Interactive Query Language User's Guide

Order Number AA-H282A-TK

December 1978

ABSTRACT

The TOPS-10/TOPS-20 *Interactive Query Language (IQL)*
User's Guide describes the Interactive Query Language
Release 3.0

digital equipment corporation · maynard, massachusetts

The postage-prepaid READER'S COMMENTS form on the last page of this
document requests the user's critical evaluation to assist us in pre-
paring future documentation.

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-11
DECCOMM	DECSYSTEM-20	TMS-11
ASSIST-11	RTS-8	ITPS-10

CONTENTS

CONTENTS (CONT.)

CONTENTS (CONT.)

CONTENTS (CONT.)

FIGURES

TABLES

PREFACE

OBJECTIVES AND READER ASSUMPTIONS

The <u>TOPS-10/TOPS-20 Interactive Query Language User's Guide</u> describes the features of IQL Release 3.0, which is a full data management system with comprehensive input, update, browse, and report capabilities.

This guide describes each function in detail. However, the guide is not intended to be a teaching manual. As a prerequisite of reading this guide, it is recommended that you read the companion manual, <u>An Introduction to IQL</u>, which presents a series of example sessions showing most of the features used in this guide.

The guide assumes that you are familiar with DBMS schemas and that you know how to read a DBMS data base through FIND and GET operations of IQL.

STRUCTURE OF THE DOCUMENT

The guide contains six chapters:

Chapter 1 explains file handling by IQL. Chapter 1 should be read by everyone.

Chapter 2 explains how to access IQL and how to enter the various levels of IQL. Chapter 2 should be read by everyone.

Chapter 3 explains how to prepare a query file, which you use to generate a report. Chapter 3 should be read by everyone.

Chapter 4 explains how to create, update, and read a data file. Chapter 4 should be read by everyone.

Chapter 5 explains how to prepare a dictionary, which defines to IQL the data file or data base. The chapter should be read by the data base administrator and programmer.

Chapter 6 explains how to use IQL in a batch job. Chapter 6 should be read by the data base administrator and programmer.

ASSOCIATED DOCUMENTS

TOPS-10 Introduction to IQL
DEC-10-LIQLA-A-D

Getting Started With TOPS-10 Commands
DEC-10-OTSCA-A-D

TOPS-10 Operating System Commands Manual
AA-0916C-TB

Getting Started With TOPS-20
AA-4187C-TM

TOPS-20 User's Guide
AA-4179B-TM

TOPS-20 EDIT User's Guide
DEC-20-UEUGA-A-D

TOPS-20 EDIT Reference Manual
AA-5415A-TM

TOPS-10 Data Base Management System Programmer's Procedures Manual
AA-0901C-TB

TOPS-20 Data Base Management System Programmer's Procedures Manual
AA-4149B-TM

TOPS-10 Beginner's Guide to Multiprogram Batch
DEC-10-OMPBA-C-D

TOPS-10 Multiprogram Batch Reference Manual
DEC-10-OMBRA-A-D

TOPS-20 Getting Started With Batch
DEC-20-OBGSA-A-D

TOPS-20 BATCH Reference Manual
DEC-20-OBRMA-A-D

STANDARD SYMBOLS

The following table explains the symbols used in this guide:

Symbol	Terminal Key	Description
(CTRL)	CTRL key	The control sequences control several of the computer operations. CTRL/C means you should hold down the CTRL key and at the same time type the letter C.
(DEL)	DELETE key RUBOUT key	The rubout sequence deletes one character each time you press the key.
(ESC)	ESC key ALT key ALTMODE key PREFIX key	One purpose of the escape sequence notifies the TOPS-20 Operating System to complete the command that you began. (When you are communicating with the Operating System, use of this key does not produce a printed character.) The second purpose of the escape sequence indicates to some system programs that you finished the work. (When you communicate with system programs using the escape key sequence, the system prints a dollar sign ($) on the terminal.)
(RET)	RETURN key CR key	The carriage return sequence notifies the system that you have finished typing a line. After you press this key, the terminal printing head or cursor goes to the beginning of the next line. Unless otherwise stated, all user input must be terminated by pressing the RETURN key.
(SP)	SPACE bar	The space bar creates a blank space by moving the terminal printing head or cursor one space to the right.
(LF)	LINE FEED key LF key	The linefeed symbol indicates a place where you press the key labeled LINE FEED or LF.

GLOSSARY

Alphavariable An alphanumeric working item created by IQL while generating a report. Names of alphavariables must start with A, can be up to thirty characters long, and can contain a - z, 0 - 9, and - . IQL creates an alphavariable when it sees an item name starting with A, does not find that name in the dictionary, and determines that the information is read and written to the same item. IQL makes the alphavariable as long as necessary to contain the longest amount of information placed into it.

ASCII The American Standard Code for Information Interchange is a binary code established by the American National Standards Institute. The code as applied to DIGITAL equipment can be either six bits or seven bits and represents alphanumeric characters and special symbols such as a semicolon (;), period (.), and percent (%).

Assistance Level The initial command level in IQL. In the assistance level, you can display helping text, define and inspect dictionaries, write or store or change queries, request a report, create or update a data file, or go to another command level.

Browse A function of the immediate mode that permits you to look at information in a file but not change it; also referred to as read-only access. While browsing, you can interactively position a file, look at specific records or sets of records, and list specific items for single records or sets of records.

Constant A pure number used anywhere in an IQL command. Constants can contain decimals or can omit them. Constants can start with + or -; if there is no sign, IQL assumes the constant to be positive. It is not necessary to supply leading or trailing zeros in constants. Examples of constants are: 1, 1.2, .4567, -8.76, 00074, and 74.

Current Record If a file is positioned at a particular record, that record is called the current record. If the file is positioned at the beginning (or top), it is positioned before the first record and no current record exits. If the file is at the end (or bottom), it is positioned after the last record and no current record exits.

Data Item | An item (or field) in a record read by IQL from a data file or data base. Data items can be numeric or alphanumeric. Numeric data items can be binary or ASCII characters. Descriptions of data-items are stored by IQL in dictionaries. Data item names can be up to thirty alphanumeric characters and dashes and must start with a letter other than X.

IQL lists data-item names under the ITEM NAME column title of a dictionary display.

DBMS | The full network CODASYL-standard data base manager of Digital Equipment Corporation.

Deferred Mode | The deferred mode and the immediate mode are the two modes of IQL. In the deferred mode, IQL processes query statements to generate reports and write specific records from the data file to a new file.

Dictionary | A dictionary is a table stored by IQL that describes a data file or data base. IQL refers to dictionaries when IQL generates reports and updates or browses a data file. A dictionary stores the location of a data file or data base, as well as information about the individual data-items in the file. Data-item entries in dictionaries contain information describing how the items are stored and cosmetic information anticipating printing (such as column titles and editing pictures). Example dictionaries are shown in Appendix C.

Dictionary names must begin with an alphabetic character and can contain up to thirty alphanumeric characters and dashes.

DISK6 | DISK6 is a 6-bit ASCII code.

DISK7 | DISK7 is a 7-bit ASCII code.

Edit Level | IQL enters the edit level when you issue either the WRITE or EDIT assistance command to write or edit a query file.

FINDLIST | A FINDLIST is a qualifying argument of an immediate mode command that instructs IQL to apply the command to selective records in the data file.

Help Word | A help word is a key word that you submit to IQL for a display of explanatory help text. To display a list of the help words at your site, use HELP as an IQL assistance command.

Item | An item is a symbolic term used in this guide to indicate a data item, alphavariable, or numeric variable.

Literal | A literal is any string of characters enclosed in quotes and used by IQL exactly as you specified. You can set off a literal with either single (') or double (") quotes as long as you start and end the literal with the same quote mark. If you set off the literal with one quote mark, you can include the other quote mark in the literal.

In the special literals used to designate page headings or column titles, double slashes (//) instruct IQL to start a new line. The following lines are examples of literals.

"FEB 28"
'BOB "RED" JONES'
'LINE 1//LINE 2'

Immediate Mode   The immediate mode and the deferred mode are the two modes of IQL. In the immediate mode, IQL processes a command when you enter the command. You can display on the terminal selected items from a data file, update a data file, and change, delete, and display a dictionary.

Integer   A special constant, always written without a decimal point.

Operating System   The IQL User's Guide refers to the TOPS-10 and TOPS-20 Operating Systems as the operating system.

Password   A password is a string of up to six alphanumeric characters that are stored in encrypted form in a dictionary. As defined in a dictionary, IQL can require you to specify a password to read or change sensitive data items in a data file. You supply a password to IQL when IQL prompts you for the password or when IQL encounters an AUTHORITY statement in a query. Examples of passwords are: CAROLE, BOB, 14WX6Z, MORDOR.

Query   A query is a collection of source statements written like English language sentences that IQL uses to generate one or more reports. The statements in a query are not carried out until you use the RUN or EXECUTE assistance command. Queries can be named and stored. Query names can be up to thirty characters long, must start with an alphabetic character, and can contain alphanumeric characters and dashes.

Report Level   The report level is the level that IQL enters when you issue the RUN or EXECUTE assistance command to generate a report.

Search Path   The Operating System uses a search path when looking for a file through two or more directories. Refer to Chapter 1 to establish a search path.

String   A string is any series of characters including spaces.

Summary   A summary is a line on a report that is calculated by the TALLY, TOTAL, AVERAGE, MAXIMUM, or MININUM query statements.

Update Level   The update level is the level that IQL enters when you issue the UPDATE assistance command to IQL to read or change a data file.

Variable   A variable is a numeric working item generated by
                  IQL for reading and writing a value. Names of
                  variables must start with X or ZZ, can be up to
                  thirty characters long, and contain alphanumeric
                  characters and dashes. Variables for IQL contain
                  thirteen' integer and five decimal places.

CHAPTER 1

FUNCTIONS OF IQL

Interactive Query Language - IQL - is a proprietary data management
system.   It provides both laymen and computer professionals with a
comprehensive ability to define data files, input or update the data
files, obtain derivative files, retrieve information from data bases
or data files, and generate sophisticated reports.

For example, to produce a printed report entitled PERSONNEL LIST on a
TOPS-20 Operating System, enter the following lines on a terminal:

```
@IQL (RET)

<QA>WRITE (RET)

%FILE NOT FOUND, CREATING NEW FILE
INPUT: QC017S.TMP.1
00100   OPEN PERSONNEL (RET)
00200   HEADING 'PERSONNEL LIST'(RET)
00300   PRINT NAME,SOC-SEC,DATE-HR,SALARY(RET)
00400   (ESC) $
*E (RET)

[QC017S.TMP.1]

<QA>RUN (RET)

@PRINT QC017S.LPT.1/DELETE (RET)
@
```

The first line in the example enters IQL.   The second line in the
example allows you to enter the TOPS-20 text editor where you begin to
write a query.

    OPEN PERSONNEL

The first query statement informs IQL which dictionary and which data
files to read when IQL generates the report. A dictionary, which the
data base manager creates with IQL, describes the data file to IQL. A
dictionary also contains formatting information for the report.

    HEADING 'PERSONNEL LIST'

The second query statement informs IQL of the heading to enter at the
top of the report.

    PRINT NAME,SOC-SEC,DATE-HR,SALARY

The final statement directs IQL to create a print-formatted line for each employee.

You terminate the text editor with the E command and generate the print-formatted report file with the RUN assistance command. Using the TOPS-20 PRINT command, you print the report on the line printer.

1.1 LEVELS AND MODES OF IQL

IQL contains several command levels, each level containing a unique set of commands. IQL displays a message when you enter a level and displays a message when you exit a level.

Figure 1-1 illustrates the IQL levels and the IQL commands you use to move from one level to another level. Upon entering IQL, as explained in Chapter 2 in this guide, you enter the assistance level. For instance to append data to a data file, you use the UPDATE assistance command to enter the update level. IQL enters the update level and displays a prompt indicating that IQL is ready to accept an update level command. You then use the APPEND update level command to enter the input level. IQL displays a prompt for each field in the data file. To terminate the input level, use the EXIT input level command and to terminate the update level, use the EXIT update level command. IQL then returns to the assistance level.

In addition to the levels of IQL, Figure 1-1 shows the two modes of IQL. In the immediate mode, IQL performs the DEFINE, BROWSE, UPDATE, and INPUT level commands as soon as you terminate a command with the carriage return. In the deferred mode, you write a query at the edit level. At the other levels, IQL analyzes the query and generates a print-formatted file.

FUNCTIONS OF IQL

MONITOR LEVEL

R IQL
RUN IQL
IQL

END
EXIT

IQL ASSISTANCE LEVEL
(Creates and displays dictionaries and queries, gives help, and controls the remainder of IQL)

DEFINE

END
EXIT
SCRUB

BROWSE

END
EXIT

UPDATE

END
EXIT

INPUT

EDIT

WRITE

EUN

RUN

EXECUTE

Only if
error in
query

automatic

UPDATE LEVEL

(Retrieves from a data
file, carrying out each
command as you enter
it. File changes are
not permitted.)

UPDATE LEVEL

(Retrieves from or
creates or changes a
data file, carrying
out each command
as you enter it.)

EDIT LEVEL

(Uses TOPS text editor
to change a nonanalyzed
query.)

ANALYSIS LEVEL

(Analyzes the state-
ments in a nonanalayzed
query.)

INSERT
APPEND

END
EXIT

escape
key

(automatic)

**DICTIONARY
DEFINITION
LEVEL**

(Prompts field by
field to create or
change dictionary
entries.)

INPUT LEVEL

(Prompts field by field
to make new data
records in a data file.)

I

**EDIT INPUT
LEVEL**

(Inputs new state-
ments to a query.)

**EXECUTION
LEVEL**

(Follows statement
analysis; Passes one or
more data files or a
data base to prepare
reports and/or output
files.)

immediate mode

deferred mode
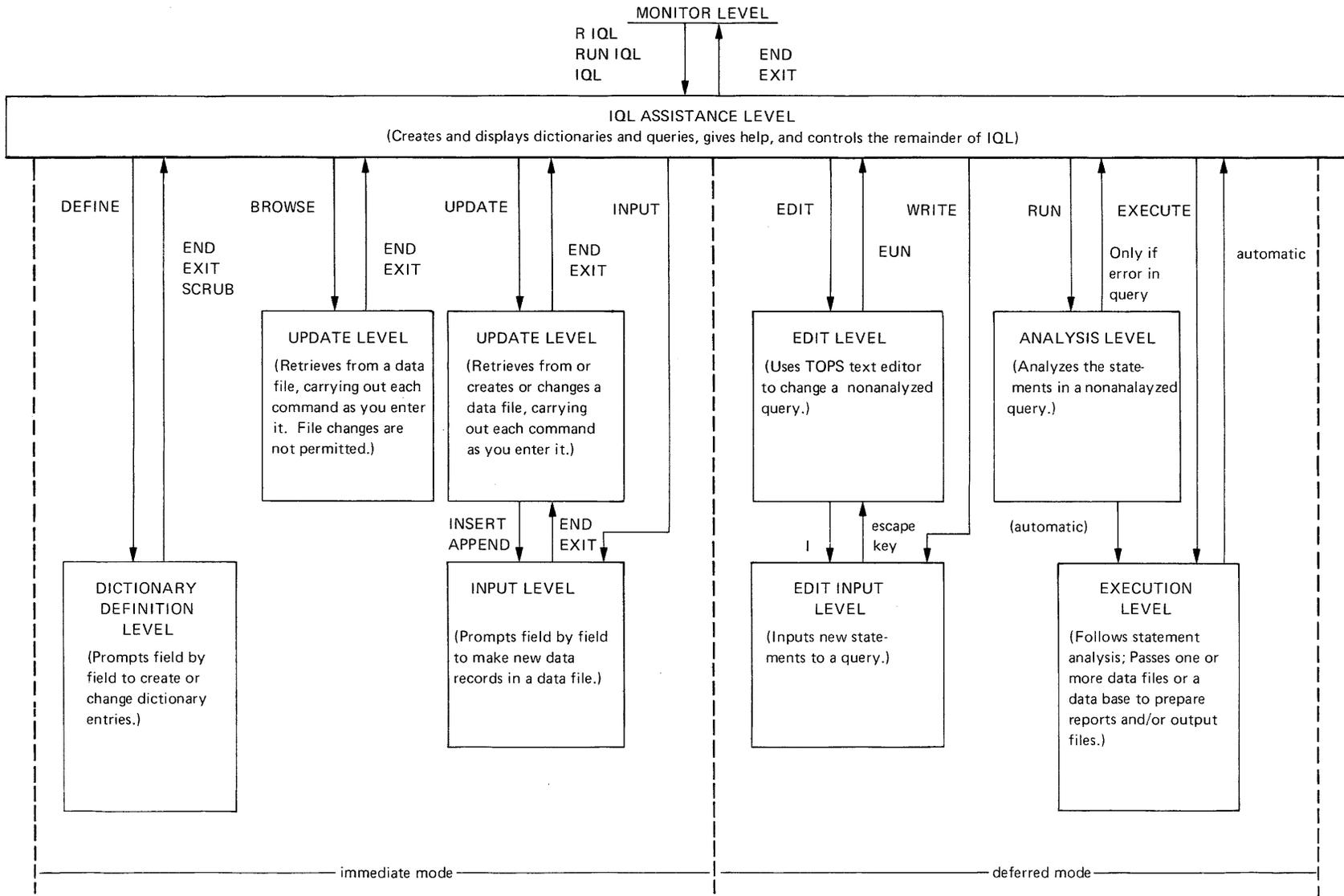
Figure 1-1 Levels and Modes of IQL

1.2 ACCESSING DIFFERENT FILE STRUCTURES

IQL handles sequential, Indexed Sequential Access Method (ISAM), and
Data Base Management Systems (DBMS) file structures. A file structure
determines the method that IQL uses to access information in a file.
To write to a file, IQL collects a set of data and writes it into the
file as a record. A record is the smallest unit of information that
the system reads and writes when it accesses a file. For instance, to
read an address of a customer or the year-to-date purchases of a
customer, the system must read every record from the beginning of the
sequential file in order to arrive at the desired information.   For
the system to read the same information in an ISAM file, the system
proceeds directly to the records associated with the customer that you
request.   For a DBMS file structure, the system reads only the record
containing the data you request.

IQL locates data files or data bases on device DSK: and follows the
search path that exists for DSK:. If you wish to access a file in a
directory other than the directory in which you logged into the
system, you must set up a search path. For instance, by using one of
the following Operating System command sequences, you can set up a
search path which the system uses to locate a file.

>   @DEFINE (LOGICAL NAME) DSK: <DIRECT1>,<DIRECT2> (RET) (TOPS-20)
>   .R SETSRC (RET)   (TOPS-10)

DIRECT1 is the directory that you logged into the system and DIRECT2
is the directory where you want the system to continue the file
search. If the system cannot find the file in DIRECT1, the system
searches DIRECT2 for the file.

Refer to the TOPS-10 Operating System Commands Manual   for   a
description of the SETSRC program.

If you specify more than two directories in the search path, IQL looks
in each directory, from left to right, until IQL finds the file. If
you frequently use a search path, you or the on-site software
specialist can set up a search path in the LOGIN.CMD file. The system
automatically reads the LOGIN.CMD file immediately after you log into
the system.   Refer to the Operating System user's guide (such as as
the TOPS-20 User's Guide) for more information.

1.3 USING RESERVED KEYWORDS

The words, abbreviations, and symbols listed below have special
meanings in IQL.   You should avoid using reserved keywords as
data-item names in dictionaries or as item values. However, you can
use a reserved keyword if you enclose it in quotes.

ALL	EXIT	INPUT	NOT	UP
AND	FIND	INSERT	NQ	UPDATE
APPEND	FIRST	IS	OR	VERIFY
BOTTOM	FOR	LE	PRINT	WHEN
CHANGE	GE	LEQ	QUERY5	WHERE
COLUMN	GEQ	LESS	RECORD	XRANDOM
DELETE	GQ	LIST	RECORDS	=
DOWN	GR	LQ	RECOVER	<
END	GREATER	LS	SAVE	>
EQ	GT	NE	TO	*
EQUAL	IF	NEQ	TODAY	/
EQUALS	IN	NEXT	TOP	+

1.4 DELETING IQL FILES

Although you need not understand this section to use IQL, DIGITAL
recommends that you read this section before you delete any files from
the directory.

IQL creates permanent and temporary files in the login directory to
store dictionaries, queries, data files, and print-formatted reports.
Table 1-1 lists the filenames created by IQL that you can display by
using the TOPS-10 or TOPS-20 DIRECTORY command. The table uses the
symbolic term, job, in filenames such as QCjobS.LPT. The symbolic
term, jobs, refers to the actual job number that the Operating System
assigns when you log into the system. For instance, if the job number
is 17, IQL creates a filename such as QC017E.LPT in the directory.

Table 1-1
Files Created by IQL

File Name	Assistance Command	File Structure	File Contents
QPDICT.SEQ	DEFINE	sequential	One or more dictionaries
QLjobD.LPT	DEFINE	sequential	Line printer file of dictionary changes or full dictionary contents.
QCjobS.TMP	EDIT	sequential	Current query area.
QCjobS.QMP	EDIT	sequential	Previous current query area.
QCjobS.TMP	WRITE	sequential	Current query area.
QCjobE.LPT	RUN or	sequential	Print-formatted report.
QCjobE.LPT	EXECUTE	sequential	Print-formatted report.
name.QRY	RUN or	sequential	One analyzed query.
name.QRY	SAVE	sequential	One analyzed query.
name.OUT	RUN or	sequential	Copied data file (same format).
name.OUT	EXECUTE	sequential	
name.type	RUN or	sequential	Copied data file (same format).
name.type	EXECUTE	sequential	
name.type	UPDATE	sequential	One data file.
name.type	UPDATE	ISAM	One data file.
name.type	INPUT	sequential	One data file.
name.type	INPUT	ISAM	One data file.
QLjobU.LPT	INPUT	sequential	One line diary file.
QLjobU.LPT	BROWSE	sequential	One line diary file.
QLjobU.LPT	UPDATE	sequential	One line diary file.
QPQRYS.SEQ	REPLACE	sequential	One or more nonanalyzed queries.
QPQRYS.SEQ	STORE	sequential	

In addition to the EXECUTE and RUN assistance commands, the COPY query statement also creates a sequential data file with the filename of either name.OUT or name.type. The .OUT file type is supplied by IQL when you omit a file type specification to IQL.

Some of the temporary files, such as the QCjobS.TMP file, are deleted by IQL when, for instance, you use a second WRITE assistance command in the same IQL session or when you use the EXIT assistance command to terminate an IQL session. Generally when you conclude an IQL session, you should either:

1. Use the TOPS-10 or TOPS-20 DELETE command to remove unwanted files that IQL does not delete.

2. Use the TOPS-10 or TOPS-20 PRINT command with the /DELETE switch to print print-formatted files.

Except for print-formatted files, IQL deletes an existing file before creating another file using the same filename. For print-formatted files, IQL creates another file with a unique filename by incrementing the first digit of the job number in the filename. For instance, if you issue a second RUN assistance command in the same session and IQL encounters the file QL017E.LPT in the directory, IQL writes the new report(s) to the new file QL117E.LPT.

CHAPTER 2

RUNNING IQL

Depending upon how IQL is implemented at the site, you begin IQL from the Operating System with one of the following command sequences:

    IQL (RET)

or

    R IQL (RET)

or

    RUN IQL (RET)

or

    RUN DEVICE:<DIRECTORY>IQL (RET)

You terminate the command with a carriage return which permits the Operating System to execute IQL. IQL displays a <QA> prompt and expects you to input an assistance command.

                              NOTE

              For IQL to run, the ISAMF6.IDX and
              ISAMF7.IDX files must exist in the
              search path. The files are supplied to
              the site on the IQL Distribution Tape.

2.1 EXPLAINING IQL PROMPTS

The following list explains the prompts that IQL displays and the responses that IQL expects. You terminate the response with a carriage return, which permits IQL to accept the response.

    1.   <QA>

         The <QA> prompt indicates that IQL is waiting for you to type an assistance command. For instance, IQL displays:

             <QA>

         You respond with an assistance command such as:

             <QA> ITEMS INVENTORY (RET)

2. <QU>

    The <QU> prompt indicates that IQL is waiting for you to type
    an immediate mode command. For instance, IQL displays:

        <QU>

    You respond with an immediate mode command such as:

        <QU>LIST ALL DEPT 421 (RET)

3. *prompt:

    A prompt that begins with an asterisk (*) and ends with a
    colon (:) indicates that IQL wants you to type an item of
    data. For instance, IQL displays:

        *DESCRIPTION (IF ANY):

    You respond by typing a line such as:

        *DESCRIPTION (IF ANY): VALUE FOR ITEM (RET)

2.2 EXPLAINING MESSAGE FORMATS

IQL displays messages as follows:

1. (message)

    IQL displays progress reports or information messages and
    encloses them in parentheses. You are not required to take
    any action.

    For instance, IQL displays:

        (END OF UPDATE SESSION)

    The message informs you that IQL is exiting the immediate
    mode.

2. % message

    Messages starting with a percent sign are warning messages.
    For instance, IQL might display:

        % CANNOT FIND THIS DICTIONARY

    The message informs you that IQL cannot locate the dictionary
    that you specified in an assistance command. Repeat the
    command with a valid dictionary or use the DICTIONARY
    assistance command to display the names of the dictionaries
    in the directory.

2.3 USING ASSISTANCE COMMANDS

To enter an assistance command, you need only supply enough of the command to uniquely identify it. (For instance, DIC is enough to identify the DICTIONARIES command.) If IQL expects more words in the command, IQL displays a prompt and ends it with a colon (:). You terminate the command with a carriage return, which permits IQL to perform the function.

2.3.1 Controlling and Displaying the IQL Job

Use the JOB assistance command to display the job number. The system uses the job number to name temporary and print files that you may wish to access outside of IQL. Use the EXIT or END assistance command to terminate IQL and to return to the Operating System.

2.3.2 Creating, Editing, and Saving a Query

When you first write a query with either the EDIT or WRITE command, IQL enters the information into the current query area. IQL uses the current query area as a work space. The current query area contains the source query statements that you enter following the EDIT or WRITE assistance command. The source query statements are described in Chapter 3. You can store the current query area into a file with either the STORE or SAVE command. With the STORE command, IQL writes the nonanalyzed current query area into a permanent disk file. With the SAVE command, IQL analyzes the source query statements from either the current query area or a nonanalyzed query file and writes the analyzed query into a permanent disk file. If you attempt to store or save a query without naming it, IQL prompts you for a query name.

To work with a nonanalyzed query file, you must return it to the current query area by using the query name with either the GET, EDIT, LIST, RUN, or SAVE command. You can edit a nonanalyzed query file after you return it to the current query area. You cannot return an analyzed query to the current query area and edit the analyzed query.

2.3.3 Using a Query File to Generate a Report

IQL first analyzes the query instructions and then performs them. Use the RUN command to analyze the query and to generate the report(s) from a nonanalyzed query. The nonanalyzed query can be located in either the current query area or in a nonanalyzed query file.

Use the EXECUTE command to generate a report from an analyzed query. Before using the EXECUTE command, you must analyze and save the nonanalyzed query with the SAVE command. Generating a report from an analyzed query is faster than generating a report from a nonanalyzed query.

2.3.4 Maintaining Query Files

IQL retains nonanalyzed queries in a single file and analyzed queries in separate files. Use the QUERIES command to list the nonanalyzed queries. Use the DELETE command to remove a nonanalyzed query and the REPLACE command to replace a stored nonanalyzed query with the contents of the current query area.

2.3.5 Creating and Changing a Data File

Use either an INPUT, UPDATE, or BROWSE command to create, update, or read a data file. You can only create and modify sequential and indexed sequential data files. Refer to Chapter 4 in this guide for more information.

2.3.6 Creating and Changing a Dictionary

A dictionary defines the data base or data file to IQL. Use the DEFINE, DICTIONARY, and ITEMS commands to manipulate a dictionary. The DEFINE command permits you to create a new dictionary or to change an existing dictionary. Although IQL checks the dictionary changes or additions as you enter them, IQL does not physically create or change the dictionary until you enter an END dictionary command. Refer to Chapter 5 in this guide for a full description.

2.4 ASSISTANCE COMMAND FORMATS

The assistance commands use the following command conventions:

1. First character location

    The command begins in the first character location afer the <QA> prompt.

2. Space

    One or more spaces separate a command from a parameter. Also one or more spaces separate parameters.

3. Lowercase and uppercase characters

    A parameter identified by lowercase characters indicates that you are to supply a variable as indicated by the variable name.

    A parameter identified by uppercase characters indicates that you are to supply the exact characters as they are shown in the text. Usually, an uppercase parameter is an option to the command and provides you with additional functions to the command.

4. Square brackets [ ]

    A parameter enclosed in square brackets indicates an optional parameter. Do not use the square brackets in the assistance commands you write; the square brackets indicate an optional parameter. When you do not supply the parameter, the system applies a default value as explained in the parameter description.

    A parameter not enclosed in square brackets indicates that the system requires the parameter.

5. Examples

In the examples, data you enter is shown in red print whereas data the system displays is shown in black print.

The remainder of the chapter presents the assistance commands in alphabetical order.

BROWSE

FUNCTION:

The BROWSE assistance command places IQL in the immediate mode but restricts to read-only access the sequential or indexed sequential data file (subject to any password requirement). Refer to Chapter 4 in this guide for a description of the immediate mode.

The BROWSE assistance command is similar to the UPDATE assistance command, except that you can only read the data file.

FORMAT:

BROWSE dictionary-name [filename]

DISCUSSION:

Specify a dictionary name that IQL can find in the login directory. You can request IQL to display a list of the dictionaries with the DICTIONARIES assistance command.

If you supply a filename, IQL uses that filename rather than the filename in the dictionary. The filename must follow the ffffff.ext naming convention, where ffffff is one to six alphanumeric characters and ext is one to three alphanumeric characters.

If the file is protected at the READ level, IQL asks you for the password and checks it before unlocking the file for browsing.

EXAMPLES:

1. <QA>BROWSE CUSTOMERS (RET)
    (YOUR DIARY FILE IS: QL039ULPT)

    <QU>LIST IF CUSTNO = 23 (RET)

    00023DEEPSEA DIVERS, INC   DENISE DU BOIS   DEEPSEA LAN
                  MIAMI   FL2937500006777700006777000100000

    <QU>END (RET)
    (END OF UPDATE SESSION)

    <QA>

    IQL lists the first record that qualifies.

2. <QA>BROWSE CUSTOMERS (RET)
    (YOUR DIARY FILE IS: QLO39ULPT)
    <QU>LIST ALL IF CUSTNO = 23 (RET)

    00023DEEPSEA DIVERS, INC   DENISE DU BOIS   DEEPSEA LAN
                   MIAMI   FL293750000677770000677770000100000
    00023JOHN JONES   BILL WILLIAMS   14 COMMERCI
    AL WHARF   BOSTON   MA018320002324790000467000000250000
    0023PACER DEVICES   WILLARD J. HOLMES   47 BAYPORT
    AVE.   CHICAGO   IL60477000233175000100000000250000

    (AT END OF FILE)

    <QU>END (RET)
    (END OF UPDATE SESSION)

    <QA>

    IQL lists all the records that qualify. Since the example
    did not request any particular items after the word list, IQL
    displays the entire record.

