MDBS QUERY RETRIEVAL SYSTEM MANUAL

MDBS QRS MANUAL

Version 3.08

Micro Data Base Systems, Inc.

P. O. Box 248

Lafayette, Indiana 47902

USA

Telex: 209147 ISE UR

(312) 303-6300 (in Illinois)

December 1985

Rev. 85A

3252335

NEW RELEASES, VERSIONS, AND A WARNING

Any programming endeavor of the magnitude of the MDBS software will necessarily continue to evolve over time. Realizing this, Micro Data Base Systems, Inc., vows to provide its users with updates to **this version** for a nominal handling fee.

New versions of MDBS software will be considered as separate products. However, bona fide owners of previous versions are generally entitled to a preferential rate structure.

Finally, each copy of our software is personalized to identify the licensee. There are several levels of this personalization, some of which involve encryption methods guaranteed to be combinatorially difficult to decipher. Our products have been produced with a very substantial investment of capital and labor, to say nothing of the years of prior involvement in the data base management area by our principals. Accordingly, we are seriously concerned about any unauthorized copying of our products and will take any and all available legal action against illegal copying or distribution of our products.

Table of Contents

MDBS QRS MANUAL

	Page
CHAPTER I. OVERVIEW	1
A. Introduction	1
B. Using QRS	2
C. Query Commands	2
D. Notational Conventions	4
E. Query Length	4
CHAPTER II. THE DISPLAY COMMAND	5
A. Syntax	5
B. Synopsis	5
C. Description	5
D. Examples	6
CHAPTER III. THE LIST COMMAND	11
A. Syntax	11
B. Synopsis	11
C. Description	11
D. Examples	20
CHAPTER IV. WRITE, SPEW, STATS, DBSTAT COMMANDS	27
A. Syntax	27
B. Synopsis	27
C. Description	27
D. Examples	29
CHAPTER V. COMPUTE COMMAND	31
A. Syntax	31
B. Synopsis	31
C. Description	31
D. Examples	31
CHAPTER VI. SET COMMAND	33
A. Syntax	33
B. Synopsis	33
C. Description	33
D. Examples	45
CHAPTER VII. DEFINE COMMAND	51
A. Syntax	51
B. Synopsis	51
C. Description	51
D. Examples	54
E. Advanced Usage of Macros with Parameters	58
CHAPTER VIII. READ COMMAND	59
A. Syntax	59
B. Synopsis	59
C. Description	59
D. Examples	59
CHAPTER IX. OPEN, CLOSE AND QUIT COMMANDS	61
A. Syntax	61
B. Synopsis	61
C. Description	61
CHAPTER X. QUERY NESTING	63
CHAPTER XI. STARTUP FILE	67
A. Overview	67
B. Creating the STARTUP file	67
CHAPTER XII. QRS ERROR MESSAGES	69

FIGURES

FIGURE II-1. Sample Schema 3

APPENDICES

APPENDIX A - List of QRS Keywords A1
APPENDIX B - Decimal-ASCII Conversion Table B1

I. OVERVIEW

A. Introduction

The Query Retrieval System is designed for use with MDBS III data bases. Called QRS, this optional MDBS III module furnishes a nonprocedural, English-like query language that allows a non-programmer to interrogate any MDBS data base on an ad hoc basis. Armed with the picture of a data base's logical structure, a non-technical (i.e., non-programming) user can quickly begin formulating powerful queries for data retrieval. In addition to the standard QRS commands, the user is allowed to customize QRS by defining new commands via a built-in macro facility. Numerous utility commands are available for tailoring the query environment to the user's needs.

The report generated in response to a query can be displayed in a standard format. Alternatively, the user can customize a report by invoking the QRS report writer features. One report writer feature allows a user to specify control breaks in a report, with options permitting statistics (such as mean, variance, etc.) to be printed at each break and at aggregate levels. Another allows labels to be specified for data item values. Value associations are automatically performed by QRS (e.g., the pre-assigned value "Hispanic" may be printed automatically when a value of "H" is encountered for the RACE data item). More elaborate reports can be generated with the optional RDL module of MDBS III.

At the user's option, a report generated by QRS can be displayed on the screen, routed to a printer, saved on a file for later use, or used as input to programs such as KnowledgeMan, Guru, and common spreadsheet packages. With QRS a user can devise arbitrarily complex expressions. Wildcard and "match-one" features are allowed for string comparisons. An optional conditional clause (the FOR clause) can be included for highly selective data retrieval. A path clause (the THRU clause) is used in each retrieval query to indicate which data relationships should be used to answer the query. For example, a query to list all females (and their phone numbers) under 35 years of age in Department 43 would be:

```
LIST NAME,PHONE FOR SEX="F",DEPT=43 AND AGE<35 THRU DEPTS,EMPLOYEES
```

Complex Boolean expressions are permitted in the conditional clause. A query can also contain arithmetic expressions, each involving one or more data items. Of course, all data extraction is contingent upon a user having read access to the data items and sets referenced in a query.

The user may choose to print out the total record accesses for a given query thus allowing him to "fine tune" a given query's efficiency by reordering relationship paths or including additional selection criteria.

QRS is valuable not only to non-programming users in the generation of ad hoc reports. QRS is also valuable to developers of application systems, in that it reduces the programming effort involved in the development of such systems.

B. Using QRS

Simple QRS installation instructions are included in the system specific manual. Once QRS has been installed, it can be executed from the operating system. The disk containing the main data base area must be on-line. QRS can be invoked with a -b argument on the command line to explicitly allocate the page buffer region. If this argument is not specified, approximately half of the available memory is automatically allocated. The remainder of available memory is reserved as a non-data base working space (e.g., file control blocks, stack, sort work areas). If it is used, this argument has the form -bnnnnn where nnnnn is the (decimal) number of bytes to be allocated for the page buffer region. This number should be at least as large as the "minimum DMS buffer region size" reported by the DDL Analyzer, otherwise DMS command status error 31 results. If too large a page buffer region is requested, an error message indicating "excessive memory request" is displayed. The remaining memory reserved for non-data base working space is normally sufficient. However, in rare cases this working space may be insufficient, resulting in various kinds of error conditions and situations where the data base may be left open. In such a case, -b can be used to allocate a smaller page buffer region to allow a larger non-data base working space.

When the QRS processor is executed (as described in the system specific manual), the QRS banner message appears on the console screen. This banner can be suppressed by including a -m argument on the command line used to invoke the query processor. QRS prompts the user for the file name of the main data base area, user name, and user password. If these prompts are not desired, QRS can be invoked with -d, -u and -p arguments. The -d argument must be immediately followed by the file name, the -u by the user name, and the -p by the password.

QRS checks to see whether the user name and password are valid for the indicated data base. If they are valid, then the --> prompt is issued. This prompt means that QRS is ready to accept a query command. A query command is terminated by pressing the carriage return key. After QRS displays its response to a query command, the --> prompt is again issued to show that QRS is ready to accept the next query.

A query session can be terminated by entering QUIT or BYE. This returns control to the operating system. Pressing the "soft interrupt" key (Escape key on most operating systems) interrupts the QRS activity and results in the --> prompt.

Chapter XI explains how to create and make use of a STARTUP file, which QRS can automatically utilize to identify the data base and user and to set various environment options and parameters.

C. Query Commands

The QRS language supports sixteen standard commands. In addition, a user is allowed to define still other commands. Each QRS command consists of a sequence of clauses. Some clauses are required within a command; some may be optional. If a user defines a command, then the user specifies the clauses that make up the command.

The 16 standard QRS commands have the following effects:

1) HELP : provides on-line help about the following commands
2) DISPLAY: display selected data dictionary contents
3) LIST : list selected data base contents in a report
4) WRITE : write selected data base contents to a disk file, without report headings
5) SPEW : write selected data to a disk file using a special format (e.g., DIF)
6) STATS : perform statistical analyses on numeric data retrieved from data base
7) DBSTAT : show various data base processing statistics
8) COMPUTE: compute a numeric value
9) SET : set environmental parameters, options, or column headings to control subsequent query processing
10) DEFINE : define value labels or new commands (macros)
11) DATFORM: choose an alternative form for presenting dates
12) READ : read and execute the queries on a disk file
13) ECHO : echo a message to the console screen
14) OPEN : open a data base area for QRS processing
15) CLOSE : close a data base area to QRS processing
16) QUIT : stop processing QRS commands

The clauses used for each of these commands are described in chapters that follow. The commands are presented in the order shown above.

Appendix A provides a list of the keywords used in QRS commands. It is advisable, when defining names (and synonyms) in the Data Description Language, to select names (synonyms) that are distinct from QRS keywords. A data item, record type, set or area that has the same name as a QRS keyword cannot be referenced in a QRS command. Synonyms, as defined with the DDL, can be used interchangeably with their corresponding data item, record type, area and set names.

Figure II-1 shows the sample logical structure presented in Chapter II of the MDBS DDL Manual (in Figures II-3 and III-1,2). One additional set (ISKL) has been added to this schema. In this manual, all example queries are based on the logical structure of Figure II-1.

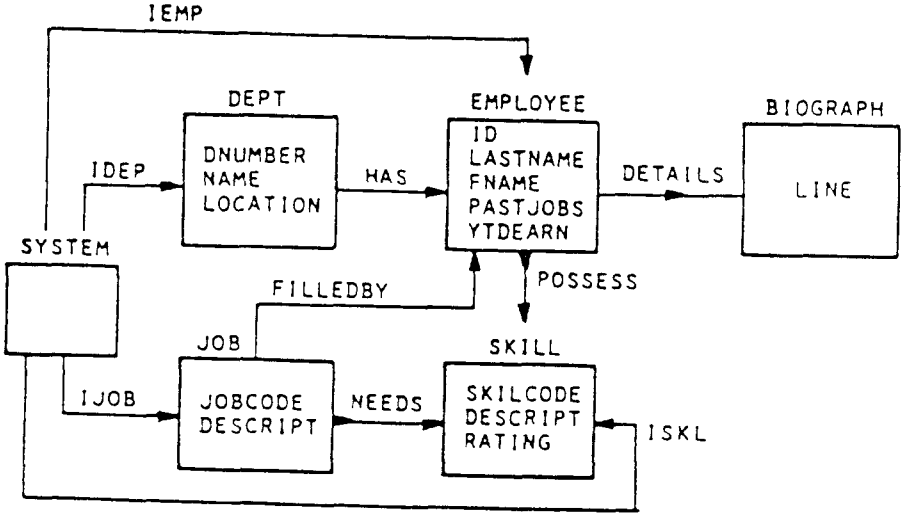

Figure II-1. Sample Schema

D. Notational Conventions

When describing the syntax of a QRS command, each clause is enclosed within the symbols < and >. These two symbols are not used to separate clauses within a query itself. In actually stating a query, two clauses in a command can be separated by either blanks or a comma.

For purposes of syntactic documentation, keywords and other literals are indicated with bold faced type. User-determined elements of a command are not in bold faced type. For example, the display command has the syntax:

<**display**><parameter><list>.

A display command must literally begin with the word **display**; the user determines the parameter and list that are to be used in a particular display command.

In practice, when accepting a command, QRS translates all non-quoted parts of a query to upper case. Thus when a user states a query, upper and lowercase can be intermixed as desired. In this manual, non-quoted portions of example queries are in upper case.

A vertical bar in the outside margin indicates that the feature being described is for experienced QRS users. This does not necessarily imply that the feature is difficult or complex to use. It does imply that the feature could be skipped on an initial pass through this manual, without a loss of continuity.

E. On-Line Help

QRS provides an on-line help facility. To get a display of helpful information about any of the QRS commands, type the word **help** in response to the --> prompt. The syntax of each command is displayed using notational conventions similar to those described above.

F. Query Length

The maximum length of a query is 255 characters, including blanks.

II. THE DISPLAY COMMAND

A. Syntax

<**display**><parameter><list>
or
<**disp**><parameter><list>

B. Synopsis

A DISPLAY command can be used to obtain information from the data dictionary about areas, record types, sets, or data items. It can also be used to acquire information about the current query environment as specified with the SET command (Chapter VI). This includes information about column headings, environment parameter values and environment option settings. A DISPLAY command cannot be used to obtain data dictionary information if the data base was initialized with the **o** option (see **MDBS DDL Manual**, Chapter VI).

C. Description

There are six permissible one-character parameters that can be used in a DISPLAY command:

<parameter>	information displayed about
a	all areas
c	all column headings
e	all environment parameter values
l	language used for sorting
o	all environment options settings
r	all record types
s	all sets

A list can optionally be specified with the **a**, **r** and **s** parameters in order to restrict the displayed information to a particular area, record type, or set (respectively). This results in more detailed information than what is obtained without specifying the list. Furthermore, an **i** parameter can be specified with the names of a record type and one of its data items to display detailed information about that data item.

<parameter>	<list>	function
a	area name	display information about the indicated area
r	record type name	display information about the indicated record type
s	set name	display information about the indicated set
i	record type name,data item name	display information about the indicated data item in the indicated record type

An area, record type, set or data item can be indicated by specifying either its name or a synonym for that name (as declared in the data base's DDL specification).

If both <parameter> and <list> are omitted from a DISPLAY command, the user is prompted with a list of the permissible parameters.

Virtually all schema information is available to a user through the DISPLAY command. The only kinds of data dictionary information that are not available with the DISPLAY command are user names, passwords, and user access codes. Read and write access codes for areas, data items, record types, or sets are not displayed. However, the user is informed whether he has read and/or write access to the area, data item, record type or set being displayed.

