value for the area
- A SET or MOVE command with reference to the field *filename*.VSAM, where *filename* is the name specified on the FILE command

The following two Reporting Facility programs illustrate various techniques available for specification of ESDS VSAM file access. Note that both programs produce the same results, but different techniques are used.

```
       USER 'XYZ COMPANY, INC.'
MASTER: FILE VSAM 'ESDS' RECORD=100
       DEF  DIVISION                            1-2 X
       DEF  DEPARTMENT                          3-5 X
       DEF  EMPLOYEE-NUMBER                     6-9 N   'EMPNO'
       DEF  EMPLOYEE-NAME                     10-25 X
       REPORT 'ESDS SAMPLE 1'
       SELECT  ALL
       CONTROL EMPLOYEE-NUMBER
       PRINT   DIVISION DEPARTMENT EMPLOYEE-NUMBER
                                    EMPLOYEE-NAME
       END
       USER 'XYZ COMPANY, INC.'
MASTER: FILE VSAM RECORD=100
       DEF DIVISION                             1-2 X
       DEF DEPARTMENT                           3-5 X
       DEF EMPLOYEE-NUMBER                      6-9 N   'EMPNO'
       DEF EMPLOYEE-NAME                   10 THRU 25 X
       MOVE 'ESDS,CAI,SEQ' TO MASTER.VSAM
       GET MASTER
       GOTO EOJ WHEN MASTER EQ END-OF-FILE
       REPORT 'ESDS SAMPLE 2'
       SELECT  ALL
       CONTROL EMPLOYEE-NUMBER
       PRINT   DIVISION DEPARTMENT EMPLOYEE-NUMBER EMPLOYEE-NAME
       END
```

Note the following:

- The '*defineparameters*' must begin in position 1 of the 80-byte Communications Area prior to each GET operation and each parameter coded must be contiguous.
- Qualify any reference to the Communications Area only when you define multiple VSAM files defined in the same run; that is, *filename*.VSAM, where *filename* is the name specified on the FILE command.
- The system supplies the keyword DRVSAM on the FILE command for downward release compatibility. When used, the field name assigned to the Communications Area is DRVSAM instead of VSAM.

KSDS Organization

Retrieval of records from VSAM files organized as KSDS can be sequential, skip sequential, or direct, using a full or partial key. You can change the method of retrieval at any point within the Reporting Facility program by modification of the '*defineparameters*' associated with the file. This section describes each method of access and the associated required '*defineparameters*'.

Sequential Retrieval of KSDS Files:

You do not need to specify '*defineparameters*' for sequential access to a KSDS file. The Reporting Facility automatically assumes the parameters to be 'KSDS,KEY,SEQ'. However, you can supply '*defineparameters*' with one of the following:

- Specification of an initial value for the 80-byte Communications Area directly on the FILE command
- A SET or MOVE command with reference to the field *filename*.VSAM, where *filename* is the name specified on the FILE command.

The following is a Reporting Facility program illustrating the techniques necessary for KSDS VSAM sequential file access:

```
       USER 'XYZ COMPANY, INC.'
MASTER: FILE VSAM RECORD=100
       DEF  DIVISION                          1-2 X
       DEF  DEPARTMENT                        3-5 X
       DEF  EMPLOYEE-NUMBER                   6-9 N   'EMPNO'
       DEF  EMPLOYEE-NAME                    10-25 X
       REPORT 'KSDS SAMPLE 1'
       SELECT  ALL
       CONTROL EMPLOYEE-NUMBER
       PRINT   DIVISION DEPARTMENT EMPLOYEE-NUMBER EMPLOYEE-NAME
       END
```

Skip Sequential Retrieval of KSDS Files:

Retrieval of KSDS files using the skip sequential method is far more efficient than direct retrieval. When using this method, the first record made available is one that is equal to or greater than the specified key. The keys specified within the Reporting Facility program definition must be in ascending sequence.

Code the '*defineparameters*' necessary for skip sequential processing of a KSDS VSAM file as follows:

```
►►─ 'KSDS,KEY,SKP ──┬─────────────┬─ ' ──────────────────────────────────────►◄
                    └─ ,GEN(n) ───┘
```

,GEN(*n*)

Is the length of the generic key. You must specify it when you request a generic key.

The following Reporting Facility program illustrates the techniques necessary for KSDS VSAM skip sequential file access. Access is performed with a full key argument.

```
       USER 'XYZ COMPANY, INC.'
MASTER: FILE VSAM RECORD=100
       DEF FILE-KEY                             1-9 X
       DEF DIVISION                             1-2 X
       DEF DEPARTMENT                           3-5 X
       DEF EMPLOYEE-NUMBER                      6-9 N   'EMPNO'
       DEF EMPLOYEE-NAME                       10-25 X
       DEF FIRST-TIME-SWITCH(1)  EQ 'F'
       GOTO DO-SEQUENTIAL-ACCESS WHEN FIRST-TIME-SWITCH NE 'F'
       MOVE 'AA0019999' TO FILE-KEY
       MOVE 'KSDS,KEY,SKP' TO MASTER.VSAM
       GET  MASTER
       MOVE 'SEQ' TO MASTER.VSAM
       MOVE 'N' TO FIRST-TIME-SWITCH
       GOTO TEST WHEN MASTER NE END-OF-FILE
       MOVE BLANK TO MASTER
       GOTO TEST
DO-SEQUENTIAL-ACCESS:  CONTINUE
       GET MASTER
       GOTO EOJ WHEN MASTER EQ END-OF-FILE
       REPORT 'KSDS SKIP SEQUENTIAL SAMPLE 1'
       SELECT ALL
       CONTROL EMPLOYEE-NUMBER
       PRINT   DIVISION DEPARTMENT EMPLOYEE-NUMBER EMPLOYEE-NAME
       END
```

The following is a Reporting Facility program illustrating the techniques necessary for KSDS VSAM skip sequential file access. Access is performed with a generic key argument.

```
       USER 'XYZ COMPANY, INC.'
MASTER: FILE VSAM RECORD=100
       DEF FILE-KEY                              1-9 X
       DEF DIVISION                              1-2 X
       DEF DEPARTMENT                            3-5 X
       DEF EMPLOYEE-NUMBER                       6-9 N   'EMPNO'
       DEF EMPLOYEE-NAME                       10-25 X
       DEF FIRST-TIME-SWITCH(1) EQ 'F'
       GOTO DO-SEQUENTIAL-ACCESS WHEN FIRST-TIME-SWITCH NE 'F'
       MOVE 'AA' TO DIVISION
       MOVE 'KSDS,KEY,SKP,GEN(2)' TO MASTER.VSAM
       GET MASTER
       MOVE 'SEQ' TO MASTER.VSAM
       MOVE 'N' TO FIRST-TIME-SWITCH
       GOTO TEST WHEN MASTER NE END-OF-FILE
       MOVE BLANK TO MASTER
       GOTO TEST
DO-SEQUENTIAL-ACCESS:  CONTINUE
       GET MASTER
       GOTO EOJ WHEN MASTER EQ END-OF-FILE
       REPORT 'KSDS SKIP SEQUENTIAL SAMPLE 2'
       SELECT ALL
       CONTROL EMPLOYEE-NUMBER
       PRINT   DIVISION DEPARTMENT EMPLOYEE-NUMBER EMPLOYEE-NAME
       END
```

Direct Retrieval of KSDS Files:

To perform direct retrieval of KSDS files, use the full key argument or a generic key for the file. If this method does not make available the first record that contains the specified key, a NO-RECORD-FOUND condition occurs.

With direct retrieval using a generic key argument, the record made available is one that is equal to or greater than the specified key. The generic key length can be equal to the full key length.

Code the '*defineparameters*' necessary for direct retrieval of a KSDS VSAM file as follows:

```
►►─ 'KSDS,KEY,DIR ─┬──────────┬─ ' ─────────────────────────────────►◄
                   └─ ,GEN(n) ┘
```

,GEN(*n*)

Is the length of the generic key. You must specify it when you request a generic key.

The following Reporting Facility program illustrates the techniques necessary for KSDS VSAM direct file access. Access is shown here with a full key argument.

```
       USER 'XYZ COMPANY, INC.'
CARDIN: FILE CARD
       DEF CARD-KEY                              1-9 X
MASTER: FILE VSAM RECORD=100
       DEF FILE-KEY                              1-9 X
       DEF DIVISION                              1-2 X
       DEF DEPARTMENT                            3-5 X
       DEF EMPLOYEE-NUMBER                       6-9 N   'EMPNO'
       DEF EMPLOYEE-NAME                        10-25 X
       MOVE CARD-KEY to FILE-KEY
       GET MASTER
       SET MESSAGE(15) TO ' '
       GOTO TEST WHEN MASTER NE 'E'
       MOVE SPACE TO MASTER
       MOVE 'NO RECORD FOUND' TO MESSAGE
       REPORT 'KSDS DIRECT RETRIEVAL SAMPLE  1'
       SELECT 'A' WHERE FILE-KEY EQ CARD-KEY
       SELECT ALL
       CONTROL EMPLOYEE-NUMBER
       PRINT   CARD-KEY A;DIVISION A;DEPARTMENT A;EMPLOYEE-NUMBER
                                 A;EMPLOYEE-NAME MESSAGE
       END
```

RRDS Organization

Retrieval of records from VSAM files organized as RRDS can be sequential, skip sequential, or direct, using a full key argument only. Specification of a generic key is not permitted.

When using the skip sequential or direct retrieval methods, you must define the key as the first four bytes of the record I/O area. Other than the above restrictions, the rules of KSDS retrieval also apply to RRDS retrieval. Code the '*defineparameters*' necessary for relative record processing of a VSAM file as follows:

```
►►─ 'RROD,KEY, ──────────── ' ──────────────────────────────────────►◄
               ├─ SEQ ─┤
               ├─ SKP ─┤
               └─ DIR ─┘
```

The following Reporting Facility program illustrates the techniques necessary for RRDS VSAM sequential file access:

```
       USER 'XYZ COMPANY, INC.'
MASTER: FILE VSAM   'RRDS,KEY,SEQ'    RECORD=29
       DEF  RELATIVE-KEY-VALUE                        1-4 B
       DEF  DIVISION                                  5-6 X
       DEF  DEPARTMENT                                7-9 X
       DEF  EMPLOYEE-NUMBER                          10-13 N   'EMPNO'
       DEF  EMPLOYEE-NAME                            14-29 X
       REPORT 'RRDS SAMPLE 1'
       SELECT ALL
       CONTROL EMPLOYEE-NUMBER
       PRINT   DIVISION DEPARTMENT EMPLOYEE-NUMBER EMPLOYEE-NAME
       END
```

DISK File Definition

This format of the FILE command defines the physical attributes of an input disk file. The DISK File Definition command is coded as follows:

```
►►─ filename: ─┬─ INPUT ─┬─ DISK ─┬──────────┬─ RECORD=recsize ───────────────►
               └─ FILE ──┘        └─ device ─┘

►─┬───────────────────┬─┬─ VARIABLE ─┬─┬──────────────┬─┬──────────────┬──────►◄
  └─ BLOCK=blksize ───┘ └─ FIXED ────┘ └─ SEQUENTIAL ─┘ └─ KEY=string ─┘
```

filename:

Is a unique file name within the Reporting Facility program. You must terminate the file name with a colon, with no intervening blanks. CA Datacom/DB Reporting Facility automatically generates an alphanumeric field with the same name as the file name (without the colon). It is to be used as a return code in conjunction with the GET command.

device

Indicates the device type on which the input file resides. When this parameter is coded, it must immediately follow the disk keyword. If you omit this parameter, the Reporting Facility assumes that the file resides on either of the following:

- Default device type selected during installation
- Device type specified by OPTION DISK= parameter for this run

Valid device type codes are 3350, 3375, 3380, 3390 and 9345 (and FBA for z/VSE).

RECORD&equals.*recsize*

Specifies the length of each logical record in the input file. The coded value must be equal to or less than track capacity of the device.

BLOCK&equals.*blksize*

Specifies the length of each physical block in the input file. If specified, it must be an even multiple of the record size. When this parameter is omitted, the Reporting Facility assumes the block size to be the same as the record size.

FIXED|VARIABLE

FIXED indicates that each record on the file is of equal length, as specified in the RECORD= parameter.

VARIABLE indicates that each record on the file can be of different lengths, but not exceeding the value coded in the RECORD= parameter. When a variable length file is defined, you must allow for the standard record descriptor word (four bytes) that is always returned in the first four positions of the I/O buffer.

SEQUENTIAL

Specifies the physical organization of the file as a sequential data set.

INDEXED(*n*)

Specifies the physical organization of the file as an indexed sequential (ISAM) data set. The length of the key defined for the file must either be coded in *n* or using the KEY= parameter. When *n* is specified, it must be a positive integer less than or equal to the value specified in the RECORD= parameter or 255, whichever is smaller.

KEY=*string*

Specifies the sequence in which the file is to be presented to the Reporting Facility. The string consists of the field names defined as comprising the key to the file.

Note the following:

- In a z/OS environment, at execution time, the appropriate values in the JCL can override the record size and block size.
- In a single run, you can define and access any number of disk files.
- The file name must match the file name in the JCL (in a z/OS environment, DDNAME; in a z/VSE environment, the DLBL name).

TAPE File Definition

This format of the FILE command defines the physical attributes of an input tape file. The TAPE File Definition command is coded as follows:

```
►►─ filename: ─┬─ INPUT ─┬─ TAPE ─ RECORD=recsize ─ UNIT=SYSnnn ──────────►
               └─ FILE ──┘

►─┬───────────────┬─┬────────────┬─┬──────────────────────────┬──────────►
  └─ BLOCK=blksize ┘ ├─ VARIABLE ─┤ └─ LABEL= ─┬─ STANDARD ─┬─┘
                     └─ FIXED ────┘            └─ NONE ─────┘

►─┬──────────────────────────┬─┬──────────────┬───────────────────────────►◄
  └─ REWIND= ─┬─ UNLOAD ─┬───┘ └─ KEY=string ─┘
              ├─ YES ────┤
              └─ NO ─────┘
```

filename:

Is a unique file name within the Reporting Facility program. The Reporting Facility automatically generates a one-character alphanumeric field to be used as a return code in conjunction with the GET command.

RECORD=*recsize*

Specifies the length of each logical record in the input file. The value coded here must be in the range of 1-32760.

UNIT=SYS*nnn*

(z/VSE only) Specifies the physical device address of the magnetic tape reel (or first reel of multivolume tape file). Code according to site standards.

BLOCK=*blksize*

Specifies the length of each physical block in the input file. If specified, it must be an even multiple of the record size. If omitted, block size is assumed to be the same as the record size.

VARIABLE|FIXED

VARIABLE indicates that each record on the file can be a different length, but may not exceed the value of the RECORD= parameter. When defining a variable length file, allow for the standard record descriptor word (four bytes) returned in the first four positions of the I/O buffer.

FIXED indicates that each record on the file is of equal length, as specified in the RECORD= parameter.

LABEL=NONE|STANDARD *(z/VSE only)*

STANDARD *(z/VSE only)* indicates that the tape file contains standard labels. If the LABEL= clause is omitted, this is the default.

NONE Indicates that the tape file is unlabeled.

REWIND=UNLOAD|YES|NO *(z/VSE only)*

UNLOAD specifies that the input tape file must be rewound and unloaded when closed. It is the default if REWIND= is omitted.

YES specifies that when the input tape file is closed, it must be rewound and left in a ready state (at the beginning of the file).

NO specifies that when the input tape file is closed, it is *not* rewound or unloaded, but left in a ready state at the position last read.

KEY=*string*

Specifies the sequence in which the file is presented to the Reporting Facility. The string consists of the field names defined as comprising the key to the file.

Note the following:

- In a z/OS environment, the JCL values can override the record size and block size for execution.
- You can define and access any number of tape files in a single run.
- The file name specified must match the file name in the JCL (in z/OS, DDNAME; in z/VSE, the DLBL name).
- In z/OS, the JCL specifies the disposition of the tape after the close.

CARD File Definition

This format of the FILE command defines input CARD file attributes:

```
►►─ filename: ─┬─ INPUT ─┬─ CARD ──────────────────────────────────────►◄
               └─ FILE ──┘
```

filename:

Is a unique file name within the Reporting Facility program, coded in accordance with standard file naming conventions. The file name must be terminated by a colon, with no intervening blanks. The Reporting Facility automatically generates a one-character alphanumeric field, with the same name as the file name. It is to be used as a return code in conjunction with the GET command.

Note the following:

- Only one CARD file can appear in a single Reporting Facility run.
- Assume the record size is 80.
- Since the control commands are read from SYSIN/SYSIPT, the input file must immediately follow the END command in the Reporting Facility program.
- Retrieve the next sequential record by issuing the GET command.

IMS-DL/I Database Definition

This format of the FILE command defines the physical attributes of an IMS-DL/I database input file. The IMS-DL/I Database Definition command is coded as follows:

```
▶▶─ filename: ─┬─ INPUT ─┬─ DLI ─ RECORD=nnn ──────────────────────────────▶◀
               └─ FILE ──┘
```

filename:

Is a unique file name within the Reporting Facility program. The Reporting Facility automatically generates a one-character alphanumeric field, with the same name as the file name. It is to be used as a return code in conjunction with the GET command.

DLI

Indicates that this is an IMS-DL/I file. An 80-byte Communications Area is automatically allocated by the Reporting Facility to be used for communication between the DL/I interface program and the Reporting Facility.

On completion of a GET command, the Reporting Facility automatically returns the PCB feedback information to this area in a specific format. Use it in conjunction with the return code to determine the status of the I/O area. The format is as follows:

Field Name	Position	Description
PCBPROPT	1—4	Processing option
PCBSC	5—6	Status code
PCBDBDNM	7—14	Database name
PCBLEV	15—16	Segment level feedback
PCBSEGNM	17—24	Segment name feedback
PCBLFBK	25—28	Length of key feedback (fullword binary)
PCBFBK	29—80	Key feedback area

For a complete description of the contents and utilization of the PCB feedback area, see the relevant IBM publication.

RECORD=*nnn*

Is an unsigned numeric constant specifying the length of the data portion of the DL/I file to be accessed, plus an additional 255 characters for communication with DL/I. The system reserves the 255 characters of control information to be the first 255 characters of the I/O area.

The data accessed from a DL/I file is returned starting at location 256. The subsequent Format 1 DEFINE commands must define all data to be returned as being relative to 256.

The following is a schematic of the format of the I/O area as required by the DL/I interface program:

1—4

4-character function code

5—6

2–byte packed decimal number, denoting the number of segment search arguments (SSAs) to follow (maximum value is ten)

7—14

8–character segment name, specifying the segment to be retrieved

15—255

Optional SSAs

256—nnn

Returned data

Note the following:

- The GET command accesses the DL/I files. In the case of the primary file, the Reporting Facility accesses the DL/I files automatically. A one-character alphanumeric field, which must be referenced by the same name as the file name on the FILE command, contains the return code from the GET operation.

  During retrieval of records from a DL/I file, two conditions can occur:

  - END-OF-FILE (return code E)
  - NO-RECORD-FOUND (return code N)

- Only GET type commands are supported in the function code bytes 1—4 of the I/O area.

- Where the PSB can contain several PCBs, the DL/I interface program accesses the correct PCB by referring to the contents of bytes 7—14 of the PCB feedback area prior to executing the GET command against the DL/I file.

  Three unique conditions can occur:

  - If bytes 7—14 are blank, the system uses the first PCB.
  - If bytes 7—8 are a valid packed decimal number in the range of 1—255, the system considers it to be the number of the required PCB within the PSB.
  - If bytes 7—14 are non-blank and non-numeric, the system considers it to be a database name and the PCBs are searched for a matching name.

The database name is returned in bytes 7—14 of the PCB feedback area after each DL/I call. You can, at any time, reference a different database simply by changing the contents of this field. You can use this method to access different databases even if only a single DL/I file is defined in the Reporting Facility program.

IBM imposes strict rules for coding SSAs and you must follow them precisely. A typical qualified SSA is structured as follows:

```
 (key/sequence fieldname = search argument)
```

The search argument is the comparative value against which the specified key/sequence field is to be checked and must therefore be the same length and format of the key/sequence field.

User-Supplied File Access Definition

If the types of input files previously described in this document do not provide the appropriate methods for accessing a particular file, you can code your own specialized routine and cause the Reporting Facility to call it.

The following format of the FILE command describes how to define a user-supplied File Access Module (user-module) to the Reporting Facility:

```
►►─ filename: ─┬─ INPUT ─┬─ modulename ─┬───────────────────────────┬──────────►
               └─ FILE ──┘              └─ 'alphanumeric literal' ─┘

 ►─ RECORD=nnn ─┬────────────────┬──────────────────────────────────────────────►◄
                └─ PROG=type ─┘
```

filename:

Is a unique identifier within the Reporting Facility program, coded in accordance with standard file naming conventions. A one-character alphanumeric field is generated by CA Datacom/DB Reporting Facility with the same name as the file name, to be used as a return code in conjunction with the GET command.

modulename

Is a maximum eight-character name by which the user program has been cataloged to the user Load Library. The first character must be in the range of A—Z and subsequent characters must be in the range of A—Z or 0—9. The module name must be different from the file name.

The system automatically allocates an 80-byte area to be used for communication between the user-module and the Reporting Facility. The 80-byte area is a field that resides in the GSA; you can reference it by the field name of the module name.

To code a multiple user routine in a single run with the same module name, qualify the reference to the Communications Area.

'alphanumeric literal'

Specifies an initial value for the 80-byte Communications Area defined above. If you omit this parameter, the system sets the Communications Area to blanks.

RECORD=*nnn*

Specifies the length of the I/O buffer where the user-module places the accessed data. It is the sole responsibility of the user-module to fill this area with data. This is a required parameter and must be a valid numeric constant greater than 0.

PROG=*type*

Specifies the source language of the user-module so that the necessary internal linkage can be set up properly. Valid entries are ASSEMBLY, ASM, BAL, COBOL, or COB. If you omit this parameter, the system assumes ASSEMBLY.

Note the following:

- The Reporting Facility calls the user-module once for each GET or, in the case of the primary file, once per cycle. On return from the user-module, the Reporting Facility expects the user-module to present a single logical record for processing.

  The user-module must assume complete responsibility for the integrity of the files it accesses, including their opening and closing.

- The 80-byte Communications Area is available for two-way communication between the user-module and the Reporting Facility. You can modify the contents to direct the actions of the user-module. Similarly, the user-module can post information in the Communications Area which can then be interrogated by the Reporting Facility.

- The user-module can signal four conditions through use of bytes 1—4 of the Communications Area. They are:

  END-OF-FILE 27

  By filling bytes 1—4 of the Communication Area with X'FF' (hexadecimal high-values), you can signal the END-OF-FILE condition to the Reporting Facility. Post the END-OF-FILE condition after the last record has been processed.

  NO-RECORD-FOUND 27

  By setting bytes 1—4 of the Communications Area to X'FFFFFFFE', you can signal the NO-RECORD-FOUND condition.

  RECORD-FOUND 27

  When bytes 1—4 contain blanks on return from the user-module, the Reporting Facility assumes that data has been placed in the I/O area and sets the return code to Y.

***abend condition* 27**

If the user-module detects a condition serious enough to terminate the run immediately, fill bytes 1—4 of the Communications Area with hexadecimal low-values.

Fill the remaining bytes in the Communications Area, bytes 5—80, with any user message to be printed along with the Reporting Facility abend message. Processing terminates, and a complete system dump is executed.

FORMAT Command

(Area 4B)

The FORMAT command specifies the contents of the optional output file that is to be created by the Reporting Facility in Write-Only mode. The FORMAT command is coded as follows:

```
            ┌────────────────────────────────────┐
►►─ FORMAT──▼──┬── fieldname ─────────────────┬──┴──────────────────────────►◄
               ├── 'alphanumeric literal' ────┤
               └── spacing value ─────────────┘
```

fieldname

Specifies the predefined field name of a field to be placed on the output file.

'alphanumeric literal'

Specifies an alphanumeric literal, enclosed in apostrophes, to be placed on the output file.

spacing value

Specifies the number of blanks to be inserted before the next field in the record (or at the end of the record if no further fields are specified).

Note the following:

- Use of this command is valid only in Write-Only mode. Do not code REPORT CONTROL and PRINT commands within the same Reporting Facility run. You can specify only one FORMAT command.
- When in Write-Only mode and the FORMAT command is omitted, the Reporting Facility creates an output disk file with the same format as the primary input file.
- It is possible to create fixed-length records only on the output file, regardless of the input format.
- Numeric spacing values on the FORMAT command cause blanks to be inserted between data fields in the output record. In the absence of such spacing values, output fields are contiguous.

- The Reporting Facility automatically calculates the size requirement for the fixed-length record. It cannot exceed track capacity for the device on which it resides.
- The JCL name of the output file must be DROUT.

GET Command

(Area 3)

The GET command allows you to determine the order in which files are processed. The GET command causes entry to a module that *only* accesses and transfers data from an input file. You can code a user-module or the Reporting Facility can supply a module. Format the GET command as follows:

```
►►─┬────────────┬─ GET ─ filename ─┬────────────────────┬──────────────────►◄
   └─ label: ───┘                  └─ logical expression ┘
```

label:

Specifies an optional identifying label that allows a GOTO branch to be made to this statement.

filename

Specifies the name of the file to be accessed. This is the same name that appears on the FILE command for the table to be accessed.

logical expression

Optionally enables you to specify constraints on the acceptance of records from the file. Such a logical expression not only can contain references to field names contained within the records for that specific file, but can specify non-key fields and those fields identified as key fields in the KEY= clause of the FILE command. Logical expressions cannot appear on GET commands issued for access to a VSAM file, a DL/I file, a CA Datacom/DB table, or a user-module.

Note the following:

- The first file defined in a Reporting Facility program is known as the primary file; all others are known as secondary files. The Reporting Facility handles the primary file automatically by default except in the case of CA Datacom/DB tables or IMS-DL/I files. If the system encounters a GET command specifying a primary file, it deletes the automatic file handling statements. You are responsible for handling the file access statements necessary for the primary file.

- GET commands present the next sequential record in the file, or (if a user-module is specified) call the user-module. In any case, the execution of a GET command controls four distinct settings of the return code. The system automatically sets up the return code when the file is defined using the FILE command with the same name as the file name. The four settings are:

  F

  You have not yet issued a GET command against this file.

  Y

  The requested record from the file is in the I/O area.

  N

  A NO-RECORD-FOUND condition exists. Contents of the I/O area are unpredictable.

  E

  END-OF-FILE has been reached on the specified file. Contents of the I/O area are unpredictable.

GOTO Command

(Area 3)

The GOTO (or GO TO) command causes conditional or unconditional transfer to another statement within the Reporting Facility program. The format of the GOTO command is as follows:

```
►►─┬──────────┬─ GOTO ─ destination ─┬──────────────────────┬────────────────────►◄
   └─ label: ─┘                      └─ logical expression ─┘
```

label:

Specifies an optional identifying label, possibly referenced within another GOTO command, that allows a GOTO branch to be made to this statement.

destination

The destination can be any of the following:

label

Specifies the label attached on the statement to which control is to be passed.

START

Specifies that control is to be passed to the predefined label START. Control is passed to the first procedural instruction.

TEST

Specifies that control is to be passed to the predefined label TEST. All further data manipulation statements are bypassed and control is passed directly to the record selection logic corresponding to the first SELECT statement.

EOJ

Specifies that control is to be passed to the predefined label EOJ. In Primary mode, file reading stops, the sort begins, and reports print (containing only records selected up to that point). In Secondary or Write-Only mode, files close and the Reporting Facility run terminates.

ABORT

Indicates that an abnormal condition exists. The system terminates processing with a complete formatted dump.

logical expression

Specifies the constraints for any branching. If omitted, the Reporting Facility takes an unconditional branch to the specified label. When coded, the logical expression must be satisfied before any branching takes place. The branch is to either the specified label or the next sequential instruction.

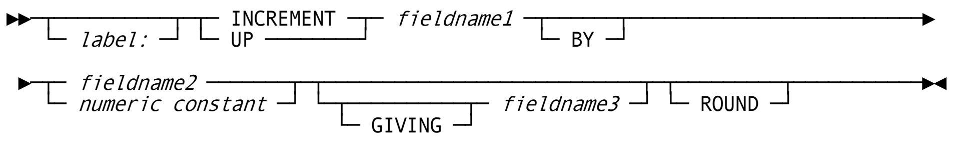
You can apply and branch labels only to these commands:

ADD	DECREMENT	GOTO	PERFORM
COMPUTE	DIVIDE	INCREMENT	PROC
CONTINUE	ENDPROC	MOVE	SET
DECODE	GET	MULTIPLY	SUBTRACT

INCREMENT Command

(Area 3)

The INCREMENT command algebraically increments a predefined numeric field by a specified numeric constant or the contents of another predefined numeric field. INCR or UP can be used in place of INCREMENT. The format of the INCREMENT command is as follows:



label:

Specifies an optional identifying label, referenced within a GOTO command, that allows a GOTO branch to be made to this statement.

fieldname1

Specifies the name given to the predefined numeric field whose value is to be used as the base for the addition of two variables. Predefine the field name as zoned decimal, packed decimal, or binary.

fieldname2

Specifies the name given to the predefined numeric field whose value is to be added to the value in *fieldname1*. Predefine the field name as zoned decimal, packed decimal, or binary.

numeric constant

Specifies an absolute numeric constant to add to *fieldname1*.

fieldname3

Optionally specifies the predefined numeric field in which the sum is placed. Predefine the field name as zoned decimal, packed decimal, or binary.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

Example

```
           Predefined field name:    FIELD1    FIELD2    FIELD3    INDEX
                  Numeric precision:      3.2       1.4       3.4      2.0
          Initial value of field:   (123.45)  (5.4321)     (1)      (10)

     INCREMENT INDEX BY 1                nc        nc         nc       11
     INCREMENT INDEX BY FIELD3           nc        nc         nc       11
     INCREMENT FIELD1 BY FIELD2
          GIVING FIELD3                  nc        nc   128.8821       nc
     INCR FIELD1 by -1.5             121.95        nc         nc       mc
     INCREMENT INDEX BY FIELD2
          GIVING FIELD3                  nc        nc    15.4321       nc
     UP INDEX BY INDEX                   nc        nc         nc       20
     UP INDEX BY 10                      nc        nc    11.0000       nc
     INCREMENT FIELD3 INDEX INDEX        nc        nc         nc       11
     INCREMENT FIELD2 BY FIELD3          nc    6.4321         nc       nc
     INCREMENT INDEX BY 2.93             nc        nc         nc       12
     INCR FIELD1 BY +.92             124.37        nc         nc       nc
     UP FIELD2 BY .333                   nc    5.7651         nc       nc
```

Note the following:

- The nc means no change in the original contents of the field.
- All significant digits of each of the operands take place in the addition, but the final result is aligned based on the DEFINE command options coded for the result field.
- The keywords BY and GIVING are optional.

LIBRARY Command

(Area 5)

The LIBRARY command allows you to enter the Library Maintenance mode. You can perform Library Maintenance commands only when you are in the Library Maintenance mode.

The format of the LIBRARY command is as follows:

```
►►─ LIBRARY ────────────────────────────────────────────────────────►◄
```

Note the following:

- The LIBRARY command must be first in the Reporting Facility command sequence, and you can use only one LIBRARY command per Reporting Facility run.
- The Reporting Facility restricts the use of the LIBRARY command to Library Maintenance mode only.
- Once the Reporting Facility Library Maintenance mode is entered, the entire run is dedicated to that purpose.

LOAD Command

(Area 5)

The LOAD command catalogs all subsequent statements in the Reporting Facility call library under the member name specified.

The format of the LOAD command is as follows:

```
►►─ LOAD ─ membername ─┬───────────┬────────────────────────────────►◄
                       └─ comment ─┘
```

membername

Specifies the unique name under which subsequent data is stored.

comment

Specifies an optional comment string that is converted into a NOTE command and is automatically inserted as the first command of the member. This NOTE command and the member name is printed on all Display Index reports.

Note the following:

- The Reporting Facility restricts the use of the LOAD command to Library Maintenance mode only.
- The LOAD command can be one of multiple Library Maintenance commands in a single Reporting Facility run.
- When the source statements that comprise the member are added to the library, they are not edited for validity by the system. Editing takes place only when you include the member in a Reporting Facility program by invoking a CALL command.
- The source statements that comprise the member to be loaded can be any of the Reporting Facility standard command sets except Library Maintenance commands.

  The system recognizes the end of the current member in one of three ways:

  – Another Library Maintenance command is found.

  – An END command is found.

  – End-of-data is reached on the input stream (SYSIPT|SYSIN).

MOVE Command

(Area 3)

The MOVE command moves a constant or the contents of a predefined field to another predefined field.

The format of the MOVE command is as follows:

```
►►─┬──────────┬─ MOVE ─ predefined field ─┬──────┬─ fieldname2 ────────────►◄
   └─ label: ─┘                           └─ TO ─┘
```

label:

Specifies an optional identifying label, possibly referenced within a GOTO command, that allows a GOTO branch to be made to this statement.

predefined field

The predefined field can be any of the following:

numeric constant

Specifies an absolute numeric constant to be used as the sending field. When a numeric constant is referenced, you must define *fieldname2* as zoned decimal, packed decimal, or binary.

'*alphanumeric literal*'

Specifies an alphanumeric constant, enclosed in apostrophes, as the sending field. When used, you must define *fieldname2* as alphanumeric.

hexadecimal

Specifies a hexadecimal literal, coded in the format literal X'nn', where nn represents an even number of valid hexadecimal digits to be used as the sending field. When used, you must define *fieldname2* as alphanumeric.

LOW-VALUE

Specifies that *fieldname2* is to be filled with binary low-values. Define *fieldname2* as alphanumeric.

HIGH-VALUE

Specifies that *fieldname2* is to be filled with binary high-values.

SPACE

Specifies that *fieldname2* is to be initialized to blanks.

ZERO

Specifies that *fieldname2* is to be initialized to zero, based on the type code associated with *fieldname2*. Predefine *fieldname2* as zoned decimal, packed decimal, or binary.

fieldname2

Specifies the name given a predefined field whose value represents the receiving field. You must predefine the field name.

The following is a sample input card file definition with some fields defined in the GSA. These fields illustrate some practical uses of the MOVE command.

