NAME

v.db.rim - RIM data base management/query interface for GRASS vector

SYNOPSIS

v.db.rim *data_base* -- to run the command version **v.db.rim** -- to run the interactive version (see SECTION TWO)

OVERVIEW

v.db.rim allows you to create, manage and query information about labeled lines and areas from a number of different vector maps in a batch mode or interactively. Operations are done on a data base through a command language defined below or an interactive set of menus described in SECTION TWO.

This program, like **s.db.rim**, is actually a marriage of the GRASS environment and the programmer's interface library of the relational data base management program RIM distributed publically by the University of Washington Academic Computing Services. Your system must have a FORTRAN 77 compiler to use this program.

DESCRIPTION

The vector data bases are stored in a subdirectory named 'rim/vect' in the user's current mapset. Data bases in other mapsets, selectable through the GRASS **g.mapsets** command, can be accessed for 'read-only' retrieval of records. Each mapset may have many data bases. Each data base within a mapset must have a different name; user supplied names are limited to seven (7) characters in order to maintain compatibility with the standard version of RIM. As with other GRASS commands, mapsets are searched in the mapset SEARCH_PATH order when a data base needs to be opened.

Each vector data base is composed of multi-field records (rows or tuples, in DBMS jargon). Each field and its position in the record form is defined via input to the .make command when a data base is originally defined. It is possible to add new fields or change the length of existing fields after data has been loaded, however this is not straightforward; deleting of fields is also possible, but requires even more experience and knowledge. The user needs to carefully design the data base fields and form (layout) and check the operation with a few pieces of test data before loading data for a large number of records.

One significant difference between **v.db.rim** and **s.db.rim** is that v.db.rim needs access to the original vector maps that data were imported from to successfully complete some commands. To keep track of which records were imported from which vector maps a new "table" has been added that keeps information about the reference vector maps. This table is called "Referencemaps" in the RIM data base. The v.db.rim commands .read_vect, .map and .maps allow the user to view and update this table. The .vector_map command, which creates a new GRASS binary vector file, requires that the reference vector maps for the database be accessible. If a reference map is not accessible, any vectors represented by the data base records that were generated from that vector map will not be transferred to a new vector map. Reference vector maps may be in any mapset in the current SEARCH_PATH.

COMMANDS

[Note: For each of the "dot commands", i.e., .make, described below there is a menu choice to selected when running the interactive version. The interactive menus are described in the SECTION TWO of this document. Some display capabilities exist in the interactive version which are not directly implemented in the command version.]

The commands are given alphabetically here for easy reference. The .make command is required to create a data base. Abbreviations down to the string shown in ( ) are accepted; this is primarily for those giving **v.db.rim** commands from a terminal, but abbreviations may also be used in batch files.

Each command is introduced with an input record (line) which starts with a period and is followed by one of the words shown below; for some commands the command line also contains one or more required or optional parameters. Additional or optional input instructions/data for a command is supplied on successive lines; a .end line is needed by some commands to signify the end of these lines.

"Alphabetical Command Summary"

"*!command*"
This is the only **v.db.rim** command not starting with a period. "command" is a single shell command line (no more than 80 characters) which is executed by a "G_system()" call (see GRASS gis library). Many UNIX utilities (e.g., vi, ls, print) and most GRASS commands (e.g., d.rast, d.sites, g.list, g.region, r.mask) may be executed. It is permitted, and often useful, to change "region" and "MASK" within **v.db.rim.** Multiple commands may be separated by ";" in the standard UNIX way. Note that a "!cd directory; ls" will change to the specified directory and list files, but the effective working directory for **v.db.rim** will not be changed when the command terminates.