D. Examples

DISPLAY R	displays summary information for all record types
DISPLAY R SKILL	displays detailed information for the SKILL record type
DISP S IEMP	displays detailed information for the set IEMP
DISP A	displays summary information for all areas
DISP I DEPT, LOCATION	displays detailed information for the LOCATION data item within the DEPT record type
DISP O	displays all environment option settings
DISP C	displays user-defined column headings
DISP E	displays all environment parameter values
DISP L	displays the language used as a basis for all sorting that takes place

Sample query session using the DISPLAY command:

```
-->disp r

SYSTEM
     title:              (none)
     synonyms:           (none)
     owned by:           (none)
     owner of:           $SYSSET, IEMP, IDEP, IJOB
     items:              (none)

DEPT
     title:              (none)
     synonyms:           (none)
     owned by:           IDEP
     owner of:           HAS
     items:              DNUMBER, NAME, LOCATION

EMPLOYEE
     title:              (none)
     synonyms:           (none)
     owned by:           IEMP, FILLEDBY, HAS
     owner of:           POSSESS, DETAILS
     items:              ID, LASTNAME, FNAME, PASTJOB, YTDEARN

BIOGRAPH
     title:              (none)
     synonyms:           (none)
     owned by:           DETAILS
     owner of:           (none)
     items:              LINE

SKILL
     title:              (none)
     synonyms:           (none)
     owned by:           POSSESS, NEEDS
     owner of:           (none)
     items:              SKILCODE, DESCRIPT, RATING

JOB
     title:              (none)
     synonyms:           (none)
     owned by:           IJOB
     owner of:           FILLEDBY, NEEDS
     items:              JOBCODE, DESCRIPT
```



```
-->disp r skill

skill
     title:              (none)
     synonyms:           (none)
     owned by:           POSSESS, NEEDS
     owner of:           (none)
     items:              SKILCODE UNSIGN   2
                         DESCRIPT STR  55
                         RATING REAL   2


-->disp s iemp

iemp
     title:              (none)
     synonyms:           (none)
     type:               N:M
     owner insertion:    MANUAL
     owner order:        IMMAT
     owners:             SYSTEM
     member insertion:   AUTO
     member order:       SORTED
     members:            EMPLOYEE


-->disp a

JOBS
     title:              (none)
     synonyms:           (none)
     access:             READ, WRITE

JOB1
     title:              (none)
     synonyms:           (none)
     access:             READ, WRITE

JOB2
     title:              (none)
     synonyms:           (none)
     access:             READ


-->disp i dept location

dept.location
     title:              (none)
     synonyms:           (none)
     type:               STR  35
```

```
-->disp o
OPTION   STATUS            MEANING IF ON
   1      OFF       interactive I/O echo to printer
   2      ON        report output displayed on console
   3      ON        suppress printer output
   4      OFF       spool printer output to disk (FN)
   5      OFF       echo alternate input
   6      OFF       write output to console
   7      OFF       print output from WRITE
   8      OFF       echo one term per line
   9      OFF       suppress column headings
  10      OFF       suppress value labels
  11      OFF       suppress macro recognition
  12      OFF       echo macro expansion output
  13      ON        suppress printer form feed
  14      OFF       perform sort on query output
  15      OFF       pause after each report page
  16      OFF       ignore case in conditionals
  17      ON        sort in ascending (a-z) order
  20      OFF       print record access count
  21      ON        print number of observations
  22      ON        print maximum observation
  23      ON        print minimum observation
  24      ON        print sum of observations
  25      ON        print mean of observations
  26      ON        print variance of observations
  27      ON        print std dev of observations
  28      OFF       suppress statistics (opt21..27)


-->disp e
   CN     class negation       ^
   CD     console depth        24
   CP     console page eject   0, 0, 0, 0
   CW     console width        80
   M1     match one char       $
   MS     match string         *
   OC     open class           [
   CC     close class          ]
   SO     select open          {
   SC     select close         }
   CL     comment leader       ;
   OF     output format        89
   PC     printer close        0, 0, 0, 0
   PD     printer depth        60
   PM     printer margin       0
   PO     printer open         0, 0, 0, 0,
   PP     printer page eject   0
   PW     printer width        120
   CS     column spacing       2
   SF     scale factor         0
   TL     title
   FN     list file name
   PU     printer utility      lpr

-->disp L

Language is:  FRENCH
```


This page intentionally left blank.

III. THE LIST COMMAND

A. Syntax

<**list**><find clause><break clause><conditional clause><path clause>

B. Synopsis

The LIST command is one kind of QRS retrieval query. Other QRS commands that perform retrieval are presented in the next chapter. When stating a retrieval query, the QRS user utilizes the find clause to indicate those data items whose values are to be retrieved. The user also specifies which data interrelationships are to be used in retrieving the data. This is accomplished with the path clause. Since MDBS supports many interrelationships between any pair (or group) of data items, it is crucial for a QRS user to be able to specify which of those interrelationships is of interest for a given query. Using different interrelationships may very well yield different results. Suppose that a QRS user wants to obtain a list of all parts for a given supplier. The user needs a concise method for indicating which supplier-part relationship is of interest: all parts that can be supplied by the supplier, or all parts that have been received from the supplier, or all parts currently on back-order from that supplier.

The find and path clauses must appear in a LIST command. If either is omitted, the query is obviously ambiguous. On the other hand, the break and conditional clauses are optional. The break clause can be used to specify the occurrence of control breaks in the report being generated. The conditional clause can be used to place conditions on the retrieval that is to be performed, thereby providing a selective retrieval capability. Synonyms (as declared in the data base's DDL specifications) can be used in any of these clauses.

A QRS retrieval command effectively defines a virtual table (flat file or "relation"). The table specified by a command does not physically exist in a data base, but is automatically generated by QRS. A multitude of tables virtually exist in any MDBS data base and QRS provides the means for viewing all of them. A find clause defines the columns of a virtual table. Rows in that table are generated by QRS, subject to the conditional clause and using the path clause. The LIST command also allows extensive wildcard and match-one comparisons.

C. Description

Illustration

The following LIST command illustrates all clauses that can be used within a LIST command:

```
LIST NAME LASTNAME BY DNUMBER FOR DNUMBER IN [1 2 4] THRU IDEP,HAS
```

This command lists the department name and employee last names for the employees in departments 1, 2 or 4. The path used in answering the query is IDEP,HAS. As shown below, the report has control breaks on DNUMBER.

```
NAME          LASTNAME

R & D         Grant
R & D         Gagster
R & D         Madison
R & D         Fiskum
R & D         Russet

     no of observations:      5

production    Sellet
production    Goeld

     no of observations:      2


     no of observations:      7
```

The remainder of this description section examines each of the clauses individually. Several LIST command examples appear at the end of this chapter. The **select** keyword can be substituted for the **list** keyword if desired.

Find Clause

A find clause consists of one or more terms. A term is a data item or an arithmetic expression involving zero, one, or more data items. Within an arithmetic expression, the +, -, *, and / arithmetic operators are allowed. Traditional operator precedence conventions are observed: left-to-right evaluation with * and / being evaluated before + or -. Quite complex arithmetic expressions containing nested parentheses, integer or real constants, and data items from different record types are supported. When nested parentheses are used, the innermost is evaluated first. Blanks within an arithmetic expression are ignored.

Commas or blanks can be used to separate the terms in a find clause. The order in which terms are specified in a find clause determines the order of columns in the resultant report. If DNUMBER is first in a find clause, then values of DNUMBER form the first column of the report.

Each reference to a data item can be prefaced by the name of the record type that contains that data item (e.g., refer to NAME as DEPT.NAME). This feature is used to avoid the ambiguity that would otherwise arise if the schema has two data items with the same name (e.g., DESCRIPT in JOB and in SKILL). If a data item name is unique within the record types along the command's path, then the record type preface may be omitted. If it is not unique, then either the preface or a synonym must be used.

In addition to terms, character constants can also appear in a find clause. A character constant is a string of characters enclosed in double quotes. This literal string will appear in every row of the output table. A character constant consisting of a backslash (\) followed by a decimal number causes the ASCII equivalent of that number (see Appendix B) to be performed. For example, "\10" forces a line feed which is useful in printing mailing labels.

The number of terms (and constants) that can appear in a find clause is restricted by the width of the terms. A term's width is the maximum possible length of any of its values. For numeric terms, the width of the term is as specified by the "OF" environment parameter (described in Chapter VI). For a non-numeric data item, the width is the number of display characters that could be required for a value of that data item. If the sum of the widths of terms in a find clause exceeds the printer width (or the console width in the case where a printer is not used) then an error message "Output length exceeded" is issued by QRS. The maximum number of terms (and constants) permitted regardless of width is 25. The maximum number of items referenced within a query is 20.

Some find clause examples are:

```
JOBCODE
DNUMBER,NAME
JOBCODE,SKILCODE,RATING
ID,DNUMBER+(YTDEARN-2)/1000,YTDEARN,"\10",LOCATION
DNUMBER,"* ",ID,JOBCODE
LOCATION SKILCODE
JOBCODE,JOB.DESCRIPT,SKILL.DESCRIPT
```

Conditional Clause

The optional conditional clause begins with the keyword: **for.** A row in the virtual table defined by the find clause will appear in the output report only if it satisfies the conditions stated in the conditional clause. For instance, the query

```
LIST ID, DNUMBER FOR LASTNAME="Lehr"
```

will list all employee identifiers and their respective department numbers, but only for those employees with the last name of Lehr. In this example, the conditional expression is: FOR LASTNAME="Lehr".

In general, a conditional clause is a Boolean expression formed from one or more relational expressions. A relational expression consists of:

<term> <relational operator> <term>.

The full host of relational operators is supported: EQ(=), NE(<>), LT(<), GT(>), LE(<=), GE(>=). Under QRS, these operators have their traditional meanings. Numeric terms cannot be compared with character terms. Terms involving string, binary, character, date and time data item types are referred to as character terms. The group inclusion operator, IN, is also supported. The IN operator is satisfied if the value of the left term is in the group of values specified by the right term.

A term in the conditional clause has the same meaning as it does for the find clause. However, in a conditional clause, a term may also be a character constant (e.g., "Smith"), a group of character constants (e.g., [Smith,Jones,Lehr]), or a group of numeric constants (e.g., [5,7,9]). The character constants or numeric constants within a group can be separated by either blanks or commas. A group begins and terminates with left and right square brackets, respectively. Whereas a stand-alone character constant must be quoted, character constants within a group do not need to be quoted (unless they contain wildcard symbols, match-one symbols, or character classes, all of which are described below).

Some examples of relational expressions are:

```
LASTNAME="Smith"
DNUMBER >= 17
YTDEARN<(5+DNUMBER)*1000
LASTNAME IN [Smith,Jones Lehr]
YTDEARN/1000 GE 13.2
```

In stating a relational expression, a blank must precede and follow the relational operator. Exceptions are the =, <>, <, >, <=, >= operators for which blanks are not needed.

Wildcard and match-one symbols can be used within character constants if the relational operator is IN, =, EQ, <>, or NE.

The wildcard symbol matches any combination of zero or more characters. The default wildcard symbol is *. As an example, if we are interested only in the employees whose first names begin with D, our conditional clause will contain the relational expression: FNAME="D*". If we are interested in employees whose biography mentions the term computer, then our conditional clause will contain the relational expression: LINE="*computer*".

The match-one symbol matches any one character. For instance, if we wish to find all the employee identifiers that have 66666666 as their last eight digits, the conditional clause will contain the relational expression: ID="$66666666". The default match-one symbol is $. To find all identifiers that have 6666 as the first four digits, the conditional clause will contain the relational expression ID="6666$$$$$".

Character classes can be used within character constants if the relational operator is IN, =, EQ, <>, or NE. Unlike the match-one feature which matches any one character, a character class allows the user to specify a match with any one of a particular class of characters. A character class is denoted by [list], where [is the default open class character, and] is the default close class character. A list may consist of a simple list of characters, such as the vowels [aeiouAEIOU], or a range of characters, such as the lower-case alphabet [a-z], or a combination of simple lists and ranges. As one example, to find employees whose last names begin with A through M, the conditional clause would contain the relational expression: LASTNAME="[A-M]*". As a second example, to find all

employees whose first name is pronounced like Karen, the conditional clause would contain the relational expression: FNAME="*Kar[aeiy]n". As a third example, to find all departments on the north, odd-numbered side of Kossuth Street, the conditional clause would contain the relational expression: LOCATION="*[13579] Kossuth Street*".

If the first character of a list is ^ then the list is for a negated character class. This means that a match is made for any character that is not in the character class. For instance, to find all employees whose last name does not begin with a capital letter, the conditional clause would contain the relational expression: LASTNAME="[^A-Z]*". This is particularly valuable for rapidly detecting faulty data entry.

Note here that the character - can be used in a character class to look for a -, if it is the first character of a list (or first after the negation) or the last character of the list. A ^ can be searched for as a ^ when it is not the first character of a list. An example of this is to find all department addresses containing a ^ or . or -. The conditional clause for this would contain the relational expression: SADD="*[.^-]*". Note also that within a character class, the wildcard and match-one characters lose their specialness, so they may be searched for as $ and *. Nesting of character classes is not allowed.

A character class is different from a group as discussed previously. A group refers to a collection of constants to be matched to terms. A character class contains a list of single characters for which a match is accepted. Note that a character class may occur within a group. For example:

LASTNAME IN [Schultz,"Sm[i,y]th"]

Default symbols can be changed. The match-one, wildcard, open class, close class, and negate class default symbols can all be changed by means of the SET command described in Chapter VI. This allows users whose character sets do not contain the default characters to still use these facilities.

Relational expressions can be combined into a single CONDITIONAL clause using any of the following Boolean operators: AND (^), OR, XOR (exclusive OR), NOT (~). If a comma or blank separates two relational expressions the AND operator is assumed. Parentheses (possibly nested) can be used to govern the precedence used in evaluating the relational expressions. Expressions within the innermost parentheses pairs are evaluated first. The precedence conventions used in evaluating arithmetic, relational and Boolean operators within a conditional clause are as follows:

1) Parentheses (and)
2) NOT, unary negative sign, IN
3) *, /
4) +,-
5) EQ, =, NE, <>, LT, <, LE, <=, GT, >, GE, >=
6) AND,
7) OR, XOR

Some conditional clause examples are:

```
FOR DNUMBER IN [1,5,14,32] AND LOCATION="*Chicago*"

FOR LINE="*data base*", JOB.DESCRIPT NE "*data base*",
    SKILL.DESCRIPT NE "*data base*"

FOR (YTDEARN/1000>18.9 OR JOBCODE <32) AND NAME="*data*"

FOR JOBCODE=49 XOR SKILCODE=14

FOR FNAME IN ["K*",Deb,"M[o,a]rk", "[T-W]*"] AND LASTNAME="T$$$"
```

Path Clause

A path clause appears in every LIST command. It can begin with either the **thru** or **from** keyword. As explained previously, the path clause allows a QRS user to specify the data interrelationships that should exist among entries in the desired report. Two queries that differ only in their path clauses will typically yield different reports. To use the path clause, examine all data items specified in the find and conditional clauses of a given query. In the schema, place a check mark on every record type that contains at least one of these data items. In order to answer the query, QRS will have to examine occurrences of each of the checked record types. In the path clause we need to specify a path of sets that will allow QRS to reach each of the checked record types. QRS automatically accesses only those records along the path that are needed in order to produce the desired report. QRS can move both upstream (member to owner) and downstream (owner to member) through a set. If the path requires an upstream movement for some set, then that set's name is prefaced with the > symbol.

A path will always

i) begin with a SYSTEM-owned set
ii) connect all checked record types
iii) be completely connected

For example,

IDEP,HAS,POSSESS is a completely connected path of sets

IDEP,HAS,>FILLEDBY,NEEDS is completely connected

IDEP,POSSESS is not completely connected (it skips HAS)

IJOB,>POSSESS,>HAS is not completely connected (it skips NEEDS)

IEMP,DETAILS,POSSESS is completely connected

IEMP,>HAS,>FILLEDBY,POSSESS is completely connected

A path is not allowed to contain loops. This means that in a legitimate path of sets, no record type may be entered more than once. For instance, the sets

IJOB,NEEDS,>POSSESS,>FILLEDBY

do not form a legitimate path. They form a loop, since the record type JOB is entered through IJOB and >FILLEDBY. Notice that IJOB,NEEDS,>POSSESS,FILLEDBY also forms a loop rather than a path since the EMPLOYEE record type is entered through both FILLEDBY and >POSSESS. The sets

IJOB,NEEDS,>POSSESS,>HAS,DETAILS

form a legitimate path; no record type is entered more than once.

