```
@@@@@  @   @  @@@@@  @@@@   @@@@@   @@@@
  @    @@  @  @      @   @  @      @
  @    @@  @  @      @   @  @      @
  @    @ @ @  @  @@  @@@@   @@@@    @@@
  @    @  @@  @   @  @ @    @          @
  @    @  @@  @   @  @  @   @          @
@@@@@  @   @   @@@@  @   @  @@@@@  @@@@
```

R E F E R E N C E   M A N U A L

Memorandum No. ERL-M519

23 April 1975

by

William Zook
Karel Youssefi
Peter Kreps
Gerald Held
James Ford

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Research sponsored by the National Science Foundation Grant GK-43024x, U.S. Army Research Office -- Durham Contract DAHCO4-74-G0087, the Naval Electronic Systems Command Contract N00039-75-C-0034, and a Grant from the Sloan Foundation.

This manual is written in the same spirit as the UNIX Programmer's Manual, and admits of the same goal: that of providing the user not with a statement of ultimate goals or ideals, but rather with a functional description of the current implementation. The manual is organized so as to present in a concise form the information needed by any user of the INGRES system, and to permit easy updating of the document itself as the system evolves. It serves, in other words, as a compendium of information describing the current capabilities and restrictions of INGRES.

Each entry in the manual consists of a page or so of information which describes a single part of INGRES. Each entry has one or more of the following sections:

NAME section
This section repeats the name of the entry and gives an indication of its purpose.

SYNOPSIS section
This section indicates the form of the command (statement). The conventions which are used are as follows:

Upper case names are used to indicate reserved words. When entering actual INGRES commands, these words need not be typed in upper case.
Lower case words indicate generic types of information which must be supplied by the user. In the DESCRIPTION section the legal values for these names are described.
Square brakets ([ ]) are used to indicate an optional field.
Ellipses (...) are used to indicate that the previous field may be repeated several times.

DESCRIPTION section
This section gives a detailed description of the entry with references to the reserved words and generic names used in the SYNOPSIS section.

EXAMPLE section
This section gives one or more examples of the use of the entry. Most of these examples are based on the two relations:

```
emp(name,sal,mgr,bdate)
and
newemp(name,sal,age)
```

SEE ALSO section
This section gives the names of entries in the manual which

are closely related to the current entry.

DIAGNOSTICS section
This section describes error messages and warning diagnostics specific to the particular command, and which may not be completely transparent to the user.

BUGS section
This section indicates known bugs or deficiencies in the command.

To get started, the section entitled INGRES describes how to log in to the INGRES system and how to submit a query. The section entitled QUEL describes the basic syntax of the query language; other sections of this manual assume a familiarity with this description.

NAME
append - append tuples to a relation

SYNOPSIS
```
APPEND [TO] relname (target_list) [WHERE qual]
```

DESCRIPTION
Append adds tuples which satisfy the qualification to the relation relname. Relname must be the name of an existing relation. The target_list specifies how new tuples are to be assembled. All of the attribute names of the relation must appear in the target_list as result_attnames either explicitly or by default (see QUEL).

Values or expressions of any numeric type may be used to set the value of a numeric type domain, with conversion to the result domain type taking place. (Exception - see below)

EXAMPLE
```
/* Make the new employee Jones work for Smith */
   range of n is newemp
   append to emp(n.name,n.sal,mgr="Smith",bdate=1975-n.age)
                where n.name = "Jones"
```

SEE ALSO
quel, retrieve

DIAGNOSTICS
Use of a numeric type expression to set a character type domain or vice versa will produce diagnostics.

BUGS
Append does not work if a variable in the target list or the qualification ranges over the result relation.

If the result relation is an unsorted table (which all results currently are), then duplicate tuples caused by append are NOT removed.

Conversion is NOT done when an F4 or F8 expression is used to set the value of an I4 domain. Such an append will presently cause error termination.

NAME

copy - copy data into (from) a relation from (into) a UNIX file.