"*.add (.a)*"
Add a new vector record (row) to the open data base. Each line following contains a field name followed by spaces and/or tabs then the value or character string to store for that field. Field information lines end with .end. Some fields may be absent and fields may appear in any order. The input of data is checked for one required field (sequence number), for field length for text type fields, and for duplicate sequence numbers. If split text fields are used in the data base layout (see .make), text data for each split field must be added as a separate line. If there are any problems, the record will not be stored and a message will be output. This format makes it relatively easy to import data from most other DBMS. The ".print -a" command, see below, outputs data in this list format.

```
Example:
 .add
 seq_num   204
 north     4690673.30
 east      601410.00
 map_num   1
 vect_type L
 reference Jones (1987)
 .end
```

"*.backup (.b) file_name*"
The .backup command is used to dump the entire data base from the RIM binary files to a text file format (see UNLOAD in the RIM User’s Manual). The file_name can be a relative path name

or full path name. The file will contain the data base definition, screen layout information, and tabular data. This text file is transportable to RIM or **v.db.rim** running on any other computer; it may also be reloaded to recreate the **v.db.rim** data base. A message will be output if there is any problem writing the .backup file. Backup can only be done on data bases in the user's current mapset.

To reload your data base from the backup file (normally not necessary):

```
GRASS 4> cd $LOCATION/rim/vect      #right directory
GRASS 4> rm db_name.rimdb*          #remove data base
GRASS 4> rim                        #run RIM manually
RIM> input "path/file"              #RIM rebuilds data base from data
                                     written by .backup
RIM> exit
```

"*.change (.c) [-l]*"
    Without the "-l" flag, each line following .change is in the same format as for the .add command. The sequence number field is required and the sequence number must match an existing site in the data base. Only those fields for which lines are provided are changed in the record. After the .end the changed record is stored, if all is ok, otherwise a message is output.

If the "-l" flag (for "list") is given, the sequence number field is omitted and the specified field values are changed in all records currently selected by .find and/or .query.

"*.delete (.d)*"
    This command is used to delete data records. Deletion of records is permanent. Each line following the command should contain only a sequence number that you want to delete, with a .end line being last.

A backup of the data base or copies of the data base files are the ways to protect your valuable data. The following command sequence will delete all the records currently on v.db.rim's query list (the result of the last .query or .find command), after asking for approval.

```
 .delete
 .end
```

"*.end (.e)*"
    Ends multi-line input for several other commands.

"*.exit (.ex)*"
    Use .exit to end operation of **v.db.rim** cleanly. In general, do not use CTRL-C to exit unless absolutely necessary. When .exit is encountered in a batch file, input will revert back to the previous file, or the terminal, if any, which called the batch file.

"*.find (.f) [-m | -w] [-a | -d]*"
    The .find command is used to find the record(s) whose location (label point) is closest to a given point (the target). The target can be defined in one of several ways. The found records are stored on an internal query list for output by other commands; however, see note 2, below. Records are stored on the query list in order of proximity to the target location. The optional .find command line parameter specifies the current MASK (-m), if any, or the current region (-w), as a filter on the retrieved records; see notes 3 and 4, below. The append (-a) or delete (-d) options allow the "found" records to be added or deleted from the currently selected ones. When adding, duplicates will be discarded.

The single required line following the .find line gives the program the necessary target information. The following examples show the possibilities.

```
find> 602793.90 4379010.00
```

will find the one record nearest these coordinates and store it, append it or delete it on the internal query list.

```
find> 619840 4599000 10
```

will find the 10 records (or fewer, if there are not that many) closest to the given location.

```
find> record 132 10
```

will find the 10 records closest to the location (label point) for record 132 in the data base (including record 132). If record 132 does not exist, no action is taken.

```
find> distance from 472910.06 5732001.0 5000
```

will find all records within 5000 (meters in UTM coordinates) of the target location.

```
find> distance from record 16 -2500
```

will find all records greater than 2500 (meters) from the location of record 16.

Notes for .find:

1. All records found by each .find are stored on the query list in order of proximity to the target location (sorted by distance from target).

2. The number of records found is automatically printed to the active output device/file.

3. If mask is specified, the effective region is automatically set to the current region (because the GRASS mask is only defined for the current region).

4. Region and mask filtering uses the current resolution for the region to test if a point falls within a cell.

5. In the last two examples the string "distance from" must be exactly matched. Also, the word "record" must be exactly matched.

6. If the "distance from" radius is given as a negative value, points outside the target circle are selected; whereas, if a positive value is given, points inside the circle are selected.

7. The current region may be changed with !g.region or !d.zoom prior to doing a .find, and the mask may be set or removed with a variety of GRASS commands.