Example

```
INPT: FILE CARD
      DEFINE  COMPANY-NAME               1-20    X     fields in an
      DEFINE  COMPANY-NUMBER            21-23    N     input buffer
      DEFINE  DEPARTMENT                   24    X
      DEFINE  COMMISSION                25-31    P2
      DEFINE  YTD-SALES                 32-40    P2
      DEFINE  EMPLOYEE-NUMBER           41-45    N

      DEFINE WORK-FIELD(10)=' '                        miscellaneous
      DEFINE WORK-FIELD-2 EQ                           fields in
           WORK-FIELD                     1-4    B     the GSA
      DEFINE  ARRAY(4)=' '
      OCCURS 10 TIMES
      DEFINE  COMMISSION(5.2)=0
      DEFINE  INDEX(2.0)=1
MOVE SPACES TO WORK-FIELD
MOVE ZERO TO WORK-FIELD-2
MOVE ARRAY(INDEX) TO WORK-FIELD
MOVE INPT.COMMISSION TO GSA.COMMISSION
MOVE LOW-VALUES TO ARRAY(9)
MOVE X'C1C2C3C4C5' TO WORK-FIELD
MOVE HIGH-VALUE TO DEPARTMENT
MOVE COMPANY-NUMBER TO ARRAY(INDEX)
```

Note the following:

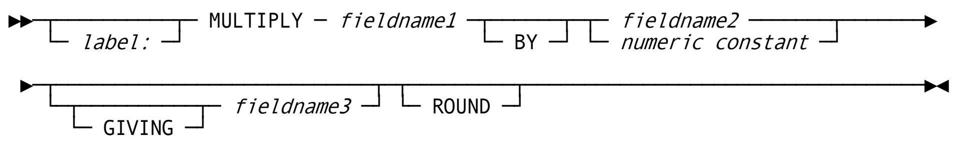
- The keyword TO is optional.
- All rules of field name definition apply to this command, that is, field names can specify scalar variables or array elements, and field name qualification is supported where appropriate.
- The sending and receiving fields must be of the same general type.

MULTIPLY Command

(Area 3)

The MULTIPLY command algebraically multiplies a predefined numeric field by a specified numeric constant or the contents of another predefined numeric field. TIMES can be used in place of BY in the following command.

The format of the MULTIPLY command is as follows:



label:

Specifies an optional identifying label, referenced within a GOTO command, that allows a GOTO branch to be made to this statement.

fieldname1

Specifies the name given a predefined numeric field whose value represents the multiplicand for the multiplication of two variables. Predefine the field name as zoned decimal, packed decimal, or binary.

fieldname2

Specifies the name given a predefined numeric field whose value represents the multiplier for the multiplication of two variables. Predefine the field name as zoned decimal, packed decimal, or binary.

numeric constant

Specifies an absolute numeric constant to be used as the multiplier in the calculation.

fieldname3

Optionally specifies a predefined numeric field in which the product will be placed. Predefine the field name as zoned decimal, packed decimal, or binary.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

Example

```
           Predefined field name:     FIELD1     FIELD2     FIELD3     INDEX
                  Numeric precision:       3.2        1.4        3.4       2.0
          Initial value of field:    (123.45)   (2.0005)    (Zero)      (3)

   MULTIPLY FIELD1 BY 2              246.90         nc         nc        nc
   MULTIPLY FIELD2 BY INDEX              nc     6.0015         nc        nc
   MULTIPLY FIELD1 BY FIELD2
        GIVING FIELD3                    nc         nc   246.9617        nc
   MULTIPLY FIELD1 BY -1.5          -185.17         nc         nc        nc
   MULTLPLY FIELD2 BY 3
        GIVING FIELD3                    nc         nc     6.0015        nc
   MULTIPLY INDEX BY INDEX               nc         nc         nc         9
   MULTIPLY FIELD3 BY INDEX              nc         nc     0.0000        nc
   MULTIPLY FIELD1 INDEX FIELD2          nc     0.3500         nc        nc
   MULTIPLY INDEX BY FIELD3              nc         nc         nc         0
   MULTIPLY INDEX TIMES 16               nc         nc         nc        48
   MULTIPLY FIELD1 TIMES +.92        113.57         nc         nc        nc
   MULTIPLY FIELD2 BY .333               nc     0.6661         nc        nc
```

Note the following:

- The nc means no change in the original contents of the field.
- All significant digits of each of the operands take place in the multiplication, but the final result is aligned based on the DEFINE command options coded for the result field.
- The keywords BY, TIMES, and GIVING are optional.
- All rules of field name definition apply to this command. That is, field names can specify scalar variables or array elements, and field name qualification is supported where appropriate.

NOTE Command

(Areas 1,2A,2B,3,4A,4B)

The NOTE command enables you to insert comments or documentation into the source program.

The format of the NOTE command is as follows:

```
►►─ NOTE ─┬───────────────┬──────────────────────────────────────────────►◄
          └─ Any comment ─┘
```

Note the following:

- NOTE commands print during the compilation only if you specify OPTION LIST ON. You can place the NOTE commands at any point within the source program. However, do not place your NOTE commands inside other commands.
- NOTE commands do not influence compilation, but serve only as comments.
- Any combination of characters is permissible within the NOTE command. Termination of the comment occurs at the end of the source record on which the NOTE command occurs. The Reporting Facility automatically resumes extraction of character strings at the beginning of the next source record.
- Multiple NOTE commands produce longer comments and graphic displays.

Example

```
NOTE*********************************************************
NOTE                                                        *
NOTE                                                        *
NOTE           THIS IS AN EXAMPLE OF HOW TO                 *
NOTE           FORM A COMMENT BLOCK WITHIN                  *
NOTE           A REPORTING FACILITY PROGRAM.                *
NOTE                                                        *
NOTE                                                        *
NOTE*********************************************************
```

OPTION Command

(Area 1)

The OPTION command serves two functions:

- It overrides, for a single processing run, the default options specified at system installation time.
- It provides parameters that influence the method of operation.

The format of the OPTION command is as follows:

Note: AUTO and *phasename* are parameters of DBCOMM.

```
►►─ OPTION ──┬─────────┬┬───────────┬┬────────────────────┬──────────────►
             └ ABORT ──┘└ COMPILE ──┘└ DASH ─┬─ ON ──┬────┘
                                             └─ OFF ─┘

►─┬────────────────────────────┬┬─────────────────────┬──────────────────►
  └ DBCOMM= ─┬─ AUTO ──────────┘└ DBPRI=priority ─────┘
             └─ phasename ─┘

►─┬────────────────────────────┬┬────────────────┬───────────────────────►
  └─┬─ DBSEQBUF= ──┬┬─ 2 ──┬───┘└ DDDBID=dbid ───┘
    └─ DBSEQBUFS= ─┘└─ nn ─┘

►─┬──────────────────────────┬┬───────────┬┬───────────────────────┬─────►
  └─┬─ DISK= ──┬── device ───┘└ EXCLUDE ──┘└ INDEXERR= ─┬─ 1 ──────┘
    └─ DISKS= ─┘                                        ├─ FIRST ─┤
                                                        └─ ABORT ─┘

►─┬─────────────────┬┬────────────────┬┬───────┬┬─────────────┬──────────►
  └ LANG=language ──┘└ LIST ─┬─ ON ───┘└ MAP ──┘└ NEWREPORT ──┘
                             └─ OFF ─┘

►─┬────────────────────────────────────┬┬───────────────────────────┬────►
  └ OMIT ──┬─ DETAIL ─┬─ BLANK LINES ──┘└ OUTLIMnnn ─┬─ LINES ──────┘
           ├─ TOTAL ──┤                              ├─ RECORDS ─┤
           └─ ALL ────┘                              └─ PAGES ───┘

►─┬────────────┬┬───────────────┬┬─────────────────────┬┬─────────┬──────►
  └ PAGE=nnn ──┘└ PRINTER=nnn ──┘└ PRTEXIT=phasename ──┘└ ROUND ──┘

►─┬─────────────────────┬────────────────────────────────────────────────►◄
  └ RUNDIAG= ─┬─ YES ───┘
              └─ NO ──┘

►►─┬────────────────┬┬──────────────────┬────────────────────────────────►
   └ SIGN= ─┬─ F ───┘└ SORT= ─┬─ NONE ──┘
            └─ C ─┘           └─ nnn ─┘

►─┬───────────────────────────────┬┬────────┬────────────────────────────►◄
  └ WRITE ONLY ─┬─────────────────┘└ XREF ──┘
                └ BLOCKED nnn ─┘
```

ABORT

Specifies that when the OPTION command is encountered, the Reporting Facility is to abort and print a system dump. This parameter is very rarely required, but you could use it for trapping and analyzing internal abend conditions.

COMPILE

Specifies that only source statements are to be compiled. The system creates no reports or output files. The Reporting Facility terminates the run immediately following the compilation. Use this feature to ensure that a clean compile is achieved before execution takes place. When this parameter is not present, the Reporting Facility defaults to COMPILE and EXECUTE.

DASH ON|OFF

Specifies whether the Reporting Facility compiler is to accept dashes as valid characters in data names. This parameter is necessary to allow for downward compatibility with earlier versions of the Reporting Facility where dashes were not allowed as valid characters in data names. If this parameter is omitted, DASH ON is the default.

- DASH ON indicates that dashes are valid characters in data names.
- DASH OFF indicates that dashes are not allowed in data names.

DBCOMM=AUTO|*phasename*

Is used in conjunction with CA Datacom/DB table access in a program. It specifies the manner in which the required CA Datacom/DB User Requirements Table (URT) is referenced. You may specify the following parameters:

AUTO

Specifies that the Reporting Facility is to generate the URT entries automatically. The system makes a single entry for each CA Datacom/DB table defined in the run and generates an entry. This option can be disallowed by site management at product modification time.

phasename

Specifies a maximum eight-character name assigned to a prelink-edited UR that contains an entry for each CA Datacom/DB table to be accessed in the run. The first character of the phase name must be in the range of A—Z and subsequent characters must be in the range of A—Z or 0—9. This option must be used if DBCOMM=NOAUTO was chosen by site management at product modification time through the IS08Z09 macro.

DBPRI=*priority*

Specifies the priority of the automatically generated URT. Valid options are 1—15. The default is 3.

DBSEQBUF[S]=*n*

Specifies the number of sequential buffers. You can specify DBSEQBUF or DBSEQBUFS. This option is valid only with GETIT processing. The default is 2. The valid entries are 2 through 128 or -2 through -128.

DDDBID=*dbid*

Specifies the physical database ID in which CA Datacom Datadictionary resides. The system maintains the default value in the System Resource Table associated with the active dictionary. You need this parameter only if an alternate CA Datacom Datadictionary needs to be referenced. Use this parameter only for processing COPYDD commands in the Reporting Facility program. .cp10

DISK=*device*

Specifies the physical disk drive device type to be used by the Reporting Facility in any of three instances:

- As the Hit File (SORTIN|SORTOUT) you execute in Primary mode
- As the default device type when you define an input disk file using a FILE command
- As the device type on which the Reporting Facility call library resides if you process CALL commands or invoke Library Maintenance mode

The only way to modify the disk device type on which the DRWORK file resides is to modify the system default value.

The system establishes the default disk device type during the installation of the Reporting Facility. This parameter overrides the default value for the current execution of the Reporting Facility.

Valid entries are 3350, 3375, 3380, 3390, 9345, and FBA (z/VSE).

EXCLUDE

Specifies that all page headings on the output reports are to be suppressed. If EXCLUDE is specified, it must precede the PAGE= parameter on the OPTION command. This parameter does not affect the printing of the Reporting Facility compile listing.

INDEXERR=FIRST|1|ABORT

Specifies the default processing to be performed when an index violation occurs during processing of an array reference. An index violation occurs when:

- The index specified is invalid, such as non-numeric or zero.
- The index value is greater than the number of entries defined in the array.

One of three parameters can be specified:

FIRST

Specifies that the Reporting Facility is to default to using the first element in the array when an index violation occurs, then to continue processing.

1

Causes the same affect as FIRST.

ABORT

Causes the Reporting Facility to abend immediately and print a system dump when an index violation occurs.

If no parameter is specified, the Reporting Facility assumes it is FIRST. The Reporting Facility automatically prints a line showing the number of index violations that occurred during execution of the Reporting Facility program.

LANG=*language*

Specifies the language translation to be performed when printing the literals 'PAGE', 'GRAND TOTAL', 'END OF REPORT', and the three-character abbreviation of the alphabetic month. The 'PAGE' and month abbreviation translations print on the Reporting Facility compiler listing and on the output reports.

'GRAND TOTAL' and 'END OF REPORT' translations appear only on the output reports. The system establishes the default language during installation of the Reporting Facility, and specification of this parameter overrides the default during a single execution of the Reporting Facility. The following is a summary of valid parameters and translations:

LANG=.	PAGE	GRAND TOTAL	END OF REPORT
ARABIC	PAGE	GRAND TOTAL	END OF REPORT
DANISH	SIDE	GRAND TOTAL	RAPPORT SLUT
DUTCH	PAGE	GRAND TOTAL	END OF REPORT
ENGLISH	PAGE	GRAND TOTAL	END OF REPORT
FINNISH	SIVU	LOBBUSUMMA	END OF REPORT
FRENCH	PAGE	TOT.GENERAL	RECAPITULATION
GERMAN	SEITE	GESAMTTOTAL	END OF REPORT
GERMAN2	BLATT	GESAMTTOTAL	REPORT ENDE
ITALIAN	PAGE	TOT.GEN	END OF REPORT
JAPANESE	PAGE	GRAND TOTAL	END OF REPORT
KOREAN	PAGE	GRAND TOTAL	END OF REPORT
NORWEGIAN	SIDE	TOTAL SUM	RAPPORT SLUTT
PORTUGESE	PAG.	TOTAL GERAL	FIM DO RELATORIO
SPANISH	PAG.	TOT. GRANDE	REPORT FINIS
SPARE	PAGE	GRAND TOTAL	END OF REPORT
SWEDISH	SIDA	GRAND TOTAL	RAPPORT SLUT

LANG=	3-CHARACTER ALPHABETIC MONTH ABBREVIATION											
ARABIC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
DANISH	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
DUTCH	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
ENGLISH	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
FINNISH	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
FRENCH	JAN	FEV	MAR	AVR	MAI	JUN	JUL	AUO	SEP	OCT	NOV	DEC
GERMAN	JAN	FEB	MRZ	APR	MAI	JUN	JUL	AUG	SEP	OKT	NOV	DEZ
GERMAN2	JAN	FEB	MRZ	APR	MAI	JUN	JUL	AUG	SEP	OKT	NOV	DEZ
ITALIAN	GEN	FEB	MAR	APR	MAG	GUI	LUG	AGO	SET	OTT	NOV	DIC
JAPANESE	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
KOREAN	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
NORWEGIAN	JAN	FEB	MAR	APR	MAI	JUN	JUL	AUG	SEP	OKT	NOV	DES
PORTUGESE	JAN	FEV	MAR	ABR	MAI	JUN	JUL	AGO	SET	OUT	NOV	DEZ
SPANISH	ENE	FEB	MAR	ABR	MAY	JUN	JUL	AGO	SEP	OCT	NOV	DIC
SPARE	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
SWEDISH	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC

LIST ON|OFF

Controls the printing of the Reporting Facility statements during compilation. You can suspend printing by specifying LIST OFF and resume printing by specifying LIST ON. LIST ON is the default. If LIST OFF is specified as the first card in the Reporting Facility, the banner page will not be printed. Otherwise, the banner page will always be printed.

MAP

Specifies that the Reporting Facility is to print a detailed analysis of the compilation process immediately following the source listing. This MAP contains internal control block analysis, a field name cross-reference listing, input and output file analysis, and detailed descriptions of each specified report.

When certain error conditions are detected during the compile, the Reporting Facility generates a limited form of the Map Listing regardless of whether OPTION MAP is specified. Specify OPTION MAP in each Reporting Facility program. When OPTION MAP is not specified, no detailed analysis is generated.

NEWREPORT

Inserts one blank page between each report. This is useful when printing on both sides of the paper.

OMIT DETAIL|TOTAL|ALL BLANK LINES

Specifies that, during report generation, the Reporting Facility is to suppress the printing of blank lines at the detail level, the total level, or both. This option does not affect page and column headings.

OUTLIM *nnn* RECORDS|LINES|PAGES

Specifies the maximum number (*nnn*) of records, lines, or pages to be processed in a single run of the Reporting Facility. You can code multiple OUTLIM parameters in a single run, each specifying one of the three options. The line and page options are effective only during the report printing stage. The record option limits either the total number of primary input records or the number of automatically generated Hit File records.

When the maximum number of records to be processed is reached, the system generates an effective GOTO EOJ. Processing then continues as normal. When the maximum number of lines or pages is reached, the Reporting Facility generates a message and terminates the run.

PAGE=*n*

Defines the maximum number of lines to be printed on a page before a page eject is forced. You can specify it as any valid numeric integer in the range of 8—88. Specifying a large page depth generates reports with initial report headings, but with no further page skipping (except where forced by the effect of a NEWPAGE or SKIP parameter on a CONTROL command).

Multiple OPTION PAGE&equals.*n* commands can be submitted in a single Reporting Facility run, but only the last one specified influences the page depth on printed reports. The system established the default page depth during the installation of the Reporting Facility. This parameter overrides it only for the current run.

PRINTER=*n*

Specifies the maximum number of characters to be printed on a single print line for the current run. You can specify it as any valid numeric integer in the range of 20—144. You can submit multiple OPTION PRINTER=nnn commands in a single Reporting Facility run to allow for different printing requirements for each report. (The printer width controls the centering of detail fields on the output report.)

The user ID is centered according to the printer width in effect when the REPORT command is processed. Positioning of the page number is also calculated based on the last printer width specified at the time of the REPORT command. The system established the default printer width during the Reporting Facility installation. The parameter overrides it only for the current run.

PRTEXIT=*phasename*

Specifies the name of a prelink-edited subroutine (developed by you) to which control is passed by the Reporting Facility in lieu of printing any lines. This subroutine can then examine or modify each line as it is being created by the Reporting Facility.

The *phasename* must begin with a character in the range of A—Z and subsequent characters must be in the range of A—Z or 0—9. The maximum length of the parameter is eight characters.

ROUND

Specifies that rounding will be done on all COMPUTEs and SETs in the program before the final total is placed in the result field. If not specified here, rounding can be specified on individual COMPUTE and SET statements. The default is not to round.

RUNDIAG=YES|NO

Specifies whether run diagnostics will be produced. The default is YES.

SIGN=F|C

Specifies the default positive sign value to be forced on packed decimal numbers as a result of a command. This is necessary since it is possible for fields in packed decimal format to be referenced in logical expressions, so a "character" compare can be generated. SIGN = F is the default.

SORT=NONE|*n*

Specifies the sorting requirements for a single run. SORT=NONE specifies that no sort will be performed during this run, and therefore limits the scope of the run to either creating an output file (Write-Only mode) or producing a single report presented in the same sequence as the input data (Secondary mode).

If sorting is to be performed, supply *n* to specify the number of intermediate work areas available for the sort. In a z/OS environment, *n* is in the range of 3—6. In a z/VSE environment, *n* is in the range of 1—8. If this parameter is omitted and a sort is required, the system assigns a default value of 3 (for z/OS) or 1 (for z/VSE).

WRITE ONLY (BLOCKED *n*)

Specifies that the Reporting Facility is to enter the Write-Only mode. The system creates a sequential, fixed-length disk file in lieu of report generation. The FORMAT command determines the format of the output file to be created. In the absence of the FORMAT command, the output file contains records of the same format and blocking factor as the primary input file.

If BLOCKED *n* is specified, it denotes the blocking factor to be used for the output file, that is, the number of records in one block of output. The Reporting Facility calculates the block size, which must be within the constraints imposed by the disk storage device track capacity for the current disk device type.

XREF

Provides a cross-reference of field names in the report. Each is listed in alphabetical order with the number of the line on which it was used.

Note the following:

- The system processes all parameters on the OPTION command from left to right.
- You can enter multiple OPTION commands in a Reporting Facility run, but you should specify the run options at the beginning of the program.
- If the default values specified during installation of the system are acceptable and no additional requirements are desired, omit the OPTION command.

PERFORM Command

The PERFORM command causes a branch to a subroutine within the Reporting Facility program. The optional logic and expression specifies the condition under which the branch is to occur. At the end of the subroutine, control will pass to the next command following the PERFORM.

The PROC and ENDPROC commands define the limits of the subroutine to be executed by the PERFORM command.

The format of the PERFORM command is as follows:

```
►►─┬───────────┬─ PERFORM ─ label2 ─┬──────────────────────┬──────────────►◄
   └─ label1: ─┘                    └─ logical expression ─┘
```

label1:

Specifies an optional identifying label, possibly referenced within a GOTO command, that allows branching to be made to this statement.

label2

Specifies the label attached to the statement to which control is to be passed. This label must be on a PROC command.

logical expression

Specifies the constraints under which any branching is to take place. If omitted, Reporting Facility performs the routine specified by the label unconditionally. When coded, the logical expression must be satisfied before any branching takes place. If the expression is not satisfied, control continues with the next command. Logical expressions cannot be ANDed together with PERFORM.

You can apply only branch labels to the following commands:

ADD	DECODE	GET	MULTIPLY
PROC,	DECREMENT	GOTO	PERFORM
COMPUTE,	DIVIDE	INCREMENT	SET
CONTINUE	ENDPROC	MOVE	SUBTRACT

Example

```
GET DATA                                   define files and variables
LABEL1: CONT
SET A = B                                  other logic
PERFORM SUB
NOTE an implied GOTO TEST is generated here by RPT Facility
SUB: PROC
COMPUTE C = A * B
SET B = C
ENDPROC
report section
```

PRINT Command

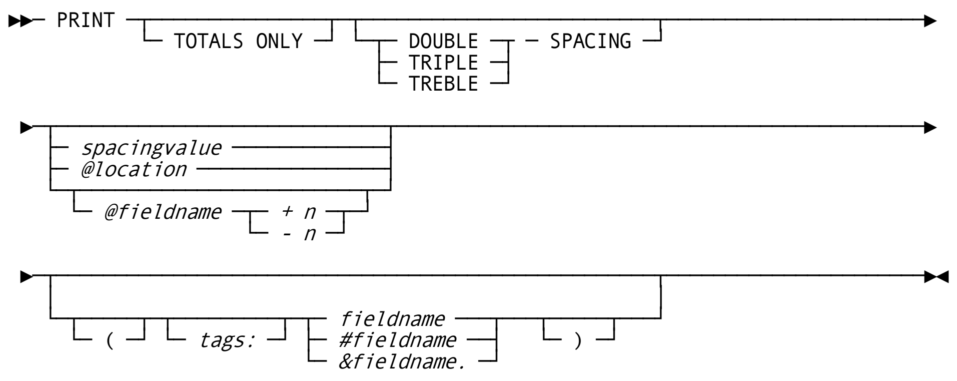
(Area 4A)

The PRINT command specifies the fields that are to be printed on each detail line within a specific report. The command also provides several options that control the format of the output report.

The PRINT command does the following:

- Defines those fields that are to be accumulated when a control break occurs. In this instance, the CONTROL command and the PRINT command work together to define the format of a total line.
- Specifies whether the report is to be generated containing both detail and total lines, or simply a summary report containing only total lines.
- Specifies conditional printing of selected fields based on conditions previously satisfied by the SELECT command.
- Controls the vertical spacing of each report.

The format of the PRINT command is as follows:



TOTALS ONLY

Specifies print of a summary report. You can generate a report where detail lines are suppressed and only totals appear at each control break level. This parameter, when specified, must immediately follow the keyword PRINT on the first PRINT command of a group and can appear only once within a report group. The only fields that print on a total line are accumulated fields (fields enclosed in parentheses), control break fields, and fields prefixed with an ampersand (&).

DOUBLE|TRIPLE|TREBLE SPACING

Specifies the vertical line spacing for the report. Normal spacing between lines (or groups of lines with multiple PRINT commands) is single spacing. This parameter, if specified, must be on the first PRINT command of a particular group and must precede any field name specifications. It provides optional spacing only for the first print line of a group of lines if multiple PRINT commands are used.

spacingvalue

Is an unsigned numeric constant specifying the number of blanks to be inserted before the next field on the line (or at the end of the line if no further fields are specified).

@location

Is an unsigned numeric constant immediately preceded by the at-sign (@), with no intervening blanks, specifying the location to print the subsequent field name. Numeric fields align on the rightmost character (inclusive of any sign) and alphanumeric fields align on the leftmost character.

The size of a field is determined by the longest of the field width, its accumulator (if accumulated), or the edit pattern (or length of the heading if the field is specified on the first PRINT command).

If an accumulated value is to be printed, the first field must begin at least 13 bytes into the print line to allow for the legend "Grand Total".

@fieldname* with *+ n* or *- n

Is to be aligned with the identified field specified in a previous print line. If you specified the field (*@fieldname*) more than once on a previous PRINT command, the system uses the first occurrence.

The optional parameter n specifies the number of positions to align to the left (- *n*) or to the right (+ *n*) of the field name specified. When the minus or plus sign is used, there must be a space both before and after the sign.

Numeric fields align on the rightmost character (inclusive of the sign) and alphanumeric fields align on the leftmost character.

( )

Parentheses surrounding the field and its associated parameters indicate that the field is to be accumulated and printed as part of the total line when a control break occurs.

tags:

Specifies that the subsequent field is to be printed only if certain selection criteria are met. Valid entries are A—Z. The SELECT command has the ability to assign specific tag characters prior to the record being selected for printing. The PRINT command has the ability to conditionally print specific fields based on those tag characters.

fieldname

Specifies the name from the Hit File record assigned a predefined scalar variable or array element to be printed. The size of the print window necessary to print the field is based on the largest of the following four variables:

- Length of the field, in bytes
- Length of the accumulator, if accumulated
- Length of the edit pattern, if specified
- Length of the headings, if specified on the first PRINT command of a group

#fieldname

Specifies the same basic processing as with the field name described above except instead of the value of the field, a running accumulation of the field is printed on each detail line.

If the *#fieldname* is enclosed in parentheses, the running total prints on each detail line only until a control break occurs, at which time the accumulator prints and the system resets the running total to zero before accumulation continues.

&fieldname

Specifies the same basic processing as with the field name described above except that the value to be printed is taken directly from the General Storage Area (GSA) rather than from the Hit File.

Note the following:

- You can include any number of PRINT commands in each report definition group. Each PRINT command generates a single output line.
- Each data field specified on the first PRINT command prints with its associated column heading. Headings are either the field name or the alternate headings supplied for the field. Second and subsequent PRINT commands generate only a single line of print, without headings.
- The Reporting Facility automatically positions and centers each field within the specified (or default) printer width. However, this automatic function can be overridden by specification of:
  - Interfield spacing values
  - Specific location printing
  - Relative location printing

Example

```
PRINT NameA 1 FldA 3 FldB
```

Note the following:

- As shown in the syntax of the PRINT command above, once the automatic spacing feature is overridden for a specific PRINT command, it is not restarted until the next PRINT command is encountered.
- The number of PRINT commands in the report group determines how many lines will be printed at total time.

PROC Command

(Area 3)

The PROC command causes the Reporting Facility to initialize the beginning of a subroutine. The PROC command must be paired with an ENDPROC command. Any valid Reporting Facility (Area 3) commands can be placed between a PROC command and an ENDPROC command.

```
►►─ label: ─ PROC ──────────────────────────────────────────────────────►◄
```

label:

Specifies a required identifying label that will be referenced in a PERFORM command.

The PROC and ENDPROC commands must be the last group of statements immediately before the REPORT section. A GOTO TEST command is automatically generated before the first PROC.

REPORT Command

(Area 4A)

The REPORT command identifies each report to be generated within the Reporting Facility run and assigns optional user heading lines to appear at the top of each page of the report.

The format of the REPORT command is as follows:

```
►►─ REPORT ─┬──────────────┬─┬──────────────────────┬──────────────────────────►◄
            └─ 'literal1' ─┘ └───┬─ fieldname ───┬──┘
                                 ├─ 'literaln' ──┤
                                 └─ n1 ──────────┘
```

'literal1'

Is a character string, enclosed in apostrophes, that is to appear on each page of the report being defined. This literal centers automatically and prints on the line immediately following the USER lines. The CURRENT-DATE and PAGE NUMBER fields print on the left and right of this literal, respectively.

fieldname

Specifies a predefined field to be printed on the optional third heading line for the report.

'literaln'

Is a character string, enclosed in apostrophes, that is to appear on the optional third line of the heading for the report.

n1

Specifies the number of spaces to appear between a field name or a literal on the optional third heading line for the report. This parameter must be a positive integer and can be interspersed between any field name or literal. If not specified, the system assumes single spacing between a field name or literal.

Note the following:

- The REPORT command is the first command in a report definition group. You can define up to 63 unique report groups in a single processing run.
- The CURRENT-DATE field to be printed on the second heading line has the format MM/DD/YY, DD/MM/YY, or DD MMM YY, depending on the option selected during the installation of the Reporting Facility.
- The CURRENT-DATE and PAGE NUMBER fields print in specific locations within the second heading line and you cannot alter their contents.
- The total length of either the second or third heading lines cannot exceed the printer width currently in force.
- When the contents of fields are to print on the optional third heading line, you can also use the SKIP or NEWPAGE parameters on the CONTROL command to ensure proper printing.

The basic format of a report and a description of each line printed follows:

```
Line   ┌──────────────────────────────────────────────────────────────┐

  1                        User Heading
  2     DATE              Report Heading               PAGE nnn
  3     Third Heading Line
  4    ├──────────────────────────────────────────────────────────────┤

  5     Column
  6             Headings
  7    ├──────────────────────────────────────────────────────────────┤

  8     Detail Line 1
  9     Detail Line 2
  .
  .
  .
  n     Detail Line n
       └──────────────────────────────────────────────────────────────┘
```

Line 1

Contains the concatenated literal specified on the command, centered within the current printer width.

Line 2

Contains the CURRENT-DATE field, the first literal from the REPORT command, and the PAGE NUMBER field. The literal centers within the current printer width.

Line 3

Contains the optional third heading line from the REPORT command. The fields, literals, or spacing values do not center automatically.

Lines 4 and 7

Are generated automatically by the system and serve to separate the report headings from the detail column headings.

Lines 5 and 6

Represent the column headings of the fields to be in the report as specified on the PRINT command.

Lines 8—n

Represent the detail lines to be printed as specified on the PRINT command. Detail lines print until the maximum page depth is reached, then new headings are forced.

SELECT Command

(Area 4A,4B)

The SELECT command specifies the criteria used to determine which input records are to appear in the output report (if operating in Primary or Secondary mode) or in the output file (if operating in the Write-Only mode). The format of the SELECT command is as follows:

SELECT Command

```
►►─ SELECT ─┬─────────┬─┬─ logical expression ─┬──────────────────────────────►◄
            └─ 'tag' ─┘ └─ ALL ────────────────┘
```

'*tag*'

Is a single character within the range of A—Z, used to identify which SELECT command caused the record or set of records to be selected. The system can then reference this tag character in any subsequent PRINT commands within the same report group to permit selection printing or accumulation.

ALL

Specifies that all records are to be selected for processing, regardless of their content.

logical expression

Specifies the logical constraints to apply before the system selects this record set for processing. You must code this parameter following the standard Reporting Facility conventions for logical expressions.

Note the following:

- When you code multiple SELECT commands, three things are important:
    1. The system evaluates each SELECT command independently, but each command is considered to be a logical OR relationship with the other SELECT commands.
    2. The first SELECT command to be satisfied causes the record set to be selected without the remaining SELECT commands in the group being evaluated.
    3. You must code SELECT commands in a particular report group or in the output file definition group, with no other Reporting Facility commands between them.
- If the tag character is used, it is moved to the predefined field named TAG. The system then references it in subsequent CONTROL, PRINT, or FORMAT commands. You can sort on, print, or place in the output file the tag character assigned to the record set.
- To selectively print data on a particular report based on the tag character, you must combine the use of the SELECT and PRINT commands. Since the SELECT command causes the tag character to be assigned, the PRINT command analyzes the tag character prior to printing or accumulating.

Example

This command sequence causes all records to be selected:

```
SELECT 'A' WHEN AMOUNT-FIELD LT 100
SELECT 'B' WHEN AMOUNT-FIELD LTE 200
SELECT 'C' ALL
PRINT   A;AMOUNT-FIELD  B;AMOUNT-FIELD  C;AMOUNT-FIELD
```

The system tags any record with an AMOUNT field containing a value of less than 100 with an A. When the AMOUNT field is in the range of 100—200, the record is tagged with a B; otherwise, it is tagged with a C.

SET Command

The SET command transfers data from a predefined sending field, numeric constant, or alphanumeric literal, to a receiving field. To provide maximum flexibility, the Reporting Facility supports two unique formats of the SET command:

- SET during record selection
- SET during report printing

The function of each type of command is basically the same, but the format and placement of the different commands within the program vary slightly. Not only does the format of the command affect the processing to take place, but the placement of the command within the program determines the type of processing to be performed.

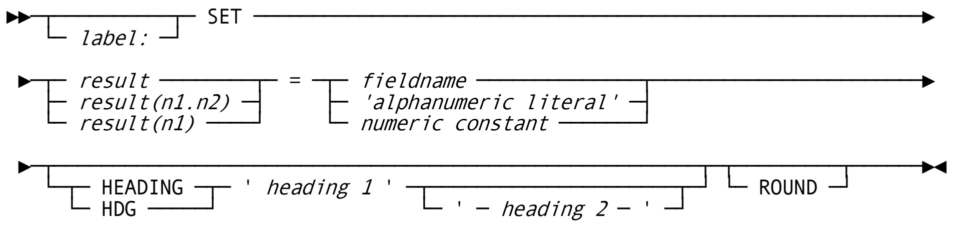
SET During Record Selection

(Area 3)

During record selection, this format of the SET command specifies data movement to be performed from one scalar variable or array element to another. You do not need to predefine the result field, but can implicitly define it in the SET command. You should, however, predefine all fields referenced in procedural commands so as to minimize the margin of error.

Data movement during record selection implies that the operation is to be performed prior to the sort, if any, and prior to printing of the reports. Fields from both the input buffer area and the General Storage Area (GSA) can be used as component parts of the SET command. When the result field is then referenced by field name in a subsequent report definition group, the system directs the value to the Hit File for processing.

The SET during record selection command is coded as follows. You can use EQ, EQUAL, TO, or EQUAL TO in place of = (equal sign) in this command.



label:

Specifies an optional identifying label, possibly referenced within a GOTO command, that allows a GOTO branch to be made to this statement.

result

Specifies the name given a predefined field or array element to which the data is moved.

result(n1.n2)

Specifies a user-supplied field name to which data is moved. Use of this parameter implicitly defines the field to contain the result.

The *n1* and *n2* indicate the number of integers and decimals, respectively. The value of *n1+n2* must be greater than 0 and must not exceed 15. When the field contains no integers, *n1* must be 0. When no decimals are necessary, *n2* must be 0.

result(n1)

Specifies a user-supplied field name to which data is moved. Use of this parameter implicitly defines the field to contain the result. The field is generated as an alphanumeric field and resides in the GSA. *n1*, which indicates the length of the field in bytes, must not exceed 256.

When you omit *n1* and do not predefine the result field, the system generates a field with the same length as the sending field, in alphanumeric format.

fieldname

Specifies the name of a predefined scalar variable or array element, defined as residing in either the input record area or the GSA, and to be moved to the result field.

numeric constant

Specifies an absolute numeric constant to be moved to the result field. Define the receiving field of the SET command as numeric, packed decimal, zoned decimal, or binary.