The order in which sets are specified in a path is important. It indicates to QRS the way in which the path is to be traversed. Values of data items at the outer reaches of the last set are the last to be examined or evaluated. In contrast, occurrences of the record type owned by the first set in a path are examined or evaluated earliest. When moving downstream through a set, the member record type is at the set's outer reach; when moving upstream, the owner record type is at the set's outer reach.

Consider the path: IDEP,HAS,POSSESS. The QRS first finds a DEPT occurrence (outer reaches of IDEP) whose data values satisfy the query. It next finds an EMPLOYEE occurrence for that department (outer reach of DEPT) whose data values satisfy the query. If there are none, QRS finds the next satisfying DEPT occurrence. If a satisfying EMPLOYEE occurrence was found, QRS examines each SKILL occurrence that it owns. For each of these SKILL occurrences (outer reach of POSSESS) that satisfies the query, a report line is printed. The next satisfying employee is then found and its skill occurrences are evaluated, and so forth until all satisfying employees for the current satisfying department have been examined. The next department occurrence that satisfies the query is found, its employees and their skills are examined, etc. Thus the ordering of entries in the report is determined by the member orders for the IDEP, HAS, and POSSESS sets respectively.

For a given find and conditional clause, the path selected may affect not only the content of the resulting report, but also the speed with which it is generated. As a general rule, one should attempt to place those records on which conditionals are based as early in the path as possible. Suppose we need a list of the IDs of all employees in departments 1, 5 and 9. The appropriate find and conditional clauses are:

```
LIST ID FOR DNUMBER IN [1,5,9]
```

There are many possible paths that could be specified with this query. The IEMP, >HAS path would generate the appropriate list of IDs, ordered according to the member order of IEMP. The IDEP, HAS path would yield the same list of IDs, but in a different order. They would be ordered primarily on the basis of the member order of IDEP and secondarily on the basis of the HAS member order. The first path causes QRS to search all employees and determine whether each is in a desired department. The second path causes QRS to find only the desired departments (1,5,9) and to list the employees of these. Clearly, the second path will allow QRS to generate the report more rapidly than the first path.

In some cases it may be feasible to place additional conditions within the conditional clause in order to reduce the size of the retrieval tree. For example, suppose we desire to list those people who have earned over $10,000. Suppose that we know that only those employees in departments 2, 3, or 7 could possibly have year-to-date earnings that exceed $10,000. We might then state the query

```
LIST LASTNAME FOR YTDEARN > 10000 AND DNUMBER IN [2,3,7] THRU IDEP,HAS
```

rather than

```
LIST LASTNAME FOR YTDEARN > 10000 THRU IEMP
```

in order to significantly reduce retrieval time.

The user is further aided in "fine tuning" retrieval by environment option 20 (Chapter VI), which causes a count of the record retrievals for each query to be displayed. The best formulation for frequently run queries can be determined with this option. The resulting query might then be preserved as a macro (Chapter VII) for recall at future times.

QRS allows the user to traverse a set in reverse (last to first) order as well as in the normal (first to last) order. To do this, a negative sign (-) is placed before those sets for which reverse traversal is desired. Suppose, for instance, that we need a list of employees in reverse alphabetic order. The path clause would be: THRU -IEMP. Traversal through the sorted IEMP set would normally list the names in alphabetic order. Using this facility, one can also extract information from LIFO sets as FIFO (and vice versa).

Break Clause

A control break can be declared for any data item. This means that whenever the value of that data item changes during the query evaluation a control break occurs. The precise effect of a control break depends upon which of the environmental options (21 - 27) are on or off. The default is to have all of these options on. Chapter VI explains how to easily turn any of these parameters on or off.

If environment options 21-27 are on, then full statistics on every numeric-valued term in the find clause are included in the report at each control break point. Reporting of a statistic can be suppressed by turning its option off. QRS generates two sets of statistics at each control break point, one for all the observations since the last control break point, and one for all the observations since the beginning of the report.

A query may contain more than one control break data item. Data items used for control breaks are specified after a find clause and are prefaced with the word **by**. For instance,

```
LIST ID,YTDEARN BY DNUMBER THRU IDEP,HAS
```

specifies a control break by department number. LIST operates just as described previously (giving IDs and year-to-date earnings) except that each time QRS detects a different department number, statistics are generated for YTDEARN across all employees of that department.

By setting environment options and parameters correctly, a printer form feed or console page clear will occur after each break's statistics. For automatic printer form feeds, the PP parameter should be set correctly and options 3 and 13 should be off. For automatic console clears, the CP parameter should be set correctly and option 2 should be on.

Report Features and Options

Page titles are printed out at the top of each page generated by the LIST command. The title printed is as specified by the TL environment parameter (Chapter VI). Titles are automatically centered at the top of pages. This positioning can be adjusted right or left by padding the title with leading or trailing blanks.

Column headings appear on the output table generated by a LIST command. There is one heading for each term specified in the find clause. The heading produced for a term depends on the composition of that term. If a term consists of a single data item, then the name of that item is used as the corresponding column heading. If the nth term is an arithmetic expression, then the default heading "TERM n" is produced. If a heading is longer than the particular item values being printed in its column, the heading is truncated to the item length. The QRS SET command (Chapter VI) can be used to specify alternative column headings.

The order of rows in a report produced by the LIST command is consistent with the ordering conventions of the sets appearing in the path clause. If a different order is desired, the user can turn on environmental option 14 with the SET command (Chapter VI). This will cause QRS to perform a line sort on the output before display or printing. Of course, performing this internal sort incurs additional processing overhead.

A statistical analysis of any numeric terms in the find clause is automatically produced at the end of the report. These analyses can be suppressed by turning off options 21-27 with the SET command as described in Chapter VI.

Repeating data items are allowed in find and conditional clauses. If a repeating data item is in a find clause, each of its values (that satisfies the conditional clause) is retrieved. When a repeating data item appears in a conditional clause, the condition is evaluated for each individual value of the repeating data item.

D. Examples

1. For each employee, produce a report (i.e., virtual table) listing the employee's ID, year-to-date earnings, and job description.

```
LIST ID,YTDEARN,JOB.DESCRIPT THRU IEMP,>FILLEDBY
```

2. Produce a report listing the ID, name and past jobs of each employee in departments 1, 3 and 12.

```
LIST ID,FNAME,LASTNAME,PASTJOBS FOR DNUMBER IN [1,3,12]
     THRU IDEP,HAS
```

3. Produce a report of job codes and the employees that fill each job code for those employees whose year-to-date earnings are less than 10000.

```
LIST ID,LASTNAME,FNAME FOR YTDEARN<10000 THRU IJOB,FILLEDBY
```

4. For each employee having a last name beginning with A-M, produce a report listing that employee's job and all of the skills needed by that job.

```
LIST   ID,JOBCODE,SKILCODE   FOR   LASTNAME="[A-M]*"   THRU
     IEMP,>FILLEDBY,NEEDS
```

5. For each employee having a last name beginning with A-M, produce a report listing that employee's skills and the jobs that need those skills.

```
LIST   ID,JOBCODE,SKILCODE   FOR   LASTNAME="[A-M]*"   THRU
     IEMP,POSSESS,>NEEDS
```


6. For each employee having a last name beginning with A-M, produce a report listing that employee's job and that employee's skills.

```
LIST ID,JOBCODE,SKILCODE FOR LASTNAME="[A-M]*" THRU
     IEMP,>FILLEDBY,POSSESS
```

7. Produce a report listing the skill code and description for all jobs whose descriptions begin with the term "data base".

```
LIST SKILCODE,SKILL.DESCRIPT FOR JOB.DESCRIPT="data base"
     THRU IJOB,NEEDS
```

8. Produce a report listing the ID and department number of each employee whose biography contains the terms "data base" and "systems analysis".

```
LIST ID,DNUMBER FOR LINE="*systems analysis*data base*" OR
     LINE="*data base*systems analysis*" THRU IEMP,DETAILS,HAS
```

9. For each employee in department 5, list the ID and past jobs of that employee.

```
LIST ID PASTJOBS FOR DNUMBER=5 THRU IDEP,HAS
```

10. List the ID and year-to-date earnings of every employee who has the word "student" appearing in a past job description.

```
LIST ID,YTDEARN FOR PASTJOBS="*student*" THRU IEMP
```


Sample query session using the LIST command:

```
-->list lastname name descript thru idep has >filledby
LASTNAME              NAME          DESCRIPT

Grant                 R & D         vice president
Gagster               R & D         vice president
Madison               R & D         programmer
Fiskum                R & D         programmer
Russet                R & D         group manager
Sellet                production    manager
Goeld                 production    group manager
Dayton                sales         salesman
Milwright             sales         salesman
More                  sales         group manager
Ramsey                cust support  group manager

     no of observations:      11

-->list name lastname descript for dnumber=1 thru idep has >filledby
NAME          LASTNAME              DESCRIPT

R and D       Duranger              programmer
R and D       Ferston               programmer
R and D       Fiskum                programmer
R and D       Laisler               programmer
R and D       Hanson                programmer
R and D       Madison               programmer
R and D       Koelder               manager
R and D       Stokes                manager
R and D       Russet                group manager
R and D       Gagster               vice president

   no of observations:    10

-->list name lastname descript for dnumber = 1 and descript = "pr
ogrammer" thru idep has >filledby
NAME          LASTNAME              DESCRIPT

R and D       Duranger              programmer
R and D       Ferston               programmer
R and D       Fiskum                programmer
R and D       Laisler               programmer
R and D       Hanson                programmer
R and D       Madison               programmer

   no of observations:     6
```

```
-->list name, descript, id, lastname, for lastname = "[A-G]*" thr
u idep has >filledby
NAME          DESCRIPT                  ID         LASTNAME

R and D       programmer                800000000  Duranger
R and D       programmer                700000000  Ferston
R and D       programmer                600000000  Fiskum
R and D       vice president            222222222  Gagster
production    clerical staff            111000000  Court
production    group manager             555555555  Goeld
sales         clerical staff            222000000  Crooner
sales         salesman                  660000000  Durman
sales         salesman                  550000000  Boumer
sales         salesman                  440000000  Gilvaston
sales         salesman                  110000000  Dayton
sales         vice president            333333333  Grant
clerical      clerical staff            222200000  Burley
tech support  clerical staff            555000000  Dahl

   no of observations:    14

-->list name id for id = "$$$2*" thru iemp >has
NAME          ID

clerical      222200000
R and D       222222222

   no of observations:     2

-->list name id for id = "$$$3*" thru iemp >has
NAME          ID

sales         333333333

   no of observations:     1

-->list lastname id pastjobs for pastjobs = "*[Ss]tudent*" thru
iemp
LASTNAME            ID             PASTJOBS

Burley              222200000      Student, Ball State Univ
Duranger            800000000      Student, Purdue
Durman              660000000      Student, ISU
Gagster             222222222      Ph D student
Hanson              400000000      Student, Cornell
Hanson              400000000      Student, Taiwan Univ
Laisler             500000000      Student, Harvard
Ramsey              345665825      Student, Univ of Illinois
Russet              220487197      Student, Purdue Univ
Woods               880000000      Student

   no of observations:    10
```

```
-->list lastname id, pastjobs for dnumber = 1 thru idep has
LASTNAME          ID              PASTJOBS

Duranger          800000000       Micro Data Base Systems
Duranger          800000000       Student, Purdue
Duranger          800000000
Ferston           700000000       Staff, Purdue University
Ferston           700000000       General Telephone
Ferston           700000000       A & P food stores
Fiskum            600000000       Northrup Defense Systems
Fiskum            600000000       Hughes Aircraft
Fiskum            600000000       Boeing Corp
Laisler           500000000       Mellon Bank
Laisler           500000000       Student, Harvard
Laisler           500000000       Analyst
Hanson            400000000       Student, Cornell
Hanson            400000000       Student, Taiwan Univ
Hanson            400000000
Madison           300000000       Digital Technology
Madison           300000000
Madison           300000000
Koelder           100000000       Data General
Koelder           100000000       U. S. Navy
Koelder           100000000
Stokes            999999999       WBAA Radio Station
Stokes            999999999
Stokes            999999999
Russet            220487197       I.B.M
Russet            220487197       Student, Purdue Univ
Russet            220487197       General Telephone
Gagster           222222222       Professor, Purdue Univ
Gagster           222222222       Ph D student
Gagster           222222222       Programmer

   no of observations:    30
```



```
-->list lastname id pastjobs for pastjobs ne "*" and dnumber = 1
thru idep has
```

LASTNAME	ID	PASTJOBS
Duranger	800000000	Micro Data Base Systems
Duranger	800000000	Student, Purdue
Ferston	700000000	Staff, Purdue University
Ferston	700000000	General Telephone
Ferston	700000000	A & P food stores
Fiskum	600000000	Northrup Defense Systems
Fiskum	600000000	Hughes Aircraft
Fiskum	600000000	Boeing Corp
Laisler	500000000	Mellon Bank
Laisler	500000000	Student, Harvard
Laisler	500000000	Analyst
Hanson	400000000	Student, Cornell
Hanson	400000000	Student, Taiwan Univ
Madison	300000000	Digital Technology
Koelder	100000000	Data General
Koelder	100000000	U. S. Navy
Stokes	999999999	WBAA Radio Station
Russet	220487197	I.B.M
Russet	220487197	Student, Purdue Univ
Russet	220487197	General Telephone
Gagster	222222222	Professor, Purdue Univ
Gagster	222222222	Ph D student
Gagster	222222222	Programmer

no of observations: 23


```
-->list name lastname by dnumber thru idep has
NAME           LASTNAME

R and D        Duranger
R and D        Ferston
R and D        Fiskum
R and D        Laisler
R and D        Hanson
R and D        Madison
R and D        Koelder
R and D        Stokes
R and D        Russet
R and D        Gagster

   no of observations:    10

production     Court
production     Sellet
production     Goeld

   no of observations:     3

sales          Crooner
sales          Durman
sales          Boumer
sales          Gilvaston
sales          Lowder
sales          Milwright
sales          Dayton
sales          More
sales          Grant

   no of observations:     9

accounting     Woods
accounting     Lochman
accounting     Handee

   no of observations:     3

clerical       Burley
clerical       Hollinger
clerical       Norlady

   no of observations:     3

tech support   Schutz
tech support   Dahl
tech support   Whistler
tech support   Ramsey
tech support   Holst

   no of observations:     5

   no of observations:    33
```

IV. WRITE, SPEW, STATS, DBSTAT COMMANDS

A. Syntax

```
<write>  <find clause><conditional clause><path clause>
<spew>   <find clause><conditional clause><path clause>
<stats>  <find clause><conditional clause><path clause>
<dbstat>
```

B. Synopsis

The WRITE, SPEW (**special write**), and STATS (**statistics**) commands are constructed in the same way as the LIST command. They retrieve the same data as a LIST command. However, they handle the retrieved data differently than the LIST command. The WRITE command writes the retrieved data on a disk file without any titles, headings, or statistics. This file can then be used as input to various programs. The special write command (i.e., the SPEW command) has the same effect as the WRITE command, except that the retrieved data is written on the disk file in a special format that allows it to be directly input to popular graphics or spread-sheet software packages available within the host operating system. The STATS command generates statistics for the retrieved data, on a term-by-term basis. Individual data values that are retrieved in order to generate the statistics are not displayed. None of the three commands uses a break clause.