8. The "find>" prompt is given only when input is from a terminal.

9. The "distance" between the target location and a record for a line or an area is actually the distance between the target location and the representative point that is stored in the data base. This can lead to unexpected results when the representative point (label point) for a line or area is not

near the "center" of the feature.

"*.help (.h)*"

Prints a help screen to the output device or file. Useful to have when using **v.db.rim** from a terminal, or when writing a script file of commands.

"*.input (.i) [file]*"

The lines in the given file are read and processed as commands or data until an end of file is reached or until a .exit command is found. Input files may call other input files, by using this command, to a nesting depth of eight. Without a file name stdin is used as the input file.

"*.list (.l)*"

Lists the available data bases in the current mapset search path.

"*.make*"

Using the .make command you create a new data base in the current mapset by specifying the following items which define the screen (page) layout for displaying and printing the records, as well as the information fields:

```
1)  The fixed text part of the screen layout.
2)  The positions, types, and lengths of data fields.
```

Five fields must always exist in a data base; each of these field types may only occur once in a data base layout:

```
1) Type 's'  Sequence number field (a unique integer for each record).
2) Type 'x'  Easting coordinate of the representative point (a double float).
3) Type 'y'  Northing coordinate of the representative point (a double float).
4) Type 'v'  Vector type field (a text field).
5) Type 'm'  Reference Map field (an integer).
```

The other field types, which may occur in any combination and order, are:

```
6) type 'i'  An integer field.
7) type 'f'  A double precision float field.
             (always 2 decimal places used for output)
8) type 't'  A text field.
```

Each of the fields can be positioned anywhere within the screen layout, which has a limit of 19 lines by 80 columns. A maximum of 70 fields may be defined within this space. A field is specified in the screen layout by a tilde (~), a field type character, a field name and enough trailing tildes to fill out the desired field length.

Each line following the .make command is taken to define a line of the screen layout until a .end is reached. If a mistake is made on any of the input lines, the .make will fail. The .make information may be prepared in advance as a text file (this facilitates fixing mistakes) and the .input command can be used to read in this file. An example text file for a data base screen layout follows, with some explanatory notes and restrictions.

```
.make
              Hydrology Vector Database
              =========================
 Record #:    ~sSeqnum~       Feature Name:  ~tName~~~~~~~~~~~~
 Vector type: ~vVtype         Reference Map: ~mRefMap~
 North:       ~yNorth~~~~~    East:          ~xEast~~~~~~
```

```
 Updated:     ~tUpdate_Date~~~~~~~

 Comments:
              ~tComments.1~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
              ~tComments.2~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
              ~tComments.3~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
 .end
```

Notes:

1) Any text not preceded by a tilde (~) character is taken to be part of the constant or fixed text portion of the form.

2) A field definition begins with a tilde (~) character immediately followed by a single character which indicates the data type of the field (s,x,y,v,m,i,f or t). Immediately following the data type character is the field name of 1 to 16 characters. Field names can be composed of any characters from the following set: [A-Z,a-z,_,0-9]; the RIM program and library do not distinguish upper and lower case in field names, so you should avoid making names which differ only in case. Field names may not begin with a numeral [0-9]. The rest of the field length is padded with tilde (~) characters to the length desired.

3) The minimum field width is three characters; e.g., "~tA". Be sure field widths for all fields are wide enough for the values and strings you expect to store there; e.g., UTM northings require at least 11 spaces.

4) For text fields it is possible to continue a field across more than one line. This is done by appending a .1 to the field name forming first portion of this "split field", a .2 for the second portion, etc. This text field splitting affects how information is organized for input and output; the composite text string is concatenated (unused portions of fields are retained as spaces) and treated as a unit for storage and queries to the data base.

"*.map (.m) [map_id map_name] | [-d map_id]*"
Without arguments this command outputs a list of all the reference vector maps that are stored in the reference maps table. If a map number (map_id) and a vector map name (map_name) are given the vector map is found and added to the reference maps table in the data base. If the map number (id_num) is already in that table an error is issued and no action is taken.

Finally, to delete a map from the reference maps table, use the ’-d’ option followed by the map number (map_id). The map information for the given map number will be displayed and the user will be asked to confirm the deletion with a ’y’. Enter a ’n’ (for no) if you do not want to delete that reference map.