3. <QA>BROWSE PERSONNEL (RET)
    (YOUR DIARY FILE IS: QLO39ULPT)
    <QU>LIST FNAME INIT LNAME PHONE ALL IF STATE = COLORADO

    KEVIN   S NELMS   (449)989-1901
    IRENE   G ZARZEWSKA   (045)211-0912
    OSCAR   P KOURANEY   (379)491-5737
    (AT END OF FILE)
    <QU>END (RET)
    (END OF UPDATE SESSION)

    <QA>

    The EMPLOY.NEW filename overrides the filename contained in
    the dictionary.   Also the example requests that IQL display
    specific items from the records that qualify.

DEFINE

FUNCTION:

The DEFINE assistance command defines the dictionary transactions
to create or update an IQL dictionary.

FORMAT:

DEFINE

DISCUSSION:

When you terminate a DEFINE session with the END dictionary
command, IQL creates or updates the dictionary based on the
transactions you enter. If you terminate a DEFINE session with
the SCRUB dictionary command, IQL aborts the DEFINE session and
does not change or create a dictionary. Refer to Chapter 5 in
this manual for more information. Three example dictionaries are
shown in Appendix C.

EXAMPLE:

1. <QA> DEFINE(RET)

    *FD,DD,PD,CD,RD,AD,SD,END,SCRUB: FD(RET)
    *ACTION CODE (A,C,D OR P):   C(RET)
    *DICT NAME - UP TO 30 CHARS: PERSONNEL(RET)
    *DICT UNLOCKING PASSWORD, IF ANY: SESAME(RET)
    *FILE IN NAME AS XXXXXX.EEE: NEXT(RET)

    *FD,DD,PD,CD,RD,AD,SD,END,SCRUB: DD(RET)
    *ACTION CODE (A,C,D):   C(RET)
    *ITEM NAME (UP TO 30 CHARS):   NAME(RET)
    *TOP COLUMN TITLE (UP TO 10 CHARS): FULL NAME(RET)
    *BOTTOM TITLE (UP TO 10 CHARS):   NEXT(RET)

    *FD,DD,PD,CD,RD,AD,SD,END,SCRUB: END(RET)

    (END DICTIONARY TRANSACTION INPUT)
    (END DICTIONARY UPDATE)
     <QA>

    The example illustates changing a dictionary named PERSONNEL.
    The first transaction - FD - supplies a password to IQL and
    explains to IQL that you wish to change information located
    in the dictionary. IQL unlocks the dictionary. The second
    transaction - DD - changes the top column title of the item,
    NAME, to FULL NAME. All the other dictionary fields remain
    unchanged.

DELETE

FUNCTION:

The DELETE assistance command removes the stored nonanalyzed query(s) from the QPQRYS.SEQ file of stored queries.

FORMAT:

DELETE   query-name [... query-name]

EXAMPLES:

1.   <QA> DELETE IF-SORT-CUTOFF JACKTEST (RET)
     (IF-SORT-CUTOFF DELETED)
     (JACKTEST DELETED)

     <QA>

2.   <QA> DELETE CHARLIE BETH CLAIRE (RET)
     (CHARLIE DELETED)
     % BETH NOT FOUND TO BE DELETED
     (CLAIRE DELETED)

     <QA>

     In the second example, IQL could not find a query with the
     name BETH.   IQL displays a warning message and then deletes
     the query CLAIRE.

DICTIONARIES

FUNCTION:

The DICTIONARIES assistance command provides a terminal display of the file information for each dictionary in the directory.

FORMAT:

DICTIONARIES [master-password]

DEFAULTS:

IQL proceeds to display a list of the dictionaries. If IQL encounters a protected dictionary, IQL omits protection information and displays the dictionary name, file type, and directory entries.

DISCUSSION:

The column headings in the display are explained as follows:

DICT NAME
The DICT NAME column lists the dictionary names.

FILE TYPE
The FILE TYPE column lists, for the data file, the file structure and the data format. The file structure can be SQ for sequential files, IS for ISAM files, or DTABASE for DBMS files. The data format can be DISK6 for 6-bit ASCII format or DISK7 for 7-bit ASCII format.

FILE-IN NAME
The FILE-IN NAME column lists the name of the data file that the dictionary describes.

DIRECT
The DIRECT column lists the directory where IQL can find the data file.

REC LEN
The REC LEN column defines as the number of ASCII characters the logical length of the data file record.

BLK FAC
The BLK FAC column describes the number of records per block; 0 means that the system takes care of the blocking factor.

KEY LOC
The KEY LOC column lists, for ISAM files, the character location in the record of the leftmost character of the key. (The leftmost character location in a record is character position 1.)

KY LN
The KY LN column lists, for ISAM files, the length in ASCII characters of the key.

KY TP
The KY TP column lists, for ISAM files, the key type. A key type of AU is an alphanumeric key. A key type of NS is a signed numeric key. A key type of NU is an unsigned numeric key.

RD PW   The RD PW column lists the read password
                  reference, which occurs later in the dictionary
                  and protects against unauthorized reading of
                  records from the data file.

CP PW   The CP PW column lists the copy or write
                  password reference, which occurs later in the
                  dictionary and protects against unauthorized
                  creating, copying, and updating of records from
                  the data file.

RW PW   The RW PW column lists the rewrite password
                  reference, which occurs later in the dictionary
                  and protects against unauthorized changing of
                  records in an ISAM data file.

EXAMPLE:

1. <QA> DICTIONARIES   SESAME (RET)

    DICTIONARIES IN YOUR DIRECTORY:

DICT NAME	FILE TYPE	FILE-IN NAME	DIRECT	REC LEN	BLK FAC	KEY LOC	KY LN	KY TP	RD PW	CP PW	RW PW
INVENTORY	SQ	DSK7 INVENT.SEQ	JANE	180	0						
PERSONNEL	IS	DSK7 PERSON.IDX		600	5	1	5	AU	10	20	30

    . . . .
    (END LIST OF DICTIONARIES)

    <QA>

    As shown in the example, the directory contains the
    dictionaries INVENTORY and PERSONNEL. The INVENTORY
    dictionary describes a sequential data file by the name of
    INVENT.SEQ located in the directory, <JANE>. The PERSONNEL
    dictionary describes an ISAM data file by the name of
    PERSON.IDX located in the login directory.

EDIT

FUNCTION:

The EDIT assistance command transfers control to a text editor to
edit or change either the current query area or a stored
nonanalyzed query in the QPQRYS.SEQ file of stored queries.

FORMAT:

EDIT [query-name] [LIST]

DEFAULTS:

If you omit a query name, IQL edits the current query. If you
omit the LIST option, IQL does not list the query before entering
the editor.

DISCUSSION:

If you furnish a query name, IQL moves the contents of the
related query to the current query area before entering the
editor.

IQL enters an editor within IQL when you execute IQL under the
TOPS-10 Operating System. IQL enters the EDIT editor when you
execute IQL under the TOPS-20 Operating System. Appendix E in
this guide explains the IQL editor and provides a summary of some
of the TOPS-20 EDITOR commands.

EXAMPLES:

1. <QA> EDIT NEWHIRES LIST(RET)
    **NEWHIRES
    OPEN PERSONNEL.
    HEADING 'LIST OF NEW HIRES'.
    IF DATE-HR LEQ 77 PRINT NAME, DEPT, DATE-HR.

    EDIT: QC014S.QRY
    * (LF)
    00100   OPEN PERSONNEL.
    *I. (RET)
    00150   DISPLAY ON.(RET)
    *FHEADING(ESC)$
    00200   HEADING 'LIST OF NEW HIRES'.
    *SHIRES (ESC)$HIRES//SINCE JANUARY 1(ESC)$. (RET)
    00200   HEADING 'LIST OF NEW HIRES//SINCE JANUARY 1'.
    * (LF)
    00300   IF DATE-HR LEQ 77 PRINT NAME, DEPT, DATE-HR.
    *SLEQ (ESC)$ GEQ (ESC)$. (RET)
    00300   IF DATE-HR GEQ 77 PRINT NAME, DEPT, DATE-HR.
    *P100 (RET)
    00100   OPEN PERSONNEL.

```
*P (RET)
00100   OPEN PERSONNEL.
00150   DISPLAY ON.
00200   HEADING 'LIST OF NEW HIRES//SINCE JANUARY 1'.
00300   IF DATE-HR GEQ 77 PRINT NAME, DEPT, DATE-HR.
*EUN (RET)
[QC014S.QRY]
<QA>
```

In example 1, line 150 is added, the report heading is
changed, and the IF clause is changed from LEQ to GEQ.

IQL lists the NEWHIRES query and enters the text editor. The
first editor command, the linefeed key, displays the first
query statement, which is the first line in the query.   The
second editor command, the I character and a period, enters
the editor into the insert mode. You then add a line between
the   first   and second query statements.   The editor
automatically terminates the insert mode when you press the
carriage return key.   With   the   third editor command, the
FHEADING$ sequence, you request the editor to find the first
line that contains the character string HEADING. The editor
displays the line when the editor finds the character string.
With the fourth editor command, another line feed, you
display the line following the report heading line.   The
sixth editor command, the SLEQGEQ. character string,
requests the editor to change the LEQ sequence to GEQ.   The
P100 editor command · lists the query with all the changes.
The E editor command exits the editor and returns control to
IQL.

2.   <QA>EDIT   NEWHIRES   LIST (RET)
      **NEWHIRES
        OPEN PERSONNEL.
        HEADING 'LIST OF NEW HIRES'
        IF HIRE-YR LEQ 77 PRINT NAME, DEPT, HIRE.

      *L (RET)
        OPEN PERSONNEL.
      *I. (RET)
        DISPLAY ON. (RET)
      *FHEADING: (RET)
        HEADING 'LIST OF NEW HIRES'.
      *SHIRES:HIRES//SINCE JANUARY 1:. (RET)
        HEADING 'LIST OF NEW HIRES//SINCE JANUARY 1'.
      *L (RET)
        IF HIRE-YR LEQ 77 PRINT NAME, DEPT, HIRE-YR.
      *SLEQ:GEQ:. (RET)
        IF HIRE-YR GEQ 77 PRINT NAME, DEPT, HIRE-YR.
      *TOP (RET)
        OPEN PERSONNEL.
      *P;ALL (RET)
        OPEN PERSONNEL.
        DISPLAY ON.
        HEADING 'LIST OF NEW HIRES//SINCE JANUARY 1'.
        IF HIRE-YR GEQ 77 PRINT NAME, DEPT, HIRE-YR.
      *END (RET)
      <QA>
```

In example 2, the same edits are performed  as  explained  in
example  1.    However,  example 2 shows the IQL editor on the
TOPS-10 Operating System.

EXECUTE


FUNCTION:

The EXECUTE assistance command generates a report from an analyzed query file.


FORMAT:

EXECUTE query-name


DISCUSSION:

You must specify an analyzed query.  Each analyzed query file uses the extension of .QRY.  Refer to the SAVE command for more information on analyzed query files.

If you expect to use a query frequently without changing it, you can save computer time by analyzing the query once, saving it, and using the EXECUTE assistance command each time you want a report.  With the EXECUTE assistance command, IQL bypasses the analyze stage and produces the report.  This procedure is particularly useful if you are using IQL as a report writer in a production stream with other programs.


EXAMPLE:

1.   <QA> EXECUTE  NEWHIRES (RET)

     ...(the report goes here if routed to terminal)...

     <QA>

     Immediately after you issue the EXECUTIVE assistance command, IQL looks for the HIRE.QRY analyzed query file, reads the data file, and generates the report(s).

EXIT

FUNCTION:

The EXIT assistance command returns control from IQL to the Operating System and displays an exit message.

FORMAT:

EXIT

DISCUSSION:

When you terminate the IQL session, IQL deletes all the working files it creates, including the current query area. To preserve the current query area before you terminate IQL, use the STORE assistance command.

EXAMPLES:

1. <QA> EXIT (RET)
   EXIT
   @

   You terminate IQL and return to the TOPS-20 Operating System.

2. <QA> EXIT (RET)
   EXIT
   .

   You terminate IQL and return to the TOPS-10 Operating System.

GET

FUNCTION:

The GET assistance command moves a nonanalyzed query from the file of stored queries into the current query area.

FORMAT:

GET [query-name] [LIST]

DEFAULTS:

If you omit a query name, GET uses the current query area, which remains unchanged.

If you use the LIST option, IQL displays the name of the query and the contents. If you omit the LIST option, IQL displays the name of the query.

DISCUSSION:

If you use a query name, IQL writes the query from the file of nonanalyzed stored queries into the current query area. The GET command destroys the previous contents of the current query area.

EXAMPLES:

1. <QA>GET (RET)
   **BLUE-EYES
   <QA>

   IQL lists the current query name, which is BLUE-EYES.

2. <QA> GET    LIST(RET)
   **BLUE-EYES
   OPEN APPLICANTS.
   IF COLOR-EYES = 'BLUE' SORT BY STATE, CITY.
   PRINT NAME, STATE, CITY, AGE.
   <QA>

   IQL lists the contents of the current query area, which already contains the query BLUE-EYES.

3. <QA> GET    NEWHIRES(RET)
   **NEWHIRES
   <QA>

   IQL retrieves the query NEWHIRES from the file of nonanalyzed queries to the current query area and lists the query name.

4. <QA> GET    NEWHIRES    LIST (RET)
   **NEWHIRES
   OPEN PERSONNEL
   HEADING 'NEW HIRES SINCE//JANUARY 1, 1977'
   IF HIRE-DATE GEQ 770101 PRINT NAME, DEPT, HIRE-DATE
   <QA>

   IQL retrieves the query NEWHIRES from the file of nonanalyzed queries to the current query area and lists the query.

INPUT

FUNCTION:

The INPUT assistance command places IQL into the input level. In the input level, you enter data on the terminal keyboard to create sequential data files. Refer to Chapter 4 in this guide for more information on the input level.

FORMAT:

INPUT  dictionary-name  [filename]

DEFAULTS:

Before IQL prompts you for field values, IQL initially sets each record in the data file to all spaces.

DISCUSSION:

If you omit a dictionary name, IQL prompts you for a name.

If you also enter a filename, IQL uses the filename you enter instead of the filename in the dictionary. The filename must follow the ffffff.ext naming convention, where ffffff is one to six alphanumeric characters and ext is one to three alphanumeric characters.

If IQL finds that the file already exists in the directory, IQL displays a message and enters the update level (that is, the INPUT assistance command does not permit you to write over an existing file). In the update level, you can change the data file. Refer to Chapter 4 for more information on the update level.

If IQL finds that the dictionary contains a password protection at the READ or WRITE level, IQL prompts you for the password and checks it before permitting you access.

IQL begins the input level by issuing a prompt for the first field in the first record and waits for the response. IQL prompts you for item values in the order in which IQL finds the items in the dictionary. IQL issues a prompt for the next field in the record and continues the prompts until IQL encounters the end of the dictionary, you direct IQL to perform something else, or you enter the END or EXIT input command. When you terminate the input level with the END or EXIT input command, IQL returns to the assistance level. Refer to the INSERT command description, located in Chapter 4, for a list of the commands to control the input level and for an explanation of the error handling.

EXAMPLE:

1.  <QA> INPUT      CUSTOMERS (RET)
    (THE DIARY FILE IS QL023U.LPT)

    (NEXT RECORD)
     *NUMB (CUSTOMER NUMBER) 5 N:    34 (RET)
     *NAME (CUSTOMER NAME) 30 A:     PORTABLE DEVICES (RET)
     *SLSMAN (SALESMAN NAME) 25 A:  CARLETON SCOTT (RET)
     *CITY (CUSTOMER CITY) 15A:      UP 2 (RET)
     *NAME (CUSTOMER NAME) 30 A:     PORTABLE DEVICES (RET)
     *SLSMAN (SALESMAN NAME) 25 A:  DOWN (RET)
     *CITY (CUSTOMER CITY) 15 A:     ST. LOUIS (RET)
     *CYSLS (CURR YR SALES) 8.2 N:  $1,500. (RET)
     *LYSLS (LAST YR SALES) 8.2 N:  3012.46 (RET)
     *CLIMIT (CREDIT LIMIT) 8.2 N:  NEXT (RET)

    (NEXT RECORD)
     *NUMB (CUSTOMER NUMBER) 5 N:    37
    ...(continues until you terminate it with END or  EXIT)...
    ...(you terminate the input level with END or EXIT)...
    <QA>

    The example shows IQL prompting for the creation of a
    complete record.  IQL found the data item descriptions in the
    CUSTOMERS dictionary in the order of NUMB, NAME, ..., LYSLS,
    and CLIMIT.

    Also, notice that the UP 2 input command was used to go  back
    two prompts and correct an error for the data item, NAME.

ITEMS


FUNCTION:

The ITEMS assistance command formats and displays the contents of a dictionary.


FORMAT:

ITEMS    dictionary    [master-password]


DEFAULTS:

If you omit the password and IQL finds that the dictionary or an item is protected, IQL obscures any information that would allow you to locate the protected item.

DISCUSSION:

IQL formats the dictionary contents for a 72-character terminal display line.

You can also print the dictionary contents on the line printer. As explained in Chapter 5, the FD dictionary command formats the dictionary contents for a 132-character line and displays the formatted contents on the line printer.

IQL displays the information in two sections. The first section is a description of the dictionary and the data file. Refer to the DICTIONARY assistance command for an explaination of the first section. The second section of the display is a description of the dictionary contents labeled with column headings. Refer to Chapter 5 for a description of the second section.


EXAMPLE:

1. <QA> ITEMS   PERSONNEL   SESAME (RET)


| DICT NAME | FILE TYPE | FILE-IN NAME | DIRECT | REC LEN | BLK FAC | KEY LOC | KY LN | KY TP | RD PW | CP PW | RW PW |
|---|---|---|---|---|---|---|---|---|---|---|---|
| PERSONNEL | IS DSK7 | PERSON.IDX | <JANE> | 600 | 5 | 1 | 5 | AU | 10 | 30 | 50 |


| ID | ITEM NAME | TOP TITLE | BOTTOM TITLE | 1ST CHAR | NO. CHAR | T Y | S C | PRINTING PICTURE | SCAN GNNS | PT |
|---|---|---|---|---|---|---|---|---|---|---|
| FD | TIGER | | | 50 | | | | | | |
| FD | BEAR | | | 30 | | | | | | |
| FD | FOX | | | 10 | | | | | | |
| DD | EMPNO | EMPLOY | NUMBER | 1 | 5 | N | O | ZZZZ9 | | |
| DD | LNAME-CH | LNAME | CHAR | 28 | 1 | A | O | | | |

. . . . .
     <QA>

The example shows a display of the PERSONNEL dictionary. The password, SESAME, was specified to IQL so that IQL could display the passwords in the PD entries.

JOB

FUNCTION:

The JOB assistance command displays the job number that the
Operating System assigns to the IQL session.

FORMAT:

JOB

DISCUSSION:

IQL uses the job number as an ingredient for naming its temporary
and print files. The job number can be useful if you wish to
work with these files outside of IQL. Chapter 1 in this guide
describes the specific files that IQL uses.

EXAMPLE:

1.  &lt;QA&gt; JOB (RET)
    JOB NUMBER: 32
    &lt;QA&gt;

LIST

FUNCTION:

The LIST assistance command displays the contents of a nonanalyzed query.

FORMAT:

LIST      [query-name]

DEFAULTS:

If you omit a query name, IQL lists the current query area.

If you supply a query name, IQL moves the nonanalyzed query from the file of stored queries into the current query area and displays it.

EXAMPLES:

1.  &lt;QA&gt; **LIST** ⟨**RET**⟩
    **MARKET-PROJECT
    OPEN PROSPECTS
    COMPUTE EXPECTED = EXPECTED  * 1.1
    TOTAL EXPECTED BY REGION, DISTRICT
    MAXIMUM EXPECTED BY REGION, DISTRICT
    &lt;QA&gt;

    The example lists the current query area, which is named MARKET-PROJECT.

2.  &lt;QA&gt; **LIST    PROSPECT-DETAIL** ⟨**RET**⟩
    **PROSPECT-DETAIL
    OPEN PROSPECTS
    HEADING 'PROSPECT//DETAIL'
    PRINT PROSPECT, DATE-IN, EXPECTED, PROBABILITY

    &lt;QA&gt;

    In the example, IQL retrieves and lists the stored nonanalyzed query named PROSPECT-DETAIL.

QUERIES

FUNCTION:

The QUERIES assistance command displays a list of all nonanalyzed queries stored by IQL in the file of stored queries.

FORMAT:

QUERIES

EXAMPLE:

1.  <QA> QUERIES (RET)

    QUERIES STORED IN THE DIRECTORY:

    NEWHIRES        SALARY-ANALYSIS         PROJECTS
    SALES-ANALYSIS  CUSTOMERS-OVER-CREDIT    LEADS
    LOANS           RISK-ANALYSIS

    (END LIST OF STORED QUERIES)

    <QA>

REPLACE

FUNCTION:

   The REPLACE assistance command deletes the named query  from  the
   file  of  stored  queries  and writes the contents of the current
   query area into the file of stored queries.

FORMAT:

   REPLACE      [query-name]

DEFAULTS:

   If you omit the query name from the command line but you name the
   current  query  area,  IQL  replaces the stored nonanalyzed query
   with the contents of the current query area.  If you do not  name
   the  current  query  area and do not supply a query name with the
   command, IQL prompts you for a query name.

DISCUSSION:

   The contents of the current query area are not  affected  by  the
   REPLACE command.

EXAMPLES:

   1.   <QA> REPLACE (RET)
        (PO-AUDIT REPLACED)
        <QA>

   2.   <QA> REPLACE  OPEN-PO (RET)
        (OPEN-PO REPLACED)
        <QA>

RUN

FUNCTION:

The RUN assistance command processes a nonanalyzed query and generates a report.

FORMAT:

RUN    [query-name]    [NOLIST]    [SAVE]

DEFAULTS:

If you omit the query name, IQL runs the query located in the current query area. If you furnish a query name, IQL moves that query from the stored nonanalyzed query into the current query area, analyzes the query, and generates a report.

If you use the NOLIST option, IQL does not list the query contents. If you omit the NOLIST option, IQL lists the query contents.

If you use the SAVE option, IQL creates an analyzed query file with a file type of .QRY. Refer to the SAVE command description for more information.

DISCUSSION:

The query in the current query area is not affected by IQL when you use the RUN assistance command to generate a report from the current query area.

If IQL finds an error in the query during an analysis, IQL issues the appropriate error messages and returns to the assistance level so that you can correct the errors (by using the EDIT assistance command).

EXAMPLES:

1.   <QA> RUN (RET)
     ...(listing of current query goes here)...
     ...(report produced by current query goes here)...
     <QA>


2.   <QA> RUN    DEPT-MGRS    NOLIST (RET)
     **DEPT-MGRS
     ...(report produced by query DEPT-MGRS goes here)..
     (END QUERY PHASE; PRINT FILE IS QL215E.LPT)
     <QA>

SAVE

FUNCTION:

The SAVE assistance command analyzes and saves in an analyzed query file the current query area or a stored nonanalyzed query. You can subsequently generate a report from an analyzed query file with the EXECUTE assistance command.

FORMAT:

SAVE      [query-name]    [NOLIST]    [RUN]

DEFAULTS:

If you supply a query name, IQL analyzes the nonanalyzed query in the file of stored queries and writes the analyzed query into a file named name.QRY. The filename is the first six characters of the query name.

If you omit the query name, IQL uses the name of the current query area. If you do not name the current query area, IQL requests a query name.

If you use the NOLIST option, IQL does not list the query contents. If you omit the NOLIST option, IQL lists the query contents.

If you use the RUN option, IQL analyzes the query, generates the report, and saves the analyzed query into a disk file.

DISCUSSION:

IQL writes the analyzed query in a disk file with a filename consisting of the first six characters of the query name combined with the extension .QRY. Since some versions of the system utilities do not fully process filenames with dashes (-) in them, you should avoid using dashes in the first six characters of the query name.

If a .QRY file already exists with the target name, IQL notifies you of the conflict and refuses to save the new query; neither the existing .QRY file nor the current query area is affected.

EXAMPLES:

1.   <QA> SAVE (RET)
     **NEWHIRES

     OPEN PERSONNEL.
     HEADING 'LIST OF NEW HIRES'.
     IF DATE-HR LEQ 77 PRINT NAME, DEPT, DATE-HR.
     % IQA014 INVALID/MISSING PASSWORD FOR  PERSONNEL

     <QA>EDIT (RET)
     EDIT: QC039S.TMP.3
     * (LF)
     00100    OPEN PERSONNEL.

```
*I.(RET)
00150  AUTHORITY TIGER.(RET)
*EUN (RET)
         -

[QC039S.TMP.4]

<QA>SAVE (RET)
**NEWHIRES

OPEN PERSONNEL.
AUTHORITY TIGER.
HEADING 'LIST OF NEW HIRES'.
IF DATE-HR LEQ 77 PRINT NAME, DEPT, DATE-HR.

<QA>
```

2.  ```
    <QA> SAVE  QUARTERLY  NOLIST  RUN (RET)
      **QUARTERLY
      ...(report produced by QUARTERLY goes here)...
      (QUARTERLY SAVED AS QUARTE.QRY)
      <QA>
    ```

STORE

FUNCTION:

The STORE assistance command stores the current query area into the file of nonanalyzed queries.

FORMAT:

STORE [query-name]

DEFAULT:

If you omit a query name, IQL uses the name of the current query area. If you did not name the current query area, IQL prompts you for a query name.

DISCUSSION:

IQL writes the nonanalyzed query in the QPQRYS.SEQ file of stored queries. You can retrieve a stored nonanalyzed query at any time with the LIST, EDIT, RUN, or SAVE assistance command.

If you attempt to store a query and a query already exists by that name, IQL notifies you of the conflict and refuses to store the query. Neither the current query area nor the existing stored query is affected.

To generate a report from a stored query, use the RUN command with the query-name option. Refer to the RUN assistance command for more information.

EXAMPLES:

1. <QA> STORE (RET) (the current query, R2D2, is stored)
    (R2D2 STORED)

    <QA>

2. <QA> STORE   C3PO (RET)
    (C3PO STORED)

    <QA>

UPDATE

FUNCTION:

The UPDATE assistance command places IQL into the update level of the immediate mode so that you can update or inspect either sequential or indexed sequential data files (subject to any password requirement). Refer to Chapter 4 in this guide for more information on the immediate mode.

FORMAT:

UPDATE   dictionary-name   [filename]

DISCUSSION:

IF IQL does not find the file referenced in the dictionary, IQL notifies you and enters the immediate mode to create the file.

If you supply a filename, IQL uses the filename you supply rather than the filename in the dictionary. The filename must follow the ffffff.ext naming convention, where ffffff is one to six alphanumeric characters and ext is one to three alphanumeric characters.

If IQL finds that the dictionary is protected from either READ or WRITE access, IQL asks you for the password and checks it. If you respond with the READ password, IQL notifies you that IQL is opening the file for only read access.

EXAMPLE:

1. <QA> UPDATE   CUSTOMERS (RET)
    *PASSWORD: MICKEY(RET)
    (THE DIARY FILE IS QL032U.LFT)
    (AT TOP OF FILE)
    <QU> LIST CUSTOMER SALESMAN SALES ALL (RET)
    PORTABLE DEVICES INC CARLETON SCOTT   1,472.00
    ELECTROSCAN LABS BILL GLASS 21,500.00
    <QU> TOP (RET)
    <QU> CHANGE SALESMAN TO 'JIM REILLY' FOR CUSTNO 23(RET)
    SEAFOOD BROKERS INC   JIM REILLY023456780023302
    0301111020
    ...(UPDATE session continues)...
    ...(you end UPDATE session via END or EXIT)...
    <QA>

WRITE

FUNCTION:

The WRITE assistance command allows you to begin writing a new query in the current query area. IQL places you into the insert level of the appropriate text editor. Appendix E explains the IQL editor and provides a summary of some of the TOPS-20 EDITOR commands.

FORMAT:

WRITE   [query-name]

DEFAULT:

If you omit a query name, IQL does not name the current query area. If you later attempt to save or to store the current query area and omit the query name from the assistance command, IQL prompts you for a query name.

DISCUSSION:

The WRITE assistance command destroys the previous contents of the current query area.

EXAMPLES:

1. <QA> WRITE (RET)
    INPUT: QC022S.QRY
     00100   OPEN PERSONNEL.(RET)
     00200   HEADING 'NEW HIRE LIST'. (RET)
     00300   IF DATE-HIRE GEQ 770101 PRINT NAME, DEPT.(RET)
     00400   (ESC) $
    *EUN(RET)
    [QC022SC.QRY]
    <QA>

2. <QA> WRITE   NEW-HIRE-LIST(RET)
    INPUT: QC022S.QRY
     00100   OPER PERSONNEL.(NOTE ERROR)(RET)
     00200   HEADING 'NEW HIRE LIST'.(RET)
     00300   IF DATE-HIRE GEQ 770101 PRINT NAME, DEPT.
     00400   (ESC) $
    *P100 (RET)
     00100   OPER PERSONNEL.
    *SOPER(ESC)$OPEN (ESC)$.(RET)(FIX AN ERROR)
     00100   OPEN PERSONNEL.
    *EUN(RET)
    [QC022S.QRY]
    <QA>

CHAPTER 3

CREATING AND EDITING A QUERY

A query is a collection of query statements that IQL reads to generate one or more reports. The query statements direct IQL to do the following:

1. Read sequential and ISAM data files and read DBMS data base files

2. Select records or data

3. Calculate derivative information

4. Summarize the data

5. Format reports

6. Resequence the order of records

7. Write output files

With the RUN assistance command, you instruct IQL to read a query, analyze it, and generate the report(s). You can store the query into a file with the REPLACE, RUN, SAVE, and STORE assistance commands. If you use the RUN or SAVE assistance command to create an analyzed query file, you must use the EXECUTE assistance command to generate the report(s). If you use the REPLACE or STORE assistance command to create a nonanalyzed query file, you must use the RUN assistance command to generate the report(s). Refer to Chapter 2 for more information on these assistance commands.

3.1 FORMATTING DEFAULTS

IQL uses the following defaults in formatting reports if you do not override these defaults in a query:

1. Single spacing vertically

2. Three spaces between elements horizontally

3. Fifty-nine printed lines (including heading) per page

4. Date of run as mm/dd/yy at top left of each page

5. Page number at top right of each page

6. Left margin at 1

7. Right margin at 132

3.2   ORDERING QUERY STATEMENTS

IQL determines which file or schema you wish to process from the dictionaries you named in the OPEN statement. However, you do not need to use an OPEN statement if you are using IQL only as a data generator or calculator and are not reading any files. Any statements that IQL finds before the OPEN statement are executed once at the start of the query.

NOTE

> The query runs much faster if you put
> overall format statements at the
> beginning of the query.

3.3   STAGING A QUERY

The execution of a query proceeds in stages. The first stage starts with the first OPEN statement and ends with the first SORT statement or the end of the query, whichever occurs first.   The first stage takes as its primary input the original file or data base. Successive stages, if any, start with one SORT statement and end with the next SORT statement or the end of the query. Each stage takes as input the sorted output of the previous stage. This can be the entire original file sorted or an extracted and sorted subset. In this way, you can read in a file, extract some records, sort and produce one report, do a further extract, sort again, and produce another report; all without passing the entire original file more than once.

The following statements illustrate a 3-stage query:

```
OPEN GRANTS.                          (start stage 1)
HEADING 'OVERALL GRANT TOTAL'.
TOTAL AMOUNT.
IF ZIP GEQ 5000 SORT BY STATE, GRANTOR.
HEADING 'WESTERN GRANT SUMMARY'.        (start stage 2)
TALLY GRANTOR BY STATE.
TOTAL AMOUNT BY STATE.
AVERAGE AMOUNT BY STATE.
IF STATE EQ 'CA' SORT BY DESCENDING AMOUNT.
HEADING 'CALIFORNIA GRANT DETAIL'.      (start stage 3)
PRINT GRANTOR, GRANTEE, AMOUNT, DATE, DESCRIPTION.
```

3.4   QUALIFYING ITEM NAMES

If you process more than one input file in the same query and the same item name occurs in different dictionaries, you can qualify the item name by prefixing it with the name of the dictionary and a dash.   For instance if you are working with dictionaries named PAYROLL and PERSONNEL in the same query, and each contains the item named SSNO, you can identify which SSNO you mean by writing PERSONNEL-SSNO or PAYROLL-SSNO. If you do not qualify a duplicated item name, IQL uses the item value from the first dictionary named in the OPEN statement.

3.5 COMMENTING IN A QUERY

Any statement that starts with an asterisk (*) is a comment. The comment continues until IQL encounters a period.

3.6 READING A DBMS DATA BASE

To read a DBMS data base, use the FIND statement in the query where you wish to read the data base.

IQL reads different logical records from a DBMS data base and assembles the logical records into one physical record of a sequential file created by IQL. IQL assembles the physical records in the order in which you specify the logical records in the FIND statement.

If you sort the subsequent sequential file, IQL reads each of the physical records and sorts them as composite records. You cannot read from a data base after you sort it.

For IQL to read from a data base, you must make available to IQL a dictionary describing that data base. A dictionary is required (in addition to the schema) since it contains formatting information that:

1. Anticipates printing reports

2. Defines column titles

3. Edits pictures

This information is not available in a DBMS schema.

3.7 QUERY STATEMENT FORMATS

The query statements use the following statement conventions:

1. First character location

    The statement begins in the first character location of the line.

2. Space

    One or more spaces separate a statement verb from an argument. Also one or more spaces separate arguments.

3. Lowercase and uppercase characters

    An argument identified by lowercase characters implies that you are to supply a variable as indicated by the variable name.

    An argument identified by uppercase characters indicates that you are to supply the exact characters as they are shown in the text.

4. Square brackets [ ]

    An argument enclosed in square brackets indicates an optional argument. Do not use the square brackets in the query statements you write; the square brackets indicate only an

optional argument. When you do not supply the argument, the system applies a default value as explained in the argument description.

An argument not enclosed in square brackets indicates that the system requires the argument.

5. Special characters

An argument containing special characters, such as the equal sign, quotation marks, parentheses, and arithmetic symbols, requires the special characters as shown in the statement format.

6. Alphanumeric values

Enter an alphanumeric value as a string of characters including quoted spaces. If you do not furnish all the characters needed to fill the field, the system left justifies the value and inserts spaces to the right of the value. For instance if you specify SMITH to a 10-character field, the system stores the value as SMITH△△△△△, where each delta symbol indicates a space.

If the string contains spaces, enclose the string in either single quotes (') or double quotes ("). The system understands either quote mark, but be sure you end the quoted space with the same quote mark with which you started. If you use quotes in the value, set off the quoted value with the other quote mark. For instance:

        'SMITHSONIAN INSTITUTION'
        'BOB "CHIP" JONES'
        "ROBERT C. SMITH"

To enter all spaces as the value for an alphanumeric field, enter the value as ' '.

7. Numeric values

Enter a numeric value as a string of special symbols and digits. You can use the following special symbols:

        comma   ,
        dollar symbol $
        decimal point   .
        signs   + and -

Valid formats include:

        $1,234.56   +1,234.56   1,234.56   1234.56
        $1,234.56+   1,234.56+   -100   100-
          +100   -100.0   -100.00   100.00-

The system ignores dollar signs and commas in the numeric value and accepts signs in the leading or trailing position of the value. If you do not enter a sign, the system assumes a positive value.

The system uses the decimal point that you supply to vertically align the value, but the system does not actually store the decimal point. The system knows from the scale in the dictionary where the decimal point is located in the numeric field.

For integer constants, you should omit the decimal point or embed it (such as, 475 or 475.0) to avoid any possible confusion between a decimal point and a period ending a statement.

If you do not furnish enough digits to fill the field, the system provides leading and trailing zeros (after the decimal point) to fill the field.

To furnish an all zero value for a numeric item, simply enter the single digit 0.

8. Continuation

You can continue a query statement on the next line. However, do not split words. If you must split a literal, divide the literal into two literals with the first part on one line and the second part on the second line.

9. Reserved keywords

There are a number of reserved keywords with special meaning to the system. These words are listed in Chapter 1. If you must use a reserved keyword as a value, enclose the keyword in quotes.

The TODAY reserved keyword interrogates the Operating System for the current date. The format is YY/MM/DD where:

    YY is the last two digits of the year
    MM is the month of the year
    DD is the day of the month

For instance, you can use TODAY in an IF statement as part of the condition test.

The XRANDOM reserved keyword provides you with a random 10-digit number. For instance to report a sampling, you can use XRANDOM in a FIND statement.

The remainder of the chapter presents the query statements in alphabetical order.

ACCEPT

FUNCTION:

The ACCEPT statement requests item values from the terminal to allow you to change information or directives during the query run. During the generation of the report(s), IQL issues a prompt for the value and then waits for you to enter the value.

FORMATS:

ACCEPT item [... item]
ACCEPT variable [... variable]

DISCUSSION:

IQL accepts any alphanumeric or numeric data. The accepted value replaces the previous value in either the current record (in core) or the variable.

You can use the ACCEPT statement anywhere and as often as necessary.

IQL creates a prompt from the information that IQL finds in the dictionary entry for the item to be accepted. IQL uses the item name, top and bottom column titles, item type, item length, and the scale (if any).

As soon as you enter a value, IQL checks the value for the following syntax errors:

1. Alphanumeric values must be equal to or less than the length of the field.

2. Numeric values must be limited to $, + - and numbers and must be equal to or less than the length of the field. In addition, IQL fills in leading and trailing zeros, ignores the dollar sign and comma, and uses the decimal point only to vertically align the value.

A frequent use of ACCEPT is to provide search values for a specific run of a query. To provide search values, use the ACCEPT statement before the OPEN statement so you need only use one statement or use the FIRSTIME statement.

EXAMPLES:

1. Using an accepted alphavariable to control an extraction.

    ACCEPT ASTATE.
    HEADING 'PROSPECTS IN ' ASTATE.
    OPEN PROSPECTS.
    IF STATE = ASTATE
    PRINT DIRECTOR, SITE, HARDWARE.

    Might result in the prompt and response:

    *ASTATE 2A: AZ (RET)

2. Using repeated ACCEPT statements to find keyed ISAM records.

```
DISPLAY 'TO END SEARCH, ENTER "QUIT"'.
OPEN PROSPECTS.
ACCEPT ANAME.
IF ANAME = "QUIT" GO TO XT.
FIND KEY = ANAME.
DISPLAY ANAME, EQUIPMENT, SALES, PERCENT.
```

3. Using IQL like a desk calculator (no OPEN statement).

```
   DISPLAY 'TO QUIT, ENTER 0'.
   TITLES X = 'ANNUAL//SALARY'.
10ACCEPT X.
   IF X = 0 GO TO XT.
   COMPUTE X1 = X / 2080.
   DISPLAY 'ANNUAL SALARY:',X,'HOURLY SALARY:',X1.
   GO TO 10.
```

ACROSS

FUNCTION:

The ACROSS statement sets IQL to assemble more than one logical print line across a physical print line. Thus, IQL permits printing several identically formatted lines from different records across the page.

FORMAT:

ACROSS   integer

DEFAULT:

If you omit the ACROSS statement from a query, IQL prints one physical line per print statement; that is, IQL uses ACROSS 1.

DISCUSSION:

Each time IQL executes a print statement, it sets up a logical line. If you do not use the ACROSS statement, IQL prints the logical line immediately. If you use the ACROSS statement, IQL places the logical lines end to end until IQL assembles the number you specify. IQL then prints the composite line. At the end of each stage in the query, IQL prints any in-progress print lines.

If you use a horizontal spacing constant at the end of a print line, IQL reserves the required number of spaces at the end of the line before IQL appends the next line. This feature is handy for vertical alignment.

If an assembled line is too long to fit in the space between the left and right margins, IQL truncates the assembled line on the right before printing it. IQL does not display a warning message.

The ACROSS statement is frequently used for setting up mailing labels when several labels across are printed on special forms. However, the ACROSS statement is useful in many other cases for tightening up reports.

The current setting for ACROSS is carried from one stage to the next and from one report to the next in a multiple-report query. If you need a new across setting, even if it is 1, you must specifically reset it with another ACROSS statement.

CREATING AND EDITING A QUERY

EXAMPLE:

1. Mailing labels; (note paging turned off).

```
PAGING OFF.              because special form
TITLES OFF.
ACROSS 2.                two across
OPEN ALUMNI.
VSPACE 3.                three spaces between label
PRINT NAME, 5.           name is 25 characters long
VSPACE 1.                single space within label
PRINT STREET,5.          street is 25 characters long
PRINT CITY,3,STATE,5.    city is 19 characters long, state is 2
                         characters long
PRINT ZIP,25.            zip is 5 characters long
```

might give:

```
ALVIN J COWENS          JEROME C DAVIDSON
45 MEADOWBROOK BLVD.    4762 FAIRMOUNT
CLEVELAND HEIGHTS    OH SHAKER HEIGHTS        OH
44118                   44120


ERIE C HOPWOOD          WILLIAM F FRANKLIN
  ...                        ...
```

AUTHORITY

FUNCTION:

The AUTHORITY statement provides authorized passwords to permit access to protected files or data items.

FORMAT:

AUTHORITY password [... password]

DEFAULT:

If you omit the AUTHORITY statement in a query, IQL processes unprotected files or items, but does not permit you to run a query that refers to protected files or items.

DISCUSSION:

Passwords that protect files or individual items are stored (in encrypted form) as PD (Password Definition) entries in an IQL dictionary. For a discussion of these passwords, see Chapter 5.

When you use IQL with DBMS data bases, one of the passwords specified in the AUTHORITY statement must correspond to the privacy key for compile in the DBMS schema.

You can specify up to 10 passwords in an AUTHORITY statement. You can specify the passwords in any order.

If you use an exclusivity option in a dictionary you own to protect a specific data item, you must specify the exact password in the AUTHORITY statement in order to use that item.

If you do not use an exclusivity option in protecting an item, or for file level protection, the use of a particular password unlocks all file accesses or nonexclusive items protected at the same or a lower level (password number).

EXAMPLES:

1. Assume dictionary PERSONNEL is read protected by password BEAR at level 10, item PHONE is protected by exclusive password FOX at level 30, and item SALARY is further protected by the exclusive password TIGER at level 50.

    OPEN PERSONNEL.
    AUTHORITY BEAR.
    PRINT NAME, DEPT.

    In example 1, the password BEAR unlocks the file to read it. If however, the example attempted to print the item SALARY, IQL would refuse to read the item SALARY.

2. OPEN PERSONNEL.
    AUTHORITY TIGER.
    PRINT NAME, SALARY, DEPT.

    In example 2, the password TIGER permits access to SALARY.

Since TIGER is a higher ranked password than BEAR, the file is also unlocked for reading.

3. OPEN PERSONNEL
    AUTHORITY TIGER, FOX
    PRINT NAME, PHONE, SALARY

    In example 3, both passwords are required since the items PHONE and SALARY are protected by exclusive passwords.

4. Assume the dictionary ACCOUNTS is read protected by the password ALICE, copy (writing to a new output file) is protected by the password CARL, and rewrite (writing back into an ISAM file) is protected by password DENISE.

    OPEN ACCOUNTS
    AUTHORITY ALICE
    IF ACCT-CLASS = 6 COPY RECORD

    IQL does not permit this query to run. However, if either the password CARL or DENISE is included in the AUTHORITY statement, IQL would run the query.

CREATING AND EDITING A QUERY

AVERAGE

FUNCTION:

The AVERAGE statement calculates the average of a numeric item in one of two ways:

1. Over an entire report .

2. Within a specific value of one or more controlling items (breaks).

FORMAT:

AVERAGE item [BY item [... item]] [(= variable)]

DEFAULT:

If you do not specify a break item (that is, BY item), the average is calculated over all records that pass through the AVERAGE statement.

Unless you specify otherwise, each AVERAGE statement calculates the specified average and prints it at the proper time.

DISCUSSION:

The format and use of the AVERAGE STATEMENT are exactly the same as the other summary statements MAXIMUM, MINIMUM, TALLY, and TOTAL.

The AVERAGE statement is a stand-alone statement; it is not necessary for you to use TOTAL or TALLY in order to use AVERAGE.

If you do not specify a break item (that is, BY item), IQL calculates the average over all records that pass through the AVERAGE statement.

If you specify a break item, IQL calculates the average and prints it when IQL detects a change in the item.

You can specify more than one break item as long as they are minor to the one preceding it. An average is calculated for each break item. When a change in any break item occurs, IQL assumes all the break items to the right of it in the AVERAGE statement also change. The purpose of this is to prevent inadvertent overlap of minor break items over major break items. For instance if you were averaging a grade point average for each sex in each school and IQL encounters an all-girls school, AVERAGE GPA BY SEX gives a misleading answer whereas AVERAGE GPA BY SCHOOL, SEX gives the correct answer.

Since averaging is an arithmetic operation, the item you average must be numeric. However the item(s) controlling the break can be any type.

Unless the SUMPRINT OFF statement is in effect, IQL prints the average line on the report as an average title on the left and the average value on the right. IQL constructs the average title for an item from the top and bottom dictionary titles and from

the word AVG:. IQL constructs the title for a break item from
the top and bottom dictionary titles of the break item and
constructs the overall title for all the break items with the
word OVERALL.

You can give a name to the average by using the (= variable)
argument at the end of the statement.   You must begin the
variable name with either X or ZZ.

After defining the variable, you can use it like any other item.
For instance, you can include the variable in computations. One
such usage is to calculate percentages at the end of a query.
Another usage is to print the average in a print line with other
data, such as printing summaries directly under details.

EXAMPLES:

    1.   HEADING 'ORGANIZATIONAL BUDGETS'.
        OPEN BUDGET-FILE.
        AVERAGE BUDGET BY DIV, DEPT.
        AVERAGE BUDGET.

        might give:

        HOME DEPT 421 ANNUAL BUDGET AVERAGE: 23,921.46
        HOME DEPT 427 ANNUAL BUDGET AVERAGE: 37,446.00
        HOME DEPT 433 ANNUAL BUDGET AVERAGE: 12,777.50
        HOME DIV 110 ANNUAL BUDGET AVERAGE: 24,714.82

    2.   HEADING 'PRINTING AVERAGES UNDER DETAIL'.
        SUMPRINT OFF.
        OPEN LABOR-DETAIL.
        IF NEWGROUP OF DIV PRINT XACTUAL XLASTYR XCURR.
        AVERAGE ACTUAL-HRS BY DIV (= XACTUAL).
        AVERAGE LASTYR-HRS BY DIV (= XLASTYR).
        AVERAGE CURR-HRS BY DIV (= XCURR).
        PRINT ACTUAL-HRS LASTYR-HRS CURR-HRS.

COMPUTE

FUNCTION:

The COMPUTE statement calculates a new value for a numeric item.

FORMAT:

```
                    item-2
COMPUTE item-1 =    constant
                    arithmetic expression
```

DISCUSSION:

Item-1 and item-2 can be a numeric data-item or a numeric variable. A numeric data-item name must begin with a letter, other than the letter X, and can be up to 30 alphanumeric characters and dashes. IQL lists data-item names under the ITEM NAME column title of a dictionary display.

A numeric variable must begin with the letter X or letters ZZ and can be up to 30 alphanumeric characters and dashes. The initial value of a numeric variable is zero. You can initialize a numeric variable with the ACCEPT, CHANGE, RESET, and SET statements.

Constants can have a leading sign (+ or -) and can contain embedded decimal points. IQL assumes a positive constant if you omit a leading sign. For integer constants, you should omit the decimal point or embed it (that is, write 475 or 475.0 rather than 475.) to avoid any possible confusion between a decimal point and a period ending a statement.

An arithmetic expression can be any logical combination of item names, variable names, summary names, constants, arithmetic operators, and left and right parentheses. Nested parentheses are calculated from the inside to the outside.

The arithmetic operators are:

    * multiply
    / divide
    + add
    - subtract

The arithmetic operators must be preceded and followed by one or more spaces. Otherwise, IQL might confuse the operator with an item name or variable name.

IQL multiplies and divides from left to right, and then adds and subtracts from left to right. Intermediate computation results are carried to thirteen integer and five decimal places. IQL rounds off to the fifth place decimal at each step of a computation. If a computation overflows, the integer becomes too large; IQL prints and displays an error message, truncates the integer, and continues.

CREATING AND EDITING A QUERY

EXAMPLES:

1. HEADING 'HIGH VOLUME//STORES'.
   OPEN STORES.
   COMPUTE XRATIO = SALES / SQ-FT.
   IF XRATIO GR 4.99 SORT BY DESCENDING XRATIO.
   PRINT STORE-NAME, MANAGER, SALES, SQ-FT.

   The query in example 1 shows computation of a ratio and the sorting on a computed ratio.

2. HEADING 'SALES BONUSES'.
   OPEN SALESMEN.
   COMPUTE BONUS =
   (WKLY-SAL * 52) * (SALES - QUOTA) / QUOTA.
   IF BONUS LS 0 COMPUTE BONUS = 0.
   AVERAGE BONUS. TOTAL BONUS.
   PRINT SALESMAN, QUOTA, SALES, BONUS.

   The query in example 2 shows a COMPUTE statement using parentheses. BONUS is a data item in the record for the salesman file. Therefore, the query is calculating new values for BONUS to show in the report.

COPY

FUNCTION:

The COPY statement writes the current record from the primary file to a new sequential file.

FORMAT:

COPY   [RECORD]   [TO   "filename"]

DEFAULT:

If you do not use the filename option, IQL gives the file the same name as the primary input file and adds the extension .OUT. The file is written on device DSK:.

DISCUSSION:

The copied file always contains the same data format (6-bit ASCII or 7-bit ASCII) as the primary input file. If the primary input file is a DBMS data base, the copy output is 7-bit ASCII.

You can use the COPY statement after you use one or more SORT statements. In this case, the order of the copied file is not the same as the original input file but is the order after the most recent sort. There can be multiple COPY statements in a query. However, only one COPY file can be used in each stage of a query.

If you change the value of item(s) in the current record with the SET, RESET, or COMPUTE statements, the copied record contains the changed item(s). This feature is useful for making systematic changes in files.

If you use the COPY statement under control of the IF statement, IQL copies only those records where the conditions meet the relationship in the IF statement.

Unless you specify the correct password(s), IQL does not permit you to copy records when a dictionary protects the primary input file at either the read or copy level.

The copied file can be queried using the same dictionary that you used to query the original file. Be sure to furnish the new filename in the OPEN statement so that you query the copied and not the original file.

If you use the filename option and you specify a device, IQL writes out the file under the specified name and on the specified device.

IQL always writes output files in the same directory in which the job is being run.

If a file already exists with the same name as IQL is to write, IQL writes over the old file.

CREATING AND EDITING A QUERY

EXAMPLES:

    1.   OPEN ACCOUNTS.
        IF VENDOR-NO = 100 SET SALESMAN TO 'JONES'.
        COPY TO "ACCTS.SEQ".

        In example 1, IQL copies each record to the file ACCTS.SEQ.
        for each record where item VENDOR-NO is 100, IQL changed the
        salesman name to JONES in the copied record.

    2.   OPEN ACCOUNTS.
        SORT ACCOUNTS BY BALANCE.
        COPY RECORD.

        In example 2, IQL sorts and copies the records to the file
        ACCTS.OUT if the file described by the dictionary ACCOUNTS is
        named ACCTS.SEQ.

    3.   OPEN COURSES
        FIND KEY = '01000' THRU '02000'
        COPY RECORD TO COURSE-SEQ
        IF NO-STUDENTS GR 20
            SORT BY NO-STUDENTS
        * SECOND STAGE OF THE QUERY STARTS HERE
        COPY RECORD TO 'ENROLL-SEQ'

        Here the original file described by the dictionary COURSES is
        an ISAM file.   The selected records (with keys between 999
        and 2000) are copied to one sequential file, COURSE.SEQ.
        From the COURSE.SEQ file, IQL selects the records where the
        course enrollment exceeds 20, sorts the selected records, and
        copies them to a second sequential file, ENROLL.SEQ.

CREATING AND EDITING A QUERY

CREATE

FUNCTION:

The CREATE statement writes a file whose records contain only specific items and formats the records as you specify in the CREATE statement. The records that the CREATE statement writes can contain record or data items from any input file as well as working items or filler positions.

FORMAT:

CREATE ["filename"] [=] value [... value]

DEFAULT:

If you do not use the filename option, CREATE writes the created file on device DSK: under the name CREATE.OUT.

DISCUSSION:

Value can be an item or literal. The record created is formatted from left to right by putting the item value side by side in the order named in the CREATE statement. Numeric variable values are written as numeric (not binary) with 13 integer and 5 decimal places. Binary items are converted to numeric before being put in the created record.

Created files are always written in ASCII on device DSK:.

IQL always writes output files in the same directory in which the job is run.

If a file already exists with the same filename as the one IQL is to write, IQL writes over the old file.

You can use a special value of:

FILLER = integer

This special value reserves an integer number of blank spaces at that point in the created record.

EXAMPLE:

1. HEADING 'HIGH VALUE//INVENTORY ITEM//ANALYSIS'.
    TITLES X = 'ITEM//VALUE'.
    PICTURE X = '$$$,$$$.99'
    OPEN INVENTORY PRICES.
    FIND PRICE-PART = PARTNO.
    IF PRICE GR 100 COMPUTE X = PRICE * QTY
    CREATE 'COST.SEQ' = PARTNO,PRICE,QTY,FILLER=5,'VALUE',X
    PRINT PARTNO, PRICE, QTY, X.

The query reads the primary file, INVENTORY, and then looks up a corresponding record in a secondary file, PRICES. For instance, assume that PARTNO is a length of 6 and a value of 123456 and that QTY is a length of 5 and a value of 00740 in the INVENTORY file. Also assume that PRICE-PART is a length of 6 and a value of 123456 and that PRICE is a length of 5 and a value of 00198 in the PRICES file. IQL then creates a record in the COST.SEQ file as follows:

    1234560019800740ΔΔΔΔΔVALUE000000000146520000

Notice that the decimal points are not written in the record; the decimal points are implied.

CREATING AND EDITING A QUERY

DATE

FUNCTION:

The DATE statement turns report dating on or off or gives a report a specific date value to print in page headings.

FORMATS:

1. DATE [ON]
   DATE [OFF]

2. DATE value

DEFAULT:

If you do not use DATE in a query, report dating is automatically turned on - IQL puts the date of the run in report headings.

DISCUSSION:

Value can be be an item, constant, or literal. You can turn report dating on or off, or change the value of the report date at any point.

If you use DATE OFF, IQL omits the date from the page heading.

If you use DATE ON, report dating is turned on and IQL places the date the report is run, formatted as mm/dd/yy, at the top left of each page heading.

If you furnish a value in DATE, report dating is turned on and the value is printed at the top left of the page heading. The value can be a constant, a literal, or an item value. If it is a constant, it must contain six digits and it is printed with slashes inserted after the second and fourth digits. If it is a literal, it is printed exactly as furnished. If it is an item value, it is edited per the picture in the dictionary entry for that item.

RPTDATE is a synonym for DATE. It is carried over from previous versions of IQL but should not be used.

EXAMPLES:

1. DATE 060377   (prints as 06/03/77)

2. DATE ON

3. DATE OFF

4. DATE '3 JUNE 1977'   (prints as 3 JUNE 1977)

CREATING AND EDITING A QUERY

DISPLAY

FUNCTION:

>   The DISPLAY statement displays messages or reports on the
>   terminal while IQL runs the query or turns on or off an automatic
>   terminal display of reports.

FORMATS:

>   1. DISPLAY [ON]
>   DISPLAY [OFF]
>
>   2. DISPLAY value [... value]

DEFAULT:

>   If you do not use DISPLAY ON or DISPLAY OFF in a query, IQL turns
>   on the display option.

DISCUSSION:

>   Value can be an item, literal, or constant.
>
>   IQL sends single-spaced lines with no column titles to the
>   terminal. IQL displays the item values in the display line from
>   left to right as they occur in the DISPLAY statement. Unless you
>   specify otherwise (as described below), IQL places three spaces
>   horizontally between item values.
>
>   If you use an integer in the display statement, IQL places that
>   many spaces between the item values for the remainder of the
>   display line or until IQL encounters another such integer in the
>   DISPLAY statement. The use of a spacing integer overrides any
>   other horizontal spacing.
>
>   If you omit a spacing value and use the HSPACE statement in a
>   query, IQL reserves as many spaces between items as you specify
>   in the HSPACE statement.
>
>   Before IQL displays the line, IQL edits the item values as
>   specified by the picture in the dictionary (or any override
>   picture you furnish with the PICTURE statement).
>
>   Generally you use the DISPLAY statement for short prompts or
>   progress report messages. To display reports on a terminal, use
>   the PRINT statement; it contains more powerful built-in
>   formatting capability.

EXAMPLE:

>   1. DISPLAY 'ENTER CUT OFF TIME'.
>   ACCEPT XTIME.
>   OPEN RACE-RESULTS.
>   IF RACE-TIME LEQ XTIME TALLY XRACER-NAME.
>   IF XTALLY GEQ 100
>   DISPLAY 'TOO MANY BEAT THIS TIME; LOWER THE LIMIT'
>   GO TO XT.
>   GO TO NR.

CREATING AND EDITING A QUERY

FIND ITEM
(for sequential and ISAM files)

FUNCTION:

The FIND item statement performs a sequential read of a secondary file (either the second- or third- named file in the OPEN statement). Thus the FIND item statement permits you to position a file on either a specific value or the value of some other data item.

FORMATS:

1. FIND data-item = item [FROM BEGINNING]
   FIND data-item = literal [FROM BEGINNING]

2. FIND data-item = NEXT [FROM BEGINNING]

DEFAULT:

If you omit the clause, FROM BEGINNING, IQL starts reading from the current position of the file; that is, the first record read is the next record after the current record. If you use the FROM BEGINNING clause, IQL searches for the desired record from the beginning of the specified file.

DISCUSSION:

You can use the FIND item on either sequential or ISAM files. For ISAM files, the statement is useful for positioning on other than the key item.

The item named on the left of the = tells IQL which item in a file IQL is to read. The item name must be unique to the dictionary describing that file. If it is not unique, make it unique by prefixing the name of the dictionary and a dash to the item name (that is, SSNO in PERSONNEL is the same as PERSONNEL-SSNO).

IQL searches the file until IQL either satisfies the = relationship or until IQL reaches the end of the file. If IQL reaches the end of the file, IQL returns all spaces to the record and continues.

You can test to see if you are at the end of a file by using IF EOFn statement. See the discussion of the IF statement in this chapter.

EXAMPLES:

1. OPEN PROJECTS, DEPT-MASTER
   FIND DEPT-NO = DEPT FROM BEGGINNING
   PRINT PROJECT-LEADER, DEPT, DEPT-NAME

   In example 1, the file PROJECTS is automatically read by IQL. The query uses the data item DEPT in the PROJECTS file to locate a record in the DEPT-MASTER file that has a matching value of DEPT-NO. Then the query uses the department name value from item DEPT-NAME in the DEPT-MASTER record.

2. OPEN PAYROLL, PERSONNEL.
    FIND PERSONNEL-SSNO = PAYROLL-SSNO.
    PRINT NAME, SSNO, WITHHOLDING-RATE.

    In example 2, the social security number is called SSNO in
    both the PAYROLL and PERSONNEL dictionaries. In the FIND
    statement, it is necessary to qualify which SSNO to find.   In
    the PRINT statement, you can use SSNO since by that time both
    values of SSNO are the same.

FIND KEY
(for ISAM files)

FUNCTION:

The FIND KEY statement reads ISAM files randomly based on either
the value of a data item or one or more specific key values. The
FIND KEY statement permits reading on either full or partial
keys.

FORMATS:

1. FIND KEY = literal [... literal]

    FIND KEYn literal THRU literal [... literal THRU literal]

    FIND literal THRU EOF [... literal THRU EOF]

2. FIND KEY = item
    FIND KEYn = item

DISCUSSION:

If you omit the FIND KEY statement and you process an ISAM file,
IQL reads the ISAM file sequentially.   This is useful for
searching items that are not key items.

If you use FIND KEY = literal, IQL advances the file directly to
the record that contains the value of the literal as the full or
partial (leading portion) of the key.   IQL provides   an
end-of-file response if it does not find the record.

If you use a range of literals (literal THRU literal or literal
THRU EOF), IQL starts with the first record whose key is in the
range and each time IQL executes the FIND statement, IQL reads
the next record in the range. If you use several such literal
ranges, IQL goes to the first record in the next range when IQL
exhausts the range before it. The end-of-file condition occurs
only when IQL exhausts the last range. It is not necessary that
the ranges be in any specific order of key.

If you use FIND KEY = item, IQL looks for a record that has as a
key the value of the indicated item. If it does not find such a
record, it returns an all-spaces record and continues.

To read a primary file, use the FIND KEY or FIND KEY1 statement.
To read a secondary file, use the FIND KEY2 statement; and to
read a tertiary file, use the FIND KEY3 statement.

EXAMPLES:

1. OPEN PERSONNEL
    FIND KEY = '00010', '00020' THRU '00047', '00060' THRU EOF
    PRINT EMPLOYEE-NO, DEPT, DATE-HIRED

2. OPEN PERSONNEL
    FIND KEY = '00047' THRU '00096'
    FIND KEY2 = EMPLOYEE-NO
    PRINT EMPLOYEE-NO, HOURS, OVERTIME

    In example 2, IQL reads directly in the secondary file,
    PAYROLL, based on a value of item EMPLOYEE-NO from the record
    IQL just read from PERSONNEL.

CREATING AND EDITING A QUERY

FIND
(for DBMS data base)

FUNCTION:

The FIND statement reads a DBMS data base file.

FORMATS:

1. FIND recordname USING item

2. FIND NEXT   recordname RECORD OF
    FIND PRIOR recordname RECORD OF {setname } SET
    FIND FIRST recordname RECORD OF {areaname} AREA
    FIND LAST   recordname RECORD OF

3. FIND OWNER RECORD OF setname SET

4. FIND recordname RECORD

DISCUSSION:

If IQL cannot find a record, or encounters any other data base error,   IQL   returns   an   error   status   in   the   special   item ERROR-STATUS and the error count in special item ERROR-COUNT. You can test ERROR-STATUS and ERROR-COUNT for specific values. Refer to the DBMS programmer's manual (such as the TOPS-20 Data Base Management System Programmer's Procedures Manual), for the meaning of these error/status codes.   For   convenience   of reference,   307 means end of area and 326 means record not found. Often you end a query when receiving an error-status of 307, such as:

IF ERROR-STATUS = 307 GO TO XT.

If you are using the FIND FIRST or FIND NEXT statement, check the query to be sure that you do not place IQL into an unending loop.

You can suppress currency updates for ALL, RECORD, AREA or SET by appending a suppression clause to the FIND formats described above.   The format of a suppression clause is:

SUPPRESS ALL   CURRENCY UPDATES
SUPPRESS RECORD   CURRENCY UPDATES
SUPPRESS AREA   CURRENCY UPDATES
SUPPRESS SET   CURRENCY UPDATES

EXAMPLES:

1. HEADING 'BRANCH//SORTED'.
    OPEN MASTER.
    IF FIRSTIME FIND FIRST BRANCH RECORD OF AREA1 AREA
        ELSE FIND NEXT BRANCH RECORD OF AREA1 AREA.
    IF ERROR-STATUS = 307 GO TO XT.
    IF ERROR-STATUS NEQ 0 GO TO QT.
    SORT MASTER BY BRANCH-NUMBER.
    PRINT BRANCH-NUMBER, BRANCH-NAME.

    The example prints out, in branch number sequence, all the branch names and the numbers.