The DBSTAT command is used to obtain data base processing statistics pertaining to data retrievals performed by QRS.

C. Description

1. WRITE

A WRITE command is formulated in the same way as a LIST command except that **write** is substituted for **list** and no break clause is used. A write command retrieves the same data as a LIST command. The retrieved data is written onto the disk file indicated by the FN environment parameter. The SET command is used to assign a disk file name to the FN parameter (Chapter VI).

The format of a file generated by a WRITE command is governed by the sequence of terms in the find clause and the types of those data items. Term values are written out one per line. If the virtual table of retrieved data has n columns, then the first n lines of the file are formed from the first row of the virtual table. The next n lines consist of the data values in the second row of the virtual table, and so forth. Each data value is written on the disk file exactly as it would appear if displayed through a LIST command.

A file produced by the WRITE command can be incorporated into a KnowledgeMan or Guru table without any intermediate processing. This incorporation is accomplished with the

KnowledgeMan or Guru ATTACH command. Once the data is extracted from the central MDBS data base and incorporated into a table, it can be processed with the various KnowledgeMan or Guru facilities (e.g., third generation spreadsheet, relational data management, screen I/O manager, report forms manager, etc.). The file that results from a WRITE command can also be utilized as input to other programs, (e.g., a DML program that loads the file's data into another data base).

Note that a tabular file may be created according to the usual printer form by turning on environment option 4 (see Chapter VI).

2. SPEW

A SPEW command is formulated in the same way as a LIST command except that **spew** is substituted for **list** and no break clause is used. A SPEW command retrieves the same data as a LIST command. The retrieved data is written onto the disk file indicated by the FN environment parameter. The set command is used to assign a disk file name to the FN parameter (Chapter VI).

The format of a file generated by the SPEW command is operating system dependent and is described in the various MDBS System Specific Manuals. Generally, the formatting is consistent with the file format (e.g., DIF) used by spreadsheet and/or graphics packages available within an operating system. This command is unavailable in operating systems that do not support such packages.

Suppose, for example, that year-to-date earnings represents sales commissions for employees in department 5 and that we desire to build a spread-sheet that will use present commissions to project future commissions. Then the appropriate SPEW command is

```
SPEW LASTNAME,YTDEARN FOR DNUMBER=5 THRU IDEP,HAS
```

3. STATS

A STATS command is formulated in the same way as a LIST command, except that **stats** is substituted for **list** and no break clause is used. A STATS command retrieves the same data as a LIST command. The retrieved data is not displayed; only statistics on the retrieved data are displayed. For each numeric term, the list of values that satisfies the find and conditional clauses is subjected to standard statistical analyses. The minimum, maximum, average, standard deviation, variance, sum, and number of observations for each term's retrieved values are computed and displayed. For data items having non-numeric values, only the number of observations is computed and displayed. Display of any of the preceding statistics can be suppressed by turning off the appropriate environment options (21-27) with the set command (Chapter VI).

For instance, to obtain statistical analyses of the year-to-date earnings of all employees working for a department in Indianapolis, the appropriate query is:

```
STATS YTDEARN FOR LOCATION="*Indianapolis*" THRU IDEP,HAS
```

4. DBSTAT

The DBSTAT command has the same effect as it does when invoked from within a DML application program (Chapter XI of the DMS Manual). DBSTAT shows the following statistics:

a) the number of page buffers currently allocated in main memory,

b) the number of times (since opening the data base) that the most recent page access was to a different page than the last page access,

c) the number of read requests issued by the data base control system since opening the data base,

d) the number of write requests issued since the data base was opened that were due to DBSAVE or to background processing in a multiuser environment,

e) the total number of write requests issued by the data base control system since opening the data base.

This command is invoked by simply typing:

```
DBSTAT
```

D. Examples

```
WRITE LASTNAME,FNAME,ID,YTDEARN THRU IEMP

WRITE DNUMBER,JOBCODE FOR DNUMBER<13 THRU IDEP,HAS,>FILLEDBY

SPEW LASTNAME,YTDEARN THRU IEMP

SPEW LASTNAME,YTDEARN FOR LOCATION="*Indianapolis*" THRU IDEP,HAS

STATS YTDEARN FOR JOBCODE=5 THRU IJOB,FILLEDBY

STATS YTDEARN FOR LINE="*data base*" AND DNUMBER=7 THRU IDEP,
     HAS,DETAILS

DBSTAT
```


Sample query session using the STATS command:

```
-->stats ytdearn thru iemp

YTDEARN:
   no of observations:   11
  20410.00 sum
   4000.00 max
     10.00 min
   1855.45 ave
1579027.27 var
   1256.59 std

-->stats ytdearn for name = "sales" thru idep has

YTDEARN:
   no of observations:     9
    22800.00 sum
     4000.00 max
      100.00 min
     2533.33 ave
  1612500.00 var
     1269.84 std

-->stats ytdearn/1000 for name = "R and D" and descript = "progra
mmer" thru idep has >filledby

Term 1:
   no of observations:     6
       17.00 sum
        4.40 max
        0.50 min
        2.83 ave
        2.95 var
        1.72 std

-->
```

V. COMPUTE COMMAND

A. Syntax

<compute>

B. Synopsis

The compute command is used to evaluate arithmetic expressions that do not involve data items. The result of a computation can be used in the next computation or in subsequent LIST, WRITE, SPEW, and STATS commands. Simply put, the compute command provides the user with an interactive calculator.

C. Description

To enter the compute mode, simply type: **compute.** The system then prompts (with C:) for an arithmetic expression to be evaluated. The evaluation result is returned to the console and the user is prompted for another arithmetic expression. To exit the compute mode, type Q. QRS responds with the --> prompt.

If the user desires to use the result of the preceding computation in the next computation, then the # symbol is used (e.g., #+5 will add 5 to the result of the preceding computation). The # symbol can similarly be used in the find and/or conditional clauses of subsequent LIST, WRITE, SPEW, and STATS commands. Computations are accurate to at least 7 significant digits. More precision is available under some operating systems.

D. Examples

```
--> COMPUTE

C:  7/5
    1.4000000
C:  #+0.6
    2.0000000
C:  Q

--> LIST ID(YTDEARN+#)/1000 FOR YTDEARN/1000>6.2*# THRU IEMP
                        ...report...

--> STATS YTDEARN THRU IEMP
    YTDEARN:  15780.00  AVE
               3010.76  STD
```

```
--> COMPUTE

C:  2*3010.76+15780
    21801.5200000
C:  Q

--> LIST NAME FOR YTDEARN>#  THRU IEMP
                      ...report...
```


VI. SET COMMAND

A. Syntax

<**set**><parameter><value>
or
<**set**><**opt**><number><status>
or
<**set**><heading list>

B. Synopsis

The SET command is used to specify the environment within which a query command will be executed. The environment is described in terms of environment parameters and environment options. Environment parameters can be set to have various numeric or character values. Environment options have a status of either **on** or **off**.

The SET command can also be used to replace column headings that would normally appear for a LIST command, with alternative headings. SET commands can be used any time during a query session to alter the environment. The values of individual parameters can be altered whenever desired with the SET command. The status of an option can be changed whenever desired with the SET command. Column headings can be revised whenever desired.

C. Description

Setting Environment Parameters

Summary

The following is a summary of the available environment parameters and their default values. A full description of each is provided after this summary.

parameter	description	default value
CC	close character class symbol	]
CD	console depth	24
CL	comment lead symbol	;
CN	character class negation symbol	^
CP	console page eject	0, 0, 0, 0
CS	column spacing	2
CW	console width	80
FN	file name	none
M1	match-one character	$
MS	match string (wildcard) symbol	*
OC	open character class symbol	[
OF	output format	89
PC	printer close	0, 0, 0, 0

parameter	description	default value
PD	printer depth	60
PM	printer margin	0
PO	printer open	0, 0, 0, 0
PP	printer page eject	0
PU	printer utility	various
PW	printer width	120
SC	nested select close symbol	}
SF	scale factor	0
SO	nested select open symbol	{
TL	title	none

To set any of these parameters to have a value other than the default value, the syntax is

<**set**><parameter><value>.

When setting the FN, M1, MS, and TL parameters, the value must be enclosed in quotes. When setting the CP, PC and PO parameters, a sequence of one, two, three, or four values can be specified. These values are separated by commas. The values specified for the CD, CP, CW, OF, PC, PD, PM, PO, PP, PW, SP and SF parameters must be integers.

The CC, CN, M1, MS and OC parameters allow all QRS features to be used with widely diverse terminal character sets. The absence of a default character from a terminal character set does not preclude a user from utilizing the corresponding QRS feature. The SET command can be used to specify values other than the default values for these parameters.

Environment Parameter Details

CC — **C**lose Character **C**lass Symbol — **CC**

The value of the CC parameter indicates the close class symbol for a character class. For example, to change this symbol from its] default to), the following SET command is used:

```
SET CC )
```

CD — Console Depth — **CD**

The value of the CD parameter indicates the depth, in lines, of a console page. For example, to change the console depth from its present value to, say, 26 lines the following SET command is used:

```
SET CD 26
```

The default value for CD is 24.

CL Comment Lead Symbol **CL**

The value of the CL parameter indicates the comment lead symbol. Whenever this symbol appears in a command, it and the remainder of the line are ignored by QRS. It also has this same effect when it appears in macro text (Chapter VII), a READ file (Chapter VIII, or a startup file (Chapter XI). As an example, the comment lead symbol can be changed from its default of ; to \ with the following SET command:

```
SET CL \
```

CN Character Class Negation Symbol **CN**

The value of the CN parameter indicates the negation symbol for use within a character class. For instance, to change this symbol from its ^ default value to -, the following SET command is used:

```
SET CN -
```

CP Console Page Eject **CP**

The value of the CP parameter is used to indicate the console page eject sequence. This can consist of up to four integer values, any of which can be changed from its 0 default value. The permissible range for each value is 0 through 255. These should be the ASCII numbers (Appendix B) that correspond to the console's page eject sequence. Each time the console depth (see CD) is reached, the CP character sequence is used to clear the screen. For example, if the console's page eject character sequence is ESC [2 J (e.g., under MSDOS), then the appropriate SET command is:

```
SET CP 27,91,50,74
```

If CP is left at its default value of 0,0,0,0 then no page eject will occur when the screen is full during the QRS generation of a report. In other words, the screen is not cleared before the start of a new page. Its contents simply scroll upward instead.

CS Column Spacing **CS**

The value of the CS parameter specifies how many spaces will appear between columns in reports generated by the LIST command. For instance, to set the column spacing at 4 spaces, the following SET command is used:

```
SET CS 4
```

The default value for CS is 2.

CW Console Width CW

The value of the CW parameter specifies how many display characters are permitted per console line. If 79, rather than the default of 80, characters are desired per console line, then the appropriate SET command is:

```
SET CW 79
```

The default value of CW is 80.

FN File Name FN

The value of FN specifies the name of a file that is to be used to hold the output from a WRITE or SPEW command. Using environmental option 4 (described below) output from a LIST command can also be written to the named file. When assigning a file name to the FN parameter, the file name must be fully qualified within the host operating system and must be quoted within the SET command. There is no default value for FN. For instance, to set the FN parameter to be the file named WFILE on drive A under the PCDOS operating system, the proper SET command is:

```
SET FN  "A:WFILE"
```

M1 Match-One Symbol M1

The value of the M1 parameter indicates the match-one symbol that is to be used by QRS in evaluating relational expressions. As noted in Chapter 2, $ is the default match-one symbol. If a different match-one symbol is desired, say ?, then the SET command is used (quoting the new match-one symbol):

```
SET M1  "?"
```

MS Match String (wildcard) Symbol MS

The value of the MS parameter indicates the wildcard symbol that is to be used by QRS in evaluating relational expressions. As noted in Chapter 2, * is the default match-string symbol. If a different match-string symbol is desired, say :, then the SET command is used (quoting the new match-string symbol):

```
SET MS  ":"
```

OC Open Character Class Symbol OC

The value of the OC parameter indicates the open class symbol for a character class. For example, to change this symbol from its [default to (, the following SET command is used:

```
SET CC  (
```


OF Output Format OF

The value of the OF parameter is two digits. The two digits indicate the format that QRS will use in displaying numeric data values. The first digit shows the number of digits that are to be displayed to the left of a decimal point. The second digit indicates the number of digits that are to be displayed to the right of the decimal point (and counting the decimal point as one of these). An OF value of 30 will result in the display of the 3 low order digits to the left of the decimal point (no decimal point is displayed). An OF value of 31 has the same effect except that the decimal point is also displayed. An OF value of 32 yields the display of 3 digits to the left and 1 digit to the right of a decimal point. The OF default is 89. This is the maximum display precision allowed. As an example, if an output format consisting of 2 digits to the left and 6 digits to the right is needed, then the appropriate SET command is:

```
SET OF 27.
```

PC Printer Close PC

The PC parameter has a sequence of four integer values, any of which can be changed from its 0 default value. The permissible range for each value is 0 through 255. Integer equivalents to the various ACSII characters are given in Appendix B. These values are routed to the printer at any point where printer output is suppressed, by setting environment option 3 to on (as described later in this chapter). The values are used to control printer operations when QRS is informed that printer output is to be suppressed. The default values have no effect on printer operations. As an example, the close control sequence for the printer can be set to 2 (Control-B), 4 (Control-D), 23 (Control-W), 0 by using the SET command:

```
SET PC 2, 4, 23, 0.
```

The values can be separated by commas or blanks.

PD Printer Depth PD

The value of the PD parameter specifies how many lines to print before the next printer page eject occurs. The default is 60. To cause a page eject after each 40 lines of printer output, the appropriate SET command is:

```
SET PD 40
```


PM Printer Margin PM

The value of the PM parameter indicates the width of the left printer margin. For example, to cause printer output to begin in column 6, the following SET command is used:

SET PM 5.

PO Printer Open PO

The PO parameter has a sequence of four integer values, any of which can be changed from its 0 default value. The permissible range for each value is 0 through 255. Integer equivalents to the various ACSII characters are given in Appendix B. These values are routed to the printer whenever the printer is enabled, by setting environment option 3 to off (as described later in this chapter). The values are used to control printer operations when QRS is informed that printer output is to be suppressed.

The default values have no effect on printer operations. As an example, the open control sequence for the printer can be set to 1 (Control-A), 2 (Control-B), 5 (Control-E), 26 (Control-Z) by using the SET command:

SET PO 1, 2, 5, 60.

The values can be separated by commas or blanks.

PP Printer Page Eject PP

The value of the PP parameter is used to indicate the printer page eject character. The PP value must be numeric. If a printer's page eject character (see the printer manual) is not numeric, then that character is represented by the corresponding ASCII number (Appendix B). For example, if the printer's page eject character is a Control-E, then the appropriate SET command is:

SET PP 5.