Remember, that if you delete a reference map for which there are still records in the data base, you cannot make a new vector map (using the .vector_map command) that includes those records unless you put that number and vector map name back in the reference map table.

"*.output (.o) [file or | process]*"
Causes all output (except some error messages) from **v.db.rim**, including that from the .print command, to go to the named path/file (may be a full or relative path name), or to be used as standard input by the process (a pipe). If no parameter is given, output returns to stdout, usually the user’s terminal. An example of the pipe usage would be

```
.output | grep "easting" | wc -l > /tmp/my_count
```

A pipe is closed whenever the .output command is given again, or on a .exit command.

"*.pack (.pa)*"

This should be used when numerous data records have been deleted or changed to recover disk space in the RIM binary data base files. It works by doing a .backup to a temporary file; moving the data base files to new names (*.bakdb*); running RIM to rebuild the data base; and, if the rebuilt data base can be opened and read, the temporary files are deleted. The user is informed if this process fails. Packing can only be done on an open data base located in the user's current mapset.

"*.print (.p) [-a | -l]* "

This command outputs the full record for the records currently stored on the internal query list (result of last .query or .find). Without the flag, the screen layout format is used. With the -l flag, for list format, the field name followed by the contents are output one field per line. The -a flag also outputs in the list format but with a .add line and a .end line surrounding each record printed; data files in this form can be read with .input, thus they form one kind of backup mechanism and can be used to transfer data (not the data base layout) from one GRASS system to another. The destination for the output is set by a previous .output command (default is stdout).

"*.query (.q) [-m | -w] [-a | -d]*"

The .query command is used to retrieve records via an SQL-like request to RIM, including a user specified "where clause." All fields for each record meeting the selection criteria are retrieved.

The optional .query command line parameters cause records whose representative points are not in the region (-w) and/or mask (-m) to be rejected, so these conditions need not be tested in the "where clause". See .find for a full explanation of the command line options.

After the query command line, any number of lines (each no more than 80 characters) may be entered to define the SQL "where" clause. A .end line is required to finish the request and begin data retrieval. See examples below.

The "distance from" clause may also be used as an additional selection criteria exactly as described in the examples and notes for .find. It must be entered as a separate line to the query prompt.

The retrieved records may be printed at time of retrieval, rather than after the completion of the query command by including a .print (.p) line with the same options for print format as in the .print command (see above); e.g. .p -a to output in the "list add" format. The .print clause must be entered as a separate line to the query prompt. This feature is most useful when working with very large data bases where retreval time is significant. See example 2 below.

Example 1

```
query> where density < 20 and (date = "10/14/89"
query> or county eq "San Marcos")
query> .end
```

Example 2

```
query> where east <600000 and name like "*Jones*"
query> distance from record 12 3000
query> .print -a
query> .end
```

Example 3

```
query>.end
```

The where and distance from clauses are each optional. If both are omitted, only the mask and region on the .query command line restrict the search; if mask and region are also omitted, all records will be retrieved (Example 3). When querying for records the where clause is processed first, the current region and mask tested (if requested), then the distance from clause is applied; a record must pass all tests to be put on the internal query list (or appended or deleted) for output by other commands.

Notes: (Also see Notes for .find)

1. The retrieved records are stored on the internal query list in the order returned from the data base by RIM, not necessarily in sequence number order or the order the data was loaded. A "distance from" clause results in a final sorting by proximity to the target.

2. See the RIM User's Manual and the s.db.rim manual page for additional information on the "where clause" in the "select" command, especially the quotes required for matching character string (text) fields, and the allowed comparison operators. (These are also described in SECTION TWO of this manual entry.)

3. In the example where clauses above, "density", "date", "county", east", and "name" are field names (column names in RIM) defined when the user initially makes the data base.

4. Each .query or .find resets the internal query list, unless the append or delete options are used. In no case is a record allowed to be duplicated on the query list.