'*alphanumeric literal*'

Specifies an alphanumeric constant, enclosed in apostrophes, as the sending field. When used, the result field is to be defined as alphanumeric. If the receiving field is larger than the sending field, the receiving field is padded with blanks. If the receiving field is smaller, the literal in the final result is left-justified and truncated.

HEADING

Is an optional separator used to denote that subsequent literals are to be interpreted as headings. Heading specifications are valid only when the result field of the computation is implicitly defined. HDG can be used in place of HEADING.

'heading 1' 'heading 2'

Allows the user to assign one or two heading lines to the result field when the field is implicitly defined in the SET command. The heading lines appear as column headings if the field is printed in a report.

If you omit the parameter and implicitly define the result field, the Reporting Facility assigns the field name as heading line 1, with no second heading line.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

The following command specifications illustrate some practical uses of the SET command:

```
SET RESULT-FIELD EQUAL TO FIELDB
SET RESULT-FIELD(7.2) = 0 HDG 'AMOUNT' 'FIELD'
SET FIELDA EQUAL TO ZERO
SET INDEX EQ 1
SET FIELDA TO -123.45
SET FIELDB TO 'ABCDEFG'
```

Note: You cannot specify this format of the SET command in a report definition group.

SET During Report Printing

(Area 4A)

This format of the SET command functions the same way during report printing as during record selection. The intention is to move data from one location to another. The differences are the timing of the execution of the movement and the types of data that can be processed.

Since the movement is performed during the printing of the reports, this format of the command can appear only within the report definition group. In addition, the data movement pertains only to the particular report in which it is coded.

You can specify data movement to be performed at detail time, total time, or both. When data is moved at detail time, input to the operation can be from the GSA and from the Hit File. When data is moved at total time, input can also be from field accumulators or control break values.

The SET during report printing command is coded as follows:

```
►►─┬─────────┬─ SET ─┬─ (D) ──┬──────────────────────────────────────────►
   └─ label: ┘       ├─ (DT) ─┤
                     └─ (T) ──┘

►─┬─ result ─────────┬─ = ─┬─ fieldname ──────────────────┬─────────────►
  ├─ result(n1.n2) ──┤     ├─ &'alphanumeric literal' ────┤
  └─ result(n1) ─────┘     └─ &numeric constant ──────────┘

►─┬──────────────────────────────────────────────────────┬─┬─────────┬──►◄
  └─┬─ HEADING ─┬─ ' heading 1 ' ─┬────────────────────┬─┘ └─ ROUND ─┘
    └─ HDG ─────┘                 └─ 'heading 2' ──────┘
```

D

Specifies that the system is to perform data movement at detail time during printing of the report.

DT

Specifies that the system is to perform data movement at both detail time and total time.

T

Specifies that the system is to perform data movement at total time, (when a control break occurs).

result

Specifies the name of the field to which the data is moved. The field must reside in the GSA and be a packed decimal or an alphanumeric field.

result(n1.n2)

Specifies a user-supplied field name to which data is moved. Use of this parameter implicitly defines the field to contain the final result. The field generates in packed decimal format and resides in the GSA.

When the precision clause *(n1.n2)* does not follow the field name specifications, the Reporting Facility automatically calculates the precision of the result based on the attributes of the sending field. *n1* and *n2* indicate that the field is to contain *n1* integers and *n2* decimals, respectively.

The value of *n1+n2* must be greater than 0 but not exceed 15. This field truncates if you are accumulating and the value exceeds 15. If the field contains no integers, *n1* must not be 0. If no decimals are necessary, *n2* must be 0.

result(n1)

Specifies a user-supplied field name to which data is moved. Using this parameter implicitly defines the field to contain the result.

The field is generated as an alphanumeric field and resides in the GSA. *n1* indicates the length of the field, in bytes, and must not exceed 256. When you omit *n1* and the field name denoted by *result* is not predefined, a field is generated with the same length as the sending field.

fieldname

Specifies the name of a predefined scalar variable or array element, residing in either the input record area or the GSA, to be moved to the result field. In other words, this is the sending field. When the field name is prefixed with an ampersand (&), the value is to be taken during report printing from the final presort of the GSA or the result field of a previously executed SET or COMPUTE command.

&*numeric constant*

Specifies an absolute numeric constant to be moved to the result field. It can be any valid numeric literal. The receiving field of the SET operation must be defined as numeric, packed decimal, zoned decimal, or binary.

&'*alphanumeric literal*'

Specifies an alphanumeric constant, enclosed in apostrophes, as the sending field. When used, the result field must be defined as alphanumeric.

HEADING

Is an optional separator used to denote that subsequent literals are to be interpreted as headings. Heading specifications are valid only when the result field of the SET operation is being implicitly defined. HDG can be used in place of HEADING.

'*heading 1*' '*heading 2*'

Allows you to assign one or two heading lines to the result field when the field is implicitly defined in the SET command. The heading lines appear as column headings if the field is printed in a report. If you omit the parameter and implicitly define the result field, the Reporting Facility assigns the field name as heading line 1, with no second heading line.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

Note the following:

- Fields referenced in the GSA contain their final presort value, since the system performs the data movement just prior to report printing. The processing sequence within the Reporting Facility is:

  1. Read and process all input files
  2. Sort (A CONTROL command is required in order for the GSA value, &field-name., to be accessed. This is true even if the data is already generated in correct output sequence.)
  3. Print reports

  The GSA stabilizes at the end of the record selection phase; you can only reference or modify it with this type of command.

- You can later reference the result field of the SET (D) or (T) in another special COMPUTE or SET command or in a PRINT command. In every case, the current GSA field must be prefixed by an ampersand (&).

- A field name specified within the special SET command causes a unique reference to either a field in the Hit File or the GSA. The following are some special coding considerations:

  – Sending fields enclosed in parentheses indicate that the accumulated value is to be used in the operation. If accumulation is not specified, the Reporting Facility assigns zero to the field when it is referenced at total time.

  – You must prefix numeric constants and alphanumeric literals used in this format of the SET command with an ampersand (&).

  – If you set an alphanumeric field at total time, you must include the alphanumeric field in parentheses. This will make the last occurrence of the alpha field available in the result field at total time. This is a good way to get alphanumeric fields on total lines without putting them in the CONTROL statement.

Example

Examples of using the SET command during report printing:

```
SET(D) RESULT-FIELD EQ &100.5 ROUND

SET(T) RESULT-FIELD EQUAL TO (INPUT-FIELD)

SET(DT) ALPHA-FIELD(7) TO &'ABCDEFG' HDG 'ALPHA' 'FIELD' ROUND

SET(T) ALPHA-FIELD-A(7) = (ALPHA-FIELD-B)
```

SKIPn Command

(Areas 1,2A,2B,3,4A,4B)

The SKIP*n* command is a compiler control command and is processed only if OPTION LIST ON is specified, that is, the Reporting Facility compile listing is to be printed. Each invocation of this command forces the printing of the specified number of blank lines on the compile listing.

The format of the SKIP command is as follows:

```
►►─ SKIPn ──────────────────────────────────────────────────────────►◄
```

n

Is a numeric value in the range of 1-3, representing the number of blank lines to print before printing the next command line. *n* must follow the keyword SKIP, with no intervening blanks.

Example

```
SKIP1      space 1 line
SKIP2      space 2 lines
SKIP3      space 3 lines
```

Note the following:

- The SKIP*n* command, when coded, must be the only command specified on the entire line.
- If OPTION LIST OFF is specified, the system still edits the command during the compile, but the command is ignored.
- Use this command in conjunction with the EJECT and NOTE commands to properly document each program.

SUBTRACT Command

(Area 3)

The SUBTRACT command algebraically subtracts a specified numeric constant or the contents of a predefined field from another predefined numeric field.

The format of the SUBTRACT command is as follows:

```
►►─┬──────────┬─ SUBTRACT ─┬─ fieldname1 ──────┬─┬──────┬─ fieldname2 ───►
   └─ label: ─┘            └─ numeric constant ┘ └ FROM ┘

►─┬──────────────────────────┬─┬─────────┬──────────────────────────────►◄
  └─┬─────────┬─ fieldname3 ─┘ └─ ROUND ─┘
    └─ GIVING ┘
```

label:

Specifies an optional identifying label, referenced within a GOTO command, that allows a GOTO branch to be made to this statement.

fieldname1

Specifies the name given a predefined numeric field whose value represents the subtrahend for the subtraction of two variables. Predefine the field name as zoned decimal, packed decimal, or binary.

numeric constant

Specifies an absolute numeric constant to be used as the subtrahend in the calculation.

fieldname2

Specifies the name given a predefined numeric field whose value represents the minuend for the subtraction of two variables. Predefine the field name as zoned decimal, packed decimal, or binary.

fieldname3

Optionally specifies a predefined numeric field in which the difference is placed. Predefine the field name as zoned decimal, packed decimal, or binary.

ROUND

Optionally defines whether the result will be rounded before being placed into the result field. For rounding to occur, the computed value must have more decimal places than the result field. The default is not to round.

Example

```
           Predefined field name:    FIELD1    FIELD2      FIELD3   INDEX
               Numeric precision:      3.2       1.4         3.4     2.0
          Initial value of field:  (123.45)  (5.4321)        (1)    (10)

SUBTRACT 2 FROM FIELD1               121.45        nc         nc      nc
SUBTRACT FIELD3 FROM INDEX               nc        nc         nc       9
SUBTRACT FIELD1 FROM
     FIELD2 GIVING FIELD3                nc        nc  -118.0179      nc
SUBTRACT -1.5 FROM FIELD1            124.95        nc         nc      nc
SUBTRACT 3 FROM FIELD2
     GIVING FIELD3                       nc        nc     2.4321      nc
SUBTRACT INDEX FROM INDEX                nc        nc         nc       0
SUBTRACT INDEX FROM FIELD3               nc        nc    -9.0000      nc
SUBTRACT FIELD2 INDEX FIELD2             nc    4.5679         nc      nc
SUBTRACT FIELD2 FROM INDEX               nc        nc         nc       4
SUBTRACT 16 FROM INDEX                   nc        nc         nc      -6
SUBTRACT +.92 FROM FIELD1            122.53        nc         nc      nc
SUBTRACT .333 FROM FIELD2                nc    5.0991         nc      nc
```

Note the following:

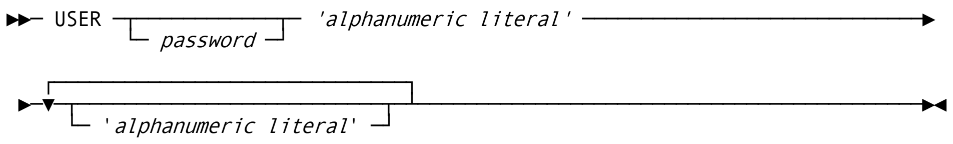
- The nc means no change in the original contents of the field.
- All significant digits of each of the operands take place in the subtraction, but the final result is aligned based on the DEFINE command options coded for the result field.
- The keywords FROM and GIVING are optional.

USER Command

(Area 1)

The USER command generates a report identification line as the first heading line for each page of each report. The command is required and must appear as the very first command or must immediately follow the first OPTION command.

The format of the USER command is as follows:



password

Is a 1- through 8-character password which must match the one selected at product modification time, identifying authority to delete call library members.

'*alphanumeric literal*'

Is a character string, enclosed in apostrophes, that is to appear on every page of each report generated by the Reporting Facility. You can specify multiple literals, such as 'CAI', 'REPORT', or 'CAI REPORT'.

Note the following:

- You can specify only one USER command for each processing run.
- The USER command must contain one or more alphanumeric literals. The system strings these literals together and the resultant literal prints on each page of each report.
- The user ID literal will be centered within the printer width currently in force. Subsequent resetting of the printer REPORT command width will only take effect at the time the REPORT command is processed.
- Since each literal must be contained on one source record, the system limits the length of the literal to 70 characters. The system limits the total length of the stringed literal to the width of the print line.

Example

The command:

```
USER 'XYZ COMPANY, INC.'
```

Is equal to:

```
USER  'XYZ'  'COMPANY,'  'INC.'
```

Chapter 12: Sample Reporting Facility Programs

The following pages demonstrate the use of Primary, Secondary, Write-Only, and Library Maintenance modes. For details about the modes, see Reporting Facility Modes of Operation (see page 105).

Primary Mode

The Primary mode is the most frequently entered mode of operation. It provides methods for resequencing input data or generating multiple reports.

There are four main processes performed in Primary mode:

- Compilation/Code Generation
- Record Selection/Data Manipulation
- Sort
- Report Generation

In Primary mode, a single pass of the input data files can produce multiple reports, with each report printed in a different sequence and containing different data.

Primary Mode Example 1

Following is a sample of the first page of the report. For an example of the report header (not shown here), see Sample Report Headers).

```
   1           OPTION  LIST ON MAP PRINTER=80
     2           USER    CA

   4           NOTE    **********************************************
   5           NOTE    *                                            *
   6           NOTE    *  THIS REPORTING FACILITY PROGRAM PRODUCES  *
   7           NOTE    *  A SINGLE REPORT, SHOWING A SUMMARY OF ALL *
   8           NOTE    *  CHECKS WRITTEN BY PAYEE.                  *
   9           NOTE    *                                            *
  10           NOTE    **********************************************

  12   TRANS:  INPUT   CARD
  13           DEF     TRANSACTION-CODE      1-2  X
  14           DEF     ID-NUMBER             3-6  X
  15           DEF     AMOUNT               7-13  N2 'TRANSACTION' 'AMOUNT  '
  16           DEF     PAYEE               23-42  X

  18           DEFINE  TRANSACTION-COUNT=1         'NO. OF   ' 'TRANSACTIONS'

  20           GO TO   START WHEN PAYEE EQ 'V.O.I.D.'

  22           REPORT  'CHECKS WRITTEN BY PAYEE'
  23           SELECT  (TRANSACTION-CODE EQ 'CK')
┌───────────────────────────────────────────┐
│ 24           CONTROL (PAYEE) ID-NUMBER    │ ◄  Enter Primary mode implicitly (sort required)
└───────────────────────────────────────────┘

┌──────────────────────────────────────────────────────────────────────┐
│ 25           PRINT   TOTALS ONLY PAYEE (TRANSACTION-COUNT) (AMOUNT)  │◄  Print summary report (no detail lines)
└──────────────────────────────────────────────────────────────────────┘

  26   END
       COMPILE PHASE COMPLETED - NO ERRORS FOUND
                                 NO WARNINGS ISSUED
       START 14:00:26 - STOP 14:00:27
```

Following is a sample of the second page of the report.

```
I/P  00001 X 00080 = 00080   TRANS
O/P  00085 X 00054 = 04590   HITFILE
SRT  00005 - 00030 = 00025   .SORT KEY INFO.

GSA  00000000919 (00000000060 ENTRIES)

     GSA 1 = 00000000001 NUMERIC CONSTANT ...... CON
     GSA 2 = 00000000010 QUOTED LITERAL ........ LIT
     GSA 3 = 00000000001 INTERMEDIATE RESULT ... RES
     GSA 4 = 00000000041 UNQUOTED LITERAL ...... INT
     GSA 5 = 00000000004 ACCUMULATOR ........... ACC
     GSA 6 = 00000000002 VALUE ................. VAL
     GSA 7 = 00000000000 STRING ................ STR
     GSA 8 = 00000000001 HEX LITERAL ........... HEX

FNT  00000000603 (00000000035 ENTRIES)

     FNT 0 = 00000000005 LABELS ...............  01
     FNT 1 = 00000000029 SCALAR VARIABLES ......  01
     FNT 2 = 00000000001 FILE DESC. BLOCKS .....  02
     FNT 3 = 00000000000 ARRAYS ...............  03

FST  00000002492 (00000000089 ENTRIES)

     FST 0 = 00000000004 INTERNAL ONLY ......... -02
     FST 1 = 00000000002 USER/REPORT ...........  01
     FST 2 = 00000000004 INPUT AREA VARIABLE ...  02
     FST 3 = 00000000063 GSA VARIABLE ..........  03
     FST 4 = 00000000001 DATA MOVEMENT .........  04
     FST 5 = 00000000000 DECODE ...............  05
     FST 6 = 00000000002 SEQUENCE AND CONTROL ..  06
     FST 7 = 00000000003 CONDITION TEST ........  07
     FST 8 = 00000000002 SELECTION ............  08
     FST 9 = 00000000004 PRINT LAYOUT ..........  09
     FST A = 00000000003 HIT RECORD EXTENSION ..  0A
     FST B = 00000000001 GET RECORD ...........  0B
     FST C = 00000000000 CONTROL BREAK CALC ....  0C
     FST D = 00000000000 UNDEFINED ............  0D
     FST E = 00000000000 ARRAY DEF/REL. INDEX ..  0E
     FST F = 00000000000 DUMMY ................  0F

GETMAIN REQUIREMENT FOR GSA/FNT/FST ... APPROX. 004K
```

Following is a sample of the third page of the report.

```
------ FIELD NAME ------ ORIGIN  QUAL  HIT ARY  ----------------------------------- REPORTS ------------------------------

ABORT:                   *LABEL*
AMOUNT                   TRANS   (01)   X         1
BLANK                    * GSA * (00)
CURRDATE                 * GSA * (00)
CURRENT-DATE             * GSA * (00)
CURRENT-TIME             * GSA * (00)
CURRTIME                 * GSA * (00)
DD                       * GSA * (00)
END-OF-FILE              * GSA * (00)
EOJ:                     *LABEL*
HR                       * GSA * (00)
ID-NUMBER                TRANS   (01)
MM                       * GSA * (00)
MN                       * GSA * (00)
NO-RECORD-FOUND          * GSA * (00)
PAYEE                    TRANS   (01)   X         1
QSEQ                     * GSA * (00)
RECORD-FOUND             * GSA * (00)
SPACE                    * GSA * (00)
SPACES                   * GSA * (00)
SS                       * GSA * (00)
START:                   *LABEL*
TAG                      * GSA * (00)
TEST:                    *LABEL*
TRANS                    * GSA * (00)
TRANS:                   *LABEL*
TRANSACTION-CODE         TRANS   (01)
TRANSACTION-COUNT        * GSA * (00)   X         1
XD                       * GSA * (00)
XM                       * GSA * (00)
XY                       * GSA * (00)
YY                       * GSA * (00)
ZERO                     * GSA * (00)
ZEROS                    * GSA * (00)
```

Following is a sample of the fourth page of the report.

```
                                                 (3350   ) ......H I T   F I L E   L A Y O U T......

FD  DROUT                                    DROUT:  FILE DISK SEQUENTIAL VARIABLE RECORD=         58 BLOCK=       4594
      BLOCK CONTAINS        4594 CHARACTERS
      RECORD CONTAINS          58 CHARACTERS
      RECORDING MODE V
      DATA RECORD IS HITFILE.

01  HITFILE.
      02 RDW                             PICTURE X(4)                DEFINE RDW                            1 -    4 X
      02 TAG-CHARACTER                   PICTURE X                   DEFINE TAG-CHARACTER                  5 -    5 X
      02 SORT-KEY.
         03 REPORT-NUMBER                PICTURE B                   DEFINE REPORT-NUMBER                  6 -    6 B
         03 USER-SORT-KEY                PICTURE X(24)               DEFINE USER-SORT-KEY                  7 -   30 X
      02 REPORT-01.                      CHECKS WRITTEN BY PAYEE
         03 01.PAYEE                     PICTURE X(20).              DEFINE 01.PAYEE                      31 -   50 X
         03 00.TRANSACTION-COUNT         PICTURE S9          COMP-3  DEFINE 00.TRANSACTION-COUNT          51 -   51 P
         03 01.AMOUNT                    PICTURE 9(5)V99             DEFINE 01.AMOUNT                     52 -   58 N2
```

```
CA                                         16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                  PAGE      5
  REPORT   01 FIELD NAME                   HDR1  HDR2  DTL   ACC   TOTAL       PICTURE            TAG   FROM   TO

  LINE   01   01.PAYEE                        5          20             20                              1    20
              00.TRANSACTION-COUNT            9    12     2     4       12                 9-          37    48
              01.AMOUNT                      11     8     9    11       11          99999.99-          65    75
```

```
RUN DIAGNOSTICS
INDEX VIOLATIONS       00000000000
PROGRAM CHECKS         00000000000
PRIMARY I/P RECORDS    00000000093  Run diagnostics appear before
HITS FOR REPORT   1  - 00000000074  output records in Primary mode.
TOTAL RECORDS SELECTED 00000000074
RECS DROPPED BY USER   00000000002
```

```
CA
24 JUN 2010                  CHECKS WRITTEN BY PAYEE                  PAGE     1
--------------------------------------------------------------------------------
PAYEE                                  NO. OF                      TRANSACTION
                                    TRANSACTIONS                      AMOUNT
--------------------------------------------------------------------------------
ABC NATIONAL BANK                            17                    1150.00
ACME FINANCE                                  4                     449.90
ACME HAIR DESIGN                              2                      34.00
ACME NATIONAL LIFE                            5                     222.12
ALL-NITE BOWLING                              5                      83.40
BERNY'S CHEVROLET                             4                    1537.00
CITY OF DALLAS                                1                       5.00
DISCOUNT AUTO STORE                           3                      99.34
FOOD MART                                     6                     326.15
H & L FLORISTS                                1                      44.95
JOHN DOE                                      2                     245.00
MODERNAGE TV                                  2                     347.72
RICHBURG INS. AGENCY                          1                     412.89
RX DRUGS                                      1                      14.80
COMPANY XYZ                                   3                     195.87
SEARS ROEBUCK                                 2                     470.31
SOUTHERN AIRWAYS                              1                     227.89
SOUTHWEST TITLE CO.                           5                    2598.00
SOUTHWESTERN BELL                             3                     157.24
TEXAS POWER & LIGHT                           6                     339.74
                                            ----                ----------- 
GRAND TOTAL                                  74                    8961.32
                                            ----                -----------
```

Primary Mode Example 2

Following is a sample of the first page of the report.

```
         OPTION  LIST ON MAP
           USER    CA

         NOTE    **********************************************
         NOTE    *                                            *
         NOTE    *     A C T I V I T Y    R E G I S T E R     *
         NOTE    *                  BY MONTH                  *
         NOTE    *                                            *
         NOTE    *  THIS REPORTING FACILITY PROGRAM PRODUCES  *
         NOTE    *  A SINGLE REPORT, SHOWING A DETAIL LIST OF *
         NOTE    *  ALL ACTIVITY, WITH CONTROL TOTALS TAKEN   *
         NOTE    *  BY MONTH.                                 *
         NOTE    *                                            *
         NOTE    **********************************************

TRANS:   INPUT   CARD
         DEF     TRANSACTION-CODE       1-2  X
         DEF     ID-NUMBER              3-6  X
         DEF     AMOUNT                7-13  N2
         DEF     ACTIVITY-DATE        14-21  X 'DATE  OF' 'ACTIVITY'
         DEF     CLEARED                 22  X
         DEF     PAYEE                23-42  X
         DEF     REASON               44-75  X
         DEF     ACTMONTH(2.0) = 0           N
         DEF     ALPHAMONTH(9)=' '           N
         DEF     ACTDAY=ACTIVITY-DATE   4-5  N
         DEF     ACTYEAR=ACTIVITY-DATE  7-8  N
         DEF     PRINTTYPE(9)=' '              '  TYPE /' '  STATUS'
         DEF     PRTCODE=PRINTTYPE      1-2  X
         DEF     PRTIDNUMBER=PRINTTYPE  4-7  X
         DEF     PRTSTAT=PRINTTYPE        9  X
         DEFINE  RUNNING-BALANCE(9.2)=0        'RUNNING  ' 'BALANCE  '
                                                PIC '$$$,$$$,$$9.99-'
         DEFINE  NEGATIVE(5.2) EQ 0 ' ' '-(MINUS)'   PIC '$$,$$9.99-'
         DEFINE  POSITIVE(5.2) EQ 0 ' ' '+(PLUS)'    PIC '$$,$$9.99-'

         DECODE    ACTMONTH INTO ALPHAMONTH
                   01 = 'JANUARY'
                   02 = 'FEBRUARY'
                   03 = 'MARCH'
                   04 = 'APRIL'
                   05 = 'MAY'
                   06 = 'JUNE'
                   07 = 'JULY'
                   08 = 'AUGUST'
                   09 = 'SEPTEMBER'
                   10 = 'OCTOBER'
                   11 = 'NOVEMBER'
                         ELSE 'DECEMBER'
```

Following is a sample of the second page of the report.

```
 51          SET       PRTCODE EQUAL TO TRANSACTION-CODE
 52          SET       PRTIDNUMBER EQUAL TO ID-NUMBER
 53          MOVE      CLEARED TO PRTSTAT
 54          GO TO     PLUS WHEN TRANSACTION-CODE EQ 'BF'
 55                              OR WHEN TRANSACTION-CODE EQ 'DP'
 56          MOVE      AMOUNT TO NEGATIVE
 57          SUBTRACT  AMOUNT FROM RUNNING-BALANCE
 58          GOTO      TEST
 59   PLUS:  CONT
 60          MOVE      AMOUNT TO POSITIVE
 61          INCREMENT RUNNING-BALANCE BY AMOUNT

 63          REPORT    'ACTIVITY REGISTER BY MONTH'
 64                    'TOTALS FOR MONTH OF' ALPHAMONTH '19' 0 ACTYEAR
 65          SELECT    'A' WHERE TRANSACTION-CODE EQ 'BF'
 66                        OR WHEN TRANSACTION-CODE EQ 'DP'
 67          SELECT    'B' ALL

 68          CONTROL   ACTYEAR (ACTMONTH) SKIP ACTDAY QSEQ   ◄  Sort required

 69          PRINT     ACTIVITY-DATE PRINTTYPE PAYEE REASON
 70                    (A;POSITIVE) (B;NEGATIVE) RUNNING-BALANCE
 71   END


      COMPILE PHASE COMPLETED - NO ERRORS FOUND
                                NO WARNINGS ISSUED
      START 14:01:16 - STOP 14:01:17
```

Following is a sample of the third page of the report.

```
I/P  00001 X 00080 = 00080   TRANS
O/P  00035 X 00132 = 04620   HITFILE
SRT  00005 - 00042 = 00037   .SORT KEY INFO.

GSA  00000001249 (00000000117 ENTRIES)

     GSA 1 = 00000000012 NUMERIC CONSTANT ...... CON
     GSA 2 = 00000000039 QUOTED LITERAL ........ LIT
     GSA 3 = 00000000001 INTERMEDIATE RESULT ... RES
     GSA 4 = 00000000056 UNQUOTED LITERAL ...... INT
     GSA 5 = 00000000004 ACCUMULATOR ........... ACC
     GSA 6 = 00000000004 VALUE ................. VAL
     GSA 7 = 00000000000 STRING ................ STR
     GSA 8 = 00000000001 HEX LITERAL ........... HEX

FNT  00000000853 (00000000049 ENTRIES)

     FNT 0 = 00000000006 LABELS ................  01
     FNT 1 = 00000000042 SCALAR VARIABLES ......  01
     FNT 2 = 00000000001 FILE DESC. BLOCKS .....  02
     FNT 3 = 00000000000 ARRAYS ................  03

FST  00000005348 (00000000191 ENTRIES)

     FST 0 = 00000000004 INTERNAL ONLY ......... -02
     FST 1 = 00000000002 USER/REPORT ...........  01
     FST 2 = 00000000010 INPUT AREA VARIABLE ...  02
     FST 3 = 00000000119 GSA VARIABLE ..........  03
     FST 4 = 00000000009 DATA MOVEMENT .........  04
     FST 5 = 00000000012 DECODE ................  05
     FST 6 = 00000000004 SEQUENCE AND CONTROL ..  06
     FST 7 = 00000000006 CONDITION TEST ........  07
     FST 8 = 00000000003 SELECTION .............  08
     FST 9 = 00000000012 PRINT LAYOUT ..........  09
     FST A = 00000000009 HIT RECORD EXTENSION ..  0A
     FST B = 00000000001 GET RECORD ............  0B
     FST C = 00000000000 CONTROL BREAK CALC ....  0C
     FST D = 00000000000 UNDEFINED .............  0D
     FST E = 00000000000 ARRAY DEF/REL. INDEX ..  0E
     FST F = 00000000000 DUMMY .................  0F

GETMAIN REQUIREMENT FOR GSA/FNT/FST ... APPROX. 008K
```

Following is a sample of the fourth page of the report.

```
------ FIELD NAME ------ ORIGIN  QUAL  HIT ARY  ---------------------------------- REPORTS ------------------------------

ABORT:                   *LABEL*
ACTDAY                   TRANS   (01)
ACTIVITY-DATE            TRANS   (01)   X         1
ACTMONTH                 TRANS   (01)
ACTYEAR                  TRANS   (01)   X         1
ALPHAMONTH               * GSA * (00)   X         1
AMOUNT                   TRANS   (01)
BLANK                    * GSA * (00)
CLEARED                  TRANS   (01)
CURRDATE                 * GSA * (00)
CURRENT-DATE             * GSA * (00)
CURRENT-TIME             * GSA * (00)
CURRTIME                 * GSA * (00)
DD                       * GSA * (00)
END-OF-FILE              * GSA * (00)
EOJ:                     *LABEL*
HR                       * GSA * (00)
ID-NUMBER                TRANS   (01)
MM                       * GSA * (00)
MN                       * GSA * (00)
NEGATIVE                 * GSA * (00)   X         1
NO-RECORD-FOUND          * GSA * (00)
PAYEE                    TRANS   (01)   X         1
PLUS:                    *LABEL*
POSITIVE                 * GSA * (00)   X         1
PRINTTYPE                * GSA * (00)   X         1
PRTCODE                  * GSA * (00)
PRTIDNUMBER              * GSA * (00)
PRTSTAT                  * GSA * (00)
QSEQ                     * GSA * (00)
REASON                   TRANS   (01)   X         1
RECORD-FOUND             * GSA * (00)
RUNNING-BALANCE          * GSA * (00)   X         1
SPACE                    * GSA * (00)
SPACES                   * GSA * (00)
SS                       * GSA * (00)
START:                   *LABEL*
TAG                      * GSA * (00)
TEST:                    *LABEL*
TRANS                    * GSA * (00)
TRANS:                   *LABEL*
TRANSACTION-CODE         TRANS   (01)
XD                       * GSA * (00)
XM                       * GSA * (00)
XY                       * GSA * (00)
YY                       * GSA * (00)
ZERO                     * GSA * (00)
ZEROS                    * GSA * (00)
```

Following is a sample of the fifth page of the report.

```
                                              (3350   ) ......H I T   F I L E   L A Y O U T......

 FD  DROUT                                DROUT:  FILE DISK SEQUENTIAL VARIABLE RECORD=        136 BLOCK=       4624
       BLOCK CONTAINS        4624 CHARACTERS
       RECORD CONTAINS        136 CHARACTERS
       RECORDING MODE V
       DATA RECORD IS HITFILE.

 01  HITFILE.
       02 RDW                             PICTURE X(4)                DEFINE RDW                             1 -    4 X
       02 TAG-CHARACTER                   PICTURE X                   DEFINE TAG-CHARACTER                   5 -    5 X
       02 SORT-KEY.
          03 REPORT-NUMBER                PICTURE B                   DEFINE REPORT-NUMBER                   6 -    6 B
          03 USER-SORT-KEY                PICTURE X(36)               DEFINE USER-SORT-KEY                   7 -   42 X
       02 REPORT-01.                      ACTIVITY REGISTER BY MONTH
          03 00.ALPHAMONTH                PICTURE X(9).               DEFINE 00.ALPHAMONTH                  43 -   51 X
          03 01.ACTYEAR                   PICTURE 99                  DEFINE 01.ACTYEAR                     52 -   53 N
          03 01.ACTIVITY-DATE             PICTURE X(8)                DEFINE 01.ACTIVITY-DATE               54 -   61 X
          03 00.PRINTTYPE                 PICTURE X(9)                DEFINE 00.PRINTTYPE                   62 -   70 X
          03 01.PAYEE                     PICTURE X(20)               DEFINE 01.PAYEE                       71 -   90 X
          03 01.REASON                    PICTURE X(32)               DEFINE 01.REASON                      91 -  122 X
          03 00.POSITIVE                  PICTURE S9(5)V99     COMP-3 DEFINE 00.POSITIVE                   123 -  126 P2
          03 00.NEGATIVE                  PICTURE S9(5)V99     COMP-3 DEFINE 00.NEGATIVE                   127 -  130 P2
          03 00.RUNNING-BALANCE           PICTURE S9(9)V99     COMP-3 DEFINE 00.RUNNING-BALANCE            131 -  136 P2
```