```
2.   OPEN STUDENT-DATA-BASE
        HEADING 'STUDENT COURSES AND TEACHERS'
        * GET THE FIRST OR NEXT STUDENT RECORD
      10IF FIRSTIME FIND FIRST STUDENT RECORD OF STU AREA
            ELSE FIND NEXT STUDENT RECORD OF STU AREA
        * IF END, IQL IS DONE; QUIT
        IF ERROR-STATUS = 307 GO TO XT
        * HERE IS A NEW STUDENT; GET THEIR FIRST COURSE RECORD
        FIND FIRST COURSE-RECORD OF STU-COURSE SET
        GO TO 30
        FIND NEXT COURSE RECORD OF STU-COURSE SET
      20*IF END OF SET, GO TO THE NEXT STUDENT
      30IF ERROR-STATUS = 307 GO TO 10
        FIND FIRST TEACHER RECORD OF COURSE-TEACH SET
        PRINT STUDENT-NAME, COURSE-NAME, TEACHER-NAME
        *GO BACK FOR THE NEXT COURSE FOR THIS STUDENT
        GO TO 20
```

Example 2 illustrates a look-up query in a DMBS data base.
Notice that the query uses a bottom-up type of look-up logic.

FORM-LINES

FUNCTION:

The FORM-LINES statement specifies the number of printed lines from perforation to perforation on the paper. IQL uses the count to position to the top of each page when output is on a terminal.

FORMAT:

FORM-LINES integer

DEFAULT:

If you omit the FORM-LINES statement in a query, IQL uses a count of 66 printed lines from one perforation to the next.

DISCUSSION:

The companion to FORM-LINES is PAGE-LINES.

Since you can dynamically change the value of the FORM-LINES setting at any point, it would be possible to specify more lines per page than the physical page (form) could hold. IQL resolves this conflict by not permitting more lines per page than lines per form. If the integer in FORM-LINES is less than the current setting of PAGE-LINES, IQL resets FORM-LINES as directed but also resets PAGE-LINES down to the new value of FORM-LINES.

FORM-LINES and PAGE-LINES are useful when you are preparing reports on a terminal or a printer and the paper is other than standard size.

EXAMPLE:

1. FORM-LINES 44.
   PAGE-LINES 36.
   OPEN MASTER.
   PRINT TITLE, PURPOSE, AMOUNT.

The query in the example produces reports with four blank lines at the top of the page, the report body formatted with 36 lines per page (including headings and column titles), and four blank lines at the bottom of the page.

CREATING AND EDITING A QUERY

GO TO

FUNCTION:

The GO TO statement transfers control from the current statement by directing IQL to do the following:

1. Go to a designated statement

2. Read the next record

3. End the query stage

4. End the query run

FORMATS:

1. GO TO integer
   GO TO NR
   GO TO XT
   GO TO QT

DISCUSSION:

If you use GO TO NR, IQL goes immediately to the statement just after the OPEN or SORT statement that begins the current stage of the query. Unless you are doing your own reading of a data base or an ISAM file, IQL does a sequential read of the next record of the primary file at this point.

If you use GO TO XT, IQL closes the current stage of the query, closes any open files, closes any in-progress summary statements, and goes to the next stage of the query. If there is no next stage, IQL returns to the assistance level. Do not use the GO TO XT statement to transfer from one report to the next report in a multiple report query. Use GO TO integer, where integer is the next REPORT statement. See the example following this discussion.

If you use GO TO QT, IQL closes any open files, closes any in-progress summary statements, and returns to the assistance level regardless of whether there is another stage to the query. The purpose of the GO TO QT statement is to permit you to terminate a multistage query at any point.

If you use GO TO integer, the integer must be a statement number used elsewhere in this stage of a query. This form of the GO TO statement is useful for routing control in a query. Be sure you do not route control from the current stage; the results are unpredictable.

Each query acts as if it had GO TO NR as the last statement in each stage; it is not necessary to write it.

CREATING AND EDITING A QUERY

EXAMPLE:

```
1.   *THE FIRST STAGE OF THE QUERY STARTS HERE.
     OPEN ACCOUNTS-RECEIVABLE.
     *IGNORE ALL BUT THE 'A' RECORDS.
     IF STATUS NOT EQ 'A' GO TO NR.
     *IGNORE RECORDS WITH AMOUNT LESS THAN $1000.
     IF AMT LS 1000 GO TO NR.
     TALLY ACCT ( = X ).
     *IF TOO MANY RECORDS QUALIFY, QUIT; THE REPORT IS TOO BIG.
     IF X GEQ 500 PRINT 'TOO MANY HITS'.
         GO TO QT.
     SORT BY DESCENDING AMT, ACCT.
     *THE SECOND STAGE OF THE QUERY STARTS HERE.
     *NOW DECIDE WHICH REPORT THE SORTED RECORD GOES INTO.
     IF ZIP LEQ 50000 GO TO 10.
     *HERE IF THE ZIP IS IN THE WEST; ZIP IS A HIGH VALUE.
     REPORT 1.
     HEADING 'WESTERN HIGH BALANCE ACCOUNTS RECEIVABLE'.
     PRINT ACCT, AMT, VENDOR, CREDIT-BAL, DATE.
     GO TO 20.
  10REPORT 2.
     HEADING 'EASTERN HIGH BALANCE ACCOUNTS RECEIVABLE'.
     PRINT ACCT, AMT, VENDOR, CREDIT-BAL, DATE.
     *BOTH REPORTS TERMINATE AT THE NEXT STATEMENT SO IQL CAN
     SORT.
  20SORT BY STATE, DESCENDING AMT.
     *THE THIRD STAGE OF THE QUERY STARTS HERE.
     HEADING 'HIGH BALANCE ACCOUNTS RECEIVABLE BY STATE'.
     PRINT STATE, ACCT, AMT.
```

Notice the use of comments in the example query. The query
is lengthy in order to illustrate the many uses of the GO TO
statement.

HEADING

FUNCTION:

The HEADING statement turns automatic page headings on or off or furnishes specific text for page headings.

FORMATS:

1. HEADING ON
    HEADING OFF

2. HEADING value [,..., value]

DEFAULT:

If you omit the HEADING statement in a query, IQL automatically turns on headings, and the heading text in the center remains blank.

DISCUSSION:

Value can be an item, constant, or literal; a constant or literal must be enclosed in parentheses. You can turn headings on or off at any point and as often as necessary.

If you turn headings on or off, IQL prints the body of the report with or without the heading as soon as IQL advances to a new page.

A page heading consists of the report date on the top left, one or more lines of report heading text in the center, and a page number on the top right. You can turn on or off any or all of these ingredients of the page heading with the DATE, HEADING, and PAGE statements.

If you use Format 2, headings are turned on.

The text in Format 2 is exactly the same as the text for a PRINT statement, with the exception that IQL starts a new line if the literal contains IQL sees a double slash (//). Note that since you can use items in the heading, part or all of the heading can come from the current record or variable values.

Each line of heading text is centered at the top of each page. The center is halfway between the current left and right margins.

If you are creating only one report, the query runs faster if you put the HEADING statement before the OPEN statement so that IQL only executes the HEADING statement once.

Headings, once set, remain set until you change them.

RESUMEHEADING, a synonym for HEADING ON, and NOHEADING, a synonym for HEADING OFF, are formats carried over from earlier releases of IQL and will not be supported in future version of IQL. HEADING ON and HEADING OFF are the recommended formats.

Also see the NEWPAGE and PAGING statements.

EXAMPLES:

(The examples are extracted from a query.)

```
1.   HEADING OFF
     OPEN ACCOUNTS
     IF FIRSTIME PRINT 'THIS IS MY OWN CUSTOM HEADING'
     PRINT ACCOUNT, BUYER, BALANCE.

2.   HEADING 'ACCOUNTS//PAYABLE//SUMMARY BY DEPT', DEPT.
     OPEN PAYABLES.
     SORT BY DEPT, PO-DATE.
     IF NEWGROUP OF DEPT NEWPAGE.
     PRINT PO-DATE, VENDOR, APPROVAL, AMOUNT.
```

HOLD

FUNCTION:

> The HOLD statement retains the value of each named item in a special holding area so you can retrieve it at any time.

FORMAT:

```
HOLD item [... item]
HOLD variable [... variable]
```

DISCUSSION:

> You can hold any numeric or alphanumeric value.

> To refer to a held value, prefix the value name with HELD-. For instance: HOLD NAME... PRINT HELD-NAME.

> A held value remains intact until you hold that specific value again. The length of a held item is exactly the same as the length of the item being held.

> The initial value of a held item is blank for an alphanumeric item and zero for a numeric item.

EXAMPLES:

```
1.  HEADING 'DUPLCATED NAMES'.
    OPEN CLIENT-FILE.
    SORT BY NAME.
    IF NAME = HELD-NAME PRINT NAME.
    HOLD NAME.

2.  HEADING 'BLIZZARD//DAMAGE//BY CITY'.
    SUMPRINT OFF.
    OPEN DAMAGE-REPORTS.
    SORT BY CITY.
    IF NEWGROUP OF CITY PRINT HELD-CITY, X.
    TOTAL DAMAGE BY CITY ( = X ).
    HOLD CITY.
```

HSPACE

FUNCTION:

The HSPACE statement sets the default horizontal spacing
increment between items when IQL sets up a print or display line.

FORMAT:

HSPACE integer

DEFAULT:

If you omit the HSPACE statement in a query, IQL places three
spaces between items in a print or display line.

DISCUSSION:

You can use HSPACE anyplace in a query, and as often as
necessary.

If you use specific horizontal spacing in a PRINT statement, it
overrides the current HSPACE setting. The HSPACE value remains
unchanged.

If you change HSPACE in one report of a multiple report query,
IQL retains the setting from one report to the next report,
unless you change it.

If you are only using one HSPACE for all reports, place the
HSPACE statement before the OPEN statement so it is executed only
once. IQL runs faster if the HSPACE statement follows the OPEN
statement.

EXAMPLES:

1. HSPACE 5.
    LMARGIN 15.
    OPEN PUBLICATIONS.
    PRINT LIB-CONGRESS-NO, AUTHOR, TITLE.

2. OPEN PUBLICATIONS.
    REPORT 1.
    HSPACE 3.   (to reset from next report)
    PRINT AUTHOR, TITLE.
    REPORT 2.
    HSPACE 10.
    IF CAT = 'REF' PRINT PUB-DATE,TITLE,PUBLISHER.

FUNCTION:

The IF statement specifies a condition for IQL to test. Depending on whether the condition is satisfied or not, the query statements that follow are performed. Control of the IF ends when IQL encounters a period.

FORMATS:

1. IF value1 relation value2 [AND ...] [ELSE ...]
                             [OR ...] [ELSE ...]

2. IF FIRSTIME ... [ELSE ...]
    IF LASTIME ... [ELSE ...]
    IF BOF1 ... [ELSE ...]
    IF BOF2 ... [ELSE ...]
    IF BOF3 ... [ELSE ...]
    IF EOF1 ... [ELSE ...]
    IF EOF2 ... [ELSE ...]
    IF EOF3 ... [ELSE ...]

DISCUSSION:

Value1 and value2 can be one or more items, constants, or literals separated by a comma.

You can write relationships in any of the following ways:

Relation	IQL Accepted Words/Symbols			
equal	EQ	EQUALS	=	IS
not equal	NQ	NE	<>	NEQ
less than	LS	LT	<	LESS
greater than	GR	GT	>	GREATER
less than or equal to	LEQ	LE	<=	LQ
greater than or equal to	GEQ	GE	>=	GQ

Note that you can use NOT to reverse a relationship. For instance: NOT EQ is the same as NE; and NOT LEQ is the same as GR.

You can combine simple relationships to make a complex relationship by joining them with AND or OR clauses. You can use parentheses to clarify complex relationships. IQL evaluates all OR clauses before any AND clauses. You cannot nest IF statements; that is, IQL does not permit the construct of IF ... IF ... ELSE ... ELSE.

If you use the ELSE clause, IQL performs the statement(s) that follow the ELSE clause if the condition preceding the clause is not satisfied.

If you are comparing two numbers, the decimal points are lined up and any necessary leading or trailing zeros are supplied before the comparison is made.

If you are comparing two alphanumeric items or literals, any necessary trailing spaces are added to the shorter item before the comparison is made.

EXAMPLES:

1. Cutting off a query.

    ```
    HEADING 'SALES DETAIL'.
    OPEN SALES.
    PRINT PO-NO,ACCOUNT.,LINE,ITEM,QTY,AMOUNT.
    COMPUTE X = X + 1. IF X GEQ 50 GO TO XT.
    ```

2. Comparing against a held item; exception report.

    ```
    HEADING 'DUPLICATES IN INVENTORY FILE'.
    OPEN INVENTORY.
    SORT BY PART-NO.
    IF FIRSTIME HOLD PART-NO GO TO NR.
    IF PART-NO EQ HELD-PART-NO
        PRINT PART-NO GO TO NR.
    HOLD PART-NO.
    ```

3. Using AND clauses and OR clauses.

    ```
    HEADING 'CANDIDATES FOR EXECUTIVE BOARD'.
    OPEN PERSONNEL.
    IF JOB = 'EXC' OR JOB EQ 'MGR' GO TO 10.
    IF SALARY-CLASS GEQ 7 AND YR-HR LEQ 68 GO TO 10.
    GO TO NR.
    10 PRINT NAME,JOB,DIV,DEPT,SALARY-CLASS,YR-HR.
    ```

4. Using parentheses and ELSE.

    ```
    HEADING 'REGION 9 SPECIAL ATTENTION ACCOUNTS'.
    OPEN ACCOUNTS.
    IF ( LASTYR GEQ 50000 OR THISYR GEQ 25000 OR
        ( RATING = 'A' AND PROJECTION GEQ 50000 ) ) AND
        ( STATE EQ 'CA', 'OR', 'WA', 'UT' )
          PRINT CUSTNAME,STATE,LASTYR,THISYR
          TOTAL LASTYR-SALES
        ELSE TALLY CUSTNAME.
    ```

LMARGIN

FUNCTION:

The LMARGIN statement sets the left margin for printing reports.

FORMAT:

LMARGIN integer

DEFAULT:

If you omit the LMARGIN statement in a query, IQL sets the left margin to 1.

DISCUSSION:

You can use LMARGIN anyplace in a query, and as often as necessary.

LMARGIN settings are carried from one multiple report to the next unless you use LMARGIN to change them.

LMARGIN affects page headings, summary lines, and detail print lines.

EXAMPLES:

    1. LMARGIN 15.
        HEADING 'MEMBER EXTRACT'.
        OPEN MEMBER-FILE.
        IF SUBSCRIPTION GEQ 1000
            PRINT NAME, DATE, SUBSCRIPTION.

    2. OPEN SALES.
        SORT BY STATE.
        REPORT 1.
        HEADING 'SALES DETAIL'.
        LMARGIN 1.
        PRINT ACCT, SALESMAN, GROSS, PROJECTION.
        TOTAL GROSS.   TOTAL PROJECTION.
        REPORT 2.
        HEADING 'HIGH VOLUME//SUMMARY'.
        LMARGIN 10.
        IF GROSS GEQ 10000
            TALLY ACCOUNT BY STATE
            TOTAL GROSS BY STATE.

    The LMARGIN 1 statement is necessary to reset the query from the LMARGIN 10 statement in report 2.

MAXIMUM

FUNCTION:

The MAXIMUM statement calculates the maximum value of a numeric
item in one of two ways:

1. Over an entire report.

2. Within a specific value of one or more controlling items
    (breaks).

FORMAT:

MAXIMUM item [BY item [... item]] [(= variable)]

DEFAULTS:

If you do not specify a break item (that is, BY item), the
maximum is calculated over all records that pass through the
MAXIMUM statement.

Unless you specify otherwise, each maximum statement   both
calculates the specified maximum and prints it at the proper
time.

DISCUSSION:

The format and use of the MAXIMUM statement are exactly the same
as the other summary statements AVERAGE, MINIMUM, TALLY, and
TOTAL.

If you specify a break item, the maximum is calculated and
printed on the change in the indicated item. If you specify more
than one break item, MAXIMUM breaks on a change in any of the
break items.

Since taking a maximum is an arithmetic operation, the item for
which you get the maximum must be numeric. However, the item
controlling the break can be any type.

Unless the SUMPRINT OFF statement is in effect, IQL prints the
maximum line on the report as a maximum title on the left and the
maximum value on the right. IQL constructs the title for an item
from the top and bottom dictionary titles and from the word MAX:.
IQL constructs the title for a break item from the top and bottom
dictionary titles of the break item and constructs the overall
title for all the break items with the word OVERALL.

You can give a name to the maximum by using the (= variable)
argument at the end of the statement.   You must begin the
variable name with either X or ZZ.

After defining the variable argument, you can use it like any
other item.   For instance, you can include the variable in
computations. The most frequent use of giving a name to a
maximum is so that you can use it in a print line with other
items, often for printing summaries under detail.

EXAMPLES:

    1. OPEN PAYROLL.
        SORT BY JOB-CLASS.
        MAXIMUM SALARY BY JOB-CLASS.
        MAXIMUM SALARY.

    2. OPEN TEAM-SCORES.
        MINIMUM SCORE (= XMIN).
        MAXIMUM SCORE (= XMAX).
        IF LASTTIME
            COMPUTE XPCT = 100 * (XMAX - XMIN) / XMIN
            PRINT 'MAX:MIN PCT ' XPCT.

FUNCTION:

The MINIMUM statement calculates the minimum value of a numeric item in one of two ways:

1. Over an entire report.

2. Within a specific value of one or more controlling items (breaks).

FORMAT:

MINIMUM item [BY item [... item]] [(= variable)]

DEFAULT:

If you do not specify a break item (that is, BY item), the minimum is calculated over all records that pass through the MINIMUM statement.

Unless you specify otherwise, each MINIMUM statement both calculates the specified minimums and prints them at the proper time.

DISCUSSION:

The format and use of the MINIMUM statement are exactly the same as the other summary statements AVERAGE, MAXIMUM, TALLY, and TOTAL.

If you specify a break item, the minimum is calculated and printed on the change in the indicated item. If you specify more than one break item, MINIMUM breaks on a change in any of the break items.

Since taking a minimum is an arithmetic operation, the item you take the minimum of must be numeric. However the break item can be any type.

Unless the SUMPRINT OFF statement is in effect, IQL prints the minimum line on the report as a minimum title on the left and the minimum value on the right. IQL constructs the title for an item from the top and bottom dictionary titles and from the word MIN:. IQL constructs the title for a break item from the top and bottom dictionary titles of the break item and constructs the overall title for all the break items with the word OVERALL.

You can give a name to the minimum by using the (= variable) argument at the end of the statement. You must begin the variable name with either X or ZZ.

After defining the variable, you can use it like any other item. For instance, you can include the variable in computations. One frequent use of naming the minimum is for calculating percentages. Another is for using the minimum in a print line with other items, often for printing minimums under detail.

EXAMPLES:

1. OPEN ACCOUNTS.
    MINIMUM BALANCE BY ACCOUNT-CLASS.
    TOTAL BALANCE BY ACCOUNT-CLASS.
    MINIMUM BALANCE.   MAXIMUM BALANCE.
    TOTAL BALANCE.   AVERAGE BALANCE.

2. OPEN INVENTORY.
    MINIMUM QUANTITY (= ZZQTY).
    MINIMUM VALUE (= ZZVAL).
    IF NEWGROUP OF CLASS
        PRINT ZZQTY, ZZVAL.
    PRINT QUANTITY,VALUE,CLASS.

NEWPAGE

FUNCTION:

The NEWPAGE statement advances immediately to a new page of the
current report.

FORMAT:

NEWPAGE

DISCUSSION:

Any time you use NEWPAGE, IQL advances immediately to a new page
of the report regardless of whether paging is turned on or off.

When IQL advances a page, IQL prints a page heading, which
consists of the report date on the left, the page number on the
right, and one or more lines of report heading prose in the
center.   IQL does this before starting to print the body of the
report on that page. You can individually turn off or set any of
these ingredients of the page heading.

Also see HEADING and PAGING descriptions.

EXAMPLE

1. HEADING 'BUDGET SUMMARY BY DEPARTMENT'.
    OPEN BUDGET-FILE.
    SORT BY DEPT, ACCOUNT.
    IF NEWGROUP OF DEPT NEWPAGE.
    TOTAL BUDGET-AMOUNT BY ACCOUNT.
    TOTAL BUDGET AMOUNT BY DEPT.

The example uses the NEWPAGE statement in an IF statement.

FUNCTION:

The OPEN statement informs IQL which dictionaries and files to use in a query.

FORMAT:

OPEN dictionary["file"] [dictionary["file"]] dictionary["file"]]

DEFAULT:

If you omit a filename in quotes after a dictionary name, IQL reads from the file whose name it finds in the dictionary.

DISCUSSION:

Only one OPEN statement can be used in a query. There must be an OPEN statement before any SORT statements.

You can use from one to three dictionary names in an OPEN statement, depending on how many files you wish to process in the query.

The file described by the first dictionary in the OPEN statement is called the primary file. This file controls the current phase of the query. If it is a sequential file, IQL automatically reads it, the read coming immediately after the OPEN statement. If the primary file is an indexed sequential (ISAM) file and you do not do your own reading with a FIND KEY statement, IQL automatically reads it sequentially.

If you use more than one dictionary name in OPEN statement, the files described by the second and third dictionaries are secondary files. You must do your own reading of these files by appropriate use of FIND item or FIND KEYn. Normally secondary files are read under control of information you get from the primary file.

When IQL comes to the last statement in the current stage or it encounters a GO TO NR, IQL goes directly to the statement just after the OPEN. If it is reading sequentially as described above, it reads the next record before carrying out the statement.

Any statements that come before the OPEN statement are carried out only once at the beginning of the query and before any records are read. The query runs faster if you put one-time formatting statements at the beginning of the query when you are not going to change formats during the run.

EXAMPLES:

1. OPEN PERSONNEL.
    PRINT SSNO, DATE-HIRE.

    In example 1, IQL first looks up dictionary PERSONNEL and then determines which data file to physically open.

2. OPEN PERSONNEL '/EMPLOY.IDX'.
   PRINT SSNO, DATE-HIRE.

   In example 2, IQL overrides the filename in the dictionary
   PERSONNEL and physically opens the EMPLOY.IDX data file.
   Notice, however, that the attributes (such as the word length
   and the blocking factor) are taken from the dictionary
   personnel.

3. OPEN PERSONNEL JOB-TABLE 'JOBS.IDX' COLLEGE-TABLE.
   FIND KEY2 =PERS-JOB.
   FIND COLLEGE-ID = PERS-COLLEGE-ID FROM BEGINNING.
   PRINT SSNO, NAME, JOB-TITLE, COLLEGE-NAME.

   In example 3, IQL processes three files. For the first and
   third files, IQL uses the files named in the dictionaries
   PERSONNEL and COLLEGE-TABLE. However, for JOB-TABLE, IQL
   ignores the filename in the dictionary and opens the file
   named JOBS.IDX. The first file is read automatically by IQL.
   The second dictionary, JOB-TABLE, describes an ISAM file.
   IQL reads the second file randomly with the FIND KEY2 =
   PERS-JOB statement. IQL reads the second file for a record
   whose key matches the item PERS-JOB from the record in the
   file named in the PERSONNEL dictionary. For the third data
   file, IQL reads the sequential file from the beginning of the
   file and searchs for a record where the item COLLEGE-ID
   matches the item PERS-COLLEGE-ID, which is read from the file
   named in the PERSONNEL dictionary. Finally, IQL prints the
   information from each of the three files.

FUNCTION:

The PAGE statement specifies a new page number that overrides automatic page numbering.

FORMAT:

PAGE value

DEFAULT:

If you do not use PAGE in a query, IQL starts page numbers at 1 and automatically increases the page number each time IQL advances a page.

DISCUSSION:

Value can be an integer, variable, or item. Unpredictable results occur if you do not assign the variable a numeric value before a PAGE variable statement.

EXAMPLE:

1.   PAGE 10.
     OPEN BANKS.
     PRINT BANK-NAME, TOTAL-DEPOSITS.

IQL assigns the first page as page 10.

FUNCTION:

    The PAGE-LINES statement specifies the maximum number of printed
    lines IQL places on each page of the report.

FORMAT:

    PAGE-LINES integer

DEFUALT:

    If you do not use PAGE-LINES in a query, IQL uses a count of 58
    printed lines as the maximum for each page.

DISCUSSION:

    The companion to PAGE-LINES is FORM-LINES, which specifies the
    number of lines from perforation to perforation of the physical
    page.

    The difference between FORM-LINES and PAGE-LINES specifies the
    total number of blank lines at the top and bottom of each page.

    Since you can dynamically change the value of the PAGE-LINES
    setting at any point in the query, it would be possible to
    specify more lines per page than the physical page. IQL resolves
    this conflict by not permitting more lines per page than lines
    per form. If the integer in PAGE-LINES is greater than the
    current value of FORM-LINES, IQL resets the PAGE-LINES as
    directed but also resets FORM-LINES up to the new value of
    PAGE-LINES.

    FORM-LINES and PAGE-LINES are useful when you are preparing
    reports on a printer or terminal and the paper is other than
    standard size.

EXAMPLE:

    1. FORM-LINES 44.
        PAGE-LINES 36.
        OPEN MASTER.
        PRINT TITLE, PURPOSE, AMOUNT.

        The query in example 1 produces reports with four blank lines
        at the top of the page, the report body formatted with 36
        lines per page (including headings and column titles), and
        four blank lines at the bottom of the page.

PAGING

FUNCTION:

       The PAGING statement turns on or off automatic report paging.

FORMAT:

       PAGING [ON]
       PAGING [OFF]

DEFAULT:

       If you do not use PAGING in a query, paging is automatically
       turned on. IQL goes to a new page when it reaches the bottom of
       the current page.

DISCUSSION:

       You can turn paging on or off at any point, and as often as
       necessary.

       If you have turned paging off, IQL prints past the perforation
       (if any) on the paper.

       When IQL pages, it prints a page heading consisting of the report
       date on the left, the page number on the right, and one or more
       lines of report heading prose before it starts to print the body
       of the report. You can individually turn off or set any or all
       of these ingredients of the page heading.

       If paging is off, then headings are automatically off also. If
       you want a heading at the beginning of the run, use PAGING OFF
       after your first PRINT statement.

       RESUMEPAGING, a synonym for PAGING ON, and NOPAGING, a synonym
       for PAGING OFF, are formats carried over from earlier releases of
       IQL and will not be supported in future versions of IQL.

       Also see HEADING, NEWPAGE, RPTDATE, RPTHEAD statements.

EXAMPLE:

       1. OPEN ACCOUNTS.
           REPORT 1.
           PAGING OFF.
           ACROSS 3.
           PRINT ACCOUNT-NAME,3.
           PRINT CITY,1,STATE,ZIP,3.
           REPORT 2.
           PAGING ON.
           HEADING "SUMMARY OF BALANCES BY ZIP".
           TOTAL BALANCE BY ZIP.
           PRINT ACCOUNT-NAME, BALANCE.

           The example is taken from a query. The first report prints
           three columns of information without any page numbers. The
           second report provides a report heading and page numbers.

PICTURE

FUNCTION:

The PICTURE statement furnishes a picture for one or more items to override dictionary pictures or to furnish pictures for variables.

FORMAT:

PICTURE item = "picture" [... item = "picture"]

DISCUSSION:

You can use PICTURE anywhere and repeatedly. However, usual usage is to set a picture for a full run; in this case, put the PICTURE statement before the OPEN statement so it is set up only once.

The rules IQL uses for editing a picture are the same for pictures in a dictionary and pictures in a query.

1. Pictures for alphanumeric items can contain any character. The character X indicates a substitution position – a character from the item is substituted for each X. IQL fills the characters by proceeding from left to right from the item value. If there are not enough substitution positions in the picture, IQL truncates the item; if there are too many substitution positions, IQL fills in spaces.

2. Pictures for numeric items can contain only the characters 9 Z $ ( ) , . S or R. The characters 9 Z $ R and (designate substitution positions; their specific functions are described below. In editing a numeric item, IQL lines up the item with the decimal point (if any), and then substitutes from the decimal point outwards.

3. Each 9 is a substitution position – the corresponding digit from the item value is filled in. A 9 stops any zero suppression or floating characters from the left, and a 9 stops rounding via R from the right.

4. Z can be used only on the left of the picture. Z positions indicate where zero suppression is to be used. Only a single S or $ can be located to the left of the leftmost Z, and only commas or decimal points can be used between Zs.

5. Two or more adjacent $ characters on the left of the decimal point indicate floating dollar sign editing; a $ character is placed to the left of the most significant digit. There can be no character to the left of the $ character in the picture and only decimal points or commas can be used between the $ characters. All the $ characters, except the leftmost character, are substitution positions.

6. Two or more adjacent (characters on the left of the decimal
    point indicate a floating left parenthesis. This works the
    same as the floating $ character, except that the parentheses
    are printed only if the quantity is negative. There must be
    a single) character on the right of the picture.

7. The R characters at the right of the picture indicate
    rounding. IQL rounds from right to left until it encounters
    no more R characters. The R positions are not printed.

EXAMPLES:

Item Values	IQL Picture	Result
MARLBORO	XXXXXXX	MARLBORO
MARLBORO	XXXX	MARL
MARLBORO	XXXXXXXXXX	MARLBOROBB
MARLBORO	XX-XX-XXOOOXX	MA-RL-BOOOORO
00123	99999	00123
00123	ZZZZ9	123
00123	Z9999	0123
00123	$$$$9	$123
00123	$99999	$00123
467939	9999.99	4679.39
467939	Z,ZZZ.ZZ	4,679.39
467939	9999	4,679
467939	999	*679 (showing overflow)
092173^84	$$$$,$$$.99	$92,173.84
092173^84	ZZZ,ZZZ.RR	92,174
092173^84	$$$RRR.RR	$92
092946^72	$$$RRR.RR	$93
092946^72 MINUS	SZZZ,ZZZ.99	-092,946.72
000000^00	ZZZ,ZZZ.ZZ	
092946^72 MINUS	((((,(((.99)	(92,946.72)
092946^72	((((,(((.99)	92,946.72
092173^84	$$$$,$$$.99	$92,173.84

PRINT

FUNCTION:

The PRINT statement specifies a print line in a report.

FORMATS:

PRINT value [, ..., value]

DEFAULTS:

If you do not specify otherwise, IQL turns titling on, uses three spaces horizontally between item values, and single spaces vertically.

DISCUSSION:

Value can be an item, literal, or integer. Item values are laid out in the print line from left to right as they occur in your PRINT statement.

Item values are edited according to their picture in the dictionary (or any override picture you use in the query) before being put in the print line.

If you use an integer in the statement, the integer overrides standard spacing for the PRINT statement. IQL sets the integer to the horizontal spacing increment for the remainder of the print statement or until you use another such spacing integer. For instance, if you use HSPACE 2 and the print statement:

PRINT NAME, STREET, 5, PHONE, TITLE, 1, YEAR

IQL provides two spaces between NAME and STREET, five spaces between STREET and PHONE, five spaces between PHONE and TITLE, and one space between TITLE and YEAR. If you do not use a spacing integer and use HSPACE in your query, as many spaces are left between item values as you designate with HSPACE.

If your print line is too long for the space between your current left and right margins, IQL truncates the print line on the right and continues. No error message is issued.

If titling is on, IQL prints two lines of column titles at the top of each new page. In laying out space for each item horizontally, IQL uses the longer of the edited item length or the length of the column title. Column titles for alphanumeric items are left justified. Column titles for numeric items are right justified.

If titling is on, and the current print line is from a different PRINT statement than the preceding line, IQL prints titles before printing the body of the current line.

If titling is off, IQL prints only the current line. In laying out this line, IQL ignores the title lengths. Note that if you print a line, turn titling off, and print the line again. There can be a difference in alignment since the length of the column titles can affect the first line but not the second line.

Vertical spacing is done just before the print line (and title if any) is printed.

EXAMPLES:

    1.   OPEN PERSONNEL.
        PRINT LAST-NAME,FIRST-NAME,INITIAL,OFFICE-PHONE.

    2.   SUMPRINT OFF.
        PICTURE X = '$$$$,$$$.99'.
        OPEN RETAIL-OUTLETS.
        COMPUTE X = SALES / SQ-FT.
        TOTAL SALES (= XSALES).
        PRINT STORE,MGR,'SALES/SQ FT RATIO:',1,X.
        IF LASTTIME PRINT 'TOTAL SALES:', XSALES.

REPORT

FUNCTION:

The REPORT statement denotes a specific report in a multiple
report query run. All print, report formatting, or heading
statements between this REPORT statement and the next REPORT
statement (or the end of the query) pertain to that specific
report.

FORMAT:

REPORT integer

DEFAULT:

If you do not use REPORT anywhere in your query, IQL produces a
single report and acts as if you had given REPORT 1 at the
beginning of the query.

DISCUSSION:

You can specify up to 99 reports in a query, that is, the integer
can take any value from 1 to 99. Report number 1 is always
routed directly to the printer. All other reports are spooled
out to a disk file and later separated and printed or displayed
at the end of the run.

When you use a report format setting such as LMARGIN 10 or a
heading such as HEADING "EXCEPTION REPORT", IQL uses the same
setting in each subsequent report until you change it.

EXAMPLE:

```
1.  OPEN TICKETS.
    REPORT 1.
    RMARGIN 72.
    HEADING 'SEPTEMBER//TICKETS'.
    IF MONTH = 'SEPT'
        PRINT NAME, DESTINATION, AIRLINE, FLIGHT, AMT.
    REPORT 2.
    RMARGIN 132.
    HEADING 'BURMUDA EXCURSIONS - FALL'.
    IF DESTINATION = 'BERMUDA'
        AND MONTH = 'SEPT', 'OCT', 'NOV'
            PRINT NAME, AGENT, HOTEL, DATE, PLAN.
```

RESET

FUNCTION:

The RESET statement resets the value of an item or variable.

FORMAT:

    RESET item [... item]
    RESET variable [... variable]

DISCUSSION:

    IQL resets items or variables to spaces. For instance,

        RESET ACCOUNT-NAME

    is equivalent to:

        SET ACCOUNT-NAME TO ' '.

    IQL resets numeric items or variables to zero. For instance,

        RESET SALARY

    is equivalent to:

        COMPUTE SALARY = 0.

EXAMPLE:

    1. HEADING 'BONUS PROJECTION'.
        OPEN SALES-FILE.
        COMPUTE XBONUS = SALARY * (SALES - QUOTA) / QUOTA.
        IF XBONUS LEQ 0 RESET XBONUS.
        PRINT NAME, SALES, QUOTA, XBONUS.

REWRITE

FUNCTION:

The REWRITE statement replaces an existing record in an ISAM file.

FORMAT:

REWRITE [RECORD]

DISCUSSION:

The record rewritten is the last one read from the ISAM file. You can use REWRITE even if you are reading the ISAM file sequentially (that is, you did not use FIND KEY in your query).

If you have changed information in the record, the changed values are rewritten into the file as part of the rewritten record.

You should be careful in using REWRITE since it writes over the original information in the ISAM file. It is a good idea to protect ISAM files with a fairly high level password in the dictionary at the FD rewrite password level.

EXAMPLE:

1. AUTHORITY Q2R3YA.
   OPEN PERSONNEL.
   FIND KEY = '00010', '00025' THRU '00047'.
   SET ASSIGNMENT TO 'SUPERVISOR'.
   REWRITE RECORD.

FUNCTION:

The RMARGIN statement sets the right margin for printing reports.

FORMAT:

RMARGIN integer

DEFAULTS:

If you do not use RMARGIN in your query, IQL uses a right margin of 132.

DISCUSSION:

You can use RMARGIN any place in your query, and as often as necessary.

RMARGIN settings are carried from one multiple report to the next unless you use RMARGIN to change them.

RMARGIN affects page headings, summary lines, and detail print lines.

EXAMPLES:

```
1.   RMARGIN 72.
     HEADING 'MEMBER EXTRACT'.
     OPEN MEMBER-FILE.
     IF SUBSCRIPTION GEQ 1000
         PRINT NAME, DATE SUBSCRIPTION.

2.   OPEN SALES.
     SORT BY STATE.
     REPORT 1.
     HEADING 'SALES DETAIL'.
     RMARGIN 132.            (to reset from 2nd report)
     PRINT ACCT SALESMAN GROSS PROJECTION.
     REPORT 2.
     HEADING 'HIGH VOLUME//SUMMARY'.
     RMARGIN 72.
     IF GROSS GEQ 1000
         TALLY ACCOUNT BY STATE
         TOTAL GROSS BY STATE.
```

FUNCTION:

The SET statement sets a new value into an item. The value can come from another item, constant, or literal.

FORMAT:

SET value TO value [... value TO value]

DISCUSSION:

Value can be an item, constant, or literal. If the item being set is in a record, the new value replaces the old value in the record IQL is working with but not in the file itself. However if you subsequently write out the record, it contains the value.

When a numeric item is set to a numeric value, the decimal points are lined up first, then leading and/or trailing zeros are supplied if necessary. If the integer or decimal portions of the supplied value are too long, they are truncated with no error message.

When an alphanumeric item is set to an alphanumeric value, the supplied value is left justified with blank right fill if necessary. If the value is too long, it is truncated on the right with no error message.

SET X TO ... is equivalent to COMPUTE X = ...

You can set alphanumeric items to numeric values. The integer part of the numeric item is left justified in the alphanumeric item with blank right fill if necessary.

You can also set numeric items to alphanumeric values, but this is only legitimate if the alphanumeric value is all digits. As many digits as necessary are taken from the left of the alphanumeric value to fill the integer portion of the numeric item. IQL zeroes the fractional portion, if any.

You should avoid mixing item types, that is, setting alphanumeric to numeric and vice versa. If done often, these operations are expensive in computer time.

EXAMPLE:

1. OPEN SALES-MASTER.
    FIND KEY = '456701'
    SET QUOTA TO 200,000 REGION TO 'NORTHWEST'.
    REWRITE RECORD.

SORT

FUNCTION:

The SORT statement sorts the primary input file on one or more keys. The keys can be data items or variables and can be sorted in ascending or descending order.

FORMAT:

SORT BY [ASCENDING]   value [,... [ASCENDING]   value]
        [DESCENDING]   [DESCENDING]

DEFAULT:

If you do not specify ASCENDING or DESCENDING, IQL sorts on an ascending order.

DISCUSSION:

Value can be an item or a variable. There is no maximum number of sort keys that can be used other than that the number of characters in all sort keys -taken together- cannot exceed the total length of the record.

You can sort on calculated variable values. However, the calculated value does not pass through the SORT statement. For instance in example 2 below, if you want to print XRATIO, you must either recompute it after the sort or use the HOLD statement to save XRATIO before the SORT statement.

EXAMPLES:

1. OPEN CONGLOMERATE.
    SORT BY ASCENDING UNIT-NO DESCENDING SALES.
    TOTAL SALES BY UNIT-NO.

2. OPEN STORES.
    COMPUTE XRATIO = SALES / SQ-FEET.
    SORT BY XRATIO, SALES.
    PRINT SALES, SQ-FEET, STORE-NAME.

3. OPEN CANDIDATE-FILE.
    COMPUTE XTOTAL = POLL-PCT * PRECINCT-POPULATION.
    SORT BY PRECINCT, XTOTAL.
    IF NEWGROUP OF PRECINCT
         PRINT PRECINCT, CANDIDATE
       ELSE PRINT CANDIDATE.

SUMPRINT

FUNCTION:

The SUMPRINT statement turns on or off the summary print line in
a report.

FORMAT:

SUMPRINT [ON]
SUMPRINT [OFF]

DEFAULT:

Unless you specify otherwise, IQL automatically determines the
explanatory text to accompany the printed lines generated by the
AVERAGE, MAXIMUM, MINIMUM, TOTAL, and TALLY statements.

DISCUSSION:

If you use the SUMPRINT OFF statement, IQL calculates summary
quantities (such as averages) but IQL does not automatically
print in a report. If you do not use the SUMPRINT OFF statement
and you want to print a summary quantity, you must give the
summary quantity a name and then use that name in a print or
display statement.

EXAMPLE:

    1.   HEADING 'PRINTING AVERAGES UNDER DETAIL'.
         SUMPRINT OFF.
         OPEN LABOR-DETAIL.
         IF NEWGROUP OF DIV PRINT XACTUAL XLASTYR XCURR.
         AVERAGE ACTUAL-HRS BY DIV (= XACTUAL).
         AVERAGE LASTYR-HRS BY DIV (= XLASTYR).
         AVERAGE CURR-HRS BY DIV (= XCURR).
         PRINT ACTUAL-HRS LASTYR-HRS CURR-HRS.

TALLY

FUNCTION:

The TALLY statement counts the occurrences of an item and enters the total in one of two ways:

1. Over an entire report.

2. Within a specific value of one or more controlling items (breaks).

FORMAT:

TALLY item [BY item [... item]] [(= variable)]

DEFAULT:

If you do not specify a break item (that is, BY item), the tally is calculated over all records that pass through the TALLY statement.

Unless you specify otherwise, each TALLY statement calculates the specified tally and prints it at the proper time.

DISCUSSION:

The format and use of the TALLY statement are exactly the same as the other susmmary statements AVERAGE, MAXIMUM, MINIMUM, and TOTAL.

If you do not specify a break item (that is, BY item), the tally is calculated overall records that pass through the TALLY statement.

If you specify a break item, the tally is calculated and printed on the change in the indicated item. If you specify more than one break item, TALLY breaks on a change in any of the indicated break items.

You can tally any type of item or break item, since TALLY merely runs an internal counter that increments by one each time IQL encounters the TALLY statement.

Unless the SUMPRINT OFF statement is in effect, IQL prints the tally line on the report as a tally title on the left and the tally value on the right. IQL constructs the title for an item from the top and bottom dictionary titles and from the word TALLY:. IQL constructs the title for a break item from the top and bottom dictionary titles of the break item and constructs the overall title for all the break items with the word OVERALL.

You can give a name to the tally by using the (= variable) argument at the end of the statement. You must begin the variable name with either X or ZZ.

After defining the variable, you can use it like any other item. For instance, you can include the variable in computations.

EXAMPLE:

1. OPEN SCHOOLS.
    TALLY STUDENT BY CLASS.

TITLES

FUNCTION:

The TITLES statement turns on or off the displaying of column
titles or furnishes specific column titles for designated items.
You can provide titles for working items or override the
dictionary titles.

FORMATS:

1. TITLES ON
    TITLES OFF

2. TITLES item = literal [... item = literal]

DEFAULTS:

If you do not use TITLES in your query, titling is automatically
turned on.

For data items, column titles are obtained from their description
in the dictionary.

For working items, the name of the item is used as the column
title.

DISCUSSION:

When IQL is about to print a line, it looks to see if this line
is from a different print statement than the most recent one and
if titling is on. If both conditions are satisfied, IQL prints
two lines of column titles and a line of spaces before printing
the actual line. Otherwise, it prints only the actual line.

If you use TITLES OFF, titling is turned off and no column titles
are printed before the print line.

If you use TITLES ON, titling is turned on. Column titles are
printed as described above.

If you use the second format, IQL turns titling on and sets the
column title for the named item to the literal value. If you use
a double slash (//) in the literal, the top title is to the left
of the // and the bottom title is to the right. If you do not
use a double slash, the literal is used as the top title and the
bottom title is blank. To force a single title to the bottom,
start with double slashes. For instance: //LAST NAME.

EXAMPLE:

1. TITLES X = 'SALES//SQ FT'.
    OPEN STORES.
    COMPUTE X = SALES AREA.
    PRINT STORE-NAME, SALES, AREA, X.

TOTAL

FUNCTION:

The TOTAL statement calculates the total of a numeric item in one
of two ways:

1. Over an entire report.

2. Within a specific value of one or more controlling items
    (breaks).

FORMAT:

TOTAL item [BY item [... item]] [(= variable)]

DEFAULTS:

If you do not specify a break item (that is, BY item), the total
is calculated over all records that pass through the TOTAL
statement.

Unless you specify otherwise, each TOTAL statement calculates the
specified total and prints it at the proper time.

DISCUSSION:

The format and use of the TOTAL statement are exactly the same as
the other summary statements AVERAGE, MAXIMUM, MINIMUM, and
TALLY.

If you do not specify a break item (that is, BY item), the total
is calculated over all records that pass through the TOTAL
statement.

If you specify a break item, the total is calculated and printed
when IQL sees a change in the indicated item. If you specify
more than one break item, TOTAL breaks if it sees a change in any
of the indicated items.

You can specify more than one break item as long as each is minor
to one preceding it. A total is calculated for each such break
item. When a change in any break item occurs, a change is
assumed to have occurred in each break item to the right of it in
the TOTAL statement. The purpose of this is to prevent
inadvertent overlap of minor break items over major ones. For
instance, if you are computing a grade point total for each sex
in each school and you encountered an all-girls school, TOTAL GPA
BY SEX gives a misleading answer whereas TOTAL GPA BY SCHOOL, SEX
gives the correct answer.

Since totaling is an arithmetic operation, the item you total
must be numeric. However the item(s) controlling the break can
be any type.

Unless the SUMPRINT OFF statement is in effect, IQL prints the
total line on the report as a total title on the left and the
total value on the right. IQL constructs the title for an item
from the top and bottom dictionary titles and from the word

TOTAL:. IQL constructs the title for a break item from the top and bottom dictionary titles of the break item and constructs the overall title for all the break items with the word OVERALL.

You can give a name to the total by using the (= variable) argument at the end of the statement. You must begin the variable name with either X or ZZ.

After defining the variable, you can use it like any other item. For instance, you can include the variable in computations. One such usage is to calculate percentages at the end of a query. Another usage is to print the total in a print line with other data; often for printing summaries directly under details.

EXAMPLE:

1. HEADING 'ORGANIZATIONAL BUDGETS'.
    OPEN BUDGET-FILE.
    TOTAL BUDGET BY DIV, DEPT.
    TOTAL BUDGET.

    might give:

    HOME DEPT 421 ANNUAL BUDGET TOTAL: 23,921.46
    HOME DEPT 427 ANNUAL BUDGET TOTAL: 37,446.00
    HOME DEPT 433 ANNUAL BUDGET TOTAL: 12,777.50
    HOME DIV 110 ANNUAL BUDGET TOTAL: 74,144.96
    ....
    OVERALL ANNUAL BUDGET TOTAL: 549,873.64

VSPACE

FUNCTION:

The VSPACE statement changes the vertical spacing of reports.

FORMAT:

VSPACE integer

DEFAULT:

If you do not use VSPACE in a query, IQL single spaces reports;
that is, it acts as if you had used VSPACE 1.

DISCUSSION:

IQL physically vertical spaces just before it prints a line or
the titles that correspond with the line.

You can use VSPACE at any point and repeatedly.

VSPACE only affects reports produced by PRINT;   it does not
affect output from DISPLAY.

If you are producing multiple reports, any VSPACE setting remains
in effect from one report to the next. If you wish to use a
different vertical spacing in reports, use VSPACE to change it.
You must use VSPACE again in the second report to reset vertical
spacing. See the example below.

The integer in VSPACE can be any reasonable value.

EXAMPLE:

1. OPEN STUDENTS.
    REPORT 1.
    VSPACE 1.
    PAGING ON.
    HEADING 'HIGH GPA STUDENTS'.
    IF GPA GEQ 3.5 PRINT NAME, CLASS, GPA, YEAR.
    REPORT 2.
    STUDENT ADDRESS LABELS.
    PAGING OFF.
    VSPACE 3.
    PRINT NAME.
    VSPACE 1.
    PRINT STREET.
    PRINT CITY.
    PRINT STATE, ZIP.

    IQL single spaces the first report. In the second report,
    IQL triple spaces between the labels for each student and
    single spaces within the label.

CHAPTER 4

CREATING AND EDITING A DATA FILE

Through IQL, you can create, read, and update a data file. The create, read, and update data file functions work with sequential and indexed sequential (ISAM) file structures. For IQL to work with these file structures, there must be a dictionary that describes the data file to IQL. Chapter 5 explains the dictionary and the dictionary commands that you use to create, read, and change the dictionary. You are not required to read Chapter 5 in order to understand Chapter 4. You need only understand that IQL requires the dictionary before you can create, read, or update a data file.

Since IQL performs the functions described in Chapters 4 and 5 as soon as you enter a command, this guide collectively refers to these functions as the immediate mode.

4.1 BEGINNING THE IMMEDIATE MODE

You enter the immediate mode to create, read, and update a data file with the INPUT, BROWSE, and UPDATE assistance commands.

If you enter through the UPDATE assistance command, IQL displays the prompt <QU> and waits for you to respond with a command. The file you are updating must reside in the directory in which you are logged.

If you enter through the BROWSE assistance command, IQL displays the prompt <QU> and waits for you to respond with a command. At the browse level, IQL permits you to read the data file. However, IQL does not permit you to input data, change data, or delete data from the data file. The file you are reading must reside in the directory in which you are logged.

If you enter through the INPUT assistance command, IQL displays (NEXT RECORD), displays a prompt for the first item in the record, and waits for you to respond with a value. Upon receiving the value, IQL issues the prompt for the next item. IQL continues down the dictionary this way, issuing a prompt for each input field in the record, until IQL encounters the end of the dictionary or you tell IQL to perform another function. See the example session in Appendix B. If you enter the INPUT assistance command and the file already exists, IQL informs you and proceeds to the update level.

4.2 RECOVERING SEQUENTIAL DATA FILES

IQL updates sequential data files by writing to one file while it is reading from another file. One of these files is the original file and the other is a work file. Whenever you enter the immediate commands TOP or UP or use a findlist containing FIRST IQL switches the status of the files. For instance, the work file becomes the original file. The latest original file is the file that is being read at that particular time.

If you enter the SAVE command, IQL forces all updates into the original file, which then becomes the latest original file. If you experience a computer crash (and you cannot continue after the computer returns), you lose all updates (from CHANGE, APPEND, INSERT, or DELETE) from the time the files are last switched For this reason, it is good practice to SAVE the file at regular intervals.

If a computer crash does occur, the latest valid file can be either the original file or the work file. To recover in the latter case, you must copy the work file to the original file. IQL provides a one-record diary file updated with the name of the file. The name of the diary file is displayed when you enter the immediate mode.

To demonstrate a typical recovery, assume you are updating a file called MYDATA.SEQ described by a dictionary named MYDATA. The dialogue appears as follows:

```
@R IQL (RET)
 <QA>UPDATE (RET)
*DICTIONARY NAME: MYDATA (RET)
 (YOUR DIARY FILE IS QL012U.LPT)
 <QU>CHANGE BAL TO 231 IN FIRST ACCT 50 (RET)   (you make a change)
 <QU>CHANGE DEPT 42 TO 51 IN ALL RECORDS (RET)
DECSYSTEM-20 IS NOT RUNNING                     (machine crashes)
@LOG...                                         (you log back in)
@TYPE QL012U.LPT (RET)                           (and look at diary)
 (YOUR LATEST GOOD FILE IS QT012U.TMP)
@COPY   (FROM) QT012U.TMP   (TO) DATA.SEQ(RET)(This step is
                                              needed only if
                                              the latest good
                                              file is not
                                              your original.)
```

4.3 IMMEDIATE MODE COMMAND FORMATS

The immediate mode commands use the following command conventions:

1. First character location

    The command begins in the first character location after the <QU> prompt.

2. Space

    One or more spaces separate a command from a parameter and separate parameters

3. Lowercase and uppercase characters

    A parameter identified by lowercase characters indicates that you are to supply a variable as indicated by the variable name.

A parameter identified by uppercase characters indicates that you are to supply the exact characters as they are shown in the text.

4. Square brackets [ ]

A parameter enclosed in square brackets indicates an optional parameter Do not use the square brackets in the immediate commands you write; the square brackets indicate only an optional parameter. When you do not supply the parameter, the system applies a default value as explained in the parameter description.

A parameter not enclosed in square brackets indicates that the system requires the parameter.

5. Special characters

A parameter containing special characters, such as the equal sign, quotation marks, parentheses, and arithmetic symbols, requires the special characters as shown in the command format.

6. Alphanumeric values

Enter an alphanumeric value as a string of characters including quoted spaces. If you do not furnish all the characters needed to fill the field, the system left justifies the value and inserts spaces to the right of the value. For instance if you specify SMITH to a 10-character field, the system stores the value as SMITH∆∆∆∆∆, where each delta symbol indicates a space.

If the string contains spaces, enclose the string in either single quotes (') or double quotes ("). The system understands either quote mark but be sure you end the quoted space with the same quote mark with which you started. If you use quotes in the value, set off the quoted value with the other quote mark. For instance:

        'SMITHSONIAN INSTITUTION'
        'BOB "CHIP" JONES'
        "ROBERT C. SMITH"

To enter all spaces as the value for an alphanumeric field, enter the value as ' '.

7. Numeric values

Enter a numeric value as a string of special symbols and digits. You can use the following special symbols:

        comma   ,
        dollar symbol $
        decimal point   .
        signs   + and -

Valid formats include:

        $1,234.56   +1,234.56   1,234.56   1234.56
        $1,234.56+   1,234.56+   -100   100-
          +100   -100.0   -100.00   100.00-

The system ignores dollar signs and commas in the numeric value and accepts signs in the leading or trailing position of the value. If you do not enter a sign, the system assumes a positive value.

The system uses the decimal point that you supply to vertically align the value, but the system does not actually store the decimal point. The system knows from the scale in the dictionary where the decimal point is located in the numeric field.

For integer constants, you should omit the decimal point or embed it (such as, 475 or 475.0) to avoid any possible confusion between a decimal point and a period ending a statement.

If you do not furnish enough digits to fill the field, the system provides leading and trailing zeros (after the decimal point) to fill the field.

To furnish an all-zero value for a numeric item, simply enter the single digit 0.

8. Continuation

    You can continue a command on the next line by entering the characters ++ as the last word on the line you wish to continue. For instance:

        <QU> DELETE ALL RECORDS WHERE ++ (RET)
        <QU> DEPT = 442 AND CODE = 'A' (RET)

    is the same as:

        <QU> DELETE ALL RECORDS WHERE DEPT = 422 AND CODE = 'A' (RET)

9. Reserved keywords

    There are a number of reserved keywords with special meaning to the system. These keywords are listed in Chapter 1. If you must use a reserved keyword as a value, enclose the keyword in quotes.

10. Maximums

    a. Field length

        The maximum alphanumeric field length is 54 characters. The maximum numeric field length is 18 digits.

    b. Record length

        The maximum length of a record is 768 characters. Since the maximum length of a record can be changed at a site, check with the site software specialist.

    c. Number of items in a dictionary

        There is no maximum.

d.   Number of records in a file.

The maximum of records in a file is limited by the disk
quota in a directory.

11.   Examples

In the examples, data you enter is shown in red print whereas
the data the system displays is shown in black print.

APPEND

FUNCTION:

The APPEND command positions the file at the end (bottom) and then goes into the input level to receive new records. This command is exactly the same as (if you entered) BOTTOM followed by INSERT.

FORMAT:

APPEND

DISCUSSION:

The first record inserted by APPEND follows the last record of the current file. For a detailed description of entering records, refer to the INSERT command description in this chapter.

EXAMPLE:

1. <QU> APPEND (RET)
    (AT END OF FILE)

    (NEXT RECORD)
    *CUSTNO (CUSTOMER NUMBER) 5 N: 1 (RET)
    *CUSTOMER (CUSTOMER NAME) 16 A: JONES (RET)
    *BUYER (BUYER NAME) 15 A: BARNABAS (RET)
    *CYSALES ....

BOTTOM

FUNCTION:

Positions the file at the end (bottom).

FORMAT:

BOTTOM

DISCUSSION:

If a file is positioned at the end, it is positioned after the last record. There is no current record. An attempt to list or otherwise act on the current record results in an error message and the file is not moved.

EXAMPLE:

1. <QU>BOTTOM (RET)
   (AT END OF FILE)
   <QU>

CHANGE

FUNCTION:

The CHANGE command replaces the contents of an item with a new value. IQL can change up to 15 items in either the current record or selected records of the data file. IQL displays the changed record(s) if the verify option is on.

FORMAT:

```
CHANGE item [TO]   value [... item [TO]   value] [IN      findlist  ]
           [EQ]                    [EQ]         [IF                ]
           [=]                     [=]          [FOR               ]
           [EQUAL]                 [EQUAL]      [WHERE            ].
           [EQUALS]                [EQUALS]     [WHEN              ]
```

DEFAULTS:

If you do not furnish a findlist, the changes are made in the current record only and the file is not repositioned.

DISCUSSION:

Item can be a data item, alphavariable, or numeric variable. A data-item name must begin with a letter, other than the letter X, and can be up to 30 alphanumeric characters and dashes. IQL lists data-item names under the ITEM NAME column title of a dictionary display.

An alphavariable must begin with the letter A and can be up to 30 alphanumeric characters and dashes.

A numeric variable must begin with the letter X or the letters ZZ and can be up to 30 alphanumeric characters and dashes. The initial value of a numeric variable is zero.

Value can be an alphanumeric or numeric string.

If you specify a findlist, the system makes the specified changes in the records located by the findlist. The file remains positioned at the last record changed. If the findlist starts with ALL or if a record is not found, the file remains positioned at the end of the file. Refer to the FINDLIST section in this chapter for an explanation of the FINDLIST.

CREATING AND EDITING A DATA FILE

EXAMPLES:

1. CHANGE DEPT TO 421

2. CH DEPT 421

3. CH NAME 'SEISS' IN SSNO 572409987

4. CHANGE VENDOR TO 'BRAND X' SALESMAN TO ++
   JONES REGION = 6 FOR ALL VEND-NO ++
   = 50 SALESMAN= 'SMITH'

5. CHANGE NAME 'SEISS' FOR KEY = '572409987'

COLUMNS

FUNCTION:

Designates the left and right columns of the portion of the record to be displayed by LIST (or CHANGE, DOWN, FIND, UP if verify option is on).

FORMAT:

COLUMNS   integer   [integer]

DEFAULTS:

If you do not use COLUMNS in the session, IQL displays the full record, which causes IQL to display the record on as many consecutive lines as necessary.

If you do not specify a second integer, IQL displays a 72-character portion of the record, beginning at the column designated by the first integer.

DISCUSSION:

The leftmost character in a record is in column 1. The first integer designates the first (leftmost) column to be displayed and the second integer is the last (rightmost) column to be displayed.

EXAMPLES:

1.   COLUMNS   40   75

2.   CO   40   75

3.   CO   601

DELETE

FUNCTION:

Deletes one or more selected records.

FORMAT:

DELETE   [findlist]
           [integer]

DEFAULTS:

If you specify neither a findlist nor a number of records, IQL deletes the current record.

DISCUSSION:

If you specify an integer, IQL deletes that many sequential records, starting with the current one.

If you specify a findlist, IQL deletes the records located by the findlist. See the FINDLIST section.

Regardless of the form of the DELETE command that you use, the file remains positioned at the record after the last record deleted. However, if you use a findlist that starts with ALL or if the system cannot find a specified record, the file is left positioned at its end.

EXAMPLES:

1.   DELETE

2.   DE

3.   DE 10

4.   DELETE ALL RECORDS FOR REGION 56

5.   DELETE FIRST RECORD ACCT 7252

6.   DELETE NEXT RECORD WHERE VENDOR NOT = 'BLT'

7.   DE VENDOR NOT BLT

DOWN

FUNCTION:

Moves the file down (toward the end or bottom) a specified number of records. If verify option is on, IQL displays the new current record.

FORMAT:

DOWN   [integer]

DEFAULT:

The integer is the number of records the file is to be moved down. If you omit the integer, IQL moves the file down one record.

DISCUSSION:

If IQL encounters the end of the file before it has moved down the requested number of records, it issues an AT END OF FILE message and leaves the file positioned at its end. If the file is positioned at its beginning and you use DOWN or DOWN 1, the new current record is the first record in the file.

EXAMPLES:

1.   DOWN

2.   DO

3.   DOWN   12

4.   DO   600

EXIT

FUNCTION:

The EXIT command returns you to the assistance level.

FORMAT:

EXIT

DISCUSSION:

Ends the update or browse session and saves the final version of the file.

EXIT and END are synonyms.

EXAMPLES:

1. EXIT

2. END

EXTRACT

FUNCTION:

The EXTRACT command reads selected records from the current ISAM
or sequential data file and writes the records to a new
sequential file.

FORMAT:

EXTRACT   [TO 'file.ext']   [findlist]

DEFAULT:

If you do not furnish a filename, IQL writes the records to the
QTjobX.TMP file, where the smbolic term, job, is the job number
the Operating System assigns when you log into the system.

DISCUSSION:

See the FINDLIST discussion in this chapter. If you omit the
findlist, IQL copies the first record to the new file.

The sequential file, which IQL writes, retains the same data
format (that is, 6-bit ASCII or 7-bit ASCII) as the original
file.

EXTRACT always goes to the beginning of the input file before
doing the extract.

After EXTRACT, IQL switches to processing the extracted file.

EXAMPLE:

1. EXTRACT

    Copy the first record from the input data file to the new
    data file and use the new data file as the input data file.

FIND

FUNCTION:

Positions the file at a selected record   If verify option is on,
displays the new current record.

FORMAT:

FIND   [findlist]

DEFAULT:

If you do not furnish a findlist, FIND uses the last findlist
that you furnished, which permits you to continue a search by
simply entering FIND or FI.

DISCUSSION:

See the FINDLIST discussion in this chapter.   If IQL does not
find a record that qualifies, it issues an AT END OF FILE message
and leaves the file positioned at the end.

EXAMPLES:

1. FIND NEXT RECORD WHERE NAME EQUALS 'SMITH'

2. FIND NEXT NAME EQ 'SMITH'

3. FIND NAME 'SMITH'

4. FI NAME 'SMITH'

5. FI   (if used after the above FIND, continues the search)

6. FIND KEY = '234581099'

7. FIND DEPT GR 421 AND DIV EQ 3

8. FIND ACCOUNT GEQ 6000 AND YEAR NOT LESS 75

9. FIND ACCOUNT GE 6000 AND YEAR GE 75

FINDLIST

FUNCTION:

A findlist is a set of item names, values, and relationships that direct IQL to locate records in a file. Findlists are used in the commands CHANGE, DELETE, FIND, and LIST.   An example of a findlist is:

NEXT RECORD WHERE NAME EQUALS 'SMITH'

FORMATS:

1. [ALL]   [RECORDS] [IF] [item [NOT] [EQ] value] [[IF...value] [AND]
    [NEXT]   [RECORD]   [WHEN]   [NEQ]
    [FIRST]   [WHERE]   [GEQ]
                        [FOR]   [LEQ]
                                           [GR]
                                           [LS]
2. KEY = value

DEFAULT:

If you do not specify a relationship, IQL assumes EQUALS.

If you do not use any of the keywords NEXT, FIRST, or ALL, the search proceeds as if you had used NEXT.

DISCUSSION:

Remember that a findlist is only part of another command.   See the discussions of CHANGE, DELETE, FIND and LIST for uses of findlists.

The referenced item(s) can be any of the defined items in the dictionary controlling this run. You can specify up to 15 items and relationships in one statement.

You can write relationships in any of the following ways:

Relation	IQL Accepted Words/Symbols			
equal	EQ	EQUALS	=	IS
not equal	NQ	NE	<>	NEQ
less than	LS	LT	<	LESS
greater than	GR	GT	>	GREATER
less than or equal to	LEQ	LE	<=	LQ
greater than or equal to	GEQ	GE	>=	GQ

Note that you can use NOT to reverse a relationship.   For instance, NOT EQ is the same as NE and NOT LEQ is the same as GR.

You can combine simple relationships to make a complex relationship by joining them with AND clauses. You can use parentheses to clarify complex relationships.

The words RECORDS, RECORD, AND, WHEN, WHERE, FOR, and IF are optional.   They have no effect on the search and are processed simply so you can use phrases that read like English.

If you use NEXT, the search begins at the record after the current record and the file is positioned at the next record that satisfies the findlist.

If you use FIRST, the search begins at the first record in the file. This is the same as if you had used the TOP command and then used NEXT in your findlist.

If you use ALL, the search begins at the record after the current one and the file is positioned in turn at each down-stream record that satisfies the findlist. The file is left positioned at the end of the file.

The rules for formatting item values in a findlist are the same as for furnishing item values in the INSERT command. A discussion on formatting item values is presented at the beginning of Section 4.3 in this guide.

ERRORS:

IQL issues an error message and rejects the command if it detects any of the following in a findlist:

1. Item not contained in a dictionary.

2. Alphanumeric value too long.

3. Numeric value contains illegal alphabetic characters.

4. Too many decimal or integer places in a numeric value.

INSERT

FUNCTION:

The INSERT command enters IQL into the input level, where you can insert one or more new records immediately after the current record. IQL prompts you for each new item value in each record and allows you to furnish numeric values in a variety of formats. IQL takes care of decimal point alignment, leading zeros, and trailing zeros or blanks.

FORMAT:

INSERT

DISCUSSION:

If your file is currently positioned at the top (beginning), INSERT adds the new records before the first record.

If your file is currently positioned at the bottom (end), INSERT adds the new records after the last record.

INSERT operates under control of a dictionary. For each new record, IQL starts at the beginning of the dictionary and issues a prompt for the first input data item. When you respond with a value for the item, IQL issues the prompt for the next input item. This process continues until either the end of the dictionary is reached or you enter an immediate command NEXT, KILL, or UP.

The prompt issued for each item is contructed from the dictionary entry for the top and bottom title, the item length, and the scale (if any). For alphanumeric items, the prompt ends with nnA, where nn is the length of the item. For numeric items, the prompt ends with nn.mmN, where nn is the maximum number of integer positions and mm is the number of decimal positions (if any). Some examples of prompts appear as:

    *LAST NAME 16 A :
    *EARNINGS 7.2 N :

The rules for entering data item values are the same for INSERT as for other commands. These rules are explained at the beginning of Section 4.3 in this guide.

The input level commands you can use to control INSERT are described as follows:

1. Carriage return, space(s), or quoted space(s)

    If you enter a carriage return, all spaces, or ' ', alphanumeric items are set to all spaces and numeric items are set to zero.

2. KILL

    If you make a serious error, enter the KILL command. IQL rejects the record and permits you to begin entering the record again from the beginning.

3. UP [integer]

   If you make an error in entering a value and catch it later in the same record, you can back up to correct the entry. When you enter the UP command, IQL moves back that many entries, prompts you for that item, and continues down the dictionary again. If you omit the integer, IQL moves back one prompt.

4. DOWN [integer]

   Move down integer prompts. If you do not specify an integer, IQL moves down one prompt.

5. NEXT

   To go on to the next record, enter the NEXT command. IQL goes through the rest of the current record, blanking out alphanumeric input items and setting numeric input items to zero. Then it writes out the record and prompts you for the first value in the next record.

6. END or EXIT

   To terminate INSERT, enter the END or EXIT command in response to the first prompt.

ERRORS:

   If IQL detects an improper value, it rejects the value with an error message, reissues the prompt, and waits for you to enter the value correctly. Improper values are: (a) alphanumeric value too long, (b) too many integer or decimal places in a numeric value, or (c) numeric item contains illegal characters.

CREATING AND EDITING A DATA FILE

EXAMPLE:

1. <QU>INSERT (RET)

    (NEXT RECORD)
    *CUSTNO (CUSTOMER NUMBER) 5 N: 23 (RET)
    *CUSTOMER (CUSTOMER NAME) 16 A: 'SAVINGS TRUST' (RET)
    *BUYER (BUYER NAME) 15 A: PURCHASING (RET)
    *CYSALES (CURR YR SALES) 7.2 N: L000 (RET)
    % VALUE TOO LONG OR CONTAINS ALPHA
    *CYSALES (CURR YR SALES) 7.2 N: 1000 (RET)
    *LYSALES (LAST YR SALES) 7.2 N: 575.98 (RET)
    *LPDATE (LAST PO DATE) 6 N: 780112 (RET)

    (NEXT RECORD)
    *CUSTNO (CUSTOMER NUMBER) 5 N: 24 (RET)
    *CUSTOMER (CUSTOMER NAME) 16 A: "WINNEBAGO SALES" (RET)
    *BUYER (BUYER NAME) 15 A: 'JERRY COWENS' (RET)
    *CYSALES (CURR YR SALES) 7.2 N: NEXT (RET)

    (NEXT RECORD)
    *CUSTNO (CUSTOMER NUMBER) 5 N: 25 (RET)
    *CUSTOMER (CUSTOMER NAME) 16 A: 'ROYAL CYLLE' (RET)
    *BUYER (BUYER NAME) 15 A: 'RON KLINE' (RET)
    *CYSALES (CURR YR SALES) 7.2 N: UP 2 (RET)
    *CUSTOMER (CUSTOMER NAME) 16 A: 'ROYAL CYCLE' (RET)
    *BUYER (BUYER NAME) 15 A: 'RON KLINE' (RET)
    *CYSALES (CURR YR SALES) 7.2 N: 451 (RET)
    *LYSALES (LAST YR SALES) 7.2 N: $217.45 (RET)
    *LPDATE (LAST PO DATE) 6 N: 780210 (RET)

    (NEXT RECORD)
    *CUSTNO (CUSTOMER NUMBER) 5 N: 16 (RET)
    *CUSTOMER (CUSTOMER NAME) 16 A: RIETRONICX (RET)
    *BUYER (BUYER NAME) 15 A: KILL (RET)

    (NEXT RECORD)
    *CUSTNO (CUSTOMER NUMBER) 5 N: 26 (RET)
    *CUSTOMER (CUSTOMER NAME) 16 A: RIATRONICS (RET)
    *BUYER (BUYER NAME) 15 A: "LAUREN L LEE" (RET)
    *CYSALES (CURR YR SALES) 7 2 N: -250. (RET)
    *LYSALES (LAST YR SALES) 7.2 N: 22,250 (RET)
    *LPDATE (LAST PO DATE) 6 N: 780130 (RET)

    (NEXT RECORD)
    *CUSTNO (CUSTOMER NUMBER) 5 N: END (RET)

    <QU>

LIST

FUNCTION:

The LIST command displays one or more records on the terminal. IQL can display either a selected portion of each record or only selected items, depending on the format of the LIST command. You can list up to 15 items for each record.

FORMATS:

1. LIST [integer] [item ... item]

2. LIST [item [... item]] [findlist]

DEFAULTS:

If you use only the word LIST, IQL lists the current record and the file is not moved.

DISCUSSION:

If you do not specify any item names to be listed, LIST displays the entire record or a part of it, depending on whether you have used COLUMNS or not.

The following paragraphs summarize the LIST command options.

1. LIST

   Lists the entire current record.

2. LIST integer

   Lists the current and following records until the integer number of records are listed.

3. LIST integer item-name item-name ...

   Lists only the named items, starting with the current record, until these items have been listed for the specified records. Signs and decimal points are shown for numeric items. The items format is generally easier to read than a continuous portion of the record, especially if binary items are involved.

4. LIST item-name item-name ...

   Lists only named items for the current record. Does not move the file.

5. LIST findlist

   Lists all records located by the findlist.

6. LIST item-name item-name ... findlist

   Lists only named items for each record located by the findlist.

IQL leaves the file positioned at the last record listed unless
(1) you used ALL in your findlist or (2) in listing the system
encountered the end of the file. In these cases, it issues an AT
END OF FILE message and leaves the file positioned at the end of
the file.

EXAMPLES:

1. LIST

2. LIST   10

3. LIST   20   ACCOUNT   LEDGER   AMOUNT

4. LIST   ACCOUNT   LEDGER   AMOUNT

5. LIST FIRST ACCOUNT = 6000

6. LIST ALL ACCOUNT = 6000

7. LIST LEDGER AMOUNT FOR ALL ACCOUNT = 6000

8. LIST ACCOUNT LEDGER AMOUNT RATE FOR ALL ++
      ACCOUNT GEQ 6000 AND YEAR 76

SAVE

FUNCTION:

Saves the current version of the file.

FORMAT:

SAVE

DISCUSSION:

The current version replaces the original version of the file.
The file is left positioned at its beginning.

EXAMPLES:

1. SAVE

2. SA

TOP

FUNCTION:

Positions the file at the beginning (top).

FORMAT:

TOP

DEFAULT:

If the file is positioned at the beginning of the file, the position remains unchanged.

DISCUSSION:

If the file is positioned at the beginning, it is positioned before the first record.

EXAMPLES:

1. TOP

2. TO

UP

FUNCTION:

Moves the file up (toward its top or beginning) a specified number of records. If verify option is on, IQL displays the new current record.

FORMAT:

UP [integer]

DEFAULT:

If you omit the integer value, the system moves the file up one record.

DISCUSSION:

The integer is the number of records the file is to be moved up. If IQL encounters the beginning of the file before it has moved up the specified number of records, it issues an AT BEGINNING OF FILE message and leaves the file positioned at its beginning (before the first record). If the file is positioned at its end (after the last record) and UP or UP 1 is used, IQL moves the file up to the last record in the file.

ERRORS:

If you enter alphabetic characters in the integer parameter, IQL issues an error message, rejects the UP command, and does not move the file.

EXAMPLES:

1. UP

2. UP 600

VERIFY

FUNCTION:

The VERIFY command turns on or off the verify option.

FORMAT:

VERIFY   [ON]
             [OFF]

DEFAULT:

When you begin your update session, the verify option is automatically on.

DISCUSSION:

If the verify option is on, IQL (1) automatically displays the new current record located by UP, DOWN, or FIND commands and (2) automatically displays each changed record as a CHANGE command is carried out. If the verify option is off, these records are not automatically displayed.

ERRORS:

If IQL encounters a word other than ON or OFF after VERIFY, IQL issues an error message and turns on the verify option.

EXAMPLES:

    1.   VERIFY

    2.   VE   ON

    3.   VERIFY ON

    4.   VERIFY OFF

CHAPTER 5

CREATING AND EDITING A DICTIONARY

IQL controls the access to data files and data bases by using a prestored description of the data files or data bases. The description is called a dictionary, which IQL stores as a table on disk. A dictionary contains information about where and how each data item is stored, cosmetic information anticipating printing, and password information to protect the data file or data base from unauthorized access using IQL.

For IQL to access a data file or a data base, you need to define a dictionary only once; moreover you need define only those specific items in which you are interested.

To define or change a dictionary, use the DEFINE assistance command. IQL issues a full prompt for each field in each dictionary entry, and you can back up if you make a mistake. Appendix C shows a full example of a dictionary. Also, in Section 5.5, full examples are given of DEFINE sequences to create or change a dictionary entry.

A dictionary begins with an FD - file definition - entry, which identifies the dictionary and describes the data file or data-base. The main entry in a dictionary that describes a file is the DD - data definition - entry, which describes each data item in the file. Other dictionary entries for files describe comments and passwords.

In addition to the item, comment, and password entries, a dictionary that describes a DBMS data base must contain record, area, and set entries.

5.1 DEFINING A DICTIONARY FOR DBMS

You can use the DBMS schema (.SCH) file and a program called IQSCH to initially build a dictionary for a DBMS data base. IQSCH creates specifications for column titles and editing pictures since the schema does not contain these specifications.

It is strongly recommended that you review the dictionary created by IQSCH and change the column titles or editing pictures you do not like. Please note that dictionaries created with IQSCH do not contain items below the 02 level in the schema.

Use the following commands to initially create a dictionary for a DBMS data base.

        @R IQSCH (RET)
        (respond to prompt with name of the .SCH file)
        @R IQD (RET)
        (now inspect the dictionary generated)

```
@R IQL.(RET)
<QA>ITEMS  SUBSCHEMA-NAME (RET)
```

(IQL then displays the new dictionary. To change it, use the
DEFINE assistance command.)

5.2 DISPLAYING A DICTIONARY

You can request either a terminal display or a full printer listing of
any dictionary in a directory. The DICTIONARIES assistance command of
IQL produces a terminal display of the file definitions in all
dictionaries.   The ITEMS assistance command produces a full terminal
display of a specific dictionary.

Since terminal displays of dictionary information are formatted 72
characters wide, you may find it easier to work with the 132-character
wide printer listing of dictionary. To produce a printer dictionary
listing, use an action code of P in an FD transaction (see Section 5.5
for more information on the FD transaction).

The types of individual entries in a dictionary are listed below. All
entry types are not necessarily needed in any one dictionary. Each
entry type is described in considerable detail in its own section
later in this chapter.

   1. FD (file definition) entry

       Defines the overall structure and location of a file or data
       base.   This is the master definition in each dictionary and
       identifies the dictionary itself. The FD entry must be the
       first entry in the dictionary.

   2. DD (data definition) entry

       Defines a specific data item in a file or dataset.   DD
       entries make up the bulk of any dictionary. Each DD entry
       defines not only where and how the data item is stored, but
       also contains column titles and a printing picture to format
       the display of the item.

   3. PD (password definition) entry

       Defines a password, which protects the dictionary against
       unauthorized use.   Passwords can be applied at either the
       file or individual item level.   Passwords are stored
       scrambled.   Refer to Section 5.3 for more information on
       protecting a data base.

   4. CD (comment) entry

       The CD entry places explanatory comments in the dictionary
       itself. It does not serve any other purpose.

   5. AD (area definition) entry

       The AD entry is used only in dictionaries that define DBMS
       data bases.   IQL uses AD entries to check area names in DBMS
       FIND commands.

6. RD (record definition) entry

    The RD entry is used either in dictionaries describing files
    with multiple record types or defining DBMS data bases. All
    DD (data definition) entries that follow a particular RD
    entry belong to the record the RD describes.

7. SD (set definition) entry

    The SD entry is used only in dictionaries that define DBMS
    data bases.   IQL uses SD entries to check set relationships
    that you specify to direct IQL to specific records in a data
    base.

5.3 DEFINING DICTIONARY PROTECTION

You define passwords with the FD and PD transactions.   Passwords
protect the data file or data base against anyone accessing sensitive
data through IQL. If you protect a dictionary with a master password
when defining a dictionary with a FD transaction, IQL requests
password reference numbers at the end of the FD transaction.   The
password reference numbers point to specific PD entries in the
dictionary that define the individual passwords.

You can protect the data file or data base at any or all of the
following levels:

1. READ

    A reference to a password here means that IQL will not permit
    reading the file or data base unless it receives a password
    at the same or higher level than that referenced.

2. WRITE

    A reference to a password here means that IQL will not write
    information from the file (with COPY or CREATE) unless it
    receives a password at the same or higher level than that
    referenced.

3. REWRITE

    Rewrite access only applies to ISAM files. A reference to a
    password here means that IQL will not write back into that
    ISAM file unless it receives a password at the same or higher
    level than that referenced.

Normal practice is to make WRITE protection higher than READ
protection and REWRITE protection higher than WRITE protection. Since
mistakenly writing back into an ISAM file can be catastrophic, it is a
good idea to password protect (at least at the REWRITE level) each
dictionary describing an ISAM file.

You can also protect any individual item in a dictionary by putting a
password reference number in the DD entry describing that item. This
overrides READ protection at the file level. You can also indicate
exclusivity in the DD so that only the referenced password permits
access to the protected item. Individual item password protection is
convenient if you wish to protect different items at different levels.

Since password numbers are a level of protection numbers, any password with a higher number unlocks access to anything protected in that dictionary with a lower level password. The only exception to this rule is an exclusive item protection.

5.4   ORDERING DICTIONARY ENTRIES

The only order required of entries in a dictionary is that the FD entry must come first and DD entries must directly follow any RD entry describing the record in which the data items are located. Within a record, no specific order is required for DD entries. You can arrange DD entries by occurrence in the record, by function, or alphabetically by name.

The following general order makes dictionaries easy to use and efficient for IQL to process:

        FD entry (required)
        PD entries (if any)
        AD entries (if any) - only used for DBMS
        SD entries (if any) - only used for DBMS
        RD entry (if any)
        DD entries for above RD or whole dictionary if no RD
        RD entry (if any)
        DD entries for above RD
        ... CD entries can be anywhere

5.5 DICTIONARY COMMAND FORMATS

Each type of dictionary command is described in detail in the remainder of Chapter 5. The AD, CD, DD, FD, PD, RD, and SD commands invoke a set of prompts referred to as a transaction. As you respond to the prompts, IQL generates a temporary file. When you use the END dictionary command, IQL uses the temporary file to change the actual dictionary entries.

Unless otherwise stated, you can use the following dictionary commands at any time to control the progress of a transaction.

   1. NEXT

       Go on to the next transaction. This generally means you completed the meaningful part of this transaction.

   2. KILL

       Kill this transaction and start it over.

   3. END

       End prompted input of transactions and use the transactions to physically define or change the dictionary. When the dictionary update is complete, the system returns to the assistance level.

   4. SCRUB

       End prompted input of transactions. Do not physically define or update the dictionary but go up directly to the assistance level.

5. BACK integer

    Back up integer prompts. You should use BACK if you find you
    made a mistake in the transaction and wish to return to
    correct it.   The integer can be one or two digits.

6. DOWN integer

    Move down integer prompts.

Some transactions display the prompt:

    *ACTION CODE (A,C,D,P):
or
    *ACTION CODE (A,C,D):

Respond with one of the following action codes. IQL uses the action
code to control the transaction.

1. The A action code

    Add the entry defined by this transaction. IQL inserts the
    entry   immediately   following   the entry that you just
    identified to IQL. For instance, to add a DD entry to the
    end of a dictionary, you must first use a FD transaction with
    the C action code to identify to IQL the dictionary.   Then
    you must use a DD transaction with the C action code to
    identify to IQL the last DD entry in the dictionary.
    Finally, you must use the DD transaction with the A action
    code to add a DD entry to the end of the dictionary.

2. The D action code

    Delete the entry defined by this transaction.   For FD, DD,
    RD, AD, and SD entries, only the name is necessary to locate
    the entry to be deleted. For PD entries, the full password
    is necessary. For CD entries, the comment number is needed.

3. The C action code

    Change or identify to IQL the entry defined by this
    transaction.   Only the nonblank fields in the entry replace
    the corresponding fields in the entry. If you wish to force
    blanks into a field in the entry, enter the special word
    BLANK in response to the DEFINE prompt.

4. The P action code

    Create a print-formatted file of the contents of   the
    dictionary named by this transaction (FD entry only). IQL
    creates a print-formatted file with the filename QLjobD.LPT,
    which you can later print using the TOPS-10 or TOPS-20 PRINT
    command.

CREATING AND EDITING A DICTIONARY

AD

FUNCTION:

    The AD transaction creates an AD entry that describes an AREA in
a schema. The AD entry is used only in dictionaries describing a
DBMS data base.

ELEMENTS:

    1.   COMMAND: AD

    2.   AREA NAME:

        The name of the area. Must be the same as the name of the
area in the DBMS schema.

    3.   AREA ORIGIN:

        Always 1. At present, IQL does not use the area origin, but
may in future versions.

    4.   DESCRIPTION:

        Descriptive remarks up to 76 characters.

DISCUSSION:

    IQL uses AD entries to check area names you use in the DBMS
record selection expressions in the FIND query statements.

EXAMPLES:

    1.   Defining a new AD:

        *FD,DD,PD,CD,RD,AD,SD,END,SCRUB: AD (RET)
        *ACTION CODE (A,C,D):   A (RET)
        *AREA NAME: PAYROLL (RET)
        *AREA ORIGIN: 1 (RET)
        *DESCRIPTION (IF ANY):   HOLDS PAYROLL INFO (RET)

        Gives the area description entry:

            AD PAYROLL 1 HOLDS PAYROLL INFO

    2.   Changing the above AD entry:

        *FD,DD,PD,CD,RD,AD,SD,END,SCRUB: AD (RET)
        *ACTION CODE (A,C,D):   C (RET)
        *AREA NAME: PAYROLL (RET)
        *DESCRIPTION (IF ANY):   NEXT (RET)

        Gives the revised AD entry:

            AD PAYROLL 1 HOLDS PAYROLL INFO

FUNCTION:

The CD transaction creates a comment entry anywhere in a dictionary.

ELEMENTS:

1. COMMAND: CD

2. COMMENT NUMBER:

    Where number is one or two digits and can be 0.

3. COMMENT TEXT:

    Any string up to 115 characters.

DISCUSSION:

Comment entries serve as notes in a dictionary. These entries are especially useful if you are making up a dictionary to be used later by someone else.

The comment number is there so that you can differentiate between comment entries if you later wish to change or delete them. Although you can use a comment number value of 00 (which prints as spaces because IQL suppresses zeros), you must be careful in positioning the dictionary at any specific comment if you ever wish to change or delete it.

EXAMPLES:

1. Defining a new CD:

    ```
    *FD,DD,PD,CD,RD,AD,SD,END,SCRUB:    CD (RET)
    *ACTION CODE (A, C, OR D):          A (RET)
    *COMMENT NUMBER (2 DIGITS):         1 (RET)
    *COMMENT TEXT:                      JOB OPENINGS (RET)
    ```

    Gives the comment entry:

    CD 1 JOB OPENINGS

2. Changing a CD:

    ```
    *FD,DD,PD,CD,RD,AD,SD,END,SCRUB:    CD (RET)
    *ACTION CODE (A,C,D):               C (RET)
    *COMMENT NUMBER (2 DIGITS):         1 (RET)
    *COMMENT TEXT:                      IOWA JOB OPENINGS (RET)
    ```

    Gives the comment entry:

    CD 1 IOWA JOB OPENINGS

DD

FUNCTION:

The DD transaction creates a DD entry that describes an individual data item in a data file or data base. The DD entry contains information enabling IQL to locate, extract, and work with the item that the DD entry describes. The DD entry also contains information for IQL to print or to display the item in a meaningful format.

ELEMENTS:

1. COMMAND: DD

2. ITEM NAME:

The item name can be up to 30 characters and can contain any combination of letters numbers and dashes. The item name must contain at least one letter and cannot start or end with a dash.

3. TOP COLUMN TITLE:

The top column title can be up to 10 characters.   These characters are printed at the top of each column in a report listing the item in default column print format.

4. BOTTOM COLUMN TITLE:

The bottom column title can be up to 10 characters.   The title is printed in a report right under the Top Column Title.

5. LOCATION OF FIRST CHARACTER OF ITEM:

The location in your record of the leftmost character of your item.   The first character location in a record is numbered 1. Binary items start on a word boundary.

6. LENGTH OF ITEM:

For an alphabetic or alphanumeric item, the length of the item is expressed in characters. For a numeric item, the length is expressed in digits, exclusive of sign.   For a binary item, it is the length of the item if converted to decimal digits, exclusive of sign (that is, a binary item that contains 9 digits physically occupies one word; use item length of 9).

7. ITEM TYPE:

The item type is A for alphabetic or alphanumeric, N for numeric, and B for binary. Generally it is good practice to make an item alpha (even if it contains only digits) unless you are actually going to compute with it. For instance, there is no need to make a zip code or a social security number a numeric item.

8. SCALE:

The scale is the number of decimal positions in the item. Use 0 if there is no decimal point or if the item is alpha. IQL uses scale both in lining up for computations and for placing the item value in a picture for displays. The decimal point is not actually recorded in the data. For instance, if an item with length 6 and scale 2 is recorded as 039508, IQL prints it (if the picture is ZZZZ.99) as 395.08.

9. DISPLAY PICTURE:

A group of characters that IQL uses to format the item value for displaying or printing so that you can read it conveniently.

10. INPUT FIELD? (Y = YES, N = NO):

If you respond with Y, IQL prompts you for the data item while in the update or input level. If you respond with N, IQL does not prompt you for the data item. If you supply neither response, IQL assumes the Y response.

11. SCAN GROUP NAME:

Respond with a scan group name only if you are defining an item that repeats. These items can be processed as one item by the SCAN capability. The scan group name is a 1-character common identifier given to all member items in the same scan group. When IQL is scanning left to right in a scan group, it jumps ahead from member to next occurrence of the same member. Each jump is indicated by the length of the longest member. For instance, assume DATE is the longest member of a group; and the items following in the dictionary are YEAR, MONTH, and DAY, all being members of the same scan group as DATE. The scan group name for each of DATE, YEAR, MONTH, and DAY would be the same (for example: W). In scanning, IQL would jump from YEAR to YEAR by the length of DATE.

12. REPEATS IN GROUP:

The number of times each member of the group is repeated.

13. SCAN STOP CHARACTER:

The scan stop character is a unique character that IQL displays only if the item is a member of a scan group. When IQL encounters the scan stop character in all positions of a value for the item, IQL stops scanning for the item, which serves to end the scan when all the buckets of the item are not used.

14. PASSWORD REFERENCE:

Used only if this item is to be individually protected by a password. This is a 2-digit number pointing to an earlier Password Definition (PD) entry in this dictionary. IQL does not permit this item be used unless the referenced password or a higher level password, depending on the exclusivity, if furnished.

15. EXCLUSIVITY:

IQL displays the exclusivity prompt if you specified an item
level of password protection in the previous prompt. If you
enter Y (yes), then only the referenced password unlocks the
item.   If you enter N (no), then either the referenced
password or any higher level password in this dictionary
unlocks the item.

EXAMPLES:

1. Defining a DD entry for a numeric item:

```
*FD,DD,PD,CD,RD,AD,SD,END,SCRUB:        DD (RET)
*ACTION CODE (A,C,D):                   A (RET)
*ITEM NAME (UP TO 30 CHARS): ˉ          LY-DONATE (RET)
*TOP COLUMN TITLE (UP TO 10 CHARS):     LAST YEAR (RET)
*BOTTOM TITLE (UP TO 10 CHARS):         DONATION (RET)
*LOC OF FIRST CHAR OF ITEM:             73 (RET)
*LENGTH OF ITEM IN DIGITS OR CHARS:     8 (RET)
*ITEM TYPE (A,N OR B):                  N (RET)
*SCALE (NO. OF DECIMAL PLACES):         2 (RET)
*DISPLAY PICTURE (IF ANY):              $$$$,$$$.99 (RET)
*INPUT FIELD? (Y = YES, N = NO):        N (RET)
*SCAN GROUP NAME (A - Z):               NEXT (RET)
```

Gives the following DD entry:

DD LY-DONATE LAST YEAR DONATION 73 8 N 2 $$$$,$$$.99

2. Defining an alpha item:

```
*FD,DD,PD,CD,RD,AD,SD,END,SCRUB:        DD (RET)
*ACTION CODE (A,C,D):                   A (RET)
*ITEM NAME (UP TO 30 CHARS):            COMPANY-NAME (RET)
*TOP COLUMN TITLE (UP TO 10 CHARS):     COMPANY (RET)
*BOTTOM TITLE (UP TO 10 CHARS):         NAME (RET)
*LOC OF FIRST CHAR OF ITEM:             108 (RET)
*LENGTH OF ITEM IN DIGITS OR CHARS:     26 (RET)
*ITEM TYPE (A,N OR B):                  A (RET)
*SCALE (NO. OF DECIMAL PLACES):         NEXT (RET)
```

Gives the following dd:

DD COMPANY-NAME COMPANY NAME 108 26 A

CREATING AND EDITING A DICTIONARY

FD

FUNCTION:

The FD transaction provides an FD entry that identifies the dictionary, defines the location and overall attributes of the data file or data base, and protects the dictionary at the file level.

ELEMENTS:

1. COMMAND: FD

2. DICTIONARY NAME:

    Up to 30 characters, consisting of any combination of letters, numbers, and dashes. You cannot start or end a dictionary name with a dash. The dictionary name must contain one or more letters. For dictionaries describing DBMS data bases, the dictionary name must be the same as the name of the subschema the dictionary describes.

3. DICTIONARY UNLOCKING PASSWORD:

    The password that you specify defines the master password to IQL. To display or change privileged areas of a dictionary, you must supply to IQL the master password.

4. FILENAME:

    The name of the file the dictionary describes. If the file is part of a DBMS data base, the filename must be the name of the .SCH file. You can specify up to six filename characters and up to three extension characters. The filename follows the regular DIGITAL standard for filenames.

5. FILE DIRECTORY:

    The directory where the file is located. You should respond with the regular DIGITAL standard for directory names. A value of spaces is acceptable and is recommended if you are going to access a file only from the same directory in which it is located. The directory information in a dictionary is only documentary. Refer to Chapter 1 for a description of how IQL locates files.

6. FILE TYPE:

    Respond with one of the following file types:

TAPE	magnetic tape, ASCII format
DISK6	disk, 6-bit ASCII format
DISK7	disk, 7-bit ASCII format
DBMS	DBMS data base

7. RECORD LENGTH:

    Maximum length of the record in characters. Files written with the COPY command have this record length. Not used for DBMS processing. For 6-bit files, the record length must be a multiple of six.

8. BLOCKING FACTOR:

    The blocking factor defines to IQL the number of records per physical block of an ISAM file. Most sequential disk files use a blocking factor of 0, but some have specific blocking factors.

9. LOCATION OF ISAM KEY:

    Applicable only to ISAM (indexed sequential) files. This is the location in the record of the leftmost character of the ISAM key. (The first character in a record is numbered 1). IQL assumes the file is ISAM if this field is specified (that is, not blank or 0).

10. ISAM KEY LENGTH:

    The length of the ISAM key in characters. Used only for ISAM files.

11. ISAM KEY TYPE:

    Respond with A for alphabetic and N for numeric. Used only for ISAM files.

12. ISAM KEY SIGN:

    a. S

        Respond with S for signed (only applies to numeric).

    b. U

        Respond with U for unsigned (only used for ISAM files).

13. READ PASSWORD REFERENCE:

    The level of the password, if any, that protects against unauthorized use of this dictionary to read the specific file or data base. (The actual password is defined in a later PD entry in this dictionary).

14. COPY PASSWORD REFERENCE:

    The level of the password, if any, that protects against the unauthorized use of the dictionary to write (using the COPY, CREATE, or UPDATE assistance commands) to a data file or data base described by this dictionary. (The actual password is defined in a later PD entry in this dictionary.)

15. REWRITE PASSWORD REFERENCE:

    The level of the password, if any, used to protect against unauthorized use of this dictionary to write records back in place in an ISAM file. Not applicable to other than ISAM files.

EXAMPLES:

1. Defining a new FD.

   <QA>DEFINE (RET)

   ```
   *FD,DD,PD,CD,RD,AD,SD,END,SCRUB:     FD (RET)
   *ACTION CODE (A,C,D OR P):           A (RET)
   *DICT NAME - UP TO 30 CHARS:         JOBS (RET)
   *DICT UNLOCKING PASSWORD, IF ANY:    FERN (RET)
   *FILENAME AS XXXXXX.EEE:             JOB.SEQ (RET)
   *FILE DIRECTORY:                     BILL (RET)
   *FILETYPE (TAPE,DISK6,DISK7,DBMS):   DISK6 (RET)
   *RECORD LENGTH IN CHARS:             240 (RET)
   *BLOCKING FACTOR (RECS/BLOCK):       (RET)
   *LOC OF ISAM KEY:                    (RET)
   *ISAM KEY LENGTH:                    (RET)
   *ISAM KEY TYPE (A OR N):             (RET)
   *ISAM KEY SIGN (S OR U):             (RET)
   *READ PASSWORD REF:                  10 (RET)
   *COPY PASSWORD REF:                  NEXT (RET)
   ```

   Gives the FD entry:

DICT ID	FILE NAME	FILE NAME	DIRECT	TYPE	REC LEN	BLK FAC	KEY LOC	KEY LEN	K T	K S	RD PW	CP PW	RW PW
FD	JOBS	JOB.SEQ	BILL	DISK6	240	0	0	0			10	0	0

2. Changing the previous FD (to ISAM file).

   <QA>DEFINE (RET)

   ```
   *FD,DD,PD,CD,RD,AD,SD,END,SCRUB:     FD (RET)
   *ACTION CODE (A,C,D OR P):           C (RET)
   *DICT NAME - UP TO 30 CHARS:         JOBS (RET)
   *DICT UNLOCKING PASSWORD, IF ANY:    FERN (RET)
   *FILENAME AS XXXXXX.EEE:             JOB.IDX (RET)
   *FILE DIRECTORY:                     (RET)
   *FILETYPE (TAPE,DISK6,DISK7,DBMS):   (RET)
   *RECORD LENGTH IN CHARS:             (RET)
   *BLOCKING  FACTOR (RECS/BLOCK):      3 (RET)
   *LOC OF ISAM KEY:                    10 (RET)
   *ISAM KEY LENGTH:                    12 (RET)
   *ISAM KEY TYPE (A OR N):             A (RET)
   *ISAM KEY SIGN (S OR U):             U (RET)
   *READ PASSWORD REF:                  NEXT (RET)
   ```

   Gives the following revised FD entry:

DICT ID	FILE NAME	FILE NAME	DIRECT	TYPE	REC LEN	BLK FAC	KEY LOC	KEY LEN	K T	K S	RD PW	CP PW	RW PW
FD	JOBS	JOB.IDX	BILL	DISK6	240	3	10	12	A	U	10	0	0

3. Delete a dictionary.

<QA>DEFINE (RET)

```
*FD,DD,FD,CD,RD,AD,SD,END,SCRUB:      FD (RET)
*ACTION CODE (A,C,D OR P):            D (RET)
*DICT NAME - UP TO 30 CHARS:          JACKDIC (RET)

*FD,DD,FD,CD,RD,AD,SD,END,SCRUB:      END (RET)

[END OF DICTIONARY TRANSACTION INPUT.]
 (END OF DICTIONARY RUN)

<QA>
```

CREATING AND EDITING A DICTIONARY

PD

FUNCTION:

The PD transaction creates a PD entry that defines a password to the corresponding level number. The password is used to guard against unauthorized use of the dictionary through IQL. The FD entry and/or one or more DD entries can point to this PD entry using the password level number in order to reference the password.

ELEMENTS:

1. COMMAND: PD

2. PASSWORD REFERENCE:

The password reference is 1 or 2 digits, which identifies the password and gives it a level of protection power. This number is used in the FD entry if you wish to apply overall read, copy, or rewrite protection at the file or data base level. The password level number can also be used in a DD entry if you apply individual password protection to a specific item.

3. PASSWORD:

Any string of up to six characters. Cannot contain embedded spaces.

DISCUSSION:

IQL stores passwords in scrambled form in the dictionary.

You can change passwords by deleting the old PD entry and adding the new one. This is done so that anyone changing the password must show proper authority by furnishing the old password.

IQL does not let you change PD entries or password reference numbers in FD or DD entries unless you have furnished in the FD entry a dictionary unlocking password for such change. This unlocking password is one which is at the same or higher level than the one being changed.

EXAMPLES:

1. Defining a PD entry:

        *FD,DD,PD,CD,RD,AD,SD,END,SCRUB: PD (RET)
        *ACTION CODE (A,C,D):   A (RET)
        *PASSWORD REF (2 DIGITS):   20 (RET)
        *PASSWORD (6 CHARS):   CAROLE (RET)

    Gives the PD entry:

    PD 20 CAROLE

2. Changing a PD entry:

    ```
    *FD,DD,PD,CD,RD,AD,SD,END,SCRUB:    PD (RET)
    *ACTION CODE (A,C,D):               D (RET)
    *PASSWORD REF (2 DIGITS):           20 (RET)
    *PASSWORD:                          CAROLE (RET)

    *FD,DD,PD,CD,RD,AD,SD,END,SCRUB:    PD (RET)
    *ACTION CODE (A,C,D):               A (RET)
    *PASSWORD REF (2 DIGITS):           20 (RET)
    *PASSWORD:                          SHAZAM (RET)
    ```

    Gives the revised PD entry:

    PD 20 SHAZAM

CREATING AND EDITING A DICTIONARY

FUNCTION:

The RD transaction creates a RD entry that describes a data record for DBMS or for data files with more than one record type. For DBMS, the RD transaction maps the IQL input buffers. For data files, the RD transaction identifies a data item -DD- as belonging to a specific record.

ELEMENTS:

1. COMMAND: RD

2. RECORD NAME:

   The name of the record in your schema.

3. RECORD ORIGIN:

   For data files, use a record origin of 1. For DBMS data bases, the record origin maps the record location in the IQL composite input buffer. The record origin must start on a word boundary. When IQL is reading a data base, IQL reads in different record types side by side into a composite buffer. The record origin is the leftmost character position where IQL stores the record. For instance, if you wish to read a data base with two 6-bit record types, the first record type is a length of 100 and the second record type is a length of 200. For it to start on a 6-bit word boundary, the record origin for the RD for the first record is 1 and the record origin for the second record is 103. By mapping the records like this, you have all the items from all the current related records available at the same time.

4. RECORD LENGTH:

   The length of the record in characters.

5. RECORD NUMBER:

   Use 1 in the first RD in the dictionary, 2 in the next RD, and so on. This Record Number is used in IQL's SD (Set Definition) entry to show which records the set relates.

6. DESCRIPTION:

   Up to 76 characters of remarks.

CREATING AND EDITING A DICTIONARY

EXAMPLE:

    1. Defining a new RD entry:

```
*FD,DD,FD,CD,RD,AD,SD,END,SCRUB      RD (RET)
*ACTION CODE (A,C,D):                A (RET)
*RECORD NAME (UP TO 30 CHARS):       DETAIL-RECORD (RET)
*RECORD ORIGIN (UP TO 4 DIGITS):     439 (RET)
*RECORD LENGTH (UP TO 4 DIGITS):     120 (RET)
*RECORD NUMBER (UP TO 3 DIGITS):     2 (RET)
*DESCRIPTION (IF ANY):               DETAIL RECORD FOR MEMBER (RET)
```

Gives the RD entry:

RD DETAIL-RECORD 438 120 2 DETAIL RECORD FOR MEMBER

CREATING AND EDITING A DICTIONARY

SD

FUNCTION:

The SD transaction creates a SD entry that describes a DBMS data base and the records it relates.

ELEMENTS:

1. COMMAND: SD

2. SET NAME:

    The name of the set. Must be the same as the name of the set in your schema.

3. OWNER RECORD NUMBER:

    The number of the record (see the RD entry) that is the owner of the set.

4. MEMBER RECORD NUMBER:

    The number of the record (see the RD entry) that is the member record of the set.

5. DESCRIPTION:

    Up to 76 characters of remarks or descriptive text.

DISCUSSION:

IQL uses SD entries to check set names in the record selection expressions that you specify in the FIND commands.

EXAMPLE:

1. Defining a new PD entry:

    ```
    *FD,DD,PD,CD,RD,AD,SD,END,SCRUB:    SD (RET)
    *ACTION CODE (A,C,D):               A (RET)
    *SET NAME:                          CUSTOMER-SET (RET)
    *SET OWNER RECORD NO.:              10 (RET)
    *SET MEMBER RECORD NO.:             12 (RET)
    *DESCRIPTION (IF ANY):              RELATES SALESMAN/OWNER (RET)
    ```

    Gives the SD entry:

    SD CUSTOMER-SET 10 12 RELATES SALESMAN/OWNER

CHAPTER 6

RUNNING IQL IN BATCH

You can specify any IQL function in batch. Write a control file (a
file with the extension .CTL) using a DIGITAL text editor, and place
the IQL commands and responses in the proper place preceded by an
asterisk (*). If you wish to place comments in the control file, use
a semicolon (;) as the first character of the line.

To submit the batch job, enter a monitor command, such as the TOPS-20
SUBMIT command, followed by the name of the control file. To specify
a time limit or other limits, see the appropriate Operating System
reference manual, such as the <u>TOPS-20 Batch Reference Manual</u>.

The following three batch control files show representative IQL batch
jobs:

    1. QBATCH.CTL

```
@R IQL RET
;SPECIFY A REPORT AND RUN IT RET
*WRITE RET
*HEADING "BROKEN REJECTS - MARCH". RET
*OPEN REJECTS. RET
*IF REASON = 'B' AND MO = 3 AND YR = 78 RET
*   PRINT PART-NO, PRICE, CUSTOMER. RET
* ESC $
*EUN RET
*RUN RET
*EXIT RET
```

        The first line begins with @ to signify an Operating System
        command to run IQL. The lines starting with an asterisk (*)
        contain exactly the same information that you enter on the
        terminal keyboard.   To submit the above control file, you
        might enter:

            @SUBMIT QBATCH.CTL/TIME:00:05:00 RET

    2. QSTACK.CTL

```
@R IQL RET
;RUN A STACKED SET OF ALREADY-WRITTEN QUERIES RET
*RUN REPAIR-SCHEDULE RET
;THE FOLLOWING QUERY HAS ALREADY BEEN ANALYZED AND SAVED RET
*EXECUTE PERFORMANCE-ANALYSIS RET
*RUN MAINTENANCE-BACKLOG RET
;WE KNOW THAT THE ABOVE QUERY WILL ASK FOR A DATE: RET
; SUPPLY IT HERE. RET
*020178 RET
*EXIT RET
```

        You might submit the above with @SUBMIT QSTACK/PAGES:100.

3. QFIX.CTL

```
@R IQL (RET)
;UPDATE THE REPAIR FILE (RET)
*UPDATE REPAIRS (RET)
*CHANGE MONTH TO 05 FOR ALL DEPT 21 AND MO = 04 (RET)
*TOP (RET)
*CHANGE DATE TO 051578 FOR CUSTOMER 'SUBANO' (RET)
*EXIT (RET)
;HERE ARE BACK AT IQL ASSISTANCE LEVEL (RET)
;NOW RUN A REPORT AGAINST THE UPDATED FILE (RET)
*RUN REPAIR-SCHEDULE (RET)
*EXIT (RET)
```

You might submit the above with @SUBMIT QFIX.

APPENDIX A

EXAMPLE ASSISTANCE SESSION

The example below shows a representative assistance session. Although
it does not demonstrate all of the assistance commands, the example
illustrates the various levels of IQL.

```
@R IQL (RET)
<QA>JOB (RET)
 JOB NUMBER: 14

<QA>QUERIES (RET)
 QUERIES STORED IN YOUR DIRECTORY ARE:

 WEEKLY-SUMMARY   ACCOUNT-DETAIL   WX1Z                        WRAPUP
 QRTLY-FORECAST   EXECUTIVES       202-REPORT

<QA>ITEMS REJECTS (RET)

 DICT          FILE     FILE-IN         DIR   REC BLK KEY KY KY RD CP RW
 NAME          TYPE     NAME            ECT   LEN FAC LOC LN TP PW PW PW
 ----          ----     -------         ---   --- --- --- -- -- -- -- --
 REJECTS    SQ DSK7 REJECTSEQ                 150   0

     ITEM         TOP           BOTTOM      1ST  NO.  T S PRINTING      SCAN
  ID NAME         TITLE         TITLE       CHAR CHAR Y C PICTURE       GNNS
 --- ----         -----         ------      ---- ---- - - -------       ----
 DD  PARTNO       PART          NUMBER         1    8 A   XX-XXXX-XX
 DD  SUPPLIER     SUPPLIER      NAME           9   18 A
 DD  PRICE        UNIT          PRICE         63    4 N 2 ZZ.99
     ...(dictionary printout continues)...

<QA>UPDATE REJECTS (RET)
 (YOUR DIARY FILE IS QL014U.LPT)
<QU>FIND PARTNO AUBELT47 (RET)
 AUBELT47DELBERTSON DRIVES 42 AIRDALE ST.  PROVIDENCE RI
 0123921472567012178SUBANDA
<QU>CHANGE REJECT-DATE TO 012378 (RET)
 AUBELT47DELBERTSON DRIVES 42 AIRDALE ST.  PROVIDENCE RI
 0123921472567012378SUBANDA
<QU>EXIT (RET)
 (END OF UPDATE SESSION)
```

```
<QA>WRITE (RET)
INPUT: QC014S.CRD
00100    HEADING 'REJECTS//DUE TO CORROSION'.(RET)
00200    DISPLAY ON.,(RET)
00300    OPEN REJECTS.(RET)
00400    IF REASON = 'C'(RET)
00500        PRINT PARTNO,SUPPLIER,PRICE,REJECT-DATE,CUSTOMER(RET)
00600        TALLY PARTNO.(RET)
00700  (ESC)  $
*E (RET)
  [QC014S.CRD]

<QA>RUN (RET)

   03/30/78                    REJECTS                    PAGE 1
                            DUE TO CORROSION

   PART            SUPPLIER         UNIT    DATE       CUSTOMER
   NO.             NAME             PRICE   REJECTED   NAME

   A2-BKKT-72      ALUMINIZER         .78   01/21/78   SUBANDA
   W3-FRME-21      WESTERN           2.47   02/22/78   HAWTHORNE
   P3-PSTN-76      RANDALL           7.50   12/20/77   RACINE PRODUCTS
        ...(report continues)...

<QA>STORE CORROSION-RPT (RET)
  (CORROSION-RPT STORED)

<QA>DELETE ACCOUNT-DETAIL WX1Z (RET)
  (ACCOUNT-DETAIL DELETED)
  (WX1Z DELETED)

<QA>QUERIES (RET)
  QUERIES STORED IN YOUR DIRECTORY ARE:

  WEEKLY-SUMMARY   WRAPUP            QRTRLY-FORECAST   EXECUTIVES
  202-REPORT       CORROSION-RPT

<QA>EXIT (RET)
  (END OF IQL SESSION)
```

APPENDIX B

EXAMPLE IMMEDIATE MODE SESSION

Shown below is an example of a representative immediate mode session.
First, the example illustrates creating a new file at the input level
and inputing a few records. Then the example illustrates ending the
input level and entering the update level to do the following:

    1. Browse through the file.

    2. Change a few records.

    3. Insert more records.

    4. Browse through the file.

The following condensed dictionary is referenced in the example. Note
that the fields marked with asterisks (that is: DD*) are used to
control retrieval but do not prompt for input. Thus, you can redefine
fields for use in retrieval without IQL prompting for the field while
at the input level.

DICT NAME	FILE TYPE	FILE-IN NAME	DIR ECT	REC LEN	BLK LEN
SALES	DISK6	SALES.SEA	DIR-SALES	120	0

ITEM ID NAME	TOP TITLE	BOTTOM TITLE	1ST CHAR	NO. CHAR	T Y	S C	PRINTING PICTURE
DD CUSTNO	CUSTOMER	NUMBER	1	5	N	0	99999
DD CUSTOMER	CUSTOMER	NAME	6	16	A	0	
DD BUYER	BUYER	NAME	22	15	A	0	
DD CYSALES	CURR	SALES	37	9	N	2	ZZZZZZZ.99
DD LYSALES	LAST YR	SALES	46	9	N	2	$$$$$,$$$.99
DD LPDATE	LAST PO	DATE	55	6	N	0	99/99/99
DD*LPYR	LAST PO	YEAR	55	2	N	0	99
DD*LPMO	LAST PO	MONTH	57	2	N	0	99
DD*LPDAY	LAST PO	DAY	59	2	N	0	99

In the dialogue shown below, the information you would enter is shown
in red print.

The example shows starting the session by running IQL and then
entering INPUT as an IQL command. The system prompts you for the name
of the dictionary, displays progress report information, and goes
immediately to the input level. You then furnish the data to build
the file from scratch. Next you list all the records in full, look
for a specific record on a condition, change a value in a record, and
insert a new record.

Following the insertion, you go to the top of the file and this time
you list selected fields for all records. Then you move up two
records, make a single change, move down one record, and make a double
change. Now you append a record to the end of the file. After
appending, you go to the top of the file and list specific fields for
only records that meet a specific criterion. Finally, you list all
the fields in all the records.

This example shows some of the different formats of the commands and
some of the different ways item values can be furnished.

```
@R IQL (RET)
<QA> INPUT SALES (RET)

(NEXT RECORD)
 *CUSTNO (CUSTOMER NUMBER) 5 N:   1(RET)
 *CUSTOMER (CUSTOMER NAME) 16 A: HEARTHSIDE HOMES (RET)
 *BUYER (BUYER NAME) 15 A:        'LOIS J SCRIBNER'(RET)
 *CYSALES (CURR YR SALES) 7.2 N: 45,000 (RET)
 *LYSALES (LAST YR SALES) 7.2 N: $23,456.78 (RET)
 *LPDATE (LAST PO DATE) 6 N:       780127 (RET)

(NEXT RECORD)
 *CUSTNO (CUSTOMER NUMBER) 5 N:   5 (RET)
 *CUSTOMER (CUSTOMER NAME) 16 A: "RIORDAN INC." (RET)
 *BUYER (BUYER NAME) 15 A:        "RALPH HOPWOOD" (RET)
 *CYSALES (CURR YR SALES) 7.2 N: 500 (RET)
 *LYSALES (LAST YR SALES) 7.2 N: 0 (RET)
 *LPDATE (LAST PO DATE) 6 N:       780110 (RET)

(NEXT RECORD)
 *CUSTNO (CUSTOMER NUMBER) 5 N:   12 (RET)
 *CUSTOMER (CUSTOMER NAME) 16 A: 'HEART ASSOCIATES' (RET)
 *BUYER (BUYER NAME) 15 A:        PURCHASING (RET)
 *CYSALES (CURR YR SALES) 7.2 N: $11,463 (RET)
 *LYSALES (LAST YR SALES) 7.2 N: 10000 (RET)
 *LPDATE (LAST PO DATE) 6 N:       780217 (RET)

(NEXT RECORD)
 *CUSTNO (CUSTOMER NUMBER) 5 N:   END (RET)
<QU> TOP (RET)
(AT BEGINNING OF FILE)
<QU> LIST ALL (RET)
00001HEARTHSIDE HOMESLOIS J SCRIBNER004500000002345678780127
00005RIORDAN INC.    RALPH HOPWOOD   00005000000000000780110
00012HEART ASSOCIATESPURCHASING      00114630000100000780217
(AT END OF FILE)
<QU> FIND FIRST CUSTNO = 5 (RET)
00005RIORDAN INC.    RALPH HOPWOOD   00005000000000000780110
<QU> CHANGE CYSALES TO 1,111 (RET)
00005RIORDAN INC.    RALPH HOPWOOD   00011110000000000780110
<QU> INSERT (RET)

(NEXT RECORD)
 *CUSTNO (CUSTOMER NUMBER) 5 N:   7 (RET)
 *CUSTOMER (CUSTOMER NAME) 16 A: "FLORIDA YACHTS" (RET)
 *BUYER (BUYER NAME) 15 A:        'COL. RAY AHAB'(RET)
 *CYSALES (CURR YR SALES) 7.2 N: 650.03(RET)
 *LYSALES (LAST YR SALES) 7.2 N: 650 (RET)
 *LPDATE (LAST PO DATE) 6 N:       780130 (RET)
```

```
(NEXT RECORD)
 *CUSTNO (CUSTOMER NUMBER) 5 N:   END (RET)


<QU> TOP (RET)
(AT BEGINNING OF FILE)
<QU> LIST CUSTNO CUSTOMER BUYER CYSALES FOR ALL RECORDS (RET)
00001   HEARTHSIDE HOMES LOIS J SCRIBNER    45000.00
00005   RIORDAN INC.     RALPH HOPWOOD       1111.00
00007   FLORIDA YACHTS   COL. RAY AHAB         650.03
00012   HEART ASSOCIATES PURCHASING         11463.00
(AT END OF FILE)
<QU> UP 2 (RET)
00007FLORIDA YACHTS  COL. RAY AHAB   000065003000065000780130
<QU> CHANGE CYSALES TO -500 (RET)
00007FLORIDA YACHTS  COL. RAY AHAB   000050003000065000780230
<QU> DOWN (RET)
00012HEART ASSOCIATESPURCHASING      00114630000100000780217
<QU> CHANGE BUYER TO "FLOYD C CALHOUN" CYSALES TO 12,463 (RET)
00012HEART ASSOCIATESFLOYD C CALHOUN00124630000100000780217
<QU> APPEND (RET)
(AT END OF FILE)


(NEXT RECORD)
 *CUSTNO (CUSTOMER NUMBER) 5 N:   17 (RET)
 *CUSTOMER (CUSTOMER NAME) 16 A: "BATATTA BRANDY" (RET)
 *BUYER (BUYER NAME) 15 A:        "JACQUES LEROUX" (RET)
 *CYSALES (CURR YR SALES) 7.2 N: 1000 (RET)
 *LYSALES (LAST YR SALES) 7.2 N: $1,234.7 (RET)
 *LPDATE (LAST PO DATE) 6 N:      780115 (RET)


(NEXT RECORD)
 *CUSTNO (CUSTOMER NUMBER) 5 N:   END (RET)
<QU> TOP (RET)
(AT BEGINNING OF FILE)
<QU> LIST CUSTNO CUSTOMER CYSALES FOR ALL CYSALES GEQ 10,000 (RET)
00001   HEARTHSIDE HOMES    45000.00
00012   HEART ASSOCIATES    12463.00
(AT END OF FILE)
<QU> TOP (RET)
(AT BEGINNING OF FILE)
<QU> LIST CUSTOMER BUYER CYSALES LYSALES ALL (RET)
HEARTHSIDE HOMES LOIS J SCRIBNER    45000.00     $23,456.78
RIORDAN INC.     RALPH HOPWOOD       1111.00          $.00
FLORIDA YACHTS   COL. RAY AHAB        500.00-       $650.00
HEART ASSOCIATES FLOYD C CALHOUN    12463.00     $10,000.00
BATTATA BRANDY   JACQUES LEROUX      1000.00      $1,234.70
(AT END OF FILE)
<QU> EXIT (RET)
(AT END OF UPDATE SESSION)