"*.read_vect (.re) vector_map_name [attribute_field [text_field]]*"
This command will read an existing GRASS vector map and create a data base record for each labelled area, line, and point. The sequence number field will automatically be generated starting from one greater than the highest current number in the database. If the optional *attribute_field* is provided it must be an integer field and it will be filled with the area, line, or point attribute label. If the optional *text_field* is provided and a category description file (a dig_cats file) exists for the vector map, the category descriptions will be copied into the given text field.

Once the records have been loaded by .read_vect, use .change to add data to other fields for those records.

Note: Only **labeled** areas, lines and points are imported from the vector map.

"*.remove*"
This command, which requires a "y" as confirmation on the next line, entirely removes the three binary files which constitute your RIM data base. Use with care. Backup files must be removed individually by the user, if desired.

"*.show (.sh)*"
This command is used to output the screen or page layout as defined for the current data base. It serves as documentation of the data base definition and as a reminder for field names, types and lengths. By using an editor to surround the output of .show with .make and .end lines, it can be used to reload the data base definition with .input.

"*.site_list (.si) file_name [field_name]*"

This command writes the location coordinates (representative point) and a comment to the specified file in the site_list directory in the current mapset for each record currently selected. If the site file exists, the sites are appended to the current list, otherwise, a new site list file is created. A "field name" may be optionally specified; if so, the contents of that field (retrieved from the appropriate site record) are inserted as the comment (following a '#') in the site list; the record number is used if no field name is given. Such site lists may be used as input to s.db.rim.

A comment line is inserted in the site_list file with the current date and time and the name of the data base producing the site locations. The format used for each site is:

```
easting|northing|#number or comment
```

"*.tables (.t)*"

Prints the table structure of the currently opened RIM data base. This is the same output generated by a "list *" command when running RIM manually. The information for the table named "data" is useful for review of the user's field definitions, and the table named "Referencemaps" contains the reference map information. The information in the two other tables is for internal use by **v.db.rim.**

"*.vector_map (.v) file_name [attribute_field [text_field]]*"

This command creates a new binary vector map by copying the vectors associated with each record on the query list from the reference vector maps into the new vector map.

If the optional *attribute_field* parameter is provided, the areas and lines in the new vector map will be labeled from the given integer (i, m or s type) field value for each record. You may supply a fixed integer value (such as 1, 908, -7, etc.) instead of a field name; each line written to the new map will be given this constant attribute/label.

If the optional *text_field* is provided, it will be used to build a category description file (dig_cats file) for the vector map. Instead of a field name, a constant text string of up to 100 characters may be given, if enclosed in single or double quotes; this string is used as the category description for each line written to the new vector file.

The header of the new binary vector file will contain the current date and indicate that **v.db.rim** was used to create the vector map.

The topology information (the dig_plus file) is not automatically built for the new vector map and the user must run **support.vect** to do so before **v.digit** can be used to edit the vector map or some other programs can be used. The vector map can immediately be displayed, from within **v.db.rim** by issuing the following command (assuming you have a graphics monitor selected):

```
!d.vect file_name c=color
```

SECTION TWO -- THE INTERACTIVE VERSION MENUS AND COMMENTS

SYNOPSIS

DESCRIPTION

v.db.rim The interactive version allows you to create, manage and query information about vectors across the landscape on a data base through a series of menus (VASK screens) explained below.

THE MAIN MENU

Below is the main menu. Option 1 is the default. Note the status line at the top of the menu, and the fact that 8 records have been selected by the last find or query operation (between items 2 and 3). Note, also, that CTRL-C can be used to exit from this menu (and most other menus in the program) back to the GRASS prompt. The specifics of each menu choice are described below. Except for 6, and mouse options in 3 and 4, each choice has a direct counterpart in the command version.

```
      v.db.rim                MAIN  MENU                Version 1.4
        Data base <rivers> in mapset <kittco> open. 325 records.




       1  Open a data base
       2  List available vector data bases
  --------  Retrieve/Output Site Records (8 currently)  --
       3  Find records in proximity to a Target point
       4  Query to select records (SQL)
       5  Show selected records on Terminal
       6  Display maps/selected vectors on graphics terminal
       7  Output selected records to Printer or File
       8  Create vector/site maps from selected records
  ------------  Add/Edit Site Records  ----------
       9  View a single record
      10  Add a record
      11  Change a record
      12  Delete a single record or all selected records
  ------- Other functions -- Shell Command -- Exit ----------
      13  Make a new data base & Management Functions
      14  Execute a shell command
       0  Done -- Exit from v.db.rim
    AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
            (OR <Ctrl-C> TO EXIT THIS PROGRAM)
```

1. Open a data base. If a data base is already open, it is closed before the requested one is opened. Only data bases in the user's current mapset may be modified; others are opened in read-only mode; this will be indicated on line 2 of the menu.