If PP is left at its default value of 0, then no page eject will occur during the QRS generation of a report. In other words, a form feed to the next page does not occur when the printer depth (controlled by PD) is exceeded.

PU Printer Utility **PU**

This parameter normally exists only in UNIX and UNIX-like environments. Its value is a quoted string of characters that identify the printer utility that is being used to send output to the printer. For instance, to identify lpr as the printer utility, the appropriate SET command is:

```
SET PU "lpr"
```

The default value depends on the environment. In many environments the default is "lpr". However, there are exceptions such as the ATT 3B series where "lp" is the default value of PU.

PW Printer Width **PW**

The value of the PW parameter specifies how many display characters are permitted per printer line. If 100, rather than the default of 120, characters are desired per printer line, then the appropriate set command is:

```
SET PW 100.
```

The default value of PW is 120.

SC Nested Select Close Symbol **SC**

The value of the SC parameter is the symbol that can be used to terminate a SELECT command that is nested within a query (Chapter X). For example, to change this symbol from its } default to a backslash, the following SET command is used:

```
SET SC \
```

SF Scale Factor **SF**

The value of the SF parameter is the scale factor that QRS applies to idec data values prior to listing them or computing statistics with them. The SF value indicates the number of positions that will appear to the right of the decimal point for each idec value in the output report. The SF default value is 0.

Data item values of the idec type do not have a decimal location stored with them. Thus the scale factor parameter is used to control the decimal point placement in the display of these values. For instance, if the idec values are to be displayed with two digits to the right of a decimal point then the set command is:

```
SET SF 2.
```

SO Nested Select Open Symbol **SO**

The value of the SO parameter is the symbol that can be used to begin a SELECT command that is nested within a query (Chapter X). For example, to change this symbol from its { default to a colon, the following SET command is used:

```
SET SO :
```

TL Title **TL**

The value of the TL parameter is a quoted sequence of up to 80 characters. This character sequence appears as the title on each page of a report generated by the QRS list command. If no title is specified (the default), then report pages will have no titles. Titles can be changed with a set command at any time during a query session. For instance, to set the page titles to be EMPLOYEE EARNINGS REPORT, the SET command is:

```
SET TL "EMPLOYEE EARNINGS REPORT".
```


Setting Environmental Options

Summary

The following is a summary of the available environment options and the default status (on or off) of each. A full description of each is provided after this summary.

Option number	Default status	Meaning of **on**
OPT 1	OFF	interactive I/O echo to printer
OPT 2	ON	report output displayed on console
OPT 3	ON	suppress printer output
OPT 4	OFF	spool printer output to disk (FN)
OPT 5	OFF	echo alternate input
OPT 6	OFF	echo write output to console
OPT 7	OFF	print output from write
OPT 8	OFF	echo one term per line
OPT 9	OFF	suppress column headings
OPT 10	OFF	suppress value labels
OPT 11	OFF	suppress macro recognition
OPT 12	OFF	echo macro expansion output
OPT 13	ON	suppress printer form feed
OPT 14	OFF	perform sort on query output
OPT 15	OFF	pause after each report page
OPT 16	OFF	ignore case in relational expressions
OPT 17	ON	ascending sort order
OPT 20	OFF	print record access count
OPT 21	ON	print number of observations
OPT 22	ON	print maximum observations
OPT 23	ON	print minimum observations
OPT 24	ON	print sum of observations
OPT 25	ON	print mean of observations
OPT 26	ON	print variance of observations
OPT 27	ON	print std dev of observations
OPT 28	OFF	suppress all statistics

To switch the status of any of these environment options to **on** or **off**, the syntax is

<**set**><option><status>

When setting an option, if no status is specified then the status is toggled (i.e., if it was previously on, it will be turned off and vice versa).

Environment Option Details

OPT 1 Interactive I/O Echo to Printer **OPT 1**

When this option is off, only report output is routed to the printer. When this option is on, all interactive input and output is routed to the printer, as well as to the console. The default status is off. This option only has effect if Opt 3 is off (i.e., the printer has not been suppressed).

OPT 2 Report Output Displayed on Console **OPT 2**

When this option is on, output that does not result from a SPEW or WRITE command is displayed on the console. The default status is on. If CP has been set to the console's page eject sequence and this option is on, the screen will clear whenever it is full and then output continues. The screen is considered to be full when the console depth (controlled by CD) is reached. If CP is not set in this way, output scrolls on the screen.

OPT 3 Suppress Printer Output **OPT 3**

When this option is on, output is not routed to the printer. Whenever this option is turned on, the printer close control sequence is routed to the printer (see the PC environment parameter). Whenever this option is turned off, the printer open control sequence is routed to the printer (see the PO environment parameter). The printer should be on-line and switched on before turning off this option. The default status is on.

OPT 4 Spool Printer/Console Output to Disk **OPT 4**

When this option is on, all output that would normally appear at the console, when OPT 2 is on, is also spooled to the disk file specified by the FN environment parameter. The default status is off.

OPT 5 Echo Alternate Input **OPT 5**

A sequence of queries can be batched onto a text file and then executed non-interactively with the READ command (Chapter VIII). To have the text file queries echoed to the console, this option should be on. The default status is off.

OPT 6 Echo Write Output to Console **OPT 6**

When this option is on, output from the WRITE command is echoed to the console. The default status is off.

OPT 7 Print Output from Write **OPT 7**

When this option is on, output from the WRITE command is also routed to the printer. The default status is off.

OPT 8 Echo One Term per Line **OPT 8**

When this option is on, data values in the QRS output are displayed one value per line. Since there are no columns, column headings do not appear. The default status is off.

OPT 9 Suppress Column Headings **OPT 9**

While this option is on, no column headings appear in any QRS output. The default status is off.

OPT 10 Suppress Value Labels **OPT 10**

While this option is on, no value label assignments (as defined with the DEFINE command: see Chapter VII) are made. The default status is off.

OPT 11 Suppress Macro Recognition **OPT 11**

While this option is on, all macros that have been defined with the DEFINE command (see Chapter VII) are ignored by QRS. If this option is on when DEFINE is used to define a macro, QRS keywords can be defined as macros. Those macros will later be recognized if this option is turned off. The default status is off.

OPT 12 Echo Macro Expansion Output **OPT 12**

When this option is on, every command is echoed to the console. If a command contains macros, the full expansions of those macros are inserted into the command echo. The default status is off. This command is particularly helpful when first becoming acquainted with macros.

OPT 13 Suppress Printer Form Feed **OPT 13**

While this option is on, the printer page eject parameter (PP) is temporarily viewed as being 0. The default status is on.

OPT 14 Perform Sort on Query Output **OPT 14**

When this option is on, QRS will perform an internal line sort on the output from a LIST command. Dates are sorted chronologically. Control breaks are not allowed if this option is on. The default status is off. The user may see no visible response from his query for quite some time if this option is on. This is because the entire retrieval is completed before sorting begins. The user is given the message "retrieval complete - sort in process" upon completion of the retrieval phase and prior to the sort phase.

OPT 15 Pause After Each Report Page **OPT 15**

When this option is on, QRS will pause at the end of each report page. Pressing any key will cause the next report page to be displayed. The default status is off.

OPT 16 Ignore Case **OPT 16**

When this option is on, all character values of terms in the conditional clause are converted to upper case before evaluating relational expressions. This is useful when making name comparisons. For example, LASTNAME="deBeer" will be satisfied regardless of whether last names are stored in upper or lower case (or a mixture thereof): DEBEER, debeer, DeBeer, etc. This option does not affect the cases (upper or lower) in which retrieved data are displayed.

OPT 17 Ascending Sort Order **OPT 17**

When this option and option 14 are both on, an ascending sort order is used. The default status on. If this option is off while option 14 is on, a descending sort order is used.

OPT 20 Print Record Access Count **OPT 20**

When this option is on, QRS will indicate the number of record occurrence accesses that were required to answer a query. The default status is off. This option is very useful for "fine tuning" the formulation of those queries that may receive extensive use. Note that the record access count is not necessarily equal to the number of observations statistic, since many records may be retrieved that do not satisfy the conditional clause. This count is not displayed at control breaks; it is only provided upon completion of the entire query.

OPT 21 Print Number of Observations **OPT 21**

When this option is on, any statistical computations (involved in the STATS or LIST command) will include a calculation of the number of observations. This statistic will be displayed. The default status is on.

OPT 22 Print Maximum Observation **OPT 22**

When this option is on, any statistical computations (involved in the STATS or LIST command) will include a determination of the maximum observations. This statistic will be displayed. The default status is on. This value is of course only printed for numeric terms.

OPT 23 Print Minimum Observation **OPT 23**

When this option is on, any statistical computations (involved in the STATS or LIST command) will include a determination of the minimum observations. This statistic will be displayed. The default status is on. This value will only be printed for numeric terms.

OPT 24 Print Sum of Observations **OPT 24**

When this option is on, any statistical computations (involved in the STATS or LIST command) will include a calculation of the sum of observations. This statistic will be displayed. The default status is on. This value will only be printed for numeric terms.

OPT 25 Print Mean of Observations **OPT 25**

When this option is on, any statistical computations (involved in the STATS or LIST command) will include a calculation of the mean of observations. This statistic will be displayed. The default status is on. This value is only displayed for numeric terms.

OPT 26 Print Variance of Observations **OPT 26**

When this option is on, any statistical computations (involved in the STATS or LIST command) will include a calculation of the variance of observations. This statistic will be displayed. The default status is on. This value is only displayed for numeric terms.

OPT 27 Print Standard Deviation of Observations **OPT 27**

When this option is on, any statistical computations (involved in the STATS or LIST command) will include a calculation of the standard deviation of observations. This statistic will be displayed. The default status is on.

OPT 28 Suppress Statistics **OPT 28**

While this option is on, all statistics that would otherwise be computed by a LIST command are suppressed. When this option is on, it overrides the status of each of options 21 through 27. The default status is off. **Turning this option on results in a very substantial reduction in the cpu time required to answer a retrieval query** (up to a 30% reduction depending on the nature of the query and cpu).

This option is used to reduce the execution time of those queries for which statistics are not needed. Suppressing statistics elminiates time-consuming floating point operations which are normally performed for each retrieved line. In addition, significantly less memory is used since large statistical tables are not generated.

Setting Column Headings

The SET command can also be used to specify column headings that will override the headings that would normally appear in the output report from a LIST command. The syntax for accomplishing this is:

<**set**><**ch**><heading list>

where the heading list is a sequence of the alternative column headings. Headings are separated by commas or blanks. A heading must be within quotes if it contains a non-alphanumeric character. The position of a heading in the heading list indicates the report column for which it will be used. Notice that the width of a column heading cannot exceed the width of the data values printed in that column. If the column heading does exceed the width of the corresponding term's values, then the column heading is truncated.

Use of the command SET CH resets to null any previously defined column headings. Hence, to turn off column headings when they are no longer needed, the SET CH command should be used with no column heading list.

The following examples illustrate the manner in which alternative column headings can be set.

SET CH DEPT	sets the heading of the first report column to DEPT
SET CH FIRST,,,EARNINGS	sets the first report column heading to be FIRST and the fourth to be EARNINGS
SET CH,,"LAST NAME"	sets the second report column heading to be LAST NAME
SET CH	eliminates all alternative column headings
DISPLAY C	displays all alternative column headings that are presently in force

D. Examples

```
SET CW  75
SET MS  ":"
SET TL  "JOB and SKILLS Report"
SET OPT 1  ON
SET OPT 24  OFF
SET CH, "BIOGRAPHY INFORMATION"
```

Sample query session using the SET and LIST commands:

```
-->list name lastname for name = 'accounting' thru idep has
NAME          LASTNAME

accounting    Woods
accounting    Lochman
accounting    Handee

   no of observations:     3

      10 record accesses


-->list lastname fname for dnumber in [1 3 4] thru idep has
LASTNAME      FNAME

Duranger      Kevin
Ferston       Greg
Fiskum        Mark
Laisler       Cameron
Hanson        Richard
Madison       Greg
Koelder       Dave
Stokes        Tim
Russet        Gary
Gagster       Mike
Crooner       Sharon
Durman        Jim
Boumer        Diane
Gilvaston     Dave
Lowder        Dave
Milwright     Tom
Dayton        Jim
More          John
Grant         Bill
Woods         Jean
Lochman       Beverly
Handee        Kevin

   no of observations:    22

      29 record accesses
```

```
-->set opt 14
-->list lastname name for dnumber in [1 3 4] thru idep has
retrieval complete - sort in process
LASTNAME      NAME

Boumer        sales
Crooner       sales
Dayton        sales
Duranger      R and D
Durman        sales
Ferston       R and D
Fiskum        R and D
Gagster       R and D
Gilvaston     sales
Grant         sales
Hanson        R and D
Handee        accounting
Koelder       R and D
Laisler       R and D
Lochman       accounting
Lowder        sales
Madison       R and D
Milwright     sales
More          sales
Russet        R and D
Stokes        R and D
Woods         accounting

   no of observations:    22

      29 record accesses
```

```
-->list lastname fname for dnumber in [1 3 4] thru iemp >has
retrieval complete - sort in process
LASTNAME        FNAME

Boumer          Diane
Crooner         Sharon
Dayton          Jim
Duranger        Kevin
Durman          Jim
Ferston         Greg
Fiskum          Mark
Gagster         Mike
Gilvaston       Dave
Grant           Bill
Hanson          Richard
Handee          Kevin
Koelder         Dave
Laisler         Cameron
Lochman         Beverly
Lowder          Dave
Madison         Greg
Milwright       Tom
More            John
Russet          Gary
Stokes          Tim
Woods           Jean

   no of observations:    22

      68 record accesses
```

```
-->set opt 14
-->stats ytdearn thru iemp

YTDEARN:
   no of observations:    34
 114050.000000000 sum
  10000.000000000 max
    100.000000000 min
   3354.411764706 ave
5266722.370766488 var
   2294.934066758 std

      34 record accesses

-->set of 83
-->stats ytdearn thru iemp

YTDEARN:
   no of observations:    34
  114050.00 sum
   10000.00 max
     100.00 min
    3354.41 ave
 5266722.37 var
    2294.93 std

      34 record accesses

-->
```

This page intentionally left blank.

VII. DEFINE AND DATFORM COMMANDS

A. Syntax

<define>
<datform>

B. Synopsis

The DEFINE command can be used to define macros (i.e., new, user-defined commands or parts of commands) and value labels. These definitions are held in the data base and therefore exist across query sessions. The definition process is interactive and menu-driven.

A macro is a word that can be equated to any section of a query and that can be freely substituted for that portion of a query. It can also be equated to an entire query command or to a sequence of queries (involving many lines of input). Thus macros provide a convenient means for storing and later utilizing often-used queries or sections of a query (e.g., a particular path clause). For instance, one might store a query that provides end of the month employee earnings data under the macro name EARNINGS. This has the effect of defining a new command. Typing the macro name EARNINGS will generate the desired report. A macro for a series of queries is especially useful for setting up environment parameters and options. This allows the environment to be quickly changed by submitting a single user-defined (i.e., macro) command. The macro facility for defining new, extremely high level commands permits QRS to be easily tailored for use by end user's who do not have the slightest technical knowledge.