SYNOPSIS

```
COPY relname(attname = format[, attname = format ...] )
     direction filename
```

DESCRIPTION

Copy is used to move data between INGRES and standard UNIX files. The relation specified by relname must exist and each of the attribute names in the relation must appear in the list of attnames. For each attname, the associated format indicates the format of the domain as it appears (or is to appear) in the UNIX file. The order of the attnames must be the same as the order that their values appear in the UNIX file. The format is of the form Xn where X is a character (c, i, or f) and n is a number (0-255). The direction must be either INTO or FROM.

When the direction is FROM, copy transfers data into the relation FROM the UNIX file. For data transfer in this direction, the formats have the following meanings:

i1,i2,i4 - The data for the related attribute is stored as an integer of length 1, 2, or 4 bytes in the UNIX file.

f4,f8 - The data for the related attribute is stored as a floating point number (either single or double precision).

c0 - The data for the related attribute is stored as a variable length string of characters. This format is used for reading attributes of type i or f. The character string may consist of numbers, a minus sign, and a decimal point. The string is terminated by any other character. The terminating character is ignored (not used in the next field).

c1,c2,c3,...,c255 - The data is stored as a fixed length string of characters. If the corresponding attribute is of numeric type (i or f) then the characters are converted to the proper type. If the attribute is of type character, the data is simply copied.

EXAMPLE

```
/* Copy data into the emp relation */
   copy emp(name=c10,sal=f4,bdate=i2,mgr=c10) from "myfile"
```

SEE ALSO
     create

DIAGNOSTICS

BUGS
     Copy into a domain with format i4 does not work.

NAME

create - create a new relation

SYNOPSIS

```
CREATE relname(attname = format[, attname = format ...] )
```

DESCRIPTION

Create will enter a new relation into the data base. The name of the relation is relname and the domains are named attname1, attname2, etc. The domains are created with the type specified by format. Formats are of the form Xn where X is a chareacter (i, f, or c) and n is an integer (1 to 255). The type of the domain may be integer (i), floating (f), or character (c). and the length of the domain may be 1,2 or 4 for integers; 4 or 8 for floating; or 1 to 255 for characters.

The relation is created with no data initially in it.

EXAMPLE

```
/* Create relation emp with domains name, sal and bdate */
   create emp(name = c10, salary = f4, bdate = i2)
```

SEE ALSO

destroy, copy

DIAGNOSTICS

BUGS

NAME
delete - delete tuples from a relation

SYNOPSIS
DELETE tuple_variable [WHERE qual]

DESCRIPTION
Delete removes tuples which satisfy the qualification from the relation that they belong to. The tuple_variable must be a variable which has been declared in a range statement to range over an existing relation. Note that delete does not have a target_list. Also, if the qualification is not given, the effect is to delete all tuples in the relation. The result is a valid, but empty relation.

EXAMPLE
```
/* Remove all employees who make over $30,000 */
   range of e is emp
   delete e where e.sal > 30000
```

SEE ALSO
quel, range, destroy

DIAGNOSTICS

BUGS

NAME

destroy - destroy an existing relation

SYNOPSIS

```
DESTROY relname
```

DESCRIPTION

Destroy removes a relation from the data base. Only the relation owner or the data base administrator may destroy a relation. A relation may be emptied of tuples but not destroyed using the delete statement.

EXAMPLE

```
/* Destroy the emp relation */
   destroy emp
```

SEE ALSO

create, delete

DIAGNOSTICS

BUGS

NAME
    help - get information about how to use INGRES

SYNOPSIS
    HELP ["item-in-question"]

DESCRIPTION
    HELP may be used to obtain information about any section of this manual, the content of the current data base, or a specific relation in the data base, depending on the item-in-question. Omission of that argument is functionally equivalent to HELP "help" . The other legal forms are as follow:

    HELP "section" - Produces a copy of the specified section of the INGRES Programmer's Manual, and prints it on the standard output device.

    HELP "" - Gives information about all relations that exist in the current database.

    HELP "relname" - Gives information about the specified relation, but in greater detail than would HELP "" .