2. List available data bases. For each mapset in the current GRASS mapset search path, the names of the existing vector data bases are listed.

3. "Find" records in the data base relative to a specified target location. This is used to select records based on proximity to the target and, optionally, records within the current region and, optionally, records falling in active cells within the current GRASS mask. The label point coordinates are used for these spatial tests. Two modes of targeting are provided: the N records

closest to the target, and all records within (or outside) a circle of specified radius from the target. The FIND/QUERY TARGET MENU discussed below accepts region/mask/target specifications from the user. The selected records are then displayed one at a time until CTRL-C is entered; then other operations, choices 5-8, can be done with these records. The line on the menu between 2 and 3 shows the number of records currently selected by choices 3 or 4.

4. "Query" records in the data base using an SQL-like "where clause," including specifications for region/mask/target (circle only) as in 3, above; see FIND/QUERY TARGET MENU section below. The where clause can test for ranges or matches for numeric data base fields, or matches on full strings or substrings for text fields. The selected records are then displayed one at a time until CTRL-C is entered; then other operations, choices 5-8, can be done with these records. This clause is entered on a QUERY COMMAND MENU described below.

The where clause may use parentheses ( ) to control the order of comparisons. Field names are not case sensitive within where clauses. The following comparison operators are valid for all types of fields:

```
eq   or   =           ne   or   <>
ge   or  >=           le   or   <=
gt   or   >           lt   or    <
```

String comparisons are case sensitive and are done character by character. Substrings comparisons may be done with the "like" operator as in:

```
where name like "*Jones*"
```

Note that the string being tested against the name field for each record is in quotes (single or double) and that wild card comparisons can be done in the standard way with '*' and '?' characters.

Logical comparisons may also be combined with those operators above. The permitted logical operators are:

```
and       or       not
```

The following complex example should be examined. The line breaks can occur between any tokens (words, values, operators), except within quoted strings.

```
where (name like "*Jones*" or name = "Smith")
and ( ( site < 300 and not (site = 251 or site eq 15) )
or east < 601000 )
```

5. This choice will display the records resulting from the last find/query one at a time on the terminal. Use ESC or enter a number to display another record and CTRL-C to end the display.

6. If a graphics monitor is active, the selected vectors will be displayed. The user may choose to erase the screen; display cell, vector, and/or site maps; and/or display the selected vectors from the data base; these maps are requested through the following interactive screen. Just enter ESC to skip this step. If no data base vectors are currently selected, that section of the menu will not appear; but the menu can still be used to display the other types of maps.

```
               SELECTION MENU FOR ITEMS TO DISPLAY

Enter cell and/or vector map names, if desired

 ______________  Cell file to display
```

```
  _______________  Vector file to display in color: _________
  _______________  Site list to display
                   Dpoints with: size=3_ type=box____ color=white____
                 _ Display currently selected vectors (enter x)
                   Dvect red______
                 _ Erase graphics screen (enter x)
                   Derase  black____

              AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
                           (OR <Ctrl-C> TO CANCEL)
```

7. This selection results in a screen prompting for the name of the file to output the selected records to, and for optional formatting selection. If the file name is lp, the site records are sent to the printer. The optional formatting choices are for export of data in list and add formats; see the .print description in SECTION ONE of this manual page for information and examples.

8. Using this choice you can create a new GRASS vector map consisting of the vectors for the currently selected records, and/or a site list consisting of the representative points (label points) for the currently selected records, in your current mapset. A short menu prompts for the map names and other information.

If a vector map name is given you can choose an integer field (i, s, or m types), or a fixed value, to write as the label value for each vector in the dig_att file for the new map. You may also specify a field name (any type), or a fixed text string in single or double quotes, to write as the category description in the dig_cats file for the new map.