A value label can be defined for a particular data item value. When such a definition is made, the item value will not appear in a report. Its value label is displayed instead. A value label is always defined with respect to a particular data item. As an example, for the JOBCODE data item, we may want to define the value label MANAGER for the value 32. Whenever QRS retrieves the value 32 for JOBCODE, the MANAGER value label appears in the report.

Because synonyms are treated very similarly to macros, a synonym will not be recognized within macro text.

The DATFORM command is used to convert the standard MDBS form of date data (mm/dd/yyyy) to an alternative form. This conversion is automatically performed as date values are output by a retrieval command.

C. DEFINE Description

When the DEFINE command is given, QRS responds with the prompt:

Define Macros or Value Labels? (M/V)

If the user responds with **M**, a menu for defining macros appears. If the user responds with **V**, a menu for defining value labels appears.

Macro Definition

When a QRS user selects the macro definition option, the following menu is displayed:

```
                    Macro Definition Utility

        Macro/Synonym Functions:
        (A)  Add a Macro
        (C)  Change a Macro
        (D)  Delete a Macro
        (I)  Index of Macro/Synonym Names
        (L)  List Macros and Synonyms
        (?)  Print This Command List
        (Q)  Quit Define Mode

        Function?
```

The effects of these seven macro definition functions are described below.

A: The user is prompted for the name of the new macro and for the query text that it will represent. A macro name must be alphanumeric. It cannot begin with a digit. Nor can it be a keyword (unless option 11 is on), an environment parameter name, or the single letter A, E, I, O, R, or S. Use of this macro name within a query will cause the macro's query text to be substituted for the macro name. The text for a macro plus the length of the macro name cannot exceed 255 characters and macros occurring within the text of another macro are ignored. The macro is not translated to upper case; it must be referenced in the same case with which it is defined in this function.

C: The user is prompted for the name of an existing macro. The user is then prompted for the new text for that macro.

D: The user is prompted for the name of the macro to be deleted.

I: An index of all macros and synonyms is displayed. Synonyms are indented, macros are not.

L: The user is prompted for a macro or synonym name. Pressing the return key will result in a listing of all macros (and their text) and all synonyms. If the user enters the name of a particular macro, its text is displayed. If the user enters the name of a particular synonym, then the name of the area, set, record type or data item for which it is a synonym is displayed. A user may not define a macro that has the same name as a synonym.

?: The macro definition functions are displayed.

Q: The user leaves the macro definition mode. QRS gives the --> prompt.

Value Label Definition

Value labels are not permitted for all data item types. Those types for which value labels are allowed are: STRING, INTEGER, CHARACTER and UNSIGNED. When a QRS user selects the value label definition option, the following menu is displayed:

```
        Value Label Definition Utility

    Value Label Functions:
      (A)  Add a Value Label
      (C)  Change a Value Label
      (D)  Delete a Value Label
      (I)  Index of Value Labels
      (L)  List Value Labels
      (?)  Print This Command List
      (Q)  Quit Edit Mode

    Value Label Function?
```

The effects of these seven value label definition functions are described below.

A: The user is prompted for the names of the record type and data item for which value labels are to be created. The user is also prompted for the maximum value label length. If an attempt is made to add a value label that exceeds this length, the label is truncated. The user is then iteratively prompted for values and labels. This iterative prompting is terminated by entering a blank line, which results in a prompt for another value label function. If labels are assigned to some values of a data item, but not to other values of that data item, then in a report a series of asterisks are substituted for each value not having a label.

C: The user is prompted for the names of the record type and data item for which value labels are to be changed. An error message results if this data item presently has no value labels. If the data item presently has value labels, they are deleted and the user is prompted just as with the A function.

D: The user is prompted for the names of the record type and data item whose values are to be deleted. All value labels that have been defined for the data item are deleted.

I: An index of all data items (and their respective record types) for which value labels are defined is displayed.

L: The user is prompted for a record type name. Pressing the return key will result in a listing of all value labels. Entering a record type name will result in a prompt for a data item name. When a data item name is entered, a listing of value labels defined for that data item is generated.

?: The value label functions are displayed.

Q: The user leaves the value label definition mode. QRS gives the --> prompt.

D. DEFINE Examples (underlined characters are entered by the user)

```
--> define
Define Macros or Value Labels? (M/V)M

                    Macro Definition Utility

     Macro/Synonym Functions:
     (A)  Add a Macro
     (C)  Change a Macro
     (D)  Delete a Macro
     (I)  Index of Macro/Synonym Names
     (L)  List Macros and Synonyms
     (?)  Print This Command List
     (Q)  Quit Define Mode
Function ?A
Name?Ldept

Enter text below (terminate with an empty line)
* list dnumber, location thru idep
*

Function?I
     DNO
     SC
Ldept

Function?L
Name?SC

For item SKILCODE
Synonym is SC



Function?C
Name?Ldept

Enter text below (terminate with an empty line)
* list dnumber,name,location thru idep
*

Function?A
Name?Path1

Enter text below (terminate with an empty line)
* THRU IDEP,HAS,FILLEDBY
*

Function?D
Name?Ldept
```



```
Function?A
Name?JS

Enter text below (terminate with an empty line)
* set_ch_"Job_description","Skills_needed"
* list_job.descript_skill.descript_thru_ijob_needs
*

Function?A
Name?Sorter

Enter text below (terminate with an empty line)
* set_opt_12_on
* set_opt_14_on
* set_opt_15_on
* set_cp_26
* list_lastname,_name_thru_idep_has
* set_opt_14
* set_opt_12_off
*

Function?A
Name?Techs

Enter text below (terminate with an empty line)
* set_opt_12_on
* set_cp_26
* set_opt_15_on
* list_lastname_name_for_dnumber_in_[1,2,6]_thru_idep_has
* set_opt_12_off
*

Function?Q
```



```
-->define

Define Macros or Value Labels?(M/V) v

     Value Label Definition Utility

Value Label Functions:
 (A)  Add a Value Label
 (C)  Change a Value Label
 (D)  Delete a Value Label
 (I)  Index of Value Labels
 (L)  List Value Labels
 (?)  Print This Command List
 (Q)  Quit Edit Mode

Value Label Function? a

Record Name: dept
Item Name: dnumber
Label Length? 20

Value: 1
Label: accounting
Value: 2
Label: engineering
Value: 3
Label: physical plant
Value: 4
Label: marketing
Value: 6
Label: clerical
Value:

Value Label Function? l

Record Name:

For item DNUMBER in record DEPT, value labels are:

          *Value* *Label*

                1 accounting
                2 engineering
                3 physical plant
                4 personnel
                5 marketing
                6 clerical

Value Label Function? l

Record Name: dept
Item Name: dnumber
```

```
For item DNUMBER in record DEPT, value labels are:

          *Value* *Label*

                1 accounting
                2 engineering
                3 physical plant
                4 personnel
                5 marketing
                6 clerical

Value Label Function? i

Index of Value Labels

        RECORD    ITEM

        DEPT      DNUMBER

Value Label Function? a

Record Name: dept
Item Name: name
Label Length? 25

Value: acctng
Label: accounting
Value: mkting
Label: marketing
Value: engring
Label: engineering
Value: psnl
Label: personnel
Value: clk
Label: clerical
Value: physpl
Label: physical plant
Value:

Value Label Function? l

Record Name:

For item DNUMBER in record DEPT, value labels are:

          *Value* *Label*

                1 accounting
                2 engineering
                3 physical plant
                4 personnel
                5 marketing
                6 clerical
```



```
For item NAME in record DEPT, value labels are:

     *Value*         *Label*

     acctng          accounting
     mkting          marketing
     engring         engineering
     psnl            personnel
     clk             clerical
     physpl          physical plant

Value Label Function? i

Index of Value Labels

        RECORD    ITEM

        DEPT      DNUMBER
        DEPT      NAME

Value label Function? q
-->
```

E. Advanced Usage of Macros with Parameters

When specifying the text of a macro with the DEFINE command, up to 9 parameters can be embedded within the macro text. Each macro parameter functions as a variable, whose value can be stated by a user whenever the macro is executed. For example, the text of a macro called WORKERS might be defined as:

```
LIST &1,&2 THRU IEMP
```

where &1 and &2 are used to denote macro parameters. If a user is interested in the ID and LASTNAME of workers, then the macro would be executed as:

```
WORKERS(ID,LASTNAME)
```

If the user is later interested in FNAME and LASTNAME of workers, then the same macro is executed, but with different parameter values:

```
WORKERS(FNAME,LASTNAME)
```

Another user may want a listing of employee IDs and year-to-date earnings in thousands:

```
WORKERS(ID,YTDEARN/1000)
```

As the above examples show, QRS substitutes the values specified by a user for the parameters declared in the original macro text. The first value appearing in the parentheses following the macro name is substituted for &1, the second value is substituted for &2, and so forth.

As a parameter is specified in macro text, it can be defined to be an optional parameter. When a user executes a macro, no value needs to be indicated for an optional parameter. A parameter is defined to be optional by appending the parameter with the = symbol followed by a space. For example, if the macro text for WORKERS is

```
LIST &1 &2= &3 THRU IEMP
```

then the second parameter is optional. When executing the macro, a value for the second parameter may be omitted

```
WORKERS(LASTNAME,,FNAME)
```

or it may be included

```
WORKERS(ID,FNAME,LASTNAME)
```

Default values for parameters can also be specified in the macro text. For instance, if the macro text for WORKERS is

```
LIST &1 &2 &3=ID THRU IEMP
```

then ID is the default value for this macro's third parameter. Whenever a user executes WORKERS without specifying a value for the third parameter, QRS uses ID as the value. For instance,

```
WORKERS(FNAME,LASTNAME,)
```

yields a listing of first names, last names, and IDs. If a user explicitly states a value for the third parameter, then QRS ignores the default value. The query:

```
WORKERS(FNAME,LASTNAME,YTDEARN)
```

yields an output table having first names, last names and year-to-date earnings.

A parameter can appear more than once in a macro's text. As an example, suppose the text for a macro named TNAMES is defined to be:

```
LIST &1 &2= FOR &1 = "T*" THRU IEMP
```

This macro will produce a listing of names beginning with the letter T. The query:

```
TNAMES(FNAME,LASTNAME)
```

yields a listing of employees whose first names begin with T. In order to get a listing of last names that begin with T, the query would be:

```
TNAMES(LASTNAME,)
```

As another example of multiple occurrences of the same parameter, consider the SHOWEMP macro defined to be:

```
LIST &1 FOR &1 = &2="*" THRU IEMP
```


To get a list of all first names beginning with K, the query would be:

```
SHOWEMP(FNAME,"K*")
```

Alternatively, to get a list of all last names, the query would be:

```
SHOWEMP(LASTNAME,)
```

Rules for Specifying Parameters in Macro Text

1. &1, &2, ..., &9 are valid macro parameters. If n parameters are specified, they should be &1, ... &n (where $1 \leq n \leq 9$); &1 should be the leftmost parameter in the macro text; the remaining parameters can be specified in any order within the macro text.

2. Any parameter appearing in a macro's text can appear multiple times within that text.

3. To signify that a parameter is optional, an equal sign and space must immediately follow the parameter number (e.g., &5=) wherever it appears in the macro text.

4. To specify a default value for a parameter, the parameter number is immediately followed by an equal sign and the desired default value. Any letter or digit can appear in a default value. Any of the following characters is also allowed

   " [ ] . / * + - < > = # $

   No spaces, commas, or parentheses are allowed in the default value unless they are within a matching pair of double quotes. A parameter or macro cannot appear as part of a default value.

   Examples:

   ```
   &2=100                    &2="George Washington"
   &1="programmer"           &1=18/2.3*
   ```

   If a parameter appears multiple times in a macro's text, it can have a different default value at each place it appears.

Rules for Executing a Parameterized Macro

1. The macro name is followed immediately (no space) by a matching set of parentheses containing the desired values. The values are separated by commas.

2. Macro evaluation is based on positional substitutions. QRS substitutes the first value in the parentheses for &1 (the leftmost parameter), the second value is substituted for &2 (the next parameter), and so forth. If no value is given for a parameter (i.e., there are two consecutive commas), then the parameter's default value is used. If it does not have a default value, but is optional, then the parameter is ignored. Omitting a value for a required parameter results in an error message.

3. The number of commas in a macro's list of values must always be one less than the number of parameters. If too many or too few values are specified, an error message results.

4. A parameter value cannot contain spaces, commas, or parentheses unless they are enclosed in a matching pair of double quotes. A macro cannot be used as a parameter value. In addition to digits and numbers, a value can contain any of the following characters:

" [ ] . / * + - < > = # $

F. DATFORM Description and Examples

The DATFORM command can be invoked with or without an argument. If an argument is specified it must be 1, 2, 3, or 4. The number specified with DATFORM indicates the form that QRS should use for the output of date values. The format correspondence for these numbers is as follows:

1	dates are output in the form:	mm/dd/yyyy
2	dates are output in the form:	dd/mm/yyyy
3	dates are output in the form:	yyyy/mm/dd
4	dates are output in the form:	yyyy/dd/mm

Fo example,

--> datform 2

causes all subsequently retrieved dates to be presented in the day-month-year form. DATFORM can also be invoked with no argument, which results in prompts for the four date conversion options. For instance, the effect of the previous DATFORM example could also be accomplished by:

--> datform

Database Date Conversion Options:

(1) Set date format to mm/dd/yyyy
(2) Set date format to dd/mm/yyyy
(3) Set date format to yyyy/mm/dd
(4) Set date format to yyyy/dd/mm

Option? 2

The DATFORM command can be invoked wherever and whenever desired during the course of a QRS session to change date formats. Upon initial entry into a QRS session, the standard format (1) is in force unless a DATFORM command in a STARTUP file has been used to convert to a different form.

This page intentionally left blank.

VIII. READ AND ECHO COMMANDS

A. Syntax

<**read**><file-name>
<**echo**><message>

B. Synopsis

The READ command causes QRS to read a disk file containing a series of QRS commands. QRS then executes all of these QRS commands. The file that contains the QRS commands is indicated by the file-name clause.

An ECHO command causes a specified message to appear on the console screen. By embedding ECHO commands within the file processed by a READ command, an ongoing commentary of the READ command's processing actions can be provided.

C. Description

In using the READ command, the indicated file name must be quoted and must be fully qualified within the host operating system, including the drive number if it is not on the default drive. The file of QRS commands can be read, prepared, and/or saved through the same MDBS.DDL facilities that are used for creating, editing, and saving DDL specifications. Alternatively, any standard text editor can be used to create, read, and save the file of QRS commands. Any valid QRS command can appear in the READ command file, with the exception of a READ command. Hence, environment parameters and options can easily be set to control where the reports will be routed (console, printer, disk). If an error is encountered, the READ command is terminated and any remaining lines in the READ command file will not be read.