Following is a sample of the sixth page of the report.

```
  REPORT   01 FIELD NAME                 HDR1  HDR2  DTL   ACC   TOTAL      PICTURE          TAG   FROM   TO

  HEADING     00.* LITERAL *                           19           19                                 1    19
              00.ALPHAMONTH                             9            9                                21    29
              00.* LITERAL *                            2            2                                31    32
              01.ACTYEAR                                3            3              99-               33    35

  LINE   01   01.ACTIVITY-DATE              8     8     8            8                                 1     8
              00.PRINTTYPE                  8     8     9            9                                12    20
              01.PAYEE                      5    20    20    24     43
              01.REASON                     6          32           32                                47    78
              00.POSITIVE                         7    11    13     13        $$,$$9.99-        A     83    95
              00.NEGATIVE                         8    11    13     13        $$,$$9.99-        B    100   112
              00.RUNNING-BALANCE            9     9    16           16    $$$,$$$,$$9.99-            117   132
```

```
RUN DIAGNOSTICS
INDEX VIOLATIONS       00000000000
PROGRAM CHECKS         00000000000
PRIMARY I/P RECORDS    00000000093
HITS FOR REPORT   1  - 00000000093
TOTAL RECORDS SELECTED 00000000093
```

```
CA
24 JUN 2010                                        ACTIVITY REGISTER BY MONTH                            PAGE      1
TOTALS FOR MONTH OF JANUARY   2010
----------------------------------------------------------------------------------------------------------------------------
DATE  OF     TYPE /    PAYEE                 REASON                                                             RUNNING
ACTIVITY     STATUS                                                         +(PLUS)           -(MINUS)          BALANCE
----------------------------------------------------------------------------------------------------------------------------
01/01/10   BF                                BALANCE BROUGHT FORWARD      000      $109.12                      $109.12
01/01/10   DP                                                             000      $500.00                      $609.12
01/01/10   CK 1200 X   ABC NATIONAL BANK      CASH                         000                       $50.00      $559.12
01/01/10   CK 1201 X   ACME NATIONAL LIFE     JANUARY INS. PREMIUM         000                       $37.02      $522.10
01/01/10   CK 1202 X   SOUTHWEST TITLE CO.    JANUARY MORTGAGE PAYMNT      000                      $433.00       $89.10
01/03/10   CK 1203 X   TEXAS POWER & LIGHT    UTILITY BILL FOR DEC'00      000                       $62.15       $26.95
01/10/10   CK 1204 X   FOOD MART              GROCERIES                    000                       $22.87        $4.08
01/15/10   DP                                                             000      $824.07                      $828.15
01/15/10   CK 1205 X   RICHBURG INS. AGENCY   CAR INSURANCE                000                      $412.89      $415.26
01/22/10   CK 1206 X   ABC NATIONAL BANK      CASH                         000                      $100.00      $315.26
01/22/10   CK 1207 X   SOUTHWESTERN BELL      PHONE BILL                   000                       $22.03      $293.23
01/23/10   CK 1208 X   ACME HAIR DESIGN       HAIRCUT                      000                       $17.00      $276.23
01/25/10   CK 1209 X   SEARS ROEBUCK          DRYER                        000                      $213.56       $62.67
01/27/10   CK 1210 X   ALL-NITE BOWLING       BOWLING BAG                  000                       $16.75       $45.92
01/30/10   CK 1211 X   ACME FINANCE           LOAN PAYMENT 1               000                       $89.98       $44.06-
01/31/10   NS                                                             000                        $6.00       $50.06-
01/31/10   SC                                                             000                        $4.00       $54.06-
                                                                      ------------      ------------
                                                                        $1,433.19        $1,487.25
                                                                      ------------      ------------
                                                          Page skip at control break
```

```
CA
24 JUN 2010                                        ACTIVITY REGISTER BY MONTH                            PAGE     2
2
TOTALS FOR MONTH OF FEBRUARY  2010
-----------------------------------------------------------------------------------------------------------------------------
DATE  OF     TYPE /    PAYEE              REASON                                                                 RUNNING
ACTIVITY     STATUS                                                         +(PLUS)           -(MINUS)           BALANCE
-----------------------------------------------------------------------------------------------------------------------------
02/01/10   DP                                                             000      $922.00                       $867.94
02/01/10   CK 1212 X   JOHN DOE               RETURN OF CASH ADVANCE       000                      $150.00       $717.94
02/03/10   CK 1213 X   SOUTHWEST TITLE CO.    FEBRUARY MORTGAGE PAYMENT    000                      $433.00       $284.94
02/05/10   CK 1214 X   TEXAS POWER & LIGHT    UTILITY BILL FOR JAN'01      000                       $59.11       $225.83
02/09/10   CK 1215 X   ACME NATIONAL LIFE     FEBRUARY INS. PREMIUM        000                       $37.02       $188.81
02/11/10   CK 1216 X   FOOD MART              GROCERIES                    000                       $68.29       $120.52
02/11/10   CK 1217 X   ABC NATIONAL BANK      CASH                         000                       $25.00        $95.52
02/17/10   DP                                1992 INCOME TAX REFUND       000    $1,538.00                     $1,633.52
02/20/10   CK 1218 X   ABC NATIONAL BANK      CASH                         000                       $50.00     $1,583.52
02/21/10   CK 1219 X   ACME FINANCE           LOAN PAYMENTS 2 AND 3        000                      $179.96     $1,403.56
02/27/10   CK 1220 X   SOUTHWEST TITLE CO.    MARCH/APRIL MORTGAGE PAYMENT 000                      $866.00       $537.56
02/28/10   CK 1221 X   COMPANY XYZ            SUIT                         000                      $120.42       $417.14
```

```
CA
24 JUN 2010                                        ACTIVITY REGISTER BY MONTH                            PAGE     3

TOTALS FOR MONTH OF MARCH     2010
-----------------------------------------------------------------------------------------------------------------------------
DATE  OF     TYPE /    PAYEE              REASON                                                                 RUNNING
ACTIVITY     STATUS                                                         +(PLUS)           -(MINUS)           BALANCE
-----------------------------------------------------------------------------------------------------------------------------
03/02/10   CK 1222 X   TEXAS POWER & LIGHT    UTILITY BILL FOR FEB'01      000                       $62.11       $355.03
03/02/10   CK 1223 X   ABC NATIONAL BANK      CASH                         000                       $25.00       $330.03
03/03/10   SC                                                             000                        $4.00       $326.03
03/09/10   CK 1224 X   FOOD MART              GROCERIES                    000                       $72.80       $253.23
03/12/10   CK 1225 X   SOUTHERN AIRWAYS       TICKET TO LUBBOCK            000                      $227.89        $25.34
03/15/10   DP                                                             000      $922.00                       $947.34
03/16/10   CK 1226 X   ABC NATIONAL BANK      CASH                         000                      $250.00       $697.34
03/27/10   CK 1227 X   ACME NATIONAL LIFE     MARCH/APRIL INS. PREMIUMS    000                       $74.04       $623.30
03/30/10   CK 1228 X   DISCOUNT AUTO STORE    BATTERY                      000                       $35.70       $587.60
```

```
CA
24 JUN 2010                                            ACTIVITY REGISTER BY MONTH                                  PAGE    4

TOTALS FOR MONTH OF APRIL     2010
--------------------------------------------------------------------------------------------------------------------------------
DATE  OF     TYPE /    PAYEE                  REASON                                                                    RUNNING
ACTIVITY     STATUS                                                                +(PLUS)          -(MINUS)           BALANCE
--------------------------------------------------------------------------------------------------------------------------------
04/01/10   DP                                                              000       $922.00                           $1,509.60
04/02/10   CK 1229 X   V.O.I.D.                                            000                          $0.00          $1,509.60
04/02/10   CK 1230 X   TEXAS POWER & LIGHT    UTILITY BILL FOR MAR'01      000                         $47.18          $1,462.42
04/02/10   CK 1231 X   FOOD MART              GROCERIES                    000                         $25.90          $1,436.52
04/02/10   CK 1232 X   ABC NATIONAL BANK      CASH                         000                        $100.00          $1,336.52
04/10/10   CK 1233 X   ACME FINANCE           LOAN PAYMENT 4               000                         $89.98          $1,246.54
04/10/10   CK 1234 X   SOUTHWESTERN BELL      PHONE BILLS                  000                         $79.82          $1,166.72
04/10/10   SC                                                              000                          $4.00          $1,162.72
04/10/10   CK 1235 X   ABC NATIONAL BANK      CASH                         000                         $50.00          $1,112.72
04/11/10   CK 1236 X   DISCOUNT AUTO STORE    TUNE-UP KIT                  000                         $21.60          $1,091.12
04/15/10   CK 1237 X   BERNY'S CHEVROLET      DOWN PAYMENT ON CORVETTE     000                      $1,000.00             $91.12
04/15/10   DP                                 PARTIAL SAVINGS WITHDRAWAL   000     $2,922.00                           $3,013.12
04/15/10   CK 1238 X   ABC NATIONAL BANK      CASH                         000                         $75.00          $2,938.12
04/20/10   CK 1239 X   CITY OF DALLAS         PARKING TICKET               000                          $5.00          $2,933.12
04/20/10   CK 1240 X   ALL-NITE BOWLING       ENTERTAINMENT                000                         $11.55          $2,921.57
04/22/10   CK 1241 X   MODERNAGE TV           REPAIRS                      000                         $75.22          $2,846.35
04/25/10   CK 1242 X   BERNY'S CHEVROLET      CAR PAYMENT 1                000                        $179.00          $2,667.35
04/26/10   CK 1243 X   ABC NATIONAL BANK      CASH                         000                         $25.00          $2,642.35
04/26/10   CK 1244 X   ACME FINANCE           LOAN PAYMENT 5               000                         $89.98          $2,552.37
04/29/10   CK 1245 X   SOUTHWEST TITLE CO.    MAY MORTGAGE PAYMENT         000                        $433.00          $2,119.37
```

```
CA
24 JUN 2010                                            ACTIVITY REGISTER BY MONTH                                  PAGE    5

TOTALS FOR MONTH OF MAY       2010
--------------------------------------------------------------------------------------------------------------------------------
DATE  OF     TYPE /    PAYEE                  REASON                                                                    RUNNING
ACTIVITY     STATUS                                                                +(PLUS)          -(MINUS)           BALANCE
--------------------------------------------------------------------------------------------------------------------------------
05/01/10   DP                                                              000       $922.00                           $3,041.37
05/01/10   CK 1246 X   ABC NATIONAL BANK      CASH                         000                         $50.00          $2,991.37
05/01/10   CK 1247 X   ACME NATIONAL LIFE     MAY INS. PREMIUM             000                         $37.02          $2,954.35
05/01/10   CK 1248 X   TEXAS POWER & LIGHT    UTILITY BILL FOR APR'01      000                         $55.98          $2,898.37
05/01/10   CK 1249 X   BERNY'S CHEVROLET      CAR PAYMENT 2                000                        $179.00          $2,719.37
05/02/10   CK 1250 X   COMPANY XYZ            BRIEFCASE                    000                         $47.56          $2,671.81
05/02/10   SC                                                              000                          $4.00          $2,667.81
05/05/10   CK 1251 X   ALL-NITE BOWLING       ENTERTAINMENT                000                         $21.60          $2,646.21
05/06/10   CK 1252 X   ABC NATIONAL BANK      CASH                         000                         $50.00          $2,596.21
05/10/10   CK 1253 X   SEARS ROEBUCK          WASHER                       000                        $256.75          $2,339.46
05/11/10   CK 1254 X   FOOD MART              GROCERIES                    000                         $53.89          $2,285.57
05/15/10   CK 1255 X   DISCOUNT AUTO STORE    MISC. AUTO PARTS             000                         $42.04          $2,243.53
05/15/10   DP                                                              000       $962.00                           $3,205.53
05/15/10   CK 1256 X   BERNY'S CHEVROLET      CAR PAYMENT 3                000                        $179.00          $3,026.53
05/15/10   CK 1257 X   H & L FLORISTS         ROSES                        000                         $44.95          $2,981.58
05/15/10   CK 1258 X   COMPANY XYZ            MISC.                        000                         $27.89          $2,953.69
05/16/10   CK 1259 X   ABC NATIONAL BANK      CASH                         000                         $50.00          $2,903.69
05/17/10   CK 1260 X   MODERNAGE TV           TV                           000                        $272.50          $2,631.19
05/17/10   CK 1261 X   RX DRUGS               PRESCRIPTION                 000                         $14.80          $2,616.39
05/17/10   CK 1262 X   ACME HAIR DESIGN       HAIRCUT                      000                         $17.00          $2,599.39
05/18/10   CK 1263 X   SOUTHWESTERN BELL      PHONE BILL                   000                         $55.39          $2,544.00
05/18/10   CK 1264 X   ALL-NITE BOWLING       ENTERTAINMENT                000                         $22.50          $2,521.50
05/18/10   CK 1265 X   JOHN DOE               RETURN CASH ADVANCE          000                         $95.00          $2,426.50
05/18/10   CK 1266 X   ABC NATIONAL BANK      CASH                         000                         $25.00          $2,401.50
05/20/10   CK 1267     ACME NATIONAL LIFE     JUNE INS. PREMIUM            000                         $37.02          $2,364.48
05/21/10   CK 1268     SOUTHWEST TITLE CO.    JUNE MORTGAGE PAYMENT        000                        $433.00          $1,931.48
05/21/10   CK 1269     ABC NATIONAL BANK      CASH                         000                         $25.00          $1,906.48
```

```
CA
24 JUN 2010                                 ACTIVITY REGISTER BY MONTH                              PAGE    6

TOTALS FOR MONTH OF JUNE      2010
----------------------------------------------------------------------------------------------------------------------------------
DATE  OF     TYPE /    PAYEE                  REASON                                                                RUNNING
ACTIVITY     STATUS                                                            +(PLUS)          -(MINUS)            BALANCE
----------------------------------------------------------------------------------------------------------------------------------
06/01/10   DP                                                         000      $962.00                            $2,868.48
06/01/10   CK 1270     TEXAS POWER & LIGHT    UTILITY BILL FOR MAY'01      000                     $53.21            $2,815.27
06/02/10   CK 1271     ABC NATIONAL BANK      CASH                         000                     $50.00            $2,765.27
06/02/10   CK 1272     FOOD MART              GROCERIES                    000                     $82.40            $2,682.87
06/02/10   CK 1273     ALL-NITE BOWLING       ENTERTAINMENT                000                     $11.00            $2,671.87
06/05/10   CK 1274 X   V.O.I.D.                                            000                      $0.00            $2,671.87
06/05/10   CK 1275     ABC NATIONAL BANK      CASH                         000                    $150.00            $2,521.87
06/15/10   DP                                                         000      $962.00                            $3,483.87
                                                                            -------------    -------------
                                                                              $1,924.00          $346.61
                                                                            -------------    -------------
                                                                            -------------    -------------
GRAND TOTAL                                                                   $12,467.19       $8,983.32
```

Primary Mode Example 3

Following is a sample of the first page of the report.

```
                 ********************************************************************************     1           OPTION LIST ON MAP
     2           USER   'XYZ INC.'
                        ┌──────────┐
     3           CALL   │ CARDFILE │  ◄  Cataloged library member
                        └──────────┘
     4  NOTE   CUSTOMER/SALESMAN DATA FILE
     5  INPT:    FILE   CARD
     6           DEFINE NAME                  1 TO  2 X
     7           DEFINE CUSTOMER-NUMBER       3 TO  5 X  'CUSTOMER'  'NUMBER'
     8           DEFINE CITY                        6 X
     9           DEFINE STATE                 7 TO  8 X
    10           DEFINE ZIP-CODE              9 TO 13 X  'ZIP'  'CODE'
    11           DEFINE CUSTOMER-SALE-ID     14 TO 15 X  'SALESMAN'  'ID'
    12           DEFINE CREDIT-LIMIT         16 TO 23 N2 'CREDIT'  'LIMIT'
    13                                                   PIC '$$$,$$9.99'
    14           DEFINE CURRENT-BALANCE      24 TO 30 N2 'CURRENT'  'BALANCE'
    15                                                   PIC '$$,$$9.99'
    16           DEFINE SALESMAN-NAME        32 TO 39 X  'SALESMAN'  'NAME'
    17           DEFINE SALESMAN-ID          40 TO 41 X  'SALESMAN'  '   ID   '
    18           DEFINE YTD-SALES            42 TO 49 N2 'YEAR-TO-DATE'  'SALES'
    19                                                   PIC '$$$,$$9.99'
    20           DEFINE BRANCH-ID            50 TO 52 X  'BRANCH'  '   ID   '

    22           DEF    UNUSED-CREDIT(6.2)=0       'UNUSED' 'CREDIT'
    23                                              PIC '***,**9.99-'
Cataloged library member
                       ▼
                        ┌───────┐     ┌─────┐
    24           CALL   │DECODE2│USING│STATE│ST    ◄  Library macro substitution
                        └───────┘     └─────┘
    25  NOTE   STATE CODE TRANSLATION TABLE
                         ┌───┐
    26           DECODE  │:01│INTO :02             ◄  Library macro substitution
                         └───┘
    27                   'TX' EQ 'TEXAS'
    28                   'TN' EQ 'TENNESSEE'
    29                   'NC' EQ 'NORTH CAROLINA'
    30                   'GA' EQ 'GEORGIA'
    31                   'OK' EQ 'OKLAHOMA'
    32                   'NY' EQ 'NEW YORK'
    33                   'OH' EQ 'OHIO'
    34                   'NJ' EQ 'NEW JERSEY'
    35                   'CT' EQ 'CONNECTICUT'
    36                   'IL' EQ 'ILLINOIS'
    37                   'PN' EQ 'PENNSYLVANIA'
    38                   'MI' EQ 'MICHIGAN'
    39                   'CO' EQ 'COLORADO'
    40                   'WA' EQ 'WASHINGTON'
    41                   'CA' EQ 'CALIFORNIA'
    42                            ELSE  'UNKNOWN'   'STATE'  'NAME'
```

Following is a sample of the second page of the report.

```
44           CALL   DECODE3 USING NAME CUSTOMER-NAME
44
45   NOTE   CUSTOMER NAME TRANSLATION TABLE
46           DECODE :01 INTO :02
47                   '01' EQ 'HIGH-ROLLING INVESTMENT '
48                   '02' EQ 'SOUTHERN FRIED FOODS    '
49                   '03' EQ 'LEGAL TOBACCO CO.       '
50                   '04' EQ 'SOUTHERN PINE INDUSTRIES'
51                   '05' EQ 'BARONIAL OIL CO.        '
52                   '06' EQ 'COSMOPOLITAN FASHIONS   '
53                   '07' EQ 'HEAVY METAL MACHINERY   '
54                   '08' EQ 'AIRPORT SERVICES CORP.  '
55                   '09' EQ 'STOLID INSURANCE CORP.  '
56                   '10' EQ 'INLAND GRAIN TERMINALS  '
57                   '11' EQ 'STEEL CURTAIN STEEL INC.'
58                   '12' EQ 'PERFECT BEARING CORP.   '
59                   '13' EQ 'MOUNTAIN STATES MINING  '
60                   '14' EQ 'NORTHWEST PLYWOOD MILLS '
61                   '15' EQ 'ORIENTAL TRADING CO.    '
62                   '16' EQ 'WEST COAST LIFESTYLES   '
63                            OTHERWISE ' '  'NAME'
64           CALL   DECODE4 USING CITY CUSTOMER-CITY
64
65   NOTE   CITY TRANSLATION TABLE
66           DECODE :01 INTO :02
67                   'A' EQ 'DALLAS'
68                   'B' EQ 'MEMPHIS'
69                   'C' EQ 'CHARLOTTE'
70                   'D' EQ 'ATLANTA'
71                   'E' EQ 'OKLAHOMA CITY'
72                   'F' EQ 'NEW YORK'
73                   'G' EQ 'CLEVELAND'
74                   'H' EQ 'NEWARK'
75                   'I' EQ 'HARTFORD'
76                   'J' EQ 'CHICAGO'
77                   'K' EQ 'PITTSBURGH'
78                   'L' EQ 'DETROIT'
79                   'M' EQ 'DENVER'
80                   'N' EQ 'SEATTLE'
81                   'O' EQ 'SAN FRANCISCO'
82                   'P' EQ 'LOS ANGELES'    ELSE ' '  'CITY'
```

Following is a sample of the third page of the report.

```
  84          SUBTRACT CURRENT-BALANCE FROM CREDIT-LIMIT
  85                   GIVING UNUSED-CREDIT

  87          REPORT  'XYZ SAMPLE REPORT 01'
  88          SELECT  ALL
  89          CONTROL QSEQ
  90          PRINT   CUSTOMER-NAME CUSTOMER-SALE-ID CUSTOMER-CITY STATE
  91                  ZIP-CODE CREDIT-LIMIT SALESMAN-ID CURRENT-BALANCE

  93          REPORT  'XYZ SAMPLE REPORT 02'
  94          SELECT  ALL
  95          CONTROL NAME
  96          PRINT   DOUBLE SPACING CUSTOMER-NAME CUSTOMER-NUMBER
  97                  CUSTOMER-CITY ST ZIP-CODE
  98                  CUSTOMER-SALE-ID CREDIT-LIMIT

 100          REPORT  'XYZ SAMPLE REPORT 03'
 101          SELECT  'A' (UNUSED-CREDIT GT 0)                         Multiple report definition
 102          SELECT  ALL                                              groups (sort required)
 103          CONTROL ST CITY
                       ▼  Exact location printing
                      ┌───┐
 104          PRINT   │@20│CUSTOMER-NAME 20 CUSTOMER-NUMBER 3 CREDIT-LIMIT
                      └───┘
 105                  3 (CURRENT-BALANCE) 3 (A;UNUSED-CREDIT)
                            ▼  Exact location printing
                      ┌──────────────┐
 106          PRINT   │@CUSTOMER-NAME│CUSTOMER-CITY 3 ST 3 ZIP-CODE
                      └──────────────┘

 108          REPORT  'XYZ SAMPLE REPORT 04'
 109          SELECT  ALL
 110          CONTROL NAME
 111          PRINT   CUSTOMER-NAME CUSTOMER-NUMBER CUSTOMER-CITY
 112                  ST ZIP-CODE CUSTOMER-SALE-ID  CREDIT-LIMIT
 113   END
       COMPILE PHASE COMPLETED - NO ERRORS FOUND
                                  001 WARNING(S) ISSUED
       START 16:54:54 - STOP 16:54:57
```

Following is a sample of the fourth page of the report.

```
I/P  00001 X 00080 = 00080   INPT
O/P  00048 X 00096 = 04608   HITFILE
SRT  00005 - 00021 = 00016   .SORT KEY INFO.

GSA  00000001919 (00000000193 ENTRIES)

     GSA 1 = 00000000002 NUMERIC CONSTANT ...... CON
     GSA 2 = 00000000136 QUOTED LITERAL ........ LIT
     GSA 3 = 00000000001 INTERMEDIATE RESULT ... RES
     GSA 4 = 00000000049 UNQUOTED LITERAL ...... INT
     GSA 5 = 00000000002 ACCUMULATOR ........... ACC
     GSA 6 = 00000000002 VALUE ................. VAL
     GSA 7 = 00000000000 STRING ................ STR
     GSA 8 = 00000000001 HEX LITERAL ........... HEX

FNT  00000000820 (00000000046 ENTRIES)

     FNT 0 = 00000000005 LABELS ...............  01
     FNT 1 = 00000000040 SCALAR VARIABLES ......  01
     FNT 2 = 00000000001 FILE DESC. BLOCKS .....  02
     FNT 3 = 00000000000 ARRAYS ...............  03

FST  00000009828 (00000000351 ENTRIES)

     FST 0 = 00000000004 INTERNAL ONLY ......... -02
     FST 1 = 00000000005 USER/REPORT ...........  01
     FST 2 = 00000000012 INPUT AREA VARIABLE ...  02
     FST 3 = 00000000191 GSA VARIABLE ..........  03
     FST 4 = 00000000002 DATA MOVEMENT .........  04
     FST 5 = 00000000050 DECODE ...............  05
     FST 6 = 00000000005 SEQUENCE AND CONTROL ..  06
     FST 7 = 00000000002 CONDITION TEST ........  07
     FST 8 = 00000000006 SELECTION ............  08
     FST 9 = 00000000035 PRINT LAYOUT ..........  09
     FST A = 00000000030 HIT RECORD EXTENSION ..  0A
     FST B = 00000000001 GET RECORD ...........  0B
     FST C = 00000000000 CONTROL BREAK CALC ....  0C
     FST D = 00000000000 UNDEFINED ............  0D
     FST E = 00000000000 ARRAY DEF/REL. INDEX ..  0E
     FST F = 00000000008 DUMMY ................  0F

GETMAIN REQUIREMENT FOR GSA/FNT/FST ... APPROX. 013K
```

Following is a sample of the fifth page of the report.

```
------ FIELD NAME ------ ORIGIN  QUAL  HIT ARY  ---------------------------------- REPORTS -----------------------------

ABORT:                   *LABEL*
BLANK                    * GSA * (00)
BRANCH-ID                INPT    (01)
CITY                     INPT    (01)
CREDIT-LIMIT             INPT    (01)   X         1  2  3  4
CURRDATE                 * GSA * (00)
CURRENT-BALANCE          INPT    (01)   X         1  3
CURRENT-DATE             * GSA * (00)
CURRENT-TIME             * GSA * (00)
CURRTIME                 * GSA * (00)
CUSTOMER-CITY            * GSA * (00)   X         1  2  3  4
CUSTOMER-NAME            * GSA * (00)   X         1  2  3  4
CUSTOMER-NUMBER          INPT    (01)   X         2  3  4
CUSTOMER-SALE-ID         INPT    (01)   X         1  2  4
DD                       * GSA * (00)
END-OF-FILE              * GSA * (00)
EOJ:                     *LABEL*
HR                       * GSA * (00)
INPT                     * GSA * (00)
INPT:                    *LABEL*
MM                       * GSA * (00)
MN                       * GSA * (00)
NAME                     INPT    (01)
NO-RECORD-FOUND          * GSA * (00)
QSEQ                     * GSA * (00)
RECORD-FOUND             * GSA * (00)
SALESMAN-ID              INPT    (01)   X         1
SALESMAN-NAME            INPT    (01)
SPACE                    * GSA * (00)
SPACES                   * GSA * (00)
SS                       * GSA * (00)
ST                       * GSA * (00)   X         2  3  4
START:                   *LABEL*
STATE                    INPT    (01)   X         1
TAG                      * GSA * (00)
TEST:                    *LABEL*
UNUSED-CREDIT            * GSA * (00)   X         3
XD                       * GSA * (00)
XM                       * GSA * (00)
XY                       * GSA * (00)
YTD-SALES                INPT    (01)
YY                       * GSA * (00)
ZERO                     * GSA * (00)
ZEROS                    * GSA * (00)
ZIP-CODE                 INPT    (01)   X         1  2  3  4
```

Following is a sample of the sixth page of the report.

```
                                                  (3350   ) ......H I T   F I L E   L A Y O U T......

FD  DROUT                                  DROUT:  FILE DISK SEQUENTIAL VARIABLE RECORD=        100 BLOCK=       4612
      BLOCK CONTAINS        4612 CHARACTERS
      RECORD CONTAINS        100 CHARACTERS
      RECORDING MODE V
      DATA RECORD IS HITFILE.

01  HITFILE.
      02 RDW                                PICTURE X(4)                DEFINE RDW                        1 -     4 X
      02 TAG-CHARACTER                      PICTURE X                   DEFINE TAG-CHARACTER              5 -     5 X
      02 SORT-KEY.
         03 REPORT-NUMBER                   PICTURE B                   DEFINE REPORT-NUMBER              6 -     6 B
         03 USER-SORT-KEY                   PICTURE X(15)               DEFINE USER-SORT-KEY              7 -    21 X
      02 REPORT-01.                         XYZ SAMPLE REPORT 01
         03 00.CUSTOMER-NAME                PICTURE X(24).              DEFINE 00.CUSTOMER-NAME          22 -    45 X
         03 01.CUSTOMER-SALE-ID             PICTURE XX                  DEFINE 01.CUSTOMER-SALE-ID       46 -    47 X
         03 00.CUSTOMER-CITY                PICTURE X(13)               DEFINE 00.CUSTOMER-CITY          48 -    60 X
         03 01.STATE                        PICTURE XX                  DEFINE 01.STATE                  61 -    62 X
         03 01.ZIP-CODE                     PICTURE X(5)                DEFINE 01.ZIP-CODE               63 -    67 X
         03 01.CREDIT-LIMIT                 PICTURE 9(6)V99             DEFINE 01.CREDIT-LIMIT           68 -    75 N2
         03 01.SALESMAN-ID                  PICTURE XX                  DEFINE 01.SALESMAN-ID            76 -    77 X
         03 01.CURRENT-BALANCE              PICTURE 9(5)V99             DEFINE 01.CURRENT-BALANCE        78 -    84 N2
      02 REPORT-02.                         XYZ SAMPLE REPORT 02
         03 00.CUSTOMER-NAME                PICTURE X(24)               DEFINE 00.CUSTOMER-NAME          22 -    45 X
         03 01.CUSTOMER-NUMBER              PICTURE XXX                 DEFINE 01.CUSTOMER-NUMBER        46 -    48 X
         03 00.CUSTOMER-CITY                PICTURE X(13)               DEFINE 00.CUSTOMER-CITY          49 -    61 X
         03 00.ST                           PICTURE X(14)               DEFINE 00.ST                     62 -    75 X
         03 01.ZIP-CODE                     PICTURE X(5)                DEFINE 01.ZIP-CODE               76 -    80 X
         03 01.CUSTOMER-SALE-ID             PICTURE XX                  DEFINE 01.CUSTOMER-SALE-ID       81 -    82 X
         03 01.CREDIT-LIMIT                 PICTURE 9(6)V99             DEFINE 01.CREDIT-LIMIT           83 -    90 N2
      02 REPORT-03.                         XYZ SAMPLE REPORT 03
         03 00.CUSTOMER-NAME                PICTURE X(24)               DEFINE 00.CUSTOMER-NAME          22 -    45 X
         03 01.CUSTOMER-NUMBER              PICTURE XXX                 DEFINE 01.CUSTOMER-NUMBER        46 -    48 X
         03 01.CREDIT-LIMIT                 PICTURE 9(6)V99             DEFINE 01.CREDIT-LIMIT           49 -    56 N2
         03 01.CURRENT-BALANCE              PICTURE 9(5)V99             DEFINE 01.CURRENT-BALANCE        57 -    63 N2
         03 00.UNUSED-CREDIT                PICTURE S9(7)V99     COMP-3 DEFINE 00.UNUSED-CREDIT          64 -    68 P2
         03 00.CUSTOMER-CITY                PICTURE X(13)               DEFINE 00.CUSTOMER-CITY          69 -    81 X
         03 00.ST                           PICTURE X(14)               DEFINE 00.ST                     82 -    95 X
         03 01.ZIP-CODE                     PICTURE X(5)                DEFINE 01.ZIP-CODE               96 -   100 X
      02 REPORT-04.                         XYZ SAMPLE REPORT 04
         03 00.CUSTOMER-NAME                PICTURE X(24)               DEFINE 00.CUSTOMER-NAME          22 -    45 X
         03 01.CUSTOMER-NUMBER              PICTURE XXX                 DEFINE 01.CUSTOMER-NUMBER        46 -    48 X
         03 00.CUSTOMER-CITY                PICTURE X(13)               DEFINE 00.CUSTOMER-CITY          49 -    61 X
         03 00.ST                           PICTURE X(14)               DEFINE 00.ST                     62 -    75 X
         03 01.ZIP-CODE                     PICTURE X(5)                DEFINE 01.ZIP-CODE               76 -    80 X
         03 01.CUSTOMER-SALE-ID             PICTURE XX                  DEFINE 01.CUSTOMER-SALE-ID       81 -    82 X
         03 01.CREDIT-LIMIT                 PICTURE 9(6)V99             DEFINE 01.CREDIT-LIMIT           83 -    90 N2
```

Following is a sample of the seventh page of the report.

```
 REPORT   01 FIELD NAME                  HDR1  HDR2  DTL   ACC   TOTAL       PICTURE            TAG   FROM   TO

 LINE   01   00.CUSTOMER-NAME               4          24              24                                   1    24
             01.CUSTOMER-SALE-ID            8     2     2               8                                  31    38
             00.CUSTOMER-CITY               4          13              13                                  46    58
             01.STATE                       5           2               5                                  66    70
             01.ZIP-CODE                    3     4     5               5                                  78    82
             01.CREDIT-LIMIT                6     5    11              11         $$$,$$9.99               90   100
             01.SALESMAN-ID                 8     8     2               8                                 108   115
             01.CURRENT-BALANCE             7     7    10              10          $$,$$9.99              123   132
```

Following is a sample of the eighth page of the report.

```
  REPORT   02 FIELD NAME                  HDR1  HDR2   DTL   ACC   TOTAL        PICTURE           TAG   FROM   TO

  LINE   01   00.CUSTOMER-NAME               4          24            24                                  1   24
              01.CUSTOMER-NUMBER             8     6     3             8                                 33   40
              00.CUSTOMER-CITY               4          13            13                                 49   61
              00.ST                          5     4    14            14                                 70   83
              01.ZIP-CODE                    3     4     5             5                                 92   96
              01.CUSTOMER-SALE-ID            8     2     2             8                                105  112
              01.CREDIT-LIMIT                6     5    11            11         $$$,$$9.99             122  132
```

Following is a sample of the ninth page of the report.

```
Date: mm/dd/ccyy    ********************************************************************************         Page:     9
                    *                                CA Datacom/DB                                  *
Time: hh.mm.ss      *                               General Utility                                 *      Release: 12
                    *                    Copyright © 2009 CA. All rights reserved.                  *               SPnn
                    ********************************************************************************
  REPORT   03 FIELD NAME                  HDR1  HDR2   DTL   ACC   TOTAL        PICTURE           TAG   FROM   TO
                                                                                                        ┌──┐
  LINE   01   00.CUSTOMER-NAME               4          24            24                                │20│   43
                                                                                                        └──┘
              01.CUSTOMER-NUMBER             8     6     3             8                                 64   71
              01.CREDIT-LIMIT                6     5    11            11         $$$,$$9.99              75   85
              01.CURRENT-BALANCE             7     7    10    12      12          $$,$$9.99              89  100
              00.UNUSED-CREDIT               6     6    11    13      13        ***,**9.99-       A     104  116
                                                                                                        ┌──┐
  LINE   02   00.CUSTOMER-CITY                          13            13                                │20│   32
                                                                                                        └──┘
              00.ST                                     14            14                                 36   49
              01.ZIP-CODE                                5             5                                 53   57
                                                                                                        ▲
                                                                                  Result of exact location printing
```