EXAMPLE
```
HELP
HELP "quel"
HELP ""
HELP "emp"
```

SEE ALSO

DIAGNOSTICS
    Unknown name - The item-in-question could not be recognized.

BUGS
    Alphabetics appearing within the item-in-question must be lower-case to be recognized.

NAME

ingres - INGRES relational data base management system

SYNOPSIS

INGRES [-options...] data_base_name

DESCRIPTION

This is the UNIX command which is used to invoke INGRES. Data_base_name is the name of an existing data base. The options, if selected, have the following effects:

-v : Print out additional (probably profuse and possibly incomprehensible) debugging information.

-s : Do not print the login dayfile.

-m : Invoke the terminal monitor as the input interface.

-c : Invoke the C-QUEL interface instead of the monitor.

-e : Invoke the error-correcting precompiler and interactive debugging facility.

Where conflicting options are present, the last one recognized will be enforced. The default options are equivalent to -m .

EXAMPLE

```
ingres demo
ingres -m -s demo
```

SEE ALSO

monitor

DIAGNOSTICS

Cannot open data base xxx : It was not possible to log you in to the speficied data base.

Out of space : This error is symptomatic of a query so long that its processing required more memory than the system could provide. Try splitting the query into smaller chunks.

BUGS

There shou'd be a way to abort a running INGRES command and return to the INGRES monitor level (i.e. the way del (rub out) works in UNIX.) Currently del causes INGRES to exit entirely.

At most 10 options may be present in the command list.

The presence of anything following the data_base_name may cause fatal execution errors later on -- such arguments should be ignored.

NAME

monitor - interactive terminal monitor

DESCRIPTION

This is the process which is typically used to submit queries to the INGRES data management system. As characters are typed on the terminal, they are entered into a *virtual* query buffer, where they can be edited at any time. The three characters '#', '@', and '\' are used to indicate to the monitor that one of the following actions are to be taken:

# :	Erase the previous character. Successive uses of this command will erase back to, but not beyond, the beginning of the current line.
@ :	Erase the current line. Successive uses of this command are ignored.
\r :	Erase the entire query (reset the query buffer). The former contents of the buffer are irretrieveably lost.
\p :	Print the current query. The contents of the buffer are printed on the user's terminal.
\e :	Enter the UNIX text editor (see ED in the UNIX Programmar's Manual); use the ED command 'w' followed by 'q' to return to the INGRES monitor.
\g :	Process the current query (go). The contents of the buffer are transmitted to the parser and run.
\q :	Exit from INGRES.
\ :	Ignore special meaning of following character. This character must preceed either of (#, @) in order to escape the editing effects of those characters. (see QUEL - strings).

SEE ALSO

ingres, quel

DIAGNOSTICS

go : You may begin a fresh query.

continue : Further input will be appended to the end of the last query.

>>ED : You have entered the UNIX text editor.

MONITOR<< : You have left the text editor, and have returned to INGRES.

BUGS

Only the first 512 bytes of the dayfile are printed.

Use of del (a.k.a. rubout) to exit from the text editor WILL kill INGRES, but WILL NOT kill the editor. Both should ignore that signal.

NAME
print - print a relation

SYNOPSIS
PRINT relname

DESCRIPTION
Print displays the entire relation specified by relname on the terminal (standard output).

EXAMPLE
```
/* Print the emp relation */
   print emp
```

SEE ALSO
retrieve

DIAGNOSTICS

BUGS
There is no way to stop output in the middle of printing.

Print does not handle long lines of output correctly - no wrap around.

Print should have more formating features to make printouts more readable.

Print should have an option to print on the line printer.