Queries on the READ command file will be echoed to the console whenever option 5 is on. In addition, ECHO commands can be used in the file to cause messages to appear interspersed among query results on the console screen. The message following an **echo** keyword will be output to the next line on the screen. The message can be either a number or a string of characters enclosed in quotes. If there is no message, a blank line is output. Other situations where the ECHO command can be useful are within the text of a macro (Chapter VII) or within a startup file (Chapter XI).

D. Examples

```
READ "QFILE"
READ "A:BATCH.TXT"
ECHO 7
ECHO "     Sales Summary     "
ECHO 'end of summary'
ECHO "This is the 'best' result"
ECHO "The macro text for SE is "SELECT"'
ECHO
```

This page intentionally left blank.

IX. OPEN, CLOSE AND QUIT COMMANDS

A. Syntax

<**open**><area-name><file-name>
<**close**><area-name>
<**quit**> or <**bye**>

B. Synopsis

When QRS needs to access a record occurrence that is in an unopened area, QRS will automatically open the area, using the file that was specified for that area in the DDL specifications. If the area's file name has been changed so that it differs from the file name in the DDL specifications, then the QRS open command can be used to open the area by specifying its new file name.

The close command closes an area. The quit (or bye) command is used to terminate a query session.

C. Description

1. OPEN

The open command is used to open an area for QRS accessing the following syntax:

<**open**><area-name><file-name>

The optional file name must be the name of the file on which the indicated area resides. It is necessary only if the area's file name has changed since the data dictionary was generated.

2. CLOSE

The close command is used to close areas previously opened with a QRS open command. Only the area name needs to be specified using the syntax:

<**close**><area-name>

Closing an opened (but no longer needed) area will free up buffer space, resulting in faster QRS response times.

3. QUIT

The quit command terminates QRS and returns the user to the operating system. QUIT closes the data base (saving all macro and value label definitions in the main data base area). If a data base is not closed, it cannot be used in the future. The "soft interrupt" key (see the appropriate system specific manual) will abort any current QRS activity, but will not return control to the operating system. The QRS --> prompt results.

In addition to **quit, bye** may also be used to close the data base and return the user to the operating system.

Quit and **bye** are not recognized from either DEFINE mode or COMPUTE mode.

Successful execution of the QUIT command is indicated by the "data base successfully closed" message.

X. QUERY NESTING

There is a special type of QRS command that can be nested within the conditional clause of a LIST command. This nested command has the following syntax:

{<**select**><term><conditional clause><path clause>}

where <term> refers to any permissible find clause term. A nested command can be used in place of a group of numeric or character constants, following the IN operator in a relational expression.

The SELECT command is comparable to a LIST command, except that it has only one term in place of a find clause, it does not have a break clause, and it does not produce a listing. The term can have either character or numeric data values. QRS retrieves the term's data values, but does not list them. Instead, QRS treats them as a group of numeric or character constants for use in evaluating the relational expression that contains a nested query. In effect, a nested SELECT command defines an implicit group of either character or numeric constants. Instead of explicitly specifying numeric or character constants within the [] symbols, the user specifies an implicit group of constants with the SELECT command.

As an example, suppose we want a list of all employees that fill the same job as Lehr. The appropriate query, with nesting is[1]:

```
LIST ID,LASTNAME,FNAME FOR JOBCODE IN {SELECT JOBCODE FOR
     LASTNAME="Lehr" THRU IEMP,>FILLEDBY} THRU IJOB, FILLEDBY
```

An alternative way to generate the same report involves the use of two LIST commands. The first would be:

```
LIST JOBCODE FOR LASTNAME="Lehr" THRU IEMP,>FILLEDBY
```

Suppose that this command yields a report showing that Lehr has a job code of 3. Then a second LIST command is used to generate the desired report:

```
LIST ID,LASTNAME,FNAME FOR JOBCODE=3 THRU IJOB,FILLEDBY
```

This report will be identical to the one produced by using the nested query shown above.

As a second example, suppose we want a list of all employees in department 2 who fill the same job as Lehr. The appropriate query, with nesting is[1]:

```
LIST ID,LASTNAME,FNAME FOR DNUMBER=2 AND JOBCODE IN
     {SELECT JOBCODE FOR LASTNAME="LEHR" THRU IEMP,>FILLEDBY}
     THRU IJOB, FILLEDBY
```

[1] Recall that a QRS command is terminated by pressing the carriage return key.

As a third example, suppose we want a report of all employees that have a skill in common with Lehr. The report should show which skills(s) each such employee has in common with Lehr. The appropriate QRS command[1] is:

```
LIST ID,SKILCODE FOR SKILCODE IN
     {SELECT SKILCODE FOR LASTNAME="LEHR" THRU IEMP,POSSESS}
     THRU IEMP, POSSESS
```

A nested SELECT command can have many relational expressions in its conditional clause. Furthermore, several SELECT commands can be nested into the same LIST command. Suppose we need a list of all employees possessing all the skills that are needed for job code 5. The command

```
LIST SKILCODE FOR JOBCODE=5 THRU IJOB,NEEDS
```

yields a list of the skills needed by job 5. Suppose that these skilcodes are 9 and 13. Then the desired employee list is produced by[1]:

```
LIST ID FOR ID IN {SELECT ID FOR SKILCODE=9 THRU ISKL,>POSSESS}
           ID IN {SELECT ID FOR SKILCODE=13 THRU ISKL,>POSSESS}
           THRU IEMP
```

To generate a list of employees possessing any of the skills needed by job 5, the appropriate LIST command[1] is:

```
LIST ID,SKILCODE FOR SKILCODE IN
     {SELECT SKILCODE FOR JOBCODE=5 THRU IJOB,NEEDS}
     THRU IEMP,POSSESS
```

This report also contains, for each employee, a list of the skills needed by job 5 which that employee possesses.

Note that macros can be defined for frequently used SELECT commands. For instance, the nested SELECT command:

```
{SELECT JOBCODE FOR LASTNAME="Lehr" THRU IEMP,>FILLEDBY}
```

could be replaced with macros. If LNAM is defined to be a macro for {SELECT JOBCODE FOR LASTNAME= and JOBPATH is a macro for THRU IEMP,>FILLEDBY}, then

```
LNAM "Lehr" JOBPATH
```

is equivalent to the above SELECT command.

---

[1] Recall that a QRS command is terminated by pressing the carriage return key.

Restrictions:

SELECT commands can be nested within the conditional clause of a SELECT command. This nesting of SELECT commands within SELECT commands cannot exceed a depth of 6 levels.

Use of nested queries requires significant amounts of memory. To reduce this memory requirement, a user of SELECT commands (i.e., nested queries) should attempt to formulate the nesting to minimize the data selected.

This page intentionally left blank.

XI. STARTUP FILE

A. Overview

A QRS user can begin a query session as described in Chapter I, Section B. Alternatively, a STARTUP file can be created which specifies the file name of the main data base area, user name, user password, and/or a series of QRS commands. When QRS is executed, its first activity is to read a file named STARTUP if one exists. If there is none, QRS looks for a file named STARTQ and treats it as the startup file if it exists. If the main area file name, user name, or user password is absent from the STARTUP file, the QRS user is prompted for the missing value(s). Then, all QRS commands on the STARTUP file are executed. When these commands have been executed, the QRS --> prompt is displayed.

When QRS commands are used in a STARTUP file, they are typically a series of set commands. These are used to initialize environment parameters and options to something other than their defaults. Of course, environment parameters and options can be altered with the set command at any time during a query session. In addition to set commands, any other kind of QRS command can appear in the STARTUP file.

B. Creating the STARTUP file

The STARTUP file can be created, edited, and saved by using the MDBS.DDL software or any other standard text editing facilities. The approach used is similar to that used for creating an input file of QRS commands for the READ command (Chapter VIII).

The first line of the STARTUP file must begin with the word **START.** A file name (for a data base's main area), user name and/or user password can optionally be specified on this line. If present, they must appear in this order. For example:

```
START "JOBS.DB" "D LEHR" "smiles"
```

Here JOBS.DB is the fully qualified name of the file containing the main data base area, D LEHR is a user name, and "smiles" is the password. If any of these arguments is omitted from the first line of STARTUP, the user is prompted for the missing value(s). If the user fails to provide appropriate values, QRS returns control to the operating system. Each subsequent line of the STARTUP file can consist of any valid QRS command. Command lines in the STARTUP file are echoed to the console if environment option 5 is on.

The STARTUP file must reside on the currently logged-on drive/directory and its name must be consistent with the operating system under which the QRS user is working.

In many operating systems, startup information can be created on files other than the STARTUP file. In this case, the startup information on such a file can be used in place of the STARTUP file by entering the file name on the operating system command line that is used to begin the execution of QRS. See the pertinent system specific manual for details.

XII. QRS ERROR MESSAGES

***** Alternate input files may not be nested.**

Explanation:
A file cannot be read from within another file which is being read.
Possible Causes and/or Solutions:
1. A 'READ' command has been used on a file which itself contains a 'READ' command.

***** Argument required.**

Explanation:
A parameterized macro must be followed by a matching pair of parentheses.
Possible Causes and/or Solutions:
1. Parentheses were omitted following a parameterized macro.
2. A space was left between the last letter of the macro name and the left parenthesis.

***** Catastrophe.**

Explanation:
The integrity of the data base has been destroyed.
Possible Causes and/or Solutions:
1. See DMS errors 100-199.
2. It is also possible that during overlay execution, a problem was encountered, or an overlay file was not present.

***** Column headings too long.**

Explanation:
The column headings specified (not including item names) together exceed a maximum of 80 characters.
Possible Causes and/or Solutions:
It may be possible to reduce the number of column headings and use item names as headings.

***** Conditional clause must contain a boolean.**

Explanation:
The conditional clause must contain an expresison which evaluates to either TRUE or FALSE.
Possible Causes and/or Solutions:
1. Conditional clause must contain a Boolean expression formed from one or more of the following: EQ(=), NE(≠), LT(<), GT(>), LE(≤), GE(≥), IN.
2. Check conditional clause specification in query command.

***** Data base full.**

Explanation:
There is insufficient room to store the user-defined commands or value label definitions.
Possible Causes and/or Solutions:
1. There is insufficient room in the main data base area to store the macro or value label being defined.
2. More room should be made available by freeing up space on the main data base area disk.
3. It is time to consider enlarging the data base.

***** Data dictionary not accessible.**

Explanation:
The data dictionary cannot be accessed with the DISPLAY command.
Possible Causes and/or Solutions:
1. The data base was initialized with the **o** option.

***** Define already in progress.**

Explanation:
In a multi-user system, the define command has already been issued by another user.
Possible Causes and/or Solutions:
1. This error message would be obtained in a multi-user system only.
2. Two users of the system cannot use the define command at the same time.

***** Enclose file name in quotes.**

Explanation:
Because of non-alphanumeric characters which are allowed for file names, for many operating systems, QRS requires that file names be in quotes.
Possible Causes and/or Solutions:
The file name specified must be enclosed in quotes.

***** File name too long.**

Explanation:
The file name specified is longer than the maximum size of a file name permitted by the DMS for the user's operating system.
Possible Causes and/or Solutions:
1. The file name specified is longer than the maximum length of a file name, under the given operating system.
2. Check file name conventions under the user's operating system.

***** File not present.**

Explanation:
The file specified is not present on the disk.
Possible Causes and/or Solutions:
The system could not find the file indicated in the READ command.

***** Illegal character in macro.**

Explanation:
A value specified for a parameterized macro contained an illegal character.
Possible Causes and/or Solutions:
An unquoted value can contain only alphanumeric characters or special characters: " [ ] . / * + - < > = # $. Any other characters (including spaces and commas) are illegal.

***** Illegal item type for value label.**

Explanation:
The item specified for a value label definition is not of a valid type.
Possible Causes and/or Solutions:
1. The item specified in a value label definition must be of one of the following types: string, character, integer, unsigned.
2. Check item specifications using the DISP I command.

***** Improper conditional clause.**

Explanation:
The conditional clause has an invalid syntax.
Possible Causes and/or Solutions:
The conditional clause does not end with the keyword 'THRU' (refer to Appendix A for list of keywords).

***** IN operator must occur within conditional clause.**

Explanation:
Self-explanatory.
Possible Causes and/or Solutions:
The IN operator must be preceded by the word FOR.

***** Insufficient room in memory.**

Explanation:
There is not enough room in memory to carry out any further processing by the query system.
Possible Causes and/or Solutions:
1. There is not enough room in memory to open the data base.
2. If the data base has already been opened, there is insufficient memory to process the given query.
3. Turn off options 21 through 27 or turn on option 28.

***** Invalid area name**

Explanation:
Self-explanatory.
Possible Causes and/or Solutions:
1. The area name specified in an OPEN or CLOSE command is not a valid area, as defined in the DDL specifications.
2. Check list of valid areas using the DISP A command.

***** Invalid BY usage.**

Explanation:
The break clause has not been properly specified.
Possible Causes and/or Solutions:
1. The break clause is not followed by the conditional clause or the path clause.
2. The break clause cannot be used with STAT or WRITE.

***** Invalid calculation.**

Explanation:
The operands specified in the COMPUTE mode are not valid or meaningful.
Possible Causes and/or Solutions:
1. Something other than a number has been specified as an operand in the COMPUTE mode.

***** Invalid character.**

Explanation:
The indicated field contains an invalid character.
Possible Causes and/or Solutions:
1. An invalid character has been specified within a character field.
2. Check character field specifications in query command.

***** Invalid DISPLAY parameter.**

Explanation:
The parameter specified is not a valid DISPLAY parameter.
Possible Causes and/or Solutions:
The parameter specified is not one of a, c, e, i, o, r, s.

***** Invalid file operation.**

Explanation:
The file name is incorrect or there is insufficient room in memory to use the file.
Possible Causes and/or Solutions:
1. The file name is not alphanumeric or quoted.
2. The file name is too long.
3. There is insufficient room in memory to allocate the file buffer (needed by a SET FN or READ command). Use the -b option as described in the pertinent system specific manual.

***** Invalid IN usage.**

Explanation:
An improper syntax has been used for the IN operator.
Possible Causes and/or Solutions:
1. The IN clause does not begin or end with a bracket ('[' or ']').
2. A $ path clause is present within the brackets.
3. Check proper specification of IN clause.

***** Invalid label length.**

Explanation:
A label of length zero or a negative label length has been defined.
Possible Causes and/or Solutions:
1. A label of length zero has been defined for the item specified. The label length must be positive.
2. Redefine label length.

***** Invalid macro definition.**

Explanation:
The text of the macro being defined does not obey the rules of macro definition.
Possible Causes and/or Solutions:
1. Macro parameters are not being used in a valid way (e.g., &1 is not the first referenced parameter in the text).
2. See Chapter VII.

***** Invalid match syntax.**

Explanation:
The character class specification is not valid.
Possible Causes and/or Solutions:
A ']' has not been specified in the character-class specification in the conditional clause.