Following is a sample of the tenth page of the report.

```
Date: mm/dd/ccyy    ********************************************************************************         Page:     10
                    *                                CA Datacom/DB                                  *
Time: hh.mm.ss      *                               General Utility                                 *      Release: 12
                    *                    Copyright © 2009 CA. All rights reserved.                  *               SPnn
                    ********************************************************************************
  REPORT   04 FIELD NAME                  HDR1  HDR2  DTL   ACC   TOTAL       PICTURE           TAG   FROM   TO

  LINE   01   00.CUSTOMER-NAME               4          24            24                                  1   24
              01.CUSTOMER-NUMBER             8     6     3             8                                 33   40
              00.CUSTOMER-CITY               4          13            13                                 49   61
              00.ST                          5     4    14            14                                 70   83
              01.ZIP-CODE                    3     4     5             5                                 92   96
              01.CUSTOMER-SALE-ID            8     2     2             8                                105  112
              01.CREDIT-LIMIT                6     5    11            11         $$$,$$9.99             122  132
```

```
RUN DIAGNOSTICS
INDEX VIOLATIONS       00000000000
PROGRAM CHECKS         00000000000
PRIMARY I/P RECORDS    00000000016
HITS FOR REPORT   1  - 00000000016
HITS FOR REPORT   2  - 00000000016
HITS FOR REPORT   3  - 00000000016
HITS FOR REPORT   4  - 00000000016
TOTAL RECORDS SELECTED 00000000064
```

```
XYZ INC.
24 JUN 2010                                              XYZ SAMPLE REPORT 01                                    PAGE     1

-------------------------------------------------------------------------------------------------------------------------------
NAME                          SALESMAN       CITY                STATE       ZIP              CREDIT     SALESMAN       CURRENT
                              ID                                             CODE              LIMIT        ID          BALANCE
-------------------------------------------------------------------------------------------------------------------------------
HIGH-ROLLING INVESTMENT        7             DALLAS              TX          75260       $100,000.00      7          $54,000.00
SOUTHERN FRIED FOODS          10             MEMPHIS             TN          38101        $30,000.00     10          $36,500.00
LEGAL TOBACCO CO.             10             CHARLOTTE           NC          28228        $40,000.00     10           $5,400.00
SOUTHERN PINE INDUSTRIES       7             ATLANTA             GA          30304        $20,000.00      7               $0.00
BARONIAL OIL CO.               7             OKLAHOMA CITY       OK          73125        $90,000.00      7          $27,000.00
COSMOPOLITAN FASHIONS          1             NEW YORK            NY          10001        $60,000.00      1          $22,800.00
HEAVY METAL MACHINERY          2             CLEVELAND           OH          44101        $50,000.00      2          $48,900.00
AIRPORT SERVICES CORP.         8             NEWARK              NJ          07102        $40,000.00      8          $12,500.00
STOLID INSURANCE CORP.         8             HARTFORD            CT          06101        $80,000.00      8          $86,000.00
INLAND GRAIN TERMINALS         1             CHICAGO             IL          60607        $40,000.00      1           $4,000.00
STEEL CURTAIN STEEL INC.       2             PITTSBURGH          PN          15219        $30,000.00      2               $0.00
PERFECT BEARING CORP.          1             DETROIT             MI          48233        $20,000.00      1           $2,300.00
MOUNTAIN STATES MINING         5             DENVER              CO          80202        $50,000.00      5          $47,200.00
NORTHWEST PLYWOOD MILLS        5             SEATTLE             WA          98134        $20,000.00      5               $0.00
ORIENTAL TRADING CO.           5             SAN FRANCISCO       CA          94119        $60,000.00      5          $54,000.00
WEST COAST LIFESTYLES          5             LOS ANGELES         CA          90052        $10,000.00      5          $11,000.00
END OF REPORT
```

```
XYZ INC.
24 JUN 2010                                              XYZ SAMPLE REPORT 02                                    PAGE    1

-------------------------------------------------------------------------------------------------------------------------------
NAME                            CUSTOMER        CITY                 STATE                 ZIP        SALESMAN           CREDIT
                                NUMBER                               NAME                  CODE       ID                  LIMIT
-------------------------------------------------------------------------------------------------------------------------------
HIGH-ROLLING INVESTMENT         101             DALLAS               TEXAS                 75260       7            $100,000.00
SOUTHERN FRIED FOODS            102             MEMPHIS              TENNESSEE             38101      10             $30,000.00
LEGAL TOBACCO CO.               103             CHARLOTTE            NORTH CAROLINA        28228      10             $40,000.00
SOUTHERN PINE INDUSTRIES        104             ATLANTA              GEORGIA               30304       7             $20,000.00
BARONIAL OIL CO.                108             OKLAHOMA CITY        OKLAHOMA              73125       7             $90,000.00
COSMOPOLITAN FASHIONS           201             NEW YORK             NEW YORK              10001       1             $60,000.00
HEAVY METAL MACHINERY           203             CLEVELAND            OHIO                  44101       2             $50,000.00
AIRPORT SERVICES CORP.          207             NEWARK               NEW JERSEY            07102       8             $40,000.00
STOLID INSURANCE CORP.          210             HARTFORD             CONNECTICUT           06101       8             $80,000.00
INLAND GRAIN TERMINALS          211             CHICAGO              ILLINOIS              60607       1             $40,000.00
STEEL CURTAIN STEEL INC.        212             PITTSBURGH           PENNSYLVANIA          15219       2             $30,000.00
PERFECT BEARING CORP.           214             DETROIT              MICHIGAN              48233       1             $20,000.00
MOUNTAIN STATES MINING          303             DENVER               COLORADO              80202       5             $50,000.00
NORTHWEST PLYWOOD MILLS         306             SEATTLE              WASHINGTON            98134       5             $20,000.00
ORIENTAL TRADING CO.            308             SAN FRANCISCO        CALIFORNIA            94119       5             $60,000.00
WEST COAST LIFESTYLES           309             LOS ANGELES          CALIFORNIA            90052       5             $10,000.00
END OF REPORT
```

```
                                                             XYZ INC.
24 JUN 2010                                              XYZ SAMPLE REPORT 03                                    PAGE    1

------------------------------------------------------------------------------------------------------------------------------
                   NAME                                        CUSTOMER        CREDIT        CURRENT          UNUSED
                                                               NUMBER           LIMIT        BALANCE          CREDIT
------------------------------------------------------------------------------------------------------------------------------
                   ORIENTAL TRADING CO.                        308         $60,000.00     $54,000.00   ****6,000.00
                   SAN FRANCISCO   CALIFORNIA       94119
                   WEST COAST LIFESTYLES                       309         $10,000.00     $11,000.00
                   LOS ANGELES     CALIFORNIA       90052
                   MOUNTAIN STATES MINING                      303         $50,000.00     $47,200.00   ****2,800.00
                   DENVER          COLORADO         80202
                   STOLID INSURANCE CORP.                      210         $80,000.00     $86,000.00
                   HARTFORD        CONNECTICUT      06101
                   SOUTHERN PINE INDUSTRIES                    104         $20,000.00          $0.00   ***20,000.00
                   ATLANTA         GEORGIA          30304
                   INLAND GRAIN TERMINALS                      211         $40,000.00      $4,000.00   ***36,000.00
                   CHICAGO         ILLINOIS         60607
                   PERFECT BEARING CORP.                       214         $20,000.00      $2,300.00   ***17,700.00
                   DETROIT         MICHIGAN         48233
                   AIRPORT SERVICES CORP.                      207         $40,000.00     $12,500.00   ***27,200.00
                   NEWARK          NEW JERSEY       07102
                   COSMOPOLITAN FASHIONS                       201         $60,000.00     $22,800.00   ***37,200.00
                   NEW YORK        NEW YORK         10001
                   LEGAL TOBACCO CO.                           103         $40,000.00      $5,400.00   ***34,600.00
                   CHARLOTTE       NORTH CAROLINA   28228
                   HEAVY METAL MACHINERY                       203         $50,000.00     $48,900.00   ****1,100.00
                   CLEVELAND       OHIO             44101
                   BARONIAL OIL CO.                            108         $90,000.00     $27,000.00   ***63,000.00
                   OKLAHOMA CITY   OKLAHOMA         73125
                   STEEL CURTAIN STEEL INC.                    212         $30,000.00          $0.00   ***30,000.00
                   PITTSBURGH      PENNSYLVANIA     15219
                   SOUTHERN FRIED FOODS                        102         $30,000.00     $36,500.00
                   MEMPHIS         TENNESSEE        38101
                   HIGH-ROLLING INVESTMENT                     101        $100,000.00     $54,000.00   ***46,000.00
                   DALLAS          TEXAS            75260
                   NORTHWEST PLYWOOD MILLS                     306         $20,000.00          $0.00   ***20,000.00
                   SEATTLE         WASHINGTON       98134
                                                                                        ------------   -------------
GRAND TOTAL                                                                              $411,600.00 ****341,900.00
```

```
                                                              XYZ INC.
24 JUN 2010                                              XYZ SAMPLE REPORT 04                                    PAGE    1

-------------------------------------------------------------------------------------------------------------------------------
NAME                            CUSTOMER        CITY                 STATE                 ZIP        SALESMAN           CREDIT
                                NUMBER                               NAME                  CODE       ID                  LIMIT
-------------------------------------------------------------------------------------------------------------------------------
HIGH-ROLLING INVESTMENT         101             DALLAS               TEXAS                 75260       7            $100,000.00
SOUTHERN FRIED FOODS            102             MEMPHIS              TENNESSEE             38101      10             $30,000.00
LEGAL TOBACCO CO.               103             CHARLOTTE            NORTH CAROLINA        28228      10             $40,000.00
SOUTHERN PINE INDUSTRIES        104             ATLANTA              GEORGIA               30304       7             $20,000.00
BARONIAL OIL CO.                108             OKLAHOMA CITY        OKLAHOMA              73125       7             $90,000.00
COSMOPOLITAN FASHIONS           201             NEW YORK             NEW YORK              10001       1             $60,000.00
HEAVY METAL MACHINERY           203             CLEVELAND            OHIO                  44101       2             $50,000.00
AIRPORT SERVICES CORP.          207             NEWARK               NEW JERSEY            07102       8             $40,000.00
STOLID INSURANCE CORP.          210             HARTFORD             CONNECTICUT           06101       8             $80,000.00
INLAND GRAIN TERMINALS          211             CHICAGO              ILLINOIS              60607       1             $40,000.00
STEEL CURTAIN STEEL INC.        212             PITTSBURGH           PENNSYLVANIA          15219       2             $30,000.00
PERFECT BEARING CORP.           214             DETROIT              MICHIGAN              48233       1             $20,000.00
MOUNTAIN STATES MINING          303             DENVER               COLORADO              80202       5             $50,000.00
NORTHWEST PLYWOOD MILLS         306             SEATTLE              WASHINGTON            98134       5             $20,000.00
ORIENTAL TRADING CO.            308             SAN FRANCISCO        CALIFORNIA            94119       5             $60,000.00
WEST COAST LIFESTYLES           309             LOS ANGELES          CALIFORNIA            90052       5             $10,000.00
END OF REPORT
```

Secondary Mode

Secondary mode is well suited to the functions involved in report generation, but is more efficient than Primary mode. Each request is processed faster and the system uses minimum resources.

The system enters this mode of operation either *explicitly*, by specifying the OPTION SORT=NONE parameter in the Reporting Facility source program, or *implicitly*, by the Reporting Facility system where it is possible to establish from the source commands that the system is to generate a single report in the same sequence as the primary input file.

Two main processes are performed in this mode:

- Compilation/Code Generation
- Record Selection/Data Manipulation/Report Printing

Secondary Mode Example 1

```
CA                                         16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                   PAGE      1

   ┌────────────────────────────────────────────────────┐
   │ 1           OPTION  LIST ON MAP SORT=NONE          │ ◄  Enter Secondary mode explicitly
   └────────────────────────────────────────────────────┘

     2            USER    CA

    4           NOTE   ****************************************
    5           NOTE   *                                      *
    6           NOTE   *  THIS REPORTING FACILITY PROGRAM     *
    7           NOTE   *  PRODUCES A SINGLE REPORT SHOWING A  *
    8           NOTE   *  DETAIL LISTING OF THE TRANSACTION   *
    9           NOTE   *  INPUT.  THE INPUT IS ASSUMED TO BE  *
   10           NOTE   *  IN THE PROPER SEQUENCE, HENCE NO    *
   11           NOTE   *  SORT IS NECESSARY.                  *
   12           NOTE   *                                      *
   13           NOTE   ****************************************
   14           TRANSF:   INPUT CARD

  ┌─────────────────────────────────────────────────────────────────────────────┐
  │ 16           DEF    TRANSACTION-CODE      1-2  X                            │
  │ 17           DEF    ID-NUMBER             3-6  X                            │
  │ 18           DEF    AMOUNT               7-13  N2  'TRANSACTION' 'AMOUNT '  │
  │ 19           DEF    ACTIVITY-DATE       14-21  X  'DATE  OF'    'ACTIVITY'  │ ◄  Single input card file
  │ 20           DEF    CLEARED                22  X                            │
  │ 21           DEF    PAYEE               23-42  X                            │
  │ 22           DEF    REASON              44-75  X                            │
  └─────────────────────────────────────────────────────────────────────────────┘

  ┌─────────────────────────────────────────────────────┐
  │ 24           GET    TRANS                           │
  │                                                     │ ◄  Could be omitted for primary file
  │ 25           GOTO   EOJ WHEN TRANS EQ END-OF-FILE   │
  └─────────────────────────────────────────────────────┘

  ┌─────────────────────────────────────────────────────┐
  │ 26           DECODE TRANSACTION-CODE INTO TYPE      │ ◄  Date translation
  └─────────────────────────────────────────────────────┘

   27                          'BF' = 'BALANCE BROUGHT FORWARD'
   28                          'CK' = 'CHECK WRITTEN'
   29                          'DP' = 'DEPOSIT'
   30                          'NS' = 'NON-SUFFICIENT FUNDS'
   31                          'SC' = 'SERVICE CHARGE'
   32                           ELSE  'UNKNOWN TRANSACTION TYPE'
   33                                 ' ' 'TYPE OF TRANSACTION'

  ┌─────────────────────────────────────────────────────┐
  │ 35           REPORT 'TRANSACTION DETAIL LISTING'    │
  │ 36           SELECT ALL                             │ ◄ Single report definition group
  │ 37           PRINT QSEQ TYPE ACTIVITY-DATE PAYEE    │ (Highlighted words in lines 37 and 38 are
  │ 38                  REASON AMOUNT CURRENT-TIME      │ predefined field names/reserved words.)
  │ 39   END                                            │
  └─────────────────────────────────────────────────────┘


         COMPILE PHASE COMPLETED - NO ERRORS FOUND
                                   NO WARNINGS ISSUED
         START 13:58:00 - STOP 13:58:00
```

```
CA                                       16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                PAGE      2
COMPILE DIAGNOSTICS

I/P  00001 X 00080 = 00080   TRANS

GSA  00000001059 (00000000070 ENTRIES)

     GSA 1 = 00000000001 NUMERIC CONSTANT ...... CON
     GSA 2 = 00000000021 QUOTED LITERAL ........ LIT
     GSA 3 = 00000000001 INTERMEDIATE RESULT ... RES
     GSA 4 = 00000000045 UNQUOTED LITERAL ...... INT
     GSA 5 = 00000000000 ACCUMULATOR ........... ACC
     GSA 6 = 00000000001 VALUE ................. VAL
     GSA 7 = 00000000000 STRING ................ STR
     GSA 8 = 00000000001 HEX LITERAL ........... HEX

FNT  00000000646 (00000000038 ENTRIES)

     FNT 0 = 00000000005 LABELS ................  01
     FNT 1 = 00000000032 SCALAR VARIABLES ......  01
     FNT 2 = 00000000001 FILE DESC. BLOCKS .....  02
     FNT 3 = 00000000000 ARRAYS ................  03

FST  00000003416 (00000000122 ENTRIES)

     FST 0 = 00000000004 INTERNAL ONLY ......... -02
     FST 1 = 00000000002 USER/REPORT ...........  01
     FST 2 = 00000000007 INPUT AREA VARIABLE ...  02
     FST 3 = 00000000076 GSA VARIABLE ..........  03
     FST 4 = 00000000002 DATA MOVEMENT .........  04
     FST 5 = 00000000006 DECODE ................  05
     FST 6 = 00000000001 SEQUENCE AND CONTROL ..  06
     FST 7 = 00000000005 CONDITION TEST ........  07
     FST 8 = 00000000002 SELECTION .............  08
     FST 9 = 00000000008 PRINT LAYOUT ..........  09
     FST A = 00000000007 HIT RECORD EXTENSION ..  0A
     FST B = 00000000002 GET RECORD ............  0B
     FST C = 00000000000 CONTROL BREAK CALC ....  0C
     FST D = 00000000000 UNDEFINED .............  0D
     FST E = 00000000000 ARRAY DEF/REL. INDEX ..  0E
     FST F = 00000000000 DUMMY .................  0F

GETMAIN REQUIREMENT FOR GSA/FNT/FST ... APPROX. 006K
```

```
CA                                       16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                PAGE      3
     ------ FIELD NAME ------ ORIGIN  QUAL  HIT ARY  ---------------------------------- REPORTS -----------------------------

     ABORT:                   *LABEL*
     ACTIVITY-DATE            TRANS   (01)   X         1
     AMOUNT                   TRANS   (01)   X         1
     BLANK                    * GSA * (00)
     CLEARED                  TRANS   (01)
     CURRDATE                 * GSA * (00)
     CURRENT-DATE             * GSA * (00)
     CURRENT-TIME             * GSA * (00)   X         1
     CURRTIME                 * GSA * (00)
     DD                       * GSA * (00)
     END-OF-FILE              * GSA * (00)
     EOJ:                     *LABEL*
     HR                       * GSA * (00)
     ID-NUMBER                TRANS   (01)
     MM                       * GSA * (00)
     MN                       * GSA * (00)
     NO-RECORD-FOUND          * GSA * (00)
     PAYEE                    TRANS   (01)   X         1
     QSEQ                     * GSA * (00)   X         1
     REASON                   TRANS   (01)   X         1
     RECORD-FOUND             * GSA * (00)
     SPACE                    * GSA * (00)
     SPACES                   * GSA * (00)
     SS                       * GSA * (00)
     START:                   *LABEL*
     TAG                      * GSA * (00)
     TEST:                    *LABEL*
     TRANS                    * GSA * (00)
     TRANS:                   *LABEL*
     TRANSACTION-CODE         TRANS   (01)
     TYPE                     * GSA * (00)   X         1
     XD                       * GSA * (00)
     XM                       * GSA * (00)
     XY                       * GSA * (00)
     YY                       * GSA * (00)
     ZERO                     * GSA * (00)
     ZEROS                    * GSA * (00)
```

```
CA                                       16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                PAGE      4
                                                  (3350   ) ......H I T   F I L E   L A Y O U T......

 FD  DROUT                                  DROUT:  FILE DISK SEQUENTIAL VARIABLE RECORD=        119 BLOCK=       4604
       BLOCK CONTAINS        4604 CHARACTERS
       RECORD CONTAINS         119 CHARACTERS
       RECORDING MODE V
       DATA RECORD IS HITFILE.

 01  HITFILE.
       02 RDW                           PICTURE X(4)                 DEFINE RDW                             1 -    4 X
       02 TAG-CHARACTER                 PICTURE X                    DEFINE TAG-CHARACTER                   5 -    5 X
       02 SORT-KEY.
          03 REPORT-NUMBER              PICTURE B                    DEFINE REPORT-NUMBER                   6 -    6 B
          03 USER-SORT-KEY              PICTURE X(9)                 DEFINE USER-SORT-KEY                   7 -   15 X
       02 REPORT-01.                    TRANSACTION DETAIL LISTING
          03 00.QSEQ                    PICTURE S9(9)        COMP-3  DEFINE 00.QSEQ                        16 -   20 P
          03 00.TYPE                    PICTURE X(24)                DEFINE 00.TYPE                        21 -   44 X
          03 01.ACTIVITY-DATE           PICTURE X(8)                 DEFINE 01.ACTIVITY-DATE               45 -   52 X
          03 01.PAYEE                   PICTURE X(20)                DEFINE 01.PAYEE                       53 -   72 X
          03 01.REASON                  PICTURE X(32)                DEFINE 01.REASON                      73 -  104 X
          03 01.AMOUNT                  PICTURE 9(5)V99              DEFINE 01.AMOUNT                     105 -  111 N2
          03 00.CURRENT-TIME            PICTURE X(8)                 DEFINE 00.CURRENT-TIME               112 -  119 X
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      5
  REPORT   01 FIELD NAME                  HDR1  HDR2  DTL   ACC   TOTAL       PICTURE            TAG   FROM   TO

  LINE   01   00.QSEQ                        4          10            10         999999999-               1    10
              00.TYPE                             19    24            24                                13    36
              01.ACTIVITY-DATE               8     8     8             8                                39    46
              01.PAYEE                       5          20            20                                49    68
              01.REASON                      6          32            32                                72   103
              01.AMOUNT                     11     8     9            11          99999.99-            107   117
              00.CURRENT-TIME               12           8            12                               121   132
```

```
CA
24 JUN 2010                                          TRANSACTION DETAIL LISTING                                  PAGE     1

--------------------------------------------------------------------------------------------------------------------------------
   QSEQ                            DATE  OF  PAYEE                  REASON                                 TRANSACTION  CURRENT-TIME
         TYPE OF TRANSACTION       ACTIVITY                                                                   AMOUNT
--------------------------------------------------------------------------------------------------------------------------------
     1   BALANCE BROUGHT FORWARD   01/01/10/* *                       BALANCE BROUGHT FORWARD    000       109.12    13:58:03
     2   DEPOSIT                   01/01/10/* *                                                  000       500.00    13:58:03
     3   CHECK WRITTEN             01/01/10  ABC NATIONAL BANK      CASH                         000        50.00    13:58:03
     4   CHECK WRITTEN             01/01/10  ACME NATIONAL LIFE     JANUARY INS. PREMIUM         000        37.02    13:58:03
     5   CHECK WRITTEN             01/01/10  SOUTHWEST TITLE CO.    JANUARY MORTGAGE PAYMENT     000       433.00    13:58:03
     6   CHECK WRITTEN             01/03/10  TEXAS POWER & LIGHT    UTILITY BILL FOR DEC'07      000        62.15    13:58:03
     7   CHECK WRITTEN             01/10/10  FOOD MART              GROCERIES                    000        22.87    13:58:03
     8   DEPOSIT                   01/15/10                                                      000       824.07    13:58:03
     9   CHECK WRITTEN             01/15/10  RICHBURG INS. AGENCY   CAR INSURANCE                000       412.89    13:58:03
    10   CHECK WRITTEN             01/22/10  ABC NATIONAL BANK      CASH                         000       100.00    13:58:03
    11   CHECK WRITTEN             01/22/10  SOUTHWESTERN BELL      PHONE BILL                   000        22.03    13:58:03
    12   CHECK WRITTEN             01/23/10  ACME HAIR DESIGN       HAIRCUT                      000        17.00    13:58:03
    13   CHECK WRITTEN             01/25/10  SEARS ROEBUCK          DRYER                        000       213.56    13:58:03
    14   CHECK WRITTEN             01/27/10  ALL-NITE BOWLING       BOWLING BAG                  000        16.75    13:58:03
    15   CHECK WRITTEN             01/30/10  ACME FINANCE           LOAN PAYMENT 1               000        89.98    13:58:03
    16   NON-SUFFICIENT FUNDS      01/31/10                                                      000         6.00    13:58:03
    17   SERVICE CHARGE            01/31/10                                                      000         4.00    13:58:03
    18   DEPOSIT                   02/01/10                                                      000       922.00    13:58:03
    19   CHECK WRITTEN             02/01/10  JOHN DOE               RETURN OF CASH ADVANCE       000       150.00    13:58:03
    20   CHECK WRITTEN             02/03/10  SOUTHWEST TITLE CO.    FEBRUARY MORTGAGE PAYMENT    000       433.00    13:58:03
    21   CHECK WRITTEN             02/05/10  TEXAS POWER & LIGHT    UTILITY BILL FOR JAN'08      000        59.11    13:58:03
    22   CHECK WRITTEN             02/09/10  ACME NATIONAL LIFE     FEBRUARY INS. PREMIUM        000        37.02    13:58:03
    23   CHECK WRITTEN             02/11/10  FOOD MART              GROCERIES                    000        68.29    13:58:03
    24   CHECK WRITTEN             02/11/10  ABC NATIONAL BANK      CASH                         000        25.00    13:58:03
    25   DEPOSIT                   02/17/10                         2007 INCOME TAX REFUND       000      1538.00    13:58:03
    26   CHECK WRITTEN             02/20/10  ABC NATIONAL BANK      CASH                         000        50.00    13:58:03
    27   CHECK WRITTEN             02/21/10  ACME FINANCE           LOAN PAYMENTS 2 AND 3        000       179.96    13:58:03
    28   CHECK WRITTEN             02/27/10  SOUTHWEST TITLE CO.    MARCH/APRIL MORTGAGE PAYMENT 000       866.00    13:58:03
    29   CHECK WRITTEN             02/28/10  COMPANY XYZ            SUIT                         000       120.42    13:58:03
    30   CHECK WRITTEN             03/02/10  TEXAS POWER & LIGHT    UTILITY BILL FOR FEB'08      000        62.11    13:58:03
    31   CHECK WRITTEN             03/02/10  ABC NATIONAL BANK      CASH                         000        25.00    13:58:03
    32   SERVICE CHARGE            03/03/10                                                      000         4.00    13:58:03
    33   CHECK WRITTEN             03/09/10  FOOD MART              GROCERIES                    000        72.80    13:58:03
    34   CHECK WRITTEN             03/12/10  SOUTHERN AIRWAYS       TICKET TO LUBBOCK            000       227.89    13:58:03
    35   DEPOSIT                   03/15/10                                                      000       922.00    13:58:03
    36   CHECK WRITTEN             03/16/10  ABC NATIONAL BANK      CASH                         000       250.00    13:58:03
    37   CHECK WRITTEN             03/27/10  ACME NATIONAL LIFE     MARCH/APRIL INS. PREMIUMS    000        74.04    13:58:03
    38   CHECK WRITTEN             03/30/10  DISCOUNT AUTO STORE    BATTERY                      000        35.70    13:58:03
    39   DEPOSIT                   04/01/10                                                      000       922.00    13:58:03
    40   CHECK WRITTEN             04/02/10  V.O.I.D.                                            000         0.00    13:58:03
    41   CHECK WRITTEN             04/02/10  TEXAS POWER & LIGHT    UTILITY BILL FOR MAR'08      000        47.18    13:58:03
    42   CHECK WRITTEN             04/02/10  FOOD MART              GROCERIES                    000        25.90    13:58:03
    43   CHECK WRITTEN             04/02/10  ABC NATIONAL BANK      CASH                         000       100.00    13:58:03
    44   CHECK WRITTEN             04/10/10  ACME FINANCE           LOAN PAYMENT 4               000        89.98    13:58:03
    45   CHECK WRITTEN             04/10/10  SOUTHWESTERN BELL      PHONE BILLS                  000        79.82    13:58:03
    46   SERVICE CHARGE            04/10/10                                                      000         4.00    13:58:03
    47   CHECK WRITTEN             04/10/10  ABC NATIONAL BANK      CASH                         000        50.00    13:58:03
    48   CHECK WRITTEN             04/11/10  DISCOUNT AUTO STORE    TUNE-UP KIT                  000        21.60    13:58:03
    49   CHECK WRITTEN             04/15/10  BERNY'S CHEVROLET      DOWN PAYMENT ON CORVETTE     000      1000.00    13:58:03
    50   DEPOSIT                   04/15/10                         PARTIAL SAVINGS WITHDRAWAL   000      2922.00    13:58:03
    51   CHECK WRITTEN             04/15/10  ABC NATIONAL BANK      CASH                         000        75.00    13:58:03
```

```
CA
24 JUN 2010                                              TRANSACTION DETAIL LISTING                                   PAGE     2

-----------------------------------------------------------------------------------------------------------------------------------
   QSEQ                              DATE  OF  PAYEE                  REASON                                TRANSACTION  CURRENT-TIME
         TYPE OF TRANSACTION         ACTIVITY                                                                  AMOUNT
-----------------------------------------------------------------------------------------------------------------------------------
    52   CHECK WRITTEN               04/20/10  CITY OF DALLAS         PARKING TICKET               000           5.00    13:58:03
    53   CHECK WRITTEN               04/20/10  ALL-NITE BOWLING       ENTERTAINMENT                000          11.55    13:58:03
    54   CHECK WRITTEN               04/22/10  MODERNAGE TV           REPAIRS                      000          75.22    13:58:03
    55   CHECK WRITTEN               04/25/10  BERNY'S CHEVROLET      CAR PAYMENT 1                000         179.00    13:58:03
    56   CHECK WRITTEN               04/26/10  ABC NATIONAL BANK      CASH                         000          25.00    13:58:03
    57   CHECK WRITTEN               04/26/10  ACME FINANCE           LOAN PAYMENT 5               000          89.98    13:58:03
    58   CHECK WRITTEN               04/29/10  SOUTHWEST TITLE CO.    MAY MORTGAGE PAYMENT         000         433.00    13:58:03
    59   DEPOSIT                     05/01/10                                                      000         922.00    13:58:03
    60   CHECK WRITTEN               05/01/10  ABC NATIONAL BANK      CASH                         000          50.00    13:58:03
    61   CHECK WRITTEN               05/01/10  ACME NATIONAL LIFE     MAY INS. PREMIUM             000          37.02    13:58:03
    62   CHECK WRITTEN               05/01/10  TEXAS POWER & LIGHT    UTILITY BILL FOR APR'08      000          55.98    13:58:03
    63   CHECK WRITTEN               05/01/10  BERNY'S CHEVROLET      CAR PAYMENT 2                000         179.00    13:58:03
    64   CHECK WRITTEN               05/02/10  COMPANY XYZ            BRIEFCASE                    000          47.56    13:58:03
    65   SERVICE CHARGE              05/02/10                                                      000           4.00    13:58:03
    66   CHECK WRITTEN               05/05/10  ALL-NITE BOWLING       ENTERTAINMENT                000          21.60    13:58:03
    67   CHECK WRITTEN               05/06/10  ABC NATIONAL BANK      CASH                         000          50.00    13:58:03
    68   CHECK WRITTEN               05/10/10  SEARS ROEBUCK          WASHER                       000         256.75    13:58:03
    69   CHECK WRITTEN               05/11/10  FOOD MART              GROCERIES                    000          53.89    13:58:03
    70   CHECK WRITTEN               05/15/10  DISCOUNT AUTO STORE    MISC. AUTO PARTS             000          42.04    13:58:03
    71   DEPOSIT                     05/15/10                                                      000         962.00    13:58:03
    72   CHECK WRITTEN               05/15/10  BERNY'S CHEVROLET      CAR PAYMENT 3                000         179.00    13:58:03
    73   CHECK WRITTEN               05/15/10  H & L FLORISTS         ROSES                        000          44.01    13:58:03
    74   CHECK WRITTEN               05/15/10  COMPZNY XYZ            MISC.                        000          27.89    13:58:03
    75   CHECK WRITTEN               05/16/10  ABC NATIONAL BANK      CASH                         000          50.00    13:58:03
    76   CHECK WRITTEN               05/17/10  MODERNAGE TV           TV                           000         272.50    13:58:03
    77   CHECK WRITTEN               05/17/10  RX DRUGS               PRESCRIPTION                 000          14.80    13:58:03
    78   CHECK WRITTEN               05/17/10  ACME HAIR DESIGN       HAIRCUT                      000          17.00    13:58:03
    79   CHECK WRITTEN               05/18/10  SOUTHWESTERN BELL      PHONE BILL                   000          55.39    13:58:03
    80   CHECK WRITTEN               05/18/10  ALL-NITE BOWLING       ENTERTAINMENT                000          22.50    13:58:03
    81   CHECK WRITTEN               05/18/10  JOHN DOE               RETURN CASH ADVANCE          000          01.00    13:58:03
    82   CHECK WRITTEN               05/18/10  ABC NATIONAL BANK      CASH                         000          25.00    13:58:03
    83   CHECK WRITTEN               05/20/10  ACME NATIONAL LIFE     JUNE INS. PREMIUM            000          37.02    13:58:03
    84   CHECK WRITTEN               05/21/10  SOUTHWEST TITLE CO.    JUNE MORTGAGE PAYMENT        000         433.00    13:58:03
    85   CHECK WRITTEN               05/21/10  ABC NATIONAL BANK      CASH                         000          25.00    13:58:03
    86   DEPOSIT                     06/01/10                                                      000         962.00    13:58:03
    87   CHECK WRITTEN               06/01/10  TEXAS POWER & LIGHT    UTILITY BILL FOR MAY'08      000          53.21    13:58:03
    88   CHECK WRITTEN               06/02/10  ABC NATIONAL BANK      CASH                         000          50.00    13:58:03
    89   CHECK WRITTEN               06/02/10  FOOD MART              GROCERIES                    000          82.40    13:58:03
    90   CHECK WRITTEN               06/02/10  ALL-NITE BOWLING       ENTERTAINMENT                000          11.00    13:58:03
    91   CHECK WRITTEN               06/05/10  V.O.I.D.                                            000           0.00    13:58:03
    92   CHECK WRITTEN               06/05/10  ABC NATIONAL BANK      CASH                         000         150.00    13:58:03
    93   DEPOSIT                     06/15/10                                                      000         962.00    13:58:03
END OF REPORT
```

```
RUN DIAGNOSTICS
INDEX VIOLATIONS       00000000000
PROGRAM CHECKS         00000000000
HITS FOR REPORT   1  - 00000000093
TOTAL RECORDS SELECTED 00000000093
```

Secondary Mode Example 2

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                  PAGE      1
     1            OPTION  LIST ON MAP SORT=NONE
       2            USER    CA

     4            NOTE    **********************************************
     5            NOTE    *                                            *
     6            NOTE    *     A C T I V I T Y    R E G I S T E R     *
     7            NOTE    *                                            *
     8            NOTE    *  THIS REPORTING FACILITY PROGRAM PRODUCES  *
     9            NOTE    *  A SINGLE REPORT, SHOWING A DETAIL LIST    *
    10            NOTE    *  OF ALL ACTIVITY.                          *
    11            NOTE    *                                            *
    12            NOTE    **********************************************
  ┌──────────────────────────────────────────────────────────────────────────────┐
  | 14   TRANS:   FILE    CARD                                                   |
  | 15            DEF     TRANSACTION-CODE    1 TO  2  X                         |
  | 16            DEF     ID-NUMBER           3 TO  6  X                         |
  |                                                         Alternate Headings   |
  | 17            DEF     AMOUNT              7 TO 13  N2    ▼              ▼    |
  | 18            DEF     ACTIVITY-DATE      14 TO 21  X 'DATE  OF'   'ACTIVITY' |  ◄  Input data file fields
  | 19            DEF     CLEARED                  22  X                         |
  | 20            DEF     PAYEE              23 TO 42  X                         |
  | 21            DEF     REASON             44 TO 75  X                         |
  └──────────────────────────────────────────────────────────────────────────────┘

  ┌──────────────────────────────────────────────────────────────────────────────┐
  |             ┌─────────────────────────────────────────────────────────┐      |
  | 23          | DEF     PRINTTYPE(9)=' '          '  TYPE /' '  STATUS' |  ◄  Field redefinition
  | 24          | DEF     PRTCODE=PRINTTYPE       1-2 X                   |      |
  | 25          | DEF     PRTIDNUMBER=PRINTTYPE   4-7 X                   |      |
  | 26          | DEF     PRTSTAT=PRINTTYPE         9 X                   |      |
  |             └─────────────────────────────────────────────────────────┘      |  ◄  GSA fields
  | 27            DEFINE  RUNNING-BALANCE(9.2)=0 'RUNNING  ' 'BALANCE  '         |
  | 28                                          PIC '$$$,$$$,$$9.99-'           |
  | 29            DEFINE  NEGATIVE(5.2) EQ 0 ' ' '-(MINUS)'   PIC '$$,$$9.99-'   |
  | 30            DEFINE  POSITIVE(5.2) EQ 0 ' ' '┘(PLUS)'    PIC '$$,$$9.99-'   |
  └──────────────────────────────────────────────────────────────────────────────┘
    32            SET     PRTCODE EQUAL TO TRANSACTION-CODE                        Notice that automatic read
    33            SET     PRTIDNUMBER EQUAL TO ID-NUMBER                           of primary file is assumed.
    34            MOVE    CLEARED TO PRTSTAT
    35            GO TO   PLUS WHEN TRANSACTION-CODE EQ 'BF'
    36            GO TO   PLUS WHEN TRANSACTION-CODE EQ 'DP'
    37            MOVE    AMOUNT TO NEGATIVE
    38            REDUCE  RUNNING-BALANCE BY AMOUNT
    39            GOTO    TEST
    40   PLUS:    CONTINUE
    41            MOVE    AMOUNT TO POSITIVE
    42            ADD     AMOUNT TO RUNNING-BALANCE

    44            REPORT  'ACTIVITY REGISTER'
  ┌──────────────────────────────────────────────────────────────────┐
  | 45            SELECT  'A' WHEN TRANSACTION-CODE EQ 'BF'          |
  | 46            SELECT  'A' WHEN TRANSACTION-CODE EQ 'DP'          |  ◄  Conditional record selection
  | 47            SELECT  'B' ALL                                    |
  └──────────────────────────────────────────────────────────────────┘
  ┌──────────────────────────────────────────────────────────────────┐
  | 48            PRINT   PRINTTYPE ACTIVITY-DATE PAYEE REASON       |  ◄  Conditional record printing
  | 49                    (A;POSITIVE) (B;NEGATIVE) RUNNING-BALANCE  |     with field accumulation
  └──────────────────────────────────────────────────────────────────┘

    50   END
         COMPILE PHASE COMPLETED - NO ERRORS FOUND
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      2
                                   NO WARNINGS ISSUED
         START 13:58:59 - STOP 13:59:00
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      3
COMPILE DIAGNOSTICS

I/P  00001 X 00080 = 00080   TRANS

GSA  00000001071 (00000000084 ENTRIES)

     GSA 1 = 00000000001 NUMERIC CONSTANT ...... CON
     GSA 2 = 00000000024 QUOTED LITERAL ........ LIT
     GSA 3 = 00000000001 INTERMEDIATE RESULT ... RES
     GSA 4 = 00000000051 UNQUOTED LITERAL ...... INT
     GSA 5 = 00000000002 ACCUMULATOR ........... ACC
     GSA 6 = 00000000004 VALUE ................. VAL
     GSA 7 = 00000000000 STRING ................ STR
     GSA 8 = 00000000001 HEX LITERAL ........... HEX

FNT  00000000782 (00000000045 ENTRIES)

     FNT 0 = 00000000006 LABELS ...............  01
     FNT 1 = 00000000038 SCALAR VARIABLES ......  01
     FNT 2 = 00000000001 FILE DESC. BLOCKS .....  02
     FNT 3 = 00000000000 ARRAYS ...............  03

FST  00000003892 (00000000139 ENTRIES)

     FST 0 = 00000000004 INTERNAL ONLY ......... -02
     FST 1 = 00000000002 USER/REPORT ...........  01
     FST 2 = 00000000007 INPUT AREA VARIABLE ...  02
     FST 3 = 00000000090 GSA VARIABLE ..........  03
     FST 4 = 00000000009 DATA MOVEMENT .........  04
     FST 5 = 00000000000 DECODE ................  05
     FST 6 = 00000000001 SEQUENCE AND CONTROL ..  06
     FST 7 = 00000000006 CONDITION TEST ........  07
     FST 8 = 00000000004 SELECTION .............  08
     FST 9 = 00000000008 PRINT LAYOUT ..........  09
     FST A = 00000000007 HIT RECORD EXTENSION ..  0A
     FST B = 00000000001 GET RECORD ............  0B
     FST C = 00000000000 CONTROL BREAK CALC ....  0C
     FST D = 00000000000 UNDEFINED .............  0D
     FST E = 00000000000 ARRAY DEF/REL. INDEX ..  0E
     FST F = 00000000000 DUMMY .................  0F

GETMAIN REQUIREMENT FOR GSA/FNT/FST ... APPROX. 006K
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      4
     ------ FIELD NAME ------ ORIGIN  QUAL  HIT ARY  ----------------------------------- REPORTS ------------------------------

     ABORT:                   *LABEL*
     ACTIVITY-DATE            TRANS   (01)   X         1
     AMOUNT                   TRANS   (01)
     BLANK                    * GSA * (00)
     CLEARED                  TRANS   (01)
     CURRDATE                 * GSA * (00)
     CURRENT-DATE             * GSA * (00)
     CURRENT-TIME             * GSA * (00)
     CURRTIME                 * GSA * (00)
     DD                       * GSA * (00)
     END-OF-FILE              * GSA * (00)
     EOJ:                     *LABEL*
     HR                       * GSA * (00)
     ID-NUMBER                TRANS   (01)
     MM                       * GSA * (00)
     MN                       * GSA * (00)
     NEGATIVE                 * GSA * (00)   X         1
     NO-RECORD-FOUND          * GSA * (00)
     PAYEE                    TRANS   (01)   X         1
     PLUS:                    *LABEL*
     POSITIVE                 * GSA * (00)   X         1
     PRINTTYPE                * GSA * (00)   X         1
     PRTCODE                  * GSA * (00)
     PRTIDNUMBER              * GSA * (00)
     PRTSTAT                  * GSA * (00)
     QSEQ                     * GSA * (00)
     REASON                   TRANS   (01)   X         1
     RECORD-FOUND             * GSA * (00)
     RUNNING-BALANCE          * GSA * (00)   X         1
     SPACE                    * GSA * (00)
     SPACES                   * GSA * (00)
     SS                       * GSA * (00)
     START:                   *LABEL*
     TAG                      * GSA * (00)
     TEST:                    *LABEL*
     TRANS                    * GSA * (00)
     TRANS:                   *LABEL*
     TRANSACTION-CODE         TRANS   (01)
     XD                       * GSA * (00)
     XM                       * GSA * (00)
     XY                       * GSA * (00)
     YY                       * GSA * (00)
     ZERO                     * GSA * (00)
     ZEROS                    * GSA * (00)
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                  PAGE      5
                                                   (3350   ) ......H I T   F I L E   L A Y O U T......