NAME

quel - QUEry Language for INGRES

DESCRIPTION

The following is a description of the general syntax of QUEL. Individual QUEL statements are treated seperately in this document; this section restricts itself to the constituent parts common to all QUEL expressions.

names

Names in QUEL are sequences of no more than 12 alphanumeric characters, starting with an alphabetic. Underscore (_) is considered an alphabetic. All upper-case alphabetics appearing within names are automatically and silently mapped into their lower-case counterparts.

strings

Strings in QUEL are sequences of no more than 255 arbitrary ASCII characters bounded by quotes (" "). Upper case alphabetics within strings are accepted literally. Also, in order to imbed quotes and new-line characters within strings, it is necessary to prefix them with \ . The same convention applies to \ itself.

numbers

Numeric constants in QUEL range from approximately 10**38 to -10**38, with a floating point precision of at most 17 decimal digits.

operators

Arithmetic operators include

```
        +   (unary plus or addition)
        -   (unary minus or subtraction)
        *   (multiplication)
        /   (division)
        **  (exponentiation)
```

These operators group left-to-right (except for unary operators, which group right-to-left). Moreover, their precedence (in descending order) is

```
        1.  unary operators
        2.  exponentiation
        3.  multiplicative operators
        4.  additive operators
```

Parentheses may be used to arbitrarily override the implied precedence of operators.

Relational operators recognized by QUEL include

```
        <   (less than)
```

```
          <=  (less than or equal)
          >   (greater than)
          >=  (greater than or equal)
           =  (equal to)
          !=  (not equal to)
They are all of equal precedence.
```

Logical operators group left-to-right:

```
          and  (logical AND; conjunction)
          or   (logical OR; disjunction)
          not  (logical NOT; negation)
```

NOT has the highest precedence of the three.

Functional operators currently supported include

```
          atan(f) - arctangent function
          cos(f) - cosine function
          gamma(f) - log gamma function
          log(f) - natural logarithm
          mod(a,b) - a modulo b
          rand - pseudo-random number generator
                 in the range 1 - (2**15 -1)
          sin(f) - sine function
          sqrt(f) - square root function
```

Where f appears as an argument, a floating point constant is expected; otherwise, an integer should be used.

Aggregation operators include

```
          count - count of occurences
          sum   -  summation
          avg   -  average (sum/count)
          max   -  maximum
          min   -  minimum
```

attribute

Attributes are expressions of the form

```
          variable_name.attribute_name
```

where variable_name is a tuple_variable (see RANGE) and attribute_name is the name of a domain or attribute of a relation.

a-fcns

An a-fcn (or attribute function) is an expression consisting of

```
          1.  an aggregate,
          2.  a simple attribute,
          3.  a constant,
          4.  a functional operator evaluating to any
              of (1. - 3.),
       or 5.  an arithmetic expression involving any
              of (1. - 4.) and arithmetic operators.
```

aggregate

An aggregate consists of an aggregation operator followed by a parethesized expression containing a single a-fcn and (possibly) a qualification. Current restrictions on the use of aggregate operators with a-fcn's of different types are:

I2	all aggregate operators permitted.
I4,F4	count and avg only.
F8	count, sum, and avg only.
Cn	count, min, and max only.

agg-fcn

Agg-fcns (or aggregate functions) are presently unimplemented

target-list

A target list is a parenthesized, comma separated list of one or more elements , each of which must be of one of the following forms:

1. result_attname IS a-fcn

Where result_attname is the name of the attribute to be created (or an already existing attribute name in the case of update statements.) Both "=" and BY may be used interchangeably with IS.

or 2. attribute

For this element, in the case of a RETRIEVE, the resultant domain will have the same name as that of the attribute being retrieved. In the case of update statements, the relation being updated must have a domain with that name.

qualification

A qualification consists of any number of clauses connected by Boolean connectives "and" and "or". Currently, the qualification must be expressed in conjunctive normal form (i.e. a sequence of groups of or'ed together clauses with the groups and'ed together.)

A clause consists of:
1. Two a-fcns connected by a relational operator :
    a-fcn relop a-fcn
2. Three a-fcns connected by bounds operators :
    a-fcn bdop a-fcn bdop a-fcn
3. Either of (1) or (2) above preceded by the negation operator "not".