***** Invalid number.**

Explanation:
The indicated field contains an invalid number.
Possible Causes and/or Solutions:
1. An invalid character has been specified within a numeric field.
2. Check numeric field specifications in query command.

***** Invalid record type.**

Explanation:
The record type specified in a DISPLAY command is not valid.
Possible Causes and/or Solutions:
1. The record type specified in a DISP R "record type name" command is not valid.
2. Check possible record type names using a DISP R command.

***** Invalid record type name.**

Explanation:
An invalid record type name has been specified while defining value labels.
Possible Causes and/or Solutions:
1. Check record type names (by using DISP R).

***** Invalid set name.**

Explanation:
Self-explanatory.
Possible Causes and/or Solutions:
1. A set name which is not the same as or synonymous with any of the sets specified through the DDL has been used in a query command.
2. Check set name specifications in path clause of query.
3. Check possible set names through use of DISP S command.

***** Invalid SET usage.**

Explanation:
The SET command has an improper syntax.
Possible Causes and/or Solutions:
1. If an 'option' is used with the SET command, a number does not follow the word 'OPTION'.
2. An environment parameter, option or heading list does not follow the word 'SET'.

***** Invalid startup specification.**
Data base could not be opened.

Explanation:
The first line of the startup file has an improper specification.
Possible Causes and/or Solutions:
1. The first line of the startup file does not start with 'START'.
2. The file name, user and password specifications in the startup file have non-alphanumeric characters, or their length is greater than the maximum allowable length.

***** Invalid syntax in path clause.**

Explanation:
The path clause has been improperly specified.
Possible Causes and/or Solutions:
A delimiter other than '<', '>', ',', '-', or a blank has been specified in the path clause.

***** Item "data item name" of record type "record type name" is not within the specified path.**

Explanation:
The path specified does not reference the indicated item in the specified record type.
Possible Causes and/or Solutions:
Incorrect path clause specification. Make sure that the stated path touches the record type containing the item name specified.

***** Item "data item name" is not within the specified path.**

Explanation:
The path specified does not reference the indicated item.
Possible Causes and/or Solutions:
Incorrect path clause specifications. Make sure that the stated path touches the record type containing the item name specified.

***** Item "item-name" is not in record type "record type name".**

Explanation:
Self-explanatory.
Possible Causes and/or Solutions:
1. The item specified is not in the indicated record type.
2. Check item specifications using the DISP R "record type name" command.

***** Item name not unique within path.**

Explanation:
An item name has been specified in the query which is not unique within the path specified.
Possible Causes and/or Solutions:
1. The same item name appears in two or more record types within the path specified.
2. The specified path touches a given item more than once.
3. Preface item names with names of the record types.
4. Use synonyms for the item name(s).

***** Item not in record type.**

Explanation:
The item specified in the value label definition is not part of the record type indicated.
Possible Causes and/or Solutions:
1. Check item names for the indicated record type by using DISP R (followed by the record type name).

***** Label too long.**

Explanation:
The label for the indicated item in the value label definition is too long.
Possible Causes and/or Solutions:
The label for the indicated item in the value label definition is longer than the maximum length specified earlier.

***** Macro already defined.**

Explanation:
A macro having the indicated name has been defined earlier.
Possible Causes and/or Solutions:
1. A macro having the name indicated in the (A) option has already been defined by the user.
2. Check list of user-defined commands (macros) via (L) or (I) option.

***** Macro not defined.**

Explanation:
The macro having the indicated name has not already been defined.
Possible Causes and/or Solutions:
1. The macro indicated in the (C), (D) or (L) option has not already been defined.
2. Check macro definitions via (L) or (I) option.

***** Maximum macro length exceeded.**

Explanation:
The length of the macro (i.e., name of macro together with all the text defined for that macro) exceeds 255 characters.
Possible Causes and/or Solutions:
1. Redefine macro. If a macro longer than 255 characters is desired, the macro must be broken up into two shorter macros.

***** Mismatched parenthesis.**

Explanation:
Improper use of parenthesis in query command.
Possible Causes and/or Solutions:
An uneven number of '(' and ')' have been specified.

***** Name too long.**

Explanation:
The name of the item or record type specified is too long.
Possible Causes and/or Solutions:
1. The name of an item or record type used in a value label definition is longer than 8 characters.
2. Check item and record type specifications using the DISPLAY command.

***** Nested query may have only one term.**

Explanation:
Only one term can be specified in the find clause of a nested query.
Possible Causes and/or Solutions:
1. More than one term (which may be an expression involving one or more items) has been specified in the find clause of a nested query.
2. Only one term can be specified in a nested query. Use a separate nested query for each term desired.
3. No term was specified or the conditional clause is not syntactically valid.

***** Nested query must use SELECT command.**

Explanation:
A query has been nested, and the nested query does not start with 'SELECT'.
Possible Causes and/or Solutions:
A nested query does not begin with the word 'SELECT'.

***** No items specified.**

Explanation:
No items have been specified following a list, write, spew or stats command.
Possible Causes and/or Solutions:
1. A list, write, spew or stats command does not have any find clause.
2. The specified find clause does not involve any information from the data base. Each find clause must involve some data base retrieval.

***** No output file defined.**

Explanation:
No file has been specified to which the output of a query can be written.
Possible Causes and/or Solutions:
A file to which query output can be written has not been defined using the "SET FN file-name" command.

***** No path clause found in query.**

Explanation:
A query command has been specified without a path clause.
Possible Causes and/or Solutions:
A path clause is required in every query command. The path clause is designated by the keyword THRU.

***** No sets specified.**

Explanation:
The path clause contains no sets.
Possible Causes and/or Solutions:
1. One or more sets must be specified in the path clause of the query.
2. Check scnema for proper specification of sets in path clause.

***** Operator-operand mismatch.**

Explanation:
There is an inconsistency or mismatch in the operator/operand specification.
Possible Causes and/or Solutions:
1. A numeric field has been specified where the system was expecting a character field.
2. A character field has been specified where the system was expecting a numeric field.
3. The operands specified are inconsistent with the operator specified (e.g., attempting to add two strings).

***** Output length exceeded.**

Explanation:
The sum of the widths of the terms in the find clause exceeds the console/printer width.
Possible Causes and/or Solutions:
1. The console/printer width specified is too small to hold the terms listed in the query.
2. Reduce number of items in the find clause or increase the console/printer width.

***** Query has no output data.**

Explanation:
The system cannot find any record occurrences when it processes the query.
Possible Causes and/or Solutions:
1. A system-owned set specified in the path clause has no members.
2. It may be possible to obtain valid query output by specifying a different path.

***** Sorted output may not have breaks.**

Explanation:
No control breaks can be specified when the output is being sorted.

Possible Causes and/or Solutions:
1. A break clause has been specified when the query output is being sorted (using option 14).
2. Either restate the query without control breaks or turn off option 14.

***** Syntax error-command expected.**

Explanation:
The system was expecting a command name and encountered a non-alphanumeric cnaracter within the command name.

Possible Causes and/or Solutions:
A character within the command name was not alphanumeric.

***** Too many (few) arguments.**

Explanation:
The number of values specified for a parameterized macro was too many (or too few).

Possible Causes and/or Solutions:
1. A value followed by a comma or a comma (for optional value omission) must be specified for each of the macro's parameters.
2. A leading comma, a trailing comma, or two consecutive commas were not used to indicate the omission of an optional (leading, embedded, or trailing) value.

***** Too many items specified.**

Explanation:
The number of items specified in the query exceed the maximum number of items permitted.

Possible Causes and/or Solutions:
1. The number of items used in the query command exceeds 20, the maximum allowable number of items.

***** Too many operands/operators.**

Explanation:
The system cannot process the given query. The query uses too many operands or operators to be handled internally.
Possible Causes and/or Solutions:
1. The number of operators/operands in the query make it too complex to be processed by the system.
2. Regroup and/or reorder the operators/operands in the query command.
3. Utilize additional parentheses to indicate precedence.

***** Too many terms in query.**

Explanation:
The number of terms in the query exceeds the maximum permissible number of terms.
Possible Causes and/or Solutions:
The number of terms (which may be expressions involving several items) exceeds 15, which is the maximum number of terms permissible.

***** Unconnected set in path "path".**

Explanation:
A completely connected line cannot be drawn through all the sets specified within the path clause (see the path clause description in Chapter III).
Possible Causes and/or Solutions:
Consult the schema drawing to insure a completely connected path is specified.

***** Unmatched quotes.**

Explanation:
Self-explanatory.
Possible Causes and/or Solutions:
Quotes have been improperly specified or improperly matched.

***** Unrecognized command.**

Explanation:
The command specified is not a valid QRS command.
Possible Causes and/or Solutions:
1. The command specified is other than one of the twelve standard QRS commands, and is not a user defined command (macro).
2. Check command specification for bad (non-alphanumeric) character.
3. Turn on option 12 to view the query actually being examined.

***** Value labels already defined.**

Explanation:
The value label for the indicated item has already been defined.
Possible Causes and/or Solutions:
1. The value label specified in the (A) option for the indicated item has been defined earlier.
2. Check list of value label definitions via (L) or (I) option.

***** Value labels not defined.**

Explanation:
The value label specified has not been defined earlier.
Possible Causes and/or Solutions:
1. The value label specified in the changed (C), delete (D) or list (L) option has not been defined earlier.
2. Check list of value label definitions via (L) or (I) option.

***** Value too long.**

Explanation:
The value of the item specified in the value label definition is too long.
Possible Causes and/or Solutions:
1. The value of the item specified in the value label definition is longer than the length in the DDL specification for that item.
2. Check item specifications using the DISP I command.

MDBS QRS MANUAL

Appendix A

List of QRS Keywords

Appendix A

AND
BY
BYE
CLOSE
COMPUTE
DATFORM
DBSTAT
DEFINE
DISP
DISPLAY
ECHO
EDIT
EQ
FOR
FROM
GE
GT
HELP
IN
LE
LIST
LT
NE
NOT
OPEN
OR
QUIT
READ
SELECT
SET
SPEW
STATS
THRU
WRITE
XOR

MDBS QRS MANUAL

This page intentionally left blank.

MDBS QRS MANUAL

Appendix B

Decimal-ASCII Converstion Table

Appendix B

Decimal Number	ASCII Equivalent	Decimal Number	ASCII Equivalent	Decimal Number	ASCII Equivalent
0	NUL	49	1	98	b
1	Cntrl-A	50	2	99	c
2	Cntrl-B	51	3	100	d
3	Cntrl-C	52	4	101	e
4	Cntrl-D	53	5	102	f
5	Cntrl-E	54	6	103	g
6	Cntrl-F	55	7	104	h
7	Cntrl-G	56	8	105	i
8	Cntrl-H	57	9	106	j
9	Cntrl-I	58	:	107	k
10	Cntrl-J	59	;	108	l
11	Cntrl-K	60	<	109	m
12	Cntrl-L	61	=	110	n
13	Cntrl-M	62	>	111	o
14	Cntrl-N	63	?	112	p
15	Cntrl-O	64	@	113	q
16	Cntrl-P	65	A	114	r
17	Cntrl-Q	66	B	115	s
18	Cntrl-R	67	C	116	t
19	Cntrl-S	68	D	117	u
20	Cntrl-T	69	E	118	v
21	Cntrl-U	70	F	119	w
22	Cntrl-V	71	G	120	x
23	Cntrl-W	72	H	121	y
24	Cntrl-X	73	I	122	z
25	Cntrl-Y	74	J	123	{
26	Cntrl-Z	75	K	124	|
27	ESC	76	L	125	} (ALT MODE)
28	FS	77	M	126	~
29	GS	78	N	127	DEL (RUB OUT)
30	RS	79	O		
31	US	80	P		
32	SP	81	Q		
33	!	82	R		
34	"	83	S		
35	#	84	T		
36	$	85	U		
37	%	86	V		
38	&	87	W		
39	'	88	X		
40	(	89	Y		
41	)	90	Z		
42	*	91	[		
43	+	92	\		
44	,	93	]		
45	-	94	^ (^)		
46	.	95	_ (<)		
47	/	96			
48	0	97	a		

MDBS QRS MANUAL

INDEX

application development 2
areas 5,61
arithmetic expressions 1,12,31
Boolean expressions 1,13,15-16
break clause 11,19,27
case comparisons 39,42
character classes 14-15
CLOSE command 3,61
column headings 5,19,41,44
command length 4
commands 1-3
COMPUTE command 3,31-32
conditional clause 1,11,13-16,20,63
console control 34-35,40-42
control breaks 1,11,19
data base initialization 5
data dictionary 2,5-6,61
data items 5,12-13
data security 1,6
DEFINE command 3,51-58
DISPLAY command 2,5-9
environment options 2,3,5,33,39-44,59
environment parameters 2,3,5,33-39,59
error messages 69-82
find clause 11-13,20
fine tuning 1,18,42-44
FN parameter 27-28,36
FOR clause
(see conditional clause)
group of constants 14
IN operator 13,63
installation 2,67-68
keywords 3
LIST command 2,11,26-27,31,63
loops 17
macros
(see user-defined commands)
mailing labels 13
match-one 1,14,36
notational conventions 4
OF parameter 13,37
OPEN command 3,61
operator precedence 12,15-16
option 5 40,59,67
option 14 19,39,42
option 20 18,39,42
options 21-27 19-20,28,39,42-43
page title 19,39
parameterized macros 58-58.2
path clause 1,11,16-18
printer control 37-38,40-41
prompt 2,62,67
query nesting 63-65
QUIT command 3,61,62
READ command 3,40,59
record access count 42
record types 5
relational expression 13-15,63
relational oerators 13-14
repeating data items 20
report formatting 33-49
retrieval query 11
scale factor 38
SELECT command 63-65
SET command 3,5,19,27,33-49
sets 5,16-17
sorting 19
special format output 27-28
SPEW command 2,27-29,31,36
STARTUP file 2,67-68
statistics 1,3,19-20,42-44
STATS command 3,27-29,31
synonyms 3,12,51
SYSTEM-owned set 16
term 12-14,19,41,63
THRU clause
(see path clause)
TL parameter 19,39
user-defined commands 1,3-4,41,
51-52,54-55
58,58.2
value labels 1,3,41,51,53,56-58
virtual table 11
wildcard 1,14,36
WRITE command 2,27-29,31,36,41

DOCUMENTATION COMMENT FORM

MDBS Document Title:_____

We welcome and appreciate all comments and suggestions that can help us to improve our manuals and products. Use this form to express your views concerning this manual.

Please do not use this form to report system problems or to request materials, etc. System problems should be reported to **MDBS** by phone or telex, or in a separate letter addressed to the attention of the technical support division. Requests for published materials should be addressed to the attention of the marketing division.

Sender:

_____ _____
(name) (position)

_____ _____
(company) (telephone)

_____
(address)

_____
(city, state, zip)

COMMENTS:

Areas of comment are general presentation, format, organization, completeness, clarity, accuracy, etc. If a comment applies to a specific page or pages, please cite the page number(s).

_____

_____

_____

_____

_____

_____

_____

_____

Continue on additional pages, as needed. Thank you for your response.