 FD  DROUT                                   DROUT:  FILE DISK SEQUENTIAL VARIABLE RECORD=          98 BLOCK=        4610
       BLOCK CONTAINS        4610 CHARACTERS
       RECORD CONTAINS          98 CHARACTERS
       RECORDING MODE V
       DATA RECORD IS HITFILE.

 01  HITFILE.
       02 RDW                             PICTURE X(4)                 DEFINE RDW                             1 -    4 X
       02 TAG-CHARACTER                   PICTURE X                    DEFINE TAG-CHARACTER                   5 -    5 X
       02 SORT-KEY.
          03 REPORT-NUMBER                PICTURE B                    DEFINE REPORT-NUMBER                   6 -    6 B
          03 USER-SORT-KEY                PICTURE X(9)                 DEFINE USER-SORT-KEY                   7 -   15 X

    ┌─────────────────────────────────────────────────────────────────────────────────────────────────────────────────────┐
    │  02 REPORT-01.                       ACTIVITY REGISTER                                                               │
    │     03 00.PRINTTYPE                 PICTURE X(9).                DEFINE 00.PRINTTYPE                   16 -   24 X  │
    │     03 01.ACTIVITY-DATE             PICTURE X(8)                 DEFINE 01.ACTIVITY-DATE               25 -   32 X  │
    │     03 01.PAYEE                     PICTURE X(20)                DEFINE 01.PAYEE                       33 -   52 X  │
    │     03 01.REASON                    PICTURE X(32)                DEFINE 01.REASON                      53 -   84 X  │
    │     03 00.POSITIVE                  PICTURE S9(5)V99     COMP-3  DEFINE 00.POSITIVE                    85 -   88 P2 │
    │     03 00.NEGATIVE                  PICTURE S9(5)V99     COMP-3  DEFINE 00.NEGATIVE                    89 -   92 P2 │
    │     03 00.RUNNING-BALANCE           PICTURE S9(9)V99     COMP-3  DEFINE 00.RUNNING-BALANCE             93 -   98 P2 │
    └─────────────────────────────────────────────────────────────────────────────────────────────────────────────────────┘

             ▲
      Single report
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      6
  REPORT   01 FIELD NAME                  HDR1  HDR2  DTL   ACC   TOTAL       PICTURE            TAG   FROM   TO

  LINE   01   00.PRINTTYPE                   8     8     9             9                                   1     9
              01.ACTIVITY-DATE               8     8     8             8                                  13    20
              01.PAYEE                       5          20            20                                  24    43
              01.REASON                      6          32            32                                  47    78
                                                                                                ┌───┐
              00.POSITIVE                          7    11    13      13         $$,$$9.99-     │ A │     83    95
              00.NEGATIVE                          8    11    13      13         $$,$$9.99-     │ B │    100   112
                                                                                                └───┘
              00.RUNNING-BALANCE             9     9    16            16    $$$,$$$,$$9.99-             117   132
                                                                                                  ▲
                                                                                     Tag characters (see
                                                                                     SELECT and PRINT commands)
```

```
24 JUN 2010                                                ACTIVITY REGISTER      Conditional Printing          PAGE
-------------------------------------------------------------------------------------------------------------------------------
  TYPE /    DATE  OF   PAYEE                  REASON                                       ▼             ▼         RUNNING
  STATUS    ACTIVITY                                                                +(PLUS)       -(MINUS)          BALANCE
-------------------------------------------------------------------------------------------------------------------------------
BF          01/01/10                          BALANCE BROUGHT FORWARD      000     $109.12                        $109.12  09.12
DP          01/01/10                                                       000     $500.00                        $609.12  09.12
CK 1200 X   01/01/10   ABC NATIONAL BANK      CASH                         000                      $50.00        $559.12  59.12
CK 1201 X   01/01/10   ACME NATIONAL LIFE     JANUARY INS. PREMIUM         000                      $37.02        $522.10  22.10
CK 1202 X   01/01/10   SOUTHWEST TITLE CO.    JANUARY MORTGAGE PAYMENT     000                     $433.00         $89.10  89.10
CK 1203 X   01/03/10   TEXAS POWER & LIGHT    UTILITY BILL FOR DEC'07      000                      $62.15         $26.95  26.95
CK 1204 X   01/10/10   FOOD MART              GROCERIES                    000                      $22.87          $4.08  $4.08
DP          01/15/10                                                       000     $824.07                        $828.15  28.15
CK 1205 X   01/15/10   RICHBURG INS. AGENCY   CAR INSURANCE                000                     $412.89        $415.26  15.26
CK 1206 X   01/22/10   ABC NATIONAL BANK      CASH                         000                     $100.00        $315.26  15.26
CK 1207 X   01/22/10   SOUTHWESTERN BELL      PHONE BILL                   000                      $22.03        $293.23  93.23
CK 1208 X   01/23/10   ACME HAIR DESIGN       HAIRCUT                      000                      $17.00        $276.23  76.23
CK 1209 X   01/25/10   SEARS ROEBUCK          DRYER                        000                     $213.56         $62.67  62.67
CK 1210 X   01/27/10   ALL-NITE BOWLING       BOWLING BAG                  000                      $16.75         $45.92  45.92
CK 1211 X   01/30/10   ACME FINANCE           LOAN PAYMENT 1               000                      $89.98         $44.06- 44.06-
NS          01/31/10                                                       000                       $6.00         $50.06- 50.06-
SC          01/31/10                                                       000                       $4.00         $54.06- 54.06-
DP          02/01/10                                                       000     $922.00                        $867.94  67.94
CK 1212 X   02/01/10   JOHN DOE               RETURN OF CASH ADVANCE       000                     $150.00        $717.94  17.94
CK 1213 X   02/03/10   SOUTHWEST TITLE CO.    FEBRUARY MORTGAGE PAYMENT    000                     $433.00        $284.94  84.94
CK 1214 X   02/05/10   TEXAS POWER & LIGHT    UTILITY BILL FOR JAN'08      000                      $59.11        $225.83  25.83
CK 1215 X   02/09/10   ACME NATIONAL LIFE     FEBRUARY INS. PREMIUM        000                      $37.02        $188.81  88.81
CK 1216 X   02/11/10   FOOD MART              GROCERIES                    000                      $68.29        $120.52  20.52
CK 1217 X   02/11/10   ABC NATIONAL BANK      CASH                         000                      $25.00         $95.52  95.52
DP          02/17/10                          2007 INCOME TAX REFUND       000   $1,538.00                      $1,633.52  33.52
CK 1218 X   02/20/10   ABC NATIONAL BANK      CASH                         000                      $50.00      $1,583.52  83.52
CK 1219 X   02/21/10   ACME FINANCE           LOAN PAYMENTS 2 AND 3        000                     $179.96      $1,403.56  03.56
CK 1220 X   02/27/10   SOUTHWEST TITLE CO.    MARCH/APRIL MORTGAGE PAYMENT 000                     $866.00        $537.56  37.56
CK 1221 X   02/28/10   COMPANY XYZ            SUIT                         000                     $120.42        $417.14  17.14
CK 1222 X   03/02/10   TEXAS POWER & LIGHT    UTILITY BILL FOR FEB'08      000                      $62.11        $355.56  55.03
CK 1223 X   03/02/10   ABC NATIONAL BANK      CASH                         000                      $25.00        $330.03  30.03
SC          03/03/10                                                       000                       $4.00        $326.03  26.03
CK 1224 X   03/09/10   FOOD MART              GROCERIES                    000                      $72.80        $253.23  53.23
CK 1225 X   03/12/10   SOUTHERN AIRWAYS       TICKET TO LUBBOCK            000                     $227.89         $25.34  25.34
DP          03/15/10                                                       000     $922.00                        $947.34  47.34
CK 1226 X   03/16/10   ABC NATIONAL BANK      CASH                         000                     $250.00        $697.34  97.34
CK 1227 X   03/27/10   ACME NATIONAL LIFE     MARCH/APRIL INS. PREMIUMS    000                      $74.04        $623.30  23.30
CK 1228 X   03/30/10   DISCOUNT AUTO STORE    BATTERY                      000                      $35.70        $587.60  87.60
DP          04/01/10                                                       000     $922.00                      $1,509.60  09.60
CK 1229 X   04/02/10   V.O.I.D.                                            000                       $0.00      $1,509.60  09.60
CK 1230 X   04/02/10   TEXAS POWER & LIGHT    UTILITY BILL FOR MAR'08      000                      $47.18      $1,462.42  62.42
CK 1231 X   04/02/10   FOOD MART              GROCERIES                    000                      $25.90      $1,436.52  36.52
CK 1232 X   04/02/10   ABC NATIONAL BANK      CASH                         000                     $100.00      $1,336.52  36.52
CK 1233 X   04/10/10   ACME FINANCE           LOAN PAYMENT 4               000                      $89.98      $1,246.54  46.54
CK 1234 X   04/10/10   SOUTHWESTERN BELL      PHONE BILLS                  000                      $79.82      $1,166.72  66.72
SC          04/10/10                                                       000                       $4.00      $1,162.72  62.72
CK 1235 X   04/10/10   ABC NATIONAL BANK      CASH                         000                      $50.00      $1,112.72  12.72
CK 1236 X   04/11/10   DISCOUNT AUTO STORE    TUNE-UP KIT                  000                      $21.60      $1,091.12  91.12
CK 1237 X   04/15/10   BERNY'S CHEVROLET      DOWN PAYMENT ON CORVETTE     000                   $1,000.00         $91.12  91.12
DP          04/15/10                          PARTIAL SAVINGS WITHDRAWAL   000   $2,922.00                      $3,013.12  13.12
CK 1238 X   04/15/10   ABC NATIONAL BANK      CASH                         000                      $75.00      $2,938.12  38.12
```

```
CA
24 JUN 2010                                                ACTIVITY REGISTER                                       PAGE     2

-------------------------------------------------------------------------------------------------------------------------------
  TYPE /    DATE  OF   PAYEE                  REASON                                                                 RUNNING
  STATUS    ACTIVITY                                                                +(PLUS)         -(MINUS)         BALANCE
-------------------------------------------------------------------------------------------------------------------------------
CK 1239 X   04/20/10   CITY OF DALLAS         PARKING TICKET               000                          $5.00         $2,933.12
CK 1240 X   04/20/10   ALL-NITE BOWLING       ENTERTAINMENT                000                         $11.55         $2,921.57
CK 1241 X   04/22/10   MODERNAGE TV           REPAIRS                      000                         $75.22         $2,846.35
CK 1242 X   04/25/10   BERNY'S CHEVROLET      CAR PAYMENT 1                000                        $179.00         $2,667.35
CK 1243 X   04/26/10   ABC NATIONAL BANK      CASH                         000                         $25.00         $2,642.35
CK 1244 X   04/26/10   ACME FINANCE           LOAN PAYMENT 5               000                         $89.98         $2,552.37
CK 1245 X   04/29/10   SOUTHWEST TITLE CO.    MAY MORTGAGE PAYMENT         000                        $433.00         $2,119.37
DP          05/01/10                                                       000       $922.00                          $3,041.37
CK 1246 X   05/01/10   ABC NATIONAL BANK      CASH                         000                         $50.00         $2,991.37
CK 1247 X   05/01/10   ACME NATIONAL LIFE     MAY INS. PREMIUM             000                         $37.02         $2,954.35
CK 1248 X   05/01/10   TEXAS POWER & LIGHT    UTILITY BILL FOR APR'08      000                         $55.98         $2,898.37
CK 1249 X   05/01/10   BERNY'S CHEVROLET      CAR PAYMENT 2                000                        $179.00         $2,719.37
CK 1250 X   05/02/10   COMPANY XYZ            BRIEFCASE                    000                         $47.56         $2,671.81
SC          05/02/10                                                       000                          $4.00         $2,667.81
CK 1251 X   05/05/10   ALL-NITE BOWLING       ENTERTAINMENT                000                         $21.60         $2,646.21
CK 1252 X   05/06/10   ABC NATIONAL BANK      CASH                         000                         $50.00         $2,596.21
CK 1253 X   05/10/10   SEARS ROEBUCK          WASHER                       000                        $256.75         $2,339.46
CK 1254 X   05/11/10   FOOD MART              GROCERIES                    000                         $53.89         $2,285.57
CK 1255 X   05/15/10   DISCOUNT AUTO STORE    MISC. AUTO PARTS             000                         $42.04         $2,243.53
DP          05/15/10                                                       000       $962.00                          $3,205.53
CK 1256 X   05/15/10   BERNY'S CHEVROLET      CAR PAYMENT 3                000                        $179.00         $3,026.53
CK 1257 X   05/15/10   H & L FLORISTS         ROSES                        000                         $44.95         $2,981.58
CK 1258 X   05/15/10   COMPANY XYZ            MISC.                        000                         $27.89         $2,953.69
CK 1259 X   05/16/10   ABC NATIONAL BANK      CASH                         000                         $50.00         $2,903.69
CK 1260 X   05/17/10   MODERNAGE TV           TV                           000                        $272.50         $2,631.19
CK 1261 X   05/17/10   RX DRUGS               PRESCRIPTION                 000                         $14.80         $2,616.39
CK 1262 X   05/17/10   ACME HAIR DESIGN       HAIRCUT                      000                         $17.00         $2,599.39
CK 1263 X   05/18/10   SOUTHWESTERN BELL      PHONE BILL                   000                         $55.39         $2,544.00
CK 1264 X   05/18/10   ALL-NITE BOWLING       ENTERTAINMENT                000                         $22.50         $2,521.50
CK 1265 X   05/18/10   JOHN DOE               RETURN CASH ADVANCE          000                         $95.00         $2,426.50
CK 1266 X   05/18/10   ABC NATIONAL BANK      CASH                         000                         $25.00         $2,401.50
CK 1267     05/20/10   ACME NATIONAL LIFE     JUNE INS. PREMIUM            000                         $37.02         $2,364.48
CK 1268     05/21/10   SOUTHWEST TITLE CO.    JUNE MORTGAGE PAYMENT        000                        $433.00         $1,931.48
CK 1269     05/21/10   ABC NATIONAL BANK      CASH                         000                         $25.00         $1,906.48
DP          06/01/10                                                       000       $962.00                          $2,868.48
CK 1270     06/01/10   TEXAS POWER & LIGHT    UTILITY BILL FOR MAY'08      000                         $53.21         $2,815.27
CK 1271     06/02/10   ABC NATIONAL BANK      CASH                         000                         $50.00         $2,765.27
CK 1272     06/02/10   FOOD MART              GROCERIES                    000                         $82.40         $2,682.87
CK 1273     06/02/10   ALL-NITE BOWLING       ENTERTAINMENT                000                         $11.00         $2,671.87
CK 1274 X   06/05/10   V.O.I.D.                                            000                          $0.00         $2,671.87
CK 1275     06/05/10   ABC NATIONAL BANK      CASH                         000                        $150.00         $2,521.87
DP          06/15/10                                                       000       $962.00                          $3,483.87
                                                                                -------------    -------------
GRAND TOTAL                                                                       $12,467.19        $8,983.32
```

```
RUN DIAGNOSTICS
INDEX VIOLATIONS       00000000000
PROGRAM CHECKS         00000000000
PRIMARY I/P RECORDS    00000000093
HITS FOR REPORT   1  - 00000000093
TOTAL RECORDS SELECTED 00000000093
```

Secondary Mode Example 3

```
CA                                       16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                PAGE      1
     1           OPTION  LIST ON MAP SORT EQ NONE
       2           USER    CA

     4           NOTE    ************************************************
     5           NOTE    *                                              *
     6           NOTE    *  THIS REPORTING FACILITY PROGRAM ILLUSTRATES  *
     7           NOTE    *  MULTIPLE DATACOM/DB FILE ACCESS.            *
     8           NOTE    *                                              *
     9           NOTE    *  (THE NEW TECHNIQUES FOR DB FILE             *
    10           NOTE    *   DEFINITION AND ACCESS ARE USED HERE.       *         Multiple CA Datacom/DB tables
    11           NOTE    *                                              *         with automatic User Requirements
    12           NOTE    ************************************************         Table generation
       ┌─────────┐
    14 │ PMF001: │INPUT   DATACOM  RECORD=329
    15 │         │DEFINE  DB-COMMAND            1-5  X
    16 │         │DEFINE  DB-KEY-NAME          6-10  X
    17 │         │DEFINE  DB-ELEMENT-LIST    191-201  X
    18 │         │DEFINE  EMPLOYEE-NUMBER    301-305  X 'EMPLOYEE' ' NUMBER '
    19 │         │DEFINE  NAME               306-329  X
       │         │
    21 │ PAYROLL:│INPUT   DATACOM  RECORD=340 NAME=PAY DBID=001
    22 └─────────┘DEFINE  DB-COMMAND            1-5  X
    23           DEFINE  DB-KEY-NAME          6-10  X
    24           DEFINE  DB-KEY-VALUE        11-15  X
    25           DEFINE  DB-ELEMENT-LIST    191-201  X
    26           DEFINE  EMPLOYEE-CODE          306  X
    27           DEFINE  STATUS                 307  X
    28           DEFINE  CURRENT-RATE       308-315  N3 ' CURRENT ' 'PAY  RATE'
    29           DEFINE  YTD-WAGES          316-323  N2
    30                   'YEAR-TO-DATE'  '   WAGES'  PIC  '$$$,$$9.99'
    31           DEFINE  YTD-COMMISSION     324-331  N2
    32                   'YEAR-TO-DATE'  ' COMMISSION '
    33           DEFINE  YTD-TAXES          332-339  N2
    34                   'YEAR-TO-DATE'  '   TAXES'

           ▲  Notice alternate means of defining CA Datacom/DB tables and duplicate field names in both I/O areas.
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                  PAGE      2
                                                ▼  Field Name Qualifiers
              ┌─────────────────────────────────────────────────────────┐
              │                              ┌──────┐                   │
    36        │   MOVE    'GETIT'         TO │PMF001│.DB-COMMAND        │
    37        │   MOVE    'EMPNO'         TO │PMF001│.DB-KEY-NAME       │  ◄  Sequential access
    38        │   MOVE    'IDEMP      '   TO │PMF001│.DB-ELEMENT-LIST   │
              │                              └──────┘                   │
              └─────────────────────────────────────────────────────────┘
    39            GET     PMF001
    40            GO TO   EOJ WHEN PMF001 EQ END-OF-FILE
              ┌─────────────────────────────────────────────────────────┐
              │                              ┌────────┐                 │
    41        │   MOVE    'REDKG'         TO │PAYROLL.│DB-COMMAND       │
    42        │   MOVE    'EMPNO'         TO │PAYROLL.│DB-KEY-NAME      │  ◄  Random access
    43        │   MOVE    EMPLOYEE-NUMBER TO │PAYROLL.│DB-KEY-VALUE     │
    44        │   MOVE    'PAYRC'         TO │PAYROLL.│DB-ELEMENT-LIST  │
              │                              └────────┘                 │
              └─────────────────────────────────────────────────────────┘
    45            GET     PAYROLL
    46            GO TO   START WHEN PAYROLL EQ NO-RECORD-FOUND

    48            DECODE  STATUS INTO CODE
    49                    'H'  EQ  'HOURLY'
    50                    'S'  EQ  'SALARY'
    51                    ELSE     '******'
    52                             'PAY' 'CODE'
    53            DECODE  EMPLOYEE-CODE INTO EMPCODE
    54                    'A'  EQ  'ACTIVE'
    55                    'I'  EQ  'INACTIVE'
    56                    ELSE     '******'
    57                             'EMPLOYEE' 'CODE'

    59            REPORT  'ACTIVE/INACTIVE PERSONEL REPORT'
    60            SELECT  'A' WHERE EMPLOYEE-CODE EQ 'A'
    61            SELECT  ALL
    62            PRINT   EMPLOYEE-NUMBER NAME EMPCODE A;CODE
    63                    A;CURRENT-RATE (A;YTD-WAGES) (A;YTD-COMMISSION)
    64                    (A;YTD-TAXES)
    65     END

           COMPILE PHASE COMPLETED - NO ERRORS FOUND
                                     NO WARNINGS ISSUED
           START 10:05:58 - STOP 10:06:00
```

```
CA                                         16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                  PAGE      3
COMPILE DIAGNOSTICS

I/P  00001 X 00340 = 00340   PAYROLL
     00001 X 00329 = 00329   PMF001

GSA  00000001677 (00000000118 ENTRIES)

     GSA 1 = 00000000001 NUMERIC CONSTANT ...... CON
     GSA 2 = 00000000031 QUOTED LITERAL ........ LIT
     GSA 3 = 00000000001 INTERMEDIATE RESULT ... RES
     GSA 4 = 00000000076 UNQUOTED LITERAL ...... INT
     GSA 5 = 00000000003 ACCUMULATOR ........... ACC
     GSA 6 = 00000000001 VALUE ................. VAL
     GSA 7 = 00000000000 STRING ................ STR
     GSA 8 = 00000000005 HEX LITERAL ........... HEX

FNT  00000000968 (00000000052 ENTRIES)

     FNT 0 = 00000000006 LABELS ................  01
     FNT 1 = 00000000044 SCALAR VARIABLES ......  01
     FNT 2 = 00000000002 FILE DESC. BLOCKS .....  02
     FNT 3 = 00000000000 ARRAYS ................  03

FST  00000005124 (00000000183 ENTRIES)

     FST 0 = 00000000004 INTERNAL ONLY ......... -02
     FST 1 = 00000000002 USER/REPORT ...........  01
     FST 2 = 00000000015 INPUT AREA VARIABLE ...  02
     FST 3 = 00000000112 GSA VARIABLE ..........  03
     FST 4 = 00000000010 DATA MOVEMENT .........  04
     FST 5 = 00000000006 DECODE ................  05
     FST 6 = 00000000001 SEQUENCE AND CONTROL ..  06
     FST 7 = 00000000010 CONDITION TEST ........  07
     FST 8 = 00000000003 SELECTION .............  08
     FST 9 = 00000000009 PRINT LAYOUT ..........  09
     FST A = 00000000008 HIT RECORD EXTENSION ..  0A
     FST B = 00000000003 GET RECORD ............  0B
     FST C = 00000000000 CONTROL BREAK CALC ....  0C
     FST D = 00000000000 UNDEFINED .............  0D
     FST E = 00000000000 ARRAY DEF/REL. INDEX ..  0E
     FST F = 00000000000 DUMMY .................  0F

GETMAIN REQUIREMENT FOR GSA/FNT/FST ... APPROX. 008K
```

```
CA                                          16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                   PAGE      4
     ------ FIELD NAME ------ ORIGIN  QUAL  HIT ARY  ----------------------------------- REPORTS ------------------------------

     ABORT:                   *LABEL*
     BLANK                    * GSA * (00)
     CODE                     * GSA * (00)   X         1
     CURRDATE                 * GSA * (00)
     CURRENT-DATE             * GSA * (00)
     CURRENT-RATE             PAYROLL (02)   X         1
     CURRENT-TIME             * GSA * (00)
     CURRTIME                 * GSA * (00)
     DATACOM                  * GSA * (01)
     DATACOM                  * GSA * (02)
     DB-COMMAND               PMF001  (01)
     DB-COMMAND               PAYROLL (02)
     DB-ELEMENT-LIST          PMF001  (01)
     DB-ELEMENT-LIST          PAYROLL (02)
     DB-KEY-NAME              PMF001  (01)
     DB-KEY-NAME              PAYROLL (02)
     DB-KEY-VALUE             PAYROLL (02)
     DD                       * GSA * (00)
     EMPCODE                  * GSA * (00)   X         1
     EMPLOYEE-CODE            PAYROLL (02)
     EMPLOYEE-NUMBER          PMF001  (01)   X         1
     END-OF-FILE              * GSA * (00)
     EOJ:                     *LABEL*
     HR                       * GSA * (00)
     MM                       * GSA * (00)
     MN                       * GSA * (00)
     NAME                     PMF001  (01)   X         1
     NO-RECORD-FOUND          * GSA * (00)
     PAYROLL                  * GSA * (00)
     PAYROLL:                 *LABEL*
     PMF001                   * GSA * (00)
     PMF001:                  *LABEL*
     QSEQ                     * GSA * (00)
     RECORD-FOUND             * GSA * (00)
     SPACE                    * GSA * (00)
     SPACES                   * GSA * (00)
     SS                       * GSA * (00)
     START:                   *LABEL*
     STATUS                   PAYROLL (02)
     TAG                      * GSA * (00)
     TEST:                    *LABEL*
     XD                       * GSA * (00)
     XM                       * GSA * (00)
     XY                       * GSA * (00)
     YTD-COMMISSION           PAYROLL (02)   X         1
     YTD-TAXES                PAYROLL (02)   X         1
     YTD-WAGES                PAYROLL (02)   X         1
     YY                       * GSA * (00)
     ZERO                     * GSA * (00)
     ZEROS                    * GSA * (00)
```

```
CA                                         16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                  PAGE      5
                                                    (3350   ) ......H I T   F I L E   L A Y O U T......

 FD  DROUT                                    DROUT:  FILE DISK SEQUENTIAL VARIABLE RECORD=         90 BLOCK=        4562
       BLOCK CONTAINS        4562 CHARACTERS
       RECORD CONTAINS          90 CHARACTERS
       RECORDING MODE V
       DATA RECORD IS HITFILE.