<QA>          (here we are back at assistance level)
```

APPENDIX C

EXAMPLE DICTIONARIES

The following dictionary describes an ASCII Indexed Sequential file:

DICT NAME	FILE TYPE	FILE-IN NAME	DIRECT	REC LEN	BLK FAC	KEY LOC	KY LN	KY TP	RD PW	CP PW	RW PW
PERSONNEL	IS DSK7	PERSONIDX	<IQL30-DIST>	300	2	1	5	AU	10	30	50

ID	ITEM NAME	TOP TITLE	BOTTOM TITLE	1ST CHAR	NO. CHAR	T Y	S C	PRINTING PICTURE	SCAN GNNS	PT
PD	TIGER			50						
PD	BEAR			30						
PD	FOX			10						
DD	EMPNO	EMPLOY	NUMBER	1	5	N	0	ZZZZ9		
DD	LNAME-CH	LNAME	CHAR	28	1	A	0			
DD	SEX		SEX	271	1	A	0			
DD	CAT	CAT		6	1	A	0			
DD	CO	CO		7	3	N	0	ZZ9		
DD	DIV	DIV		10	4	N	0	ZZZ9		
DD	DEPT	DEPT		14	5	A	0			
DD	LOCATOR		LOCATOR	7	12	A	0	XXX-XXXX-XXXXX		
DD	PHONE	FULL	PHONE	212	10	N	0	(999)999-9999		
DD	SOC-SEC	SOCIAL	SECURITY	19	9	N	0	999-99-9999		
DD	LNAME	LAST	NAME	28	15	A	0			
DD	FNAME	FIRST	NAME	43	10	A	0			
DD	INIT	I		53	1	A	0			
DD	NAME	FULL NAME		28	26	A	0			
DD	STREET	STREET		55	20	A	0			
DD	COLLEGE-ID	COLL	CODE	77	2	A	0			
DD	JOB-ID	JOB	CODE	79	2	N	0	99		
DD	CITY	CITY		154	20	A	0			
DD	STATE	STATE		181	14	A	0			
DD	MAILCODE	ZIP		207	5	N	0	99999		
DD	ZIP	ZIP	CODE	207	5	N	0	99999		
DD	AREA	AREA	CODE	212	3	N	0	(999)		
DD	LCL-PHN	LOCAL	PHONE	215	7	N	0	999-9999		
DD	SALARY	SALARY	WEEKLY	222	6	N	2	SZZZ9.99	30	
DD	HRLY	HOURLY	RATE	228	5	N	3	Z9.999	30	
DD	OT-HRLY	OVT-TME	RATE	233	5	N	3	Z9.999		
DD	OLD-SAL	PREVIOUS	SALARY	238	6	N	2	ZZZ9.99		
DD	OLD-HRL	PREV-HRLY	RATE	244	5	N	3	Z9.999		
DD	DATE-LR	LAST RAISE	YY-MM-DD	249	6	N	0	99-99-99		
DD	DATE-LRA	DATE	LRA	249	6	A	0	XX-XX-XX		
DD	YR-LR	YR	LR	249	2	N	0			
DD	MO-LR	MO	LR	251	2	N	0			

DD DA-LR	DAY	LR	253	2	N	0		
DD DATE-HR	DATE	HIRED	255	6	N	0	99-99-99	
DD YR-HR	YR	HR	255	2	N	0		
DD MO-HR	MO	HR	257	2	N	0		
DD DA-HR	DAY	HR	259	2	N	0		
DD DATE-BD	DATE	BIRTH	261	6	N	0	99-99-99	
DD YR-BD	YR	BD	261	2	N	0		
DD MO-BD	MO	BD	263	2	N	0		
DD DA-BD	DAY	BD	265	2	N	0		
DD TEST	TEST	RESULT	267	4	N	2	Z9.99%	
DD ACT	ACT	STAT	271	1	A	0		
DD SKILL1	SKILL	1	273	4	A	0	XXX-X	
DD CODE1	CDE	1	273	3	A	0		
DD LVL1	LVL	1	276	1	A	0		
DD SKILL2	SKILL	2	277	4	A	0	XXX-X	
DD CODE2	CDE	2	277	3	A	0		
DD LVL2	LVL	2	280	1	A	0		
DD SKILL3	SKILL	3	281	4	A	0	XXX-X	
DD CODE3	CDE	3	281	3	A	0		
DD LVL3	LVL	3	284	1	A	0		
DD SKILL4	SKILL	4	285	4	A	0	XXX-X	
DD CODE4	CDE	4	285	3	A	0		
DD LVL4	LVL	4	288	1	A	0		
DD SKILL5	SKILL	5	289	4	A	0	XXX-X	
DD CODE5	CDE	5	289	3	A	0		
DD LVL5	LVL	5	292	1	A	0		
DD SKILL6	SKILL	6	293	4	A	0	XXX-X	
DD CODE6	CDE	6	293	3	A	0		
DD LVL6	LVL	6	296	1	A	0		
DD SKILL7	SKILL	7	297	4	A	0	XXX-X	
DD CODE7	CDE	7	297	3	A	0		
DD LVL7	LVL	7	300	1	N	0		
DD DATE-CH	CHANGE	DATE	81	6	N	0	99-99-99	A129
DD YR-CH	CH	YR	81	2	N	0	SZ9	A129
DD MO-CH	CH	MO	83	2	N	0	SZ9	A129
DD DA-CH	CH	DA	85	2	N	0	SZ9	A129
DD SKILL	JOB	SKILL	273	4	A	0	XXX-X	B07
DD SK-CDE	SKL	CDE	273	3	A	0		B07
DD SK-C1	S	1	273	3	A	0		B07
DD SK-C2	S	2	274	3	A	0		B07
DD SK-C3	S	3	275	3	A	0		B07
DD SK-LVL	SKL	LVL	276	1	A	0	X	B07

(END LIST OF ITEMS)

EXAMPLE DICTIONARIES

The following dictionary describes an ASCII sequential file:

DICT NAME	FILE TYPE	FILE-IN NAME	DIRECT	REC LEN	BLK FAC	KEY LOC	KY LN	KY TP	RD PW	CP PW	RW PW
CUSTOMERS	SQ DSK7	CUSTMRSEQ		145	0	0	0				

ID	ITEM NAME	TOP TITLE	BOTTOM TITLE	1ST CHAR	NO. CHAR	TY	SC	PRINTING PICTURE	SCAN GNNS	PT
DD	CUSTNO	CUSTOMER'S	VENDOR NO.	1	5	N	0	SZZZZ9		
DD	CNAME	CUSTOMER	NAME	6	30	A	0			
DD	BUYER	BUYER	NAME	36	26	A	0			
DD	STREET	CUST	STREET	62	25	A	0			
DD	CITY	CUST	CITY	87	25	A	0			
DD	STATE	CUST	ST	112	2	A	0			
DD	ZIP	CUST	ZIP	114	5	N	0	SZZZZ9		
DD	CYSALES	YR TO DATE	PURCHASES	119	9	N	2	$$,$$$,$$$.$$		
DD	CYPAID	Y-T-D	PAID	128	9	N	2	$$,$$$,$$$.$$		
DD	CLIMIT	CREDIT	LIMIT	137	9	N	2	$$,$$$,$$$.$$		
DD	SLSMAN	SALESMAN	NUMBER	2	4	N	0	SZZZ9		

(END LIST OF ITEMS)

The following dictionary describes a DBMS data base:

DICT NAME	FILE TYPE	FILE-IN NAME	DIRECT	REC LEN	BLK FAC	KEY LOC	KY LN	KY TP	RD FW	CP FW	RW FW
EMPLOYEES	DTABASE	EMPLOYSCH		600	1	0	0				

ID	ITEM NAME	TOP TITLE	BOTTOM TITLE	1ST CHAR	NO. CHAR	T Y	S C	PRINTING PICTURE	SCAN GNNS	PT
AD	PERSON-AREA			0	0					
AD	PAYROLL-AREA			0	0					
AD	COLLEGE-AREA			0	0					
AD	JOBNAME-AREA			0	0					
RD	PERSONNEL			1	300	001	Q5			
DD	EMPNO	EMPLOY	NO	1	5	N	0	SZZZZ9		
DD	LNAME-CH	LNAME	CHAR	28	1	A	0			
DD	SEX		SEX	271	1	A	0			
DD	CAT	CAT		6	1	A	0			
DD	CO	CO		7	3	N	0	ZZ9		
DD	DIV	DIV		10	4	N	0	ZZZ9		
DD	DEPT	DEPT		14	5	A	0			
DD	LOCATOR	EMPLOYEE	LOCATOR	1	18	A	0			
DD	PHONE	FULL	PHONE	212	10	N	0	(999)999-9999		
DD	SOC-SEC	SOCIAL	SECURITY	19	9	N	0	999-99-9999		
DD	LNAME	LAST	NAME	28	15	A	0			
DD	FNAME	FIRST	NAME	43	10	A	0			
DD	INIT	I		53	1	A	0			
DD	NAME	NAME		28	26	A	0			
DD	STREET	STREET		55	20	A	0			
DD	YR1	FIRST	YEAR	81	2	N	0			
DD	MO1	FRST	MNTH	83	2	N	0			
DD	DA1	FRST	DAY	85	2	N	0			
DD	DATE1	FIRST	DATE	81	6	N	0	99/99/99		
DD	CITY	CITY		154	20	A	0			
DD	STATE	STATE		181	14	A	0			
DD	MAILCODE	ZIP		207	5	N	0			
DD	AREA	AREA	CODE	212	3	N	0	(999)		
DD	LCL-PHN	LOCAL	PHONE	215	7	N	0	999-9999		
DD	SALARY	SALARY	WEEKLY	222	6	N	2	SZZZ9.99		
DD	HRLY	HOURLY	RATE	228	5	N	3	Z9.999		
DD	OT-HRLY	OVT-TME	RATE	233	5	N	3	Z9.999		
DD	OLD-SAL	PREVIOUS	SALARY	238	6	N	2	ZZZ9.99		
DD	OLD-HRL	PREV-HRLY	RATE	244	5	N	3	Z9.999		
DD	DATE-LR	LAST RAISE	YY-MM-DD	249	6	N	0	99-99-99		
DD	YR-LR	YR	LR	249	2	N	0			
DD	MO-LR	MO	LR	251	2	N	0			
DD	DA-LR	DAY	LR	253	2	N	0			
DD	DATE-HR	DATE	HIRED	255	6	N	0	99-99-99		
DD	YR-HR	YR	HR	255	2	N	0			
DD	MO-HR	MO	HR	257	2	N	0			
DD	DA-HR	DAY	HR	259	2	N	0			
DD	DATE-BD	DATE	BIRTH	261	6	N	0	99-99-99		
DD	YR-BD	YR	BD	261	2	N	0			
DD	MO-BD	MO	BD	263	2	N	0			
DD	DA-BD	DAY	BD	265	2	N	0			
DD	TEST	TEST	RESULT	267	4	N	2	Z9.99%		

```
DD ACT       ACT       STAT      271    1 A 0
DD SKILL1    SKILL     1         273    4 A 0  XXX-X
DD CODE1     CDE       1         273    3 A 0
DD LVL1      LVL       1         276    1 A 0
DD SKILL2    SKILL     2         277    4 A 0  XXX-X
DD CODE2     CDE       2         277    3 A 0
DD LVL2      LVL       2         280    1 A 0
DD SKILL3    SKILL     3         281    4 A 0  XXX-X
DD CODE3     CDE       3         281    3 A 0
DD LVL3      LVL       3         284    1 A 0
DD SKILL4    SKILL     4         285    4 A 0  XXX-X
DD CODE4     CDE       4         285    3 A 0
DD LVL4      LVL       4         288    1 A 0
DD SKILL5    SKILL     5         289    4 A 0  XXX-X
DD CODE5     CDE       5         289    3 A 0
DD LVL5      LVL       5         292    1 A 0
DD SKILL6    SKILL     6         293    4 A 0  XXX-X
DD CODE6     CDE       6         293    3 A 0
DD LVL6      LVL       6         296    1 A 0
DD SKILL7    SKILL     7         297    4 A 0  XXX-X
DD CODE7     CDE       7         297    3 A 0
DD LVL7      LVL       7         300    1 N 0
DD DATE-CH   CHANGE    DATE      81     6 N 0  99-99-99      A129
DD YR-CH     CH        YR        81     2 N 0  SZ9           A129
DD MO-CH     CH        MO        83     2 N 0  SZ9           A129
DD DA-CH     CH        DA        85     2 N 0  SZ9           A129
DD SKILL     JOB       SKILL     273    4 A 0  XXX-X         B07
DD SK-CDE    SKL       CDE       273    3 A 0                B07
DD SK-C1     S         1         273    3 A 0                B07
DD SK-C2     S         2         274    3 A 0                B07
DD SK-C3     S         3         275    3 A 0                B07
DD SK-LVL    SKL       LVL       276    1 A 0  X             B07

RD PAYROLL                       301  100 002 Q5
DD P-JOBID   JOB       ID        6      2 N 0  SZ9
DD JOBNO     PAY       JNO       6      2 N 0  SZ9
DD COLLNO    PAY       CNO       8      2 N 0  SZ9
DD PAYNO     PAY       NO        1      5 N 0  SZZZZ9
DD P-COLID   COLLEGE   ID        8      2 N 0  SZ9
DD P-NAME                        1      5 N 0  ZZZZ9
DD P-SSNO    SOCIAL    SECURITY  10     9 N 0  999-99-9999
DD P-CAT               CATEGORY  19     1 A 0
DD DEPNDTS   DEPNDTS   CLAIMED   20     2 N 0  SZ9
DD P-HRLY    HOURLY    RATE      22     5 N 3  Z9.999
DD P-OT      OVER      TIME      27     5 N 3  Z9.999
DD P-SAL               SALARY    32     6 N 2  SZZZ9.99

RD COLLEGE                       401  100 003 Q5
DD DB-CNO                        1      2 N 0  SZ9
DD DB-CNAME  COLLEGE   NAME      3     30 A 0

RD JOBS                          501  100 004 Q5
DD DB-JNO                        1      2 N 0  SZ9
DD DB-JNAME  JOB       NAME      3     30 A 0

SD PERSON-SET                    0      0

SD PAYROLL-SET                   0      0

SD JOB-SET                       0      0

SD COLLEGE-SET                   0      0

   (END LIST OF ITEMS)
```

APPENDIX D

MAXIMUMS

The following maximums apply to each IQL query and the files and data items that IQL processes:

Input files to a query	3
Output files from a query	1 copy file 1 created file per query stage
Sorts	100 approximately The number varies depending upon the query.
Sort keys	100 approximately The number of sort keys cannot exceed the length of the record, which varies with the query and with the length of all the keys taken together.
Reports	99
Length of a query	200 statements, approximately
Length of a literal	72 characters
Length of a numeric item	18 digits
Length of an alpha item	354 characters However, IQL only tests the first 54 characters in an IF query statement.
Length of a variable	354 characters However, IQL only tests the first 54 characters in an IF query statement.

Record lengths:

Input sequential files	6-bit format	4095 characters
	7-bit format	3410 characters
Original ISAM file	6-bit format	1207 characters
	7-bit format	1207 characters
2nd and 3rd ISAM files	6-bit format	768 characters
	7-bit format	640 characters
Copied or created files	6-bit format	768 characters
	7-bit format	640 characters

MAXIMUMS

Aggregate limits   The sum of all the input file record
                                    lengths, all held item lengths, and all
                                    alphavariable lengths cannot exceed 4095
                                    characters.

                                    The sum of all the records to be sorted
                                    and the aggregate length of the sort key
                                    cannot exceed 4095 characters.

                                    The sum of the number of all the data
                                    items, alphavariables, numeric
                                    variables, constants, literals, and
                                    reports cannot exceed 200.

Many of the limits can be changed, if necessary, by the DIGITAL
software specialist assigned to the site.

APPENDIX E

SUMMARY OF EDITOR COMMANDS

The EDIT and WRITE assistance commands invoke an editor. Use the WRITE assistance command to create a query; use the EDIT assistance command to change a query. On the TOPS-10 Operating System, you invoke the editor supplied with IQL. On the TOPS-20 Operating System, you invoke the EDIT editor, which is supplied with the system.

E.1 IQL EDITOR

The IQL editor uses the colon (:) for its string delimiter. The IQL editor does not use line numbers. The editor displays an asterisk and awaits for a command. The IQL editor commands which are listed below:

D.	Delete the current line of the query.
L	Move down one line and list it.
END	If the editor is at the insert level terminate it and return to the command level of the editor. If the editor is at the command level, exit the editor and return to the assistance level.
Fstring:	Move to and list the next line which contains the designated string.
I.	Insert one line of text after this line. End the line with a carriage return.
I	Insert several lines of text after this line. Continue to insert lines until you enter a carriage return as the first character of a line.
P.	Display the current line.
P;ALL	Display the current line and all subsequent lines.
Sstring1:string2:.	Substitute string2 for string1 in the current line.
Sstring1:string2:ALL	Substitute string2 for string1 in the current and all following lines.
TOP	Go to the beginning of the query.

SUMMARY OF EDITOR COMMANDS

E.2 TOPS-20 EDITOR

The TOPS-20 editor is described in the <u>TOPS-20 EDIT Reference Manual</u>.
However, the following TOPS-20 EDIT command summary is provided:

*	The editor displays an asterisk (*) on the terminal to indicate that the editor is ready to accept a command.
$	The dollar symbol denotes the escape key on the terminal.
D.	Delete the current line of the query.
linefeed key	Move down one line.
escape key	If the editor is at the insert level, terminate it and return to the command level of the editor. If the editor is at the command level, display the previous line of text.
Fstring$	Move to the next query statement that contains the designated string and display it.
I.	Insert one line of text after this line. The editor prompts you for a new line by displaying the line number and waits for you to enter the line. End the line with a carriage return.
I.,2	Insert several lines of text after this line. Continue to insert lines until you enter the escape key as the first character in a line. The editor increments the line numbers by two.
P.	Display the current line.
P	Display the current line and the next 15 lines.
Sstring1$string2$.	Substitute string2 for string1 in the current line.
P^	Display the first line in the query.
Pnn	Display line nn.
P nn:mm	Display lines nn through mm.
e	Exit the editor and return to the assistance level.

APPENDIX F

IQL ERROR MESSAGES

Error messages from IQL are preceded by either a question mark (?), which indicates a fatal error or a percent sign (%), which indicates either a warning message or an information message. When IQL encounters a fatal error, IQL displays the error message, terminates IQL, and returns you to the Operating System. When IQL encounters a nonfatal error, IQL displays the warning or information message and continues the operation.

IQL issues the error messages listed in this appendix at various locations during the use of IQL. For instance in the immediate mode, IQL displays an error message after you enter the offending command or data value. In the deferred mode, IQL displays an error message after IQL reads the offending command in the query or at the completion of the query run. In either case, the error message and the indicated corrective action are self-explanatory; hence no additional text is given in this appendix.

A lowercase word in the message text indicates that IQL fills in the word before issuing the message.

If you receive any other error message from IQL and the meaning is not immediately clear, it may be a problem with IQL itself and you should promptly report it as an SPR.

F.1 ERROR MESSAGES FROM IMMEDIATE MODE

The following error messages apply to the overall assistance level:

    %invalid response - please reenter.
    %dummy files isamf6.idx and isamf7.idx are not in your
     directory/ppn. you need them to run a query.

The following messages can occur when you store, run, or get a query:

    %THERE IS NO QUERY TO STORE.
    %THERE IS NO QUERY TO REPLACE WITH.
    %THERE IS NO CURRENT QUERY TO RUN.
    %THERE IS NO CURRENT QUERY.
    %query-name IS ALREADY STORED. PLEASE DELETE IT OR
     STORE UNDER A NEW NAME.

IQL ERROR MESSAGES

The following messages can appear on a TOPS-10 Operating System during an edit of a query:

```
%COMMAND ERROR.
%STRING LONGER THAN 40.
%STRING NOT FOUND.
%POSSIBLE STRING OVERFLOW.
```

IQL displays the following error messages resulting from a dictionary definition:

```
%FIRST DEFINITION WAS NOT FD: RENTER.
%ILLEGAL ENTRY CODE - REJECTED.
%ACTION CODE ILLEGAL.
%TRANSACTION REJECTED - REENTER.
%THIS IS A RESERVED WORD; CHANGE OR USE A SUFFIX.
%NAME MISSING.
%NAMES MAY NOT CONTAIN SPACES OR COMMAS.
%NAMES MAY CONTAIN ONLY A-Z,0-9,DASH.
%FD ACTION TYPE NOT A,C,D,S OR P.
%THE "S" ACTION CODE WILL DESTROY ALL CURRENT
  DICTIONARIES IN YOUR DIRECTORY/PPN. DO YOU
  YOU REALLY WANT TO DO THIS? IF SO REENTER S;
  IF NOT ENTER THE ACTION CODE YOU REALLY WANT.
%FILE NAME IMPROPERLY FORMATTED.
%RECORD SIZE TOO LARGE.
%RECORD SIZE TOO SMALL.
%BLOCK SIZE TOO LARGE.
%BLOCK FACTOR MUST BE NON-0 FOR ISAM.
%FILE TYPE ILLEGAL.
%NO KEY POSITION FOR ISAM FILE.
%KEY POSITION OUTSIDE OF ISAM RECORD.
%NO LENGTH FOR ISAM KEY.
%LENGTH FOR ISAM KEY EXCEEDS MAX.
%BLOCK SIZE NOT MULTIPLE OF REC SIZE.
%ISAM KEY TYPE NOT ALPHA (0) OR NUMERIC (1).
%ISAM KEY SIGN INCORRECT.
%ITEM NAME STARTS WITH ILLEGAL "X" OR "ZZ".
%NO FIRST CHAR LOCATION.
%ITEM OUTSIDE RECORD.
%NO ITEM LENGTH.
%ITEM LENGTH TOO LONG.
%ITEM TYPE ILLEGAL.
%SCALE LARGER THAN ITEM LENGTH.
%PIC POSITIONS/ITEM MISMATCH.
%NO SCAN GROUP NAME.
%SCAN REPEATS RUN OUT OF RECORD.
%NO SCAN REPEATS.
%ILLEGAL PASSWORD EXCL FLAG.
%NOT NUMERIC.
%TITLES MAY NOT EXCEED 10 CHARS.
```

IQL displays an message listed in this section after you enter an END or EXIT dictionary command and IQL cannot apply the transactions you define to a dictionary. There are some duplications with the error messages that may occur during prompted definition, because in some cases you can generate dictionary transactions by other than prompting (that is, by a program such as IQSCH) and IQL must perform all the checks at this time.

```
%RESULTING FD ENTRY MAY NEED CORRECTION.
%RECORD LGTH WAS TOO LARGE- CHANGED TO SYSTEM MAXIMUM.
```

%RECORD LGTH WAS TOO SMALL- CHANGED TO SYSTEM MINIMUM.
%BLOCK FACT FOR ISAM FILES MUST BE > 0; SET TO 1.
%BLOCK FACT WAS TOO SMALL- CHANGED TO 1.
%FILE TYPE WAS ILLEGAL- CHANGED TO 7 (ASCII DISK).
%NO KEY POSITION FOR ISAM FILE- CHANGED TO 1.
%KEY POSITION OUTSIDE OF ISAM RECORD- CHANGED TO 1.
%NO LENGTH FOR ISAM KEY- CHANGED TO 1.
%BLOCK FACT NOT MULTIPLE OF REC LEN - SET TO 0.
%field-name WAS NOT NUMERIC- SET TO 0.
%KEY DATA FURNISHED FOR NON-ISAM FILE - IGNORED.
%ILLEGAL KEY TYPE FOR ISAM FILE- SET TO ALPHABETIC.
%ILLEGAL KEY SIGN FOR ISAM FILE- SET TO UNSIGNED.
%DICT NAME WAS BLANK- SET TO "BAD-NAME".
%0 BLOCKSIZE ONLY VALID FOR SEQUENTIAL DISK FILES.
%DD TRANSACTION REJECTED.
%ACTION CODE WAS ILLEGAL.
%ITEM NAME STARTED WITH X OR ZZ.
%NO FIRST CHAR LOCATION.
%ITEM PARTLY OR ALL OUTSIDE OF RECORD.
%NO ITEM LENGTH.
%ITEM TYPE WAS ILLEGAL.
%SCALE WAS LARGER THAN ITEM LENGTH.
%PICTURE POSITIONS/ITEM MISMATCH.
%PICTURE DECIMAL DID NOT MATCH SCALE.
%NO SCAN GROUP NAME.
%SCAN REPEATS RAN OUTSIDE OF RECORD.
%NO SCAN REPEATS.
%NO PD FOR REFERENCE PROTECTION.
%ILLEGAL VALUE FOR PROT EXCL FLAG.
%field-name WAS NOT NUMERIC.
%NAME WAS ALL BLANKS.
%LENGTH OF MEMBER EXCEEDS GROUP LENGTH.
%PD TRANSACTION REJECTED.
%PROTECTION NO. WAS NOT NUMERIC.
%DICT NOT UNLOCKED FOR PW CHANGES.

IQL displays the error messages listed in this section while you
input, update, or browse a data file.

%NO DICTIONARIES FOUND IN YOUR DIRECTORY/PPN; RUN ENDED.
%dictionary-name DICTIONARY NOT FOUND.
%CANNOT PROCESS NON-DISK FILES.
%file-name NOT FOUND; TRY AGAIN.
%file-name NOT FOUND; ENTERING INPUT.
%file-name ALREADY THERE; ENTERING UPDATE
%COMMAND ERROR; PLEASE REENTER.
%TOO MANY FIND ITEMS.
%VALUE-RELATIONSHIP ERROR.
%ITEM-ITEM TESTS NOT IN THIS VERSION.
%ONLY BROWSE PERMITTED - REJECTED.
%item-name NOT FOUND IN DICTIONARY.
%DICTIONARY CAPACITY EXCEEDED.
%OUTSIDE OF RECORD
%COLUMNS EXCEED RECORD LENGTH
%NO CURRENT RECORD TO DELETE.
%TOO MANY CHANGE ITEMS
%TOO MANY LIST ITEMS.
%DISPLAY LINE WILL BE TRUNCATED.
%item-name VALUE TOO LONG OR BAD FORMAT.
%RECORD REJECTED
%THIS VERSION CANNOT INPUT ISAM FILES.
%APPEND NOT MEANINGFUL FOR ISAM; ENTERING INPUT.
%KEY ENTRY MYST BE AT START OF FINDLIST.
%SAVE NOT MEANINGFUL FOR ISAM.

&THIS VERSION CANNOT EXTRACT ISAM FILES.
&ISAM RECORD NOT FOUND TO DELETE.
&INSERTED ISAM RECORD ALREADY EXISTS; USE CHANGE
 OR DELETE AND REINSERT.
&KEY VALUE NOT FOUND IN ISAM FILE.
&ILLEGAL TO MODIFY ISAM READ KEY.

F.2 ERROR MESSAGES FROM DEFERRED MODE

IQL displays the error messages listed in this section after you enter
the RUN or SAVE assistance command.

&ITEM-NAME NOT DEFINED.
&item-name NOT NUMERIC.
&item-name NOT ALPHABETIC.
&STATEMENT EXPECTED.
&VERB EXPECTED.
&NON VERB EXPECTED.
&INVALID CHARACTER.
&VALUE BEYOND RANGE.
&file-name FILE NOT FOUND.
&TOO MANY RIGHT PAREN.
&NO ENDING QUOTE.
&QUERY SIZE EXCEEDED.
&item-name INVALID ITEM TYPE.
&INVALID/MISSING PASSWORD FOR.
&LITERAL EXPECTED.
&LITERAL MAX EXCEEDED.
&NO ENDING PAREN.
&ISAM FILE REQUIRED.
&HOLD BUFFER FULL.
&INVALID OPERATOR.
&INVALID PICTURE.
&variable-name VARIABLE USED ONCE.
&QUERY NOT TERMINATED.
&OPERATOR EXPECTED.
&OPERAND EXPECTED.
&= INCORRECTLY USED.
&= OPERATOR EXPECTED.
&INVALID RELATIONAL OPERATOR.
&VERB USE LIMIT IS 1.
&WORD MAX 30 CHARS.
&INVALID DEV:FILE.EXT.
&INVALID (= ZZITEM).
&NUM VAR NAMES BEGIN WITH X.
&UNDEFINED DATA NAME OR ELSE AN
 ALPHA VAR SPELLED INCORRECTLY.
&DICTIONARY NOT FOUND FOR FILE.
&OPEN STATEMENT-DUPLICATE NAMES.
&IF STATEMNT HAS NO VERB PHRASE.
&FILE NAME INCORRECTLY USED.
&NO CREATE ITEMS DESIGNATED OR
 POSSIBLY FILENAME NOT QUOTED.
&INCORRECTLY USED VOCAB WORD.
&CAN TAKE NO QUALIFICATION.
&LACK DETAIL INFO DM ITEM.
&VERB EXPECTS A DATA ITEM LIST.

IQL ERROR MESSAGES

IQL can display the following error messages after you enter   the   RUN
or EXECUTE assistance command:

>   ?ILLEGAL DATA BASE...SIXBIT CANNOT BE
>   MIXED WITH ASCII IN THIS VERSION OF IQL.
>   ?CANNOT FIND INPUT FILE file-name; ENDING RUN.

APPENDIX G

ASSOCIATED DOCUMENTS DESCRIPTIONS

ASSOCIATED DOCUMENTS

TOPS-10 Introduction to IQL
DEC-10-LIQLA-A-D

The TOPS-10 Introduction to IQL provides a general understanding of
the interactive query language software. The manual presents general
concepts and several examples in the personnel, marketing, and
accounting fields.

Getting Started With TOPS-10 Commands
DEC-10-OTSCA-A-D

Getting Started With TOPS-10 Commands is a simplified guide for the
inexperienced user of the TOPS-10 Operating System. The commands
described in this manual are a subset of the monitor commands that are
most useful to the beginning user.   The TOPS-10 Operating System
Commands Manual describes all of the monitor commands.

TOPS-10 Operating System Commands Manual
AA-0916C-TB

The TOPS-10 Operating System Commands Manual is a complete reference
manual describing the commands available in the TOPS-10 Operating
System.

Getting Started With TOPS-20
AA-4187C-TM

Getting Started With TOPS-20 is a simplified guide for the
inexperienced user of the TOPS-20 Operating System. The commands
described in this manual are a subset of the monitor commands that are
most useful to the beginning user. The TOPS-20 User's Guide describes
all of the monitor commands.

TOPS-20 User's Guide
AA-4179B-TM

The TOPS-20 User's Guide is a complete reference manual describing the
commands available in the TOPS-20 Operating System.

ASSOCIATED DOCUMENTS DESCRIPTIONS

TOPS-20 EDIT User's Guide
DEC-20-UEUGA-A-D

The TOPS-20 EDIT User's Guide is a simplified guide for the inexperienced user of the TOPS-20 EDIT text editor. The commands described in this manual are a subset of the editor commands that are most useful to the beginning user. The TOPS-20 EDIT Reference Manual describes all the editor commands.

TOPS-20 EDIT Reference Manual
AA-5415A-TM

The TOPS-20 EDIT Reference Manual is a complete reference manual describing the commands available in the TOPS-20 EDIT text editor.

TOPS-10 Data Base Management System Programmer's Procedures Manual
AA-0901C-TB

The TOPS-10 Data Base Management System Programmer's Procedures Manual contains a description of the TOPS-10 Data Base Management System from the point of view of the programmer who writes programs to access the data base.

TOPS-20 Data Base Management System Programmer's Procedures Manual
AA-4149B-TM

The TOPS-20 Data Base Management System Programmer's Procedures Manual contains a description of the TOPS-20 Data Base Management System from the point of view of the programmer who writes programs to access the data base.

TOPS-10 Beginner's Guide to Multiprogram Batch
DEC-10-OMPBA-C-D

The TOPS-10 Beginner's Guide to Multiprogram Batch is a simplified guide for the inexperienced user of the TOPS-10 Batch System. The commands described in this manual are a subset of the batch commands that are most useful to the beginning user. The TOPS-10 Multiprogram Batch Reference Manual describes all of the batch commands.

TOPS-10 Multiprogram Batch Reference Manual
DEC-10-OMBRA-A-D

The TOPS-10 Multiprogram Batch Reference Manual is a complete reference manual describing the commands available in the TOPS-10 Batch System.

TOPS-20 Getting Started With Batch
DEC-20-OBGSA-A-D

TOPS-20 Getting Started With Batch is a simplified manual for the inexperienced user of the TOPS-20 Batch System. The commands described in this manual are a subset of the batch commands that are most useful to the beginning user. The TOPS-20 BATCH Reference Manual describes all of the batch commands.

ASSOCIATED DOCUMENTS DESCRIPTIONS

TOPS-20 BATCH Reference Manual
DEC-20-OBRMA-A-D

The TOPS-20 BATCH Reference Manual is a complete reference manual
describing the commands available in the TOPS-20 Batch System.

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will
use comments submitted on this form at the company's
discretion. If you require a written reply and are
eligible to receive one under Software Performance
Report (SPR) service, submit your comments on an SPR
form.

Did you find this manual understandable, usable, and well-organized?
Please make suggestions for improvement.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Did you find errors in this manual? If so, specify the error and the
page number.

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

Please indicate the type of reader that you most nearly represent.

☐ Assembly language programmer
☐ Higher-level language programmer
☐ Occasional programmer (experienced)
☐ User with little programming experience
☐ Student programmer
☐ Other (please specify)_____

Name_____ Date_____

Organization_____Telephone_____

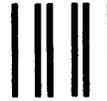Street_____

City_____ State_____ Zip Code_____
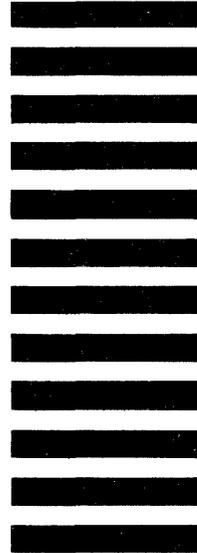   or
                                                  Country

Please cut along this line.

digital

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SOFTWARE PUBLICATIONS
200 FOREST STREET MR1-2/E37
MARLBOROUGH, MASSACHUSETTS 01752