If you give a site list name, you can specify the name of a field (or fixed text string in quotes) to be used for the "#comment" in the site list (the record number is the default field). The current date and time, and the names of the mapset and data base in use are entered as an information line in the site_list file. Note that you can create a new site list or append to an existing site list, or both.

A variety of cell maps can be produced from a **v.db.rim** data base by creating new vector files then using the *v.to.rast* program, and by writing site_lists with different fields as "comments" then converting the site_lists to cell maps with *s.menu.*

9. Choices 9-12 operate on only a single record and do not use or modify the internal list of records selected by find/query (choices 3 or 4). Choice 9 is the way to view a single record, selected by record number. After viewing, ESC will allow entry of another site number and CTRL-C will exit to the main menu.

10. Use this selection to add a new record to the data base. (A new record is one whose number does not currently exist in the open data base.) After making this selection, the data base layout will be displayed and you should enter the available information appropriate to each field; the only required entry is the site (record) number field. If values for numeric fields are not entered, zero values will be stored. Unused portions of text fields are stored as strings of spaces.

11. After making this selection and specifying the record number to change field information for, the data is entered as for choice 10, except that the record number cannot be changed. (The command version of the program has provision for making bulk changes after a find or query; see .change.)

12. To delete a single record, enter its number when requested. All records chosen by the last find/query operation may be deleted by entering "list" in place of the record number. BE CAREFUL with this, *deleted records are really gone.*

13. This choice starts a new menu with less commonly used functions. See MANAGEMENT MENU section below.

14. The program will prompt you for one-line Shell Commands until you enter just a <RETURN> to return to the main menu.

FIND/QUERY TARGET MENU

This is the screen to set up the region/mask/target information for the find choice (3) and the query choice (4), except that item B is omitted for choice (4). If a graphics monitor is not active, the "mouse" item is omitted from the menu; and, if a mask is not currently set, that line is omitted.

The choice to append or delete the selected records will only be given after a successful find or query has stored some records on the internal record list. When appending records, duplicates of those previously selected will be discorded--they will not be stored a second time. If neither append nor delete is selected, the find or query will begin a new internal record list and the previous contents will be lost.

The choices entered on this example screen will result in all the records within a 1500 (meters) radius of the target point (to be chosen with the mouse) being selected and stored on the internal record list by find or query. They are sorted and stored in order of proximity to the target. If a specific record is used as the target, it’s representative point (label point) is the target coordinates, and it is always placed first in the retrieved list. If a mouse is chosen to select the target point, a menu to display reference maps is presented, exactly as in choice (6), prior to actually activating the mouse.

```
       QUERY/FIND:  REGION/MASK/TARGET SELECTION MENU
  Data base <arch> (READONLY) in mapset <PERMANENT> open.  25 records.
    Mark requests with ’x’ and enter required values.

               Respect current region   x

               Respect current MASK     x
              (forces current region)

A.  Find all sites within (or outside) a circular target   x
            and give the radius (negative for outside)  1500.00_____
                       OR
B.  Find a number of sites nearest a point   _
         and the number of sites requested   ________

    After selecting A or B, complete one(!) of these:
        1. x to select target point with mouse    x
        2. Enter site number for target point     __________
        3. Target coordinates             east    0.00________
                                         north    0.00________

 Append(a) or Delete(d) to the current FIND/QUERY list  _
 Reset to default choices for this menu _

         AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
                       (OR <Ctrl-C> TO CANCEL)
```

QUERY COMMAND MENU

The following screen completes the information for a query (choice 4). It may be left blank if no "where clause" is required. After a successful query, the selected records are displayed one at a time by hiting ESC; CTRL-C will quit the display and return to the main menu where several choices of operation on the retrieved sites are offered. The SQL "sort by" clause may also be used after the where clause to control the order selected records are presented; however, if option A or B in the TARGET MENU has been selected it causes sorting by proximity to the target location which will override the order produced by the "sort by" clause.

```
        QUERY COMMAND CONSTRUCTION SCREEN
  Data base <wells> in mapset <grant> open.  25 records.
 The SQL select query will use the current region
 and a target clause of 'distance from 596463.15 4919041.88'

where date = 10/16/89 and ppm_Cr gt 10____________________
__________________________________________________________
__________________________________________________________
__________________________________________________________
__________________________________________________________
__________________________________________________________
__________________________________________________________
__________________________________________________________

(Enter .show on a line to review screen layout and field names.)

  AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
              (OR <Ctrl-C> TO CANCEL)
```