SEE ALSO

range, retrieve, append, delete, replace

NAME
range - declare a variable to range over a relation

SYNOPSIS
RANGE OF variable IS relname

DESCRIPTION
Range is used to declare variables which will be used in subsequent QUEL statements. The variable is associated with the relation specified by relname. When the variable is used in subsequent statements it will refer to a tuple in the named relation. A range declaration remains in effect for the entire INGRES session (until exit from INGRES) or until the variable is redeclared by a subsequent range statement.

EXAMPLE
```
/* Declare the tuple variable e to range over the relation emp */
   range of e is emp
```

SEE ALSO
quel

DIAGNOSTICS

BUGS
Currently only 10 variable declarations may be in effect at any time. After the 10th range statement, the oldest declared variable is replaced with the new declaration.

NAME
replace - replace values of domains in a relation

SYNOPSIS
REPLACE tuple_variable (target_list) [WHERE qual]

DESCRIPTION
Replace changes the values of the domains specified in the target_list for all tuples which satisfy the qualification. The tuple_variable must have been declared to range over the relation which is to be modified. Only domains which are to be modified need appear on the target_list. These domains must be specified as result_attnames in the target_list either explicitly or by default (see QUEL). .

Numeric domains may be replaced by values of any numeric type (with the exception noted below). Replacement values will be converted to the type of the result domain.

EXAMPLE
```
/* Give all employees who work for Smith a 10% raise */
   range of e is emp
   replace e(sal = 1.1*e.sal) where e.mgr = "Smith"
```

SEE ALSO
quel, range

DIAGNOSTICS
Use of a numeric type expression to replace a character type domain or vice versa will produce diagnostics.

BUGS
Functionality of updates is not checked.

If the result relation is an unsorted table (which all results are currently) then duplicate tuples caused by replace are NOT removed.

Conversion is NOT done when F4 or F8 replacement values replace an I4 result domain. An attempt to do such replacement will presently cause an error termination.

NAME

reserved - special names in INGRES

DESCRIPTION

The following names are reserved for use by INGRES and should not be used as the names of relations, variables or attributes:

all	and	append	atan	avg	by	copy
cos	count	create	delete	destroy	from	gamma
help	in	into	is	log	max	min
mod	modify	not	of	on	onto	or
print	rand	range	replace	retrieve		sin
sqrt	save	sum	sort	to	until	where

NAME

retrieve - retrieve tuples from a relation

SYNOPSIS

```
RETRIEVE [[INTO] relname] (target_list) [WHERE qual]
```

DESCRIPTION

Retrieve will get all tuples which satisfy the qualification and either display them on the terminal or store them in a new relation. If a relname is specified, the result of the query will be stored in a new relation with the indicated name. A relation with this name must not already exist. The current user will be the owner of the new relation. The relation will have domain names as specified in the target_list result_attnames. If no result relname is specified then the result of the query will be displayed on the terminal. If a result relation is specified then duplicate tuples are removed. However, if the result is displayed on the terminal, duplicates are not removed.

EXAMPLE

```
/* Find all employees who make more than their manager */
   range of e is emp
   range of m is emp
   retrieve (e.name) where e.mgr = m.name
            and e.sal > m.sal
```

SEE ALSO

quel, range

DIAGNOSTICS

BUGS

NAME

save - save a relation until a date.

SYNOPSIS

SAVE relname UNTIL month day year

DESCRIPTION

Save is used to keep relations beyond the default 7 day life span.

Only the owner of a relation can save that relation. There is an INGRES process which typically removes a relation one day after its expiration date has passed, regardless of its protection status.

EXAMPLE

```
/* Save the emp relation until the end of February 1987 */
   save emp until feb 28 1987
```

SEE ALSO

retrieve, create

DIAGNOSTICS

BUGS

NAME
    sort - sort a relation into ascending order

SYNOPSIS
    SORT relname

DESCRIPTION
    Sort is used by INGRES to order relations and remove duplicate tuples. It is currently available as a user command and will sort using the order of domains as they appeared in the create statement as the sorting order.

EXAMPLE
    sort emp

SEE ALSO
    retrieve, create

DIAGNOSTICS

BUGS
    Only the owner of the relation should be allowed to sort it.