 01  HITFILE.
       02 RDW                                 PICTURE X(4)                DEFINE RDW                             1 -    4 X
       02 TAG-CHARACTER                       PICTURE X                   DEFINE TAG-CHARACTER                   5 -    5 X
       02 SORT-KEY.
          03 REPORT-NUMBER                    PICTURE B                   DEFINE REPORT-NUMBER                   6 -    6 B
          03 USER-SORT-KEY                    PICTURE X(9)                DEFINE USER-SORT-KEY                   7 -   15 X
       02 REPORT-01.                          ACTIVE/INACTIVE PERSONEL REPORT
          03 01.EMPLOYEE-NUMBER               PICTURE X(5).               DEFINE 01.EMPLOYEE-NUMBER             16 -   20 X
          03 01.NAME                          PICTURE X(24)               DEFINE 01.NAME                        21 -   44 X
          03 00.EMPCODE                       PICTURE X(8)                DEFINE 00.EMPCODE                     45 -   52 X
          03 00.CODE                          PICTURE X(6)                DEFINE 00.CODE                        53 -   58 X
          03 02.CURRENT-RATE                  PICTURE 9(5)V999            DEFINE 02.CURRENT-RATE                59 -   66 N3
          03 02.YTD-WAGES                     PICTURE 9(6)V99             DEFINE 02.YTD-WAGES                   67 -   74 N2
          03 02.YTD-COMMISSION                PICTURE 9(6)V99             DEFINE 02.YTD-COMMISSION              75 -   82 N2
          03 02.YTD-TAXES                     PICTURE 9(6)V99             DEFINE 02.YTD-TAXES                   83 -   90 N2
```

```
CA                                         16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                    PAGE     6
  REPORT   01 FIELD NAME                   HDR1  HDR2  DTL   ACC   TOTAL       PICTURE          TAG   FROM   TO

  LINE   01   01.EMPLOYEE-NUMBER              8     8     5             8                              1     8
              01.NAME                         4          24            24                             14    37
              00.EMPCODE                      8     4     8             8                             43    50
              00.CODE                         3     4     6             6                      A      56    61
              02.CURRENT-RATE                 9     9    10            10        99999.999-    A      67    76
              02.YTD-WAGES                   12     8    11    13      13        $$$,$$9.99    A      82    94
              02.YTD-COMMISSION              12    12    10    12      12        999999.99-    A     102   113
              02.YTD-TAXES                   12     8    10    12      12        999999.99-    A     121   132
```

```
CA
24 JUN 2010                                       ACTIVE/INACTIVE PERSONEL REPORT                                  PAGE     1

----------------------------------------------------------------------------------------------------------------------------------
EMPLOYEE     NAME                         EMPLOYEE     PAY         CURRENT       YEAR-TO-DATE       YEAR-TO-DATE     YEAR-TO-DATE
 NUMBER                                   CODE         CODE        PAY RATE             WAGES        COMMISSION             TAXES
----------------------------------------------------------------------------------------------------------------------------------
00001        ALLEN ANTHONY                ACTIVE       SALARY      300.000        $78,000.00          25000.00         23000.00
00002        LENZEN RUTH                  ACTIVE       SALARY       40.500        $10,530.00           1000.00           900.00
00003        MEAD HENRY                   ACTIVE       SALARY       40.500        $10,530.00           1100.00           900.00
00004        PARKER WILLIAM               ACTIVE       SALARY       40.000        $10,400.00           1050.00           900.00
00005        RIGSBY JOHN                  ACTIVE       HOURLY        0.855         $8,892.00           1500.00           700.00
00006        SNOW TERRI                   ACTIVE       SALARY       41.000        $10,660.00           1150.00           950.00
00007        TURNER ARNOLD                INACTIVE
00008        WEEKS LEN                    ACTIVE       SALARY       40.500        $10,530.00           1050.00           900.00
00009        LUTHER GARY                  ACTIVE       HOURLY        0.855         $8,892.00           1500.00           700.00
00010        BRATTON GEORGE               ACTIVE       SALARY       40.000        $10,400.00           1000.00           850.00
00011        CONLEY KEITH                 INACTIVE
00012        EDWARDS BARRY                ACTIVE       SALARY       40.000        $10,400.00           1000.00           850.00
00013        COURTNEY ELIZABETH           ACTIVE       HOURLY        1.000        $10,400.00           2500.00          2000.00
00014        HAGAN LEO                    ACTIVE       SALARY       40.000        $10,400.00           1000.00           850.00
00015        HOUSE GREG                   ACTIVE       SALARY       40.000        $10,400.00           1000.00           850.00
00016        WALKER FRANK                 INACTIVE
00017        DAVENPORT JACQUELINE         ACTIVE       SALARY      100.000        $26,000.00          10000.00         15000.00
00018        PATTERSON AL                 ACTIVE       SALARY       40.000        $10,400.00           1000.00           850.00
00019        HAMILTON DAVID               ACTIVE       SALARY       40.000        $10,400.00           1000.00           850.00
00020        SIMMONDS FELIX               INACTIVE
00021        HOPKINS BARRY                INACTIVE
00022        SIMONSEN RONALD              INACTIVE
00023        FARRIOR HOWARD               INACTIVE
00024        PELTIER ROSS                 ACTIVE       SALARY      100.000        $26,000.00           8500.00         11500.00
00025        DRUMMOND GEORGE              ACTIVE       SALARY       90.000        $23,400.00           7500.00         10500.00
00026        SIDWEBER PATRICIA            ACTIVE       SALARY       95.000        $24,700.00           7900.00         11000.00
00027        FULLERTON CARL               ACTIVE       HOURLY        1.100        $11,440.00           2500.00          1500.00
00028        LOVELL RAYMOND               ACTIVE       SALARY       90.000        $23,400.00           8500.00         11000.00
00029        NEEDHAM TIMOTHY              INACTIVE
00030        EVERS DANNY                  ACTIVE       SALARY       97.500        $25,350.00           9000.00          5500.00
00031        YOUNG MICHAEL                ACTIVE       SALARY       92.500        $24,050.00           7500.00          6000.00
00032        SCOTT DAVID                  INACTIVE
00033        NARMOUR ROBERT               ACTIVE       HOURLY        0.865         $8,996.00           1500.00           700.00
00034        OLVERA DONALD                ACTIVE       SALARY       85.000        $22,100.00           7000.00          5500.00
00035        SEAGRAVES ROBERT             ACTIVE       SALARY       80.000        $20,800.00           7300.00          5300.00
00036        MOTT THOMAS                  ACTIVE       SALARY       90.000        $23,400.00              0.00          5800.00
00037        MCEVOY PATRICK               INACTIVE
00038        BREEDEN JERRY                ACTIVE       SALARY       78.500        $20,410.00              0.00          4500.00
00039        CARR STEVE                   ACTIVE       SALARY       87.500        $22,750.00           5000.00          7550.00
00040        DOBBS ANNA                   ACTIVE       HOURLY        0.950         $9,880.00           2000.00          2700.00
00041        FITZHUGH JAMES               INACTIVE
00042        GILMER DONALD                ACTIVE       HOURLY        0.925         $9,620.00           2000.00          2500.00
00043        AINSWORTH STEVE              ACTIVE       SALARY       88.500        $23,010.00              0.00          5300.00
00044        HIFFMAN GREGORY              INACTIVE
00045        JONES WILLIAM                ACTIVE       SALARY       75.000        $19,500.00              0.00          4500.00
00046        LANE GRACE                   ACTIVE       SALARY       75.000        $19,500.00              0.00          4700.00
00047        THOMPSON JEFF                ACTIVE       HOURLY        0.925         $9,620.00           1500.00          2300.00
00048        WALLACE RUTH                 ACTIVE       SALARY       45.000         $1,000.00              0.00          2500.00
00049        YATES FLOYD                  INACTIVE
00050        POWELL ROBERT                ACTIVE       SALARY       80.000        $20,800.00           1500.00          4600.00
00051        MOORE VICTOR                 ACTIVE       SALARY       76.500        $19,890.00           1700.00          4650.00
```

```
CA
24 JUN 2010                                         ACTIVE/INACTIVE PERSONEL REPORT                                PAGE     2

--------------------------------------------------------------------------------------------------------------------------------
EMPLOYEE     NAME                         EMPLOYEE     PAY         CURRENT      YEAR-TO-DATE       YEAR-TO-DATE     YEAR-TO-DATE
 NUMBER                                   CODE         CODE        PAY RATE            WAGES         COMMISSION            TAXES
--------------------------------------------------------------------------------------------------------------------------------
00052        JEFFERSON ANTON              ACTIVE       SALARY       72.500        $18,850.00           1600.00          4350.00
00053        GRAHAM PATRICK               INACTIVE
00054        COOK HOWARD                  ACTIVE       SALARY       75.500        $19,630.00           1900.00          4850.00
00055        BAXTER JACK                  ACTIVE       SALARY       73.000        $18,980.00           2000.00          5000.00
00056        HIGGINS BENJAMIN             INACTIVE
00057        FINLEY MARY                  ACTIVE       SALARY       76.000        $19,760.00           1900.00          4900.00
00058        RHODES JOHN                  ACTIVE       HOURLY        0.910         $9,464.00           1500.00          2250.00
00059        NASH KEITH                   ACTIVE       SALARY       72.000        $18,720.00           1200.00          5750.00
00060        KEOMIG RICHARD               ACTIVE       SALARY       72.500        $18,850.00           1500.00          2700.00
00061        GANTZ CAROLINE               ACTIVE       SALARY       70.000        $18,200.00           1500.00          2650.00
00062        CARGILL FORREST              INACTIVE
00063        AVERY SCOTT                  ACTIVE       SALARY       71.000        $18,460.00           2000.00          2600.00
00064        BLAIR RONALD                 ACTIVE       SALARY       70.000        $18,200.00           2500.00          2600.00
00065        FISHER JOHN                  INACTIVE
00066        HAWKINS ANDREW               ACTIVE       SALARY       69.000        $17,940.00           1600.00          2650.00
00067        KENT CHARLES                 ACTIVE       SALARY       68.000        $17,680.00           1850.00          2600.00
00068        MOREY ELMER                  ACTIVE       SALARY       67.500        $17,550.00           2300.00          2550.00
00069        ROBERTS MELVIN               ACTIVE       HOURLY        0.900         $9,360.00           1500.00          1000.00
00070        SANDERS JACK                 ACTIVE       SALARY       69.500        $18,070.00           2150.00          2650.00
00071        WORDEN EDWARD                ACTIVE       SALARY       68.500        $17,810.00           1200.00          2500.00
00072        SMALLEY TRAVIS               INACTIVE
00073        MORAN VERNON                 ACTIVE       SALARY       67.500        $17,550.00           1350.00          2550.00
00074        KELLY EUGENE                 ACTIVE       HOURLY        0.900         $9,360.00           1500.00          1000.00
00075        CLARKE LARRY                 ACTIVE       SALARY       65.000        $16,900.00           2150.00          2500.00
00076        AGEE FRANK                   INACTIVE
00077        BREEN GEORGE                 ACTIVE       SALARY       66.000        $17,160.00           1750.00          2550.00
00078        CHANDLER DIANE               ACTIVE       SALARY       63.000        $16,380.00           1600.00          2300.00
00079        GARRISON WILLIAM             ACTIVE       HOURLY        0.890         $9,256.00           1500.00          1000.00
00080        HILL CONRAD                  ACTIVE       SALARY       66.500        $17,290.00           1800.00          2300.00
00081        MORGAN JOSEPH                ACTIVE       SALARY       62.500        $16,250.00           1725.00          2100.00
00082        PULLIAM JAMES                ACTIVE       SALARY       63.500        $16,510.00           1250.00          2150.00
00083        TURNER ALBERT                ACTIVE       SALARY       65.000        $16,900.00           1300.00          2300.00
00084        WHITE PAUL                   ACTIVE       SALARY       64.500        $16,770.00           2300.00          2500.00
00085        SIMPSON HARVEY               ACTIVE       SALARY       63.500        $16,510.00           2000.00          2400.00
00086        RHODES ROYCE                 INACTIVE
00087        OSBORN DONALD                ACTIVE       HOURLY        0.890         $9,256.00           1500.00           850.00
00088        ERWIN ROY                    ACTIVE       SALARY       61.500        $15,990.00           1700.00          2200.00
00089        HOUSLEY EARL                 INACTIVE
00090        IVERSON HECTOR               ACTIVE       SALARY       62.500        $16,250.00           1500.00          2250.00
00091        PARKER STUART                ACTIVE       SALARY       62.000        $16,120.00           1200.00          2200.00
00092        MCDAVID KENNETH              ACTIVE       HOURLY        0.900         $9,360.00           1500.00           850.00
00093        HANSON FRED                  ACTIVE       SALARY       61.500        $15,990.00           1300.00          2000.00
00094        CARLSON RICHARD              ACTIVE       SALARY       61.000        $15,860.00           1250.00          2000.00
00095        JOPLIN TINA                  ACTIVE       SALARY       62.000        $16,120.00           1550.00          2100.00
00096        HENDRICK RONALD              ACTIVE       SALARY       61.000        $15,860.00           1800.00          2150.00
00097        BRET JOHN                    INACTIVE
00098        EVANS LARRY                  ACTIVE       SALARY       62.000        $16,120.00           2100.00          2300.00
00099        HOYLE RUTH                   ACTIVE       SALARY       61.500        $15,990.00           2350.00          2400.00
00100        MADDEN SAMUEL                ACTIVE       HOURLY        0.880         $9,152.00           1500.00           800.00
00101        PULLIAM MARK                 INACTIVE
00102        WILKES EARL                  ACTIVE       SALARY       61.000        $15,860.00           2150.00          2250.00
```

```
CA
24 JUN 2010                                         ACTIVE/INACTIVE PERSONEL REPORT                                PAGE     3

---------------------------------------------------------------------------------------------------------------------------------
EMPLOYEE     NAME                         EMPLOYEE     PAY         CURRENT       YEAR-TO-DATE       YEAR-TO-DATE     YEAR-TO-DATE
 NUMBER                                   CODE         CODE        PAY RATE             WAGES        COMMISSION             TAXES
---------------------------------------------------------------------------------------------------------------------------------
00103        JARRETT LOUIS                ACTIVE       SALARY       60.000         $15,600.00           2050.00          2150.00
00104        BURGEN JAYNE                 ACTIVE       SALARY       60.500         $15,730.00           1800.00          2150.00
00105        CHURCH PHILLIP               ACTIVE       SALARY       60.000         $15,600.00           1950.00          2100.00
00106        KIDWELL DONALD               ACTIVE       HOURLY        0.885          $9,204.00           1500.00           850.00
00107        ROCKWELL SUSAN               ACTIVE       SALARY       58.500         $15,210.00           1200.00          1900.00
00108        WALKER DENNIS                ACTIVE       SALARY       59.000         $15,340.00           1350.00          2000.00
00109        WOLF PAUL                    ACTIVE       SALARY       58.000         $15,080.00           1425.00          1950.00
00110        STEWARD BYRON                ACTIVE       SALARY       59.000         $15,340.00           1850.00          2100.00
00111        PATMAN ROBERT                ACTIVE       SALARY       57.500         $14,950.00           1900.00          2000.00
00112        LOPEZ DAVID                  ACTIVE       HOURLY        0.875          $9,100.00           1500.00           800.00
00113        FLYNN GARY                   ACTIVE       SALARY       57.000         $14,820.00           2100.00          2000.00
00114        CAREY FRED                   INACTIVE
00115        ABEL PHILIP                  ACTIVE       SALARY       58.000         $15,080.00           1250.00          1900.00
00116        HOLCOMB MARK                 INACTIVE
00117        PRICE JOEL                   ACTIVE       SALARY       57.000         $14,820.00           1350.00          1850.00
00118        SUTER MORRIS                 ACTIVE       SALARY       56.500         $14,690.00           2000.00          1950.00
00119        WEIR HARVEY                  ACTIVE       HOURLY        0.875          $9,100.00           1500.00           800.00
00120        SAUNDERS LOUIS               ACTIVE       SALARY       57.500         $14,950.00           1750.00          1950.00
00121        BAGNALL CLIFTON              ACTIVE       SALARY       56.000         $14,560.00           1650.00          1850.00
00122        FONTANA WALLACE              ACTIVE       SALARY       57.000         $14,820.00           1950.00          2000.00
00123        NEELY ROY                    ACTIVE       SALARY       56.000         $14,560.00           2150.00          2000.00
00124        WREN PATRICIA                INACTIVE
00125        SMITH DANIEL                 INACTIVE
00126        RICHMOND SCOTT               ACTIVE       SALARY       55.000         $14,300.00           1200.00          1750.00
00127        LANSING STEPHEN              ACTIVE       SALARY       51.000         $13,260.00           1850.00          1650.00
00128        DOUGAN RAYMOND               ACTIVE       SALARY       52.500         $13,650.00           1150.00          1550.00
00129        BEACH MARCUS                 ACTIVE       HOURLY        0.885          $9,204.00           1500.00           800.00
00130        DIETER RODNEY                ACTIVE       SALARY       51.000         $13,260.00           1250.00          1550.00
00131        HAYWOOD VANCE                ACTIVE       SALARY       51.500         $13,390.00           2700.00          2000.00
00132        PELTON LEWIS                 ACTIVE       SALARY       53.500         $13,910.00           1650.00          1750.00
00133        WASHBURN STELLA              ACTIVE       SALARY       52.500         $13,650.00           2050.00          1750.00
00134        FREEMAN BERNARD              ACTIVE       HOURLY        0.885          $9,204.00           1500.00           800.00
00135        JACKSON WILBURN              ACTIVE       SALARY       53.500         $13,910.00           1900.00          1800.00
00136        LINDSTROM ROGER              ACTIVE       SALARY       51.000         $13,260.00           1250.00          1550.00
00137        SHEPHERD IRA                 ACTIVE       SALARY       50.000         $13,000.00           1350.00          1500.00
00138        DECKER TERRY                 INACTIVE
00139        NOLAND TURNER                ACTIVE       SALARY       49.000         $12,740.00           1700.00          1500.00
00140        CAMPBELL WAYNE               ACTIVE       SALARY       48.500         $12,610.00           1100.00          1400.00
00141        KERBY RICHARD                ACTIVE       HOURLY        0.880          $9,152.00           1500.00           800.00
00142        MORAN SIMON                  ACTIVE       SALARY       50.000         $13,000.00           1200.00          1650.00
00143        SHULTZ IRWIN                 INACTIVE
00144        TRENT PETER                  ACTIVE       SALARY       47.500         $12,350.00           1000.00          1300.00
00145        GOODWIN PATRICK              ACTIVE       SALARY       48.000         $12,480.00           1000.00          1300.00
00146        LANE BARRY                   ACTIVE       HOURLY        0.880          $9,152.00           1500.00           800.00
00147        MAYFIELD GERALD              ACTIVE       SALARY       47.500         $12,350.00           1000.00          1300.00
00148        TENNISON CLINTON             ACTIVE       SALARY       47.500         $12,350.00           1000.00          1300.00
00149        HORN JAMES                   ACTIVE       SALARY       46.500         $12,090.00           1050.00          1250.00
00150        GEIGER ROBERT                ACTIVE       SALARY       46.500         $12,090.00           1100.00          1250.00
00151        POSEY NANA                   INACTIVE
00152        LONG ROY                     ACTIVE       SALARY       46.000         $11,960.00           1500.00          1150.00
00153        SCHAFER ARTHUR               ACTIVE       HOURLY        0.875          $9,100.00           1500.00           750.00
```

```
CA
24 JUN 2010                                         ACTIVE/INACTIVE PERSONEL REPORT                                PAGE     4

---------------------------------------------------------------------------------------------------------------------------------
EMPLOYEE     NAME                         EMPLOYEE     PAY         CURRENT       YEAR-TO-DATE       YEAR-TO-DATE     YEAR-TO-DATE
 NUMBER                                   CODE         CODE        PAY RATE             WAGES        COMMISSION             TAXES
---------------------------------------------------------------------------------------------------------------------------------
00154        OWEN JOHN                    ACTIVE       SALARY       46.000         $11,960.00           1400.00          1100.00
00155        NALL CAROL                   INACTIVE
00156        WRIGHT BILL                  ACTIVE       SALARY       45.000         $11,700.00           1350.00          1100.00
00157        TONROY RAY                   ACTIVE       HOURLY        0.875          $9,100.00           1500.00           750.00
00158        IVEY SCOTT                   ACTIVE       SALARY       45.500         $11,830.00           1200.00          1100.00
00159        FIELDS BEN                   ACTIVE       SALARY       46.000         $11,960.00           1250.00          1200.00
00160        CHILDS ROBIN                 ACTIVE       SALARY       45.000         $11,700.00           1150.00          1050.00
00161        AMADON KEVIN                 ACTIVE       SALARY       45.500         $11,830.00           1100.00          1050.00
00162        AKARD LISA                   ACTIVE       SALARY       45.000         $11,700.00           1100.00          1050.00
00163        BURCH DONALD                 INACTIVE
00164        JUDD LENA                    ACTIVE       HOURLY        0.870          $9,048.00           1500.00           750.00
00165        MOORING LOUIS                ACTIVE       SALARY       44.500         $11,570.00           1200.00          1000.00
00166        THORNBERG JERRY              ACTIVE       SALARY       44.500         $11,570.00           1000.00           950.00
00167        BICKMAN FRANK                ACTIVE       SALARY       45.500         $11,830.00           1050.00          1000.00
00168        STANYER VERNON               ACTIVE       SALARY       44.000         $11,440.00           1000.00          1000.00
00169        NATHAN TERESA                INACTIVE
00170        MC BETH CHARLES              ACTIVE       HOURLY        0.870          $9,048.00           1500.00           750.00
00171        HOSEK DONALD                 ACTIVE       SALARY       44.000         $11,440.00           1100.00           950.00
00172        EVERTS PRICE                 ACTIVE       SALARY       43.500         $11,310.00           1150.00          1000.00
00173        BRASHEAR DAVID               ACTIVE       SALARY       44.500         $11,570.00           1200.00          1050.00
00174        ABSHIRE SUE                  ACTIVE       SALARY       43.000         $11,180.00           1100.00          1050.00
00175        IVEY FRANK                   ACTIVE       SALARY       43.000         $11,180.00           1000.00          1000.00
00176        LANGSTON MELVIN              ACTIVE       SALARY       42.500         $11,050.00           1200.00          1000.00
00177        REDMOND FRED                 ACTIVE       HOURLY        0.865          $8,996.00           1500.00           700.00
00178        WEISBERG LESLIE              INACTIVE
00179        VICKERY PETE                 INACTIVE
00180        TOVAR ED                     ACTIVE       SALARY       42.000         $10,920.00           1050.00           900.00
00181        OTTO DAN                     ACTIVE       SALARY       42.500         $11,050.00           1300.00           950.00
00182        NEEDUM VERNICE               ACTIVE       SALARY       42.000         $10,920.00           1250.00           950.00
00183        MERCHAN JOHN                 ACTIVE       HOURLY        0.865          $8,996.00           1500.00           700.00
00184        LINDGREN DONALD              ACTIVE       SALARY       42.000         $10,920.00           1100.00          1000.00
00185        HOLLIS ROBERT                ACTIVE       SALARY       41.500         $10,790.00           1200.00          1000.00
00186        GRIER BRAD                   ACTIVE       SALARY       41.500         $10,790.00           1250.00           950.00
00187        FIELD RAY                    ACTIVE       SALARY       42.000         $10,920.00           1100.00          1000.00
00188        SPEERS ELLEN                 ACTIVE       SALARY       42.000         $10,920.00           1100.00          1000.00
00189        ROWELL PATRICK               ACTIVE       SALARY       41.500         $10,790.00           1000.00           900.00
00190        MILLER CHRIS                 ACTIVE       HOURLY        0.860          $8,944.00           1500.00           700.00
00191        KLEIN CAROL                  INACTIVE
00192        HORST BRUCE                  ACTIVE       SALARY       41.500         $10,790.00           1000.00           900.00
00193        GRANGER BUCK                 ACTIVE       SALARY       41.000         $10,660.00           1150.00           950.00
00194        EGGERT TOM                   INACTIVE
00195        DABBS GARY                   ACTIVE       SALARY       41.000         $10,660.00           1100.00           900.00
00196        BENHAM JERRY                 ACTIVE       HOURLY        0.860          $8,944.00           1200.00           700.00
00197        ATWELL SARAH                 ACTIVE       SALARY       40.500         $10,530.00           1100.00           900.00
00198        BROWN PETE                   ACTIVE       SALARY       41.500         $10,790.00           1050.00           900.00
00199        FISCHER ROGER                ACTIVE       SALARY       41.000         $10,660.00           1100.00           900.00
00200        HOWARD LANE                  ACTIVE       SALARY       41.000         $10,660.00           1150.00           950.00
                                                                                 -------------      ------------     ------------
GRAND TOTAL                                                                      $2315,780.00         318050.00        379000.00
```

```
RUN DIAGNOSTICS
INDEX VIOLATIONS       00000000000
PROGRAM CHECKS         00000000000
HITS FOR REPORT   1  - 00000000200
TOTAL RECORDS SELECTED 00000000200
```

Write-Only Mode

The Write-Only mode allows creation of a single sequential output file rather than report generation. Enter this mode by specifying the OPTION WRITE ONLY command in the Reporting Facility program.

You can perform two main processes in this mode.

- Compilation/Code Generation
- Record Selection/Data Manipulation/Output File Creation

Data sorting is not available in this mode. When input data reaches END-OF-FILE, the output file closes and the run immediately terminates.

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      1
  ┌──────────────────────────────────────────────────┐
  │ 1            OPTION     LIST ON MAP WRITE ONLY   │ ◄  Enter Write Only mode explicitly
  └──────────────────────────────────────────────────┘
      2            USER    CA

    4            NOTE       **********************************************
    5            NOTE       *                                            *
    6            NOTE       *      FORMAT 80 BYTE CARD IMAGES            *
    7            NOTE       *                                            *
    8            NOTE       *  THIS REPORTING FACILITY PROGRAM PRODUCES  *
    9            NOTE       *  A SEQUENTIAL DISK OUTPUT FILE.            *
   10            NOTE       *                                            *
   11            NOTE       **********************************************

   13   TRANS:   FILE       CARD
   14            DEF        TRANSACTION-RECORD 1-80 X
   15            SELECT     ALL
   16            FORMAT     TRANSACTION-RECORD
   17   END



        COMPILE PHASE COMPLETED - NO ERRORS FOUND
                                  NO WARNINGS ISSUED
        START 14:07:30 - STOP 14:07:31
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      2
COMPILE DIAGNOSTICS

I/P  00001 X 00080 = 00080   TRANS
O/P  00057 X 00080 = 04560   HITFILE

GSA  00000000816 (00000000045 ENTRIES)

     GSA 1 = 00000000001 NUMERIC CONSTANT ...... CON
     GSA 2 = 00000000003 QUOTED LITERAL ........ LIT
     GSA 3 = 00000000001 INTERMEDIATE RESULT ... RES
     GSA 4 = 00000000038 UNQUOTED LITERAL ...... INT
     GSA 5 = 00000000000 ACCUMULATOR ........... ACC
     GSA 6 = 00000000001 VALUE ................. VAL
     GSA 7 = 00000000000 STRING ................ STR
     GSA 8 = 00000000001 HEX LITERAL ........... HEX

FNT  00000000528 (00000000031 ENTRIES)

     FNT 0 = 00000000005 LABELS ................  01
     FNT 1 = 00000000025 SCALAR VARIABLES ......  01
     FNT 2 = 00000000001 FILE DESC. BLOCKS .....  02
     FNT 3 = 00000000000 ARRAYS ................  03

FST  00000001820 (00000000065 ENTRIES)

     FST 0 = 00000000004 INTERNAL ONLY ......... -02
     FST 1 = 00000000002 USER/REPORT ...........  01
     FST 2 = 00000000001 INPUT AREA VARIABLE ...  02
     FST 3 = 00000000052 GSA VARIABLE ..........  03
     FST 4 = 00000000001 DATA MOVEMENT .........  04
     FST 5 = 00000000000 DECODE ................  05
     FST 6 = 00000000000 SEQUENCE AND CONTROL ..  06
     FST 7 = 00000000001 CONDITION TEST ........  07
     FST 8 = 00000000002 SELECTION .............  08
     FST 9 = 00000000000 PRINT LAYOUT ..........  09
     FST A = 00000000001 HIT RECORD EXTENSION ..  0A
     FST B = 00000000001 GET RECORD ............  0B
     FST C = 00000000000 CONTROL BREAK CALC ....  0C
     FST D = 00000000000 UNDEFINED .............  0D
     FST E = 00000000000 ARRAY DEF/REL. INDEX ..  0E
     FST F = 00000000000 DUMMY .................  0F

GETMAIN REQUIREMENT FOR GSA/FNT/FST ... APPROX. 004K
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      3
     ------ FIELD NAME ------ ORIGIN  QUAL  HIT ARY  ---------------------------------- REPORTS -----------------------------

     ABORT:                   *LABEL*
     BLANK                    * GSA * (00)
     CURRDATE                 * GSA * (00)
     CURRENT-DATE             * GSA * (00)
     CURRENT-TIME             * GSA * (00)
     CURRTIME                 * GSA * (00)
     DD                       * GSA * (00)
     END-OF-FILE              * GSA * (00)
     EOJ:                     *LABEL*
     HR                       * GSA * (00)
     MM                       * GSA * (00)
     MN                       * GSA * (00)
     NO-RECORD-FOUND          * GSA * (00)
     QSEQ                     * GSA * (00)                  No report generation
     RECORD-FOUND             * GSA * (00)
     SPACE                    * GSA * (00)
     SPACES                   * GSA * (00)
     SS                       * GSA * (00)
     START:                   *LABEL*
     TAG                      * GSA * (00)
     TEST:                    *LABEL*
     TRANS                    * GSA * (00)
     TRANS:                   *LABEL*
     TRANSACTION-RECORD       TRANS   (01)   X
     XD                       * GSA * (00)
     XM                       * GSA * (00)
     XY                       * GSA * (00)
     YY                       * GSA * (00)
     ZERO                     * GSA * (00)
     ZEROS                    * GSA * (00)
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      4
                                                   (3350   ) ......H I T   F I L E   L A Y O U T......

                                         ┌──────┐
 FD  DROUT                               │DROUT:│ FILE DISK SEQUENTIAL VARFIXED   CORD=         80 BLOCK=       4560
                                         └──────┘
       BLOCK CONTAINS        4560 CHARACTERS     ▲  Actual name of output data set
       RECORD CONTAINS          80 CHARACTERS
       RECORDING MODE F
       DATA RECORD IS HITFILE.

 01  HITFILE.
          03 01.TRANSACTION-RECORD        PICTURE X(80).              DEFINE 01.TRANSACTION-RECORD          1 -    80 X

RUN DIAGNOSTICS
INDEX VIOLATIONS       00000000000
PROGRAM CHECKS         00000000000
PRIMARY I/P RECORDS    00000000093
HITS FOR REPORT   1  - 00000000093
                      ┌───────────┐
TOTAL RECORDS SELECTED│00000000093│
                      └───────────┘
                            ▲  Actually represents number of output records created
```

Library Maintenance Mode

The Library Maintenance mode provides the only means for maintaining the Reporting Facility call library. Using this library enables you to catalog frequently used statements or groups of statements for later retrieval.

You can enter Library Maintenance mode *only* by specifying the LIBRARY command. If you specify the LIBRARY command, it must be the first command in the input source stream.

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      1
     1           NOTE    ************************************************
     2           NOTE    *                                              *
     3           NOTE    *  THIS REPORTING FACILITY PROGRAM ILLUSTRATES *
     4           NOTE    *  THE LIBRARY MAINTENANCE FACILITY.           *
     5           NOTE    *  SPECIFICALLY, THE LOAD, DISPLAY, AND        *
     6           NOTE    *  CONDENSE FUNCTIONS.                         *
     7           NOTE    *                                              *
     8   NOTE DRTEST10    LOAD TO DRLIB
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      2
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - LOAD


          CALL NAME = TRANFILE CHECK REGISTER TRANSACTION FILE

                      *** MEMBER UPDATED ***


          TRANS:   FILE    CARD
                   DEF     TRANSACTION-CODE    1 THRU  2  X ' '
                   DEF     ID-NUMBER           3 THRU  6  X ' '
                   DEF     AMOUNT              7 THRU 13  N2
                   DEF     ACTIVITY-DATE      14 THRU 21  X 'DATE  OF' 'ACTIVITY'
                   DEF     CLEARED                    22  X
                   DEF     PAYEE              23 THRU 42  X
                   DEF     REASON             44 THRU 75  X
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      3
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - LOAD


          CALL NAME = PMF010   DATABASE PERSONEL MASTER FILE

                      *** MEMBER UPDATED ***

                                       ▼  Macro substitute parameter
                                     ┌─────┐
          PMF010:  FILE    DATACOM   │ :01 │   RECORD=329
                                     └─────┘
                   DEFINE  DB-COMMAND           1-5  X
                   DEFINE  DB-KEY-NAME          6-10  X
                   DEFINE  DB-ELEMENT-LIST    191-201  X
                   DEFINE  EMPLOYEE-NUMBER    301-305  X 'EMPLOYEE' ' NUMBER '
                   DEFINE  NAME               306-329  X
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      4
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - LOAD


          CALL NAME = PAY010   DATABASE PAYROLL FILE

                      *** MEMBER UPDATED ***


          PAY010:  FILE    DATACOM    :01     RECORD=340
                   DEFINE  DB-COMMAND            1-5  X
                   DEFINE  DB-KEY-NAME          6-10  X
                   DEFINE  DB-KEY-VALUE        11-15  X
                   DEFINE  DB-ELEMENT-LIST    191-201  X
                   DEFINE  EMPLOYEE-CODE         306  X
                   DEFINE  STATUS                307  X
                   DEFINE  CURRENT-RATE       308-315  N3 ' CURRENT ' 'PAY  RATE'
                   DEFINE  YTD-WAGES          316-323  N2
                           'YEAR-TO-DATE'  '   WAGES'  PIC  '$$$,$$9.99'
                   DEFINE  YTD-COMMISSION     324-331  N2
                           'YEAR-TO-DATE'  ' COMMISSION '
                   DEFINE  YTD-TAXES          332-339  N2
                           'YEAR-TO-DATE'  '   TAXES'
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      5
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - LOAD


          CALL NAME = DECODE1  ALPHANUMERIC MONTH TRANSLATION TABLE

                      *** MEMBER UPDATED ***


                   DECODE  :01 INTO :02
                             01 = 'JANUARY'
                             02 = 'FEBRUARY'
                             03 = 'MARCH'
                             04 = 'APRIL'
                             05 = 'MAY'
                             06 = 'JUNE'
                             07 = 'JULY'
                             08 = 'AUGUST'
                             09 = 'SEPTEMBER'
                             10 = 'OCTOBER'
                             11 = 'NOVEMBER'
                                 ELSE 'DECEMBER'
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      6
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - LOAD


          CALL NAME = CARDFILE CUSTOMER/SALESMAN DATA FILE

                      *** MEMBER UPDATED ***


          INPT:    FILE   CARD
                   DEFINE NAME                1 TO  2 X
                   DEFINE CUSTOMER-NUMBER     3 TO  5 X  'CUSTOMER'  'NUMBER'
                   DEFINE CITY                      6 X
                   DEFINE STATE               7 TO  8 X
                   DEFINE ZIP-CODE            9 TO 13 X  'ZIP'  'CODE'
                   DEFINE CUSTOMER-SALE-ID   14 TO 15 X  'SALESMAN'  'ID'
                   DEFINE CREDIT-LIMIT       16 TO 23 N2 'CREDIT'  'LIMIT'
                                                          PIC '$$$,$$9.99'
                   DEFINE CURRENT-BALANCE    24 TO 30 N2 'CURRENT'  'BALANCE'
                                                          PIC '$$,$$9.99'
                   DEFINE SALESMAN-NAME      32 TO 39 X  'SALESMAN'  'NAME'
                   DEFINE SALESMAN-ID        40 TO 41 X  'SALESMAN'  '   ID   '
                   DEFINE YTD-SALES          42 TO 49 N2 'YEAR-TO-DATE'  'SALES'
                                                          PIC '$$$,$$9.99'
                   DEFINE BRANCH-ID          50 TO 52 X  'BRANCH'  '   ID   '
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      7
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - LOAD


          CALL NAME = DECODE2  STATE CODE TRANSLATION TABLE

                      *** MEMBER UPDATED ***


                   DECODE  :01 INTO :02
                           'TX' EQ 'TEXAS'
                           'TN' EQ 'TENNESSEE'
                           'NC' EQ 'NORTH CAROLINA'
                           'GA' EQ 'GEORGIA'
                           'OK' EQ 'OKLAHOMA'
                           'NY' EQ 'NEW YORK'
                           'OH' EQ 'OHIO'
                           'NJ' EQ 'NEW JERSEY'
                           'CT' EQ 'CONNECTICUT'
                           'IL' EQ 'ILLINOIS'
                           'PN' EQ 'PENNSYLVANIA'
                           'MI' EQ 'MICHIGAN'
                           'CO' EQ 'COLORADO'
                           'WA' EQ 'WASHINGTON'
                           'CA' EQ 'CALIFORNIA'
                                    ELSE  'UNKNOWN'   'STATE'  'NAME'
```

```
CA                                       16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                PAGE      8
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - LOAD


          CALL NAME = DECODE3  CUSTOMER NAME TRANSLATION TABLE

                      *** MEMBER UPDATED ***


                   DECODE :01 INTO :02
                           '01' EQ 'HIGH-ROLLING INVESTMENT '
                           '02' EQ 'SOUTHERN FRIED FOODS    '
                           '03' EQ 'LEGAL TOBACCO CO.       '
                           '04' EQ 'SOUTHERN PINE INDUSTRIES'
                           '05' EQ 'BARONIAL OIL CO.        '
                           '06' EQ 'COSMOPOLITAN FASHIONS   '
                           '07' EQ 'HEAVY METAL MACHINERY   '
                           '08' EQ 'AIRPORT SERVICES CORP.  '
                           '09' EQ 'STOLID INSURANCE CORP.  '
                           '10' EQ 'INLAND GRAIN TERMINALS  '
                           '11' EQ 'STEEL CURTAIN STEEL INC.'
                           '12' EQ 'PERFECT BEARING CORP.   '
                           '13' EQ 'MOUNTAIN STATES MINING  '
                           '14' EQ 'NORTHWEST PLYWOOD MILLS '
                           '15' EQ 'ORIENTAL TRADING CO.    '
                           '16' EQ 'WEST COAST LIFESTYLES   '
                                    OTHERWISE ' '  'NAME'
```

```
CA                                       16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                PAGE      9
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - LOAD


          CALL NAME = DECODE4  CITY TRANSLATION TABLE

                      *** MEMBER UPDATED ***


                   DECODE :01 INTO :02
                           'A' EQ 'DALLAS'
                           'B' EQ 'MEMPHIS'
                           'C' EQ 'CHARLOTTE'
                           'D' EQ 'ATLANTA'
                           'E' EQ 'OKLAHOMA CITY'
                           'F' EQ 'NEW YORK'
                           'G' EQ 'CLEVELAND'
                           'H' EQ 'NEWARK'
                           'I' EQ 'HARTFORD'
                           'J' EQ 'CHICAGO'
                           'K' EQ 'PITTSBURGH'
                           'L' EQ 'DETROIT'
                           'M' EQ 'DENVER'
                           'N' EQ 'SEATTLE'
                           'O' EQ 'SAN FRANCISCO'
                           'P' EQ 'LOS ANGELES'    ELSE ' '  'CITY'
```

```
CA                                         16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                  PAGE     10
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - DISPLAY   INDEX