MANAGEMENT MENU

Choice 13 from the main menu presents this menu. Each item is discussed below.

```
    v.db.rim      DATA  BASE  MANAGEMENT  MENU
Data base <fires> in mapset <Yellow> open.  250 records.

  1   Make a New Data Base in Current Mapset
  2   List Available Data Bases
  3   Remove (PERMANENTLY) Data Base from Current Mapset
  4   Recover a Data Base from a RIM ASCII File
  5   Show Screen Layout of Current Data Base
  6   Backup (UNLOAD) Data Base to RIM ASCII Format File
  7   Pack the Current Data Base
  8   Read a vector map into the Current Data Base
  9   Execute a Bourne Shell Command Line

  0   Return to Main Menu

 0_ Your selection

 AFTER COMPLETING ALL ANSWERS, HIT <ESC> TO CONTINUE
             (OR <Ctrl-C> TO CANCEL)
```

1. Use this choice to create a new data base in the current GRASS mapset. See section below on MAKE A NEW DATA BASE.

2. List available data bases. Like 2 on main menu.

3. Delete an entire data base from the current mapset. The name of the data base and additional confirmation of the action are prompted for. **Be careful!**

4. Choice 6 allows backup of the definition and data parts of a data base to a transportable text file. To rebuild (or build for the first time) a **v.db.rim** data base from one of these text files do the following steps:

```
       # see if the rim directory exists.
    ls $LOCATION/rim/vect
       # if the directory was not found, make it.
    mkdir $LOCATION/rim
    mkdir $LOCATION/rim/vect
       # change directory to it.
    cd $LOCATION/rim/vect
       # have rim build and load the binary data base files.
    rim
    RIM> input '/path/to/your/textfile'
    RIM> exit
```

The data base is thus created in the current mapset. Several **v.db.rim** commands should be run to verify the integrity of the newly created data base.

5. This merely shows the screen layout of the currently open data base. It is a useful way to quickly see the layout and review the field names and types.

6. When backing up to a text file, the RIM UNLOAD command is run with the output directed to a file of the user's choice. See 4 above. It is wise to do this operation after extensive changes or additions of data records. The resulting text file can be written to tape for preservation, or shared with other GRASS systems, if desired. Data bases may also be backed up by copying the three binary files which comprise the data base to a different directory with the UNIX cp command.

7. After deleting and adding a large number of site records, some "wasted" disk space will be present in the binary data base files. This procedure will perform an unload and a reload automatically to recover this unusable disk space. If there is any problem reopening the data base after packing, the user is notified and can recover in various ways depending on the backups which have been done.

8. Data (records) may be loaded into a data base from an existing GRASS vector map. This procedure will prompt for the vector map name and then add a record to the currently open data base for each labeled(!) line in the vector field. The user may also enter the name of an integer field in which to store the label (from the dig_att file) for each vector, and a text field in which to store the descriptive text from the dig_cats file for each vector. The record number, vector type, map number and location coordinate fields (s,v,m,x and y types) are automatically loaded for each site record by this procedure; other fields may be later edited with the "change" function.

9. This choice is the same as choice 14 on the main menu.

MAKE A NEW DATA BASE

After entering the name of the new data base you wish to create (7 characters maximum), you then decide how to input the information required. This input may be from a text file, or may be entered directly using the editor of your choice; the former is recommended. See discussion in .make in SECTION ONE.

NOTES

1. A "date" type field should be added to future versions. This version only allows storing of dates as strings (unless the user codes them to integers), and thus only string type searches can be made for dates.

SEE ALSO

s.db.rim

The RIM Users manual by Jim Fox, Academic Computing Services, Univ. of Washington. See especially Appendix B on redistribution of RIM.
The RIM Installers manual.

AUTHORS

David Satnik and James Hinthorne, GIS Laboratory, Central Washington University.