                  ┌─────────┐
               1  | JUNKG   |
               2  | JUNKH   |
               3  | JUNKI   |   COMMENT-C
               4  | OFFDECOD|
               5  | TRANFILE|CHECK REGISTER TRANSACTION FILE
               6  | PMF010  |DATABASE PERSONEL MASTER FILE
               7  | PAY010  |DATABASE PAYROLL FILE
               8  | DECODE1 |ALPHANUMERIC MONTH TRANSLATION TABLE
               9  | CARDFILE|CUSTOMER/SALESMAN DATA FILE
              10  | DECODE2 |STATE CODE TRANSLATION TABLE
              11  | DECODE3 |CUSTOMER NAME TRANSLATION TABLE
              12  | DECODE4 |CITY TRANSLATION TABLE
                  └─────────┘
                       ▲
          Members are displayed in the same sequence they were added to the library.
```

```
CA                                         16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                  PAGE     11
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - DISPLAY   DECODE4


          CALL NAME = DECODE4


                   DECODE :01 INTO :02
                           'A' EQ 'DALLAS'
                           'B' EQ 'MEMPHIS'
                           'C' EQ 'CHARLOTTE'
                           'D' EQ 'ATLANTA'
                           'E' EQ 'OKLAHOMA CITY'
                           'F' EQ 'NEW YORK'
                           'G' EQ 'CLEVELAND'
                           'H' EQ 'NEWARK'
                           'I' EQ 'HARTFORD'
                           'J' EQ 'CHICAGO'
                           'K' EQ 'PITTSBURGH'
                           'L' EQ 'DETROIT'
                           'M' EQ 'DENVER'
                           'N' EQ 'SEATTLE'
                           'O' EQ 'SAN FRANCISCO'
                           'P' EQ 'LOS ANGELES'    ELSE ' '  'CITY'
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE     12
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - DISPLAY   DECODE3


          CALL NAME = DECODE3


                   DECODE :01 INTO :02
                           '01' EQ 'HIGH-ROLLING INVESTMENT '
                           '02' EQ 'SOUTHERN FRIED FOODS    '
                           '03' EQ 'LEGAL TOBACCO CO.       '
                           '04' EQ 'SOUTHERN PINE INDUSTRIES'
                           '05' EQ 'BARONIAL OIL CO.        '
                           '06' EQ 'COSMOPOLITAN FASHIONS   '
                           '07' EQ 'HEAVY METAL MACHINERY   '
                           '08' EQ 'AIRPORT SERVICES CORP.  '
                           '09' EQ 'STOLID INSURANCE CORP.  '
                           '10' EQ 'INLAND GRAIN TERMINALS  '
                           '11' EQ 'STEEL CURTAIN STEEL INC.'
                           '12' EQ 'PERFECT BEARING CORP.   '
                           '13' EQ 'MOUNTAIN STATES MINING  '
                           '14' EQ 'NORTHWEST PLYWOOD MILLS '
                           '15' EQ 'ORIENTAL TRADING CO.    '
                           '16' EQ 'WEST COAST LIFESTYLES   '
                                    OTHERWISE ' '  'NAME'
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE     13
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - DISPLAY   DECODE2


          CALL NAME = DECODE2


                   DECODE  :01 INTO :02
                           'TX' EQ 'TEXAS'
                           'TN' EQ 'TENNESSEE'
                           'NC' EQ 'NORTH CAROLINA'
                           'GA' EQ 'GEORGIA'
                           'OK' EQ 'OKLAHOMA'
                           'NY' EQ 'NEW YORK'
                           'OH' EQ 'OHIO'
                           'NJ' EQ 'NEW JERSEY'
                           'CT' EQ 'CONNECTICUT'
                           'IL' EQ 'ILLINOIS'
                           'PN' EQ 'PENNSYLVANIA'
                           'MI' EQ 'MICHIGAN'
                           'CO' EQ 'COLORADO'
                           'WA' EQ 'WASHINGTON'
                           'CA' EQ 'CALIFORNIA'
                                    ELSE  'UNKNOWN'   'STATE'  'NAME'
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE     14
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - DISPLAY   DECODE1


          CALL NAME = DECODE1


                   DECODE  :01 INTO :02
                             01 = 'JANUARY'
                             02 = 'FEBRUARY'
                             03 = 'MARCH'
                             04 = 'APRIL'
                             05 = 'MAY'
                             06 = 'JUNE'
                             07 = 'JULY'
                             08 = 'AUGUST'
                             09 = 'SEPTEMBER'
                             10 = 'OCTOBER'
                             11 = 'NOVEMBER'
                                 ELSE 'DECEMBER'
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE     15
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - DISPLAY   CARDFILE


          CALL NAME = CARDFILE


          INPT:    FILE   CARD
                   DEFINE NAME                 1 TO  2 X
                   DEFINE CUSTOMER-NUMBER      3 TO  5 X  'CUSTOMER'  'NUMBER'
                   DEFINE CITY                       6 X
                   DEFINE STATE                7 TO  8 X
                   DEFINE ZIP-CODE             9 TO 13 X  'ZIP'  'CODE'
                   DEFINE CUSTOMER-SALE-ID    14 TO 15 X  'SALESMAN'  'ID'
                   DEFINE CREDIT-LIMIT        16 TO 23 N2 'CREDIT'  'LIMIT'
                                                       PIC '$$$,$$9.99'
                   DEFINE CURRENT-BALANCE     24 TO 30 N2 'CURRENT'  'BALANCE'
                                                       PIC '$$,$$9.99'
                   DEFINE SALESMAN-NAME       32 TO 39 X  'SALESMAN'  'NAME'
                   DEFINE SALESMAN-ID         40 TO 41 X  'SALESMAN'  '   ID   '
                   DEFINE YTD-SALES           42 TO 49 N2 'YEAR-TO-DATE'  'SALES'
                                                       PIC '$$$,$$9.99'
                   DEFINE BRANCH-ID           50 TO 52 X  'BRANCH'  '   ID   '
```

```
CA                                        16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE     16
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - DISPLAY   PMF010


          CALL NAME = PMF010


          PMF010:  FILE    DATACOM    :01     RECORD=329
                   DEFINE  DB-COMMAND            1-5  X
                   DEFINE  DB-KEY-NAME          6-10  X
                   DEFINE  DB-ELEMENT-LIST    191-201  X
                   DEFINE  EMPLOYEE-NUMBER    301-305  X 'EMPLOYEE' ' NUMBER '
                   DEFINE  NAME               306-329  X
```

```
CA                                       16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                PAGE     17
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - DISPLAY   PAY010


          CALL NAME = PAY010


          PAY010:  FILE    DATACOM    :01     RECORD=340
                   DEFINE  DB-COMMAND            1-5  X
                   DEFINE  DB-KEY-NAME           6-10  X
                   DEFINE  DB-KEY-VALUE         11-15  X
                   DEFINE  DB-ELEMENT-LIST    191-201  X
                   DEFINE  EMPLOYEE-CODE          306  X
                   DEFINE  STATUS                 307  X
                   DEFINE  CURRENT-RATE       308-315  N3 ' CURRENT ' 'PAY  RATE'
                   DEFINE  YTD-WAGES          316-323  N2
                           'YEAR-TO-DATE'  '   WAGES'  PIC  '$$$,$$9.99'
                   DEFINE  YTD-COMMISSION     324-331  N2
                           'YEAR-TO-DATE'  ' COMMISSION '
                   DEFINE  YTD-TAXES          332-339  N2
                           'YEAR-TO-DATE'  '   TAXES'
```

```
CA                                       16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                PAGE     18
 DRLIB  - REPORTING FACILITY LIBRARY FILE     MAINTENANCE - DISPLAY   TRANFILE


          CALL NAME = TRANFILE


          TRANS:   FILE    CARD
                   DEF     TRANSACTION-CODE    1 THRU  2  X ' '
                   DEF     ID-NUMBER           3 THRU  6  X ' '
                   DEF     AMOUNT              7 THRU 13  N2
                   DEF     ACTIVITY-DATE      14 THRU 21  X 'DATE  OF' 'ACTIVITY'
                   DEF     CLEARED                    22  X
                   DEF     PAYEE              23 THRU 42  X
                   DEF     REASON             44 THRU 75  X
```

```
CA                                       16:28:10  24 JUN 2010
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                PAGE     19
 DRLIB  - REPORTING FACILITY LIBRARY FILE


          REPORTING FACILITY LIBRARY FILE CONDENSE - COMPLETE



LIBRARY MAINTENANCE COMPLETE
```

Dynamic System Tables Sample

```
CA                                        16:28:10  24 JUN 2011
CA DATACOM/DB REPORTING FACILITY
OS  VERSION nn.n                                 PAGE      1
USER 'CA DYNAMIC SYSTEM TABLES'
NOTE SAMPLE REPORT OF TABLES AND INDEXES LOADED.
DRT1000: INPUT DATACOM RECORD=625 NAME=DRT

                :  DEFINE  DB-COMMAND            1-5  X
                   DEFINE  DB-KEY-NAME           6-10  X
                   DEFINE  DB-KEY-VALUE         11-23  X
                   DEFINE  KEYDIR_NAME          11-18  X
                   DEFINE  KEYDBID              19-20  B
                   DEFINE  KEYAREA_NAME         21-23  X
                   DEFINE  DB-ELEMENT-LIST     191-201 X
                   COPYDD DIR_TABLE ELEMENT
                   DEFINE RDDRT(1)='Y'
                   GOTO REDNDRT (RDDRT NE 'Y')
                   MOVE 'REDKG'   TO DRT1000.DB-COMMAND
                   MOVE 'N' TP RDDRT
                   MOVE LOW-VALUE TO DRT1000.DB-KEY-VALUE
                   GOTO REDDRT
                REDNDRT:CONT
                   MOVE 'REDNX'   TO DRT1000.DB-COMMAND
                REDDRT:CONT
                   MOVE '-#DRT' TO DRT1000.DB-KEY-NAME
                   MOVE '-#DRT' TO DRT1000.DB-ELEMENT-LIST
                   GET  DRT1000
```

Chapter 13: Coding Specialized Routines

When some of the standard operating facilities of the Reporting Facility do not provide the appropriate methods to process a specific request, you can create your own specialized routine to be used in place of the standard function.

Two functional areas currently support this type of user interaction:

- Input file access
- Report printing

User-Modules

This section describes the coding requirements of a user-supplied File Access Module (user-module) on the FILE command.

You can write user-modules in either Assembler or COBOL. Follow standard IBM linkage conventions when writing a user-module. Each module must save the general registers at entry and restore them at exit. In COBOL, the compiler provides these functions when ENTER LINKAGE, ENTRY, or RETURN is specified. In Assembler, use the SAVE and RETURN macros.

You must link edit the user-module and catalog it to a system Load Library with a unique phase name assigned to it. This phase name is then specified in the Reporting Facility program on the FILE command associated with the input file.

The system calls the user-module once for each GET command, or (in the case of the primary file) once per cycle. On return from the user-module, the Reporting Facility expects a single logical record to be made available for processing. The user-module must assume complete responsibility for the integrity of the file, including its opening and closing.

Two-way communication between the Reporting Facility and the user-module is established by the 80-byte Communications Area set up when the file is defined. On entry to the user-module from the Reporting Facility, register 1 points to a parameter list containing five fullwords. Each parameter is designed for a specific purpose and must be used accordingly.

A(I/O Area)

Is the address of the I/O area, where the user-module is expected to place a record for processing. This area is as large as specified in the RECORD= parameter of the FILE command.

A(Comm Area)

Is the address of the 80-byte Communications Area automatically allocated when the file is defined. This is the area to be used for communication between the Reporting Facility and the user-module.

A(filename)

Is the address of the file name, as specified on the FILE command, padded to a length of eight characters with blanks. If multiple files are read using the same user-module, this parameter enables the user-module to determine which file name was coded on the GET command for the current call.

Original R1

Contains the original contents of register 1. On initial entry to the Reporting Facility, this is required to process IMS-DL/I files.

A(recordlength)

Is the address of a two-byte binary record length field in the GSA that contains the value specified in the RECORD= parameter of the FILE command for this file. Use this parameter to edit against the size of a data record to avoid a core overlay condition.

Examples

The following example illustrates the linkage conventions to follow when coding a user-module in Assembler language:

```
csect  START 0
        SAVE (14,12)                 Store callers registers
       BALR  11,0                   Set up base addressability
       USING *,11
       ST    13,ST13                Store savearea pointer
       LM    2,6,0(1)               Load RPT Facility parameters
       .
       .                            Program body
       .
       L     13,ST13                Restore savearea address
       RETURN (14,12)               Restore callers registers and return
       .
       .
       .
       ST13 DS F
       .
       .
       END
```

The following example illustrates the linkage conventions for coding a user-module in COBOL:

```
IDENTIFICATION DIVISION
 PROGRAM-ID.      'modulename'
  .
  .
DATA DIVISION
  .
  .
LINKAGE SECTION.
01  IO-BUFFER
                      PIC X(nnn).      ** RECORD=nnn **
01  COMMUNICATION-AREA
                      PIC X(80).
01  FILE-NAME
                      PIC X(4).
01  DUMMY
                      PIC X(4).
01  RECORD-LENGTH
                      PIC 99 COMP.
  .
  .
PROCEDURE DIVISION.
ENTER LINKAGE.
ENTRY 'modulename' USING IO-BUFFER
                        COMMUNICATION-AREA
                        FILE-NAME
                        DUMMY
                        RECORD-LENGTH
ENTER COBOL.
  .
  .
ENTER LINKAGE.
RETURN.
ENTER COBOL.
```

Note the following:

- When operating in a VS or z/VSE environment, the user-module should be link edited to the Core Image Library specifying a phase origin of * (asterisk). However, one module in a run of the Reporting Facility cannot be coded as self-relocating. If you do have one that is not self-relocating, you must catalog it immediately following the Reporting Facility root phase.
- When OPTION MAP is specified, the size of the root phase prints on the last line of the compile diagnostics. Link your non-self relocating module as PLUS that number.
- You can load multiple copies of a user-module during a single run of the Reporting Facility; each is processed separately. You can accomplish this by specifying multiple FILE commands that have the same module name, but a unique file name.

PRINT-EXIT Subroutines

You may need to override the standard printing techniques of the Reporting Facility. The compile listing is always printed in a standard manner, but you can override the printing of the reports and the run diagnostics on a per run basis.

An OPTION PRTEXIT=phasename command indicates that instead of printing the reports and run diagnostics, the Reporting Facility is to pass each print line to a user-supplied routine with the cataloged phase name.

You must write this routine in Assembler language and follow standard IBM linkage conventions. The routine should save the general registers on entry and restore them on exit in much the same manner as with user-modules. This routine loads automatically just prior to printing the reports.

On entry to the program, register 1 points to a single address of a 148-byte record. The content of the record is as follows:

1

Type code

2

CCW code

3—4

Printer width

5—148

Data

type code

Is a one-byte hexadecimal code denoting the type of data contained in the record. Valid values are:

X'F0'

Indicates that subsequent data is part of the run diagnostics

X'01'-X'3F'

Indicates that subsequent data is part of reports 1—63, respectively

X'FF'

Indicates END-OF-FILE and that no more data is to print

CCW code

Is a one-byte hexadecimal field containing the CCW code for the IBM printer carriage control-character. Valid values are:

X'0B'

Space one line immediately

X'13'

Space two lines immediately

X'1B'

Space three lines immediately

X'01'

Write, no space

X'09'

Write, space one line after print

X'11'

Write, space two lines after print

X'19'

Write, space three lines after print

X'8B'

Skip to channel one immediately

printer width

Is a two-byte constant (halfword), indicating the printer width as defined at the time of the Reporting Facility installation, or overridden for this run by specification of the OPTION PRINTER= parameter. The maximum value of the constant contained in this field is 144.

data

Represents the print line to be printed. The length of this field is dependent on the contents of the PRINTER WIDTH field. This is a print line image and, therefore, contains the data to be printed in its final edited form.

Note the following:

- When operating in a VS or z/VSE environment, the print-exit module should be link edited to the Core Image Library specifying * (asterisk) as a phase origin.
- Once OPTION PRTEXIT= has been specified, all reports in the current run print under the control of the print-exit subroutine. It is not possible to mix this facility with the Standard Printing Facility.

Executing the Reporting Facility

This section describes the resource requirements needed for successful execution of the Reporting Facility system. The system provides many specialized facilities; therefore, proper and efficient use of the resources available is an important consideration.

The Reporting Facility runs in z/OS, VS, and z/VSE environments. Other minimum requirements that apply are:

- IBM 370 architecture
- Decimal arithmetic feature
- 3350, 3375, 3380, 3390, 9345, or FBA type disk drive
- 1400, 3200, or 3800 series printer
- IBM Sort/Merge program, or compatible sort program cataloged as "Sort"
- Minimum region/partition size of 80K
- Operating system generated with user program check exits:
  - STXIT for z/VSE
  - STAE/SPIE for z/OS

The Reporting Facility executes in multiple types of environments with varying configurations.

Data Set Utilization

The Reporting Facility requires that several data sets be made available, through JCL, to process specific requests. Some of these data sets are always required and some are optional, depending on the function to be performed.

The following is a summary of each data set potentially required for execution of the Reporting Facility. The names stated below represent the names that must appear in the appropriate JCL commands:

- DRLIB (optional)
- DROUT (optional)
- DRSNAP (optional)
- DRTRACE (optional)
- DRWORK (required)
- SORTIN/SORTIN1 (optional)

- SORTLIB (optional)
- SORTOUT (optional)
- SORTWK01-06/SORTWK1-8 (optional)
- SYSIN/SYSIPT (required)
- SYSPRINT/SYSLST (required)
- SYSUDUMP (optional)

DRLIB

Defines the Reporting Facility call library. This file is optional, except:

- When Library Maintenance mode is invoked
- When a CALL command appears in the Reporting Facility program

This is a direct-access disk file and you must format and load it with data prior to its use. The file can be as large as necessary to contain all of the source members to be cataloged, but should occupy at least one full cylinder. The physical type of device on which this library file resides in a single run of the Reporting Facility is determined by either the installation default value or the OPTION DISK= override parameter.

Take extreme care when modifying the default physical device type of the Reporting Facility disk files. Protect this file from concurrent update or access to avoid potential destruction of the source library.

DROUT

Specifies an optional output file to be created rather than a report generated. The sole purpose of invoking the Reporting Facility in the Write-Only mode is to create a single, sequential, fixed-length output file. When you invoke Write-Only mode, all output is directed to this file.

This file varies in size, depending on the size and number of output records to be created by the program. When the output file is a disk file, the physical type of device on which it is to reside is determined either by the installation default value or by the OPTION DISK= override parameter.

DRSNAP

Is an optional data set used in conjunction with OPTION SNAP. When used in a z/OS environment, this is a standard SYSOUT type data set, that is, the output is assumed to be directed to a system printer. When used in a z/VSE environment, this is the sequential disk file containing print line images of the output as a result of OPTION SNAP.

The content of this file then prints separately according to the specifications of a utility function you prepared. The installation default value of the OPTION DISK= override parameter determines the physical type of device on which the data set resides.

DRTRACE

Is an optional data set used in conjunction with OPTION DUMP. When used in a z/OS environment, this is a standard SYSOUT type data set, that is, the output is assumed to be directed to a system printer.

The content of this file then prints separately according to the specifications of a utility function prepared by you. The device type on which the data set resides is determined either by the installation default value or by the OPTION DISK= override parameter.

DRWORK

Is a required data set used as a work file during the compilation of the Reporting Facility program. The device type on which it resides is determined by the installation default value. It is not possible to temporarily override the device type for this file on a per run basis. Make a valid DRWORK disk file available for every execution of the Reporting Facility, regardless of the mode of operation selected, in both z/OS and z/VSE environments.

The Reporting Facility uses this file to temporarily store volatile internal control blocks and the generated code to read input files and produce reports. The file can be a temporary file; you need to maintain it only for the current run. Define it as a direct-access disk file.

The size of the file varies based on the size of the Reporting Facility program to be compiled, the number of input files to be read, and the number of reports to be produced. The minimum suggested allocation is between 10 and 20 tracks, or 200 to 300 blocks if the file resides on an FBA device.

SORTIN/SORTIN1

Defines the Reporting Facility Hit File and is required only when executing in the Primary mode. SORTIN must be used in a z/OS environment, SORTIN1 in a z/VSE environment. The system creates this file during the record selection phase of the Reporting Facility. The file contains only the fields from the input files necessary to produce the final reports.

This file varies in size, depending on the size and number of reports to be produced and the size of the largest sort key required for a particular report. It is required for OPTION XREF.

SORTLIB

Is required in a z/OS environment only when executing in the Primary mode. This data set specifies the library, usually SYS1.SORTLIB, that contains the stand-alone sort program modules.

SORTOUT

Defines the temporary output file from the sort, and is required only when executing in the Primary mode. This file is a sequential disk data set with a variable blocked format. It must have the same physical characteristics as the SORTIN or SORTIN1 file.

Most sort packages allow the same physical disk extent to be used for both SORTIN and SORTOUT. This method of invoking an in-place sort is highly recommended to both increase the efficiency of the sort and conserve resources.

SORTWK01-06/ SORTWK1-8

Specifies multiple temporary work files for the sort and is, therefore, required only when executing in the Primary mode. You can specify from one to six work files in a z/OS environment and from one to eight in a z/VSE environment. The number of work files in a particular run is controlled by either the installation default value or the OPTION SORT= override parameter.

SYSIN/SYSIPT

Specifies the data set from which all of the Reporting Facility commands are read. This file is required for any execution of the Reporting Facility, regardless of intent. SYSIN must be specified in a z/OS environment; however, in a z/VSE environment, the command automatically reads from SYSIPT by placing them immediately behind the EXEC statement.

SYSPRINT/ SYSLST

Specifies the data set to which all printed output is directed. This includes both the compiler source listing and the output report. z/OS environments must specify SYSPRINT; however, SYSLST is usually handled automatically in a z/VSE environment.

SYSUDUMP

Specifies, in a z/OS environment only, the data set to which the printing of a formatted dump is directed if the Reporting Facility recognizes an abend condition that warrants immediate termination of the run. This data set is optional.

Core Storage Requirements

The core storage requirements for a particular run must be examined on an individual run basis. However, you must meet some specific minimum requirements. For example, each partition/region in which the Reporting Facility is to be executed should contain at least 80KB bytes of real storage. In addition, at least 48KB bytes of GETMAIN/GETVIS storage must be available. Any additional core requirements depend entirely on the types of facilities requested.

Some of the questions you might ask yourself when trying to determine the resource requirements of a particular run are:

- How many input files are defined, and how big are their associated records?
- How many reports are being printed?
- Are CA Datacom/DB tables being referenced?
- Is CA Datacom Datadictionary necessary? Does a COPYDD command appear in the program?
- Are VSAM data sets being referenced?
- Is an IMS-DL/I database being referenced?
- Are large arrays defined or are an abnormal number of numeric constants, numeric edit patterns, or alphanumeric literals coded?
- Is a sort required (a sort requires a minimum of 20KB bytes)?

Based on the answers to the above questions, you can make a reasonable assessment of the core storage requirements for each particular run.

Debugging Aids

If error conditions require the aid of a CA Support Specialist, the Reporting Facility provides debugging tools to assist in locating these errors and providing fast and efficient fixes. The output of these tools are not designed for your use, but rather for the use of the CA Support Specialist.

The specification of a special OPTION command or JCL initiates the debugging aids. There are three separate tools currently available.

TRACE

Provides a system trace of all activity performed during the compilation of the source statements and subsequent machine code generation.

SNAP

Provides options for printing critical information when program check interruptions are detected during the program execution.

DUMP

Provides a detailed analysis of all activity to and from the internal DRWORK file.

Trace Facility

The TRACE facility is the most commonly requested debugging aid. This tool allows very close inspection of the compilation processes that are performed and the machine code that is subsequently generated.

Three options are available for initiating the TRACE facility.

- OPTION command
- PARM= parameter of the JOB card (in a z/OS environment)
- UPSI control statement (in a z/VSE environment)

To initiate the Trace facility, code the OPTION command as follows:

```
►►─ OPTION ─ TRACE ─┬─ OFF ─────┬──────────────────────────────────────────►◄
                    └─ phaseid ─┘
```

OFF

Specifies that the TRACE facility is to be immediately suspended. Since the output of the facility can be quite large, it is very useful to force the tracing of control blocks associated with a particular command or group of commands.

phaseid

Indicates that a specific processing phase is to be traced. Valid values are A, C, G, H, or I. Trace multiple phases by coding multiple phaseids.

The purpose of each phase is as follows:

A

Compilation phase

H

Generation of record selection logic machine code

C

Execution of record selection logic

I

Generation of report printing logic machine code

G

Execution of report printing logic

When OPTION TRACE is coded without OFF or *phaseid*, all phases will be traced.

To invoke the TRACE facility in a z/OS environment using JCL specifications, code the following statement:

```
// EXEC PGM=DRREPORT,PARM='TRACE'
```

This method does not allow specification of the *phaseid*, as in the OPTION TRACE method.

Prior to the // EXEC command (in a z/VSE environment), you can place the following JCL command to invoke the TRACE method:

```
// UPSI 10000000
```

During compilation, the TRACE facility prints dashed lines, one at the beginning and one at the end, of each separate command. This is an attempt to show the activity being performed relating to each specific command encountered in the program.

Many of the items printed can be directly related to the compile diagnostics section, printed in conjunction with OPTION MAP. This includes:

- Internal FST entry creation and modification
- FNT entry activity
- Adding of literals and constants to the GSA
- Information about the program performing the indicated process

This information helps the CA Support Specialist determine the processing sequence being performed and the program performing it. The format of the TRACE command changes significantly when code generation routines are executed.

The FST traces activity as the FST entries are reread and used in the creation of machine code. In addition, the code generated prints, followed by the ID of the program responsible for creating the code segment.

Snap Facility

Use the SNAP facility when the Reporting Facility detects a program check interruption during the execution of the generated machine code. To initiate the SNAP facility, use the following command:

```
►►─ OPTION ─ SNAP ─ n ─ m ──────────────────────────────────────────►◄
```

n

Specifies the number of the program check to print, (2 will print every other SNAP and 10 will print every tenth SNAP).

m

Specifies the sequential number of the program check to abend and terminates processing with a full system dump.

The SNAP routines output the following information when a program check occurs. This information can be used to determine if the problem is with the data or is an internal software problem.

- PSW expanded to show components
- 16 general registers
- Save area register stack
- Linkage section for main program
- Linkage section for control program
- GSA
- I/O area for user exit
- Standard I/O buffer area
- Generated code for this phase

When used in a z/OS environment, the system directs output from the SNAP facility to the DRSNAP data set, which the system assumes is a system printer. For example:

```
//DRSNAP DD SYSOUT=*
```

In a z/VSE environment, the system directs the output to the DRSNAP data set, which it assumes is a sequential data set. This sequential file contains 132-byte records and is blocked at 1320. Each record consists of a single standard carriage control character, and a 131-byte print line image.

You can then read this file and print it with a special utility program. For example:

```
// DLBL DRSNAP,'datasetname',0,SD
// EXTENT SYSnnn,1,0,nnnn,nnn
```

The Reporting Facility can read this file and will print it.

Dump Facility

The DUMP facility traces the activity to or from the DRWORK file. The system places several different types of information on this file during the execution of the Reporting Facility. They include:

- Contents of the internally generated FST area
- All of the generated machine code

You can invoke the DUMP facility using either a special OPTION command in the Reporting Facility program or by standard JCL specifications.

```
►►─ OPTION ─ DUMP ─┬──────────────┬──────────────────────────────────►◄
                   └ control-word ┘
```

control-word

Optionally use one of the following control words with this command:

DRREAD

Specifies that only reads to the DRWORK file are to cause formatted output by the DUMP routines.

DRWRITE

Specifies that only writes to the DRWORK file are to cause formatted output by the DUMP routines.

DRADD

Specifies that output is to be formatted only when data is being added to the DRWORK file.

DRWORK <ID>

Specifies that all reads, writes, and additions to the DRWORK file are to be reported. The ID parameter denotes the specific phases of the program in which data is to be formatted.

This parameter is optional, but, when coded, must be coded in the same manner as with the OPTION TRACE phaseid parameter.

In a z/OS environment, use the PARM= parameter of the EXEC command to initiate the DUMP facility. For example:

```
 // EXEC PGM=DRREPORT,PARM='DUMP,xxxxxxx,yyyyy'
```

,xxxxxxx

Is one of the four options discussed above (DRADD, DRREAD, DRWRITE, or DRWORK).

,yyyyy

Is one or more of the optional phaseid parameters.

In a z/VSE environment, use the UPSI command as follows:

```
 // UPSI 01234567
```

bits 0-2

Correspond to DRWRITE, DRREAD, and DRADD, respectively

bits 3-7

Correspond to phases A, C, G, H, and I, respectively

Modifying the System

The CA Datacom/DB Reporting Facility, which is a part of CA Datacom/DB, is installed with the product. Information about the installation process is contained in the CA Datacom installation manual for your environment.

After the Reporting Facility is generated in the receive libraries, review the system to determine if any modifications are necessary to accommodate your total environment.

Note: The installation output can also be helpful in the modification process.

Once you have modified your system, it can again be demonstrated with the technique used to demonstrate the initial installation.

The following steps are involved in modifying the Reporting Facility:

- Code the appropriate values for the parameters of the IS08Z09I macro.
- Assemble IS08Z09I.
- Link edit the resulting output.

IS08Z09I Macro

In coding the IS08Z09I macro, standard Assembly rules for macros apply. For example, a statement to be continued on the next card must have a non-blank character in column 72, and the next card must begin in column 16. The following, in alphabetic sequence, are the IS08Z09I macro parameters.

ASAFMT=

Specifies the default carriage control type for printer output. Specify an A for ASA or an M for machine.

Valid Entries:

A or M

Default Value:

M

DATE=

Specifies the printing of the calendar date on the compile listing and on all generated reports. Three formats are available: MMDDYYYY, DDMMYYYY, and DDMMMYYYY.

Valid Entries:

M (for MMDDYYYY), D (for DDMMYYYY), or A (for DDMMMYYYY)

Default Value:

A

DBCOMM=

Specifies whether you allow the DBCOMM=AUTO option at your site. (See the OPTION command.) NOAUTO coded here disallows it.

Valid Entries:

AUTO or NOAUTO

Default Value:

AUTO

DBGETBLK=

Specifies the size of the GETBLK buffer (works similarly to the GETBLK option in the DBURTBL macro as described in the *CA Datacom/DB Database and System Administration Guide*).

Valid Entries:

Integer value 0 or 4096 through 61440

Default Value:

4096

DBPRI=

Specifies the priority of the automatically generated URT.

Valid Entries:

1—15.

Default Value:

3

DBSEQBUF= or DBSEQBUFS=

Specifies the number of sequential buffers (works similarly to the SEQBUFS option in the DBURTBL macro as described in the *CA Datacom/DB Database and System Administration Guide*).

Valid Entries:

2 through 128 or -2 through -128

Default Value:

2

DELPASS=

Specifies an optional password that will be required on the USER statement when deleting call library members.

Valid Entries:

1—8 characters

Default Value:

Null

DISK=

Specifies the device type used by the Reporting Facility. The Reporting Facility call library and system work files are assumed to reside on the device type selected.

Valid Entries:

3350, 3375, 3380, 3390, 9345, or FBA (FBA for z/VSE only)

Default Value:

3380

DL1=

Specifies whether DL/I database support is to be used.

Valid Entries:

YES, NO, or ENTRY (ENTRY for z/VSE DL/I-ENTRY support only)

Default Value:

NO

LANG=

Specifies the language to be used for the Reporting Facility report literals such as PAGE, GRAND TOTAL, END OF REPORT, and so forth.

Valid Entries:

DANISH, ENGLISH, FINNISH, FRENCH, GERMAN, GERMAN2, ITALIAN, PORTUGUESE, NORWEGIAN, SPANISH, or SWEDISH

Default Value:

ENGLISH

MAXSZE=

(z/VSE users with VS-type systems only.) Specifies the maximum storage size (in K) to be used for the sort function. If omitted, the sort is not limited in its acquisition of core.

Valid Entries:

Any 3-digit number from 000 to 999

Default Value:

000

NAME=

Designates a 1- to 48-character alphanumeric literal enclosed in apostrophes that specifies the company name to appear on the first page of every Reporting Facility run.

Valid Entries:

Any alphanumeric literal enclosed in apostrophes

Default Value:

Null

PAGE=

Specifies the maximum number of lines to be printed on each page of a report.

Valid Entries:

Any 2-digit number from 08 to 88

Default Value:

60

PRINTER=

Specifies the width (in print positions) of the printer. The Reporting Facility uses this value to calculate interfield spacing within generated reports.

Valid Entries:

Any 3-digit number from 020 to 144

Default Value:

132

SECCODE=

Specifies the required eight-byte call library password. The Reporting Facility uses this password to verify access during Library Maintenance CREATE commands.

Valid Entries:

Any 8 non-blank characters

Default Value:

DR-ADM

SYSTEM=

Specifies the operating environment in which the Reporting Facility executes.

Valid Entries:

OS or DOS

Default Value:

Null

System Option Program Assembly

Assemble a program containing the options specified in the parameters of IS08Z09I:

z/OS Example

```
// ...   JOB ...
//STEP1 EXEC ASMFC,PARM.ASM='NODECK,LOAD'
//SYSLIB DD DSN=MACLIB name DISP=SHR
//SYSGO  DD DSN=OBJLIB name(IS08Z09I),DISP=SHR
//SYSIN  DD *
         IS08Z09I DATE=A,DISK=3380,DL1=NO,SECCODE=CAI/DR,
               LANG=ENGLISH,NAME='XYZ COMPANY, INC.',
               PAGE=60,PRINTER=132,SYSTEM=OS
               END
/*
//
```

z/VSE Example

```
// EXEC ASSEMBLY
          PUNCH ' CATALR IS08Z09I'
         IS08Z09I DATE=A,DISK=3380,DL1=NO,SECCODE=CAI/DR,
               LANG=ENGLISH,MAXSZE=000,
               NAME='XYZ COMPANY, INC.',
               PAGE=60,PRINTER=132,SYSTEM=DOS
         END
/*
/&
```

Assembler Source Example

```
IS08Z09I DATE=A,
          DISK=3380,
         DL1=NO,
         SECCODE=CAI/DR,
         LANG=ENGLISH,
         MAXSZE=000,
         NAME='XYZ COMPANY, INC.',
         PAGE=60,
         PRINTER=132,
         SYSTEM=OS,
         END
```

Link-Edit Samples

Link edit the output of the assembly of the system option program:

z/OS Example

```
//JS1 EXEC LKED
//JKED.SYSLMOD DD DSN=....
//JKED.OBJLIB DD DSN=....
//LKED.SYSIN DD*
  INCLUDE OBJLIB(IS08Z09I)
  NAME DRQXXPR(R)
/*
//
```

z/VSE Example

```
// OPTION CATAL,NODECK,NODUMP,LOG
   ACTION NOAUTO
   PHASE DRQXXPR,+0
   INCLUDE IS08Z09I
/*
// EXEC LNKEDT
```